Istraživanje podataka 1

prof. dr Jelena Graovac
prof. dr Jovana Kovačević

Radna verzija

7. maj 2026.

Sadržaj

Glava 1

Uvod u istraživanje podataka

1.1 Šta je istraživanje podataka

Istraživanje podataka (engl. *Data Mining*) predstavlja proces analize velikih skupova podataka sa ciljem pronalaženja korisnih informacija i obrazaca koji nisu odmah uočljivi. U literaturi se takođe navodi da je to proces automatskog otkrivanja korisnih informacija iz velikih repozitorijuma podataka. Osnovna ideja je da se u velikim količinama podataka pronađu pravilnosti, veze i zakonitosti koje mogu biti korisne za razumevanje posmatranog sistema, donošenje odluka i predviđanje budućih ishoda.

U savremenim informacionim sistemima gotovo svaka aktivnost generiše određenu količinu podataka. Poslovni sistemi, internet servisi, senzorske mreže, mobilni uređaji i društvene mreže svakodnevno proizvode ogromne količine digitalnih informacija. Ovaj nagli rast podataka doveo je do potrebe za metodama koje omogućavaju efikasno izdvajanje korisnih saznanja iz takvih skupova podataka.

Ne postoji jedinstvena definicija istraživanja podataka. Prema jednom širem shvatanju, ono obuhvata:

- prikupljanje podataka,
- čišćenje podataka,
- obradu podataka,
- analizu i
- pronalaženje korisnih saznanja.

Često citirana definicija glasi:

> Netrivijalno izdvajanje implicitnih, prethodno nepoznatih i potencijalno korisnih informacija iz baza podataka.

Važno je naglasiti da istraživanje podataka nije isto što i obično pretraživanje baze podataka. Na primer, upit kojim tražimo sve kupce iz određenog grada nije istraživanje podataka. Takav zadatak se rešava standardnim mehanizmima sistema za upravljanje bazama podataka. Sa druge strane, otkrivanje obrasca da određene grupe kupaca često kupuju određene proizvode zajedno predstavlja tipičan zadatak istraživanja podataka.

Drugim rečima, dok klasični upiti nad bazama podataka omogućavaju pronalaženje već poznatih informacija, istraživanje podataka ima za cilj da otkrije nove obrasce i zakonitosti koji ranije nisu bili očigledni.

1.2 Zašto je istraživanje podataka važno

Potreba za istraživanjem podataka javlja se zbog velikih količina podataka koje se neprekidno prikupljaju u različitim oblastima, kao što su nauka, inženjerstvo, medicina i razne druge poslovne aplikacije.

Brz razvoj tehnologija za prikupljanje i skladištenje podataka doveo je do naglog rasta količine dostupnih podataka. Podaci se danas lako generišu, čuvaju i razmenjuju, pa se često govori o eri velikih podataka. U takvom okruženju sve je važnije iz velikih skupova podataka izdvojiti informacije koje mogu da se iskoriste u praksi.

Na primer, veb aplikacije beleže aktivnosti korisnika kroz pristupne logove, bankarski sistemi čuvaju zapise o finansijskim transakcijama, dok senzorske mreže kontinuirano prikupljaju podatke o različitim fizičkim procesima (mogu da mere temperaturu, pritisak, kretanje, vlažnost, buku, vibracije itd.). Takvi podaci mogu sadržati značajne informacije o ponašanju korisnika, funkcionisanju sistema ili promenama u okruženju.

Razvoj oblasti istraživanja podataka podstaknut je sledećim razlozima:

- stalno prikupljanje velikih količina podataka,
- razvoj snažnijih računara,
- potreba za konciznim i korisnim informacijama,
- konkurentska prednost u poslovnim aplikacijama.

Tradicionalne metode analize često nisu dovoljne, naročito kada su podaci:

- veoma obimni,
- prostorno-vremenske prirode,
- složeni i heterogeni,
- takvi da sadrže skrivene informacije koje nije lako uočiti.

Drugim rečima, količina podataka, njihova raznovrsnost i brzina prikupljanja postali su preveliki da bi ih čovek mogao efikasno analizirati bez pomoći automatizovanih metoda. Zbog toga su potrebni algoritmi i sistemi koji mogu automatski da izdvoje korisne obrasce i saznanja.

Savremeni skupovi podataka često imaju veličinu od više terabajta ili petabajta, što dodatno naglašava potrebu za skalabilnim algoritmima i efikasnim metodama analize.

1.3 Primeri primene

Značaj istraživanja podataka najbolje se vidi kroz primene u različitim oblastima.

U poslovanju i industriji, podaci sa kasa, evidencije sa veb sajtova i podaci iz korisničke podrške mogu da pomognu kompanijama da bolje razumeju potrebe svojih korisnika. Na osnovu takvih podataka mogu se podržati zadaci kao što su profilisanje kupaca, ciljano oglašavanje, otkrivanje prevara i donošenje odluka o prodaji i nabavci.

Prilikom korišćenja Interneta korisnici svakodnevno generišu velike količine podataka o pregledanju sadržaja, porukama i aktivnostima na društvenim mrežama. Takvi podaci mogu se koristiti za preporuku proizvoda, filtriranje neželjenih poruka, obradu upita u pretraživačima i predlaganje novih sadržaja ili veza među korisnicima.

U medicini, nauci i inženjerstvu istraživači raspolažu sve većim količinama podataka. Na primer, satelitska osmatranja Zemlje generišu podatke o kopnu, okeanima i atmosferi, dok savremene biološke tehnologije omogućavaju praćenje velikog broja gena u jednom eksperimentu. U medicini su sve značajniji elektronski zdravstveni kartoni, kao i složeni medicinski podaci, poput elektrokardiograma i snimaka dobijenih magnetnom rezonancom.

Primer. U prodavnici je moguće analizirati transakcije kupaca i uočiti da se neke grupe proizvoda često kupuju zajedno. Takvo saznanje može pomoći pri organizaciji prodajnog prostora ili pripremi aktivnosti marketinških akcija.

Primer. U medicini se podaci o pacijentima mogu koristiti za uočavanje obrazaca koji ukazuju na bolest ili na povećan rizik, čime se podržava donošenje medicinskih odluka.

1.4 Istraživanje podataka i KDD proces

Istraživanje podataka predstavlja deo šireg procesa otkrivanja znanja iz baza podataka, poznatog kao **Knowledge Discovery in Databases (KDD)**. KDD je celokupan proces pretvaranja sirovih podataka u korisne informacije.

Taj proces obično obuhvata više koraka:

- preprocesiranje podataka,
- samo istraživanje podataka,
- naknadnu obradu i tumačenje rezultata.

Ovaj proces se često opisuje kao niz povezanih faza koje čine svojevrsni procesni lanac obrade podataka, počev od prikupljanja i pripreme podataka, pa sve do interpretacije rezultata i njihove primene u realnim sistemima.

U fazi preprocesiranja sirovi podaci se pripremaju za analizu. To može da uključi:

- objedinjavanje podataka iz više izvora,
- čišćenje podataka radi uklanjanja šuma i duplikata,
- izbor relevantnih zapisa i atributa,
- normalizaciju,
- redukciju dimenzionalnosti.

Preprocesiranje je često najzahtevniji i najdugotrajniji deo celog procesa, jer se podaci u praksi prikupljaju i čuvaju na različite načine.

U mnogim praktičnim projektima upravo faza pripreme podataka zauzima najveći deo vremena, jer je potrebno transformisati sirove i često heterogene podatke u oblik pogodan za analizu.

Nakon toga sledi faza istraživanja podataka, u kojoj se primenjuju algoritmi radi otkrivanja obrazaca, modela i zakonitosti. Na kraju se vrši naknadna obrada rezultata, koja može uključiti:

- filtriranje obrazaca,
- vizuelizaciju,

- interpretaciju rezultata,
- proveru njihove korisnosti i ispravnosti.

Postupak se u praksi često povezuje sa donošenjem odluka. Na primer, rezultati dobijeni analizom podataka mogu se uključiti u sisteme za podršku odlučivanju kako bi se unapredile poslovne ili stručne odluke.

1.5 Glavni izazovi

Razvoj istraživanja podataka motivisan je nizom izazova koje tradicionalne metode analize ne mogu lako da prevaziđu.

1.5.1 Skalabilnost

Savremeni skupovi podataka mogu imati veličinu od terabajta, petabajta ili čak egzabajta. Algoritmi za istraživanje podataka zato moraju biti skalabilni, odnosno sposobni da rade efikasno i kada broj zapisa postane veoma veliki. To često zahteva posebne strategije pretrage, efikasne strukture podataka, kao i paralelne i distribuirane pristupe.

1.5.2 Visoka dimenzionalnost

Danas su česti skupovi podataka sa stotinama ili hiljadama atributa. Takvi podaci se javljaju, na primer, u bioinformatici, u prostorno-vremenskim merenjima i u mnogim drugim oblastima. Metode razvijene za podatke male dimenzionalnosti često ne daju dobre rezultate u takvom okruženju. Pored toga, računska složenost mnogih algoritama naglo raste sa porastom broja atributa.

1.5.3 Heterogeni i složeni podaci

Podaci više nisu samo numerički ili kategorički. U praksi se sreću tekst, slike, audio, video, sekvence, grafovi i prostorno-vremenski podaci. Zbog toga su potrebne metode koje mogu da uzmu u obzir različite tipove atributa i složene odnose među podacima.

1.5.4 Raspodela vlasništva i lokacije podataka

Podaci koji su potrebni za analizu često nisu smešteni na jednom mestu, niti pripadaju jednoj organizaciji. Mogu biti geografski raspodeljeni i čuvani kod različitih vlasnika. To nameće potrebu za distribuiranim metodama, pri čemu su važna pitanja količine komunikacije, objedinjavanja rezultata, bezbednosti i privatnosti.

1.5.5 Netradicionalna analiza

Tradicionalni statistički pristup često polazi od unapred postavljene hipoteze koja se zatim proverava na prikupljenim podacima. U savremenim zadacima analize često je potrebno generisati i proceniti veliki broj mogućih hipoteza, pa se razvijaju metode koje delimično automatizuju ovaj proces. Dodatni izazov je to što podaci u istraživanju podataka često nisu dobijeni pažljivo planiranim eksperimentom, već predstavljaju podatke prikupljene u realnim uslovima.

1.6 Poreklo i razvoj oblasti

Istraživanje podataka nastalo je kao spoj više disciplina. Najvažniji uticaji dolaze iz:

- statistike,
- veštačke inteligencije,
- mašinskog učenja,
- prepoznavanja obrazaca,
- baza podataka,
- paralelnog i distribuiranog računarstva.

Iz statistike potiču ideje kao što su uzorkovanje, procena i testiranje hipoteza. Iz veštačke inteligencije, mašinskog učenja i prepoznavanja obrazaca preuzeti su algoritmi pretrage, modelovanja i učenja. Baze podataka obezbeđuju efikasno čuvanje, indeksiranje i obradu upita, dok paralelno i distribuirano računarstvo omogućavaju rad sa veoma velikim skupovima podataka.

Istraživanje podataka je zato izrazito interdisciplinarna oblast. Za uspešnu primenu potrebna su znanja iz baza podataka, statistike, mašinskog učenja, veštačke inteligencije i programiranja.

1.7 Osnovni zadaci istraživanja podataka

Zadaci istraživanja podataka obično se dele na dve velike grupe:

- prediktivne zadatke,
- deskriptivne zadatke.

Kod prediktivnih zadataka cilj je da se predvidi vrednost nekog atributa na osnovu drugih atributa. Kod deskriptivnih zadataka cilj je da se otkriju obrasci koji opisuju odnose u podacima, kao što su korelacije, trendovi, klasteri i anomalije.

U nastavku su navedena četiri važna zadatka koja se često izdvajaju kao osnovna.

1.7.1 Klasifikacija i regresija

Prediktivno modelovanje podrazumeva izgradnju modela kojim se ciljna promenljiva izražava preko ulaznih promenljivih, odnosno atributa. Ako ciljna promenljiva uzima diskretne vrednosti, govorimo o klasifikaciji. Ako je ciljna promenljiva neprekidna, radi se o regresiji.

Primer. Ako želimo da predvidimo da li će korisnik internet prodavnice obaviti kupovinu ili ne, to je klasifikacija, jer cilj ima ograničen broj kategorija. Ako želimo da predvidimo buduću cenu akcije, to je regresija, jer je cena neprekidna vrednost.

Primer cveta Iris. Posmatrajmo zadatak određivanja vrste cveta Iris na osnovu dužine i širine latice. Na osnovu tih atributa mogu se formulisati jednostavna pravila, na primer:

- mala širina i mala dužina latice ukazuju na vrstu Setosa,

- srednja širina i srednja dužina latice ukazuju na vrstu Versicolour,
- velika širina i velika dužina latice ukazuju na vrstu Virginica.

Ova pravila ne objašnjavaju savršeno sve primere, ali dobro ilustruju ideju klasifikacije.

1.7.2 Pravila pridruživanja

Analiza pridruživanja služi za otkrivanje obrazaca koji opisuju snažno povezane osobine ili događaje u podacima. Takvi obrasci se često predstavljaju u obliku implikacionih pravila.

Primer. U analizi potrošačke korpe može se otkriti pravilo da kupci koji kupuju pelene često kupuju i mleko. Takvo pravilo može biti korisno za unakrsnu prodaju ili za bolji raspored proizvoda u prodavnici.

1.7.3 Klaster analiza

Klaster analiza ima za cilj da pronađe grupe međusobno sličnih objekata, tako da su objekti unutar iste grupe sličniji jedni drugima nego objektima iz drugih grupa.

Primer. Ako posmatramo skup novinskih članaka predstavljenih rečima koje se u njima pojavljuju, članci o ekonomiji mogu činiti jedan klaster, a članci o zdravstvu drugi. Dobar algoritam treba da izdvoji takve grupe na osnovu sličnosti sadržaja.

1.7.4 Otkrivanje anomalija

Otkrivanje anomalija bavi se pronalaženjem objekata koji se značajno razlikuju od ostatka podataka. Takvi objekti nazivaju se anomalije ili odudarajući podaci.

Primer. U sistemu za praćenje transakcija kreditnih kartica može se formirati profil uobičajenog ponašanja korisnika. Ako nova transakcija značajno odstupa od tog profila, ona može biti označena kao sumnjiva i ukazivati na potencijalnu prevaru.

1.8 Zadaci istraživanja podataka

Istraživanje podataka je zasnovano na algoritmima. Svaki algoritam nastoji da:

- pronađe model koji najbolje opisuje podatke,
- uklopi podatke u odgovarajući matematički model.

U praksi ne postoji jedan univerzalni algoritam koji je najbolji za sve probleme. Izbor pristupa zavisi od vrste podataka, cilja analize i ograničenja računarskih resursa. Zbog toga je važno razumeti i prirodu podataka i osobine metoda koje se koriste.

1.9 Srodne discipline

Sa istraživanjem podataka usko su povezane i sledeće oblasti:

- Big Data,
- Predictive Analytics,

- Data Science.

Data Science je interdisciplinarna oblast koja proučava i primenjuje alate i tehnike za dobijanje korisnih uvida iz podataka. Ona obuhvata tehnike iz istraživanja podataka, statistike, veštačke inteligencije, mašinskog učenja, baza podataka, kao i distribuiranog i paralelnog računarstva. Pojava ove oblasti ukazuje na to da složeni savremeni problemi često zahtevaju kombinovanje znanja i metoda iz više oblasti.

1.10 Zaključak

Istraživanje podataka nastalo je kao odgovor na potrebu da se iz velikih, složenih i brzo rastućih skupova podataka izdvoje korisne informacije. Ono predstavlja važan deo procesa otkrivanja znanja iz podataka i oslanja se na algoritme, statistiku, mašinsko učenje, baze podataka i druge discipline. Njegov značaj se ogleda u širokom spektru primena, od poslovanja i internet servisa do medicine, biologije, nauke i inženjerstva.

Zbog toga je razumevanje osnovnih pojmova, procesa i zadataka istraživanja podataka neophodno za dalje proučavanje ove oblasti.

Glava 2

Podaci

2.1 Uvod u pojam podataka

Istraživanje podataka polazi od podataka, pa je zato prvi korak u učenju ove oblasti razumevanje toga šta su podaci, kako se opisuju i na koji način njihov oblik utiče na izbor metoda analize. U najopštijem smislu, podaci predstavljaju skup objekata i njihovih atributa. Objekat može biti student, pacijent, kupac, dokument, merenje ili događaj, a atribut predstavlja neku njegovu karakteristiku. U literaturi se za objekat koriste i nazivi kao što su *slog*, *instanca* i *entitet*, ali i *record*, *sample*, *observation* ili *case*. Suština je ista: objekat je jedinica koju posmatramo, a atributi su osobine pomoću kojih je opisujemo.

Podaci koji se koriste u istraživanju podataka mogu biti veoma različiti po strukturi, složenosti i značenju. Jedna od najvažnijih početnih podela jeste podela na *međusobno nezavisne podatke* i *međusobno zavisne podatke*. Kod prve grupe, pojedinačni objekti se u velikoj meri mogu posmatrati kao međusobno nezavisni, pa se svaki objekat opisuje skupom atributa bez potrebe da se odmah modeluju odnosi sa drugim objektima. Tipični primeri su tabele sa demografskim, finansijskim ili administrativnim podacima. Kod druge grupe između objekata ili između njihovih vrednosti postoje implicitne ili eksplicitne veze koje se ne smeju zanemariti. Takvi su, na primer, vremenske serije, prostorni podaci, prostorno-vremenski podaci, sekvence, grafovi i mreže.

Ova podela je važna zato što značajno utiče na izbor metoda analize. Kod jednostavnijih, međusobno nezavisnih podataka često je dovoljno posmatrati svaki red kao poseban objekat. Nasuprot tome, kod međusobno zavisnih podataka analiza mora da uzme u obzir i kontekst: vreme, lokaciju, redosled ili mrežnu povezanost. Upravo zbog toga isto pravilo analize ne može jednako uspešno da se primeni na tabelu studentskih podataka, EKG signal i društvenu mrežu.

2.2 Osnovni pojmovi

2.2.1 Podaci, atributi i vrednosti atributa

Atribut je karakteristika objekta koja može da se razlikuje od objekta do objekta ili da se menja kroz vreme. Na primer, temperatura, boja automobila i veličina ekrana jesu atributi. Da bi atribut mogao da se analizira, njemu se dodeljuje vrednost. Vrednosti atributa su brojevi ili simboli koji opisuju atribut. Na primer, dužina može biti izražena u metrima ili kilometrima; boja automobila može biti opisana simboličkim vrednostima kao što su *crvena*, *plava* ili *crna*; temperatura se može izraziti numerički.

Važno je naglasiti da atribut nije isto što i način njegovog zapisivanja. Dva atributa mogu biti predstavljena brojevima, a da samo nad jednim od njih imaju smisla standardne numeričke operacije. Na primer, starost zaposlenog i identifikacioni broj zaposlenog mogu oba biti zapisani kao celi brojevi, ali prosečna starost ima smisla, dok prosečan identifikacioni broj nema. Zato je pri analizi mnogo važnije razumeti značenje atributa nego samo format u kome je zapisan.

U praksi se za atribut koriste i nazivi *obeležje*, *dimenzija* i *feature*, dok se za objekat često koriste termini *instanca*, *uzorak*, *entitet*, *record* ili *observation*. Iako terminologija može da varira od oblasti do oblasti, osnovna ideja ostaje ista: objekat je jedinica posmatranja, a atributi predstavljaju osobine na osnovu kojih tu jedinicu opisujemo. Kod tabelarnih podataka svaki red obično predstavlja jedan objekat, a svaka kolona jedan atribut. Na primer, u tabeli studentskih podataka jedan red može predstavljati jednog studenta, dok kolone mogu biti broj indeksa, godina studija, prosek, broj položenih ispita i smer. Takav prikaz je veoma prirodan i zato se često koristi kao polazna tačka za objašnjavanje osnovnih pojmova istraživanja podataka.

2.2.2 Merenje i skale merenja

Da bi atribut dobio vrednost, koristi se skala merenja, odnosno pravilo kojim se atributu objekta pridružuje broj ili simbol. Isto svojstvo se ponekad može meriti na različite načine. Na primer, dužina se može izražavati u metrima ili stopama, a temperatura u Kelvinima, Celzijusima ili Farenhajtima. Pri tome značenje atributa ostaje isto, ali se menjaju formalne osobine njegovih vrednosti. Zbog toga je uvek važno znati ne samo koju vrednost atribut ima, već i na kojoj skali je izmerena.

2.3 Tipovi atributa i skupova podataka

2.3.1 Tipovi atributa

Diskretni i kontinualni atributi

Jedna podela atributa zasniva se na broju mogućih vrednosti.

Diskretni atributi. Diskretni atributi imaju konačan ili prebrojiv skup vrednosti. Primeri su poštanski broj, broj računa i skup reči u dokumentu. Diskretni atribut može biti i numerički i kategorički. Na primer, broj položenih ispita je diskretan numerički atribut, dok poštanski broj, iako je često zapisan ciframa, u suštini služi kao oznaka. Poseban slučaj diskretnih atributa jesu binarni atributi, koji imaju samo dve vrednosti, na primer *da/ne*, *0/1*, *prisutan/odsutan*. Binarni atributi su naročito važni u analizi transakcija i pravilima pridruživanja.

Kontinualni atributi. Kontinualni atributi imaju realne vrednosti. Primeri su temperatura, visina, težina, pritisak i brzina. U teoriji, kontinualni atribut može imati beskonačno mnogo mogućih vrednosti iz nekog intervala realnih brojeva, mada se u praksi meri samo sa ograničenom preciznošću. Na primer, visina osobe može biti zabeležena kao 182.4 cm, a temperatura kao $23.7°C$.

U obradi podataka se ponekad kontinualni atribut transformiše u diskretne kategorije. Na primer, dužina se može preslikati u kategorije *kratko*, *srednje* i *dugo*. Takva transformacija se radi kada određena metoda bolje radi nad diskretnim vrednostima.

U literaturi se kontinualni atributi često nazivaju i *numeričkim*, *kvantitativnim* ili *merljivim* atributima, naročito kada imaju prirodno značenje u okviru statističke analize. Takvi atributi su posebno pogodni za računanje proseka, varijanse, korelacije i drugih numeričkih mera. Sa druge strane, diskretni atributi nisu uvek „nenumerički“. Na primer, broj položenih ispita jeste diskretan, ali je ipak numerički atribut jer ima prirodan poredak i smisao računanja razlika. Nasuprot tome, poštanski broj ili identifikacioni broj mogu biti zapisani ciframa, ali suštinski ne predstavljaju količinu. Zato se u istraživanju podataka uvek razlikuje *način zapisivanja* od *semantičkog značenja* atributa.

Tipovi atributa prema operacijama

Tip atributa zavisi od operacija koje se mogu smisleno primeniti nad njegovim vrednostima. U tom smislu razlikuju se imenski, redni, intervalni i razmerni atributi.

Vrsta operacije	Rbr	Operacija	Tip atributa
Različitost	1	$= \; i \; \neq$	Imenski (1)
Uređenje	2	$<, \; \leq, \; > \; i \; \geq$	Redni (1,2)
Aditivnost	3	$+ \; i \; -$	Intervalni (1,2,3)
Multiplikativnost	4	$\times \; i \; \div$	Razmerni (1,2,3,4)

Imenski atributi. Kod imenskih atributa vrednosti služe samo da razlikuju objekte. Nad njima ima smisla proveravati samo jednakost i nejednakost. Primeri su boja očiju, pol, poštanski broj i identifikacioni broj zaposlenog. Iako poštanski broj i identifikacioni broj mogu biti zapisani ciframa, oni ne predstavljaju količinu i ne treba ih tretirati kao numeričke veličine.

Redni atributi. Redni atributi omogućavaju poređenje i uređivanje vrednosti. Na primer, ocene *loše*, *dobro*, *vrlo dobro*, *odlično* imaju prirodan poredak, ali razlika između susednih kategorija nije nužno jednaka. Slično važi i za rangove ili nivoe kvaliteta.

Intervalni atributi. Kod intervalnih atributa razlike između vrednosti imaju značenje, odnosno jednake razlike između vrednosti mogu smisleno da se porede, ali nula nije apsolutna. Klasični primeri su temperatura u Celzijusovoj i Farenhajtovoj skali i kalendarski datumi. Na primer, razlika između $20°C$ i $30°C$ ima smisla, ali nije smisleno reći da je $20°C$ “dva puta toplije” od $10°C$, jer nula na toj skali nije fizičko odsustvo temperature.

Razmerni atributi. Kod razmernih atributa i razlike i količnici imaju značenje, jer postoji prirodna nula. Primeri su dužina, masa, starost, broj elemenata, novčani iznos i temperatura u Kelvinima. Na primer, dužina od 4 m jeste dvostruko veća od dužine od 2 m.

Ova podela je važna zato što određuje koje statističke i analitičke operacije imaju smisla. Ako se tip atributa ne razume pravilno, lako se dolazi do pogrešnih zaključaka.

2.3.2 Tipovi skupova podataka prema prirodi atributa

Kvantitativni, kategorički i mešoviti podaci

Sa stanovišta praktične analize veoma je korisno razlikovati *kvantitativne*, *kategoričke* i *mešovite* skupove podataka.

Kvantitativni podaci. Kvantitativni podaci su oni kod kojih su svi ili gotovo svi atributi numerički i imaju smisleno tumačenje kao količine. Ovakvi podaci su posebno pogodni za statističku obradu, jer se nad njima lako definišu srednje vrednosti, odstupanja, rastojanja i korelacije.

Kategorički podaci. Kategorički podaci sadrže atribute čije vrednosti predstavljaju oznake kategorija. Primeri su pol, boja, vrsta proizvoda, mesto stanovanja, zanimanje ili dijagnoza. Kod ovih podataka nije uvek moguće primenjivati standardne numeričke operacije, pa su potrebne posebne mere sličnosti i posebne tehnike kodiranja.

Mešoviti podaci. U realnim primenama veoma često se sreću mešoviti podaci, kod kojih su neki atributi numerički, a neki kategorički. Na primer, skup podataka o kupcima može sadržati godine starosti i godišnji prihod kao numeričke atribute, ali i pol, grad i status kupca kao kategoričke atribute. Takvi podaci su analitički zahtevniji, jer je potrebno uskladiti različite tipove atributa u zajedničkom postupku obrade.

Poseban slučaj: binarni atributi. Binarni atributi, koji imaju samo dve vrednosti, predstavljaju poseban i veoma važan slučaj. Po svom značenju oni su često kategorički atributi, ali se u praksi najčešće kodiraju vrednostima 0 i 1, pa ih mnoge metode mogu obrađivati i kao numeričke. Na primer, atribut *kupio/nije kupio* ili *prisutan/odsutan* često se zapisuje sa 0 i 1. Zbog toga binarni podaci često predstavljaju vezu između kategoričkih i numeričkih prikaza.

Do sada su razmatrani pre svega *tipovi atributa*, odnosno načini na koje se pojedinačne osobine objekata mogu opisivati i meriti. U nastavku se pažnja premešta na *celokupne skupove podataka*, jer izbor metode za analizu ne zavisi samo od tipa pojedinačnih atributa, već i od ukupne strukture podataka i odnosa među instancama.

2.4 Vrste skupova podataka

2.4.1 Međusobno nezavisni podaci

Kod međusobno nezavisnih podataka, nema zavisnosti ni između samih objekata (redova) niti između njihovih atributa (kolona). Primer predstavlja sledeći skup demografskih zapisa o pojedincima koji sadrže njihove godine, pol i poštanski broj.

Ovo se obično odnosi na jednostavne tipove podataka, kao što su višedimenzioni podaci ili tekstualni podaci. Ovi tipovi podataka su najjednostavniji i najčešće se susreću.

Višedimenzionalni podaci

Najjednostavniji i najčešći oblik podataka u istraživanju podataka jesu višedimenzionalni podaci. Oni se najčešće predstavljaju kao tabela sa n instanci i d atributa, pri čemu

Table 1.1: An example of a multidimensional data set

Name	Age	Gender	Race	ZIP code
John S.	45	M	African American	05139
Manyona L.	31	F	Native American	10598
Sayani A.	11	F	East Indian	10547
Jack M.	56	M	Caucasian	10562
Wei L.	63	M	Asian	90210

svaki red odgovara jednoj instanci, a svaka kolona jednom atributu. Kada su svi atributi numerički, ovakav skup podataka može se zapisati i kao matrica

$$D \in \mathbb{R}^{n \times d},$$

gde je n broj instanci, a d broj atributa.

Kada su atributi numerički ili na odgovarajući način kodirani, svaka instanca može se zapisati kao vektor

$$X_i = (x_i^1, x_i^2, \ldots, x_i^d).$$

$$\begin{matrix} x_1^1 & x_1^2 & x_1^3 & x_1^4 & \ldots & x_1^d \\ x_2^1 & x_2^2 & x_2^3 & x_2^4 & \ldots & x_2^d \\ x_3^1 & x_3^2 & x_3^3 & x_3^4 & \ldots & x_3^d \\ x_4^1 & x_4^2 & x_4^3 & x_4^4 & \ldots & x_4^d \\ x_5^1 & x_5^2 & x_5^3 & x_5^4 & \ldots & x_5^d \\ x_6^1 & x_6^2 & x_6^3 & x_6^4 & \ldots & x_6^d \\ \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\ x_n^1 & x_n^2 & x_n^3 & x_n^4 & \ldots & x_n^d \end{matrix}$$

Definicija 1. Skup višedimenzionalnih podataka $\mathcal{D}$ predstavlja skup od n slogova $\overline{X}_1, \ldots, \overline{X}_n$ takvih da svaki od slogova $\overline{X}_i$ predstavlja uređenu d-torku:

$$(x_i^1, \ldots, x_i^d).$$

Ovakav prikaz je prirodan kada svaka instanca ima isti skup atributa. Na primer, svaki student može biti opisan identifikacionim brojem, godinom studija i prosečnom ocenom; svaki pacijent nizom laboratorijskih merenja; svaki proizvod cenom, težinom i kategorijom. Kada su atributi numerički, instanca se može posmatrati kao tačka u d-dimenzionalnom prostoru. Upravo zato su matrica podataka i vektorski zapis standardni oblici predstavljanja podataka.

Broj atributa, odnosno dimenzionalnost, snažno utiče na analizu. Sa porastom broja atributa analiza često postaje teža, pa se u praksi često sprovodi smanjenje dimenzionalnosti ili izbor najvažnijih atributa.

Retki podaci

Kod retkih podataka većina vrednosti u matrici je nedostajuća ili je jednaka nuli. Takvi podaci često se javljaju u tekstualnim podacima, transakcijama i sistemima za preporuku.

Na primer, ako dokument predstavljamo skupom svih mogućih reči iz korpusa, u svakom konkretnom dokumentu većina tih reči imaće vrednost nula jer se ne pojavljuje. Slično tome, u sistemima za preporuku korisnik obično ocenjuje ili pregleda samo mali broj proizvoda ili filmova, pa je matrica korisnik–stavka veoma retka, jer je zabeležen samo mali broj interakcija.

Retkost u podacima je važna iz dva razloga. Prvo, utiče na izbor algoritama, jer neke metode dobro rade samo nad retkim podacima. Drugo, omogućava efikasnije čuvanje i obradu, jer je često dovoljno pamtiti samo nenulte vrednosti. U mnogim retkim skupovima podataka atributi su i asimetrični, što znači da je prisustvo neke osobine informativnije od njenog odsustva. Na primer, kod transakcija je značajnije da je proizvod kupljen nego da nije kupljen.

x_1^1		x_1^3	x_1^4	...	
x_2^1				...	
	x_3^2			...	x_3^d
			x_4^4	...	
x_5^1		x_5^3		...	
	x_6^2			...	
...	...	...	...	...	...
	x_n^2			...	x_n^d

2.4.2 Međusobno zavisni podaci

U mnogim skupovima podataka instance nisu nezavisne. Tada govorimo o međusobno zavisnim podacima. Primeri su vremenske serije, grafovi i mreže. Za razliku od obične matrice podataka, ovde između instanci postoji dodatna struktura ili odnos koji se ne sme zanemariti. Ako se ti odnosi ignorišu, analiza može biti nepotpuna ili pogrešna.

Jedan pristup analizi jeste da se najpre eksplicitno odrede odnosi između objekata, na primer sličnosti ili rastojanja između parova objekata, a zatim da se klasifikacija, klasterovanje ili otkrivanje anomalija zasniva upravo na tim odnosima. Izbor mere sličnosti ili rastojanja zavisi od tipa podataka i konkretne primene.

Indeks	Ime	Prezime	Datum upisa	Datum rođenja	Mesto rođenja
20140021	Miloš	Perić	06.07.2014	20.01.1995	Beograd
20140022	Marijana	Savković	05.07.2014	11.03.1995	Kraljevo
20130023	Sanja	Terzić	04.07.2013	09.11.1994	Beograd
20130024	Nikola	Vuković	04.07.2013	17.09.1994	
20140025	Marijana	Savković	06.07.2014	04.02.1995	Kraljevo
20140026	Zorica	Miladinović	06.07.2014	08.10.1995	Vranje
20130027	Milena	Stanković			

Zavisnosti među podacima mogu biti *implicitne* ili *eksplicitne*. Implicitne zavisnosti se ne zapisuju nužno direktno u podacima, ali su prirodno prisutne u datoj oblasti. Na primer, u prikazanoj tabeli postoji veza između indeksa i godine upisa: studenti sa indeksima koji počinju sa 2014 imaju godinu upisa 2014, dok studenti sa indeksima koji počinju sa 2013 imaju godinu upisa 2013. To znači da se godina upisa može zaključiti iz samog indeksa, iako ta veza nije posebno navedena kao pravilo. Eksplicitne zavisnosti su direktno

predstavljene, kao što su veze između korisnika u društvenoj mreži ili hiperlinkovi između veb stranica.

Prisustvo zavisnosti menja i značenje zakonitosti koji tražimo u podacima. Na primer, velika promena između dve susedne vremenske tačke može biti mnogo zanimljivija nego ista takva razlika između dve nepovezane instance u običnoj tabeli. Zato se kod međusobno zavisnih podataka metode istraživanja podataka moraju prilagoditi kontekstu u kome podaci nastaju.

Vremenske serije

Vremenske serije sadrže podatke prikupljene kroz vreme. Primeri su EKG signal, merenja različitim senzorima (na primer meteoroloških parametara kao što su temperatura ili atmosferski pritisak) i finansijski podaci, kao što su dnevne cene akcija. Vremenska serija dužine n i dimenzije d sadrži d numeričkih vrednosti za svaku vremensku tačku. U najjednostavnijem slučaju, za svaki trenutak postoji jedna vrednost, na primer temperatura u datom času, a u složenijim slučajevima za svaki trenutak može postojati više veličina.

Atributi se klasifikuju u dve vrste:

1. **Kontekstualni atributi**: To su atributi koji definišu kontekst na osnovu koga nastaju implicitne zavisnosti u podacima. Na primer, kod senzorskih podataka, vremenska oznaka u kojoj je merenje izvršeno može se smatrati kontekstualnim atributom. Ponekad se vremenska oznaka ne koristi eksplicitno, već se koristi indeks pozicije. Dok tip podataka vremenske serije sadrži samo jedan kontekstualni atribut, drugi tipovi podataka mogu imati više njih. Specifičan primer su prostorni podaci, o kojima će kasnije biti reči u ovom poglavlju.

2. **Bihevioralni atributi**: Oni predstavljaju vrednosti koje se mere u određenom kontekstu. U primeru sa senzorima, temperatura je bihevioralni atribut. Moguće je imati više bihevioralnih atributa. Na primer, ako više senzora beleži očitavanja u sinhronizovanim vremenskim trenucima, dobija se višedimenzioni skup podataka vremenskih serija.

Drugim rečima, kontekstualni atributi govore *kada* ili *u kom položaju* je podatak nastao, dok atributi ponašanja govore *šta je tada izmereno*. Upravo razdvajanje ove dve vrste atributa omogućava jasnije uočavanje zavisnosti u vremenski organizovanim podacima. Naime, vrednosti bihevioralnih atributa koje su bliske po vremenu često su i slične. Pored postepenih, glatkih promena, vremenske serije često pokazuju i *periodičnost, trend, sezonalnost* ili *nagle promene*. Upravo izdvajanje takvih zakonitosti predstavlja jedan od centralnih zadataka analize vremenskih serija.

Diskretne sekvence

Diskretne sekvence mogu se posmatrati kao kategorički analog vremenskih serija. I ovde postoji kontekstualni atribut, koji najčešće odgovara vremenskom trenutku ili poziciji u nizu, dok bihevioralni atribut predstavlja simbol, događaj ili kategoričku vrednost.

Na primer, razmotrimo log pristupa vebu, u kome se za 100 različitih pristupa beleže adresa veb stranice i IP adresa sa koje je zahtev upućen. Ovo predstavlja diskretan niz dužine $n = 100$ i dimenzionalnosti $d = 2$. Posebno čest slučaj kod diskretnih nizova je univarijantni scenario, u kome je vrednost $d = 1$. Takvi diskretni nizovi se takođe nazivaju

i niske (stringovi) i kao primer može da posluži naredna kolekcija nukleotidnih sekvenci (delovi DNK koji se mogu posmatrati kao niske nad azbukom $\{A, C, G, T\}$):

```
GGTTCCGCCCTTCAGCCCCGCGCC
CGCAGGGCCCCGCCCCGCGCGGTC
GAGAAGGGCCCCGCCTGGCGGGCG
GGGGGAGGCGGGGCCGCCCCGAGC
CCAACCGAGTCCGACCAGGTGCC
CCCTCTGCTCGGGCCTAGACCTGA
GCTCATTAGGCGGGCAGCGGACAG
GCCAAGTAGAACACGCGAAGCGC
TGGGCTGCCTGCTGCGACCAGGG
```

Definicija 3. Diskretna sekvenca dužine n i dimenzije d sadrži d osobina sa diskretnim vrednostima za svaku od n različitih vremenskih tačaka $t_1, t_2, \ldots, t_n$. Svaka od n komponenti $\overline{Y}_i$ sadrži atribute sa diskretnim karakteristikama (ponašanjem) koji su prikupljeni u i-toj vremenskoj tački.

Diskretne sekvence predstavljaju uređene nizove simbola, događaja ili drugih kategoričkih vrednosti. One se mogu posmatrati kao analog vremenskih serija u situacijama kada je redosled elemenata od suštinskog značaja. Umesto niza merenja, koje predstavljaju numeričke vrednosti, ovde imamo uređeni niz simbola ili događaja.

Ključna osobina sekvenci jeste redosled. Čak i kada se isti simboli pojavljuju, promena njihovog rasporeda može potpuno promeniti značenje. Zato metode za ovakve podatke moraju da vode računa o uređenosti elemenata.

Posebno važan slučaj jesu jednodimenzionalne sekvence, odnosno niske, koje se često nazivaju i *stringovima*. Primeri su DNK sekvence, proteinski lanci, nizovi klikova korisnika na veb sajtu i dnevnici događaja u informacionim sistemima. Na primer, niz uzastopnih neuspelih prijava na sistem može ukazivati na pokušaj neovlašćenog pristupa.

Za razliku od numeričkih vremenskih serija, kod diskretnih sekvenci ne postoji glatka numerička promena između susednih vrednosti. Zato su ovakvi podaci često zahtevniji za analizu i traže posebne metode koje uzimaju u obzir pojavljivanje podsekvenci, obrazaca i odstupanja u redosledu događaja.

Prostorni i prostorno-vremenski podaci

Definicija 4. Slog prostornih podataka sa d dimenzija sadrži d atributa koji prikazuju ponašanje i jedan ili više kontekstualnih podataka koji određuju lokaciju. Skup d dimenzionih prostornih podataka sadrži d dimenzionih slogova $\overline{X}_1, \ldots, \overline{X}_n$, i skup od n lokacija $L_1, \ldots, L_n$, takvih da je slog $\overline{X}_i$ pridružen lokaciji L_i.

Prostorni podaci sadrže informacije o lokaciji objekata. Obično uključuju koordinate i dodatne atribute koji opisuju objekat. Na primer, meteorološka stanica ima geografsku poziciju, ali i merene veličine kao što su temperatura, pritisak i količina padavina.

Prostorno-vremenski podaci kombinuju prostorne i vremenske atribute. Primer je praćenje kretanja objekata, kao što su vozila, brodovi ili mobilni uređaji. Takođe, meteorološki podaci su često prostorno-vremenski, jer se mere kroz vreme na više lokacija.

Kod prostornih podataka važna je prostorna autokorelacija: objekti koji su fizički blizu često imaju slične vrednosti i drugih atributa. Na primer, dve geografski bliske tačke često imaju sličnu temperaturu ili količinu padavina. Kao i kod vremenskih serija, i ovde zanemarivanje odnosa među lokacijama može dovesti do loših rezultata.

Kao i kod vremenskih serija, i kod prostornih podataka korisno je razlikovati *kontekstualne* i *bihevioralne* atribute. Kontekstualni atributi određuju lokaciju, na primer geografsku širinu i dužinu, dok bihevioralni atributi opisuju šta se na toj lokaciji meri, kao što su temperatura, vlažnost, koncentracija zagađenja ili gustina saobraćaja.

Kod prostorno-vremenskih podataka moguće su dve česte situacije. U prvoj su i prostor i vreme kontekstualni atributi, dok je merena veličina bihevioralni atribut; primer su temperature merene tokom vremena na više lokacija. U drugoj je vreme kontekst, a prostorne koordinate predstavljaju bihevioralne atribute. Tipičan primer su trajektorije kretanja vozila, brodova ili mobilnih uređaja.

Ovakvi podaci su važni zato što istovremeno sadrže i prostornu i vremensku strukturu. Zbog toga analiza mora da uzme u obzir i bliskost lokacija i bliskost trenutaka posmatranja, što ih čini složenijim od običnih tabelarnih podataka.

Grafovski podaci

Definicija 5. Mreža $G = (N, A)$ sadrži skup od N čvorova i skup grana A, pri čemu grane iz A predstavljaju relacije između čvorova. Skup atributa $\overline{X}_i$ može da bude pridružen čvoru i, ili skup atributa $\overline{Y}_{ij}$ može da bude pridružen grani (i, j).

Grafovski podaci su pogodni kada između objekata postoje eksplicitne veze. Graf se definiše kao

$$G = (N, A),$$

gde su N čvorovi, a A grane. Atributi mogu biti pridruženi čvorovima ili granama. Na primer, u društvenoj mreži čvorovi mogu predstavljati korisnike, a grane njihove veze ili interakcije. U mreži veb stranica, čvorovi su stranice, a grane hiperlinkovi između njih.

Grafovska reprezentacija je važna zato što sama struktura veza nosi informaciju. Nekada su objekti povezani grafom, a nekada su i sami objekti po prirodi grafovi. Na primer, hemijska jedinjenja mogu se predstaviti grafovima u kojima su čvorovi atomi, a grane hemijske veze.

Kod grafovskih podataka veze mogu biti usmerene ili neusmerene. Na primer, hiperlink sa jedne veb stranice na drugu prirodno se modeluje kao usmerena grana, dok se prijateljstvo na društvenoj mreži često modeluje kao neusmerena veza. Pored toga, atributi mogu biti pridruženi čvorovima, a ponekad i granama. Na primer, korisnik društvene mreže može imati atribute kao što su starost i interesovanja, dok veza između dva korisnika može imati atribut koji opisuje intenzitet komunikacije.

Važno je razlikovati dve situacije. U prvoj postoji jedan veliki graf, kao što je društvena mreža ili veb graf. U drugoj postoji baza mnogo manjih grafova, na primer hemijskih jedinjenja, gde svaki objekat sam po sebi predstavlja jedan graf. Ove dve situacije zahtevaju različite tehnike analize i često vode do različitih tipova problema.

2.5 Kvalitet podataka i razumevanje podataka

2.5.1 Kvalitet podataka i značaj upoznavanja podataka

Uspeh istraživanja podataka ne zavisi samo od izbora algoritma, već i od kvaliteta podataka. Podaci mogu sadržati šum, anomalije, nedostajuće podatke, nekonzistentne zapise, duplikate ili mogu biti pristrasni. Zbog toga je važno uzeti u obzir vrstu i kontekst podataka pre bilo kakve dalje analize.

Zato se u praksi obraća pažnja na:

- **kvalitet podataka** – prisustvo šuma, anomalija, nedostajućih, nekonzistentnih ili dupliranih vrednosti;
- **preprocesiranje** – transformacije koje podatke čine pogodnijim za analizu;
- **odnose među objektima** – sličnosti, rastojanja, vremenske i prostorne zavisnosti;
- **relevantnost i ažurnost** – da podaci zaista opisuju pojavu koja nas zanima i da nisu zastareli.

Posebno je važno naglasiti da kvalitet podataka nije samo tehničko pitanje, već i pitanje razumevanja domena. Isti zapis može imati potpuno drugačije značenje u zavisnosti od načina na koji je nastao. Na primer, identifikacioni broj može slučajno biti povezan sa ciljnom promenljivom, ali to ne znači da on nosi stvarnu uzročnu informaciju. Slično tome, nedostajuće vrednosti nekada znače da merenje nije izvršeno, a nekada da je vrednost van opsega ili da je došlo do greške pri unosu.

Zbog toga istraživanje podataka uvek podrazumeva i razumevanje porekla podataka, postupka merenja, načina kodiranja vrednosti i mogućih sistematskih grešaka. Tek kada se taj kontekst razjasni, može se očekivati da će algoritamski rezultat biti smislen, tumačiv i upotrebljiv.

2.6 Podaci u procesu istraživanja podataka

U procesu istraživanja podataka polazi se od *višedimenzionalne baze* D sa n slogova i d atributa. Takva baza se najčešće predstavlja u obliku *matrice podataka* dimenzije $n \times d$, pri čemu vrste matrice odgovaraju slogovima, odnosno instancama, a kolone atributima. Ovakvo predstavljanje je posebno pogodno zato što omogućava da se različiti zadaci istraživanja podataka posmatraju kao različiti načini analize iste matrice podataka.

Pre same analize obično je neophodna *prethodna priprema podataka*, odnosno preprocesiranje. U toj fazi podaci se pripremaju za dalju obradu, uklanjaju se očigledne nepravilnosti, vrši se izbor pogodnog oblika zapisa i po potrebi redukcija podataka. Redukcija može podrazumevati smanjenje broja atributa, sažimanje informacija ili izdvajanje samo relevantnog dela podataka, kako bi analiza bila efikasnija i preglednija.

Posmatrano iz ptičje perspektive, veliki broj zadataka istraživanja podataka može se razumeti kao traženje relacija u matrici podataka. Te relacije mogu biti:

- relacije između kolona,
- relacije između vrsta,
- relacije između grupa atributa u vrstama.

Relacije između kolona ispituju koje se vrednosti atributa često javljaju zajedno i kako pojedini atributi utiču na neki poseban, ciljni atribut. Ovakav pogled vodi ka pravilima pridruživanja i klasifikaciji. Relacije između vrsta ispituju koje su instance međusobno slične, a koje se izdvajaju od ostalih, što vodi ka klasterovanju i otkrivanju anomalija. Relacije između grupa atributa u vrstama omogućavaju uočavanje složenijih obrazaca unutar pojedinačnih instanci.

Ovakav pogled je koristan zato što pokazuje da osnovni problemi istraživanja podataka nisu nepovezani, već predstavljaju različite načine analize iste matrice podataka. Zbog toga se među najvažnije gradivne blokove istraživanja podataka ubrajaju:

- podaci,
- pravila pridruživanja,
- klasifikacija,
- klasterovanje,
- analiza i vizuelizacija rezultata.

Osnovne tehnike istraživanja podataka obuhvataju pravila pridruživanja, klasifikaciju, klasterovanje, kao i analizu i vizuelizaciju podataka. Ove tehnike rešavaju različite tipove problema i međusobno se dopunjuju. Pored toga, u okviru obrade podataka značajno mesto zauzima i *otkrivanje anomalija*, jer ono omogućava identifikaciju redova matrice koji se značajno razlikuju od ostatka podataka.

2.7 Osnovne tehnike istraživanja podataka

2.7.1 Pravila pridruživanja

Pravila pridruživanja predstavljaju jednu od osnovnih tehnika istraživanja podataka. Njihov cilj je da otkriju koje se stavke, osobine ili događaji često javljaju zajedno u istoj instanci. Ovaj zadatak se najčešće razmatra nad *retkom binarnom bazom*, odnosno nad matricom čiji su elementi 0/1, pri čemu je najveći broj elemenata jednak nuli. Takva matrica je naročito pogodna za prikaz transakcionih podataka.

Ako kolone predstavljaju stavke, a vrste transakcije, tada važi:

$$(i, j) = 1 \Rightarrow \text{transakcija } i \text{ sadrži stavku } j.$$

Drugim rečima, vrednost 1 označava prisustvo određene stavke u transakciji, dok vrednost 0 označava njeno odsustvo. Na primer, u analizi kupovina u prodavnici moguće je konstruisati tabelu u kojoj kolone odgovaraju proizvodima, a vrste pojedinačnim kupovinama.

Br. trans.	hleb	mleko	pelene	pivo	jaja	kola
1	1	1	0	0	0	0
2	1	0	1	1	1	0
3	0	1	1	1	0	1
4	1	1	1	1	0	0
5	1	1	1	0	0	1

U datoj binarnoj matrici D veličine $n \times d$ posmatraju se svi podskupovi kolona A takvi da sve vrednosti u tim kolonama u odgovarajućoj vrsti imaju vrednost 1. Pri tome se koriste sledeće oznake:

- A je skup stavki,
- $\#(A)$ označava broj pojavljivanja skupa stavki A,
- N predstavlja broj redova u kompletnom skupu,
- $A \Rightarrow B$ znači da je skup stavki B pridružen skupu A.

Za pravila pridruživanja posebno su važne dve mere: *podrška* i *pouzdanost*.
Neka su A i B dva skupa stavki. Podrška pravila pridruživanja $A \Rightarrow B$ definiše se kao

$$sup(A \Rightarrow B) = \frac{\#(A \cup B)}{N}.$$

Ova mera pokazuje u kom delu svih transakcija se zajedno pojavljuju A i B.
Pouzdanost pravila pridruživanja definiše se formulom

$$conf(A \Rightarrow B) = \frac{\#(A \cup B)}{\#(A)}.$$

Pouzdanost pokazuje kolika je verovatnoća da se pojavi B kada se pojavi A.
Pored podrške i pouzdanosti, često se koristi i mera *lift*, definisana kao

$$lift(A \Rightarrow B) = \frac{conf(A \Rightarrow B)}{sup(B)}.$$

Lift pokazuje koliko je zajedničko pojavljivanje skupova A i B jače od onoga što bi se očekivalo kada bi oni bili statistički nezavisni. Vrednost lifta veća od 1 ukazuje na pozitivnu povezanost, dok vrednost bliska 1 ukazuje na slabu ili odsutnu povezanost.

Zadatak je, dakle, odrediti pravila pridruživanja (*association rules*) koja povezuju atribute u istoj instanci. Interesantna su ona pravila koja imaju dovoljno visok nivo podrške i pouzdanosti. Za nalaženje interesantnih pravila često se ne koriste samo apsolutne frekvencije, već i druge statističke mere, na primer χ^2 mera. Takođe, iako se pravila pridruživanja najčešće uvode na binarnim podacima, elementi matrice ne moraju biti isključivo binarne vrednosti; odgovarajućim transformacijama ovaj pristup se može proširiti i na druge tipove podataka.

Može se reći da pravila pridruživanja ispituju *horizontalne obrasce* u matrici podataka, odnosno koje kolone često imaju vrednost 1 u istim redovima. Zbog toga su naročito pogodna za retke binarne podatke, kao što su transakcione baze. Iako se najčešće uvode na binarnim podacima, pravila pridruživanja mogu se proširiti i na numeričke ili mešovite podatke odgovarajućim transformacijama.

Razmotrimo sada prethodnu tabelu. Važi:

$$\#(\{mleko, hleb, pelene\}) = 2.$$

Za pravilo

$$\{mleko, pelene\} \Rightarrow \{pivo\}$$

dobijamo:

$$sup = \frac{\#(\{mleko, pelene, pivo\})}{N} = \frac{2}{5},$$

jer se skup $\{mleko, pelene, pivo\}$ pojavljuje u dve od ukupno pet transakcija. Takođe,

$$conf = \frac{\#(\{mleko, pelene, pivo\})}{\#(\{mleko, pelene\})} = \frac{2}{3}.$$

To znači da se u dve trećine transakcija koje sadrže i mleko i pelene pojavljuje i pivo. Ovakva pravila mogu biti korisna, na primer, za raspored proizvoda u prodavnici, planiranje promocija ili preporuke kupcima.

2.7.2 Klasterovanje

Klasterovanje predstavlja grupisanje instanci po sličnosti. To je jedan od osnovnih zadataka nenadgledanog učenja. Osnovna ideja je da objekti unutar istog klastera međusobno što više liče, dok objekti iz različitih klastera treba da budu što različitiji.

Klasterovanje se koristi kada nemamo unapred poznate klase, već želimo da neke grupe izdvojimo iz podataka. Na primer, kupci se mogu grupisati prema obrascima kupovine, dokumenti prema temama, a merenja prema sličnosti ponašanja.

Pri klasterovanju je presudno kako se definiše sličnost između objekata. Ta definicija zavisi od tipa podataka: za numeričke podatke često se koristi rastojanje u prostoru atributa, za binarne podatke mere preklapanja, za tekstualne podatke kosinusna sličnost, a za grafovske i sekvencijalne podatke specijalizovane mere. Zbog toga kvalitet klasterovanja ne zavisi samo od algoritma, već i od izabrane reprezentacije podataka i mere sličnosti.

Posmatrano u okviru matrice podataka, klasterovanje proučava pre svega *relacije između vrsta*. Ono pokušava da otkrije koje vrste pripadaju istim grupama na osnovu sličnosti vrednosti atributa. Na taj način klasterovanje daje sažet prikaz strukture podataka i često služi i kao priprema za druge zadatke analize.

2.7.3 Klasifikacija

Klasifikacija proučava odnos između ulaznih atributa i ciljne promenljive. Za razliku od klasterovanja, ovde unapred znamo kojim klasama pripadaju primeri u skupu za obuku, pa se na osnovu tih primera formira model koji predviđa klasu novih instanci.

Klasifikacija je metoda nadgledanog mašinskog učenja. Cilj je izgraditi model koji predviđa klasu nove instance. Drugim rečima, ciljna promenljiva usmerava proces učenja i modelu pokazuje šta treba da nauči da razlikuje.

Neka je dat skup podataka X predstavljen matricom atributa

$$X \in \mathbb{R}^{n \times d},$$

gde je n broj instanci, a d broj atributa, i neka je svakom redu matrice pridružena oznaka klase iz skupa $\{1, \ldots, k\}$. Te oznake možemo zapisati vektorom

$$y \in \{1, \ldots, k\}^n.$$

Klasifikacija podrazumeva da se na osnovu parova (X, y) formira model koji se zatim koristi za predviđanje oznake klase nove instance Y. Postupak klasifikacije obuhvata najmanje dve faze:

- fazu treniranja, u kojoj se na osnovu poznatih primera formira model,
- fazu testiranja, u kojoj se model koristi za predviđanje klase novih instanci.

Primeri primene klasifikacije su predviđanje da li je poruka spam, da li će korisnik vratiti kredit, kojoj kategoriji pripada dokument ili kojoj vrsti pripada posmatrani objekat.

2.7.4 Otkrivanje anomalija

Otkrivanje anomalija predstavlja jedan od osnovnih zadataka istraživanja podataka. Za datu matricu podataka D, cilj je odrediti redove u matrici koji su jako različiti od ostatka redova. Takvi podaci nazivaju se *odudarajući podaci* (*outlier*), jer se u značajnoj

meri razlikuju od ostalih podataka. To su instance koje se ne uklapaju u zakonitosti koje su prisutne u ostalim podacima. Važno je razlikovati šum od anomalije: šum je često posledica slučajne greške merenja, dok anomalija može biti stvaran i važan događaj koji upravo želimo da otkrijemo. Primeri gde je od značaja otkrivanja anomalija su:

- otkrivanje upada u računarski sistem,
- identifikacija SPAM poruka,
- zloupotreba kreditnih kartica,
- medicinska dijagnostika,
- sprovođenje zakona,
- analiza anomalija u veb logovima.

U svim ovim slučajevima cilj je da se među velikim brojem uobičajenih, čestih primera izdvoje oni koji ukazuju na prevaru, grešku, napad, bolest ili neki drugi važan i neuobičajen događaj.

Otkrivanje anomalija je komplementarno klasterovanju: dok klasterovanje traži guste i slične grupe objekata, detekcija anomalija traži objekte koji se ne uklapaju u takve grupe. U nekim metodama se upravo na osnovu odstupanja od klastera procenjuje koliko je neka instanca neobična.

2.7.5 Analiza i vizuelizacija rezultata

Analiza i vizuelizacija rezultata predstavljaju važnu tehniku istraživanja podataka. Njihova uloga nije samo da prikažu rezultat rada algoritma, već i da omoguće bolje razumevanje podataka, uočavanje obrazaca, odstupanja i struktura koje nije lako primetiti samo pregledom tabele.

Vizuelni prikazi su posebno važni kod višedimenzionalnih, vremenskih i prostornih podataka, jer omogućavaju intuitivnije sagledavanje odnosa među instancama i atributima. Vizuelizacija može pomoći da se:

- uoče grupe sličnih objekata,
- prepoznaju neuobičajene instance,
- proveri smislenost dobijenih pravila,
- lakše interpretiraju rezultati klasifikacije i klasterovanja.

Zbog toga, analiza i vizuelizacija nisu samo završni korak procesa, već predstavljaju fazu koja omogućava dublje razumevanje podataka i proveru kvaliteta dobijenih modela.

2.8 Povezanost osnovnih tehnika istraživanja podataka

2.8.1 Međusobni odnos osnovnih tehnika istraživanja podataka

Osnovne tehnike istraživanja podataka nisu izolovane metode, već međusobno povezani načini analize matrice podataka. Pravila pridruživanja i klasifikacija pre svega analiziraju

relacije između kolona. Klasterovanje i otkrivanje anomalija pre svega analiziraju relacije između vrsta. Analiza i vizuelizacija omogućavaju da se rezultati svih ovih metoda sagledaju, protumače i provere.

Na taj način osnovne tehnike istraživanja podataka pokrivaju više različitih vrsta odnosa:

- odnose između atributa,
- odnose između instanci,
- odnose između grupa atributa unutar jedne instance,
- odstupanja pojedinačnih instanci od opštih obrazaca.

Ovakva podela pokazuje zašto su upravo pravila pridruživanja, klasifikacija, klasterovanje i otkrivanje anomalija glavne tehnike istraživanja podataka, s obzirom da obuhvataju najvažnije načine na koje se u podacima mogu tražiti obrasci i zakonitosti.

2.9 Primeri primene

Istraživanje podataka ima veliki broj praktičnih primena. Među tipičnim primerima su:

- raspoređivanje proizvoda u radnjama,
- preporuke kupcima,
- anomalije u veb logovima.

Ove primene mogu se preciznije objasniti na sledeći način:

- **raspoređivanje proizvoda u prodavnicama** – koriste se pravila pridruživanja da bi se otkrilo koje proizvode kupci često uzimaju zajedno, pa se ti proizvodi mogu postaviti blizu jedan drugom;
- **sistemi za preporuku** – na osnovu prethodnog ponašanja korisnika predlažu se proizvodi, filmovi, muzika ili drugi sadržaji;
- **analiza veb logova** – ispituju se obrasci ponašanja korisnika na veb sajtovima, tokovi navigacije i učestalost određenih akcija, pri čemu se posebno mogu tražiti anomalije.

Pored navedenih primera, važnu ulogu istraživanje podataka ima i u:

- medicinskoj dijagnostici, gde se analiziraju laboratorijski nalazi, medicinski snimci i biomedicinski signali radi otkrivanja bolesti;
- detekciji upada i zloupotreba, gde se u logovima sistema i mrežnom saobraćaju traže neuobičajeni obrasci;
- finansijama, gde se analiziraju transakcije, kretanja cena i ponašanje klijenata;
- obradi tekstova i dokumenata, gde se vrši klasifikacija, grupisanje i pretraga velikih kolekcija tekstualnih zapisa;
- analizi prostornih i vremenskih obrazaca u meteorologiji, ekologiji i saobraćaju.

2.10 Zaključak

Najvažnije tehnike istraživanja podataka mogu se razumeti polazeći od matrice podataka D dimenzije $n \times d$, koja predstavlja osnovni oblik zapisa podataka. Nad takvom matricom istražuju se relacije između kolona, relacije između vrsta i relacije između grupa atributa unutar vrsta. Iz toga prirodno proizlaze osnovne tehnike: pravila pridruživanja, klasifikacija, klasterovanje, analiza i vizuelizacija rezultata, kao i otkrivanje anomalija.

Pravila pridruživanja otkrivaju koje se stavke često pojavljuju zajedno u istoj instanci. Klasifikacija koristi poznate oznake klasa da bi izgradila model za predviđanje novih instanci. Klasterovanje grupiše objekte prema sličnosti bez unapred poznatih klasa. Otkrivanje anomalija pronalazi instance koje se značajno razlikuju od ostatka podataka. Analiza i vizuelizacija omogućavaju da se svi ovi rezultati lakše razumeju i interpretiraju.

Posebno treba imati u vidu da vrsta podataka nije samo tehnički detalj, već jedan od ključnih činilaca koji određuju šta je uopšte moguće otkriti analizom. Zato istraživanje podataka uvek treba da započne pitanjem: *kakvi su naši podaci i kakve odnose oni sadrže?* Tek kada su ovi temelji jasni, osnovne tehnike kao što su klasifikacija, klasterovanje, pravila pridruživanja i otkrivanje anomalija mogu dati smislene i korisne rezultate.

Glava 3

Mere sličnosti i različitosti

Mere sličnosti i različitosti zauzimaju važno mesto u istraživanju podataka zato što se koriste u brojnim tehnikama, kao što su klasterovanje, klasifikacija (na primer, algoritam najbližih suseda) i otkrivanje anomalija. U mnogim primenama, nakon izračunavanja sličnosti ili različitosti između objekata, originalni skup podataka više nije ni neophodan za dalju analizu, jer se analiza može obavljati direktno u prostoru sličnosti ili različitosti. U tom smislu, ove mere predstavljaju osnovu velikog broja metoda za analizu podataka.

3.1 Osnovni pojmovi

Jedno od osnovnih pitanja u istraživanju podataka jeste: *kako odrediti koliko su dva objekta slična ili različita?* Opšti termin koji obuhvata i sličnost i različitost jeste *blizina* (*proximity*). Intuitivno:

- **sličnost** je vrednost koja opisuje koliko su dva objekta nalik jedan drugom;
- **različitost** je vrednost koja opisuje koliko su dva objekta međusobno različita;

Kod funkcija sličnosti važi: *veća vrednost znači veću sličnost.* Kod funkcija različitosti, odnosno rastojanja, važi obrnuto: *manja vrednost znači veću sličnost.* Sličnost se često izražava vrednostima iz intervala $[0, 1]$, gde 0 označava odsustvo sličnosti, a 1 potpunu sličnost. Važno je napomenuti da sličnost od 1 ne mora da znači da su dva objekta „ista", jer, na primer, možda ne raspolažemo svim atributima tih objekata. Tako, maksimalna sličnost znači da su objekti *identični u posmatranom prostoru i prema toj meri*, ali ne nužno i u apsolutnom smislu. Različitost je često u intervalu $[0, +\infty)$, mada u nekim slučajevima može biti i ograničena na $[0, 1]$.

Često se termin *rastojanje* koristi kao sinonim za različitost, ali je važno naglasiti da se rastojanje često odnosi na specijalnu klasu različitosti, odnosno da nije svaka mera različitosti ujedno i metrika. To će biti razjašnjeno kasnije.

3.2 Transformacije između sličnosti i različitosti

U praksi je često potrebno transformisati sličnost u različitost, ili obrnuto, kao i preslikati meru u određeni opseg, najčešće u $[0, 1]$. Ovaj postupak se naziva *normalizacija*. To je važno zato što pojedini algoritmi ili softverski paketi očekuju baš određeni tip mere.

Ako je sličnost već u intervalu $[0, 1]$, tada se različitost može definisati kao

$$d = 1 - s.$$

Obrnuto, ako je različitost u intervalu $[0, 1]$, tada se sličnost može definisati kao

$$s = 1 - d.$$

Za **mere sa konačnim opsegom** $[s_{\min},\ s_{\max}]$ koristi se sledeća linearna transformacija poznata pod nazivom *min-max normalizacija*:

$$\boxed{s' = \frac{s - s_{\min}}{s_{\max} - s_{\min}}} \tag{3.1}$$

Ova formula preslikava svaku vrednost u interval $[0, 1]$:

- minimalna vrednost $s_{\min}$ postaje 0,
- maksimalna vrednost $s_{\max}$ postaje 1,
- sve ostale vrednosti se proporcionalno raspoređuju između.

Ako je s mera sličnosti a d mera različitosti, sa opsezima $[s_{\min}, s_{\max}]$ i $[d_{\min}, d_{\max}]$, redom, obe normalizujemo na isti način:

$$s' = \frac{s - s_{\min}}{s_{\max} - s_{\min}}, \qquad d' = \frac{d - d_{\min}}{d_{\max} - d_{\min}}.$$

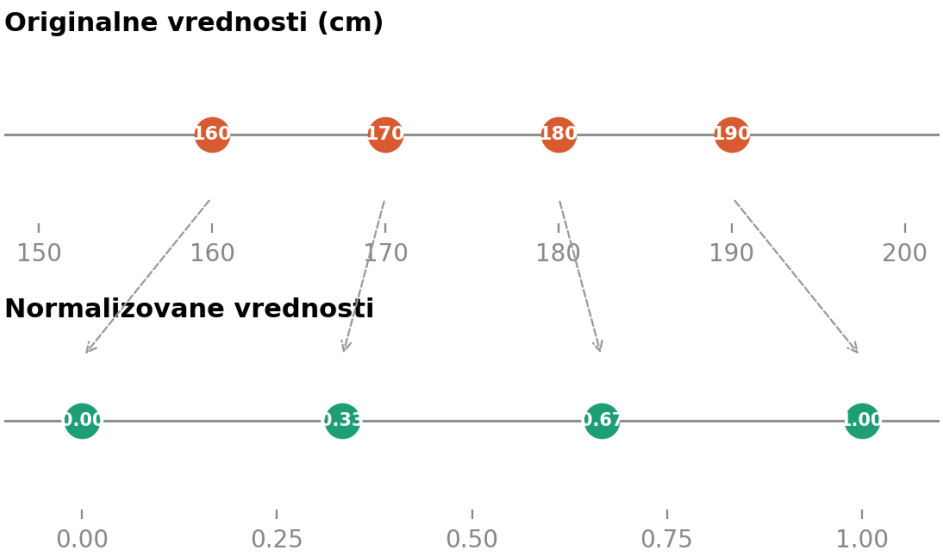
Primer: visina studenata Zamislimo da imamo podatke o visini četiri studenta u centimetrima: 160, 170, 180 i 190 cm.

Ovde je $s_{\min} = 160$ i $s_{\max} = 190$, pa je $s_{\max} - s_{\min} = 30$. Primenjujemo formulu:

Visina (cm)	Račun	Normalizovana vrednost
160	$\frac{160-160}{30} = \frac{0}{30}$	**0.00**
170	$\frac{170-160}{30} = \frac{10}{30}$	**0.33**
180	$\frac{180-160}{30} = \frac{20}{30}$	**0.67**
190	$\frac{190-160}{30} = \frac{30}{30}$	**1.00**

Tabela 3.1: Normalizacija visine studenata na interval [0, 1].

Na slici 3.1 prikazano je kako se originalne vrednosti preslikavaju na normalizovani interval:



Slika 3.1: Preslikavanje originalnih vrednosti na interval [0, 1] pomoću min-max normalizacije.

Ključna osobina: čuvanje relativnih rastojanja Najvažnija osobina ove transformacije je da čuva relativna rastojanja između tačaka.

Pogledajmo razmake pre i posle normalizacije:

$$\begin{aligned} \text{Pre:} \quad & |170 - 160| = 10\,\text{cm}, \quad |190 - 180| = 10\,\text{cm} \quad \Rightarrow \quad \text{isti razmak.} \\ \text{Posle:} \quad & |0.33 - 0.00| = 0.33, \quad |1.00 - 0.67| = 0.33 \quad \Rightarrow \quad \text{isti razmak.} \end{aligned}$$

Drugim rečima: ako su tačke x_1 i x_2 međusobno udaljene dva puta više nego tačke x_3 i x_4, isto će važiti i nakon linearne transformacije. Transformacija menja *skalu*, ali ne narušava *odnose* između podataka.

Min-max normalizacija zahteva da znamo i minimum i maksimum mere. Međutim, mere različitosti (rastojanja) često nemaju gornju granicu – njihov opseg je $[0, \infty)$. Na primer, euklidsko rastojanje između dve tačke može biti 0.5, ali i 1 000 ili 1 000 000. Pošto $d_{\max}$ nije definisan, linearna formula jednostavno ne može da se primeni.

Rešenje: nelinearne transformacije Umesto linearne formule, koristimo **monotono opadajuću funkciju** koja:

- pretvara različitost $d \in [0, \infty)$ u sličnost $s \in (0, 1]$,
- za $d = 0$ (potpuna jednakost) daje $s = 1$ (maksimalna sličnost),
- za $d \to \infty$ daje $s \to 0$ (minimalna sličnost).

Dva česta izbora su:

$$s = \frac{1}{1+d}, \qquad s = e^{-d}. \tag{3.2}$$

Obe funkcije ispunjavaju gornje uslove: opadaju od 1 ka 0 kako d raste (slika 3.2, levo).

Različitost d	**Račun**	**Sličnost** s
0	$\frac{1}{1+0} = 1$	**1.000**
0.5	$\frac{1}{1+0.5} = 0.667$	**0.667**
2	$\frac{1}{1+2} = 0.333$	**0.333**
10	$\frac{1}{1+10} = 0.091$	**0.091**
100	$\frac{1}{1+100} = 0.0099$	**0.010**
1000	$\frac{1}{1+1000} = 0.001$	**0.001**

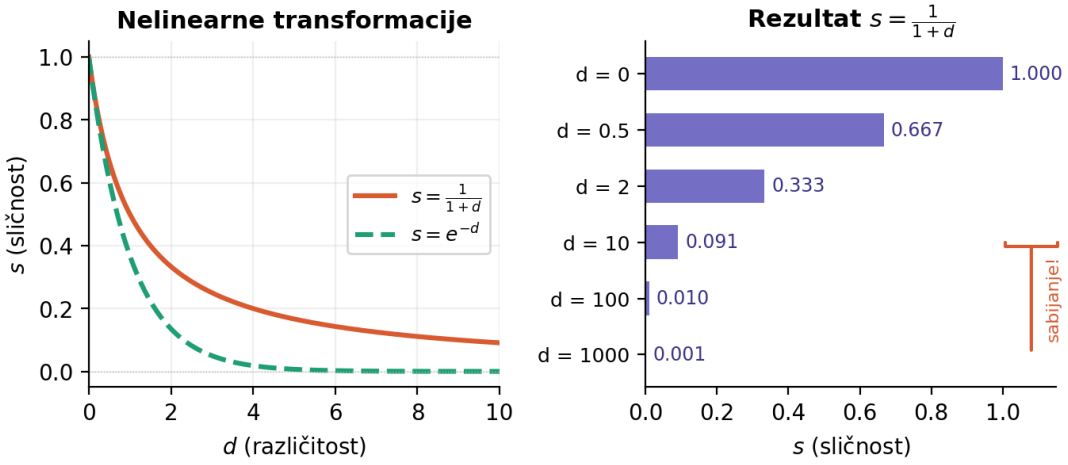
Tabela 3.2: Nelinearna transformacija sabija velike različitosti u uzan opseg blizu 0.

Primer: sabijanje velikih vrednosti Primenimo transformaciju $s = \frac{1}{1+d}$ na konkretne vrednosti različitosti:

Primetimo ključnu razliku u odnosu na linearnu normalizaciju:

- Razlika između $d = 0$ i $d = 2$ na novoj skali iznosi $1.000 - 0.333 = 0.667$ – **veliki raspon**.
- Razlika između $d = 10$ i $d = 1000$ iznosi samo $0.091 - 0.001 = 0.090$ – **mali raspon**.

Dakle, nelinearna transformacija **ne čuva relativna rastojanja**. Male različitosti se jasno razlikuju na novoj skali, dok se velike različitosti "sabijaju" u uzan opseg blizu nule. Da li je to poželjno, zavisi od konkretne primene – nekada je upravo to korisno (npr. kada nas više zanimaju male razlike), a nekada može sakriti važne odnose među podacima.



Slika 3.2: Levo: oblik dve nelinearne transformacije. Desno: rezultat primene $s = \frac{1}{1+d}$ – veće vrednosti d se sabijaju u uzan opseg.

Kada je sabijanje poželjno? Zamislimo sistem za preporuku filmova. Sistem računa rastojanje između korisnika na osnovu njihovih ocena – što je rastojanje manje, ukusi su sličniji. Tipične vrednosti rastojanja mogu biti:

- $d = 0.5$ – dva korisnika koji vole gotovo iste filmove,

- $d = 2$ – korisnici sa delimično sličnim ukusom,
- $d = 100$ – korisnici sa potpuno različitim ukusom,
- $d = 1000$ – korisnici koji nemaju nijedan zajednički žanr.

Za preporuke, ključno je razlikovati *bliske* korisnike ($d = 0.5$ vs. $d = 2$), jer upravo od njih preuzimamo preporuke. S druge strane, potpuno nam je svejedno da li je neko "veoma različit" ($d = 100$) ili "ekstremno različit" ($d = 1000$) – ni od jednog nećemo preuzeti preporuku. Nelinearna transformacija upravo to radi: fino razlikuje bliske vrednosti, a udaljene tretira gotovo isto ($s = 0.010$ vs. $s = 0.001$).

Međutim, u drugim primenama ovo sabijanje može biti **nepoželjno**. Na primer, ako analiziramo geografska rastojanja između gradova, razlika između 100 km i 1000 km je suštinski važna i ne želimo da je izgubimo.

Treba biti oprezan: Transformacija može promeniti interpretaciju mere. Uzmimo za primer *koeficijent korelacije* koji uzima vrednosti između $[-1, 1]$ pri čemu 1 označava veliku pozitivnu zavisnost (npr. visina i težina ljudi (viši ljudi su u proseku i teži)), -1 veliku negativnu zavisnost (npr. cena proizvoda i potražnja (proizvodi sa višom cenom su u proseku manje traženi)), a 0 odsustvo zavisnosti(npr. broj cipela i IQ). Detaljnije o korelaciji govorimo u narednom poglavlju.

Tako na primer, ako se korelacija, koja prirodno uzima vrednosti iz intervala $[-1, 1]$, preslika u $[0, 1]$, može se izgubiti informacija o *znaku* veze.

Ilustracija problema Neka su date korelacije $r_1 = -0.8$ i $r_2 = 0.8$. Obe ukazuju na jaku vezu između promenljivih, ali je prva negativna (kada jedna promenljiva raste, druga opada), a druga pozitivna (obe rastu zajedno). Razmotrimo dve moguće transformacije:

1. **Apsolutna vrednost:** $s = |r|$. Dobijamo $s_1 = s_2 = 0.8$ – informacija o smeru veze je potpuno izgubljena.
2. **Linearna transformacija:** $s = (1 + r)/2$. Dobijamo $s_1 = 0.1$ i $s_2 = 0.9$ – smer se čuva (nula se preslikava u 0.5 pa se razlikuju se negativna i pozitivna vrednost), ali savršena negativna korelacija ($r = -1$) postaje $s = 0$, što znači potpuno odsustvo sličnosti, iako ona zapravo ukazuje na jaku vezu suprotnog smera.

Izbor transformacije zavisi od toga da li je za konkretnu primenu smer veze relevantan ili nije.

Kada smer veze nije relevantan Zamislimo da nastavnik želi da otkrije koji faktori najviše utiče na ukupan uspeh učenika. Neki faktori su u pozitivnoj korelaciji sa ukupnim uspehom ($r = 0.8$, npr. ocena iz matematike – bolja ocena iz matematike znači bolji opšti uspeh), a neke u negativnoj ($r = -0.7$, npr. broj izostanaka – više izostanaka, slabiji uspeh). Nastavnika zanima samo koji faktori snažno utiču na uspeh, bez obzira da li ga podižu ili spuštaju jer ne donosi odluku na osnovu smera. I matematika ($r = 0.8$) i izostanci ($r = -0.7$) su korisne informacije. Ovde transformacija $s = |r|$ ima smisla: daje $s = 0.8$ i $s = 0.7$, čime se jasno vidi da su oba faktora važna.

Kada je smer veze relevantan Zamislimo sada da klimatolog proučava kako temperatura utiče na potrošnju energije u različitim gradovima. U hladnim gradovima, korelacija je negativna ($r = -0.8$): kada temperatura *padne*, potrošnja energije za grejanje *raste*. U tropskim gradovima, korelacija je pozitivna ($r = 0.8$): kada temperatura *raste*, potrošnja energije za hlađenje *raste*. Ovde bi transformacija $s = |r|$ dala istu vrednost $s = 0.8$ za oba grada, što bi sakrilo suštinsku razliku: prvi grad troši energiju *zimi*, a drugi *leti*. Za planiranje energetske mreže, smer veze je ključna informacija – transformacija ga ne sme izbrisati.

3.3 Sličnost i različitost pojedinačnih atributa

Pošto se mera sličnosti (različitosti) složenih objekata obično definiše kao kombinacija mera sličnosti (različitosti) pojedinačnih atributa, najpre treba definisati mere za slučaj kada objekti imaju samo jedan atribut.

3.3.1 Nominalni (imenski) atributi

Kod nominalnih atributa postoji samo informacija o tome da li su dve vrednosti iste ili nisu. Zato je prirodno definisati:

$$s(p,q) = \begin{cases} 1, & \text{ako } p = q, \\ 0, & \text{ako } p \neq q, \end{cases} \qquad d(p,q) = \begin{cases} 0, & \text{ako } p = q, \\ 1, & \text{ako } p \neq q. \end{cases}$$

Primer. Ako atribut *boja* uzima vrednosti `crvena`, `plava`, `zelena`, tada su `crvena` i `crvena` potpuno slične, dok su `crvena` i `plava` različite bez dodatnog stepena razlikovanja.

3.3.2 Redni atributi

Kod rednih atributa pored različitosti postoji i poredak. Vrednosti se zato preslikavaju na uzastopne cele brojeve iz skupa $\{0, 1, \ldots, n-1\}$, gde je n broj mogućih vrednosti, i sličnost i različitost se računaju nad ovim vrednostima. Dalje sličnost i različitost definišemo kao:

$$d(p,q) = \frac{|p-q|}{n-1}, \qquad s(p,q) = 1 - \frac{|p-q|}{n-1}.$$

Primer. Ako kvalitet proizvoda može biti `loš`, `osrednji`, `dobar`, `odličan`, tada se može mapirati na $0, 1, 2, 3$. Razlika između `dobar` i `odličan` tada je manja nego između `loš` i `odličan`.

Treba uočiti da ovakav pristup pretpostavlja jednaka rastojanja između susednih nivoa, što ne mora uvek biti potpuno opravdano, ali je standardan praktičan izbor kada nemamo dodatne informacije.

3.3.3 Intervalni i razmerni atributi

Kod nominalnih atributa gleda se samo da li su isti ili različiti, kod rednih atributa važan je poredak, a kod intervalnih i razmernih atributa važna je i stvarna numerička udaljenost.

Za intervalne i razmerne atribute prirodna mera različitosti jeste apsolutna razlika:

$$d(p, q) = |p - q|.$$

Ovakav izbor je prirodan zato što su intervalni i razmerni atributi numerički, pa kod njih ima smisla posmatrati koliko su dve vrednosti međusobno udaljene. Apsolutna razlika ima nekoliko poželjnih svojstava: uvek je nenegativna, jednaka je nuli kada su vrednosti iste, simetrična je:

$$|p - q| = |q - p|,$$

i raste kada se vrednosti sve više razlikuju. Na primer, vrednosti 10 i 12 imaju različitost 2, dok vrednosti 10 i 50 imaju različitost 40, što odgovara intuitivnom osećaju da su 10 i 12 sličnije nego 10 i 50.

Sličnost se zatim dobija transformacijom različitosti. Razlog za to je što različitost i sličnost imaju suprotan smisao: mala različitost treba da odgovara velikoj sličnosti, a velika različitost maloj sličnosti. Zato se od funkcije različitosti d konstruiše funkcija sličnosti s koja opada kada d raste. Na primer:

$$s = -d, \qquad s = \frac{1}{1+d}, \qquad s = e^{-d}, \qquad s = 1 - \frac{d - \min\_d}{\max\_d - \min\_d}.$$

Transformacija

$$s = -d$$

je najjednostavnija moguća: što je različitost manja, sličnost je veća. Ipak, ova definicija daje negativne vrednosti, pa je manje pogodna za intuitivno tumačenje.

Zbog toga se često koriste transformacije koje preslikavaju sličnost u interval $[0, 1]$. Na primer,

$$s = \frac{1}{1+d}$$

daje vrednost 1 kada je $d = 0$, a zatim monotono opada ka nuli kako različitost raste. Slično tome,

$$s = e^{-d}$$

takođe daje maksimalnu sličnost 1 za $d = 0$, ali opada brže od prethodne funkcije, pa jače kažnjava veće razlike.

Poslednja transformacija,

$$s = 1 - \frac{d - \min\_d}{\max\_d - \min\_d},$$

predstavlja normalizaciju različitosti na interval $[0, 1]$. Najmanja posmatrana različitost preslikava se na vrednost blisku 1, a najveća na vrednost 0. Ovakav pristup je koristan kada je potrebno porediti sličnosti na jedinstvenoj skali.

Suštinski, za intervalne i razmerne atribute apsolutna razlika je prirodna mera različitosti zato što direktno meri numeričku udaljenost između dve vrednosti, dok se sličnost dobija odgovarajućom transformacijom te udaljenosti tako da manja udaljenost znači veću sličnost.

Treba napomenuti da kod razmernih atributa apsolutna razlika nije uvek jedina moguća ni najbolja mera. Razmotrimo cene dva para proizvoda: prvi par košta 100 i 200 dinara, a drugi 50 000 i 50 100 dinara. Apsolutna razlika u oba slučaja iznosi 100 dinara. Međutim, u prvom slučaju jedan proizvod je duplo skuplji od drugog, dok je u drugom

slučaju razlika u ceni zanemarljiva i iznosi svega 0.2%. Ako bismo koristili samo apsolutnu razliku, ova dva para proizvoda bila bi podjednako različita, što ne odgovara intuiciji. U takvim situacijama važniji je odnos između vrednosti nego njihova apsolutna razlika. Tada se mogu koristiti alternativne mere, na primer relativna razlika:

$$d(p, q) = \frac{|p - q|}{\max(|p|, |q|)},$$

koja za prvi par daje $d = 100/200 = 0.5$, a za drugi $d = 100/50\,100 \approx 0.002$, čime se razlika u značaju jasno izražava.

3.4 Metrika i ultrametrika

Kada govorimo o rastojanju između objekata, prirodno je očekivati da ono zadovoljava neke osnovne osobine. Funkcija rastojanja d je **metrika** ako za sve objekte p, q, r važi:

1. **Pozitivna definitnost:**

$$d(p, q) \geq 0, \qquad d(p, q) = 0 \iff p = q.$$

   Rastojanje je uvek nenegativno, i jednako je nuli samo ako su objekti identični.

2. **Simetrija:**

$$d(p, q) = d(q, p).$$

   Rastojanje od p do q je isto kao od q do p.

3. **Nejednakost trougla:**

$$d(p, r) \leq d(p, q) + d(q, r).$$

   Put direktno od p do r ne može biti duži od puta koji ide *preko* q.

Primer: nejednakost trougla Razmotrimo rastojanja između tri grada: Beograd (p), Kragujevac (q) i Niš (r).

- $d(p, q) = 140\,\text{km}$ (Beograd – Kragujevac),
- $d(q, r) = 130\,\text{km}$ (Kragujevac – Niš),
- $d(p, r) = 240\,\text{km}$ (Beograd – Niš, direktno autoputem).

Nejednakost trougla kaže: $d(p, r) \leq d(p, q) + d(q, r)$, tj. $240 \leq 140 + 130 = 270$. Važi! Direktan put do Niša ne može biti duži od zaobilaznog puta preko Kragujevca. Ovo je intuitivno: *zaobilaženje nikada ne skraćuje put.*

Ova osobina ima i **praktičnu primenu** u algoritmima. Na primer, prilikom klasterovanja, ako znamo da je objekat A blizu objektu B, a B daleko od C, nejednakost trougla nam garantuje da A ne može biti previše blizu C – i možemo preskočiti nepotrebna poređenja, čime se algoritam ubrzava.

Ultrametrika: strožija verzija rastojanja Ako funkcija rastojanja zadovoljava sve tri osobine metrike, ali umesto nejednakosti trougla važi **jača** osobina:

$$d(p, r) \leq \max\{d(p, q),\ d(q, r)\},$$

tada je d **ultrametrika**. Razlika je suptilna ali važna: umesto zbira, koristi se *maksimum* dve strane. Pošto je $\max\{a, b\} \leq a + b$ za sve nenegativne a, b, ultrametrika je *stroži* uslov od obične metrike – svaka ultrametrika je automatski i metrika, ali ne i obrnuto.

Primer: porodično stablo Ultrametrike se najprirodnije javljaju u hijerarhijskim strukturama. Zamislimo porodično stablo i definišimo "rastojanje" između rođaka kao *koliko generacija unazad moramo ići da bismo pronašli zajedničkog pretka*:

- **Ana i Marko** su brat i sestra (isti roditelji) $\Rightarrow d(\text{Ana}, \text{Marko}) = 1$ (jedna generacija unazad).
- **Marko i Jovan** su braća od tetke (zajednički su im deda i baba) $\Rightarrow d(\text{Marko}, \text{Jovan}) = 2$ (dve generacije unazad).
- **Ana i Jovan** su takođe braća od tetke (isti deda i baba kao za Marka i Jovana!) $\Rightarrow d(\text{Ana}, \text{Jovan}) = 2$.

Proverimo ultrametričku nejednakost za $p = \text{Ana}$, $q = \text{Marko}$, $r = \text{Jovan}$:

$$d(\text{Ana}, \text{Jovan}) \leq \max\{d(\text{Ana}, \text{Marko}),\ d(\text{Marko}, \text{Jovan})\} = \max\{1,\ 2\} = 2.$$

Važi, jer $2 \leq 2$! Ovo ilustruje ključnu osobinu ultrametrike: ako su Ana i Marko bliski (ista uža porodica), a Marko i Jovan udaljeniji (šira porodica), onda su i Ana i Jovan udaljeni tačno onoliko koliko i Marko i Jovan – jer se "grananje" u porodičnom stablu desilo na istom nivou. Ana ne može biti bliža Jovanu nego što je Marko, jer ona i Marko dele istu granu stabla.

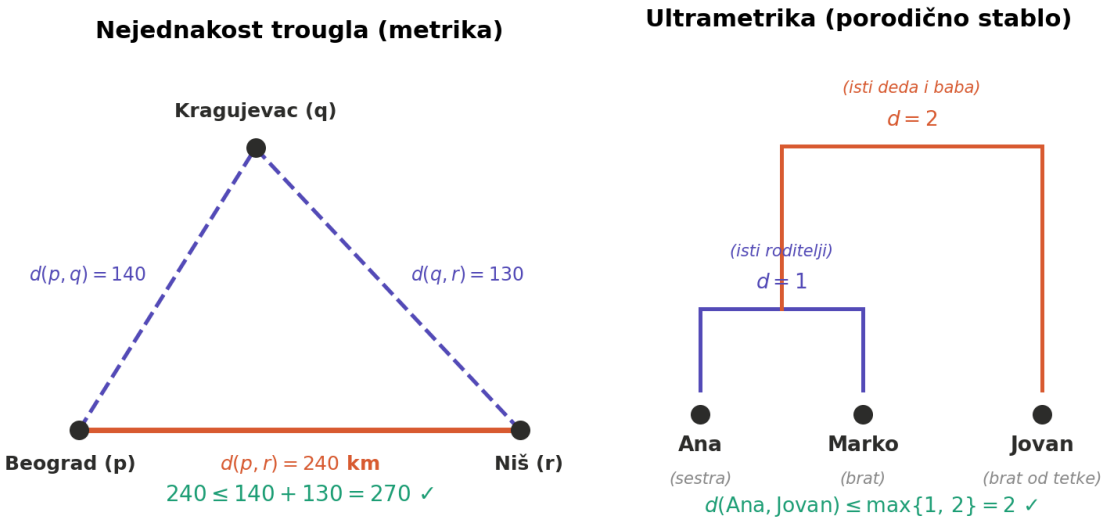
Ova osobina čini ultrametrike idealnim za hijerarhijsko klasterovanje: svaki nivo dendrograma (stabla grupisanja) definiše ultrametriku nad podacima.

Kada rastojanje nije metrika? U praksi se često koriste funkcije koje "izgledaju" kao rastojanje, ali ne zadovoljavaju sve osobine metrike. Razmotrimo konkretan primer.

Neka je $d(p, q)$ **cena avionske karte** od grada p do grada q. Proverimo osobine metrike:

1. **Pozitivna definitnost:** $d(p, q) \geq 0$ – važi (karta ne može imati negativnu cenu). Međutim, $d(p, p) = 0$ je diskutabilno – let od Beograda do Beograda nema smisla, ali formalno možemo reći da je njegova cena 0.
2. **Simetrija:** Da li je $d(p, q) = d(q, p)$? **Ne!** Na primer, karta Beograd $\to$ Njujork može koštati 350 €, dok karta Njujork $\to$ Beograd košta 500 € (veća potražnja, drugi porezi, druga valuta). Dakle:

$$d(\text{BG}, \text{NY}) = 350 \neq 500 = d(\text{NY}, \text{BG}).$$



Slika 3.3: Levo: nejednakost trougla na primeru gradova – direktan put nije duži od zaobilaznog. Desno: ultrametrika u porodičnom stablu – rastojanje se čita sa nivoa na kojem se grane spajaju.

3. **Nejednakost trougla:** Takođe ne važi! Direktan let Beograd → Tokio može koštati 800 €, dok kombinacija Beograd → Istanbul (80 €) + Istanbul → Tokio (400 €) košta ukupno 480 €. Dakle:

$$d(\text{BG}, \text{Tokio}) = 800 > 480 = d(\text{BG}, \text{Istanbul}) + d(\text{Istanbul}, \text{Tokio}).$$

   Zaobilazni put je *jeftiniji* od direktnog – upravo suprotno od onoga što nejednakost trougla zahteva.

Cena avionske karte **nije metrika** jer krši i simetriju i nejednakost trougla. Ipak, ona je savršeno korisna mera rastojanja u svakodnevnom životu. Ovo pokazuje da se u praksi često radi sa merama koje nisu metrike, ali to ne znači da su beskorisne – samo treba biti svestan koja svojstva važe, a koja ne, jer algoritmi koji pretpostavljaju metriku mogu dati pogrešne rezultate ako im se prosledi nemetričko rastojanje.

3.5 Mere različitosti za kvantitativne podatke

Zamislite da imate podatke o kupcima u obliku tabele: svaki kupac je opisan nizom brojeva (starost, prihod, broj kupovina, ...). Da bismo uporedili kupce, jedan od načina je da kvantifikujemo koliko se oni razlikuju. Tu ulogu imaju **mere različitosti** koje predstavljaju funkcije koje svakom paru objekata dodeljuju neku vrednost koja opisuje koliko se razlikuju. Što je vrednost veća, to se objekti više razlikuju.

Formalnije, objekte predstavljamo kao vektore u n-dimenzionom prostoru:

$$X = (x_1, x_2, \ldots, x_n), \qquad Y = (y_1, y_2, \ldots, y_n),$$

gde je n broj atributa (dimenzija). Možemo govoriti o razlici na nivou pojedinačnih atributa, a mera različitosti *kombinuje* ove pojedinačne razlike u jednu ukupnu vrednost.

3.5.1 Hamingovo rastojanje

Hamingovo rastojanje meri na koliko se pozicija dve sekvence iste dužine razlikuju i posebno je prirodno za binarne, kategoričke ili simboličke podatke:

$$\text{Hamming}(X, Y) = \sum_{i=1}^{n} q_i,$$

gde je

$$q_i = \begin{cases} 1, & \text{ako } x_i \neq y_i, \\ 0, & \text{inače.} \end{cases}$$

Kod binarnih vektora Hamingovo rastojanje predstavlja broj bitova koji se razlikuju.

Primer. Neka je $X = (1, 0, 1, 1, 0)$ i $Y = (1, 1, 1, 0, 0)$. Poređenje po komponentama:

	1	2	3	4	5
X	1	0	1	1	0
Y	1	1	1	0	0
	$=$	$\neq$	$=$	$\neq$	$=$

Objekti se razlikuju u drugoj i četvrtoj komponenti, pa je $\text{Hamming}(X, Y) = 2$.

Detekcija grešaka u prenosu podataka. Hamingovo rastojanje ima praktičnu primenu u telekomunikacijama. Ako pošaljete poruku `1011101`, a prijemnik primi `1001001`, Hamingovo rastojanje je 2 — znači da su se promenila tačno 2 bita tokom prenosa.

3.5.2 Rastojanje Minkovskog

Za opšte numeričke podatke potrebna nam je mera koja uzima u obzir *koliko se* vrednosti razlikuju, a ne samo *da li se* razlikuju. Rastojanje Minkovskog daje jednu familiju rastojanja parametrizovanu vrednošću p:

$$d(X, Y) = \left(\sum_{i=1}^{n} |x_i - y_i|^p \right)^{1/p}$$

Parametar $p \geq 1$ određuje *kako se kombinuju* razlike po koordinatama. Ne treba ga mešati sa brojem dimenzija n.

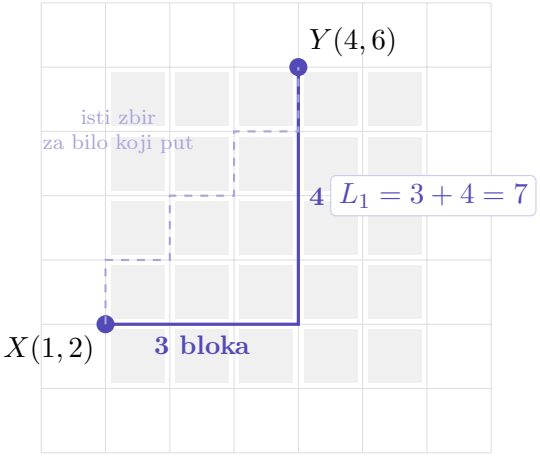
Ideja je da se za svaku dimenziju izračuna apsolutna razlika $|x_i - y_i|$, a zatim se sve razlike kombinuju pomoću stepena p i odgovarajućeg korena. Različite vrednosti p daju različita ponašanja, od ujednačenog doprinosa po svakoj dimenziji do situacije gde najveća razlika ima presudan uticaj.

Specijalni slučajevi

1. $p = 1$**: Menhetn rastojanje.**

$$d_1(X, Y) = \sum_{i=1}^{n} |x_i - y_i|$$

Zamislite da ste u Njujorku i želite da stignete od ugla Prve ulice i Druge avenije do ugla Četvrte ulice i Šeste avenije. Ne možete ići dijagonalno — zgrade vam blokiraju put. Morate ići tri bloka na istok i četiri bloka na sever, pa ukupno prelazite $3+4=7$ blokova. To je upravo L_1 rastojanje: **zbir apsolutnih razlika po svakoj dimenziji**, bez obzira na to kojim se redom krećete.

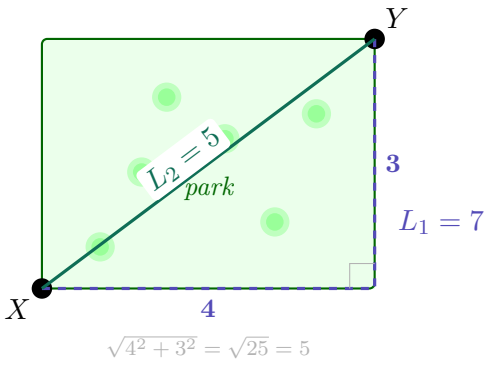


Slika 3.4: Menhetn rastojanje: bez obzira na to kojim putem idete po mreži ulica (puna ili isprekidana linija), ukupan zbir blokova je uvek isti — $L_1 = 7$.

2. $p = 2$**: Euklidsko rastojanje.**

$$d_2(X,Y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$$

Ovo je rastojanje "vazdušnom linijom" — onako kako biste lenjirom merili udaljenost između dve tačke na papiru. Upravo je ono što daje Pitagorina teorema: za pravougli trougao sa katetama a i b, hipotenuza iznosi $\sqrt{a^2+b^2}$.



Slika 3.5: Euklidsko rastojanje: prečica kroz park (zelena linija, $L_2 = 5$) je uvek kraća od puta po obodu (ljubičasta isprekidana linija, $L_1 = 7$).

Pravougaoni park. Na Slici 3.5, park ima dimenzije 4×3 bloka. Menhetn rastojanje oko oboda je $4+3=7$. Ali ako smete prečicom dijagonalno kroz park, preći ćete samo $\sqrt{16+9}=5$ blokova. To je euklidsko rastojanje — uvek manje ili jednako Menhetn rastojanju.

3. $p \to \infty$: **Supremum rastojanje (L_∞ norma).**

$$d_\infty(X, Y) = \max_{1 \le i \le n} |x_i - y_i|$$

Ovo rastojanje uzima u obzir samo dimenziju u kojoj je apsolutna razlika najveća a sve ostale ignoriše.

Kontrola kvaliteta. Fabrika proizvodi šraf i meri dve dimenzije: prečnik i dužinu (u mm). Šraf je neispravan ako *bilo koja* dimenzija prekorači dozvoljenu toleranciju (neka ona na primer iznosi 0.1 mm). Ako je idealni šraf $(10.0, 30.0)$, a proizvod je $(10.3, 30.1)$, tada je:

$$L_\infty = \max\{|10.0 - 10.3|,\ |30.0 - 30.1|\} = \max\{0.3, 0.1\} = 0.3.$$

Rastojanje od 0.3 kaže da je najgore odstupanje prečnik — i to je jedino što je bitno za odluku o ispravnosti.

Numeričko poređenje. Za tačke $X = (1, 2)$ i $Y = (4, 6)$ razlike po koordinatama su:

$$|x_1 - y_1| = |1 - 4| = 3, \qquad |x_2 - y_2| = |2 - 6| = 4.$$

Izračunavamo sva tri rastojanja:

$$L_1(X, Y) = 3 + 4 = \mathbf{7}, \qquad L_2(X, Y) = \sqrt{9 + 16} = \mathbf{5}, \qquad L_\infty(X, Y) = \max\{3, 4\} = \mathbf{4}.$$

Primetimo redosled $L_\infty \le L_2 \le L_1$. Ovo važi uvek jer sa rastom p, rastojanje sve više zavisi od najvećih razlika, ali se zbog uzimanja korena vrednost ukupnog rastojanja smanjuje ili ostaje ista.

Napomena. Hamingovo rastojanje je zapravo specijalan slučaj rastojanja Minkovskog (sa $p = 1$) primenjenog na binarne vektore. Pošto su komponente 0 ili 1, važi $|x_i - y_i| \in \{0, 1\}$, pa je zbir apsolutnih razlika isti kao broj nepodudaranja.

Svojstva metrike

Rastojanja poput euklidskog ispunjavaju tri svojstva koja ih čine **metrikom** — formalnom merom udaljenosti:

1. **Pozitivna definitnost:** $d(X, Y) \ge 0$ za sve X, Y, i $d(X, Y) = 0$ samo ako su X i Y identični.

2. **Simetrija:** $d(X, Y) = d(Y, X)$ — rastojanje od A do B jednako je rastojanju od B do A.

3. **Nejednakost trougla:** $d(X, Z) \le d(X, Y) + d(Y, Z)$ — prečica nikad nije duža od zaobilaznog puta.

Sva tri specijalna slučaja rastojanja Minkovskog (za $p \ge 1$) ispunjavaju ova svojstva što je od značaja za primenu u algoritmima klasterizacije.

Rastojanje Minkovskog sa težinama

U praksi, ne doprinose svi atributi jednako. Na primer, ako poredite stanove, razlika u broju soba je verovatno važnija od razlike u spratu. Zato se uvode **težine** a_i koje pojačavaju ili oslabljuju uticaj pojedinih dimenzija:

$$d(X, Y) = \left(\sum_{i=1}^{n} a_i \, |x_i - y_i|^p \right)^{1/p}$$

Veća težina a_i znači da razlika po i-tom atributu više utiče na ukupno rastojanje.

Poređenje stanova. Neka dva stana imaju sledeće atribute:

	Kupatila	Sobe	Sprat
Stan A	1	3	2
Stan B	2	2	5

Bez težina (L_1): $d = |1-2| + |3-2| + |2-5| = 1 + 1 + 3 = 5$.

Razlika u spratnosti dominira, dok je razlika u broju soba ili broju kupatila za nekog kupca možda praktično važnija. Ako stavimo težine $a_1 = 0.5$ (kupatila), $a_2 = 5$ (sobe), $a_3 = 0.5$ (sprat):

$$d = 0.5 \cdot 1 + 5 \cdot 1 + 0.5 \cdot 3 = 0.5 + 5 + 1.5 = 7.$$

Sada razlika u broju soba ima najveći uticaj na ukupno rastojanje, što bolje odražava stvarni značaj za kupca.

Kada rastojanje Minkovskog nije dobar izbor

Rastojanje Minkovskog nije univerzalno rešenje. Postoje situacije u kojima daje loše rezultate:

Retki i visokodimenzionalni podaci. U tekstualnim podacima (npr. reprezentacija dokumenata pomoću *bag-of-words* modela), svaki dokument se opisuje vektorom dimenzije nekoliko hiljada (po jedna pozicija u vektoru za svaku reč, vrednost na svakoj poziciji označava broj pojavljivanja reči u tekstu), gde je većina vrednosti jednaka nuli. Kako rastojanje Minkovskog uzima u obzir apsolutnu vrednost razlike na odgovarajućim pozicijama, na rastojanje između dva dokumenta utiču pozicije gde je ova vrednost različita od nule, odnosno gde jedan tekst sadrži neku reč a drugi je ne sadrži (ili ne sadrži isti broj puta). To znači da će tekstovi koji dele nekoliko ključnih reči, a većinu ostalih ne, imati veliko rastojanje (posebno kod euklidske metrike) iako semantički mogu biti slični.

Različita važnost atributa. Rastojanje Minkovskog tretira sve atribute jednako. Međutim, u mnogim primenama neki atributi su značajno važniji od drugih. Na primer, u medicinskoj dijagnostici pojedini biomarkeri mogu biti presudni, dok su drugi skoro nebitni. Bez odgovarajućeg ponderisanja, rastojanje može biti vođeno manje važnim atributima i dati pogrešnu procenu sličnosti.

Korelacije između atributa. Rastojanje Minkovskog pretpostavlja da su atributi nezavisni. Ako su atributi međusobno korelisani (npr. visina i težina), njihovi efekti se udvostručavaju, što ima uticaj na vrednost rastojanja. U takvim slučajevima prikladnije je koristiti Mahalanobisovo rastojanje koje uzima u obzir kovarijansu (detaljnije u potpoglavlju 3.5.3).

Lokalna struktura podataka. U nekim problemima, značaj atributa zavisi od lokalnog regiona prostora podataka. Na primer, pri klasterizaciji kupaca, za jednu grupu može biti ključan prihod, a za drugu učestalost kupovine. Globalna mera poput Minkovskog ne može da se prilagodi ovim lokalnim razlikama.

Ovo ilustruje važan princip: **izbor mere rastojanja treba da zavisi od toga šta u datoj primeni znači biti sličan**. Za guste numeričke podatke gde su svi atributi relevantni, euklidsko rastojanje je često dobar izbor. Međutim, za retke, visokodimenzionalne ili korelisane podatke, često su prikladnije alternativne mere (npr. kosinusna sličnost ili Mahalanobisovo rastojanje).

3.5.3 Mahalanobisovo rastojanje

Zamislite da merite zdravlje ljudi na osnovu dva parametra: **visinu** (u cm) i **težinu** (u kg). Ova dva atributa imaju dve važne osobine koje euklidsko rastojanje ignoriše:

1. **Različite varijanse.** Visina varira od recimo 150 do 200 cm (raspon ∼50), dok težina varira od 50 do 120 kg (raspon ∼70). Razlika od 5 kg u težini je uobičajena, ali razlika od 5 cm u visini je značajnija.

2. **Korelacija.** Visina i težina nisu nezavisne — viši ljudi su u proseku i teži. Ako neko ima 190 cm i 90 kg, to je očekivano. Ipak, 160 cm i 90 kg je *neobično*, jer ta kombinacija odstupa od uobičajenog obrasca.

Euklidsko rastojanje ne uzima u obzir ove specifičnosti već samo izračuna $\sqrt{(\Delta\text{visina})^2 + (\Delta\text{težina})^2}$ i tretira svaki centimetar visine isto kao svaki kilogram težine. Varijansu za atribut x_i čiji je prosek $\bar{x}_i$ računamo na sledeći način ($x_i^{(k)}$ je vrednost atributa u k-toj instanci skupa podataka):

$$\text{Var}(x_i) = \frac{1}{N}\sum_{k=1}^{N}(x_i^{(k)} - \bar{x}_i)^2$$

Šta je kovarijansa i zašto je bitna

Varijansa jednog atributa meri koliko se njegove vrednosti razlikuju od proseka. Ako svi ljudi imaju sličnu visinu, varijansa je mala; ako se visine mnogo razlikuju, varijansa je velika.

Kovarijansa dva atributa meri da li oni variraju *zajedno*:

$$\text{Cov}(x_i, x_j) = \frac{1}{N}\sum_{k=1}^{N}(x_i^{(k)} - \bar{x}_i)(x_j^{(k)} - \bar{x}_j)$$

- Cov > 0: kada jedan atribut raste, i drugi raste (visina i težina).
- Cov < 0: kada jedan raste, drugi opada (temperatura i prodaja zimskih jakni).
- Cov ≈ 0: atributi variraju nezavisno jedan od drugog.

Za dva atributa, **matrica kovarijansi** Σ je matrica 2×2:

$$\Sigma = \begin{pmatrix} \text{Var}(x_1) & \text{Cov}(x_1, x_2) \\ \text{Cov}(x_1, x_2) & \text{Var}(x_2) \end{pmatrix}$$

Na dijagonali su varijanse (koliko svaki atribut sam varira), a van dijagonale su kovarijanse (koliko variraju zajedno).

Duž pravca korelacije vs. ortogonalno na njega

Prethodni primer je poredio rastojanja od proseka. Sada pogledajmo nešto još upečatljivije: dva para osoba koji imaju **isto euklidsko rastojanje**, ali potpuno različito Mahalanobisovo.

- **Par A–B:** Osoba A (160 cm, 50 kg) i osoba B (180 cm, 70 kg).

  Razlika: +20 cm i +20 kg. Ove dve osobe leže *duž* prirodnog pravca korelacije — viša osoba je teža, što je potpuno tipičan obrazac. Razlika između njih prati ono što podaci "očekuju".

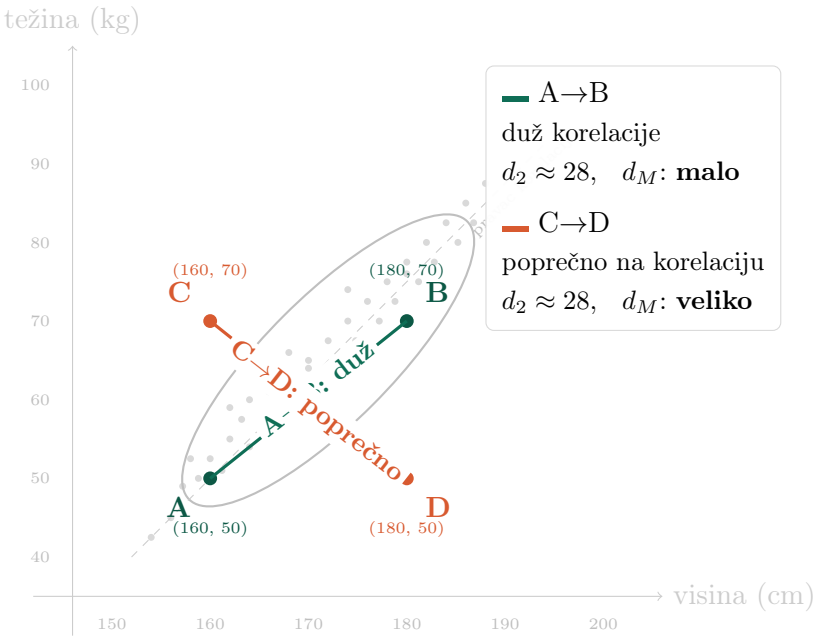
- **Par C–D:** Osoba C (160 cm, 70 kg) i osoba D (180 cm, 50 kg).

  Razlika: +20 cm, ali −20 kg. Ovde razlika ide *ortogonalno* na prirodni pravac: niža osoba je teža, a viša lakša. Takva kombinacija je netipična.

Euklidsko rastojanje u oba slučaja je isto:

$$d_2 = \sqrt{20^2 + 20^2} = \sqrt{800} \approx 28{,}3.$$

Ali Mahalanobisovo rastojanje uzima u obzir da razlika duž korelacije (A–B) nije iznenađujuća, dok je razlika poprečno na korelaciju (C–D) veoma neobična. Zato je $d_M(A, B) \ll d_M(C, D)$. Osobe A i B su sličnije od osoba C i D.



Slika 3.6: Dva para tačaka sa istim euklidskim rastojanjem (≈ 28,3). Par A–B (zeleno) leži duž pravca korelacije — Mahalanobisovo rastojanje je malo. Par C–D (narandžasto) ide poprečno na korelaciju — Mahalanobisovo rastojanje je veliko. Elipsa pokazuje oblik oblaka podataka.

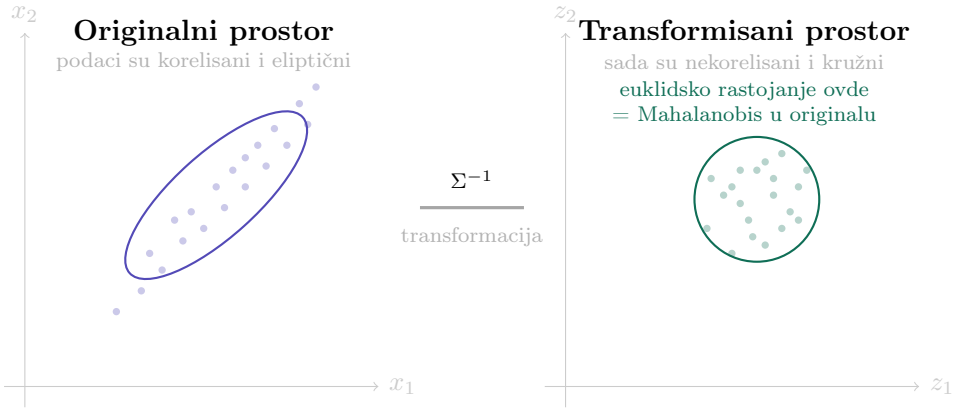
Intuicija jednom rečenicom. Mahalanobisovo rastojanje kaže: razlike koje *prate* obrazac u podacima (tačke su duž korelacije) su *jeftine* (česte, očekivane), dok razlike koje *krše* obrazac su *skupe* (retke, iznenađujuće). Elipsa na Slici 3.6 je izdužena duž pravca korelacije upravo zato — kretanje duž tog pravca *košta* manje.

Formula

Mahalanobisovo rastojanje se definiše kao:

$$\boxed{d_M(x,y) = \sqrt{(x-y)^T \Sigma^{-1} (x-y)}}$$

gde je Σ^{-1} inverzna matrica kovarijansi.



Slika 3.7: Mahalanobisovo rastojanje *ispravlja* prostor: korelisani, eliptični oblak podataka (levo) se transformiše u nekorelisani, kružni oblak (desno). U transformisanom prostoru, obično euklidsko rastojanje daje iste rezultate kao Mahalanobisovo u originalnom.

Matrica kovarijansi Σ mora biti **invertibilna** (što će biti slučaj ako je pozitivno definitna) da bi Σ^{-1} postojala. To neće biti ispunjeno ako je jedan atribut linearna kombinacija drugih (npr. ukupna cena = cena × količina, a sva tri su u podacima). U tom slučaju treba ukloniti redundantni atribut.

Mahalanobisovo rastojanje se svodi na euklidsko rastojanje *nakon* transformacije podataka tako da imaju jednake varijanse i nultu korelaciju.

3.6 Mere sličnosti za podatke sa binarnim atributima

Za binarne atribute često se posmatraju sledeće učestalosti:

$$\begin{aligned} f_{01} &= \text{broj atributa koji su 0 u } X \text{ i 1 u } Y, \\ f_{10} &= \text{broj atributa koji su 1 u } X \text{ i 0 u } Y, \\ f_{00} &= \text{broj atributa koji su 0 u } X \text{ i 0 u } Y, \\ f_{11} &= \text{broj atributa koji su 1 u } X \text{ i 1 u } Y. \end{aligned}$$

3.6.1 Jednostavni koeficijent uparivanja (SMC)

Jednostavni koeficijent uparivanja (*Simple Matching Coefficient*) jednak je:

$$\text{SMC} = \frac{f_{11} + f_{00}}{f_{01} + f_{10} + f_{11} + f_{00}}.$$

Ova mera jednako vrednuje i zajednička pojavljivanja jedinica i zajednička pojavljivanja nula.

Kada je pogodan? Pogodan je kada su i vrednost 1 i vrednost 0 podjednako informativne. Na primer, kod testa sa pitanjima *tačno/netačno*, dve osobe su slične i ako su na ista pitanja odgovorile tačno, ali i ako su na ista pitanja odgovorile netačno.

3.6.2 Žakarov koeficijent

Kada su binarni atributi *asimetrični*, tj. kada je prisustvo osobine informativnije od odsustva, koristi se **Žakarov koeficijent**:

$$J = \frac{f_{11}}{f_{01} + f_{10} + f_{11}}.$$

Ovde se parovi tipa 00 ignorišu.

Zašto je to važno? Kod transakcionih podataka u prodavnici, vektori koji prikazuju potrošačke korpe su uglavnom puni nula, jer većina proizvoda nije kupljena. Ako bismo brojali i nule na istim pozicijama, skoro sve korpe bi delovale veoma slično, što nije korisno. Zato je Žakarova mera prirodniji izbor.

Primer. Neka su

$$X = (1,0,0,0,0,0,0,0,0,0), \qquad Y = (0,0,0,0,0,0,1,0,0,1).$$

Tada je

$$f_{01} = 2, \quad f_{10} = 1, \quad f_{00} = 7, \quad f_{11} = 0.$$

Zato:

$$\text{SMC} = \frac{0+7}{2+1+7+0} = 0.7, \qquad J = \frac{0}{2+1+0} = 0.$$

Ovaj primer lepo pokazuje razliku: po SMC meri vektori deluju dosta slično zbog mnogo zajedničkih nula, dok po Žakarovoj meri nisu slični, jer nemaju nijedno zajedničko prisustvo jedinice.

3.6.3 Prošireni Žakarov koeficijent (Tanimoto)

Klasični Žakarov koeficijent radi samo sa binarnim atributima. Ali šta ako atributi imaju vrednosti poput 3, 0.7 ili 15? Na primer, umesto da znamo samo da li se neka reč pojavljuje u dokumentu, možemo znati *koliko puta* se pojavljuje. **Tanimotov koeficijent** generalizuje Žakarov koeficijent na proizvoljne vektore:

$$T(X,Y) = \frac{X \cdot Y}{\|X\|^2 + \|Y\|^2 - X \cdot Y}.$$

Intuicija. Formula se čita kao:

$$T(X,Y) = \frac{\text{preklapanje}}{\text{ukupna veličina (bez dvostrukog brojanja)}},$$

gde je $X \cdot Y$ mera preklapanja između dva vektora, a imenilac $\|X\|^2 + \|Y\|^2 - X \cdot Y$ predstavlja ukupnu "veličinu" oba vektora bez dvostrukog brojanja zajedničkog dela — analogno formuli za uniju skupova $|A \cup B| = |A| + |B| - |A \cap B|$.

Veza sa Žakarovim koeficijentom. Za binarne vektore važi $x_i^2 = x_i$, pa je $\|X\|^2 = f_{11} + f_{10}$, $\|Y\|^2 = f_{11} + f_{01}$, i $X \cdot Y = f_{11}$, odakle:

$$T(X,Y) = \frac{f_{11}}{(f_{11}+f_{10}) + (f_{11}+f_{01}) - f_{11}} = \frac{f_{11}}{f_{11}+f_{10}+f_{01}} = J(X,Y).$$

Primer. Neka su dati vektori frekvencija tri ključne reči u dva dokumenta:

	podatak	model	analiza
Dokument X	3	0	2
Dokument Y	1	4	1

Računamo: $X \cdot Y = 3{\cdot}1 + 0{\cdot}4 + 2{\cdot}1 = 5$, $\|X\|^2 = 9+0+4 = 13$, $\|Y\|^2 = 1+16+1 = 18$, pa je:

$$T(X,Y) = \frac{5}{13+18-5} = \frac{5}{26} \approx 0.19.$$

Sličnost je niska jer se X fokusira na *podatak* i *analiza*, dok se Y fokusira na *model* — preklapanje je malo u poređenju sa ukupnim sadržajem.

Kada se koristi? Tanimotov koeficijent je pogodan za **retke vektore sa nenegativnim komponentama**, gde nas zanima preklapanje sadržaja a ne zajedničko odsustvo (kao i kod klasičnog Žakara). Tipični primeri: vektori frekvencija reči u dokumentima, hemijski opisi molekula (*molecular fingerprints*), vektori ocena u sistemima preporuka.

3.7 Kosinusna sličnost

Ako su dokumenti predstavljeni vektorima

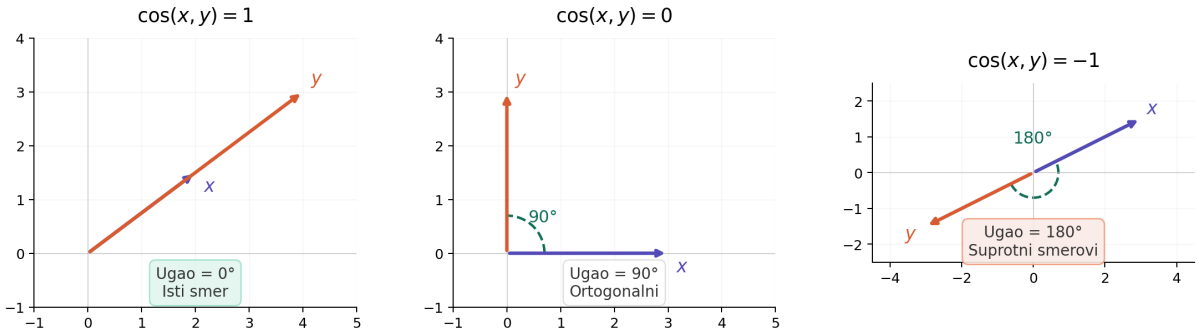
$$X = (x_1, x_2, \ldots, x_d), \qquad Y = (y_1, y_2, \ldots, y_d),$$

onda je **kosinusna sličnost** definisana kao

$$\cos(X,Y) = \frac{X \cdot Y}{\|X\|\,\|Y\|} = \frac{\sum_{i=1}^{d} x_i y_i}{\sqrt{\sum_{i=1}^{d} x_i^2}\sqrt{\sum_{i=1}^{d} y_i^2}}.$$

Kosinusna sličnost meri ugao između dva vektora:

- Ako je $\cos(x,y) = 1$, ugao je 0 — vektori imaju isti smer, što znači potpunu sličnost.
- Ako je $\cos(x,y) = 0$, ugao je 90 — vektori su ortogonalni i ne dele zajedničke vrednosti.
- Ako je $\cos(x,y) = -1$, ugao je 180 — vektori imaju suprotne smerove.



Slika 3.8: Geometrijska interpretacija kosinusne sličnosti: tri granična slučaja — isti smer ($\cos = 1$), ortogonalnost ($\cos = 0$) i suprotni smerovi ($\cos = -1$).

Razlika između kosinusne sličnosti i euklidskog rastojanja. Kosinusna sličnost i euklidsko rastojanje mogu dati suprotne zaključke o tome koji su objekti najsličniji. Razmotrimo tri vektora:

$$x = (2, 1), \qquad y = (4, 2), \qquad z = (2, 3).$$

Vektor y ima isti smer kao x (obe komponente su proporcionalno udvostručene), dok z ima drugačiji smer ali je bliži tački x u prostoru.

Po kosinusnoj sličnosti, x i y su potpuno slični jer imaju isti smer:

$$\cos(x, y) = 1.00, \qquad \cos(x, z) = 0.87.$$

Međutim, po euklidskom rastojanju, x i z su bliži jer im je manja udaljenost u prostoru:

$$d(x, y) = \sqrt{(2-4)^2 + (1-2)^2} = 2.24, \qquad d(x, z) = \sqrt{(2-2)^2 + (1-3)^2} = 2.00.$$

Dakle, kosinusna sličnost tretira y kao sličnijeg x-u (isti smer, samo duži vektor), dok euklidsko rastojanje tretira z kao sličnijeg (bliža tačka u prostoru). Ova razlika je ključna pri izboru mere: kosinusna sličnost je pogodna kada je važan *obrazac* vrednosti (na primer, raspodela reči u dokumentu), a euklidsko rastojanje kada je važna *apsolutna pozicija* u prostoru.

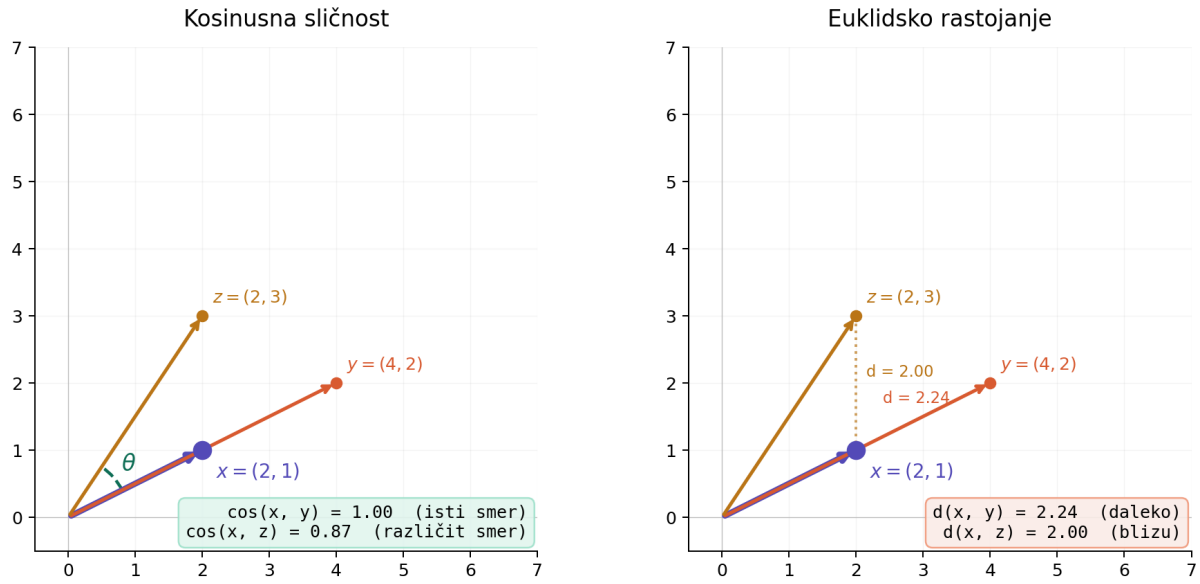
Ova mera je naročito pogodna za dokumente i druge retke vektore. Razlog je sledeći: kada se dokumenti predstave kao vektori frekvencija reči, većina komponenti je jednaka nuli jer svaki dokument sadrži samo mali deo svih mogućih reči. Ako bismo koristili meru koja uzima u obzir zajedničke nule, skoro svi dokumenti bi delovali veoma slično — baš kao što smo videli kod SMC mere za binarne podatke. Kosinusna sličnost izbegava ovaj problem jer kod skalarnog proizvoda $x \cdot y$ doprinos imaju samo pozicije gde su obe vrednosti različite od nule.

Primer. Neka su data dva dokumenta predstavljena vektorima frekvencija reči:

$$d_1 = (3, 2, 0, 5, 0, 0, 0, 2, 0, 0), \qquad d_2 = (1, 0, 0, 0, 0, 0, 0, 1, 0, 2).$$

Vektor d_1 ima nenulte vrednosti na četiri pozicije, a d_2 na tri pozicije. Od ukupno deset reči, samo na dvema pozicijama (prvoj i osmoj) oba dokumenta imaju nenulte vrednosti — samo te pozicije doprinose skalarnom proizvodu:

$$d_1 \cdot d_2 = 3 \cdot 1 + 2 \cdot 1 = 5.$$



Slika 3.9: Ista tri vektora, dve različite mere: kosinusna sličnost gleda ugao (levo), euklidsko rastojanje gleda udaljenost između tačaka (desno). Vektor y je kosinusno najsličniji x-u, ali euklidski najudaljeniji.

Svi ostali sabirci su jednaki nuli jer je bar jedan od činilaca nula. Norme vektora su:

$$\|d_1\| = \sqrt{3^2 + 2^2 + 5^2 + 2^2} = \sqrt{42} \approx 6.48,$$

$$\|d_2\| = \sqrt{1^2 + 1^2 + 2^2} = \sqrt{6} \approx 2.45,$$

pa je

$$\cos(d_1, d_2) \approx \frac{5}{6.48 \cdot 2.45} \approx 0.315.$$

Vrednost 0.315 ukazuje na umerenu sličnost — dokumenti dele neke zajedničke reči, ali se u velikoj meri razlikuju.

Kosinusna sličnost je **invarijantna na skaliranje**. Ako se sve frekvencije reči u dokumentu pomnože istom konstantom, kosinusna sličnost ostaje ista. To znači da, na primer, dva dokumenta sa identičnom raspodelom reči ali različitom dužinom (jedan ima duplo više reči od drugog) imaće kosinusnu sličnost jednaku 1. Zato je ova mera pogodna kada je važniji obrazac raspodele termina nego apsolutna dužina dokumenta.

Iako se u nekim materijalima govori o kosinusnom ili Žakarovom *rastojanju*, izrazi navedeni ovde predstavljaju mere sličnosti, dok se odgovarajuće mere različitosti dobijaju odgovarajućom transformacijom.

3.8 Korelacija

Korelacija dva objekta je mera linearnog odnosa između njihovih atributa. Preciznije, Pirsonov koeficijent korelacije između dva objekta (vektora) x i y se definiše na sledeći način:

$$\rho_{xy} = \frac{\text{cov}(x, y)}{\sigma_x \sigma_y}.$$

pri čemu je kovarijansa definisana na sledeći način:

$$\text{cov}(x, y) = \frac{1}{n-1}\sum_{k=1}^{n}(x_k - \bar{x})(y_k - \bar{y}),$$

a standardna devijacija na sledeći način

$$\sigma_x = \sqrt{\frac{1}{n-1}\sum_{k=1}^{n}(x_k - \bar{x})^2}, \qquad \bar{x} = \frac{1}{n}\sum_{k=1}^{n} x_k,$$

Vrednost korelacije je u intervalu $[-1, 1]$:

- $\rho = 1$ znači savršeno pozitivan linearan odnos;
- $\rho = -1$ znači savršeno negativan linearan odnos;
- $\rho = 0$ znači da nema *linearnog* odnosa, ali ne isključuje nelinearnu zavisnost.

Ako je korelacija jednaka 1 ili -1, tada postoji linearna veza oblika

$$x_k = ay_k + b.$$

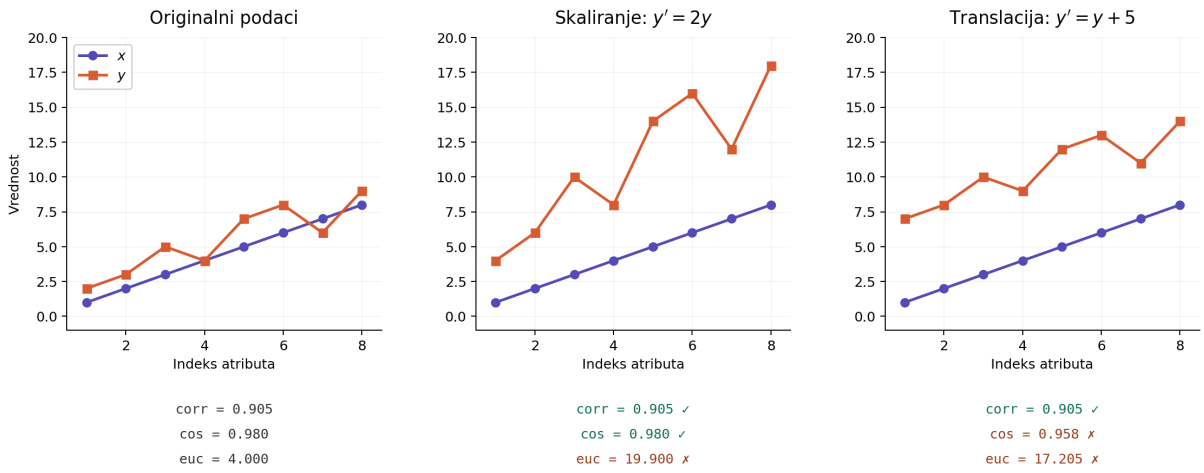
Na primer, savršenu pozitivnu korelaciju imaju $x = (2, 0, 4, 6, 1)$ i $y = (4, 0, 8, 12, 2)$, dok savršenu negativnu korelaciju imaju $x = (2, 0, 4, 6, 1)$ i $y = (-4, 0, -8, -12, -2)$.

Pirsonov koeficijent korelacije meri samo linearnu zavisnost. Na primer, objekti $x = (-3, -2, -1, 0, 1, 2, 3)$ i $y = (9, 4, 1, 0, 1, 4, 9)$ imaju savršenu nelinearnu zavisnost $y_k = x_k^2$, ali je njihova Pirsonova korelacija jednaka 0. Razlog je što je parabola simetrična: za negativne vrednosti x vrednost y opada, a za pozitivne raste, pa se ti efekti međusobno poništavaju.

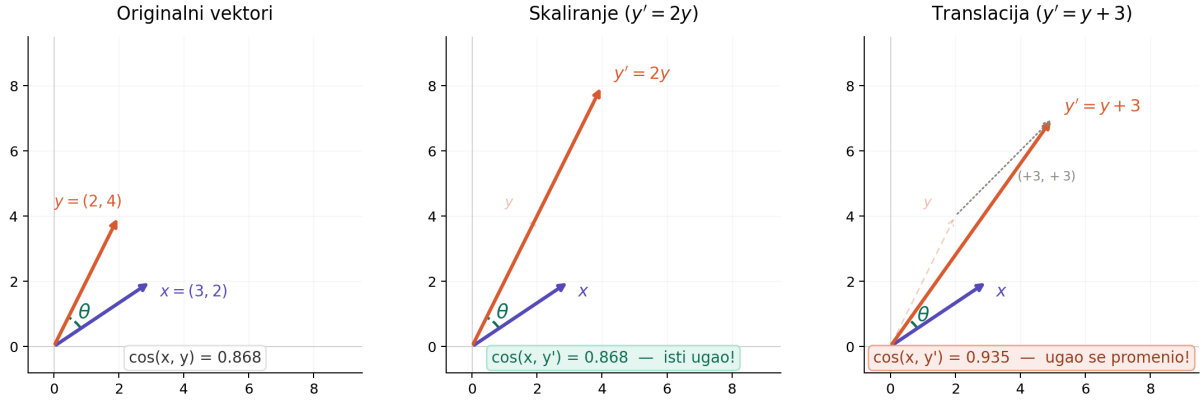
U ovakvim slučajevima može se primeniti **Spirmanov koeficijent korelacije**, koji ne meri linearnu, već *monotonu* zavisnost, odnosno, da li veće vrednosti jednog atributa konzistentno odgovaraju većim (ili manjim) vrednostima drugog. Postupak je sledeći: svaki atribut se zameni svojim rangom (rednim brojem nakon sortiranja), a zatim se nad tim rangovima izračuna Pirsonov koeficijent korelacije.

Važno poređenje. Važna razlika između korelacije, kosinusne sličnosti i rastojanja Minkovskog tiče se njihove osetljivosti na *skaliranje* i *translaciju* podataka. Skaliranje podrazumeva množenje svih vrednosti konstantom ($x \to c \cdot x$), a translacija dodavanje konstante ($x \to x + b$).

- Korelacija je invarijantna i na skaliranje i na translaciju. Ako sve vrednosti jednog objekta pomnožimo ili im dodamo konstantu, korelacija se neće promeniti.
- Kosinusna sličnost je invarijantna na skaliranje, ali ne i na translaciju. Množenje konstantom ne menja ugao između vektora, ali dodavanje konstante menja.
- Rastojanje Minkovskog nije invarijantno ni na jedno ni na drugo. I množenje i dodavanje konstante menjaju izračunato rastojanje.



Slika 3.10: Uticaj skaliranja i translacije na tri mere blizine. Oznake ✓ i × označavaju da li se vrednost mere promenila nakon transformacije.



Slika 3.11: Geometrijska intuicija: skaliranje ne menja ugao između vektora (kosinusna sličnost ostaje ista), dok translacija menja smer vektora i time menja ugao.

Na primer, neka su dati vektori $x = (1, 2, 3, 4, 5, 6, 7, 8)$ i $y = (2, 3, 5, 4, 7, 8, 6, 9)$. Ako sve vrednosti vektora y pomnožimo sa 2, korelacija i kosinusna sličnost ostaju iste, ali se euklidsko rastojanje menja. Ako pak svim vrednostima vektora y dodamo konstantu 5, samo korelacija ostaje nepromenjena, dok se i kosinusna sličnost i euklidsko rastojanje menjaju (slike 3.10 i 3.11).

Zato je korelacija često pogodna za vremenske serije i situacije kada su važni trendovi, a ne apsolutni nivoi merenja — dva senzora koji mere istu veličinu u različitim jedinicama ili sa različitim početnim pomerajem imaće istu korelaciju.

3.9 Mere sličnosti kategoričkih podataka

Za kategoričke atribute sličnost dva objekta

$$X = (x_1, x_2, \ldots, x_n), \qquad Y = (y_1, y_2, \ldots, y_n)$$

može se definisati kao zbir sličnosti po pojedinačnim atributima:

$$\mathrm{Sim}(X, Y) = \sum_{i=1}^{n} S(x_i, y_i).$$

Najjednostavniji slučaj je *prosto podudaranje*:

$$S(x_i, y_i) = \begin{cases} 1, & \text{ako } x_i = y_i, \\ 0, & \text{inače.} \end{cases}$$

Ovaj pristup je jednostavan, ali ne uzima u obzir koliko je neka vrednost česta ili retka u skupu podataka. Ako se dve instance podudaraju po vrednosti koju ima 90% slogova, to je mnogo manje informativno nego podudaranje po vrednosti koju ima samo 2% slogova.

3.9.1 Uzimanje frekvencije u obzir

Neka je $p_i(v)$ relativna frekvencija vrednosti v za i-ti atribut, tj. udeo slogova u kojima i-ti atribut uzima vrednost v:

$$p_i(v) = \frac{\text{broj slogova kod kojih je } i\text{-ti atribut} = v}{\text{ukupan broj slogova}}, \qquad p_i(v) \in [0,\, 1].$$

Inverzna učestalost (eng. inverse frequency). Što je slog ređi, podudaranje vredi više:

$$S(x_i, y_i) = \begin{cases} \dfrac{1}{\big(p_i(x_i)\big)^2}, & \text{ako } x_i = y_i, \\ 0, & \text{inače.} \end{cases}$$

Varijanta sa umerenom korekcijom. Kao i u slučaju inverzne frekvencije, veća vrednost sličnosti dodeljuje se poklapanju kada je vrednost atributa retka. Kada je vrednost atributa česta, sličnost je manja, ali ne toliko kao kod inverzne frekvencije (otud ova mera predstavlja varijantu sa *umerenom korekcijom*).

$$S(x_i, y_i) = \begin{cases} 1 - \dfrac{p_i(x_i)}{2}, & \text{ako } x_i = y_i, \\ 0, & \text{inače.} \end{cases}$$

3.9.2 Primer

Posmatrajmo skup od 100 studenata opisan sa dva kategorička atributa:

Atribut	**Vrednost**	**Frekvencija**
Smer (i=1)	Informatika	$p_1(\text{Inf}) = 0{,}80$
	Matematika	$p_1(\text{Mat}) = 0{,}20$
Stipendija (i=2)	Da	$p_2(\text{Da}) = 0{,}05$
	Ne	$p_2(\text{Ne}) = 0{,}95$

Upoređujemo tri para studenata:

$$A = (\text{Inf},\ \text{Da}), \quad B = (\text{Inf},\ \text{Da}), \quad C = (\text{Inf},\ \text{Ne}), \quad D = (\text{Mat},\ \text{Da}).$$

1. Prosto podudaranje. A i B su identični, pa je:

$$\mathrm{Sim}(A,B) = S(\mathrm{Inf},\mathrm{Inf}) + S(\mathrm{Da},\mathrm{Da}) = 1+1 = 2.$$

2. Inverzna učestalost. Za par (A,B):

$$\mathrm{Sim}(A,B) = \frac{1}{(0{,}80)^2} + \frac{1}{(0{,}05)^2} = \frac{1}{0{,}64} + \frac{1}{0{,}0025} = 1{,}5625 + 400 = 401{,}5625.$$

Podudaranje po stipendiji (retka vrednost "Da", samo 5%) dominira rezultatom — upravo to je poenta: retko podudaranje je veoma informativno.

Za par (A,C) — podudaraju se samo po smeru:

$$\mathrm{Sim}(A,C) = \frac{1}{(0{,}80)^2} + 0 = 1{,}5625.$$

Za par (A,D) — podudaraju se samo po stipendiji:

$$\mathrm{Sim}(A,D) = 0 + \frac{1}{(0{,}05)^2} = 400.$$

Dakle mera korektno prepoznaje da je podudaranje po retkoj stipendiji (Sim = 400) daleko značajnije od podudaranja po čestom smeru (Sim = 1,56).

3. Umerena korekcija. Za par (A,B):

$$\mathrm{Sim}(A,B) = \left(1 - \frac{0{,}80}{2}\right) + \left(1 - \frac{0{,}05}{2}\right) = 0{,}60 + 0{,}975 = 1{,}575.$$

Ovde su obe komponente bliže 1, ali se i dalje vidi razlika: podudaranje po stipendiji (0,975) vredi više nego podudaranje po smeru (0,60).

Zaključak. Prosto podudaranje tretira sve atribute jednako. Mere zasnovane na frekvenciji daju veću težinu retkim podudaranjima, čime bolje razlikuju zaista slične objekte od onih koji se slažu samo po trivijalno čestim osobinama.

3.10 Mere sličnosti dokumenata

Kod dokumenata je obično važno da se sličnost zasniva na zajedničkim rečima. Međutim, nisu sve reči podjednako informativne. Vrlo česte reči imaju manju diskriminativnu moć od reči koje se javljaju u malom broju dokumenata. Zato se uvodi *inverzna frekvencija dokumenta*:

$$\mathrm{id}_i = \log\left(\frac{n}{n_i}\right),$$

gde je n ukupan broj dokumenata, a n_i broj dokumenata u kojima se javlja reč i.

Da bi se smanjio uticaj vrlo čestih reči, mogu se koristiti transformacije frekvencije koje sabijaju velike vrednosti:

$$f(x_i) = \sqrt{x_i}, \qquad f(x_i) = \log(x_i).$$

Treba primetiti da je funkcija $\log(x_i)$ nedefinisana za $x_i = 0$, pa se u praktičnim implementacijama često koristi varijanta

$$f(x_i) = \log(1 + x_i).$$

Normalizovana frekvencija i-te reči može se definisati kao

$$h(x_i) = f(x_i) \cdot \mathrm{id}_i.$$

Na osnovu ovih težinskih vrednosti mogu se računati kosinusna i Žakarova sličnost dokumenata:

$$\cos(X, Y) = \frac{\sum_{i=1}^{d} h(x_i)h(y_i)}{\sqrt{\sum_{i=1}^{d} h(x_i)^2}\sqrt{\sum_{i=1}^{d} h(y_i)^2}},$$

$$J(X, Y) = \frac{\sum_{i=1}^{d} h(x_i)h(y_i)}{\sum_{i=1}^{d} h(x_i)^2 + \sum_{i=1}^{d} h(y_i)^2 - \sum_{i=1}^{d} h(x_i)h(y_i)}.$$

Ovakvo ponderisanje je praktično veoma važno, jer razlikuje ključne reči od čestih, ali slabo informativnih termina.

3.11 Mere sličnosti za podatke sa kvantitativnim i kategoričkim atributima

U realnim skupovima podataka atributi su često mešoviti. Neki su numerički (npr. starost, prihod), a neki kategorički (npr. pol, grad). Postavlja se pitanje kako izračunati sličnost dva objekta kada njihovi atributi nisu istog tipa.

Pristup po atributima. Osnovna ideja je da se sličnost računa *za svaki atribut posebno*, a zatim se pojedinačne sličnosti agregiraju u ukupnu. Za svaki atribut k definiše se:

- $s_k(X, Y) \in [0, 1]$, sličnost objekata X i Y po k-tom atributu (izračunata na način prikladan tipu tog atributa),
- $\delta_k \in \{0, 1\}$, indikator da li k-ti atribut uopšte treba uzeti u obzir.

Indikator δ_k se postavlja na 0 u dva slučaja:

1. jedan od objekata ima nedostajuću vrednost za k-ti atribut, ili
2. k-ti atribut je asimetričan binarni atribut i oba objekta imaju vrednost 0 (jer se slaganje u "odsustvu" osobine ne smatra informativnim, kao što je objašnjeno u prethodnom odeljku).

U svim ostalim slučajevima važi $\delta_k = 1$. Ukupna sličnost se tada računa kao ponderisani prosek:

$$\mathrm{Sim}(X, Y) = \frac{\sum_{k=1}^{n} \delta_k \, s_k(X, Y)}{\sum_{k=1}^{n} \delta_k}.$$

Imenilac $\sum_k \delta_k$ broji samo atribute koji su uzeti u obzir, tako da nedostajuće vrednosti i asimetrična poklapanja u nuli ne utiču na rezultat.

Jednostavniji pristup: linearna kombinacija. Ako se objekat razloži na numerički deo X_n i kategorički deo X_c, sličnost se može izračunati i kao linearna kombinacija dve odvojene mere:

$$\mathrm{Sim}(X, Y) = \lambda \cdot \mathrm{NumSim}(X_n, Y_n) + (1 - \lambda) \cdot \mathrm{CatSim}(X_c, Y_c),$$

gde parametar $\lambda \in [0, 1]$ određuje relativni značaj numeričkog u odnosu na kategorički deo. Ovaj pristup je konceptualno jednostavniji, ali je manje fleksibilan jer ne omogućava tretman pojedinačnih atributa, na primer, ne može da ignoriše konkretne nedostajuće vrednosti ili asimetrične atribute.

Stoga se u praksi preporučuje prvi pristup: izračunati sličnost za svaki atribut posebno u opsegu $[0, 1]$, a zatim ih agregirati pomoću formule sa indikatorima δ_k, čime se dobija fleksibilan postupak za rad sa heterogenim podacima.

3.12 Mere sličnosti diskretnih podataka

3.12.1 Edit rastojanje

Za diskretne nizove, kao što su niske karaktera, prirodna mera različitosti je **edit rastojanje**, odnosno cena transformacije jedne niske u drugu pomoću operacija brisanja, umetanja i zamene.

Ako su

$$X = (x_1, x_2, \ldots, x_m), \qquad Y = (y_1, y_2, \ldots, y_n),$$

onda se za prvih i simbola iz X i prvih j simbola iz Y definiše:

$$\mathrm{Edit}(i, j) = \min \begin{cases} \mathrm{Edit}(i-1, j) + \text{cena brisanja}, \\ \mathrm{Edit}(i, j-1) + \text{cena umetanja}, \\ \mathrm{Edit}(i-1, j-1) + I_{ij} \cdot \text{cena zamene}, \end{cases}$$

gde je

$$I_{ij} = \begin{cases} 0, & \text{ako } x_i = y_j, \\ 1, & \text{ako } x_i \neq y_j. \end{cases}$$

Da bi rekurzija bila potpuno definisana, uvode se i bazni uslovi:

$$\mathrm{Edit}(0, 0) = 0, \qquad \mathrm{Edit}(i, 0) = i \cdot c_{\text{brisanje}}, \qquad \mathrm{Edit}(0, j) = j \cdot c_{\text{umetanje}}.$$

Ova mera je korisna kada želimo da procenimo koliko se jedna sekvenca može "pretvoriti" u drugu uz minimalan trošak. Na primer, može se posmatrati transformacija niske `abababababab` u `bababababa`.

3.12.2 Sličnost zasnovana na najdužoj zajedničkoj podniski

Druga važna mera jeste sličnost zasnovana na *najdužoj zajedničkoj podniski* (*Longest Common SubSequence*, LCSS). Za prefikse dužine i i j niski x i y, LCSS definišemo na sledeći način:

$$\mathrm{LCSS}(i, j) = \begin{cases} \mathrm{LCSS}(i-1, j-1) + 1, & \text{ako } x_i = y_j, \\ \max\{\mathrm{LCSS}(i-1, j), \mathrm{LCSS}(i, j-1)\}, & \text{ako } x_i \neq y_j. \end{cases}$$

Uz bazne uslove važi

$$\mathrm{LCSS}(i, 0) = 0, \qquad \mathrm{LCSS}(0, j) = 0.$$

Veća vrednost znači veću sličnost. Međutim, pošto broj podudaranja zavisi i od dužine niske, u praksi se često koristi i normalizacija.

Primer. Za niske

`agbfcgdjhe` i `afbgchdise`

jedna najduža zajednička podniska je

`agcde`,

pa je odgovarajuća LCSS vrednost jednaka 5.

3.13 Kako izabrati odgovarajuću meru blizine

Izbor odgovarajuće mere nije univerzalan i zavisi od tipa podataka i cilja analize.

- Za **guste numeričke podatke** često su pogodne metričke mere rastojanja, kao što su euklidsko i druga rastojanja Minkovskog.
- Za **retke podatke**, naročito kada ima mnogo parova 00, pogodnije su mere koje ignorišu zajedničke nule, kao što su Žakardova i kosinusna sličnost.
- **Kosinusna sličnost** je pogodna kada je važan obrazac raspodele, a ne apsolutna veličina vektora, na primer kod dokumenata.
- **Korelacija** je pogodna kada su važne promene i trendovi, a želimo neosetljivost na pomeranje i skaliranje, na primer kod vremenskih serija.
- **Mahalanobisovo rastojanje** je korisno kada su atributi korelisani i imaju različite varijanse.

Pored teorijskih osobina, važni su i praktični razlozi:

- ponekad je u određenoj oblasti već ustaljena upotreba neke konkretne mere;
- izbor može biti ograničen implementacijom konkretnog algoritma ili softverskog paketa;
- efikasnost računanja može favorizovati mere koje imaju osobine poput nejednakosti trougla.

Zato se prava mera blizine bira tako da odgovara i *prirodi podataka* i *značenju sličnosti u konkretnoj primeni.*

3.14 Zaključak

Mere sličnosti i različitosti predstavljaju osnovu velikog broja metoda u istraživanju podataka. Ne postoji jedna univerzalno najbolja mera. Izbor zavisi od toga da li su podaci numerički, binarni, kategorički, tekstualni ili mešoviti, kao i od toga da li želimo da uvažimo samo zajednička prisustva osobina, linearnu povezanost, raspodelu podataka, lokalnu gustinu ili cenu transformacije jedne sekvence u drugu.

Zbog toga je pri svakoj analizi neophodno prvo razumeti *šta tačno znači da su dva objekta slična* u datom problemu, a tek zatim odabrati odgovarajuću meru.

Glava 4

Priprema podataka

U analizi podataka i mašinskom učenju, kvalitet rezultata ne zavisi isključivo od izbora algoritma, već ga u velikoj meri određuje kvalitet ulaznih podataka. U praksi, podaci gotovo nikada ne dolaze u idealnom obliku: prikupljeni su iz različitih izvora, zapisani u heterogenim formatima, a često sadrže greške, nedostajuće vrednosti ili duplirane zapise. Zato je **priprema podataka** jedan od ključnih koraka u svakom analitičkom projektu.

Tipični problemi uključuju nekonzistentne formate između izvora, nedostajuće vrednosti (nepopunjena polja), heterogene tipove atributa u jednom skupu podataka, veliku dimenzionalnost koja vodi ka preprilagođavanju modela, kao i prisustvo šuma i odstupajućih vrednosti (*outlier*-a).

Ciljevi pripreme podataka su:

- **Izdvajanje korisnih karakteristika** iz sirovih podataka,
- **Transformacija tipova** u oblik kompatibilan sa algoritmima,
- **Čišćenje i konsolidacija** — uklanjanje duplikata, ispravljanje grešaka, objedinjavanje izvora,
- **Skaliranje i normalizacija** atributa na uporedive opsege,
- **Redukcija dimenzionalnosti** bez značajnog gubitka informacija.

Prema raznim istraživanjima, priprema podataka oduzima 60–80% ukupnog vremena u analitičkom projektu, jer algoritmi očekuju čist, konzistentan i odgovarajuće predstavljen ulaz.

4.1 Transformacija i reprezentacija podataka

Atributi (*features*) su merljive osobine objekata koji služe kao ulaz za algoritme istraživanja podataka. Kod strukturiranih podataka oni su eksplicitno zadati kolonama tabele. Međutim, kod nestrukturiranih podataka kao što su slike, tekst, zvučni zapis, veb logovi, potrebna je **ekstrakcija atributa**: proces kojim se iz sirovih podataka izdvajaju informativni atributi pogodni za dalju obradu.

Vrsta atributa zavisi od izvora podataka. Kod slika to mogu biti ivice ili teksture; kod teksta frekvencije termina, TF-IDF vektori ili *word embeddings*; kod vremenskih serija trendovi, sezonske komponente i autokorelacije; kod veb logova redosled aktivnosti korisnika ili vreme zadržavanja na stranici.

4.1.1 Zašto je transformacija tipova neophodna

Mnogi algoritmi i standardni alati prihvataju samo određene tipove ulaza, najčešće numeričke matrice. Kada skup podataka istovremeno sadrži numeričke, kategorijske, tekstualne i sekvencijalne atribute, neophodno je obezbediti **zajedničku reprezentaciju** pogodnu za obradu.

Transformacije se uvode zbog: kompatibilnosti sa algoritmom, mogućnosti primene standardnih alata, lakše interpretacije i vizualizacije, kao i smanjenja složenosti modela.

Pri tome treba imati na umu da svaka transformacija nosi rizik od **gubitka informacije**. Na primer, diskretizacijom neprekidnog atributa (zamena tačne starosti kategorijama „mlad“, „sredovečan“, „star“) gubi se finoća originalnih vrednosti pa tako osobe od 29 i 35 godina završavaju u istoj grupi. Stoga je potrebno pažljivo odvagati prednosti transformacije (kompatibilnost, jednostavnost) i njenu cenu (gubitak granularnosti).

4.1.2 Glavni oblici transformacije i reprezentacije

U pripremi podataka javlja se nekoliko ključnih transformacija, od kojih svaka prevodi jedan tip podataka u oblik pogodniji za algoritamsku obradu:

- **Diskretizacija** — prevođenje neprekidnih atributa u kategorijske,
- **Binarizacija** — kodiranje kategorijskih atributa skupom binarnih atributa,
- **Vektorizacija teksta** — pretvaranje tekstualnih dokumenata u numeričke vektore,
- **Kodiranje sekvenci** — transformacija diskretnih nizova u numeričke reprezentacije,
- **Obrada vremenskih serija** — prevođenje vremenskih serija u diskretne nizove ili numeričke vektore.

4.1.3 Diskretizacija

Definicija i motivacija

Diskretizacija je postupak kojim se neprekidni atribut pretvara u kategorijski tako da umesto proizvoljne realne vrednosti, atribut nakon diskretizacije pripada jednoj od konačno mnogo kategorija. Ovaj postupak je koristan kada algoritam bolje radi sa kategorijskim ulazima, kada je cilj interpretabilniji model, ili kada se grade pravila pridruživanja i klasifikatori zasnovani na intervalima.

Opšti postupak se sastoji iz tri koraka: (1) odabere se broj kategorija n, (2) interval vrednosti atributa podeli se na n podintervala, i (3) sve vrednosti iz istog podintervala mapiraju se na istu kategoriju. Pritom, oznake dodeljene intervalima (npr. A, B, C) jesu simboli, ne numeričke veličine, ali budući da potiču od uređenog numeričkog atributa, intervali kojima odgovaraju najčešće imaju prirodno uređenje.

Jednaka širina intervala

Najjednostavniji pristup jeste podela opsega vrednosti na intervale jednake širine. Ako atribut uzima vrednosti u intervalu $[a, b]$ i želimo n intervala, širina svakog je:

$$h = \frac{b - a}{n},$$

pa intervali imaju oblik:

$$[a,\, a{+}h), \quad [a{+}h,\, a{+}2h), \quad \ldots, \quad [a{+}(n{-}1)h,\, b].$$

Prednost ovog pristupa je jednostavnost implementacije i lake interpretacije. Glavni nedostatak je što ne uzima u obzir distribuciju podataka — ako su vrednosti zgusnutene u jednom delu opsega, pojedini intervali mogu biti gotovo prazni, dok drugi sadrže većinu podataka.

Jednaki log-intervali

Kada atribut ima veoma širok opseg (na primer, od 1 do milion) ili se prirodno menja multiplikativno (na primer, (1, 2, 4, 8, 16, ...) što se dešava kod podataka o prihodima, veličini fajlova, broju pregleda veb stranica), pogodnije je raditi sa logaritamskom skalom. Granice intervala tada rastu geometrijski:

$$[a,\, a\alpha), \quad [a\alpha,\, a\alpha^2), \quad [a\alpha^2,\, a\alpha^3), \quad \ldots, \qquad \alpha > 1,$$

tako da je razlika granica podintervala $\log(b) - \log(a)$ jednaka. Ovaj pristup ima smisla samo za strogo pozitivne vrednosti atributa.

Primer: broj pregleda sadržaja

Razmotrimo atribut koji predstavlja broj pregleda nekog sadržaja i uzima vrednosti u opsegu od 1 do 1000. Pošto se radi o veličini koja prirodno raste multiplikativno (za sadržaj koji ima veći broj pregleda možemo očekivati još više pregleda nego za sadržaj koji nema), pogodnije je koristiti logaritamsku diskretizaciju.

Neka je početna vrednost $a = 1$ i faktor rasta $\alpha = 10$. Tada su intervali definisani kao:

$$[1, 10), \quad [10, 100), \quad [100, 1000).$$

Uočimo da ovi intervali nisu jednake širine u originalnoj skali (njihove dužine su 9, 90 i 900), ali jesu jednake širine u logaritamskoj skali. Naime:

$$\log_{10}(1) = 0, \quad \log_{10}(10) = 1, \quad \log_{10}(100) = 2, \quad \log_{10}(1000) = 3,$$

pa su razlike između uzastopnih granica konstantne:

$$\log_{10}(10) - \log_{10}(1) = \log_{10}(100) - \log_{10}(10) = \log_{10}(1000) - \log_{10}(100) = 1.$$

Primetimo da logaritamska skala velike vrednosti sabija mnogo više nego male: 100 se preslikava u 2, 1000 u 3, dok se 10 preslikava u 1.

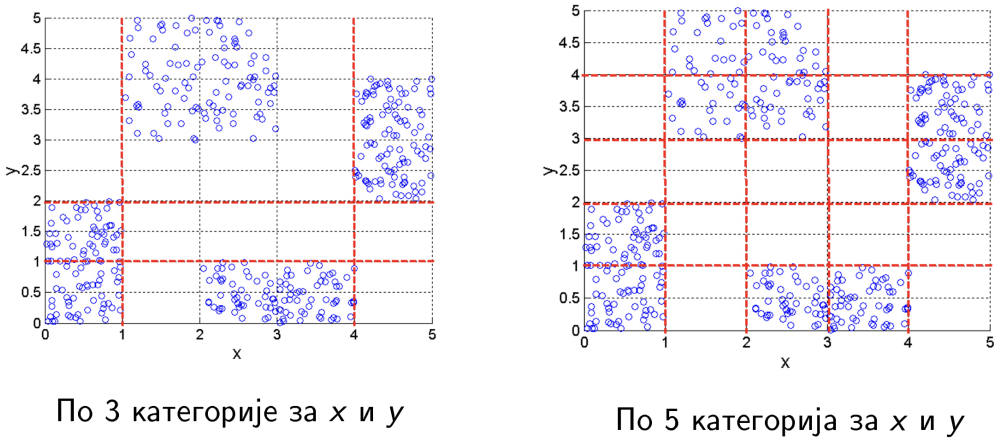
Jednak broj elemenata po intervalu

Treći pristup bira granice intervala tako da svaki sadrži približno isti broj elemenata. Za k posmatranja i n željenih intervala, svaki interval treba da obuhvati približno k/n elemenata. Postupak je sledeći: vrednosti se sortiraju, sortirani niz se podeli na n delova jednake brojnosti, a granice intervala određuju se prema pozicijama u tom nizu.

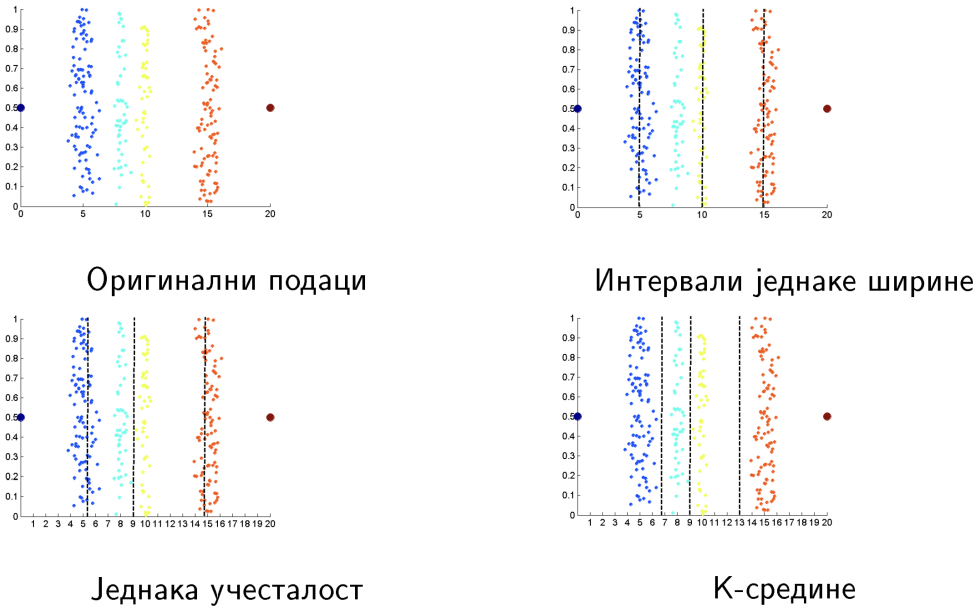
Ovaj metod izbegava problem praznih intervala i bolje prati raspodelu podataka, ali intervali mogu imati veoma različite širine, a postupak je osetljiv na duplikate i granične vrednosti.

Izbor broja klasa

Ako je broj klasa unapred poznat, diskretizacija se direktno prilagođava toj informaciji. Ako nije, mogu se koristiti heuristike za izbor broja grupa, kao i algoritmi klasterovanja (o kojima će biti reči kasnije) poput metode k-sredina (*k-means*) za formiranje samih intervala (slike 4.1 4.2)



Slika 4.1: Primer diskretizacije kada je broj klasa poznat. Manji broj klasa daje grublju ali jednostavniju reprezentaciju, dok veći broj omogućava finije razlikovanje, ali može dovesti do složenijeg modela.



Slika 4.2: Primer diskretizacije kada broj klasa nije poznat. Intervali jednake širine zanemaruju raspodelu elemenata, metoda jednake učestalosti prilagođava se gustini podataka, dok klasterovanje metodom *k-means* prati stvarnu strukturu skupa.

Ovi primeri ilustruju da izbor metode diskretizacije nije tehnički detalj, već odluka koja može značajno uticati na kvalitet dalje analize.

4.1.4 Binarizacija

Binarizacija je transformacija atributa u binarne vrednosti (0 ili 1). Kod neprekidnih atributa to obično podrazumeva uvođenje praga, dok se kod kategorijskih atributa najčešće primenjuje *one-hot* kodiranje koje podrazumeva formiranje jednog binarnog atributa za svaku moguću kategoriju. Čest pristup kombinuje oba koraka:

$$\text{neprekidni} \rightarrow \text{kategorijski} \rightarrow \text{skup binarnih atributa.}$$

Ako kategorijski atribut ima n mogućih vrednosti, formira se n binarnih atributa. Za svaki zapis, tačno jedan od njih ima vrednost 1, a ostali 0. Na primer, atribut `boja` sa vrednostima $\{\text{crvena}, \text{plava}, \text{zelena}\}$ predstavlja se sa tri binarna atributa x_{crvena}, x_{plava}, x_{zelena}, pa se vrednost `plava` kodira kao $(0, 1, 0)$.

Prednost ovog pristupa je što omogućava primenu numeričkih algoritama nad kategorijskim podacima, bez uvođenja lažnog uređenja među kategorijama. Glavni nedostatak je povećanje dimenzionalnosti — kod atributa sa mnogo različitih vrednosti nastaju veoma velike, doduše retke matrice, što može biti problematično za neke algoritme.

4.1.5 Tekstualni podaci u numeričkom obliku

Tekstualni podaci nisu direktno pogodni za većinu algoritama analize podataka, pa se dokumenti najčešće predstavljaju kao numerički vektori. Najjednostavniji pristup je model *vreće reči* (*bag-of-words*), gde svaka komponenta vektora odgovara učestalosti određene reči u dokumentu. Problem sa modelom vreće reči je što rezultujući vektori imaju onoliko dimenzija koliko ima različitih reči u korpusu — tipično desetine hiljada. Većina komponenti je nula (jer jedan dokument koristi mali deo rečnika), a semantički bliske reči poput `automobil` i `vozilo` zauzimaju potpuno nezavisne dimenzije.

Latentna semantička analiza

Latentna semantička analiza (LSA) rešava ove probleme tako što preslikava dokumente u niskodimenzionalni prostor *latentnih (skrivenih) tema*. Ideja je da se iza pojavljivanja reči kriju semantičke strukture. Na primer, dokumenti koji koriste reči `automobil`, `motor` i `benzin` verovatno pripadaju istoj temi.

Postupak. Polazi se od matrice termina-dokumenata $A \in \mathbb{R}^{m \times n}$, gde je m broj dokumenata, a n broj termina. Na nju se primenjuje singularna dekompozicija (singular value decomposition, SVD):

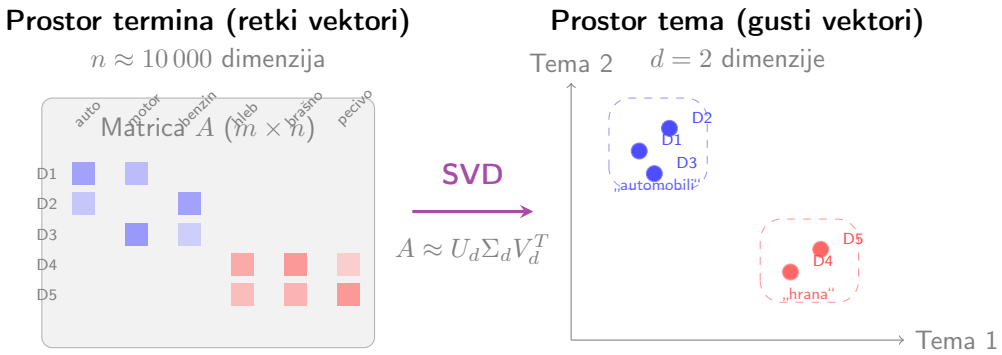
$$A = U \Sigma V^T.$$

Ako je rang (broj linearno nezavisnih redova ili kolona) matrice A jednak r, tada je $U \in \mathbb{R}^{m \times r}$, $\Sigma \in \mathbb{R}^{r \times r}$, a $V^T \in \mathbb{R}^{r \times n}$. U LSA se zadržava samo prvih $d \ll r$ najvećih singularnih vrednosti, pa se dobija aproksimacija

$$A \approx U_d \Sigma_d V_d^T,$$

gde je $U_d \in \mathbb{R}^{m \times d}$, $\Sigma_d \in \mathbb{R}^{d \times d}$, a $V_d^T \in \mathbb{R}^{d \times n}$. Redovi matrice $U_d \Sigma_d$ predstavljaju dokumente u d-dimenzionalnom latentnom prostoru, dok $V_d \Sigma_d$ analogno opisuje termine u tom prostoru. Umesto hiljada retkih komponenti, svaki dokument je sada opisan sa svega nekoliko gustih — i to upravo onih koje nose najviše informacije o semantičkoj strukturi korpusa.

Intuitivno, svaka od d zadržanih tema predstavlja grupu reči koje se često pojavljuju zajedno. Na primer, prva tema može uhvatiti vezu između reči `automobil`, `motor` i `benzin`, a druga između `hleb`, `brašno` i `pecivo`. Dokumenti koji koriste reči iz iste grupe dobiće slične koordinate u prostoru tema, čak i ako ne dele nijednu konkretnu reč.



Slika 4.3: Latentna semantička analiza: dokumenti iz visokedimenzionalnog, retkog prostora termina (levo) preslikavaju se SVD dekompozicijom u niskodimenzionalni prostor tema (desno), gde se semantički srodni dokumenti grupišu.

Normiranje. Nakon projekcije dokumenata u latentni prostor, nad dobijenim vektorima se često primenjuje L_2-normiranje:

$$\hat{x} = \frac{x}{\|x\|}.$$

Tako svi dokumenti dobijaju jediničnu dužinu, pa poređenje zavisi samo od ugla između vektora. U tom slučaju kosinusna sličnost i euklidsko rastojanje postaju povezani, jer za normirane vektore važi

$$\|\hat{x} - \hat{y}\|^2 = 2(1 - \cos\theta).$$

Hajde da vidimo zašto ovo važi. Neka su $\hat{x}, \hat{y} \in \mathbb{R}^n$ normirani vektori (tj. $\|\hat{x}\| = \|\hat{y}\| = 1$), i neka θ bude ugao između njih. Kvadrat euklidskog rastojanja između vektora je:

$$\|\hat{x} - \hat{y}\|^2 = (\hat{x} - \hat{y}) \cdot (\hat{x} - \hat{y}) = \hat{x} \cdot \hat{x} - 2\hat{x} \cdot \hat{y} + \hat{y} \cdot \hat{y}.$$

Pošto su vektori normirani:

$$\hat{x} \cdot \hat{x} = 1, \quad \hat{y} \cdot \hat{y} = 1,$$

dobijamo

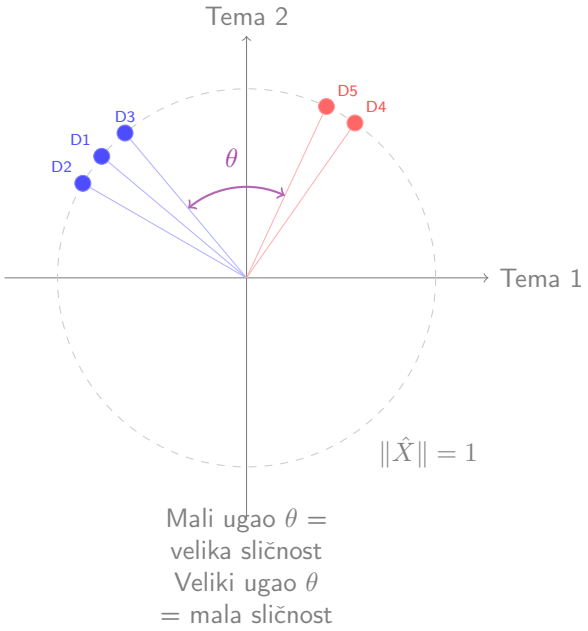
$$\|\hat{x} - \hat{y}\|^2 = 2 - 2(\hat{x} \cdot \hat{y}).$$

Podsećamo da je kosinus ugla definisan kao:

$$\cos\theta = \frac{\hat{x} \cdot \hat{y}}{\|\hat{x}\| \, \|\hat{y}\|}.$$

Za normirane vektore ovo postaje:

$$\cos\theta = \hat{x} \cdot \hat{y}.$$



Slika 4.4: Nakon normiranja, svi dokumenti leže na jediničnoj kružnici. Ugao θ između vektora direktno meri semantičku sličnost — manji ugao znači sličnije dokumente.

Tako dobijamo konačnu formulu:

$$\boxed{\|\hat{x} - \hat{y}\|^2 = 2(1 - \cos\theta)}.$$

U praksi se algoritmi često primenjuju direktno na normiranu LSA reprezentaciju, jer već sama redukcija dimenzionalnosti i uklanjanje retkosti značajno poboljšavaju kvalitet analize.

4.1.6 Diskretne niske u numeričke podatke

Transformacija diskretnih niski

Diskretne niske — biološke sekvence, simbolički nizovi, kategorijske vremenske sekvence — nisu pogodne za standardne numeričke metode. Na primer, za DNK sekvencu `ACTG` ne možemo izračunati prosek, korelaciju ili primeniti PCA, jer simboli nemaju numeričku strukturu: ne postoji smisleno rastojanje između `A` i `C`, niti operacija `A` + `C`. Zato je potrebno prevesti simboličke podatke u numerički oblik, uz očuvanje strukturnih informacija (koji simboli se gde pojavljuju, koliko često, u kakvim obrascima).

Jedan pristup se sastoji iz dva koraka:

1. **Razdvajanje po simbolima.** Diskretna niska se konvertuje u skup binarnih nizova — po jedan za svaki različit simbol u nizu. Svaki binarni niz na poziciji i ima vrednost 1 ako je na toj poziciji prisutan dati simbol, a 0 u suprotnom. Time se *jedan* niz od n simbola pretvara u k binarnih nizova dužine n, gde je k veličina alfabeta.

2. **Ekstrakcija osobina.** Svaki binarni niz se transformiše u kratak vektor osobina, a dobijeni vektori se spajaju (*konkateniraju*) u jedinstven višedimenzionalni opis originalnog niza.

Primer: DNK sekvenca

Korak 1: Razdvajanje po simbolima. Za sekvencu `ACACACTGTGACTG` (dužine $n = 14$), alfabet čine četiri simbola: `A`, `C`, `G`, `T` ($k = 4$). Svaki simbol dobija sopstveni binarni niz koji označava njegovo prisustvo na svakoj poziciji:

Pozicija	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Sekvenca	A	C	A	C	A	C	T	G	T	G	A	C	T	G
`A`	1	0	1	0	1	0	0	0	0	0	1	0	0	0
`C`	0	1	0	1	0	1	0	0	0	0	0	1	0	0
`G`	0	0	0	0	0	0	0	1	0	1	0	0	0	1
`T`	0	0	0	0	0	0	1	0	1	0	0	0	1	0

Na primer, na poziciji 3 stoji simbol `A`, pa red `A` ima vrednost 1, a redovi `C`, `G` i `T` imaju vrednost 0. Na poziciji 8 stoji `G`, pa je jedino u redu `G` vrednost 1. Na svakoj poziciji postoji tačno jedna jedinica, jer se na svakoj poziciji nalazi tačno jedan simbol — reč je zapravo o *one-hot* kodiranju primenjenom duž cele sekvence.

Rezultat su četiri binarna niza, svaka dužine 14. Ovi nizovi su dugački i retki (većina vrednosti je 0). Zato je potreban još jedan korak.

Korak 2: Ekstrakcija osobina transformacijom talasićima. Na svaki binarni niz primenjujemo transformaciju koja izvlači kompaktan opis njene strukture. Ovde koristimo **Harovu transformaciju talasićima** (Haar wavelet transform), najjednostavniju varijantu analize talasićima.

Ideja Harove transformacije. Harova transformacija rekurzivno izračunava proseke i razlike susednih parova vrednosti:

- **Prosek** para (a, b): $\dfrac{a+b}{2}$ — predstavlja *grubu aproksimaciju* signala na toj skali.
- **Razlika** para (a, b): $\dfrac{a-b}{2}$ — predstavlja *detalj* (varijaciju) na toj skali.

U svakom koraku, niz proseka postaje ulaz za sledeći nivo, dok se razlike čuvaju kao koeficijenti. Postupak se ponavlja dok ne ostane samo jedan prosek.

Harova transformacija za simbol `A`. Binarni niz za `A` ima 14 elemenata:

$$(1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0).$$

Pošto Harova transformacija zahteva dužinu koja je stepen dvojke, dopunjavamo nulama do 16 elemenata:

$$x = (1,\ 0,\ 1,\ 0,\ 1,\ 0,\ 0,\ 0,\ 0,\ 0,\ 1,\ 0,\ 0,\ 0,\ 0,\ 0).$$

Nivo 1 ($16 \to 8$ proseka + 8 razlika): Parovi su $(1,0)$, $(1,0)$, $(1,0)$, $(0,0)$, $(0,0)$, $(1,0)$, $(0,0)$, $(0,0)$.

$$\begin{aligned} \text{Proseci:} \quad & a_1 = \left(\tfrac{1}{2},\ \tfrac{1}{2},\ \tfrac{1}{2},\ 0,\ 0,\ \tfrac{1}{2},\ 0,\ 0\right), \\ \text{Razlike:} \quad & d_1 = \left(\tfrac{1}{2},\ \tfrac{1}{2},\ \tfrac{1}{2},\ 0,\ 0,\ \tfrac{1}{2},\ 0,\ 0\right). \end{aligned}$$

Nivo 2 ($8 \to 4$ proseka + 4 razlike): Parovi iz a_1 su $(\frac{1}{2}, \frac{1}{2})$, $(\frac{1}{2}, 0)$, $(0, \frac{1}{2})$, $(0, 0)$.

$$\begin{aligned} \text{Proseci:} \quad & a_2 = \left(\tfrac{1}{2},\ \tfrac{1}{4},\ \tfrac{1}{4},\ 0\right), \\ \text{Razlike:} \quad & d_2 = \left(0,\ \tfrac{1}{4},\ -\tfrac{1}{4},\ 0\right). \end{aligned}$$

Nivo 3 ($4 \to 2$ proseka + 2 razlike): Parovi iz a_2 su $(\frac{1}{2}, \frac{1}{4})$, $(\frac{1}{4}, 0)$.

$$\begin{aligned} \text{Proseci:} \quad & a_3 = \left(\tfrac{3}{8},\ \tfrac{1}{8}\right), \\ \text{Razlike:} \quad & d_3 = \left(\tfrac{1}{8},\ \tfrac{1}{8}\right). \end{aligned}$$

Nivo 4 ($2 \to 1$ prosek + 1 razlika):

$$\begin{aligned} \text{Prosek:} \quad & a_4 = \tfrac{1}{4}, \\ \text{Razlika:} \quad & d_4 = \tfrac{1}{8}. \end{aligned}$$

Vektor Harovih koeficijenata za simbol `A` sadrži konačni prosek i sve razlike, od najgrubljih ka najfinijim:

$$h_A = \left(\underbrace{\tfrac{1}{4}}_{\text{prosek}},\ \underbrace{\tfrac{1}{8}}_{d_4},\ \underbrace{\tfrac{1}{8},\ \tfrac{1}{8}}_{d_3},\ \underbrace{0,\ \tfrac{1}{4},\ -\tfrac{1}{4},\ 0}_{d_2},\ \underbrace{\tfrac{1}{2},\ \tfrac{1}{2},\ \tfrac{1}{2},\ 0,\ 0,\ \tfrac{1}{2},\ 0,\ 0}_{d_1} \right).$$

Ukupno 16 koeficijenata — isti broj kao dužina ulaznog niza, jer transformacija čuva informaciju.

Interpretacija koeficijenata.

- **Konačni prosek** $a_4 = \frac{1}{4} = 0{,}25$: simbol `A` čini 25% sekvence (4 od 16 pozicija, uključujući dopunu).
- **Razlika** $d_4 = \frac{1}{8}$: pozitivna vrednost znači da se `A` pojavljuje *češće u prvoj polovini* sekvence nego u drugoj.
- **Razlike** d_3: opisuju raspodelu unutar četvrtina sekvence.
- **Razlike** d_2**,** d_1: opisuju sve finije lokalne varijacije.

Koeficijenti sa viših nivoa (grublji) nose informaciju o globalnim obrascima, dok koeficijenti sa nižih nivoa (finiji) opisuju lokalne detalje.

Skraćivanje i konkatenacija. U praksi se ne koriste svi koeficijenti, već samo prvih nekoliko sa najviših nivoa, jer oni nose najviše informacije o strukturi signala. Na primer, ako zadržimo prva 4 koeficijenta od svakog simbola (a_4, d_4, i dva koeficijenta iz d_3), dobijamo:

$$\begin{aligned} \texttt{A}: \quad & \left(\tfrac{1}{4},\ \tfrac{1}{8},\ \tfrac{1}{8},\ \tfrac{1}{8}\right) = (0{,}250,\ \ 0{,}125,\ \ 0{,}125,\ \ 0{,}125), \\ \texttt{C}: \quad & (0{,}250,\ \ 0{,}125,\ \ 0{,}125,\ \ -0{,}125), \\ \texttt{G}: \quad & (0{,}188,\ \ -0{,}188,\ \ -0{,}063,\ \ 0{,}063), \\ \texttt{T}: \quad & (0{,}188,\ \ -0{,}188,\ \ 0{,}063,\ \ -0{,}063). \end{aligned}$$

Konačni vektor osobina sekvence `ACACACTGTGACTG` dobijamo konkatenacijom:

$$\mathbf{v} = (0{,}250,\ 0{,}125,\ 0{,}125,\ 0{,}125,\ 0{,}250,\ 0{,}125,\ 0{,}125,\ -0{,}125,\ \ldots) \in \mathbb{R}^{16}.$$

Ovaj vektor je kompaktan numerički opis cele sekvence, pogodan za primenu standardnih metoda: računanje rastojanja između sekvenci, klasterovanje, klasifikaciju i druge algoritme koji zahtevaju numerički ulaz.

Opisani pristup je opšt i primenljiv na proizvoljan diskretan alfabet — proteinske sekvence (20 aminokiselina), kategorijske vremenske serije (npr. dnevne aktivnosti), muzičke note, i bilo koje druge simboličke podatke.

4.2 Čišćenje podataka

Čišćenje podataka obuhvata tri osnovna aspekta: rad sa nedostajućim podacima, rad sa nekorektnim podacima i rad sa dupliranim podacima. Cilj je da skup podataka postane konzistentan, pouzdan i pogodan za modelovanje.

4.2.1 Rad sa nedostajućim podacima

Nedostajuće vrednosti mogu nastati iz različitih razloga — podatak nije prikupljen, ispitanik je odbio da odgovori, atribut nije primenljiv u datom slučaju (npr. atribut *plata* za decu), ili je došlo do greške pri unosu ili skladištenju podataka.

Mogući pristupi rešavanju ovog problema su:

1. **Uklanjanje zapisa** koji sadrže nedostajuće vrednosti — jednostavan pristup, ali može dovesti do značajnog gubitka podataka i pojave pristrasnosti.

2. **Unošenje nedostajućih vrednosti (imputacija)** — procena i popunjavanje nedostajućih podataka. Može biti jednostavno (aritmetička sredina, medijana, mod), uslovno (na osnovu klase ili podgrupe), zasnovano na modelima (npr. regresija, stabla odlučivanja) ili višestruko (generisanje više mogućih vrednosti kada nismo sigurni). Ovaj pristup čuva veličinu uzorka, ali može narušiti raspodelu podataka ako se ne primeni pažljivo.

3. **Algoritmi otporni na nedostajuće vrednosti** — određene metode mogu direktno raditi sa nedostajućim podacima bez prethodne obrade.

4. **Zamena specijalnim vrednostima** — primena posebnih oznaka ili vrednosti u zavisnosti od algoritma i konteksta analize.

4.2.2 Rad sa nekorektnim podacima

Nekorektni podaci nastaju usled nekonzistentnosti između izvora, grešaka pri merenju ili unosu, pogrešnog formata ili logičkih kontradikcija (npr. datum rođenja nakon datuma zaposlenja).

Za otkrivanje takvih problema koriste se tri osnovna pristupa. **Domensko znanje** omogućava proveru logičkih i poslovnih pravila — na primer, uzrast ne može biti negativan, a prosečna ocena mora pripadati dozvoljenom intervalu. **Otkrivanje nekonzistentnosti** upoređuje vrednosti iz različitih izvora ili atributa i identifikuje konflikte. **Metode orijentisane ka podacima** koriste statističke pristupe poput analize raspodele, provere raspona i otkrivanja odstupajućih vrednosti.

4.2.3 Rad sa dupliranim podacima

Duplirani podaci se često javljaju pri spajanju podataka iz heterogenih izvora — na primer, ista osoba može se pojaviti sa više različitih elektronskih adresa. Duplikati mogu preceniti frekvencije, iskriviti raspodelu i uticati na modele koji zavise od gustine ili brojnosti.

Ipak, ne treba mehanički brisati sve duplikate. Ponekad se radi o legitimnim ponovljenim događajima: isti korisnik može obaviti više kupovina, isti pacijent može imati više pregleda. Zato je važno razlikovati **duplikat zapisa** (grešku) od **ponovljenog događaja** (validnu opservaciju).

4.3 Skaliranje, normalizacija i transformacije promenljivih

Različiti atributi često imaju veoma različite opsege — jedan može biti u intervalu $[0, 1]$, a drugi u $[1, 10^9]$. Kod metoda zasnovanih na rastojanju, optimizaciji ili projekciji, atribut sa većim opsegom može neopravdano dominirati. Zbog toga se primenjuju transformacije i normalizacija.

4.3.1 Transformacije promenljivih

Transformacija promenljive znači primenu iste funkcije na sve vrednosti jednog atributa. Česte transformacije su:

$$\sqrt{x}, \qquad x^k, \qquad \log(x), \qquad e^x, \qquad |x|, \qquad \frac{1}{x}.$$

Razlozi za njihovu primenu uključuju smanjenje asimetrije raspodele, ublažavanje uticaja velikih vrednosti, približavanje normalnoj raspodeli i stabilizaciju varijanse. Na primer, logaritamska transformacija je korisna kod ekstremno širokog raspona vrednosti: ako $x \in [1, 10^9]$, tada $\log_{10} x \in [0, 9]$, što je znatno pogodnije za dalju analizu.

Treba imati na umu da transformacija može promeniti prirodu podataka — na primer, $1/x$ menja redosled između malih i velikih vrednosti, pa rezultate treba pažljivo tumačiti.

Standardizacija (Z-score normalizacija). Različiti atributi često imaju potpuno različite opsege vrednosti. Na primer, visina osobe se meri u centimetrima (tipično 150–200), a mesečna plata u dinarima (tipično 100 000–300 000). Ako bismo direktno računali euklidsko rastojanje između dve osobe, razlika u plati od 10 000 dinara bi potpuno zasenila razliku u visini od 10 cm, samo zato što su brojevi veći — ne zato što je razlika u plati objektivno važnija.

Standardizacija rešava ovaj problem tako što svaki atribut transformiše da ima srednju vrednost 0 i standardnu devijaciju 1. Za j-ti atribut sa srednjom vrednošću μ_j i standardnom devijacijom σ_j, standardizovana vrednost instance i je:

$$z_{ij} = \frac{x_{ij} - \mu_j}{\sigma_j}.$$

Oduzimanje μ_j *centrira* podatke oko nule (prosečna vrednost postaje referentna tačka), a deljenje sa σ_j ih *skalira* tako da jedna jedinica u novoj skali odgovara jednoj standardnoj

devijaciji u originalnoj. Time vrednosti različitih atributa postaju međusobno uporedive: $z = 1.5$ uvek znači 1.5 standardnih devijacija iznad proseka, bez obzira da li je reč o visini, plati ili temperaturi.

Kod približno normalne raspodele, najveći broj standardizovanih vrednosti nalazi se u intervalu $[-3, 3]$: oko 68% podataka je u $[-1, 1]$, oko 95% u $[-2, 2]$, a oko 99,7% u $[-3, 3]$.

Primer. Neka skup podataka ima tri instance sa dva atributa — visinu (u cm) i platu (u dinarima):

	Visina (cm)	Plata (RSD)
Osoba 1	170	120 000
Osoba 2	180	200 000
Osoba 3	160	160 000

Korak 1: Srednje vrednosti.

$$\mu_{\text{vis}} = \frac{170 + 180 + 160}{3} = 170, \qquad \mu_{\text{pl}} = \frac{120\,000 + 200\,000 + 160\,000}{3} = 160\,000.$$

Korak 2: Standardne devijacije.

$$\sigma_{\text{vis}} = \sqrt{\frac{(170-170)^2 + (180-170)^2 + (160-170)^2}{3}} = \sqrt{\frac{0 + 100 + 100}{3}} = \sqrt{66{,}7} \approx 8{,}16,$$

$$\sigma_{\text{pl}} = \sqrt{\frac{(120\,000-160\,000)^2 + (200\,000-160\,000)^2 + (160\,000-160\,000)^2}{3}}$$
$$= \sqrt{\frac{16 \cdot 10^8 + 16 \cdot 10^8 + 0}{3}} \approx 32\,660.$$

Korak 3: Standardizacija. Primenjujemo formulu $z_{ij} = (x_{ij} - \mu_j)/\sigma_j$ na svaku vrednost:

$$\text{Osoba 1:} \quad z_{\text{vis}} = \frac{170-170}{8{,}16} = 0, \qquad z_{\text{pl}} = \frac{120\,000-160\,000}{32\,660} = -1{,}22,$$

$$\text{Osoba 2:} \quad z_{\text{vis}} = \frac{180-170}{8{,}16} = 1{,}22, \qquad z_{\text{pl}} = \frac{200\,000-160\,000}{32\,660} = 1{,}22,$$

$$\text{Osoba 3:} \quad z_{\text{vis}} = \frac{160-170}{8{,}16} = -1{,}22, \qquad z_{\text{pl}} = \frac{160\,000-160\,000}{32\,660} = 0.$$

Standardizovani podaci su:

	Visina (z)	Plata (z)
Osoba 1	0	−1,22
Osoba 2	1,22	1,22
Osoba 3	−1,22	0

Pre standardizacije, razlika u plati između osoba 1 i 2 ($|120\,000 - 200\,000| = 80\,000$) je numerički osam hiljada puta veća od razlike u visini ($|170 - 180| = 10$). Nakon standardizacije, obe razlike iznose $|1{,}22|$ — jednako su ekstremne u odnosu na svoj atribut. Time euklidsko rastojanje u standardizovanom prostoru podjednako uzima u obzir oba atributa.

4.3.2 Min-max skaliranje

Min-max skaliranje preslikava vrednosti atributa u interval $[0, 1]$. Za j-ti atribut, transformisana vrednost instance i je:

$$y_{ij} = \frac{x_{ij} - \min_j}{\max_j - \min_j},$$

gde su $\min_j$ i $\max_j$ najmanja i najveća vrednost j-tog atributa u skupu podataka. Oduzimanje $\min_j$ pomera najmanju vrednost u nulu, a deljenje opsegom $\max_j - \min_j$ skalira sve vrednosti tako da najveća postane 1. Relativni položaji vrednosti unutar opsega ostaju očuvani — ako je neka vrednost bila na polovini puta između minimuma i maksimuma, i nakon skaliranja će biti na 0,5.

Primer. Pretpostavimo da atribut *starost* u skupu podataka ima vrednosti 20, 30, 50, 80. Tada je min = 20, max = 80, pa je opseg $80 - 20 = 60$. Primenjujemo formulu na svaku vrednost:

$$\begin{aligned} y_1 &= \frac{20 - 20}{60} = 0, \\ y_2 &= \frac{30 - 20}{60} = \frac{10}{60} \approx 0{,}17, \\ y_3 &= \frac{50 - 20}{60} = \frac{30}{60} = 0{,}50, \\ y_4 &= \frac{80 - 20}{60} = \frac{60}{60} = 1. \end{aligned}$$

Minimalna starost (20) preslikava se u 0, maksimalna (80) u 1, a ostale proporcionalno. Na primer, starost 50 je tačno na sredini opsega $[20, 80]$ i postaje 0,5.

Poređenje sa standardizacijom. Za razliku od standardizacije, min-max skaliranje garantuje *fiksne granice* izlaza ($[0, 1]$), ali je osetljivo na ekstremne vrednosti. Jedna osoba sa starošću 200 proširila bi opseg na $200 - 20 = 180$, pa bi sve ostale vrednosti bile sabijene blizu nule. Standardizacija (Z-score) nema ovaj problem jer koristi srednju vrednost i standardnu devijaciju, koje su robusnije na pojedinačne ekstremne podatke.

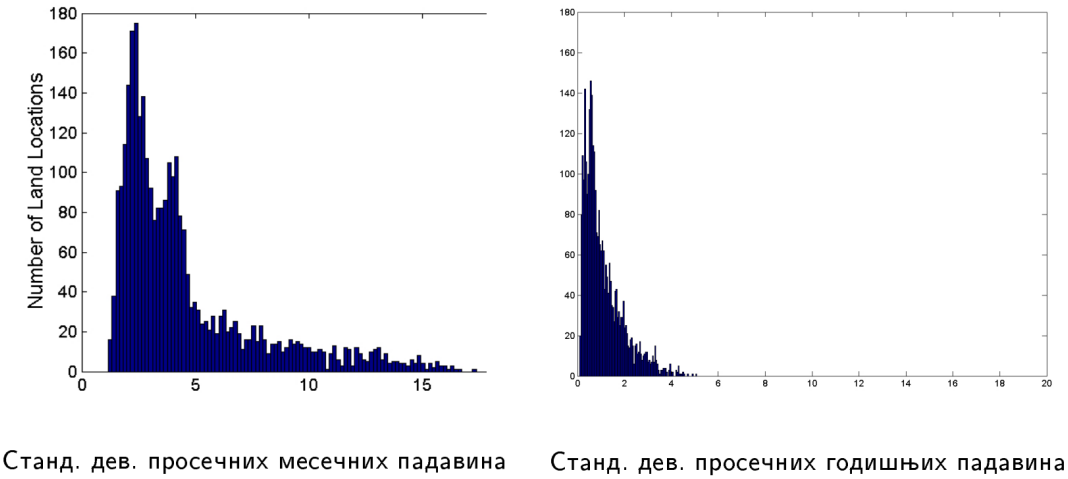
Degenerisani slučaj. Ako sve instance imaju istu vrednost nekog atributa (npr. svi imaju istu starost), tada je $\max_j = \min_j$ i imenilac postaje nula. U tom slučaju atribut ne nosi nikakvu informaciju — ne razlikuje nijednu instancu od druge — pa se obično uklanja iz analize ili se svim instancama dodeljuje konstantna vrednost (npr. $y_{ij} = 0$).

4.4 Redukcija podataka i dimenzionalnosti

Manja količina podataka omogućava bržu obradu, manju memorijsku potrošnju, lakšu interpretaciju i ponekad bolju generalizaciju modela. Glavni pristupi redukciji su agregacija, uzimanje uzorka, izbor karakteristika, redukcija pomoću rotacije osa, ostale metode dimenzione redukcije.

4.4.1 Agregacija

Agregacija kombinuje više atributa ili objekata u sažetiji oblik — na primer, umesto dnevnih podataka o padavinama koriste se mesečni ili godišnji agregati. Time se broj podataka smanjuje, lokalne oscilacije se ublažavaju, a makro obrasci postaju jasnije uočljivi.



Slika 4.5: Primer agregacije vrednosti padavina u Australiji. Prelaskom sa finije na grublju vremensku skalu lokalne oscilacije se ublažavaju, a globalni obrasci postaju pregledniji.

4.4.2 Uzimanje uzoraka

Izbor uzoraka je jedna od osnovnih tehnika u istraživanju podataka. Koristi se kako za preliminarna istraživanja (brza eksploracija pre pune analize), tako i za konačne rezultate analize podataka. Razlog je praktičan: pribavljanje i obrada kompletnog skupa podataka koji su od interesa često je preskupo ili vremenski zahtevno. Na primer, analiza svih transakcija jedne banke za period od deset godina može zahtevati dane računanja, dok se na reprezentativnom uzorku od nekoliko procenata isti zaključci mogu dobiti za nekoliko minuta.

Ključni princip je da se korišćenjem uzoraka koji su **reprezentativni** dobija efekat skoro isti kao da je rađeno na kompletnom skupu podataka. Uzorak je reprezentativan ako ima aproksimativno iste osobine kao originalni skup — istu raspodelu atributa, slične proporcije klasa i očuvanu strukturu grupa.

Tipovi uzorkovanja

Jednostavan slučajni uzorak. Svaki element u skupu podataka ima jednaku verovatnoću da bude izabran. Ovo je najosnovniji oblik uzorkovanja i dobro funkcioniše kada su podaci približno homogeni. Razlikujemo dve varijante:

- **Bez vraćanja:** jednom izabrani element se uklanja iz skupa kandidata i ne može biti ponovo izabran. Svaki element se u uzorku pojavljuje najviše jednom.
- **Sa vraćanjem:** izabrani element se vraća u skup, pa može biti izabran ponovo. Neki elementi se mogu pojaviti više puta, dok drugi uopšte ne moraju biti zastupljeni.

Pristrasno uzorkovanje. Neki podaci se namerno biraju sa većom verovatnoćom od drugih. Na primer, u detekciji prevara, slučajevi prevare čine manje od 1% transakcija. Jednostavan slučajan uzorak bi sadržao premalo prevara za smislenu analizu, pa se oni namerno uzorkuju češće kako bi se obezbedilo dovoljno primera za učenje.

Stratifikovano uzorkovanje. Podaci se najpre dele u *slojeve* prema nekom atributu — na primer, prema starosnoj grupi, regionu ili klasi — a zatim se iz svakog sloja nezavisno bira slučajni uzorak. Time se garantuje da je zastupljenost podgrupa u uzorku iste kao u originalnom skupu. Ovo je posebno važno kada su neke grupe malobrojne: bez stratifikacije, one bi mogle biti slabo zastupljene u uzorku ili čak izostavljene.

Veličina uzorka

Veličina uzorka mora biti dovoljna da sačuva strukturu podataka. Premali uzorak može izgubiti važne obrasce i retke podgrupe, dok prevelik uzorak smanjuje korist od uzorkovanja jer se približava obradi celog skupa. U praksi, odgovarajuća veličina zavisi od složenosti podataka: podaci sa više atributa, više klasa ili retkim pojavama zahtevaju veće uzorke.

Slika 4.6: Uticaj veličine uzorka na očuvanje strukture podataka. Kod većih uzoraka globalni obrasci ostaju prepoznatljivi, dok kod premalih uzoraka važne osobine mogu postati nevidljive.

4.4.3 Izbor karakteristika

Izbor karakteristika (*feature selection*) je jedan od osnovnih načina za smanjenje dimenzionalnosti. Cilj je izdvojiti podskup atributa koji najbolje opisuje problem, uz eliminaciju onih koji ne doprinose analizi. Postoje dve vrste problematičnih atributa:

- **Redundantne karakteristike** nose istu ili vrlo sličnu informaciju kao drugi atributi. Na primer, ako skup podataka sadrži i visinu u centimetrima i visinu u metrima, jedan od ta dva atributa je potpuno redundantan — oba savršeno opisuju istu osobinu, a zadržavanje oba samo povećava dimenzionalnost bez ikakve nove informacije.
- **Irelevantne karakteristike** nemaju stvaran uticaj na zadatak koji rešavamo. Na primer, identifikacioni broj klijenta je jedinstven za svaku osobu, ali ne govori ništa o njenom ponašanju — uključivanje takvog atributa u model može čak pogoršati rezultate jer unosi šum.

Postoji veliki broj tehnika za izbor atributa, posebno u kontekstu klasifikacije. Dobar izbor karakteristika poboljšava efikasnost algoritama (manje dimenzija znači brže računanje), smanjuje rizik od preprilagođavanja podacima (model ne može da se „uhvati“ za

irelevantne obrasce) i olakšava interpretaciju rezultata. Ako su karakteristike loše odabrane, čak i vrlo dobar algoritam može dati slabe rezultate.

4.4.4 Konstrukcija karakteristika

Za razliku od izbora (koji bira podskup postojećih atributa), **konstrukcija karakteristika** (*feature engineering*) podrazumeva formiranje *novih* atributa iz postojećih. Često se formiraju novi atributi koji uključuju važne karakteristike radi efikasnije obrade. Primeri uključuju:

- **Kombinovanje atributa:** odnos, razlika ili proizvod dva atributa — na primer, umesto zasebne visine i težine, indeks telesne mase BMI = težina/visina^2 je informativniji za medicinsku analizu.
- **Preslikavanje u novi prostor:** Furijeova analiza prevodi signal iz vremenskog u frekvencijski domen, transformacije talasićima (poput Harove) izvlače obrasce na različitim skalama, a projekcije poput PCA (o kojima će biti reči kasnije) otkrivaju pravce maksimalne varijanse.

Nove karakteristike mogu biti znatno informativnije od originalnih podataka i pogodnije za obradu standardnim algoritmima.

4.4.5 Linearne metode redukcije dimenzionalnosti

Ako su atributi međusobno korelisani, moguće je izvršiti promenu koordinatnog sistema tako da novi pravci osa bolje prate strukturu podataka. Osnovna ideja je rotacija osa: umesto originalnih osa koje su proizvoljno postavljene (npr. osa za visinu i osa za težinu), tražimo nove ose koje se poklapaju sa pravcima najvećeg rasipanja podataka (slika 4.7).

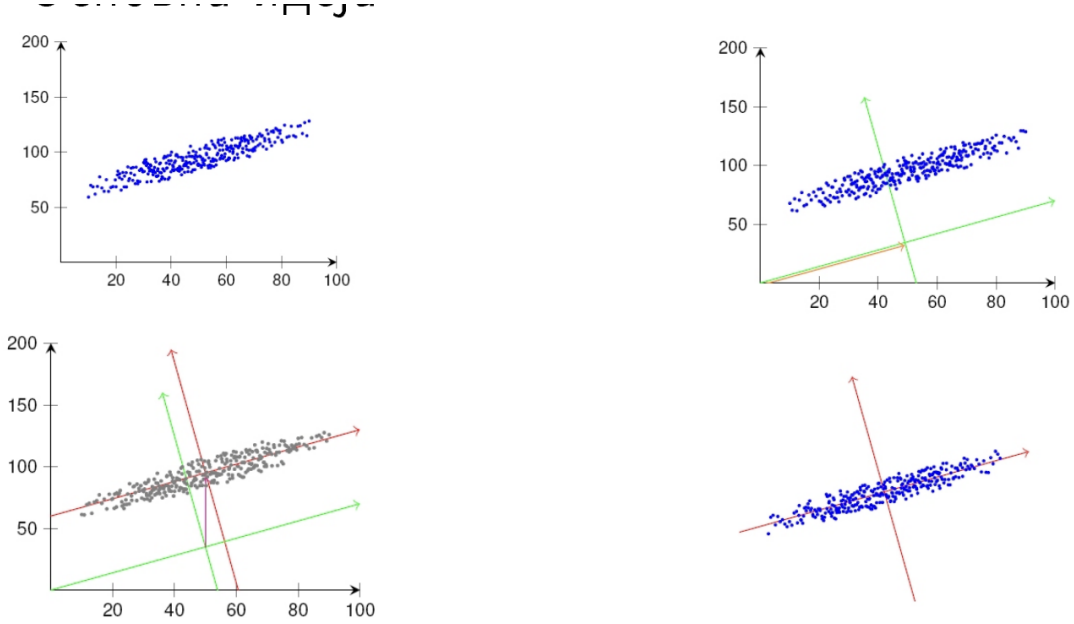
Cilj je pronaći novi koordinatni sistem u kome:

- prva osa je usmerena tako da projekcija podataka duž nje ima najveću moguću varijansu,
- svaka naredna osa je usmerena tako da projekcija podataka duž nje hvata što više varijanse koja nije obuhvaćena prethodnim osama,
- sve ose su međusobno ortogonalne.

Ako se nakon rotacije pokaže da projekcija podataka na neku od novih osa pokazuje vrlo malu raspršenost (tj. vrednosti su skoro iste duž te ose), takvu osu možemo zanemariti. Ovaj postupak smanjuje dimenzionalnost podataka, a pritom zadržava većinu informacija sadržanih u originalnim atributima.

4.4.6 Analiza glavnih komponenti (PCA)

PCA (*Principal Component Analysis*) je jedna od najvažnijih metoda redukcije dimenzionalnosti. Koristi se za smanjenje broja dimenzija, nalaženje obrazaca u višedimenzionalnim podacima i vizualizaciju — transformacijom originalnih atributa u nove, međusobno nekorelisane **glavne komponente**.



Slika 4.7: Osnovna ideja redukcije dimenzionalnosti pomoću rotacije osa. U originalnom koordinatnom sistemu (levo) podaci su korelisani duž dijagonale. Nakon rotacije (desno) prva nova osa prati smer najvećeg rasipanja, a druga opisuje malo preostalo odstupanje i može se odbaciti.

Osnovna ideja

PCA traži takvu transformaciju podataka za koju važi:

1. Svaki par novodobijenih atributa ima kovarijansu 0 (nekorelisani su).
2. Atributi su uređeni u opadajućem redosledu prema veličini varijanse koju pokrivaju.
3. Ose su međusobno ortogonalne, tako da svaki naredni atribut pokriva što je moguće veći broj preostalih varijansi.

Novi sistem osa zavisi od korelacija između originalnih atributa. PCA se najčešće primenjuje nakon centriranja podataka (oduzimanje srednje vrednosti od svake kolone), a ako atributi imaju različite merne jedinice, prethodno se radi i standardizacija.

Matrica kovarijanse

Za matricu podataka D reda $m \times n$ (gde je m broj objekata, n broj atributa) formira se matrica kovarijanse C dimenzija $n \times n$, sa elementima:

$$c_{ij} = \text{cov}(d_{*i},\ d_{*j}),$$

gde d_{*i} i d_{*j} označavaju i-tu i j-tu kolonu matrice D.

Kovarijansa je mera kako se atributi menjaju u paru:

- **Pozitivna kovarijansa:** kada jedan atribut raste, i drugi teži da raste (npr. visina i težina).

- **Negativna kovarijansa:** kada jedan raste, drugi teži da opada (npr. temperatura i prodaja zimskih jakni).

- **Kovarijansa bliska nuli:** slaba linearna zavisnost između atributa.

Za $i = j$, kovarijansa prelazi u varijansu: $c_{ii} = \text{var}(d_{*i})$, pa dijagonala matrice C sadrži varijanse pojedinih atributa.

Ako se matrica podataka prethodno pripremi tako da je srednja vrednost svakog atributa jednaka 0 (centrirana matrica $\tilde{D}$), matrica kovarijansi se može kompaktno zapisati kao:

$$C = \frac{1}{m-1}\tilde{D}^T\tilde{D}.$$

PCA transformacija

Transformacija se vrši upotrebom sopstvenih vrednosti matrice kovarijanse. Neka su $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n \geq 0$ (nenegativne) sopstvene vrednosti matrice C, uređene u opadajućem redosledu, i neka je $U = [u_1, u_2, \ldots, u_n]$ matrica odgovarajućih sopstvenih vektora, uređena tako da i-ti vektor odgovara i-toj najvećoj sopstvenoj vrednosti. Neka je matrica D prethodno pripremljena tako da je srednja vrednost svakog od atributa jednaka 0. Tada važi:

1. Matrica $D' = D\,U$ je tražena transformacija matrice podataka.

2. Novi atribut (glavna komponenta) je linearna kombinacija starih atributa; težine linearne kombinacije i-tog atributa su komponente i-tog sopstvenog vektora.

3. Varijansa i-te glavne komponente jednaka je λ_i; zbir varijansi originalnih atributa jednak je zbiru varijansi novih atributa (ukupna varijansa se ne menja — PCA je samo preraspodeljuje po novim osama).

4. Novi atributi se nazivaju *glavne komponente*; prvi novi atribut je prva glavna komponenta (sa najvećom varijansom), itd.

Udeo varijanse koji objašnjava i-ta glavna komponenta je:

$$\frac{\lambda_i}{\sum_{j=1}^{n}\lambda_j}.$$

Kumulativni zbir ovih udela za prvih r komponenti koristi se kao kriterijum pri izboru broja komponenti koje će biti zadržane.

Primer

Neka su data tri objekta sa dva atributa:

	x_1	x_2
Objekat1	2	4
Objekat2	4	6
Objekat3	6	8

Korak 1: Centriranje. Srednje vrednosti kolona su $\mu_1 = 4$, $\mu_2 = 6$. Oduzimamo ih:

$$\tilde{D} = \begin{pmatrix} 2-4 & 4-6 \\ 4-4 & 6-6 \\ 6-4 & 8-6 \end{pmatrix} = \begin{pmatrix} -2 & -2 \\ 0 & 0 \\ 2 & 2 \end{pmatrix}.$$

Korak 2: Matrica kovarijanse.

$$C = \frac{1}{3-1}\tilde{D}^T\tilde{D} = \frac{1}{2}\begin{pmatrix} 8 & 8 \\ 8 & 8 \end{pmatrix} = \begin{pmatrix} 4 & 4 \\ 4 & 4 \end{pmatrix}.$$

Korak 3: Sopstvene vrednosti. Rešavamo $\det(C - \lambda I) = 0$:

$$(4-\lambda)^2 - 16 = 0 \implies \lambda^2 - 8\lambda = 0 \implies \lambda(\lambda - 8) = 0.$$

Sopstvene vrednosti su $\lambda_1 = 8$ i $\lambda_2 = 0$.

Prva glavna komponenta objašnjava $\frac{8}{8+0} = 100\%$ varijanse, a druga 0%. To znači da su podaci savršeno raspoređeni duž jednog pravca — druga dimenzija ne nosi nikakvu informaciju.

Korak 4: Sopstveni vektori. Za $\lambda_1 = 8$:

$$(C - 8I)u = 0 \implies \begin{pmatrix} -4 & 4 \\ 4 & -4 \end{pmatrix} u = 0 \implies u_1 = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 \\ 1 \end{pmatrix}.$$

Za $\lambda_2 = 0$:

$$Cu = 0 \implies u_2 = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 \\ -1 \end{pmatrix}.$$

Korak 5: Transformacija.

$$D' = \tilde{D}\,U = \begin{pmatrix} -2 & -2 \\ 0 & 0 \\ 2 & 2 \end{pmatrix} \cdot \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} = \begin{pmatrix} -2\sqrt{2} & 0 \\ 0 & 0 \\ 2\sqrt{2} & 0 \end{pmatrix}.$$

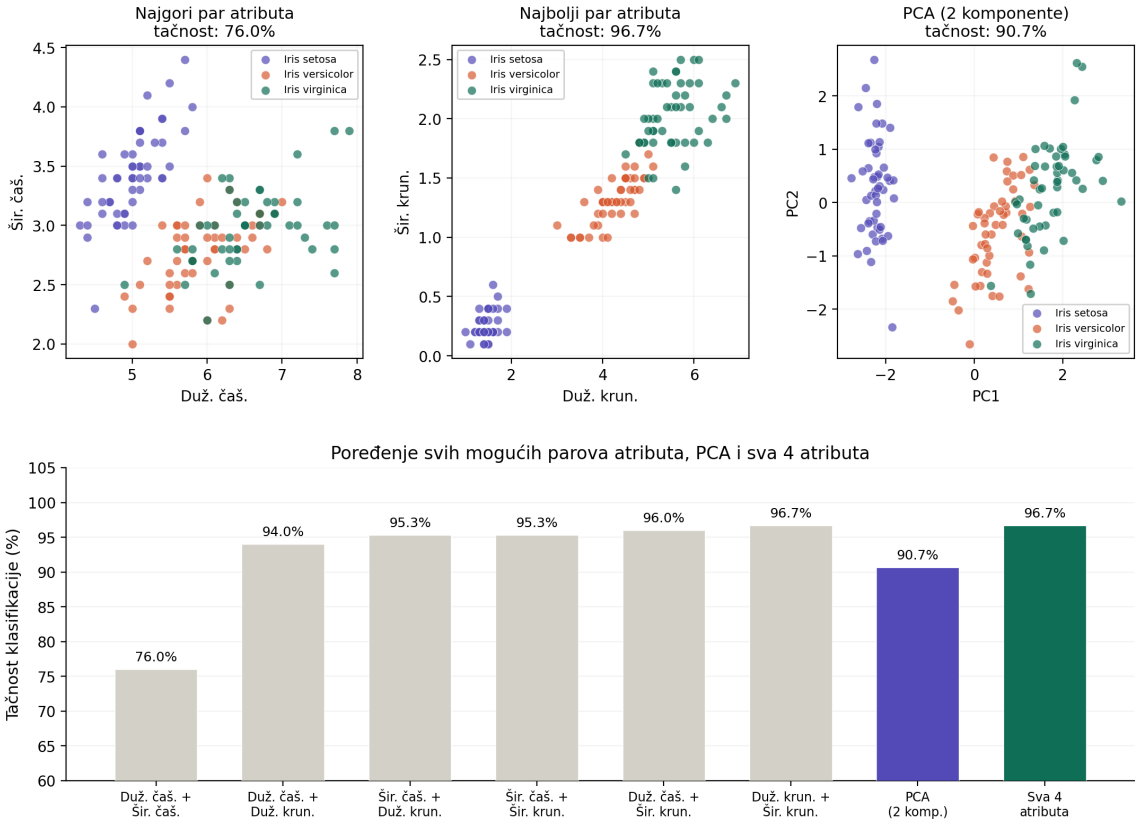
Druga kolona je potpuno nula — potvrda da druga komponenta ne nosi informaciju. Zadržavanjem samo prve kolone, trodimenzionalni opis (tri objekta u 2D prostoru) svodi se na jednodimenzionalni: $(-2\sqrt{2},\ 0,\ 2\sqrt{2})$, bez ikakvog gubitka informacije.

Redukcija dimenzionalnosti

U opštem slučaju, podaci nisu savršeno korelisani kao u primeru iznad, pa će i druga (i sledeće) komponente nositi nešto varijanse. Ipak, ako prvih nekoliko glavnih komponenti nose najveći deo ukupne varijanse (npr. 95%), preostale se mogu odbaciti uz mali gubitak informacije. Za zadržavanje prvih r komponenti uzima se $U_r = [u_1, \ldots, u_r]$ i računa:

$$D_r = \tilde{D}\,U_r,$$

čime se postiže značajno smanjenje dimenzionalnosti uz kompaktniju reprezentaciju i lakšu vizualizaciju.



Slika 4.8: PCA primenjena na Iris dataset (150 cvetova, 4 atributa, 3 vrste). Gore: scatter dijagrami za najgori par atributa (dužina i širina čašičnog listića, tačnost klasifikacije 76,0%), najbolji par (dužina i širina kruničnog listića, 96,7%) i PCA projekciju na dve komponente (90,7%). Dole: poređenje tačnosti klasifikacije za svih šest mogućih parova atributa, PCA i sva četiri atributa. PCA automatski pronalazi projekciju koja koristi informaciju iz svih atributa i daje konzistentno dobre rezultate (90,7%) bez potrebe za ručnim izborom — za razliku od proizvoljnog izbora para, koji može dati i 76% i 97% u zavisnosti od toga koliko sreće imamo.

4.4.7 Ostale metode redukcije dimenzionalnosti

Pored PCA, značajne metode uključuju:

- **SVD** (*Singular Value Decomposition*) — opšta matična dekompozicija blisko povezana sa PCA; primenjiva i na matrice koje nisu kvadratne.
- **LSA** (*Latent Semantic Analysis*) — otkrivanje latentnih semantičkih odnosa u tekstu pomoću SVD dekompozicije matrice termina-dokumenata.
- **Furijeove transformacije i transformacije talasićima** — prelazak u frekvencijski domen, gde se periodični obrasci i lokalna struktura signala opisuju kompaktnije.
- **Analiza faktora** — modelovanje latentnih (skrivenih) faktora koji objašnjavaju korelacije između posmatranih varijabli.
- **Multidimenzionalno skaliranje (MDS)** — preslikavanje objekata u niskodimenzionalni prostor uz očuvanje međusobnih rastojanja.

- **ISOMAP** i **spektralne transformacije grafova** — nelinearne metode koje mogu uhvatiti složenije strukture u podacima (zakrivljene mnogostrukosti, strukture grafova).

4.5 Zaključak

Priprema podataka obuhvata skup međusobno povezanih postupaka — izdvajanje karakteristika, transformaciju tipova, čišćenje, normalizaciju i redukciju dimenzionalnosti — čiji je cilj da se sirovi podaci pretvore u reprezentaciju pogodnu za analizu. Dobra priprema često ima veći uticaj na uspeh analize nego izbor samog algoritma, zbog čega se preprocesiranje s pravom smatra osnovom uspešnog procesa istraživanja podataka.

Pri tome treba imati u vidu da transformacije i redukcija dimenzionalnosti gotovo uvek predstavljaju kompromis između jednostavnosti reprezentacije i očuvanja informacije. Izbor postupaka mora biti usklađen sa prirodom problema, osobinama skupa podataka i ciljem analize.

Glava 5

Vizuelizacija podataka

5.1 Uvod

Vizuelizacija podataka je postupak prevođenja podataka u vizuelni ili tabelarni oblik radi lakše analize, uočavanja pravilnosti i donošenja jasnih zaključaka o strukturi podataka i odnosima među njima. Vizuelni prikaz omogućava da se velika količina podataka sagleda brzo i intuitivno, što je naročito važno u ranim fazama istraživanja, kada je cilj razumeti strukturu skupa podataka pre primene složenijih metoda.

Čovek veoma efikasno uočava obrasce, trendove, grupisanja i odstupanja. Upravo zato vizuelizacija nije samo sredstvo za predstavljanje rezultata, već i ključan alat za samo istraživanje — prvi uvid u raspodelu vrednosti, odnose među atributima ili prisustvo izdvojenih objekata najčešće se stiče grafičkim prikazom.

Vizuelizacija, pored toga, pomaže da se iskoristi domensko znanje stručnjaka: interesantni obrasci se mnogo lakše prepoznaju na slici nego u numeričkom ili tabelarnom zapisu.

5.2 Osnove vizuelizacije podataka

5.2.1 Izbor vrste vizuelizacije

Izbor odgovarajuće vizuelizacije zavisi od prirode podataka i cilja analize. Raspodela jedne numeričke osobine prirodno se prikazuje histogramom ili boks plot-om, dok se odnos između dve numeričke osobine bolje vidi na dijagramu raspršenja.

Pri izboru prikaza najčešće se primenjuju sledeće strategije:

- naglašavanje važnih atributa uz potiskivanje manje bitnih,
- izbor podskupa atributa ili objekata,
- dimenziona redukcija radi predstavljanja podataka u dve ili tri dimenzije,
- istovremeno posmatranje više parova atributa umesto pokušaja da se sve prikaže odjednom,
- proređivanje ili zumiranje kada prikaz postane previše gust.

Kada deo ekrana sadrži previše tačaka ili drugih elemenata, prikaz postaje nepregledan. Tada se može primeniti proređivanje, zumiranje ili izbor reprezentativnog podskupa, uz obaveznu pažnju da se ne izgubi informacija bitna za analizu.

5.2.2 Preslikavanje podataka u grafičke elemente

Osnova svake vizuelizacije jeste preslikavanje podataka u grafičke elemente: objekti, atributi i njihovi odnosi prevode se u tačke, linije, oblike, boje i veličine.

Najčešći pristupi podrazumevaju da se:

- objekti prikazuju kao tačke,
- vrednosti atributa određuju poziciju na osi,
- dodatni atributi kodiraju bojom, veličinom ili oblikom markera,
- odnosi među objektima prikazuju linijama ili relativnim položajem.

Prikazi zasnovani na poziciji, poput histograma i dijagrama raspršenja, spadaju među najčešće korišćene jer omogućavaju brzo uočavanje grupa, oblika raspodele i izdvojenih elemenata.

5.2.3 Uređenost prikaza

Način raspoređivanja vizuelnih elemenata može presudno uticati na razumevanje podataka. Isti podaci mogu delovati nepregledno ili veoma jasno, u zavisnosti od rasporeda redova, kolona, osa i grupa elemenata.

Primer preuređivanja tabele

U tabeli 5.1 prikazan je početni raspored binarne tabele u kome se struktura podataka teško uočava.

Tabela 5.1: Početni raspored elemenata u binarnoj tabeli.

	1	**2**	**3**	**4**	**5**	**6**
1	0	1	0	1	1	0
2	1	0	1	0	0	1
3	0	1	0	1	1	0
4	1	0	1	0	0	1
5	0	1	0	1	1	0
6	1	0	1	0	0	1
7	0	1	0	1	1	0
8	1	0	1	0	0	1
9	0	1	0	1	1	0

Iako podaci sadrže strukturu, raspored redova i kolona je takav da se ona ne vidi na prvi pogled. Ovaj primer pokazuje da sama vizuelizacija nije dovoljna, jednako je važan i način organizacije elemenata.

U tabeli 5.2 prikazana je delimična promena rasporeda kolona. Leva podtabela zadržava originalan raspored, dok je u desnoj izvršena permutacija kolona.

Poređenjem dva prikaza u tabeli 5.2 uočava se da i mala promena rasporeda može poboljšati preglednost. Struktura još uvek nije potpuno jasna, ali se naziru sličnosti među kolonama.

Tabela 5.2: Promena rasporeda kolona radi lakšeg uočavanja strukture. Levo: originalan raspored; desno: permutovane kolone (redosled: 1, 2, 3, 4, 6, 5).

	1	**2**	**3**	**4**	**5**	**6**
1	0	1	0	1	1	0
2	1	0	1	0	0	1
3	0	1	0	1	1	0
4	1	0	1	0	0	1
5	0	1	0	1	1	0
6	1	0	1	0	0	1
7	0	1	0	1	1	0
8	1	0	1	0	0	1
9	0	1	0	1	1	0

	1	**2**	**3**	**4**	**6**	**5**
1	0	1	0	1	0	1
2	1	0	1	0	1	0
3	0	1	0	1	0	1
4	1	0	1	0	1	0
5	0	1	0	1	0	1
6	1	0	1	0	1	0
7	0	1	0	1	0	1
8	1	0	1	0	1	0
9	0	1	0	1	0	1

Tabela 5.3: Preuređena tabela u kojoj se jasno vide dve grupe objekata i atributa. Gore levo: originalan raspored; gore desno: permutovane kolone; dole: potpuno preuređena tabela (redovi i kolone) sa istaknutom blok-strukturom.

	1	**2**	**3**	**4**	**5**	**6**
1	0	1	0	1	1	0
2	1	0	1	0	0	1
3	0	1	0	1	1	0
4	1	0	1	0	0	1
5	0	1	0	1	1	0
6	1	0	1	0	0	1
7	0	1	0	1	1	0
8	1	0	1	0	0	1
9	0	1	0	1	1	0

	1	**2**	**3**	**4**	**6**	**5**
1	0	1	0	1	0	1
2	1	0	1	0	1	0
3	0	1	0	1	0	1
4	1	0	1	0	1	0
5	0	1	0	1	0	1
6	1	0	1	0	1	0
7	0	1	0	1	0	1
8	1	0	1	0	1	0
9	0	1	0	1	0	1

	1	**3**	**6**	**2**	**4**	**5**
4	1	1	1	0	0	0
2	1	1	1	0	0	0
6	1	1	1	0	0	0
8	1	1	1	0	0	0
5	0	0	0	1	1	1
3	0	0	0	1	1	1
9	0	0	0	1	1	1
7	0	0	0	1	1	1
1	0	0	0	1	1	1

U tabeli 5.3 prikazano je dodatno preuređivanje i redova i kolona. Gornji deo ponovo sadrži dve verzije (pre i posle permutacije kolona), dok donji deo prikazuje rezultat nakon preuređivanja i redova i kolona, sa obojenjem koje naglašava blok-strukturu.

Sada se jasno uočavaju dve grupe objekata: jedna ima jedinice u prvom delu atributa, a druga u drugom delu (tabela 5.3, donji deo). Ključna pouka ovog primera jeste da preuređivanje može otkriti obrasce skrivene u sirovom rasporedu — ista ideja primenjuje se i kod matrica sličnosti, toplotnih mapa i mrežnih prikaza.

5.3 Osnovne tehnike vizuelizacije

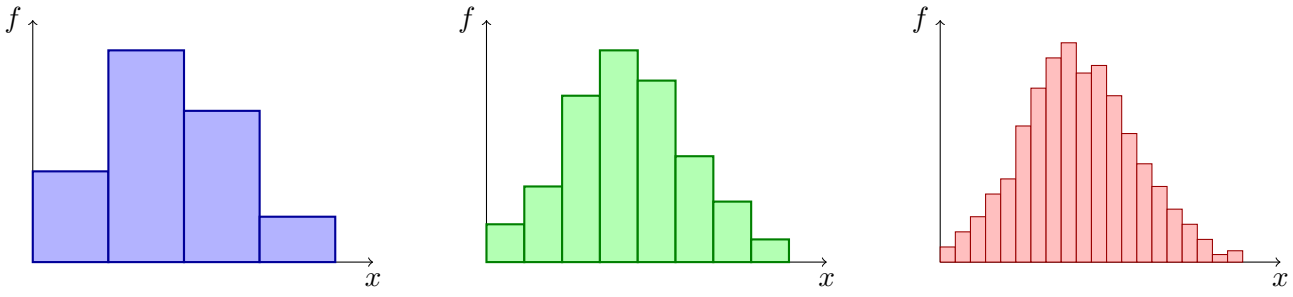
5.3.1 Histogrami

Histogram je standardna tehnika za prikaz raspodele vrednosti jednog atributa. Opseg vrednosti deli se na binove (intervale), za svaki se broji koliko objekata upada u njega, a potom se crtaju stubići čija visina odgovara broju ili relativnoj učestanosti objekata.

Histogram omogućava da se uoče:

- centralna oblast i širina raspodele,
- asimetrija,
- višemodalna raspodela (više vrhova),
- mogući izdvojeni elementi.

Broj binova direktno utiče na izgled histograma: premalo binova skriva detalje, a previše ih čini šumovitim. Zato se histogram uvek tumači uz svest da oblik delimično zavisi od izbora podele. Na slici 5.1 ilustrovan je ovaj efekat na istom skupu podataka sa različitim brojem binova.



(a) 4 bina (b) 8 binova (c) 20 binova

Slika 5.1: Uticaj broja binova na izgled histograma za iste podatke: (a) premalo binova gubi detalje, (b) umeren broj binova daje jasan oblik raspodele, (c) previše binova unosi šum.

Pored globalnog prikaza raspodele, histogram se može primeniti i za analizu pojedinačnih klasa. Tabela 5.4 prikazuje osnovne statističke mere za atribut *petal width* (širina latice) iz Iris skupa podataka, razdvojene po klasama.

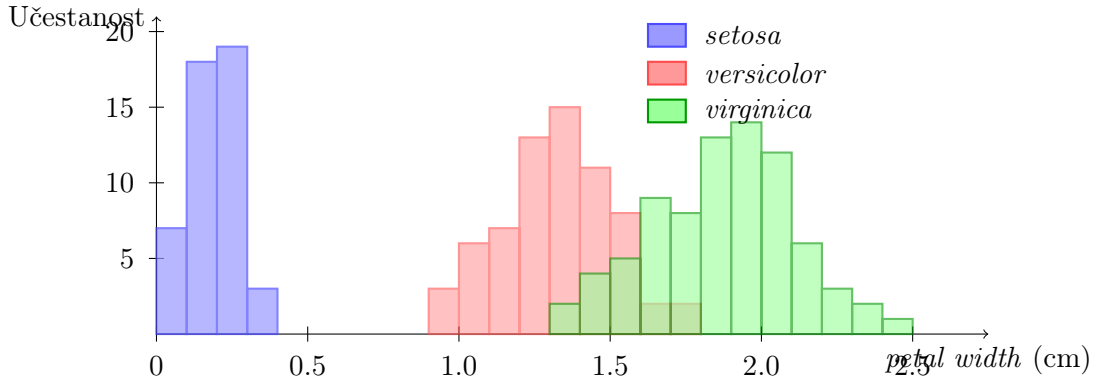
Iz tabele 5.4 vidi se da klasa *Iris-setosa* ima znatno manje vrednosti širine latice od ostale dve klase, bez preklapanja opsega. Ovo je upravo informacija koju histogram

Tabela 5.4: Statističke mere atributa *petal width* za tri klase Iris skupa podataka.

Klasa	**Srednja vrednost**	**Std. devijacija**	**Opseg**
Iris-setosa	0,25	0,11	0,1 – 0,6
Iris-versicolor	1,33	0,20	1,0 – 1,8
Iris-virginica	2,03	0,27	1,4 – 2,5

vizuelno ističe: kada se raspodele tri klase prikažu na istim osama, razdvojenost klase *Iris-setosa* odmah postaje očigledna.

Na slici 5.2 prikazani su histogrami za sve tri klase na zajedničkim osama, čime se jasno uočava stepen razdvojenosti njihovih raspodela.



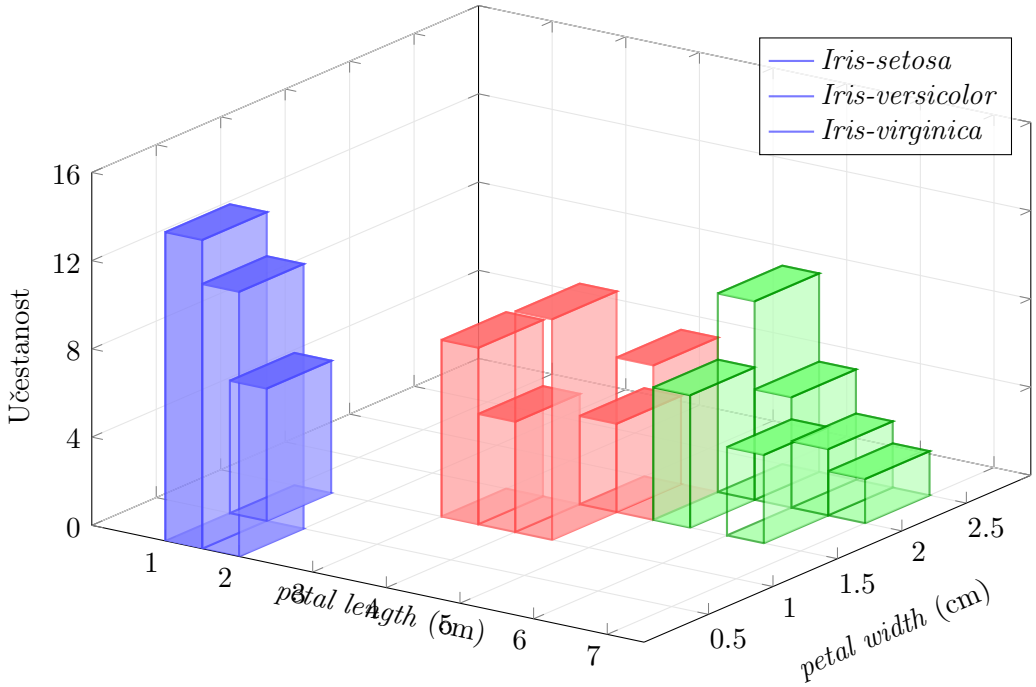
Slika 5.2: Histogrami atributa *petal width* za tri klase Iris skupa podataka. Klasa *setosa* je jasno razdvojena, dok se klase *versicolor* i *virginica* delimično preklapaju.

Kao što je vidljivo na slici 5.2, klasa *Iris-setosa* zauzima potpuno odvojen deo opsega vrednosti, dok se klase *Iris-versicolor* i *Iris-virginica* delimično preklapaju u oblasti od približno 1,4 do 1,8 cm. Ovakav prikaz ilustruje jednu od najvažnijih primena histograma u istraživanju podataka: vizuelnu procenu razdvojenosti klasa pre primene formalnih metoda klasifikacije.

Trodimenzionalni histogram

Kada želimo da istovremeno sagledamo raspodelu dve numeričke osobine, može se koristiti trodimenzionalni histogram. Obe osobine definišu ravanski prostor koji se deli na dvodimenzionalne binove, a visina svake prizme odgovara broju objekata u datom binu. Na slici 5.3 prikazan je ovakav histogram za atribute *petal length* i *petal width* iz Iris skupa podataka, razdvojeno po klasama.

Trodimenzionalni histogram na slici 5.3 otkriva ono što jednodimenzionalni prikazi ne mogu: prostorni raspored klasa u ravni definisanoj dvema osobinama. Klasa *Iris-setosa* je grupisana u donjem levom uglu (male vrednosti obe osobine), dok klase *Iris-versicolor* i *Iris-virginica* zauzimaju centralni i gornji desni deo prostora sa delimičnim preklapanjem. Ovakav prikaz je informativan, ali treba imati u vidu da je tumačenje 3D histograma teže nego kod 2D prikaza — prizme u pozadini mogu biti delimično zaklonjene, a vizuelna procena visine je manje pouzdana usled perspektivne projekcije. Zbog toga se 3D histogrami koriste kao dopuna, a ne kao zamena za standardne dvodimenzionalne prikaze.



Slika 5.3: Trodimenzionalni histogram za atribute *petal length* i *petal width* iz Iris skupa podataka. Svaka prizma predstavlja jedan dvodimenzionalni bin, a njena visina odgovara broju objekata u tom binu. Tri klase zauzimaju jasno odvojene oblasti u ravni atributa.

5.3.2 Boks plotovi

Boks plot (engl. *box-and-whisker plot*) je grafički prikaz koji se koristi za analizu raspodele numeričkih podataka. Omogućava da se u jednom prikazu sagledaju:

- medijanu podataka,
- raspon i varijansa,
- simetričnost ili asimetrija raspodele,
- prisustvo autlajera .

Boks plot je naročito koristan za poređenje raspodela više skupova podataka ili različitih grupa. Na slici 5.4 možemo videti kako izgleda boks plot za podatke sa normalnom raspodelom. U zavisnosti od raspodele podataka, širina boksa se može menjati, dužina brkova, a različite položaje unutar boksa može zauzeti linija koja predstavlja medijanu.

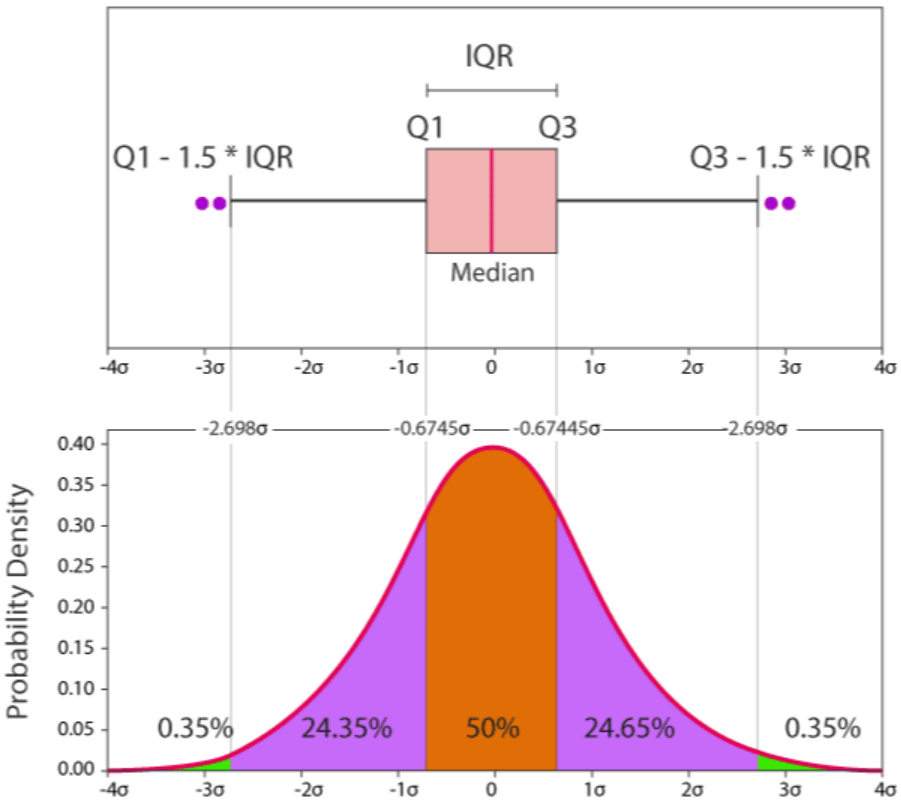
Primer skupa podataka

U nastavku se razmatra sledeći skup podataka:

$$[100,\ 120,\ 110,\ 150,\ 110,\ 140,\ 130,\ 170,\ 120,\ 220,\ 140,\ 110]$$

Nakon sortiranja, skup podataka je:

$$[100,\ 110,\ 110,\ 110,\ 120,\ 120,\ 130,\ 140,\ 140,\ 150,\ 170,\ 220]$$



Slika 5.4: Boks plot za podatke sa normalnom raspodelom

Boks u boks plotu

Za formiranje boksa u boks plotu ključni su sledeći elementi skupa podataka:

1. *Prvi kvartil* (Q_1) – vrednost ispod koje se nalazi 25% podataka
2. *Medijana* (Q_2) – centralna vrednost skupa podataka
3. *Treći kvartil* (Q_3) – vrednost ispod koje se nalazi 75% podataka

Medijana (Q_2)

Medijana (Q_2) deli skup podataka na dve jednake polovine i u boks plotu se prikazuje kao linija unutar boksa. U našem primeru, pošto skup sadrži 12 elemenata, medijana se računa kao prosečna vrednost između šestog i sedmog elementa:

$$Q_2 = \frac{120 + 130}{2} = 125$$

Prvi kvartil (Q_1)

Prvi kvartil (Q_1) je vrednost ispod koje se nalazi 25% svih podataka u sortiranom skupu i predstavlja donju ivicu boksa. U našem primeru, donju polovinu sortiranog skupa čini:

$$[100,\ 110,\ 110,\ 110,\ 120,\ 120]$$

Medijana ove polovine je:

$$Q_1 = 110$$

Treći kvartil (Q_3)

Treći kvartil (Q_3) je vrednost ispod koje se nalazi 75% podataka i predstavlja gornju ivicu boksa. U našem primeru, gornju polovinu sortiranog skupa čini:

$$[130,\ 140,\ 140,\ 150,\ 170,\ 220]$$

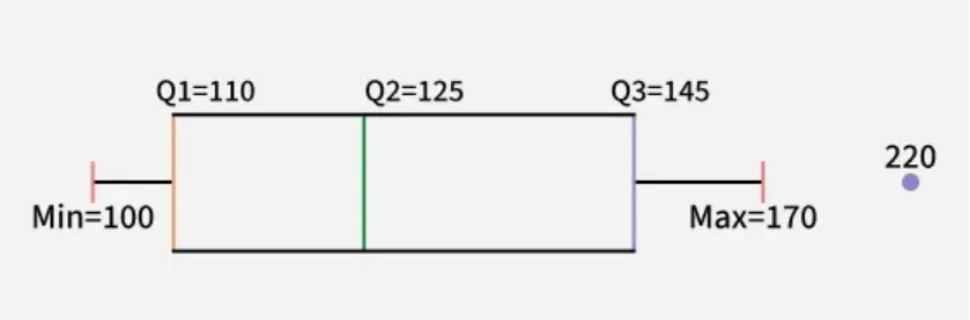
Medijana ove polovine iznosi:

$$Q_3 = 145$$

Crtanje boksa

Na osnovu ovih vrednosti možemo nacrtati boks u okviru boks plota (slika 5.5). Boks se prostire od Q_1 do Q_3, konkretno:

$$Q_1 = 110, \quad Q_3 = 145$$



Slika 5.5: Boks plot za dati skup podataka

Granice boksa su Q_1 i Q_3, a medijana Q_2 se predstavlja linijom unutar boksa. S obzirom kako su izračunate vrednosti Q_1 i Q_3, srednjih 50% vrednosti skupa podataka se nalazi unutar intervala koji boks pokriva.

Brkovi u boks plotu

Na slici 5.5 se ispred i iza boksa nalaze linije koje nazivamo brkovima (eng. *whiskers*). Da bismo njih nacrtali, potrebno je da na osnovu skupa podataka odredimo nekoliko vrednosti.

Međukvartilni raspon (IQR)

Međukvartilni raspon (IQR) meri raspon srednjih 50% podataka i definiše se kao:

$$IQR = Q_3 - Q_1$$

U našem primeru,

$$IQR = 145 - 110 = 35$$

Granice za identifikaciju autlajera

Granice za identifikaciju autlajera određuju se pomoću međukvartilnog raspona.

$$\text{Donja granica} = Q_1 - 1.5 \cdot IQR$$

$$\text{Gornja granica} = Q_3 + 1.5 \cdot IQR$$

U našem primeru:

$$\text{Donja granica} = 110 - 1.5 \cdot 35 = 57.5$$

$$\text{Gornja granica} = 145 + 1.5 \cdot 35 = 197.5$$

Pošto je u našem primeru vrednost 220 (maksimalna vrednost skupa podataka) veća od gornje granice, ona se smatra autlajerom.

U nastavku govorimo o pozicijama gde počinju levi i desni brk. U oba slučaja se podrazumeva da se brkovi završavaju na boksu.

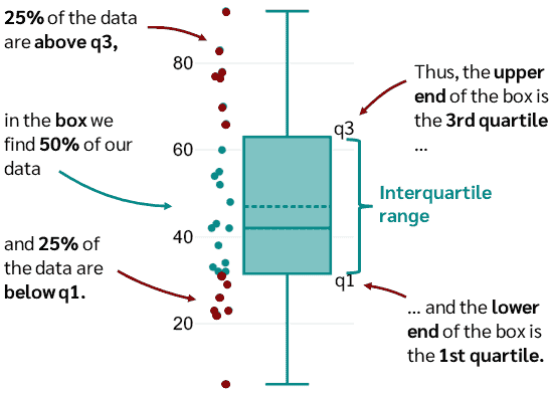
Levi (donji) brk počinje od najmanjeg elementa koji nije autlajer. To može da bude minimalni element ali i ne mora. U našem primeru, minimalni element je iznad donje granice (iznosi 100) pa će levi brk početi odatle.

Desni (gornji) brk počinje od najvećeg elementa koji nije autlajer. To može da bude maksimalni element ali i ne mora. U našem primeru, maksimalni element je veći od gornje granice (iznosi 220) pa će desni brk početi od najvećeg elementa ispod donje granice (iznosi 170). Za dati skup podataka:

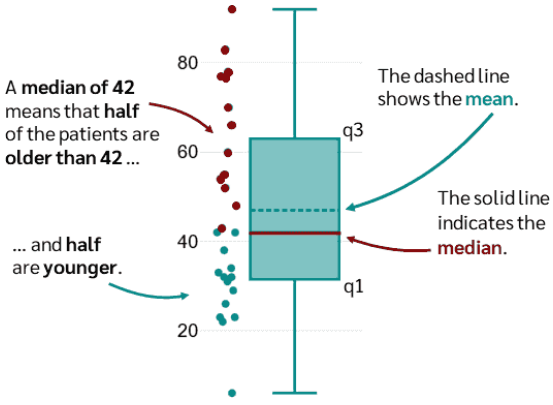
$$\text{donji brk} = 100, \quad \text{gornji brk} = 170$$

Vrednost 220 se navodi iza levog brka kao izolovana tačka koja predstavlja autlajer (slika 5.5).

Pogledajmo još jedan primer koji ilustruje boks plot i veličine na osnovu kojih se on prikazuje na slikama 5.6 i 5.7.



Slika 5.6: kvartili u boks plotu



Slika 5.7: Medijana i prosečna vrednost u boks plotu

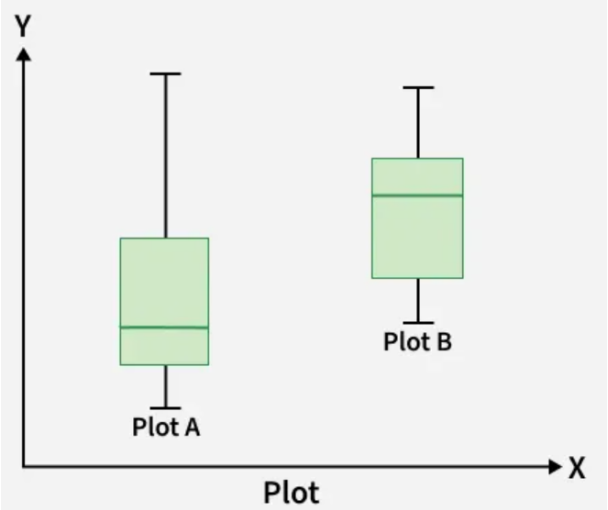
Tumačenje boks plota

Na osnovu datog primera može se zaključiti sledeće:

- raspodela je desno asimetrična,
- prisutan je jedan izražen autlajer (220).

Poređenje boks plotova

Posmatrajmo primer na slici 5.8 i razmotrimo kako pomoću boks plotova možemo porediti raspodele za dva skupa podataka.



Slika 5.8: Poređenje dva boks plota

1. *Poređenje medijana*

   Prilikom poređenja više boks plotova, važno je posmatrati položaj medijane u svakom od njih.

   - Proverava se da li se linija medijane jednog boks plota nalazi izvan boksa drugog.
   - Medijana koja je postavljena više obično ukazuje na veće ukupne vrednosti podataka.
   - Ako se medijana Grupe B nalazi izvan boksa Grupe A, to ukazuje na značajnu razliku između posmatranih grupa.

2. *Poređenje rasipanja*

   Rasipanje podataka se u boks plotu procenjuje pomoću boksa i brkova.

   - Međukvartilni raspon (IQR) predstavlja visinu boksa i opisuje rasipanje srednjih 50% podataka.
   - Duži boks ukazuje na veću varijabilnost podataka.
   - Duži brkovi označavaju veći ukupni raspon vrednosti.

- Ako Plot A ima veći boks i duže brkove u odnosu na Plot B, to znači da su podaci u Plotu A varijabilniji.

3. *Poređenje autlajera*

   Autlajeri se lako uočavaju u boks plotu i daju važnu informaciju o stabilnosti podataka.

   - Autlajeri se prikazuju kao tačke koje se nalaze izvan brkova.
   - Veći broj autlajera ukazuje na nepravilne ili nekonzistentne podatke.
   - Manji broj autlajera sugeriše stabilnije i predvidljivije vrednosti.

4. *Poređenje asimetrije raspodele*

   Asimetrija (engl. *skewness*) opisuje stepen asimetričnosti raspodele podataka.

   - Ako je medijana bliža donjoj ivici boksa, a gornji brk duži, raspodela je desno asimetrična.
   - Ako je medijana bliža gornjoj ivici boksa, a donji brk duži, raspodela je levo asimetrična.
   - Na primer, ako je Plot A levo asimetričan, a Plot B desno asimetričan, to jasno ukazuje na razlike u raspodeli podataka između tih skupova.

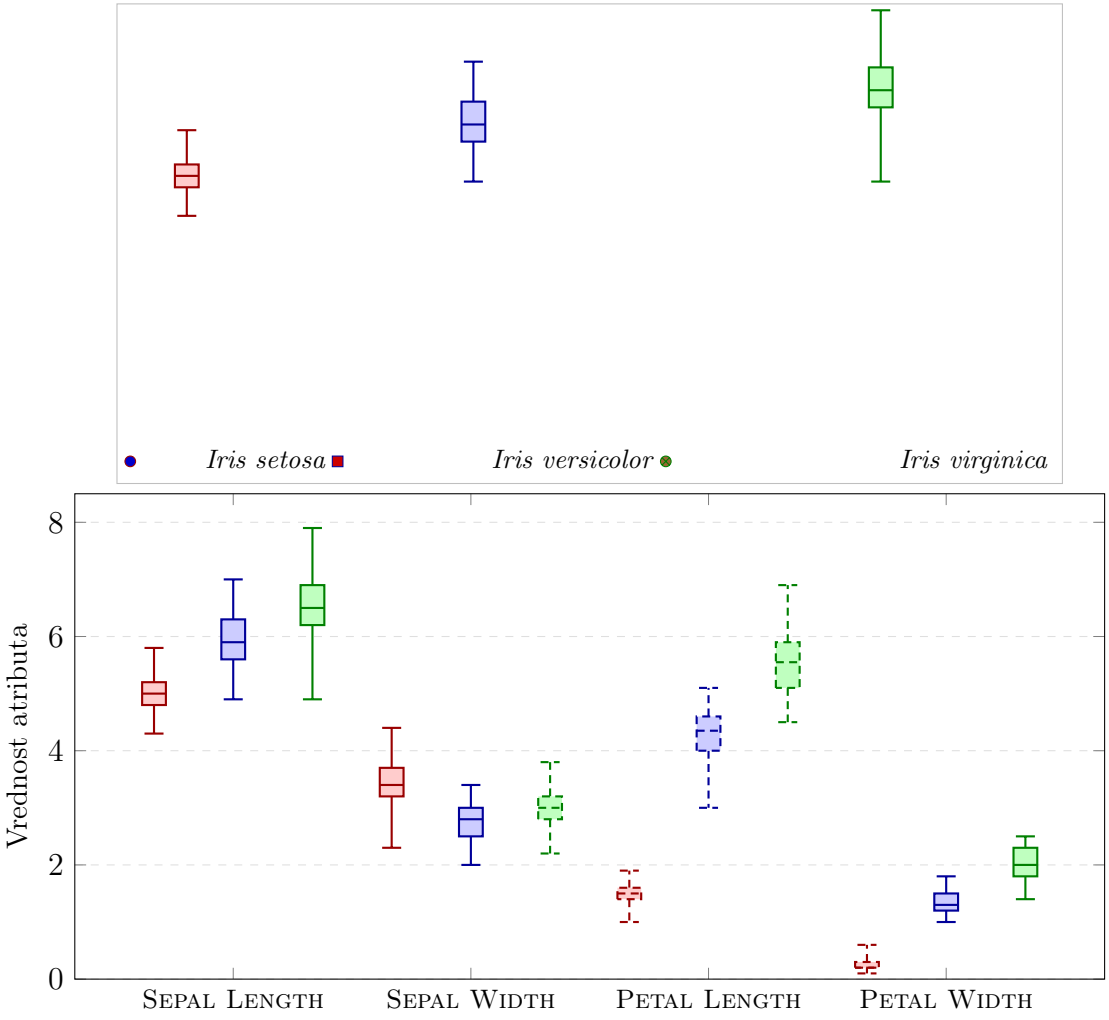
Prednosti boks plota

- sažet i pregledan prikaz raspodele podataka,
- jasno izdvajanje medijane podataka,
- jednostavna identifikacija autlajera,
- pogodnost za poređenje više skupova podataka.

Ograničenja boks plota

- *Ne prikazuje oblik raspodele:* Ne otkriva da li su podaci normalno raspoređeni, uniformni ili imaju više vrhova.
- *Ne prikazuje prosečnu vrednost:* Prikazuje medijanu i kvartile, ali ne prikazuje prosečnu vrednost (nekada se prosečna vrednost može prikazati u boksu isprekidanom linijom, pored medijane).
- *Ne detektuje multimodalnost:* Ne može da identifikuje postojanje više vrhova ili klastera u podacima.
- *Manje koristan za male uzorke:* Pruža ograničene uvide kada je broj podataka veoma mali.

Na slici 5.9 prikazano je poređenje raspodela četiri atributa iz skupa podataka *Iris*, grupisanih po klasama (*setosa*, *versicolor*, *virginica*). boks plotovi omogućavaju da se odmah uoči u kojoj meri se medijane razlikuju i koliko se raspodele preklapaju.



Slika 5.9: Boks plotovi četiri atributa iz skupa podataka *Iris*, grupisani po klasama. Razlike u medijanama i stepen preklapanja raspodela jasno su uočljivi.

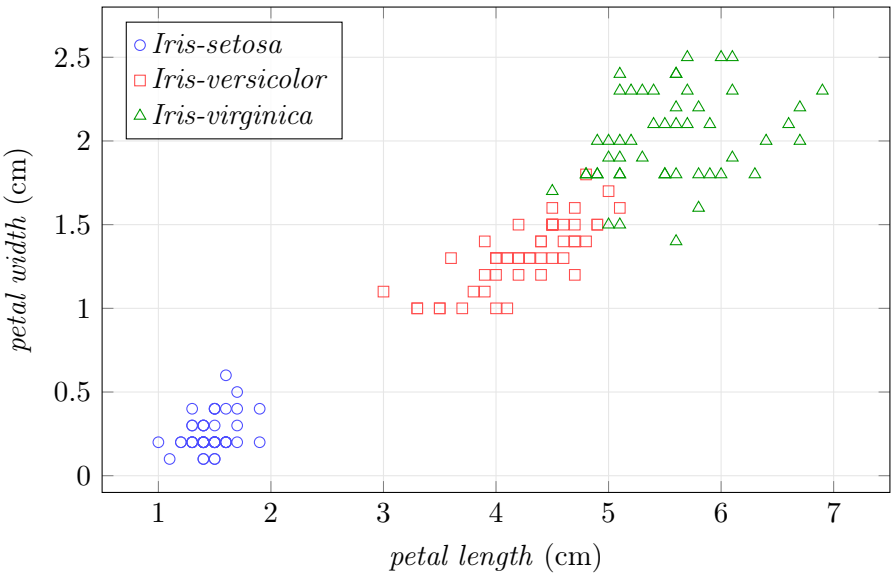
5.3.3 Dijagrami raspršenja

Dijagram raspršenja (engl. *scatter plot*) prikazuje odnos između dve numeričke osobine. Svaki objekat je tačka u ravni: koordinata po osi x odgovara jednoj osobini, a koordinata po osi y drugoj.

Ova tehnika je naročito korisna za:

- uočavanje linearne ili nelinearne povezanosti,
- otkrivanje grupisanja objekata,
- proveru razdvojenosti klasa,
- identifikaciju izdvojenih objekata.

Kada su klase poznate, objekti se mogu označiti različitim bojama ili oblicima markera, čime dijagram postaje koristan i za procenu koliko dati atributi doprinose razdvajanju klasa. Na slici 5.10 prikazan je dijagram raspršenja za atribute *petal length* i *petal width* iz Iris skupa podataka.

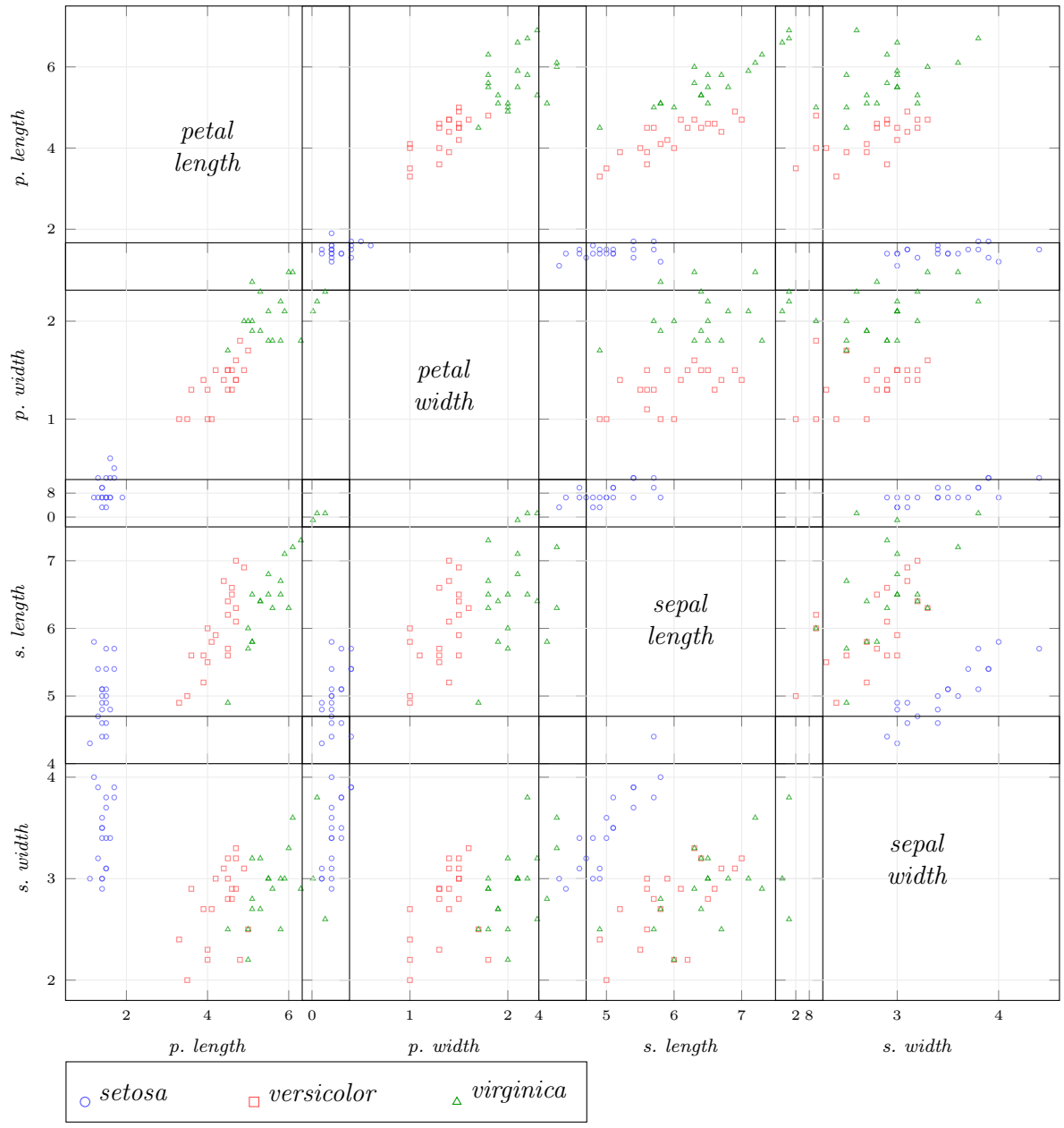


Slika 5.10: Dijagram raspršenja za atribute *petal length* i *petal width* iz Iris skupa podataka. Klasa *setosa* (kružići) jasno je razdvojena od ostale dve klase; klase *versicolor* (kvadratići) i *virginica* (trouglovi) delimično se preklapaju.

Na slici 5.10 uočavaju se tri oblasti: klasa *Iris-setosa* zauzima izolovanu grupu u donjem levom delu (kratke latice, male širine), dok se klase *Iris-versicolor* i *Iris-virginica* prostiru ka gornjem desnom delu sa delimičnim preklapanjem. Oblik markera dodatno olakšava razlikovanje klasa.

Kada skup podataka ima više od dva atributa, korisno je primeniti *matricu dijagrama raspršenja* (engl. *scatter plot matrix*), u kojoj svaka ćelija prikazuje dijagram raspršenja za jedan par atributa. Na slici 5.11 prikazana je takva matrica za sva četiri atributa Iris skupa podataka.

Matrica na slici 5.11 omogućava sistematičan pregled svih parova atributa. Već na prvi pogled uočava se da ćelije koje uključuju osobine latice pokazuju tri jasno razdvojene grupe tačaka, dok parovi sa osobinama čašičnih listića daju znatno veće preklapanje klasa. Ovo je ključna informacija za odabir relevantnih atributa u kasnijoj klasifikaciji ili klasterovanju.



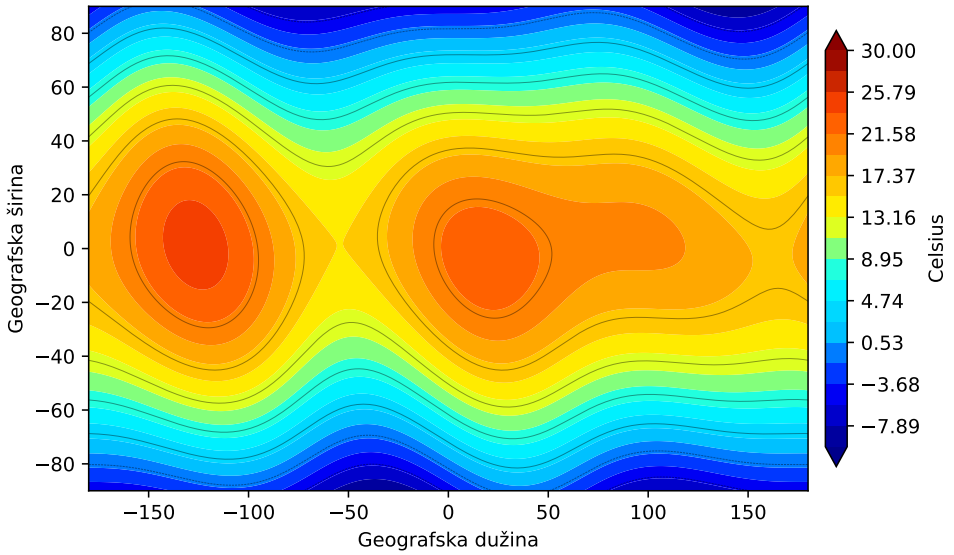
Slika 5.11: Matrica dijagrama raspršenja za četiri atributa Iris skupa podataka. Na dijagonali su nazivi atributa; svaka van-dijagonalna ćelija prikazuje dijagram raspršenja za odgovarajući par atributa. Parovi koji uključuju osobine latice (*petal length*, *petal width*) pokazuju znatno bolju razdvojenost klasa.

5.4 Specijalizovane tehnike vizuelizacije

5.4.1 Konturne šeme

Konturne šeme se koriste za prikaz neprekidne veličine u prostoru. Dve dimenzije određuju položaj u ravni, a treća veličina (npr. temperatura ili pritisak) prikazuje se pomoću konturnih linija i obojenih oblasti.

Konturne linije povezuju tačke jednakih vrednosti, a oblasti između njih prikazuju zone sličnih nivoa. Ovakav prikaz je čest u geografiji, meteorologiji i inženjerstvu.



Slika 5.12: Konturna šema raspodele temperature: najviše temperature su u blizini ekvatora, a niže prema polovima. Konturne linije povezuju tačke jednakih vrednosti.

Na slici 5.12 prikazana je raspodela temperature na površini Zemlje: najviše temperature su u blizini ekvatora, a niže prema polovima. Prednost ovakvog prikaza je što prostorni obrasci postaju očigledni — u tabeli numeričkih vrednosti bili bi teško uočljivi.

5.4.2 Matrice i toplotne mape

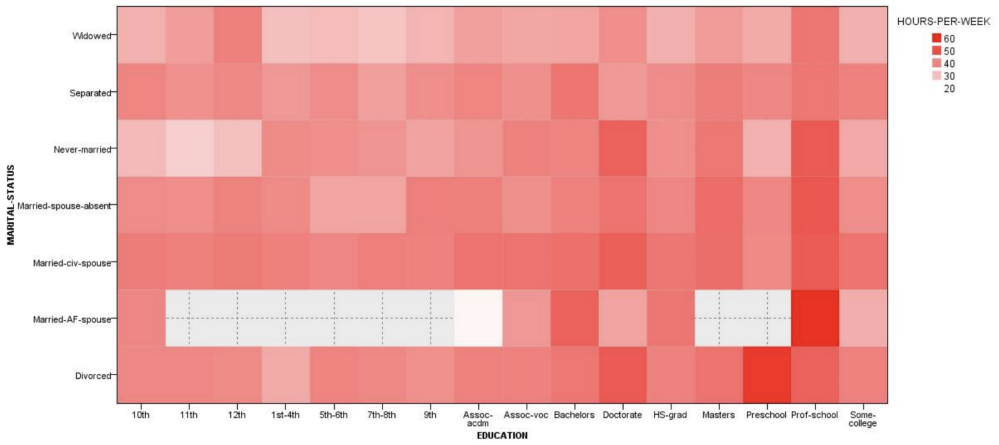
Matrice su pogodne za organizovan prikaz velikog broja vrednosti. Kada se vrednosti kodiraju bojom ili intenzitetom, dobija se toplotna mapa (*heatmap*).

Toplotne mape su naročito korisne kada:

- objekti pripadaju klasama pa se mogu grupisati po redovima ili kolonama,
- želimo da uporedimo više atributa kroz mnogo objekata,
- posmatramo matricu sličnosti ili razlika između objekata.

Atributi se često prethodno normalizuju kako jedan atribut sa velikim opsegom vrednosti ne bi vizuelno dominirao ostalima.

Na slici 5.13 nijansa boje predstavlja veličinu vrednosti u svakoj ćeliji matrice. Ovakav prikaz omogućava brzo uočavanje koncentracija, praznina i obrazaca. Ako su redovi i kolone dobro uređeni, toplotna mapa može veoma jasno pokazati grupisanje ili sličnost objekata.



Slika 5.13: Toplotna mapa kao matrični prikaz vrednosti: nijansa boje predstavlja veličinu vrednosti u svakoj ćeliji (prosečan broj radnih sati nedeljno u zavisnosti od bračnog statusa i nivoa obrazovanja).

5.4.3 Paralelne koordinate

Paralelne koordinate su tehnika za prikaz višedimenzionalnih podataka kod koje se umesto međusobno ortogonalnih osa koristi niz paralelnih osa — po jedna za svaki atribut. Svaki objekat se prikazuje kao poligonalna linija koja povezuje njegove vrednosti na svim osama. Ova tehnika je korisna kada želimo da istovremeno sagledamo više atributa. Način skaliranja osa ima veliki uticaj na preglednost: svaka osa može imati sopstveni opseg prilagođen odgovarajućem atributu, ili sve ose mogu deliti zajedničku skalu. Na slici 5.14 prikazana su dva načina skaliranja paralelnih koordinata za iste podatke. U gornjem prikazu svaka osa ima nezavisan opseg prilagođen datom atributu, dok u donjem sve ose dele zajedničku skalu. Gornji pristup bolje ističe varijabilnost unutar svakog atributa, dok donji omogućava neposredno poređenje apsolutnih vrednosti i jasnije razdvajanje klasa. Ovo ilustruje važnu praktičnu napomenu: kod paralelnih koordinata treba obratiti pažnju ne samo na sadržaj i raspored osa, već i na izbor skaliranja.

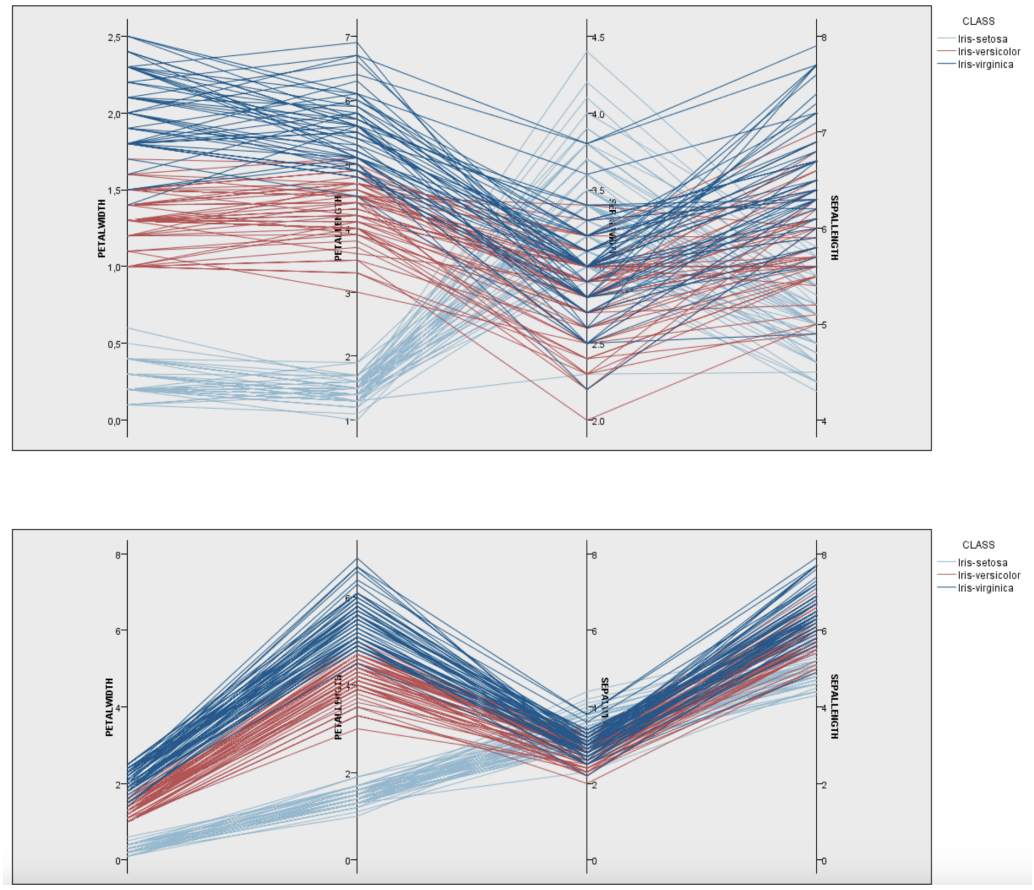
5.4.4 Zvezdasti dijagrami

Zvezdasti dijagrami (engl. *star coordinates*, *radar chart*) prikazuju višedimenzionalne objekte pomoću osa koje se granaju iz centralne tačke. Vrednosti atributa određuju rastojanje od centra na svakoj osi, a dobijene tačke se povezuju u poligon.

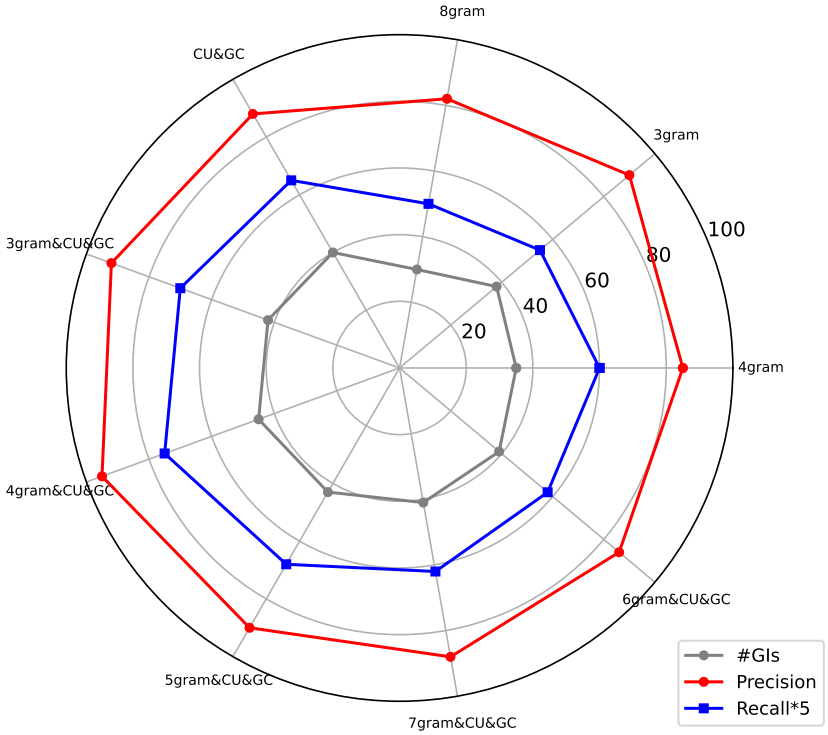
Ovakav prikaz omogućava sagledavanje objekta kao celine kroz oblik koji nastaje kombinacijom atributa. Slični objekti daju slične oblike.

Na slici 5.15 radar dijagram poredi više mera na zajedničkom skupu osa. Prikaz je koristan za brzo poređenje profila metoda ili grupa, ali pri većem broju osa ili objekata slika može postati pretrpana.

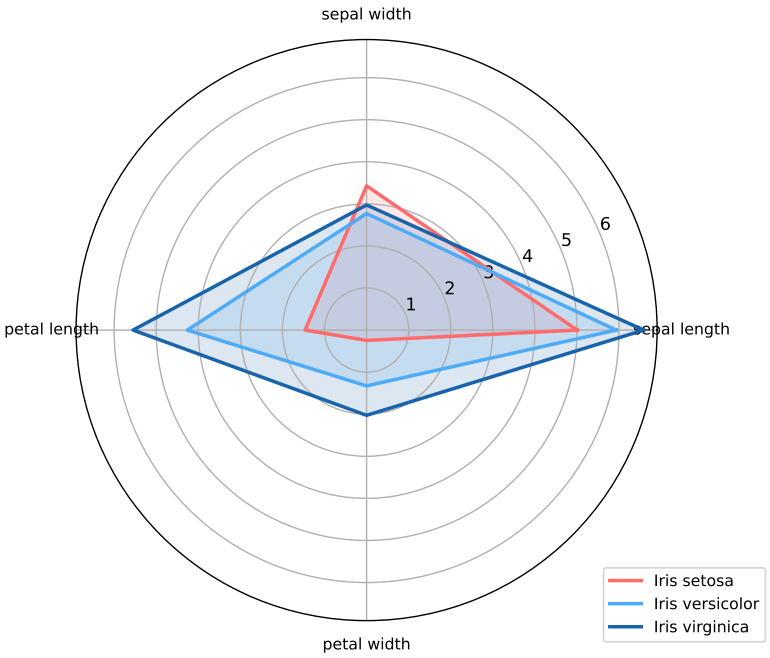
Na slici 5.16 prikazane su agregirane konture za tri klase *Iris* podataka. Različite klase imaju različite oblike u prostoru atributa, što omogućava brzu vizuelnu komparaciju na nivou klase. Nedostatak je gubitak detalja o pojedinačnim objektima.



Slika 5.14: Paralelne koordinate za višedimenzionalne podatke (*Iris* skup). Gore: svaka osa ima sopstveni opseg vrednosti, što omogućava bolji uvid u strukturu pojedinačnih atributa, ali otežava direktno poređenje među njima; dole: sve ose dele zajedničku skalu, čime se jasnije uočavaju relativni odnosi između atributa i separacija klasa.



Slika 5.15: Zvezdasti dijagram za poređenje više mera: svaka osa predstavlja jednu kombinaciju metoda, a linije povezuju vrednosti tri metrike (#GIs, Precision, Recall×5).



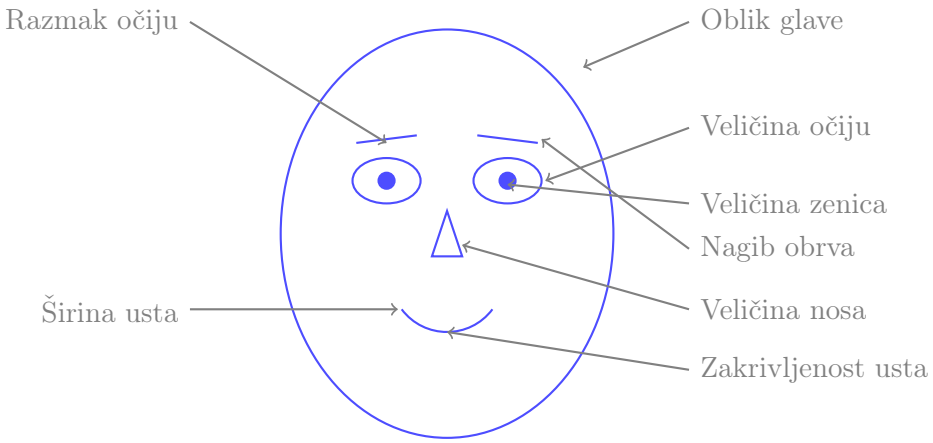
Slika 5.16: Radar prikaz klasa *Iris* skupa podataka: agregirane konture za tri klase pokazuju različite oblike u prostoru atributa.

5.4.5 Černofljeva lica

Kod Černovljevih lica (engl. *Chernoff faces*) svaki atribut se preslikava na karakteristiku lica — oblik glave, veličinu očiju, širinu nosa, zakrivljenost usta i slično. Ideja počiva na činjenici da čovek izuzetno lako uočava razlike među licima.

Ova tehnika omogućava intuitivno poređenje objekata, ali nije pogodna za veliki broj instanci. Osim toga, nisu sve preslikane osobine jednako uočljive, pa tumačenje može biti subjektivno.

Na slici 5.17 prikazano je preslikavanje atributa na osobine lica, dok slika 5.18 daje primere Černovljevih lica za devet objekata iz Iris skupa podataka — po tri iz svake klase.



Slika 5.17: Preslikavanje atributa podataka na osobine Černovljevog lica. Svaka vizuelna karakteristika — oblik glave, veličina očiju, nagib obrva, veličina nosa, zakrivljenost i širina usta — kodira vrednost jednog atributa.

Prikaz na slici 5.18 pokazuje da se klase razlikuju po opštem "izgledu" lica: objekti iste klase daju sličan oblik, dok se objekti iz različitih klasa vizuelno razlikuju. Ipak, ovakav pristup je pogodan samo za manji broj objekata i kvalitativno poređenje.

5.4.6 Voronojevi dijagrami

Za dati skup tačaka, Voronojev dijagram deli prostor na regione tako da svaka lokacija unutar jednog regiona ima najbliži odgovarajući generator (čvor) u odnosu na sve ostale.

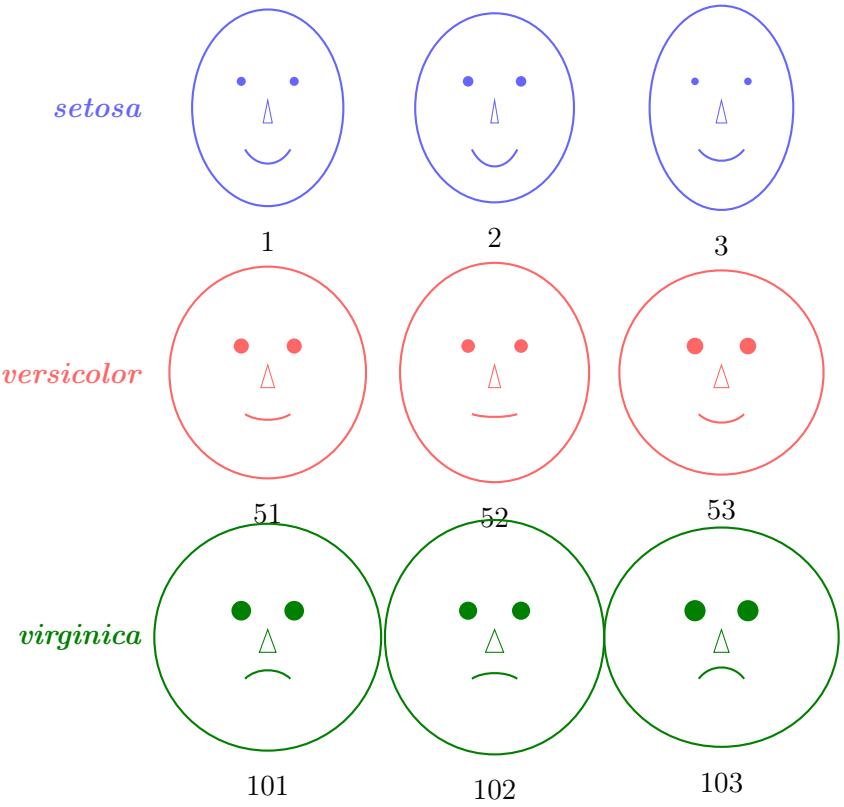
Ova tehnika je korisna za prikaz prostorne podele, zona uticaja ili susednih oblasti. Primene obuhvataju geometriju, prostornu analizu, planiranje mreža i analizu lokacija. Na slici 5.19 prikazana su dva primera Voronojeve podele za različite konfiguracije tačaka.

Na slici 5.19 svaki region pripada jednoj tački i obuhvata sve lokacije koje su joj najbliže. Prikaz jasno ilustruje kako se prostor prirodno deli prema blizini centara — sa većim brojem generatora, podela postaje finija.

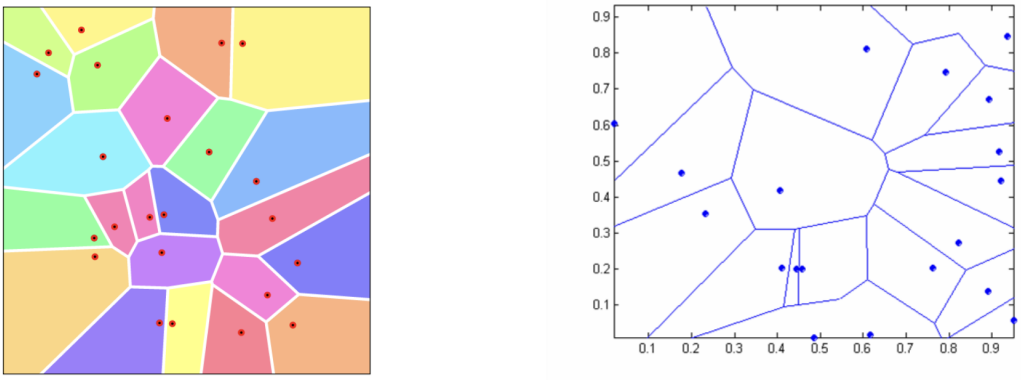
5.5 Kombinovani i uporedni prikazi

5.5.1 Kombinacija slika i tabela

U praksi je često korisno objediniti više oblika prikaza: grafikon daje opšti trend, a tabela omogućava precizno očitavanje vrednosti. Takva kombinacija po pravilu pruža više informacija nego bilo koji pojedinačni prikaz.

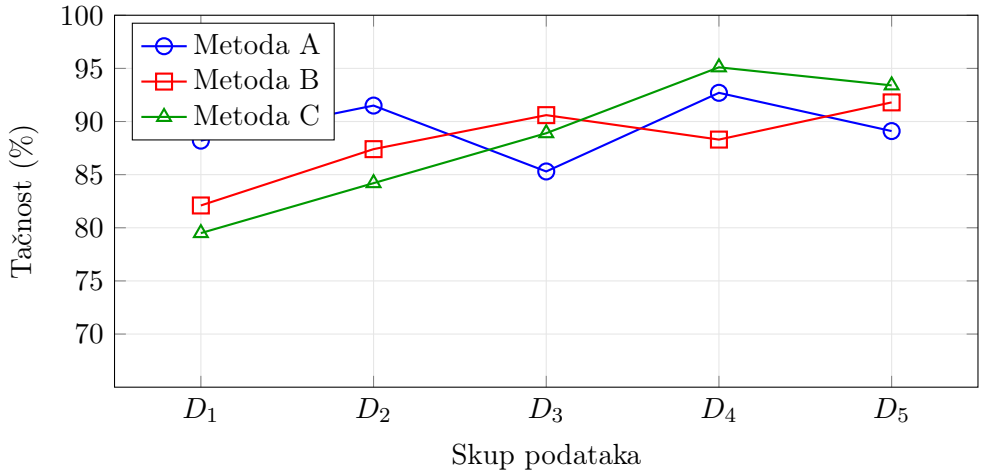


Slika 5.18: Černofljeva lica za devet objekata iz Iris skupa podataka (po tri iz svake klase). Klasa *setosa* ima manje glave i oči, klasa *versicolor* je srednje veličine, dok *virginica* ima najveće oblike. Usta se postepeno menjaju od nasmejanog oblika (*setosa*) do blago nadole savijenog (*virginica*).



Slika 5.19: Voronojevi dijagrami za različite konfiguracije tačaka.

Na slici 5.20 prikazan je linijski grafikon performansi tri metode na pet skupova podataka, a u tabeli 5.5 date su tačne numeričke vrednosti.



Slika 5.20: Linijski grafikon tačnosti tri metode klasifikacije na pet skupova podataka. Metoda C pokazuje uzlazni trend, dok metode A i B osciluju.

Tabela 5.5: Tačnost klasifikacije (%) za tri metode na pet skupova podataka — numeričke vrednosti sa slike 5.20.

Metoda	D_1	D_2	D_3	D_4	D_5
Metoda A	88,2	91,5	85,3	92,7	89,1
Metoda B	82,1	87,4	90,6	88,3	91,8
Metoda C	79,5	84,2	88,9	95,1	93,4

Na slici 5.20 se odmah uočava da metoda C ima uzlazni trend i postiže najbolji rezultat na D_4 i D_5, dok tabela 5.5 omogućava precizno poređenje — na primer, razlika između metoda A i C na D_4 iznosi tačno 2,4 procentna poena. Ovo je dobar primer međusobne dopune vizuelnog i tabelarnog prikaza.

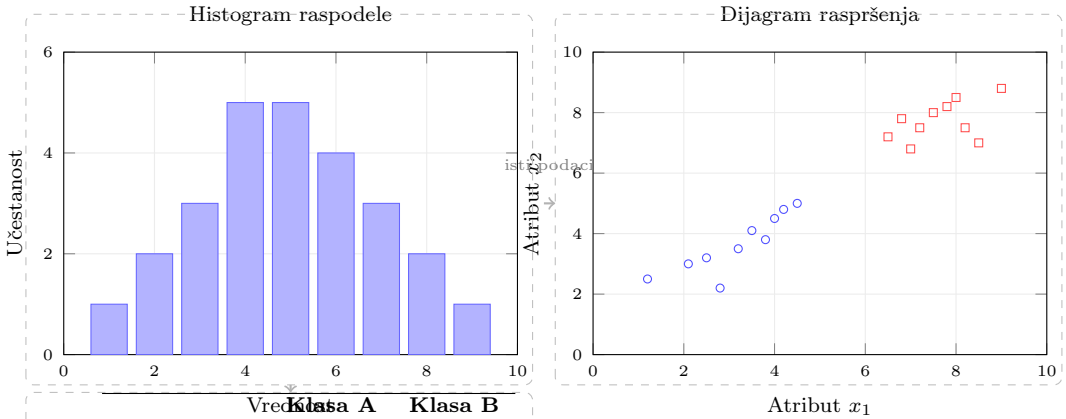
5.5.2 Koordinisani višestruki prikazi

U složenim analizama jedan prikaz često nije dovoljan. Tada se koriste koordinisani višestruki prikazi — više različitih vizuelizacija istog skupa podataka iz različitih uglova. Na slici 5.21 prikazan je primer takvog interfejsa: histogram raspodele, dijagram raspršenja i tabela sumarnih statistika za isti skup podataka.

Ovakvi sistemi su posebno korisni u interaktivnoj analizi podataka, gde korisnik može da kombinuje globalni pregled, lokalne detalje i dodatne statističke informacije.

5.6 Vizuelizacija mreža

Kada podaci opisuju odnose među objektima, prirodno je koristiti mrežni prikaz: objekti se predstavljaju čvorovima, a odnosi granama. Mrežne vizuelizacije su važne u analizi društvenih mreža, bioloških interakcija, računarskih sistema i sličnih struktura.



	Klasa A	Klasa B
Broj objekata	10	10
$\bar{x}_1$	3,18	7,65
$\bar{x}_2$	3,66	7,73
$\sigma(x_1)$	0,98	0,79

Slika 5.21: Koordinisani višestruki prikazi: histogram (gore levo), dijagram raspršenja (gore desno) i sumarna statistika (dole levo) prikazuju isti skup podataka sa dve klase iz različitih uglova. Korisnik može istovremeno sagledati raspodelu, prostorni raspored i numerički rezime.

5.6.1 Čvorovi i grupe u mreži

Na slici 5.22 prikazana je mreža u kojoj boja čvorova označava pripadnost grupi. Raspored čvorova pomaže da se uoče gusti lokalni klasteri i centralni čvorovi koji povezuju više delova mreže.

5.6.2 Kružni mrežni prikaz

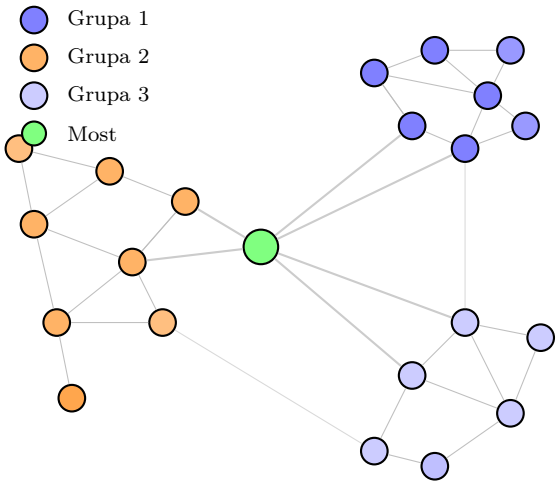
U kružnom mrežnom prikazu čvorovi su raspoređeni po obodu kruga, a veze su nacrtane kroz unutrašnjost. Takav raspored može biti veoma efektan za prikaz povezanosti između grupa, mada kod velikog broja veza dolazi do preklapanja i smanjene čitljivosti. Na slici 5.23 prikazan je ovakav prikaz.

5.7 Dinamički i praktični aspekti vizuelizacije

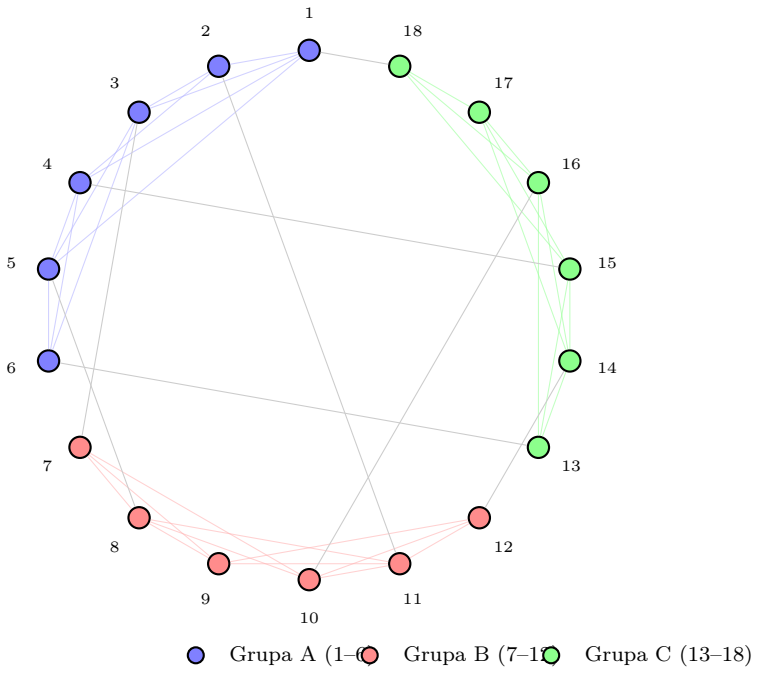
5.7.1 Animacije

Animacija prikazuje sukcesivna stanja podataka kroz vreme ili neku drugu dimenziju. Posebno je korisna kod prostorno-vremenskih podataka, kada želimo da pratimo promenu pojava kroz niz uzastopnih trenutaka.

Iako je animacija vizuelno privlačna i pogodna za uočavanje promena, pažljivo uređen statičan niz slika ponekad omogućava temeljniju analizu jer se korisnik može proizvoljno dugo zadržati na svakom prikazu.



Slika 5.22: Mrežni prikaz sa tri grupe čvorova (klastera) i centralnim čvorom koji ih povezuje. Gustina veza unutar grupa je znatno veća nego između grupa, što je tipična struktura zajednica u mreži.



Slika 5.23: Kružni mrežni prikaz sa 18 čvorova u tri grupe. Obojene veze unutar grupa su gušće, dok sive linije kroz centar kruga predstavljaju međugrupne veze. Ovakav raspored jasno prikazuje strukturu zajednica, ali kod većeg broja veza čitljivost opada.

5.7.2 Praktične smernice za dobru vizuelizaciju

Kvalitetna vizuelizacija pre svega mora biti informativna, jasna i verna podacima. Pri oblikovanju prikaza korisno je voditi računa o sledećem:

- najvažniji odnosi treba da budu vizuelno najistaknutiji,
- prikaz ne treba pretrpavati suvišnim elementima,
- boje, oblici i oznake moraju biti dosledni,
- skale i proporcije moraju tačno odražavati podatke,
- prikaz treba prilagoditi tipu podataka i cilju analize.

Grafikon koji otežava čitanje podataka ili navodi na pogrešan zaključak nije uspešan, bez obzira na estetski utisak.

5.8 Zaključak

Vizuelizacija je jedan od osnovnih alata u istraživanju podataka. Njena uloga je da omogući brzo i intuitivno razumevanje strukture podataka — raspodela vrednosti, odnosa između atributa, grupisanja, odstupanja, mrežnih i prostornih obrazaca.

Različite tehnike odgovaraju različitim vrstama podataka i analitičkim ciljevima: histogrami i boks plotovi daju uvid u raspodelu jednog atributa; dijagrami raspršenja i njihove matrice pomažu pri analizi odnosa između atributa; paralelne koordinate i zvezdasti dijagrami služe za višedimenzionalne podatke; konturne šeme, toplotne mape i mrežne vizuelizacije podržavaju specifične tipove strukture.

Najvažnija poruka jeste da uspešna vizuelizacija ne nastaje slučajno. Potrebno je pažljivo izabrati tip prikaza, način preslikavanja podataka u grafičke elemente, raspored elemenata i odgovarajući nivo detalja. Dobro osmišljena vizuelizacija postaje snažno sredstvo za razumevanje i tumačenje podataka.

Glava 6

Klasifikacija

Ljudi svakodnevno rešavaju zadatke klasifikacije. Na primer, kada uđu u prodavnicu, odmah razlikuju voće od povrća ili sveže od pokvarenog. U razgovoru, prepoznaju da li je nečiji ton prijateljski ili neprijateljski. Lekar na osnovu simptoma procenjuje kojoj bolesti oni pripadaju. Student pri učenju razdvaja pojmove na važne i manje važne. Vozač razlikuje saobraćajne znakove i reaguje u skladu sa njihovim značenjem. Čak i dete uči da razlikuje životinje — šta je pas, a šta mačka. Kada broj objekata naraste iznad onoga što čovek može ručno da obradi, potreban nam je automatizovan sistem koji će na osnovu poznatih primera naučiti da razvrstava nove, ranije neviđene slučajeve. Upravo to je suština klasifikacije.

Ovo poglavlje gradi razumevanje klasifikacije sistematski. Počinjemo od formalnih definicija, a zatim koristimo stablo odlučivanja kao ilustrativni okvir za razmatranje pitanja koja se javljaju kod svakog klasifikatora: izbor kriterijuma podele, kontrola složenosti modela, i objektivna procena kvaliteta. Posebnu pažnju posvećujemo problemima preprilagođenosti i potprilagođenosti, selekciji modela, evaluaciji putem validacionih i test skupova, kao i uobičajenim zamkama u praksi. Na kraju poglavlja uvodimo klasifikatore na bazi pravila, metod najbližih suseda i naivni Bajesov klasifikator.

6.1 Osnovni pojmovi

Podaci za klasifikaciju organizovani su kao tabela instanci. Svaka instanca (slog, zapis) opisana je parom

$$(\mathbf{x}, y), \quad \mathbf{x} = (x^1, x^2, \ldots, x^k),$$

gde $\mathbf{x}$ predstavlja skup atributa (osobina), a y je **oznaka klase**, tj. kategorička promenljiva koja određuje kojoj grupi instanca pripada.

Zadatak klasifikacije je da se odredi funkcija (klasifikacioni model)

$$f\colon \mathbf{x} \mapsto y$$

koja svakom vektoru atributa $\mathbf{x}$ pridružuje jednu od predefinisanih vrednosti ciljne promenljive y.

Atributi mogu biti različitog tipa (numerički, kategorički, binarni, redni), dok je za klasifikaciju neophodno da je ciljna promenljiva y kategorička. U suprotnom, kada je y numerička (neprekidna), reč je o regresiji.

Klasifikacija je, dakle, postupak izgradnje ciljne funkcije $f\colon \mathbf{x} \mapsto y$ koja preslikava skup atributa u jednu od konačno mnogo unapred zadatih klasa. Funkciju f nazivamo **klasifikacioni model** ili **klasifikator**.

Klasifikacija se javlja u izuzetno širokom spektru primena:

- **Ćelije tumora** – na osnovu karakteristika ćelija dobijenih biopsijom, klasifikator određuje da li je tumor benigni ili maligni. Atributi mogu uključivati veličinu ćelije, oblik jedra, teksturu tkiva i slično.
- **Ispravnost korišćenja kreditnih kartica** – banke koriste klasifikatore da u realnom vremenu razlikuju legitimne transakcije od potencijalnih prevara, na osnovu iznosa, lokacije, vremena i obrazaca potrošnje.
- **Predviđanje sekundarne strukture proteina** – u bioinformatici, klasifikatori predviđaju da li će segment aminokiselina formirati α-heliks, β-ravan ili nestrukturirani region.
- **Određivanje tipa tekstova** – klasifikator automatski razvrstava novinarske članke u kategorije poput sporta, politike, ekonomije, kulture i slično, na osnovu reči koje se javljaju u tekstu.
- **Određivanje tipa galaksija** – na osnovu teleskopskih snimaka, klasifikator razvrstava galaksije u morfološke klase: spiralne, eliptične, nepravilne, spiralne sa prečkom (barred spiral) i druge.

Kada postoje samo dve klase, govorimo o **binarnoj klasifikaciji**; kada ih je više, o **višeklasnoj klasifikaciji**. Klasifikacija se razlikuje od regresije po tome što ciljna promenljiva nije neprekidna. Takođe, klasifikacija je manje pogodna za redne kategorije (npr. nizak/srednji/visok rizik) jer ne uzima u obzir prirodan poredak među vrednostima klase. Slično se ignorišu i drugi tipovi odnosa među kategorijama, poput hijerarhije potklasa i natklasa. Za ovakve primene postoje posebne metode klasifikacije.

6.1.1 Opšti okvir: indukcija i dedukcija

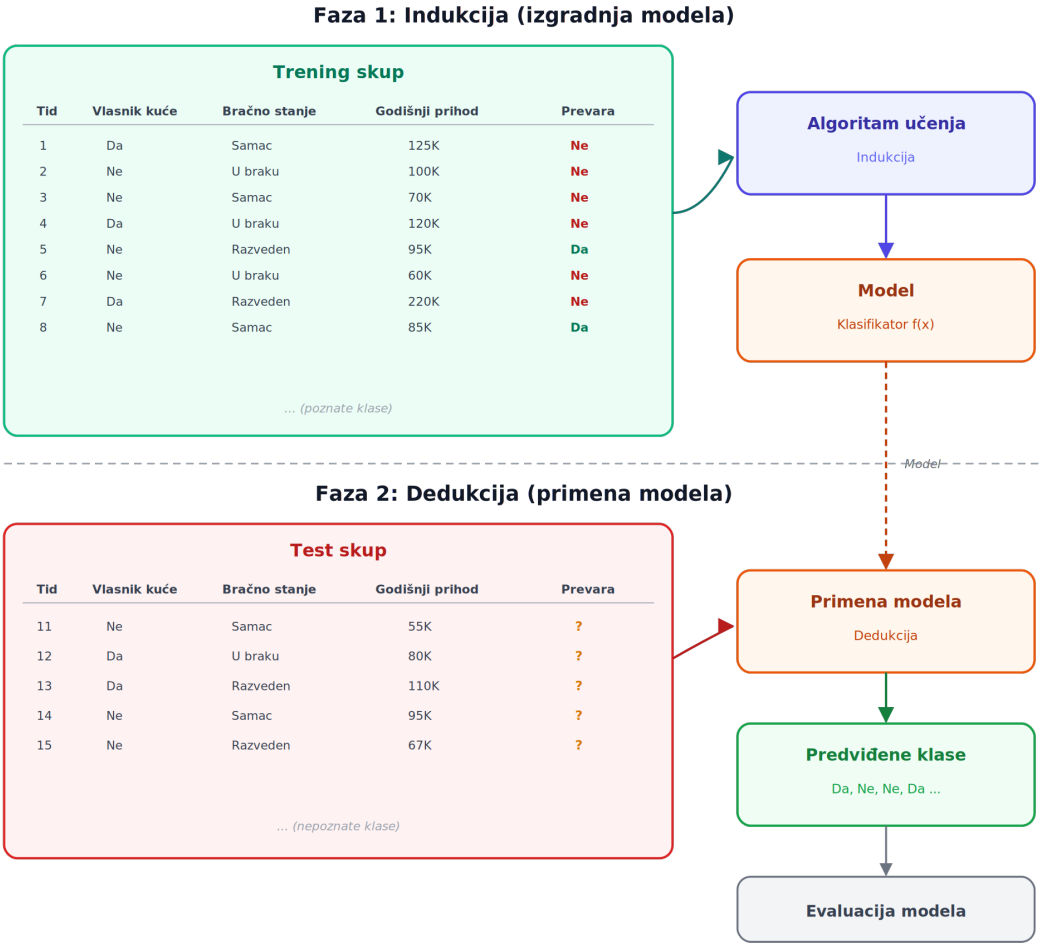
Postupak klasifikacije se odvija u dve faze:

1. **Indukcija (izgradnja modela):** Algoritam učenja gradi model na osnovu **trening skupa**, kolekciju instanci sa poznatim klasama. Kažemo da u ovoj fazi dolazi do obučavanja modela.
2. **Dedukcija (primena modela):** Izgrađeni model se primenjuje na **test skup**, instance čije klase nisu bile dostupne tokom obučavanja modela, da bi se procenilo koliko dobro model **generalizuje** na nove podatke.

Slika 6.1 ilustruje ovaj dvofazni postupak. U gornjoj polovini prikazan je trening skup sa poznatim klasama koji ulazi u algoritam učenja (faza indukcije), čiji rezultat je klasifikacioni model. U donjoj polovini, test skup sa nepoznatim klasama prolazi kroz izgrađeni model (faza dedukcije), koji na osnovu naučenih zakonitosti predviđa klase novih instanci.

Razdvajanje trening i test podataka je od suštinskog značaja. Ako bismo kvalitet modela merili na istim podacima na kojima je učio, procena bi bila preterano optimistična jer model može da "zapamti" specifičnosti trening skupa umesto da nauči opšte zakonitosti.

Performanse modela se kvantifikuju poređenjem predviđenih klasa sa stvarnim klasama test instanci. Ova informacija se kondenzuje u **matricu konfuzije** (eng. *confusion matrix*), o kojoj detaljno govorimo u Odeljku **??**.



Slika 6.1: Opšti okvir klasifikacije: faza indukcije (izgradnja modela na osnovu trening skupa) i faza dedukcije (primena modela na test skup radi predviđanja klasa novih instanci).

Primer klasifikacije je poreska uprava koja želi da automatizuje otkrivanje sumnjivih poreskih prijava. Svaka istorijska prijava opisana je atributima (iznos povraćaja, bračni status, oporezivi prihod) i oznakom da li je utvrđena prevara. Na trening skupu takvih prijava gradi se klasifikator, koji se zatim primenjuje na nove prijave čiji status nije poznat.

Klasifikacija se u praksi realizuje različitim tehnikama, koje se mogu grubo podeliti na osnovne metode i metode ansambla.

Osnovne metode klasifikacije obuhvataju: stabla odlučivanja, koja modeluju odluke kroz hijerarhiju testova nad atributima, klasifikatore na bazi pravila, koji koriste skup pravila oblika „ako–onda", neuronske mreže, koje uče složene nelinearne zavisnosti, statistički zasnovane metode, poput naivnog Bajesovog klasifikatora, metod potpornih vektora (SVM), koje određuju optimalnu granicu razdvajanja klasa, metod k najbližih suseda (k-NN), koji klasifikuje na osnovu sličnosti sa postojećim instancama.

Metode ansambla kombinuju više modela kako bi se poboljšale performanse: pojačavanje (engl. boosting), gde se modeli treniraju sekvencijalno sa fokusom na teške primere, pakovanje (engl. bagging), gde se modeli treniraju nezavisno na različitim podskupovima podataka, nasumična šuma (engl. random forest), koja predstavlja skup stabala odlučivanja treniranih na slučajnim podacima i atributima.

6.2 Stablo odlučivanja

Stablo odlučivanja je među najintuitivnijim klasifikatorima. Njegova osnovna ideja podseća na igru u kojoj postavljamo niz pitanja o atributima instance, a svaki odgovor nas vodi bliže konačnoj odluci o njenoj klasi.

Formalno, stablo se sastoji od:

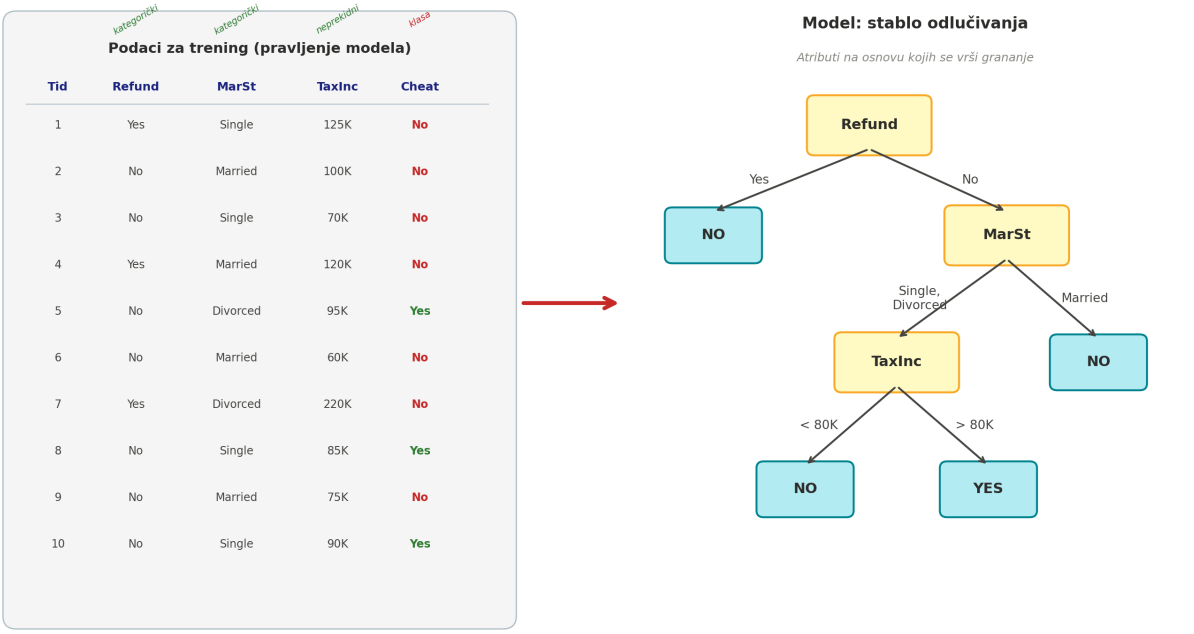
- **Korena**, polaznog čvora koji nema roditeljski čvor, iz koga polaze dve ili više grana.
- **Unutrašnjih čvorova**, svaki sadrži **test uslov** nad jednim atributom i ima tačno jednog roditelja i dvoje ili više dece.
- **Listova**, krajnjih čvorova bez potomaka; svaki list nosi oznaku jedne klase.

Klasifikacija nove instance teče od korena ka dnu: u svakom unutrašnjem čvoru proveravamo vrednost odgovarajućeg atributa, pratimo granu koja odgovara rezultatu testa, i ponavljamo dok ne dođemo do lista. Klasa lista je konačna predikcija.

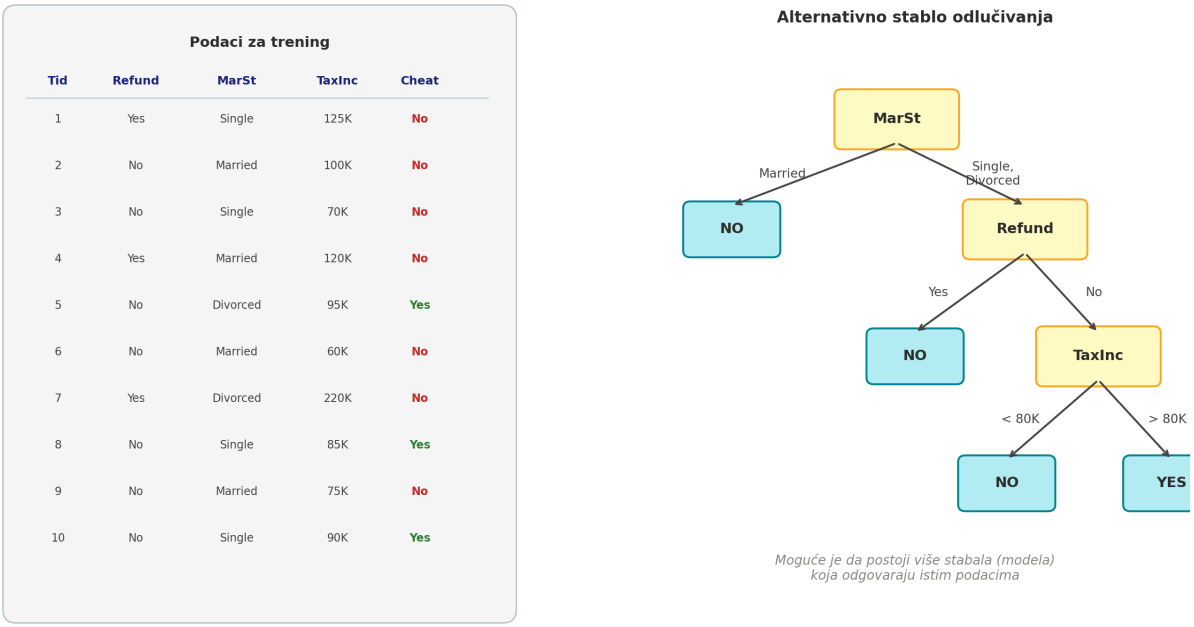
Slika 6.2 prikazuje primer stabla odlučivanja za problem detekcije poreskih prevara. Trening podaci (levo) opisuju po deset poreskih prijava sa atributima *Refund* (kategorički), *MarSt* (kategorički), *TaxInc* (neprekidni) i oznakom klase *Cheat*. Rezultujuće stablo (desno) koristi upravo ove atribute kao test uslove u unutrašnjim čvorovima, dok listovi nose konačnu predikciju.

Važno je uočiti da za iste trening podatke može postojati više različitih stabala koja ispravno klasifikuju sve trening instance. Slika 6.3 prikazuje alternativno stablo koje koristi iste atribute ali u drugačijem redosledu, koren je *MarSt* umesto *Refund*. Izbor "najboljeg" stabla zavisi od kriterijuma koji se koristi pri odabiru atributa za podelu, o čemu detaljno govorimo u narednim odeljcima.

Za razlikovanje sisara od ostalih kičmenjaka, koren stabla može testirati telesnu temperaturu (topla/hladna krv). Ako je krv hladna, instanca sigurno nije sisar. Ako je topla, sledeće pitanje može biti "Da li ženka rađa žive mladunce?", ako da, klasa je sisar; ako ne, klasa je ne-sisar (uz izuzetke poput kljunara).

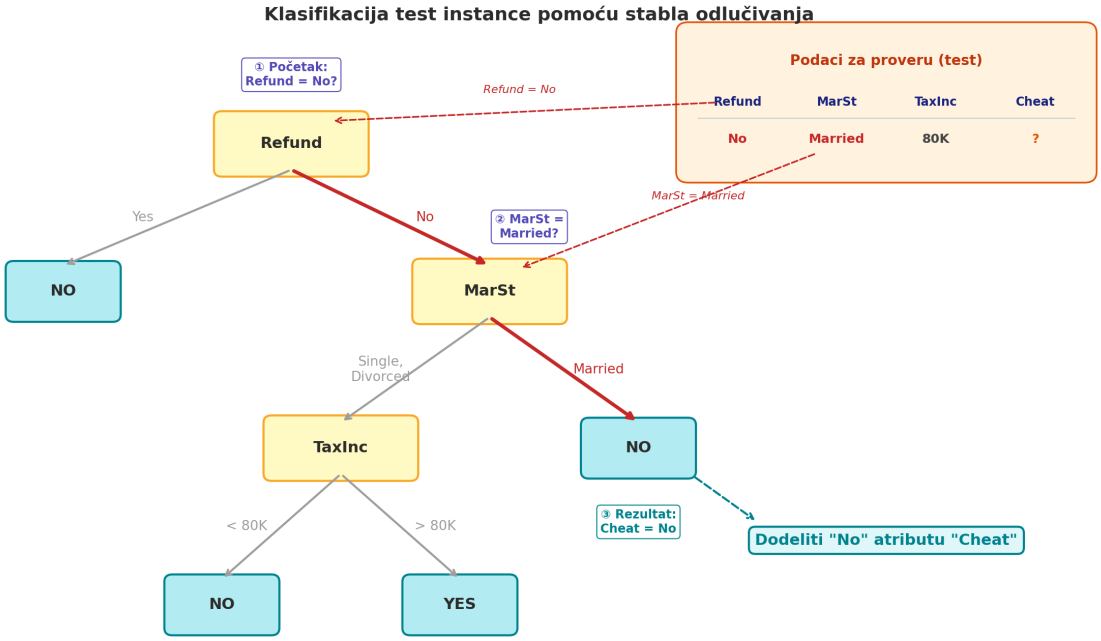


Slika 6.2: Primer stabla odlučivanja za detekciju poreskih prevara. Levo: trening podaci sa označenim tipovima atributa. Desno: stablo odlučivanja izgrađeno na osnovu tih podataka.



Slika 6.3: Alternativno stablo odlučivanja za iste trening podatke. Različit redosled testiranja atributa daje stablo drugačije strukture, ali sa istim krajnjim klasifikacijama.

Slika 6.4 ilustruje postupak klasifikacije nove, neoznačene instance. Test instanca ima vrednosti *Refund* = No, *MarSt* = Married, *TaxInc* = 80K. Polazeći od korena, prvo proveravamo *Refund*: vrednost je No, pa pratimo desnu granu do čvora *MarSt*. Zatim proveravamo *MarSt*: vrednost je Married, pa pratimo granu "Married" i dolazimo do lista sa klasom NO. Konačna predikcija je *Cheat* = No.



Slika 6.4: Klasifikacija test instance pomoću stabla odlučivanja. Crvenom bojom označena je putanja kroz stablo: koren (*Refund* = No) → *MarSt* → grana "Married" → list NO. Numerisani koraci pokazuju redosled provera.

6.2.1 Algoritam za izgradnju stabla

Konstrukcija stabla odlučivanja postavlja niz međusobno povezanih izazova:

- Kako odabrati atribut(e) po kome se vrši podela?
- Kako formulisati upit za različite tipove atributa?
- Kako odrediti najbolju podelu?
- Na koji način induktivno primeniti prethodne kriterijume na stablo po dubini?
- Kada stati sa konstrukcijom stabla?
- Kako se nositi sa nedostajućim vrednostima?
- Koji je kriterijum za procenu greške u generalizaciji?

- Kako uzeti u obzir cenu i performanse modela?

Odgovori na ova pitanja doveli su do razvoja čitavog niza algoritama. Najpoznatiji su:

- **ID3** (Iterative Dichotomiser 3) – koristi informacioni dobitak (entropiju) za izbor atributa podele; radi samo sa kategoričkim atributima.
- **C4.5** – naslednik ID3 koji uvodi odnos dobitka (gain ratio), podržava neprekidne atribute, nedostajuće vrednosti i post-potkresivanje.
- **C5.0** – komercijalna nadogradnja C4.5 sa poboljšanjima u efikasnosti i podrškom za pojačavanje (boosting).
- **CART** (Classification And Regression Trees) – koristi isključivo binarne podele i Ginijev indeks; podržava i regresiju.
- **CHAID** (CHi-squared Automatic Interaction Detection) – koristi χ^2 test za izbor podele; tretira nedostajuće vrednosti kao zasebnu kategoriju.
- **Exhaustive CHAID** – varijanta CHAID-a koja ispituje sve moguće podele pre odabira.
- **QUEST** (Quick, Unbiased, Efficient Statistical Trees) – koristi statističke testove za nepristrasnu selekciju atributa.
- **SLIQ** (Supervised Learning In Quest) i **SPRINT** (Scalable PaRallelizable INduction and decision Trees) – algoritmi dizajnirani za efikasan rad sa velikim skupovima podataka.

Atribut za podelu materijala se bira strategijom pohlepe (eng. *greedy*): u svakom čvoru, algoritam bira onaj atribut koji optimizuje odabrani kriterijum kvaliteta. Ovo zahteva donošenje tri ključne odluke: kako podeliti slogove i koja je podela najbolja, kako navesti uslove za testiranje atributa, i kada stati sa deobom.

Uprkos razlikama u detaljima, gotovo svi ovi algoritmi dele zajedničku rekurzivnu strukturu zasnovanu na Hantovom pristupu.

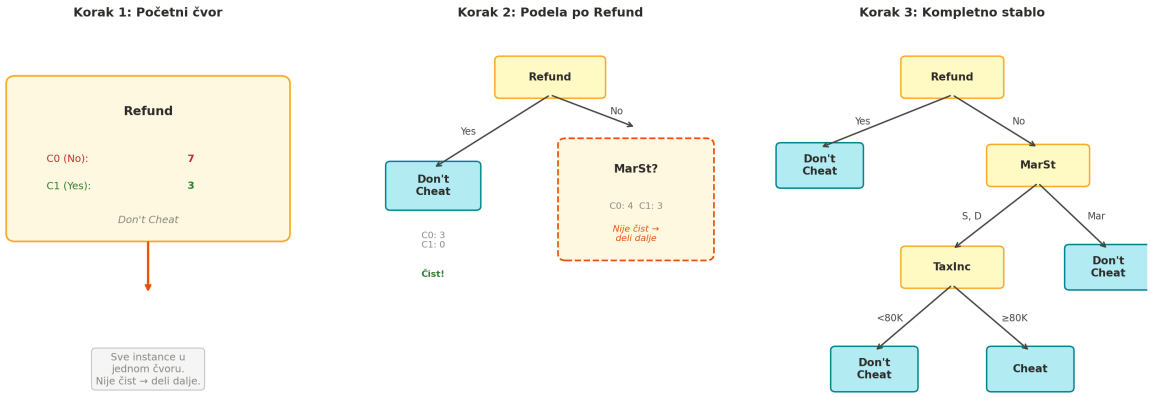
Prostor svih mogućih stabala nad datim skupom atributa je ogroman, pronalaženje globalnog optimuma je NP-kompletan problem. U praksi se koriste pohlepni algoritmi koji u svakom koraku biraju lokalno najbolji atribut za podelu.

Hantov pristup

Gotovo svi poznati algoritmi za stabla odlučivanja (ID3, C4.5, CART) zasnivaju se na rekurzivnoj šemi koju je predložio Hant (Hunt). Neka D_t označava skup instanci koje su dostigle čvor t. Postupak je sledeći:

1. Ako sve instance u D_t pripadaju istoj klasi, čvor t postaje list označen tom klasom.
2. U suprotnom, ako instance u D_t pripadaju različitim klasama, čvoru t se dodaju potomci na osnovu atributa čija podela razdvaja D_t na najbolji način prema unapred utvrđenoj meri. Za svaku moguću vrednost (ili opseg) tog atributa kreira se potomak, instance se raspodeljuju prema ishodu testa, i postupak se rekurzivno primenjuje na svako dete.

Slika 6.5 prikazuje ovaj postupak na primeru podataka o poreskim prevarama. U prvom koraku, sve instance se nalaze u jednom čvoru koji nije čist (sadrži obe klase). Algoritam bira atribut *Refund* za podelu. U drugom koraku, grana "Yes" vodi do čistog čvora (samo klasa No), koji postaje list. Grana "No" vodi do čvora koji još uvek nije čist, pa se postupak nastavlja. U trećem koraku, stablo je kompletno izgrađeno.



Slika 6.5: Ilustracija Hantovog algoritma: rekurzivna konstrukcija stabla odlučivanja u tri koraka. Isprekidani okvir označava čvor koji zahteva dalju podelu.

Dva su ključna projektna pitanja: *koji kriterijum koristiti za izbor atributa podele?* i *kada zaustaviti rast stabla?* Odgovori na ova pitanja čine suštinsku razliku između različitih algoritama.

Granični slučajevi

Mogu se javiti situacije koje osnovni algoritam ne pokriva:

1. Dete-čvor ostane prazan jer nijedna trening instanca nema tu vrednost atributa. Rešenje: proglasiti ga listom i dodeliti mu najčešću klasu roditelja.

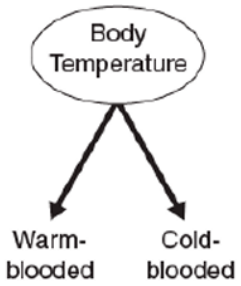
2. Sve instance u čvoru imaju iste vrednosti atributa ali različite klase (npr. zbog šuma ili nedostajućih atributa). Rešenje: proglasiti čvor listom i dodeliti mu većinsku klasu. Ovakva situacija se pojavljuje često u praksi, bilo zbog grešaka u podacima, bilo zbog toga što se nismo služili svim atributima ili nam nisu bili dostupni.

6.2.2 Formulisanje test uslova za različite tipove atributa

Binarni atributi. Tačno dva moguća ishoda (Da/Ne, Toplo/Hladno), čvor se deli na dve grane (slika 6.6).

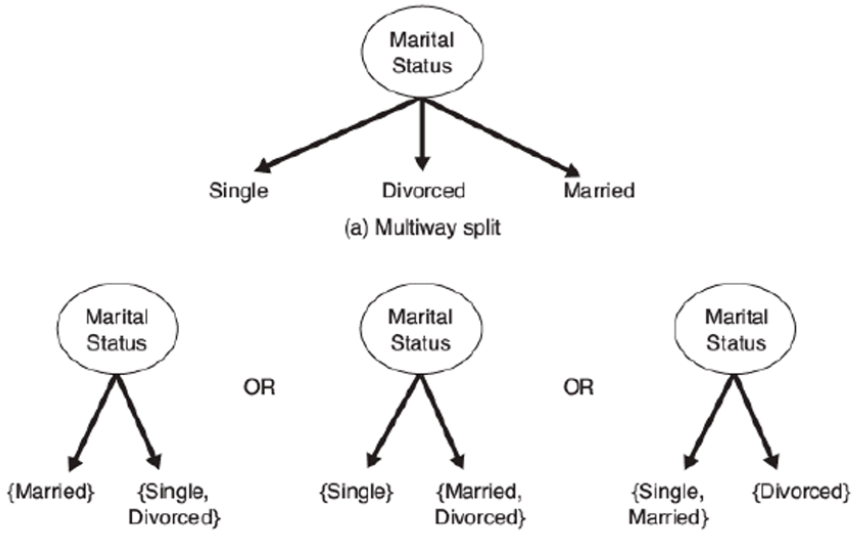
Kategorički (imenski) atributi. Za atribut sa k vrednosti (npr. bračni status $\in$ {Neoženjen, Oženjen, Razveden}) postoje dve mogućnosti:

- *Višestruka podela:* po jedna grana za svaku vrednost (slika 6.7 gore).



Slika 6.6: Test uslovi za binarne atribute.

- *Binarna podela:* grupisanje vrednosti u dva podskupa. Algoritmi poput CART-a isključivo koriste binarne podele, razmatrajući svih $2^{k-1}-1$ mogućih particija (slika 6.7 dole).

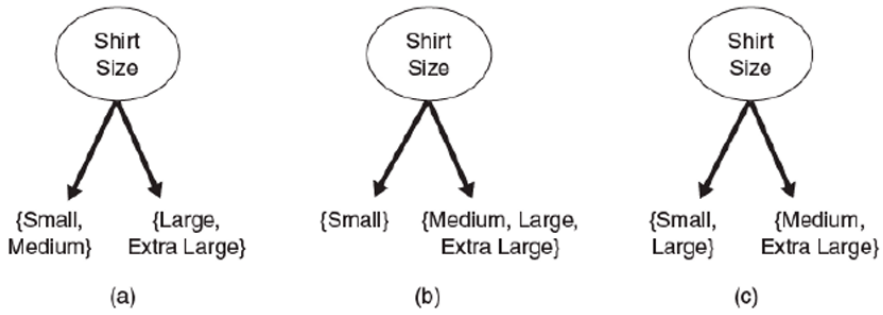


Slika 6.7: Test uslovi za imenske atribute.

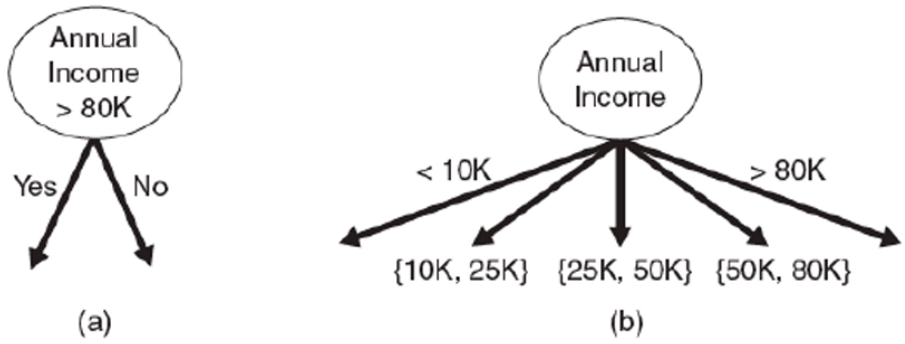
Redni atributi. Dozvoljavaju grupisanje, ali uz obavezu čuvanja prirodnog poretka. Podela {S, M} naspram {L, XL} je korektna; podela {S, L} naspram {M, XL} nije, jer narušava redosled.

Neprekidni (numerički) atributi. Za neprekidne atribute, test uslov može se izraziti kao test poređenja $(A < v)$ ili $(A \geq v)$ sa binarnim izlazima, ili razmerni upit sa izlazima oblika $v_i \leq A < v_{i+1}$, za $i = 1, \ldots, k$. Razlika ova dva pristupa prikazana je na slici 6.9.

U slučaju binarne podele, algoritam stabla odlučivanja mora da razmotri sve moguće vrednosti za v i bira onu koja proizvodi najbolju podelu. Za podelu na više grana, algoritam mora da razmotri sve moguće raspone neprekidnih vrednosti. Jedan pristup je primena diskretizacije, a nakon toga, nove vrednosti će biti dodeljene svakom diskretizovanom intervalu. Susedni intervali se takođe mogu spajati u veće intervale sve dok je redosled očuvan.



Slika 6.8: Test uslovi za redne atribute.

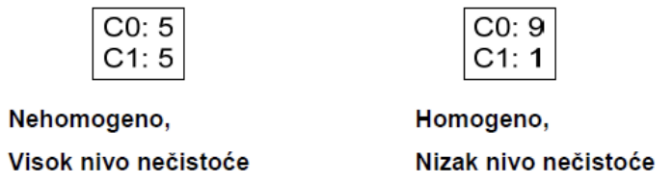


Slika 6.9: Test uslovi za neprekidne atribute.

Mere za odabir najbolje podele

Postoji mnogo mera koje se mogu koristiti za odlučivanje najboljeg načina za podelu slogova. Ove mere definišu se u smislu klasne distribucije slogova pre i posle podele. Pohlepni pristup daje prednost čvorovima sa homogenom distribucijom klasa.

Neka $p(i|t)$ označava udeo slogova koji pripadaju klasi i u datom čvoru t. Ponekad izostavljamo referencu na čvor t i deo slogova označavamo kao p_i. Mere razvijene za odabir najbolje podele često su zasnovne na stepenu nečistoće čvorova dece. Što je manji stepen nečistoće, to je distribucija klasa više nagnuta (engl. *skewed*) na jednu od dve strane. Na primer, čvor sa distribucijom klasa $(0, 1)$ ima nečistoću stepena nula, dok čvor sa ravnomernom raspodelom klasa $(0.5, 0.5)$ ima najveću nečistoću (slika 6.10).



Slika 6.10: Primer nečistoće čvora.

Primeri mera nečistoća su:

Entropija

$$\text{Entropy}(t) = -\sum_{i=0}^{c-1} p(i|t) \cdot \log_2 p(i|t) \tag{6.1}$$

Ginijev indeks

$$\text{Gini}(t) = 1 - \sum_{i=0}^{c-1} [p(i|t)]^2 \tag{6.2}$$

Greška klasifikacije

$$\text{Classification error}(t) = 1 - \max_i [p(i|t)] \tag{6.3}$$

gde je c broj klasa i u računanju entropije važi konvencija $0 \log_2 0 = 0$.

Razmotrimo primere izračunavanja opisanih mera:

Node N_1	Count
Class=0	0
Class=1	6

$$\begin{aligned} \text{Gini} &= 1 - (0/6)^2 - (6/6)^2 = 0 \\ \text{Entropy} &= -(0/6)\log_2(0/6) - (6/6)\log_2(6/6) = 0 \\ \text{Error} &= 1 - \max(0/6, 6/6) = 0 \end{aligned}$$

Node N_2	Count
Class=0	1
Class=1	5

$$\begin{aligned} \text{Gini} &= 1 - (1/6)^2 - (5/6)^2 = 0.278 \\ \text{Entropy} &= -(1/6)\log_2(1/6) - (5/6)\log_2(5/6) = 0.650 \\ \text{Error} &= 1 - \max(1/6, 5/6) = 0.167 \end{aligned}$$

Node N_3	Count
Class=0	3
Class=1	3

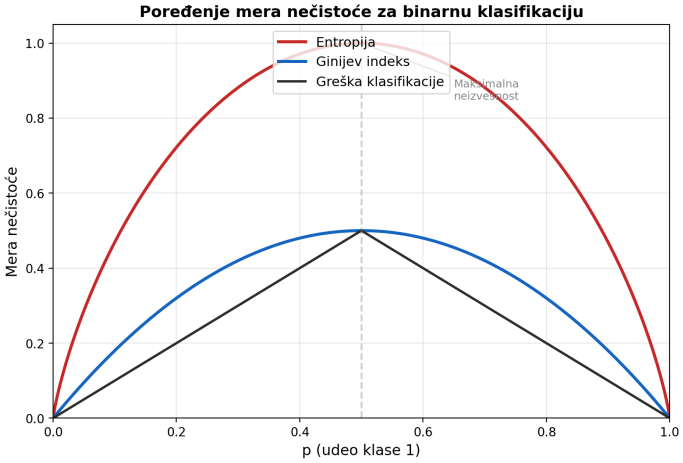
$$\begin{aligned} \text{Gini} &= 1 - (3/6)^2 - (3/6)^2 = 0.5 \\ \text{Entropy} &= -(3/6)\log_2(3/6) - (3/6)\log_2(3/6) = 1 \\ \text{Error} &= 1 - \max(3/6, 3/6) = 0.5 \end{aligned}$$

Slika 6.11 prikazuje vrednosti pomenutih mera nečistoća za problem binarne klasifikacije. Vrednost p na x-osi se odnosi na udeo slogova koji pripada jednoj od dve klase. Primetimo da sve tri mere dostižu svoj maksimum kada je raspodela klasa ravnomerna (tj. kada je $p = 0.5$). Minimalne vrednosti se dostižu kada sve vrednosti pripadaju jednoj klasi (tj. kada je p jednako 0 ili 1).

Da bismo ispitali kvalitet testa uslova, treba da poredimo stepen nečistoće roditeljskog čvora (pre razdvajanja) sa stepenima nečistoće čvorova dece (nakon razdvajanja). Što je njihova razlika veća, to je test uslov bolji. *Dobitak* (engl. *gain*), u oznaci Δ, predstavlja kriterijum koji se može koristiti za proveru kvaliteta razdvajanja:

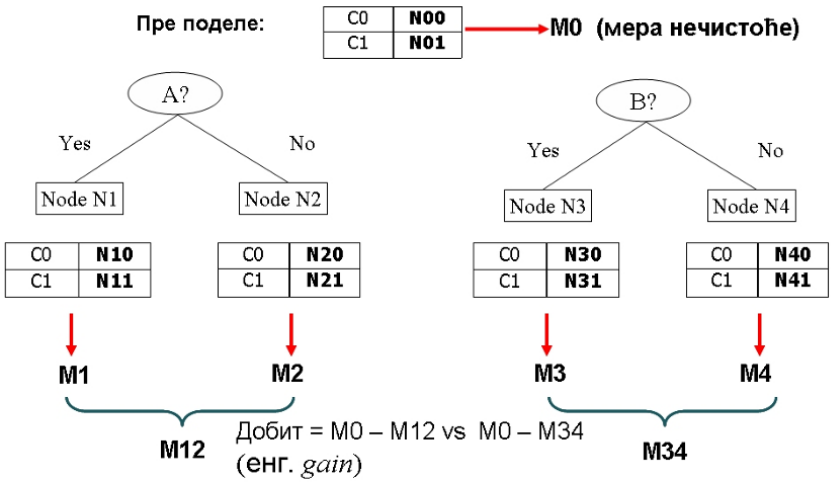
$$\Delta = I(\text{roditelj}) - \sum_{j=1}^{k} \frac{N(v_j)}{N} I(v_j), \tag{6.4}$$

gde je I funkcija koja računa meru nečistoće u zadatom čvoru (npr. Gini indeks, entropija ili greška klasifikacije), N ukupan broj slogova u roditeljskom čvoru, k broj



Slika 6.11: Poređenje mera nečistoće za problem binarne klasifikacije.

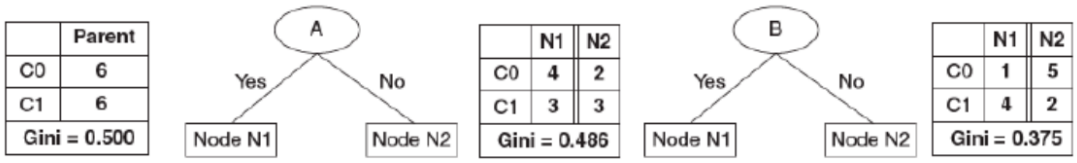
vrednosti atributa i $N(v_j)$ broj slogova u čvoru deteta v_j. Kada se za meru nečistoće I koristi entropija, takav dobitak se naziva informacioni dobitak i označava sa Δ_{info}. Možemo primetiti da umanjilac predstavlja težinski prosek mera nečistoće za novodobijene čvorove. Na slici 6.12 je data ilustracija dobitka za dve varijante razdvajanja za binarne atribute.



Slika 6.12: Dobitak za razdvajanje po binarnom atributu.

Razdvajanje binarnih atributa

Posmatrajmo dijagram na Slici 6.13. Pretpostavimo da postoje dva načina razdvajanja skupa na podskupove. Pre razdvajanja, Ginijev indeks je 0.5 jer je jednak broj slogova iz obe klase. Ako odaberemo atribut A da razdvojimo podatke, Ginijev indeks čvora $N1$ je 0.4898, a čvora $N2$ je 0.480. Težinski prosek Ginijevog indeksa za decu čvorove je $\frac{7}{12} \cdot 0.4898 + \frac{5}{12} \cdot 0.480 = 0.486$. Slično se pokazuje da je težinski prosek Ginijevog indeksa pri razdvajanju po atributu B jednak 0.375. Pošto podela po atributu B proizvodi manji Ginijev indeks, atribut B ima prednost u odnosu na atribut A.



Slika 6.13: Razdvajanje binarnih atributa.

Razdvajanje imenskih atributa

Kao što smo diskutovali, imenski atribut može proizvesti binarnu ili višestruku podelu, što je prikazano na Slici 6.14. Računanje Ginijevog indeksa za binarnu podelu slična je onoj za binarne atribute. Za prvo grupisanje (Slika 6.14(a), levo) atributa `tip automobila`, Ginijev indeks {`sportski`, `luksuzni`} je 0.4922 i Ginijev indeks {`porodični`} je 0.3750. Težinski prosek Ginijevog indeksa u ovom slučaju je

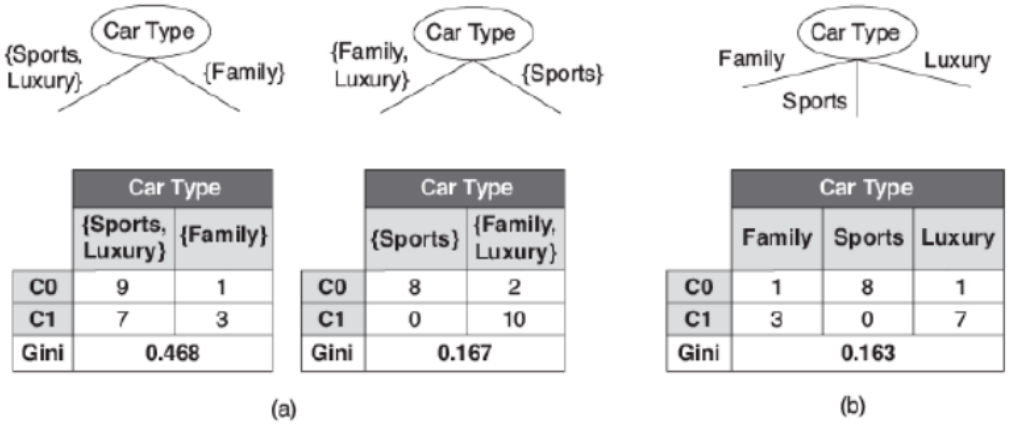
$$\frac{16}{20} \cdot 0.4922 + \frac{4}{20} \cdot 0.3750 = 0.468. \tag{6.5}$$

Slično, za drugo grupisanje (Slika 6.14(a), desno) atributa `tip automobila`, {`sportski`} i {`luksuzni`, `porodični`}, težinski prosek Ginijevog indeksa je 0.167.

Kod višestruke podele (Slika 6.14(b)), Ginijev indeks se računa za svaku vrednost atributa. Kako je Gini(`porodični`) = 0.375, Gini(`sportski`) = 0 i Gini(`luksuzni`) = 0.219, težinski prosek Ginijevog indeksa iznosi

$$\frac{4}{20} \cdot 0.375 + \frac{8}{20} \cdot 0 + \frac{8}{20} \cdot 0.219 = 0.163. \tag{6.6}$$

Odavde vidimo da višestruka podela ima manji Ginijev indeks od svih binarnih podela.



Slika 6.14: Razdvajanje imenskih atributa.

Višestruka podela je očekivano bolja od binarne jer deljenjem čvora heterogenost (nečistoća) novih čvorova ne može biti veća od roditeljskog, već samo manja. Posledica ovoga je da će algoritmi konstruisanja stabala odlučivanja koji su zasnovani na merama nečistoća, pri biranju atributa po Ginijevom indeksu, biti odabrani oni atributi koji imaju mnogo vrednosti, a sa druge strane, možda baš ti atributi nisu ključni za određivanje klase. Zbog toga ćemo nekada želeti da vršimo binarnu podelu.

Razdvajanje neprekidnih atributa

Za razdvajanje neprekidnih atributa koriste se binarne pitalice zasnovane na jednoj vrednosti ($A \leq v$, odnosno, $A > v$). Postoji više izbora za vrednost v po kojoj se deli, pri čemu je broj mogućih vrednosti za podelu jednak broju različitih vrednosti atributa po kojem se vrši podela. Granične vrednosti se mogu izabrati tako da budu jednake vrednostima atributa (korišćeno u prvom pristupu u narednom pasusu) ili da budu na sredini između dve susedne vrednosti atributa (korišćeno u drugom pristupu, dva pasusa niže).

Svaka vrednost po kojoj se deli ima pridruženu matricu brojanja. U svakoj od particija se prebrojavaju klase. Jednostavan način za izbor najbolje vrednosti za v jeste da se za svako v skeniraju podaci da bi se dobila matrica brojeva i izračunao Ginijev indeks. Ovakav pristup grube sile zahteva ponavljanje posla i složenosti je $O(n^2)$.

Da bi se smanjila kompleksnost, za svaki atribut se vrši sortiranje po vrednostima. Zatim se dobijene vrednosti linearno skeniraju uz ažuriranje matrice brojanja i izračunavanje Ginijevog indeksa. Nakon toga se bira pozicija za podelu sa najmanjim Ginijevim indeksom. Složenost ovakvog postupka je $O(n \log n)$. Primer ovakvog računanja dat je na Slici 6.15. Za vrednost v je izabrana vrednost 97, jer pri toj podeli Ginijev indeks ima najmanju vrednost (0.300).

Class	No		No		No		Yes		Yes		Yes		No		No		No		No	
	Annual Income																			
Sorted Values →	60		70		75		85		90		95		100		120		125		220	
Split Positions →	55		65		72		80		87		92		97		110		122		172		230	
	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>
Yes	0	3	0	3	0	3	0	3	1	2	2	1	3	0	3	0	3	0	3	0	3	0
No	0	7	1	6	2	5	3	4	3	4	3	4	3	4	4	3	5	2	6	1	7	0
Gini	0.420		0.400		0.375		0.343		0.417		0.400		*0.300*		0.343		0.375		0.400		0.420	

Slika 6.15: Razdvajanje neprekidnih atributa.

Karakteristike mera nečistoća

Entropija je mera koja je nešto osetljivija na heterogenost od Ginijevog indeksa. To možemo direktno zaključiti na osnovu Slike 6.11. Kako heterogenost raste, to entropija brže raste od Ginijevog indeksa.

Greška klasifikacije je najjednostavnija mera heterogenosti. Na primer, ako u nekom čvoru imamo tri klase čije su zastupljenosti $p_1 = 0.25$, $p_2 = 0.50$, i $p_3 = 0.25$, i potrebno je tom čvoru dodeliti klasu, prirodno je za taj čvor izabrati onu klasu koja ima najveću zastupljenost, tj. klasu 2. Zatim se zapitamo koliko ima instanci drugih klasa. Dakle, greška klasifikacije predstavlja procenat nesaglasnih instanci u skladu sa odlukom koju smo doneli.

Već smo konstatovali da, korišćenjem ovih mera, algoritmi konstruisanja stabala odlučivanja biraju atribute sa visokim stepenom grananja, tj. sa velikim brojem vrednosti. Zbog toga koristimo dodatne mere da bismo normirali dobitak.

Dva su pristupa za prevazilaženje ovog problema:

1. **Isključivo binarne podele** (pristup CART-a), eliminišu varijaciju u broju dece.
2. **Normalizacija dobitka**, uvodi se **odnos dobitka** (eng. *gain ratio*), kriterijum koji koristi C4.5:

$$\text{Odnos dobitka} = \frac{\Delta_{\text{info}}}{\mathit{SplitInfo}}, \qquad \mathit{SplitInfo} = -\sum_{i=1}^{k} \frac{N(v_i)}{N} \log_2 \frac{N(v_i)}{N}. \tag{6.7}$$

SplitInfo meri entropiju same podele: što je više sitnih grana, *SplitInfo* je veća i penalizuje takvu podelu.

6.2.3 Kriterijumi zaustavljanja algoritama za konstrukciju stabala odlučivanja

Na samom početku smo napomenuli da su algoritmi za konstrukciju stabala odlučivanja rekurzivni pa je neophodno razmotriti pitanje zaustavljanja ovih algoritama.

Jedan siguran kriterijum jeste da se širenje (stabla) zaustavlja kada svi slogovi pripadaju istoj klasi. Jasno je da tada nema šta da se dalje razvrstava, pa algoritam staje. Slično, ukoliko svi slogovi imaju iste vrednosti svih atributa, takođe nemamo šta dalje da razvrstavamo. Ova dva kriterijuma predstavljaju prirodna zaustavljanja. U ovom slučaju, moguće je da ostanemo sa neheterogenim podacima, a da pritom ostanemo bez uslova za grananje. To je nešto što se regularno dešava, bilo zbog grešaka, bilo zbog toga što nismo uzimali u obzir sve atribute (ili nam nisu bili dostupni).

Rano zaustavljanje predstavlja kriterijum zaustavljanja koji dovodi do prekida algoritma pre prirodnog zaustavljanja. Više reči o ovome će biti u daljem tekstu. Napomenimo da ovaj kriterijum nije jednostavno odrediti.

6.2.4 Karakteristike stabala odlučivanja

U nastavku dajemo sažetak najvažnijih karakteristika algoritama za konstrukciju stabala odlučivanja.

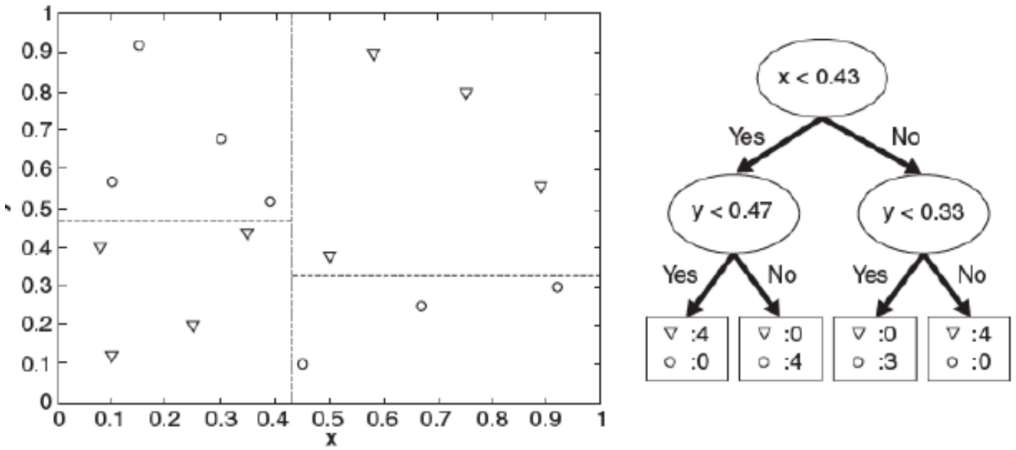
1. Tehnike koje su razvijene za izgradnju stabala odlučivanja su efikasne, pa su stabla odlučivanja jeftina za konstruisanje. Konstruisanje modela je brzo, čak i kada je veličina trening skupa veoma velika. Dalje, jednom kada je stablo konstruisano, klasifikacija novih instanci je veoma brza. U najgorem slučaju, klasifikacija novih instanci je složenosti $O(h)$, gde je h najveća dubina stabla. Ipak, napomenimo da je većina metoda klasifikacije u istraživanju podataka brza u klasifikaciji novih instanci, tako da je ključna prednost ove metode *brza konstrukcija modela*.
2. Stabla odlučivanja manjih dubina se *relativno lako interpretiraju*. Ovo je jedno od glavnih prednosti stabala odlučivanja u odnosu na druge metode klasifikacije. Stablo odlučivanja, prilikom klasifikacije nove instance, osim odgovora kojoj klasi pripada daje i spisak uslova na osnovu čega je ta instanca klasifikovana baš u toj klasi. Ovo predstavlja argumentaciju koju možemo da interpretiramo.

3. Jedna od najvećih mana stabala odlučivanja jeste što je *preciznost dobijena njihovom primenom slabija u odnosu na preciznost drugih metoda klasifikacije*, kao što su neuronske mreže, metod potpornih vektora, i drugi. Unapređenje stabla odlučivanja predstavljaju tzv. nasumične (slučajne) šume (engl. *random forests*). Ukratko, skup podataka se podeli na nezavisno izabrane podskupove (i u odnosu na vrste i u odnosu na kolone) i nad svakim od podskupova se konstruiše jedno stablo odlučivanja. Nova instanca se daje kao ulaz u svako dobijeno stablo, rezultat svih izlaza se agregira tako što se odabere klasa koja je najviše zastupljena („glasanje"), i to predstavlja konačni rezultat. Zbog nezavisnosti u biranju podskupova, stabla prave nekorelirane greške, pa se može pokazati da verovatnoća greške „glasanja" eksponencijalno opada sa brojem modela koji „glasaju". Problem je što nasumične šume gube prednosti koje su stabla imala, odnosno, konstrukcija nije jednostavna, a i interpretacija je gotovo nemoguća.

4. Pronalaženje optimalnog stabla odlučivanja je NP-kompletan problem. Zbog toga mnogi algoritmi za izgradnju stabala odlučivanja koriste pristupe zasnovane na heuristikama za usmeravanje pretrage u širokom prostoru pretrage.

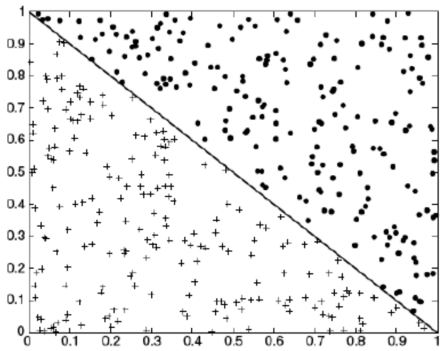
5. Jedno podstablo se može pojaviti više puta u raznim delovima stabla odlučivanja. Ovaj problem se naziva problem replikacije stabla. Ovaj problem čini stablo odlučivanja kompleksnijim nego što je neophodno i potencijalno težim za interpretaciju.

6. Test uslovi koji su opisani do sada u ovom poglavlju koriste samo jedan atribut. Posledica toga je da procedura izgradnje stabla može da se posmatra kao proces particionisanja prostora atributa u disjunktne regione sve dok svaki region ne sadrži instance iz iste klase (videti Sliku 6.16). Granica između dva susedna regiona različitih klasa se naziva granica odlučivanja (engl. *decision boundary*). Kako test uslov koristi samo jedan atribut, to su granice odlučivanja rektilinearne (engl. *rectilinear*) hiperravni, tj. hiperravni koje su paralelne koordinatnim osama. Ovo ograničava ekspresivnu moć reprezentacije stabla odlučivanja prilikom modeliranja odnosa između neprekidnih atributa.



Slika 6.16: Primer stabla odlučivanja i njegovih granica odlučivanja za dvodimenzionalni skup podataka.

Slika 6.17 ilustruje skup podataka koji ne može biti efektivno klasifikovan algoritmom drveta odlučivanja koji se bazira na test uslovima koji koriste samo jedan atribut. Bilo da li je granica odlučivanja paralelna apscisi ili ordinati, veliki broj instanci jedne ili druge klase će biti pogrešno klasifikovano. U ovom slučaju, stablo odlučivanja mora da kombinuje oba atributa pri čemu važi da, iako ne može da postigne traženu pravu, stablo može cik-cak aproksimacijom da relativno dobro postigne efekat tražene prave, koja zaista deli instance dveju klasa. Međutim, ovaj postupak je veoma zahtevan i stabla malih dubina ne mogu nikako ovo da postignu.



Slika 6.17: Primer skupa podataka koji se ne može particionisati optimalno korišćenjem test uslova koji koriste jedan atribut.

7. Prisustvo redundantnih atributa nema štetnih efekata na preciznost stabla odlučivanja. Za atribut kažemo da je redundantan ukoliko je visoko koreliran sa drugim atributom u podacima. Jedan od dva atributa neće biti korišćen za razdvajanje jednom kada je drugi atribut odabran. Ipak, ako skup podataka sadrži mnogo irelevantnih atributa, tj. atributa koji nisu korisni za proces klasifikacije, onda postoji šansa da neki od irelevantnih atributa bude izabran tokom izgradnje stabla, što dovodi do velike, a nepotrebne dubine stabla. Tehnike za odabir atributa mogu povećati performanse klasifikatora zasnovanom na stablu odlučivanja tako što će eliminisati itelevantne atribute tokom faze preprocesiranja podataka.

6.3 Praktični problemi pri klasifikaciji

U ovoj sekciji će biti reči o nekim praktičnim problemima koji važe za sve metode klasifikacije, a ne samo za stabla odlučivanja. Ipak, treba imati na umu makar jedan konkretan metod klasifikacije prilikom razmatranja ovih problema.

6.3.1 Preprilagođavanje i potprilagođavanje modela

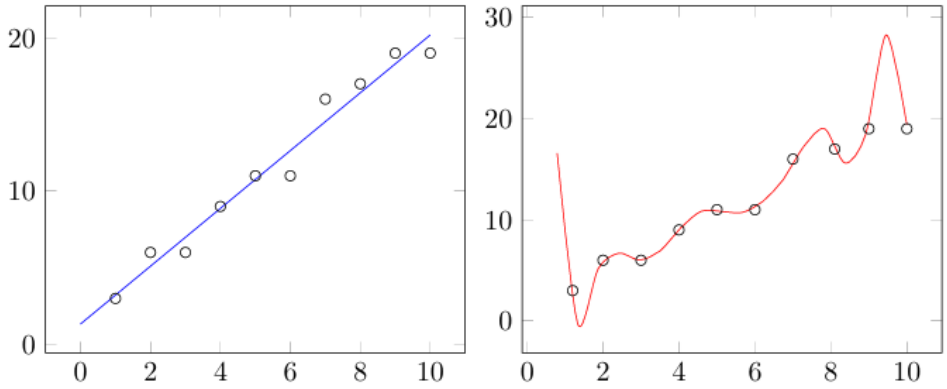
Greške u klasifikaciji se najčešće dele u dve grupe: (1) greške pri treniranju i (2) greške pri generalizaciji. Greška pri treniranju je broj grešaka u klasifikaciji za dati skup podataka za trening, a greška u generalizaciji je očekivana greška modela u odnosu na unapred nepoznate instance.

Dobar model mora korektno da klasifikuje i trening podatke i unapred nepoznate podatke (ili test podatke). Drugim rečima, dobar model mora da ima nisku grešku pri

treniranju, ali i nisku grešku pri generalizaciji. Ovo je bitno zato što model koji se previše dobro prilagodi trening podacima može imati veoma veliku grešku pri uopštavanju nego model sa većom greškom pri treniranju. Ovaj fenomen je poznat kao preprilagođavanje (engl. *overfitting*).

Preprilagođavanje modela je fenomen koji se veoma često javlja u praksi. Ukoliko ne znamo kako da rukujemo njime, gotovo je sigurno da ništa korisno nećemo postići u istraživanju podataka. Nadalje govorimo o fleksibilnosti modela i kako se ona može kontrolisati.

Posmatrajmo dva modela za skup dvodimenzionalnih tačaka na slici 6.18, nefleksibilan linearni model i fleksibilni interpolacioni polinom. Linearni model prati skup tačaka i ponegde od njega odudara, a interpolacioni polinom prolazi kroz svaku tačku skupa i savršeno im se prilagođava. To što je on uspeo da se savršeno prilagodi trening podacima je uzrokovalo vrlo loše ponašanje u predviđanju. Ukoliko bismo skup tačaka levo samo malo izmenili, koeficijenti nove prave bi toliko bili slični da bismo dobili praktično istu pravu. Sa druge strane, ako bismo malo izmenili skup tačaka desno, sva je prilika da bi interpolacioni polinom postao neprepoznatljiv. To nam dosta govori o lošoj uslovljenosti preprilagodljivog modela. Fleksibilni modeli prilično variraju u odnosu na veoma male promene u podacima.

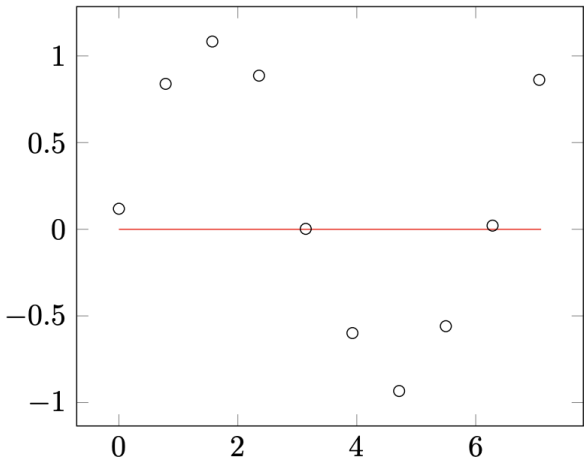


Slika 6.18: Problem potprilagođavanja modela.

Jedan način rešavanja preprilagođenosti jeste smanjivanje fleksibilnosti modela. Drugi način jeste povećanje skupa podataka. Naravno, veoma je značajno da možemo da identifikujemo da je došlo do preprilagođavanja, za šta postoje posebne metode o kojima će biti reči u nastavku.

Sa druge strane, ako je model isuviše jednostavan tada i greška pri treniranju i greška pri generalizaciji mogu biti velike. Ovaj fenomen je poznat kao potprilagođavanje (engl. *underfitting*). Na primer, Slika 6.19 predstavlja primer u kojem podatke, koji opisuju nekakvu sinusnu aktivnost pokušavamo da ukalupimo u pravu $y = 0$. Ovim smo zanemarili gotovo sve karakteristike podataka, pa će greške i na trening i na test podacima biti velike. Iako ovaj problem postoji u teoriji, on se izuzetno retko pojavljuje zbog izuzetne fleksibilnosti modela i algoritama u istraživanju podataka.

Razmotrimo sada jedan pogodan način za identifikaciju opisanih fenomena. Za početak, jasno je da stablo koje ima jedan čvor i stablo koje ima 1000 čvorova ne mogu



Slika 6.19: Problem potprilagođavanja modela.

rešiti iste probleme. Stablo sa jednim čvorom izražava jednostavne zakonitosti, dok stablo sa 1000 čvorova je vrlo fleksibilno. Dakle, sa porastom broja čvorova raste i fleksibilnost stabla. Zbog kasnijih opažanja, umesto broja čvorova možemo posmatrati dubinu stabla. Nije teško uvideti da je dubina stabla veća sa porastom broja čvorova.

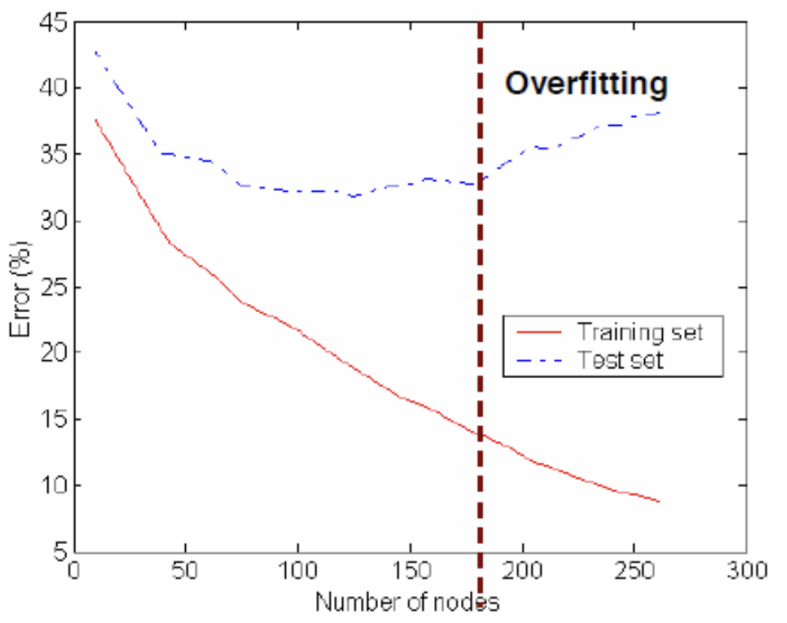
Posmatrajmo sada rezultate prikazane na slici 6.20, koja prikazuje grafik zavisnosti grešaka od broja čvorova (mi ćemo govoriti o dubini u skladu sa prethodnom napomenom) u stablu odlučivanja. Stabla niske dubine imaju visoke greške na trening skupu i test skupu. Sa druge strane, stabla veoma velike dubine imaju male greške na trening skupu, ali visoke greške na test skupu. Upravo ovaj pristup nam služi za otkrivanje da li je naš model potprilagođen ili preprilagođen. Izbegavanje preprilagođavanja se vrši tako što, posmatrajući grafik, nađemo minimum greške na test podacima, pa pogledamo za koju dubinu stabla je taj minimum dostignut. Ova primedba se može uopštiti za bilo koji drugi metod klasifikacije koji ima parametre kojima se utiče na fleksibilnost.

Preprilagođavanje usled šuma u podacima

Ukoliko je skup podataka takav da sadrži šum u podacima, klasifikator konstruisan nad trening podacima može biti prilično dobro ukalupljen nad tim podacima, ali da daje visoke greške na test podacima. Ova pojava je prirodna (u ovom slučaju) jer klasifikator pridružuje pogrešnu klasu instanci zato što se na njenom „mestu" pojavio šum u trening podacima. Na Slici 6.21 označena je tačka koja predstavlja šum, a zbog koje je napravljen loš klasifikator. Ukoliko bi se u test skupu pojavilo dosta tačaka koje su oko šuma, one bi sve bile klasifikovane pogrešno kao „plus", iako je (vizualno) očigledno da pripadaju klasi „krug". Greške usled šuma u podacima su često neizbežne i upravo one predstavljaju minimalnu grešku koja se dostiže bilo kojim klasifikatorom.

Preprilagođavanje usled nedovoljno reprezentativnih podataka

Modeli koji formiraju kriterijum klasifikacije na osnovu malog skupa za trening su podložni preprilagođenosti. Takvi modeli mogu biti konstruisani usled nedovoljno reprezentativnog uzorka u trening podacima. Dostupni su algoritmi za učenje koji nastavljaju



Slika 6.20: Greške prilikom treniranja i testiranja modela stabla odlučivanja.

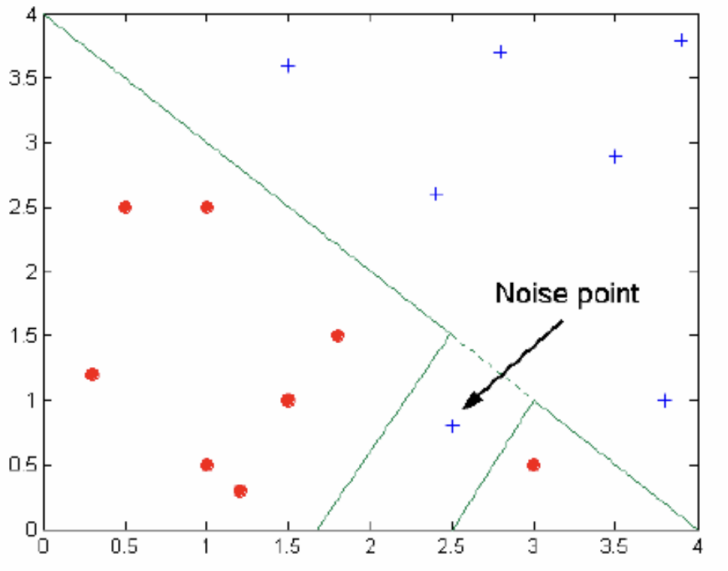
da unapređuju svoj model čak i kada je dostupan manji broj trening instanci. Na Slici 6.22 dat je primer skupa sa dve tačke koje pripadaju klasi „krug" i četiri tačke koje pripadaju klasi „×". Tačke koje su označene nepopunjenim krugom su one koje nam nisu bile dostupne prilikom klasifikacije, ali se mogu javiti u test skupu. Zbog toga je za granicu izabrana, na primer, prava $x = 1.125$. Nedostatak tačaka onemogućava korektno predviđanje oznaka klasa u donjoj polovini dijagrama. Sa dijagrama zaključujemo da je greška na test skupu 50%. U procesu klasifikacije se koriste ostali trening slogovi koji su irelevantni za klasifikaciju u tom delu.

Izbor modela

Postoji veliki broj mogućih klasifikacionih modela, sa različitim nivoima kompleksnosti, koji se mogu koristiti za otkrivanje obrazaca u skupu za učenje. Među njima, cilj je odabrati model koji pokazuje najmanju grešku generalizacije. Proces izbora modela sa odgovarajućim nivoom složenosti, za koji se očekuje da dobro generalizuje na do tada neviđene test primere, naziva se selekcija modela. Kao što je objašnjeno u prethodnom odeljku, greška na trening skupu se ne može pouzdano koristiti kao jedini kriterijum za selekciju modela. U nastavku se predstavljaju različiti pristupi za procenu sposobnosti modela za generalizaciju, sa posebnim osvrtom na indukciju stabala odlučivanja.

Korišćenje validacionog skupa

Greška generalizacije modela može se proceniti evaluacijom modela na zasebnom skupu koji se *ne koristi* za njegovo treniranje. Ovaj skup se izdvaja iz trening skupa pre treniranja modela i naziva se validacioni skup. Greška izmerena na validacionom skupu, poznata kao validaciona greška, predstavlja pouzdaniji pokazatelj generalizacionih sposobnosti modela u odnosu na grešku na trening skupu, budući da validacioni skup nije korišćen tokom procesa treniranja. Validaciona greška je stoga jedan od kriterijuma koji se mogu koristiti koristi za selekciju modela.



Slika 6.21: Preprilagođenost usled šuma.

Trening skup D_{train} može se podeliti na dva manja podskupa, D_{tr} i D_{val}, gde će se D_{tr} koristiti za treniranje a D_{val} kao validacioni skup. Ako imamo više klasifikacionih modela m treniranih na skupu D_{tr}, validaciona greška $err_{val}(m)$ računa se na skupu D_{val} i na kraju se kao preferirani bira model sa najmanjom vrednošću $err_{val}(m)$.

Validaciona greška klasifikacionog modela definiše se kao udeo instanci iz validacionog skupa koje su pogrešno klasifikovane od strane modela i može se zapisati kao

$$err_{val}(T) = \frac{\text{broj pogrešno klasifikovanih instanci u validacionom skupu}}{\text{ukupan broj instanci u validacionom skupu}}.$$

Na slici 6.23 su prikazana dva stabla odlučivanja, T_L i T_R, pri čemu svaki list ima oznaku klase (pozitivna ili negativna klasa), kao i raspodela instanci iz validacionog skupa koje u taj list dospevaju prilikom klasifikacije. Brojevi ispod svakog lista predstavljaju broj instanci pozitivne klase (+) i negativne klase (−) iz validacionog skupa.

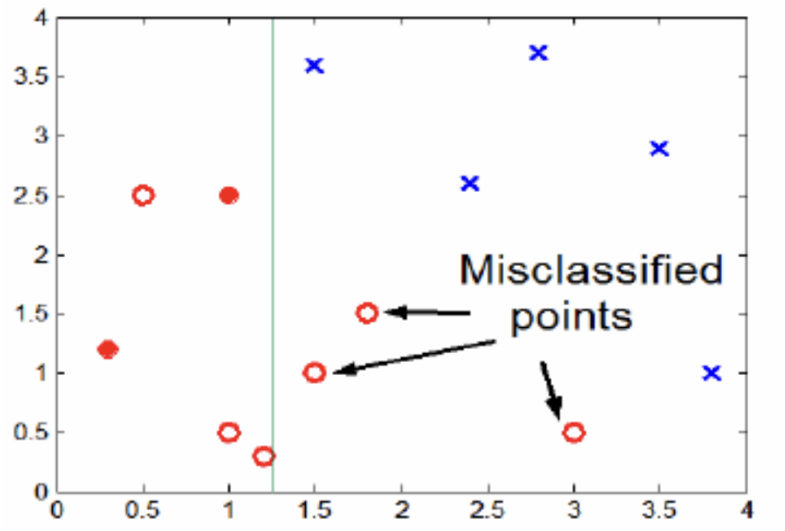
Za stablo T_L, sabiranjem pogrešno klasifikovanih instanci po svim listovima dobija se ukupno 6 grešaka na validacionom skupu koji sadrži 16 instanci, što daje validacionu grešku

$$err_{val}(T_L) = \frac{6}{16} = 0.375.$$

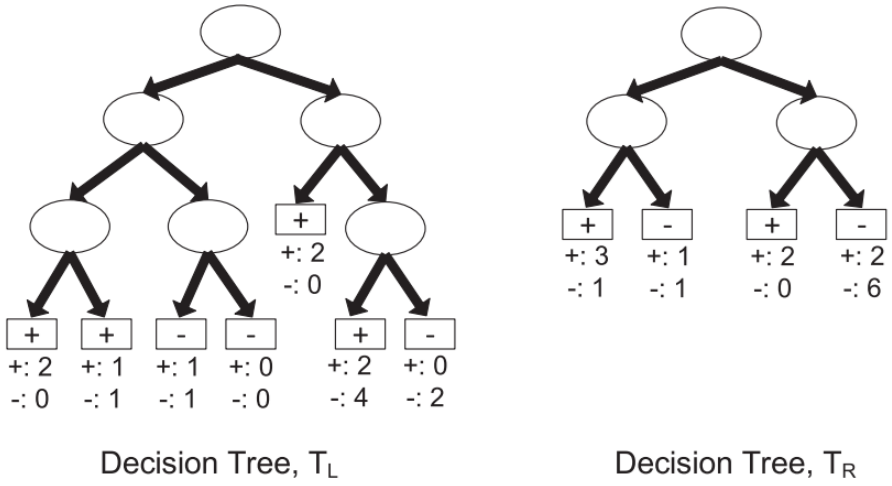
Analogno tome, za stablo T_R ukupan broj pogrešno klasifikovanih instanci iznosi 4, te je validaciona greška

$$err_{val}(T_R) = \frac{4}{16} = 0.25.$$

Pošto stablo T_R ostvaruje manju validacionu grešku u odnosu na stablo T_L, ono se preferira u procesu selekcije modela, bez obzira na činjenicu da stablo T_L ima veću složenost i veći broj listova.



Slika 6.22: Preprilagođavanje usled nedovoljno reprezentativnih podataka.



Slika 6.23: Validaciona greška

Uključivanje kompleksnosti modela

Kako kompleksnosti modela raste, povećava se i verovatnoća preprilagođavanja. Zbog toga selekcija modela ne treba da se zasniva isključivo na minimizaciji greške treniranja, već i da uzima u obzir složenost modela. Ova strategija je inspirisana poznatim principom iz filozofije nauke pod nazivom Okamova oštrica:

> Od svih objašnjenja koja su saglasna sa opažanjima, najbolje je ono objašnjenje koje je najprostije.

U ovom kontekstu, to znači da ako imamo dva modela sa istom greškom, prednost treba dati jednostavnijem modelu. U nastavku je data ilustracija ovog pristupa na modelu stabla odlučivanja.

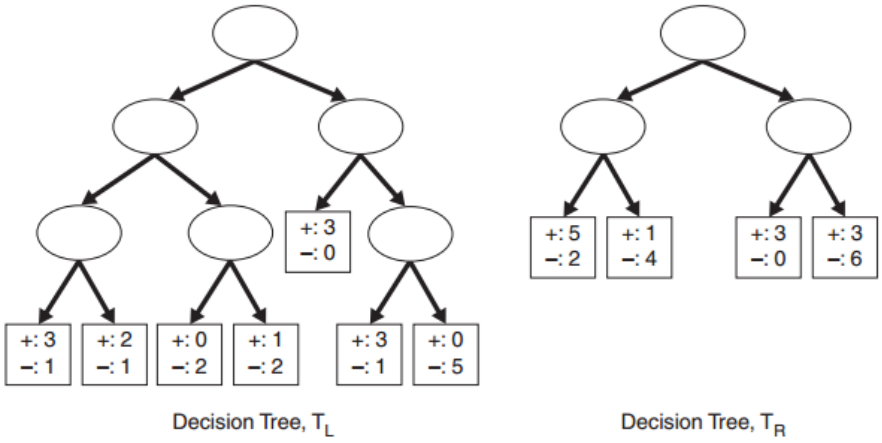
U kontekstu stabala odlučivanja, kompleksnost stabla može se izraziti odnosom broja listova k i broja instanci u skupu za treniranje N_{train}. Intuicija iza ovog pristupa je da

ima smisla da stablo bude kompleksnije ako je broj trening primera veći. Procena greške generalizacije stabla T tada može biti data kao

$$err_{gen}(T) = err(T) + \Omega \cdot \frac{k}{N_{\text{train}}}, \tag{6.8}$$

gde $err(T)$ predstavlja grešku na trening skupu a Ω predstavlja parametar koji daje manji ili veći značaj nekom od sabiraka. Ovako definisana procena greške generalizacije stabla se u literaturi naziva *pesimističnom procenom greške* jer podrazumeva da će greška generalizacije biti veća od greške na trening skupu (dodavanjem drugog sabirka). Nasuprot tome, postoji i *optimistična procena greške* generalizacije koja podrazumeva da će ona biti jednaka greški na trening skupu.

Razmotrimo ovaj pristup na sledećem primeru. Na slici 6.24 prikazana su dva stabla odlučivanja, T_L i T_R, generisana na osnovu istog trening skupa. Vidimo da stablo T_L ima veći broj listova i time veću složenost. U listovima stabala prikazana je raspodela trening instanci po klasama. Pretpostavimo da je svakom listu dodeljena većinska klasa trening instanci koje su do njega dospele.



Slika 6.24: Stabla različite kompleksnosti.

Greška treniranja stabla računa se kao udeo pogrešno klasifikovanih trening instanci u ukupnom broju instanci. Na osnovu raspodele klasa u listovima, stablo T_L pravi ukupno 4 greške na skupu od 24 trening instanci, pa je njegova trening greška jednaka

$$err(T_L) = \frac{4}{24} = 0.167.$$

Sa druge strane, stablo T_R pravi ukupno 6 grešaka, što daje trening grešku

$$err(T_R) = \frac{6}{24} = 0.25.$$

Posmatrano isključivo na osnovu greške treniranja, složenije stablo T_L bi bilo preferirano.

Međutim, s obzirom da složeniji modeli imaju veću sklonost ka preprilagođavanju, uveli smo penalizaciju složenosti kroz pesimističku procenu greške generalizacije. Posmatrajmo kako se menja ova ocena za različite vrednosti parametra Ω.

Za vrednost $\Omega = 0.5$, procena greške generalizacije za stablo T_L, koje ima $k = 7$ listova, iznosi

$$err_{gen}(T_L) = \frac{4}{24} + 0.5 \cdot \frac{7}{24} = 0.3125,$$

dok za stablo T_R, sa $k = 4$ listova, važi

$$err_{gen}(T_R) = \frac{6}{24} + 0.5 \cdot \frac{4}{24} = 0.333.$$

U ovom slučaju, stablo T_L i dalje ima manju procenjenu grešku generalizacije.

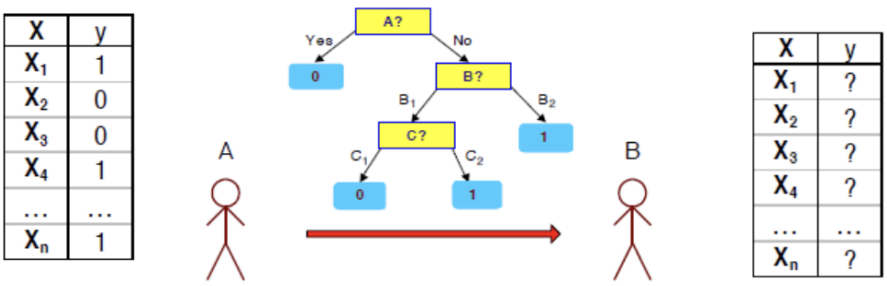
Međutim, za veću vrednost parametra $\Omega = 1$, procene greške generalizacije postaju

$$err_{gen}(T_L) = \frac{11}{24}, \qquad err_{gen}(T_R) = \frac{10}{24},$$

što dovodi do preferiranja jednostavnijeg stabla T_R. Ovaj primer jasno pokazuje da izbor parametra Ω direktno utiče na kompromis između tačnosti treniranja i složenosti modela, te samim tim i na odluku o selekciji modela.

Princip minimalne dužine opisa

Još jedan pristup selekciji modela zasniva se na informaciono-teorijskoj metodologiji poznatoj kao princip minimalne dužine opisa (*Minimum Description Length*, MDL). Da bi se ilustrovao ovaj pristup, razmotrimo primer prikazan na slici 6.25. U ovom primeru, i osoba A i osoba B poznaju vrednosti atributa za skup instanci. Međutim, osoba A takođe poznaje i klasne oznake svih instanci, dok osoba B ne zna njihove klasne oznake.



Slika 6.25: Princip MDL.

Osoba `B` može da dobije informaciju o klasama zahtevajući da osoba `A` pošalje oznake klasa sekvencijalno. Ovakva poruka bi zahtevala $O(n)$ bitova informacija, pri čemu je n ukupan broj instanci. Alternativno, `A` može da konstruiše klasifikacioni model koji opisuje odnos između x i y. Model može biti kodiran u kompaktan zapis pre slanja osobi `B`. Ako je tačnost modela 100%, onda je cena prenošenja jednaka ceni kodiranja modela. Na primer, ako je model prava, jasno je da je slanje koeficijenata prave brže od slanja svih podataka.

Ako je model 100% tačan, nije potrebno slati dodatne informacije, jer model može tačno rekonstruisati klasne oznake instanci. U ovom slučaju, trošak prenosa informacija proporcionalan je broju bitova potrebnih za kodiranje samog modela.

Međutim, u većini slučajeva model pravi određeni broj grešaka, te nije moguće tačno rekonstruisati klasne oznake samo na osnovu modela. U tom slučaju, osoba A mora dodatno poslati informacije o instancama koje su pogrešno klasifikovane modelom, kako bi osoba B mogla da ispravi dobijene klasne oznake i rekonstruše originalne podatke.

Ukupan trošak prenosa informacija, odnosno ukupna dužina opisa, može se zapisati sledećim izrazom:

$$Cost(model, data) = Cost(data \mid model) + \alpha \cdot Cost(model),$$

gde prvi član sa desne strane predstavlja broj bitova potrebnih za kodiranje pogrešno klasifikovanih instanci, dok drugi član predstavlja broj bitova potrebnih za kodiranje samog modela. Parametar α reguliše relativni uticaj složenosti modela u odnosu na cenu kodiranja grešaka u podacima.

Treba primetiti sličnost između ove jednačine i opšteg izraza za procenu greške generalizacije predstavljenog u 6.8. Prema MDL principu, dobar model treba da ima ukupnu dužinu opisa manju od broja bitova potrebnih za kodiranje celokupne sekvence klasnih oznaka bez korišćenja modela. Dalje, ukoliko se razmatraju dva konkurentna modela, prednost se daje onom modelu koji rezultuje manjom ukupnom dužinom opisa.

Izbor modela za stabla odlučivanja

Nakon što imamo pogodnu procenu greške generalizacije, algoritam za učenje može da krene u pretragu za dobrim modelom koji neće preprilagoditi trening podatke. U daljem tekstu diskutujemo o dve strategije za izbegavanje preprilagođavanja modela u kontekstu stabala odlučivanja.

Prepotkresivanje (pravilo ranijeg zaustavljanja) Algoritam se zaustavlja pre nego što stablo naraste do maksimalne veličine. Tipični uslovi zaustavljanja za određeni čvor su: „zaustavi se ako sve instance pripadaju istoj klasi" ili „zaustavi se ako su sve vrednosti atributa iste". Dodatna ograničenja mogu biti, na primer, „zaustavi se ako je broj instanci manji od neke unapred zadate granice", „zaustavi se ako je raspodela instanci nezavisna od raspodele atributa" (na primer, vidi se primenom χ^2 testa) ili „zaustavi se ako širenje tekućeg čvora ne poboljšava meru čistoće" (na primer, Ginijev indeks ili informacioni dobitak).

Potkresivanje po završetku Drvo odlučivanja raste do krajnjih granica, pa se zatim iseku čvorovi u drvetu od dna ka vrhu. Ako se greška generalizacije poboljša posle odsecanja poddrvo se zameni sa čvorom koji je list. Oznaka klase lista se određuju prema većini klasa instanci poddrveta. Za potkresivanje po završetku se može koristiti i MDL.

6.3.2 Evaluacija performansi

Evaluacija performansi klasifikatora je veoma važan aspekt klasifikacije. Do sada je bilo reči o tome da jedan model bolje ili lošije predviđa u odnosu na drugi model, ali nije bilo reči o tome kako se tačno meri „bolje", odnosno, „lošije". Cilj nam je da kvantifikujemo koliko dobro naš model predviđa.

Matrica konfuzije Matrica konfuzije (engl. *confusion matrix*) formira se od četiri veličine TP, TN, FP i LN, što je prikazano Tabelom 6.1. Vrednosti u matrici imaju sledeća značenja (pretpostavimo da imamo dve klase jedne kategorije, $C+$ i $C-$, koje posmatramo kao „pozitivno" i „negativno", redom). Stvarno pozitivni, u oznaci TP, broj je instanci koje su klasifikovane kao pozitivne i one zaista jesu pozitivne. Stvarno negativni, u oznaci TN, broj je instanci koje su klasifikovane kao negativne i one zaista jesu negativne. Lažno pozitivni, u oznaci FP, broj je instanci koje su klasifikovane kao pozitivne, ali su one zapravo negativne. Lažno negativni, u oznaci FN, broj je instanci koje su klasifikovane kao negativne, ali su one zapravo pozitivne. U slučaju dobrog klasifikatora, na glavnoj dijagonali matrice konfuzije se nalaze visoke vrednosti, a u ostalim poljima niske (idealno nule).

Tabela 6.1: Matrica konfuzije.

	Predviđena klasa		
		$C+$	$C-$
Prava klasa	$C+$	TP	FN
	$C-$	FP	TN

Tačnost Osnovna mera za procenu kvaliteta klasifikacije je tačnost (engl. *accuracy*). Tačnost se računa na sledeći način:

$$\text{Tačnost} = \frac{TP + TN}{TP + FP + TN + FN}. \tag{6.9}$$

Tačnost predstavlja udeo tačno klasifikovanih instanci u ukupnom broju instanci. Pitanje da li je dobijena tačnost niska ili visoka zavisi od problema koji rešavamo klasifikacijom. Posmatrajmo, na primer, problem prepoznavanja vrste reči. Kod onih reči kod kojih nema homonimije (a takvih je većina), vrsta reči je jednoznačno određena. Tačnost modela od 90% kod ovog problema ne bi trebalo da nas impresionira jer je dovoljno da pogledamo u rečnik i da vidimo koja je vrsta neke reči. Ukoliko bi naš model razrešavao i homonimije, onda bismo mogli da kažemo da model ima visoku tačnost.

Sa druge strane, posmatrajmo, na primer, predviđanje nekakvih klasa iz domena berzi. Tačnost od 90% je u većini takvih problema nezamisliva. Tu bismo, možda, bili zadovoljni i sa tačnošću reda 30%.

Evo još jednog primera. Na primer, prilikom identifikacije avionskog putnika kao teroriste, zapitajmo se koliki je broj takvih slučajeva koji zaista jesu teroristi. Odgovor je da je izrazito mali. Dakle, ako jedna klasa broji, na primer, 99% instanci, a druga klasa 1% instanci, onda model čija je tačnost 99% nam praktično ne služi ničemu.

I pored raznovrsnih interpretacija, tačnost je mera koja se itekako koristi u praksi, s tim da je potrebno razmotriti da li je ta mera prikladna u odnosu na konkretne podatke. Pored tačnosti, često se koristi i druge mere, od kojih neke uzimaju u obzir i raspodelu instanci po klasama.

Matrica cene Matrica cene (engl. *cost matrix*) može biti vrlo informativna ukoliko se pojam *cene* može jasno definisati u domenu koji istražujemo. Na primer, neka imamo

podatke sa informacijama o kupcima, pri čemu je 99% kupaca zadovoljno, a 1% kupaca nezadovoljno. Zbog toga ćemo podstaći da je pogrešna klasifikacija kupca koji nije zadovoljan, na primer, deset puta skuplja za nas nego pogrešna klasifikacija kupca koji je zadovoljan. Tabela 6.2 prikazuje opštiji primer matrice cene, pri čemu vrednost Cena$(i|j)$ označava cenu pogrešne klasifikacije instance klase j kao instanca klase i. U nastavku je dat primer izračunavanja korišćenjem matrice cene. Napomenimo da su brojevi iz ovog primera proizvoljni, i da stvarne cene zavise od domena konkretnog problema.

Tabela 6.2: Matrica cene.

		Predviđena klasa	
		$C+$	$C-$
Prava klasa	$C+$	Cena$(C+\|C+)$	Cena$(C-\|C+)$
	$C-$	Cena$(C+\|C-)$	Cena$(C-\|C-)$

Razmotrimo problem binarne klasifikacije u medicinskom skriningu, gde je cilj otkriti prisustvo ozbiljne bolesti kod pacijenata. Definišimo klase na sledeći način:

- pozitivna klasa (+) — pacijent ima bolest,
- negativna klasa (−) — pacijent nema bolest.

U ovakvim problemima greške nemaju simetrične posledice. Posebno je opasno pogrešno klasifikovati bolesnog pacijenta kao zdravog (lažno negativna odluka), dok je lažno pozitivna odluka obično prihvatljivija. Matrica cena data je u sledećoj tabeli:

Tabela 6.3: Matrica cene

Matrica cene	Predviđena klasa		
	Cena(a|b)	+	-
Stvarna klasa	+	-1	100
	-	1	0

Negativna vrednost cene u slučaju tačne pozitivne odluke ($C(+ \mid +) = -1$) predstavlja nagradu za pravovremeno otkrivanje bolesti, jer rana dijagnoza donosi značajnu korist i pacijentu i zdravstvenom sistemu. Nasuprot tome, lažno negativna odluka ima veoma visoku cenu ($C(- \mid +) = 100$) zbog potencijalno teških posledica propuštenog lečenja. Lažno pozitivna odluka ima relativno malu cenu ($C(+ \mid -) = 1$), jer obično rezultuje dodatnim dijagnostičkim procedurama, dok tačna negativna odluka nosi neutralnu cenu ($C(- \mid -) = 0$). Ovakva matrica cena eksplicitno favorizuje sisteme koji minimizuju rizik propuštanja obolelih pacijenata, čak i po cenu većeg broja lažno pozitivnih odluka.

Dalje, neka su data dva modela za sledećim matricama konfuzije:

Model M1	Predviđena klasa		
		+	-
Stvarna klasa	+	150	40
	-	60	250

Model M2	Predviđena klasa		
		+	-
Stvarna klasa	+	250	45
	-	5	200

Tabela 6.4: Matrice konfuzije za modele M1 i M2

“

Na osnovu konfuzionih matrica prikazanih u tabeli 6.4 moguće je izračunati kako standardnu meru tačnosti modela, tako i ukupnu cenu klasifikacije, uzimajući u obzir asimetričnu matricu cena definisanu u tabeli 6.3.

Tačnost (*accuracy*) modela računa se kao odnos broja ispravno klasifikovanih instanci i ukupnog broja instanci. Za model M1 broj tačnih klasifikacija iznosi $150+250 = 400$, dok je ukupan broj instanci 500, što daje tačnost od 0.80. Model M2 ostvaruje $250+200 = 450$ tačnih klasifikacija, što odgovara tačnosti od 0.90.

Međutim, precizniju sliku o kvalitetu modela daje ukupna cena klasifikacije. Koristeći matricu cena, ukupna cena za model M1 iznosi:

$$C_{M1} = 150 \cdot (-1) + 40 \cdot 100 + 60 \cdot 1 + 250 \cdot 0 = 3910.$$

Za model M2 ukupna cena je:

$$C_{M2} = 250 \cdot (-1) + 45 \cdot 100 + 5 \cdot 1 + 200 \cdot 0 = 4255.$$

Iako model M2 ima veću tačnost u odnosu na model M1, njegova ukupna cena je viša. To je posledica većeg broja lažno negativnih odluka (45 u odnosu na 40 kod modela M1), koje su u ovom problemu izuzetno skupe. S druge strane, model M1 pravi više lažno pozitivnih grešaka, ali one nose relativno malu cenu.

Rezultati jasno pokazuju da u problemima sa asimetričnim rizicima, kao što je medicinski skrining, veća tačnost ne znači nužno bolji model. U ovom slučaju, model M1 je poželjniji jer ima manju cenu, iako ostvaruje nižu ukupnu tačnost.

Dodatne mere evaluacije performansi U nastavku govorimo o merama koje su robustnije na disbalans klasa od tačnosti. Uvodimo dve pomoćne mere, preciznost (engl. *precision*) i odziv (engl. *recall*), sledećim formulama

$$\text{Preciznost} = p = \frac{TP}{TP + FP}, \tag{6.10}$$

$$\text{Odziv} = r = \frac{TP}{TP + FN}. \tag{6.11}$$

Što su preciznost i odziv veći, to klasifikator ima bolje performanse. Ipak, od ključne je važnosti ove mere ne posmatrati pojedinačno već zajedno. U nastavku navodimo primere koji ilustruju ovu tvrdnju.

Primer: Neadekvatnost preciznosti kao mere evaluacije performansi modela za detekciju prevara

Razmotrimo problem detekcije transakcija koje predstavljaju prevare (engl. *fraud*) u velikom skupu finansijskih podataka, gde je pozitivna klasa (+) prevara, dok je negativna klasa (−) legitimna transakcija. Tipična karakteristika ovakvih problema jeste izražena neuravnoteženost klasa, pri čemu prevarne transakcije čine veoma mali deo ukupnog broja instanci.

Pretpostavimo da je analizirani skup podataka sadržao 100 000 transakcija, od kojih su samo 100 bile stvarno prevare. Matrica konfuzije modela koji detektuje isključivo najočiglednije prevare prikazana je u tabeli 6.5.

Tabela 6.5: Matrica konfuzije za model detekcije prevara

Stvarna klasa	Predviđena klasa	
	+	-
+	5	95
-	0	99 900

Na osnovu ove matrice, preciznost (precision) modela izračunava se kao

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} = \frac{5}{5+0} = 1.$$

Dobijena vrednost sugeriše savršen kvalitet modela, budući da je svaka transakcija označena kao prevara zaista bila prevara. Međutim, iz matrice konfuzije se vidi da je model propustio 95 od ukupno 100 stvarnih prevara. Odziv modela iznosi $\text{Recall} = \frac{5}{5+95} = 0.05$, što ukazuje da model detektuje svega 5% stvarnih prevara, dok istovremeno ostvaruje visoku preciznost usled izostanka lažno pozitivnih odluka, čineći ga nepouzdanim u praktičnim sistemima detekcije prevara. U kontekstu finansijskih sistema, gde je cena propuštene prevare višestruko veća od cene lažno označene legitimne transakcije, ovakav model bi doveo do velike ekonomske štete.

Ovaj primer pokazuje da preciznost, posmatrana izolovano, nije odgovarajuća mera evaluacije u problemima detekcije prevara, te je neophodno koristiti metrike koji uzimaju u obzir cenu lažno negativnih odluka.

Odziv predstavlja udeo onih instanci koje su tačno klasifikovane kao pozitivne od svih instanci koje jesu pozitivne (bilo da li smo ih klasifikovali ispravno ili ne). Odziv je mera koja je komplementarna preciznosti, te zbog toga ni ona sama za sebe nije idealna. U vezi sa tim, razmotrimo sledeći primer.

Primer: Neadekvatnost odziva kao mere evaluacije performansi modela za filtriranje nepoželjne pošte

Razmotrimo problem binarne klasifikacije elektronske pošte, gde je pozitivna klasa (+) spam poruka, dok je negativna klasa (−) legitimna poruka. Cilj modela je da identifikuje spam poruke i preusmeri ih u poseban folder, pri čemu je od presudne važnosti da se izbegne pogrešno filtriranje legitimnih, često važnih poruka.

Pretpostavimo model koji sve dolazne poruke klasifikuje kao spam. Matrica konfuzije ovakvog modela prikazana je u tabeli 6.6.

Na osnovu matrice konfuzije, odziv (recall) za pozitivnu klasu izračunava se kao

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} = \frac{500}{500+0} = 1.$$

Dobijena vrednost odziva ukazuje da model uspešno detektuje sve spam poruke jer nijedna pozitivna instanca nije propuštena. Međutim, preciznost modela iznosi

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} = \frac{500}{500 + 9\,500} = 0.05.$$

Tabela 6.6: Matrica konfuzije za model filtriranja elektronske pošte

Stvarna klasa	Predviđena klasa	
	+	-
+	500	0
-	9 500	0

Ova vrednost pokazuje da je svega 5% poruka koje je model označio kao spam zaista spam, dok je velika većina legitimnih poruka pogrešno klasifikovana.

Iako model ostvaruje maksimalan odziv, izuzetno niska preciznost ukazuje na neprihvatljivo veliki broj lažno pozitivnih rezultata. U praktičnim sistemima za elektronsku poštu, ovakvo ponašanje dovodi do gubitka važnih poruka i ozbiljno narušava funkcionalnost sistema. Ovaj primer pokazuje da odziv, posmatran izolovano, nije odgovarajuća mera evaluacije.

Prikazali smo dve komplementarne mere za evaluaciju performansi klasifikacionih modela. Ipak, često je kao zaključak nekog merenja pogodno posmatrati jednu numeričku vrednost. Dakle, potrebno je pronaći način na koji ćemo kombinovati prethodno opisane mere u jednu vrednost, pri čemu želimo da kombinovana mera bude visoka ukoliko su i preciznost i odziv visoki. U praktičnim primenama najčešće se kao takva mera koristi tzv. F-mera i ona je data formulom

$$F\text{-mera} = F = \frac{2}{\frac{1}{p} + \frac{1}{r}} = \frac{2pr}{p + r} = \frac{2TP}{2TP + FN + FP}. \tag{6.12}$$

F-mera predstavlja harmonijsku sredinu mera p i r. U odnosu na ostale numeričke sredine (aritmetičke i geometrijske), harmonijska sredina je najoštrija. Posledica toga je da čim krene da opada preciznost ili odziv, to se veoma značajno odražava na F-meru, što je upravo to efekat koji smo želeli da postignemo uvođenjem jedinstvene mere koja ih objedinjuje.

Glava 7

Dodatni metodi klasifikacije

U prethodnom poglavlju predstavljena su stabla odlučivanja kao najjednostavnija tehnika klasifikacije kao i neki od problema koji se mogu javiti u procesu klasifikacije, poput preprilagođavanja modela. U ovom poglavlju nastavljamo da uvodimo metode klasifikacije. Problemi o kojima smo govorili se u različitim oblicima, pojavljuju i kod ostalih klasifikacionih metoda. Ovo poglavlje započinje predstavljanjem jednostavne tehnike klasifikacije zasnovane na pravilima, nakon čega će biti opisane i druge, složenije metode.

7.1 Klasifikatori zasnovani na pravilima

Klasifikator zasnovan na pravilima predstavlja tehniku kojom se instance klasifikuju pomoću skupa pravila oblika ,,ako...onda...". Pravila modela se zapisuju u disjunktivnoj normalnoj formi, $R = (r_1 \vee r_2 \vee \ldots \vee r_k)$, gde je R skup pravila, a r_i su pravila klasifikacije odnosno disjunkti. Svako pravilo se predstavlja na sledeci način:

$$r_i : (\text{uslov}_i) \longrightarrow y$$

gde je $uslov_i$ konjunkcija atributa i y oznaka klase. Leva strana pravila se naziva (pred)uslov (engl. *rule antecedent* ili *precondition*) i sastoji se od konjunkcija testova nad atributima:

$$Condition_i = (A_1 \; op \; v_1) \wedge (A_2 \; op \; v_2) \wedge (A_k \; op \; v_k)$$

gde je (A_i, v_i) par atribut–vrednost, a op je relacioni operator iz skupa $\{=, \neq, <, \leq, >, \geq\}$. Desna strana pravila naziva se posledica (engl. *rule consequent*) i ona sadrži klasu y_i koju je predvideo model.

Pretpostavimo da želimo da imamo klasifikator zasnovan na pravilima koji bi svakoj vrsti kičmenjaka (npr. čovek, kornjača, sova, ...) na osnovu nekih osobina (tip krvi, rađa žive mladunce, može da leti, živi u vodi) dodeljivao klasu (sisar, reptil, ...). Jedan skup pravila na osnovu kojih bi se mogla izvršiti takva klasifikacija dat je u tabeli 7.1.

Kažemo da pravilo P *pokriva* ili *obuhvata* (engl. *cover*) instancu x ako atribut instance x zadovoljava uslov pravila r. Kažemo još i da je pravilo r *aktivirano* (engl. *fired* ili *triggered*) kad god pokrije neku instancu. Razmotrimo sada kako pravilo P_1 pokriva instance skupa podataka iz tabele 7.2. Vidimo da pravilo P_1((Živorođeno = ne) $\wedge$ (Može da leti = da) $\rightarrow$ Ptica) *pokriva* prvu instancu jer su njegovi atributi u skladu sa preduslovom pravila. Pravilo ne pokriva drugu instancu, jer grizli medved rađa žive mladunce i ne može

P_1 : (Živorođeno = ne) ∧ (Može da leti = da) → Ptica
P_2 : (Živorođeno = ne) ∧ (Živi u vodi = da) → Riba
P_3 : (Živorođeno = da) ∧ (Tip krvi = topla) → Sisar
P_4 : (Živorođeno = ne) ∧ (Može da leti = ne) → Reptil
P_5 : (Živi u vodi = povremeno) → Amfibija

Tabela 7.1: Primer skupa klasifikacionih pravila

Naziv	**Tip krvi**	**Živorođeno**	**Može da leti**	**Živi u vodi**	**Klasa**
jastreb	topla	ne	da	ne	ptica
grizli medved	topla	da	ne	ne	sisar

Tabela 7.2: Primer atributa za dva kičmenjaka

da leti, čime se narušava preduslov pravila P_1.

Za merenje kvaliteta klasifikacionih pravila koriste se mere pokrivenost (eng. *coverage*) i tačnost (faktor pouzdanosti, eng. *accuracy* ili *confidence factor*). Da bismo ih formalno definisali, uvodimo skup podataka D i klasifikaciono pravilo $r : A \rightarrow y$. Pokrivenost pravila predstavlja udeo instanci iz skupa D koje aktiviraju pravilo r. Sa druge strane, njegova tačnost predstavlja udeo instanci koje aktiviraju pravilo r i čija je klasa jednaka y. Što su ove mere za neko pravilo veće, to je pravilo kvalitetnije. Formalne definicije ovih mera date su sledećim izrazima:

$$\text{Pokrivenost}(r) = \frac{|A|}{|D|}$$

$$\text{Tačnost}(r) = \frac{|A \cap y|}{|A|} \tag{6.3}$$

gde $|A|$ označava broj instanci koje zadovoljavaju preduslov pravila, $|A \cap y|$ broj instanci koje zadovoljavaju i preduslov i posledicu pravila, dok $|D|$ označava ukupan broj instanci u skupu podataka.

Posmatrajmo tabelu 7.3 i pravilo P_5 : (Živi u vodi = povremeno) → Amfibija. Za njega je pokrivenost = $20\% = \frac{4}{20}$ a tačnost = $50\% = \frac{2}{4}$.

7.1.1 Primer rada klasifikatora

Da bismo ilustrovali način rada klasifikatora zasnovanog na pravilima, razmotrimo prethodno dat skup pravila i skup podataka u tabeli 7.4:

- Prvi primer, lemur, ima toplu krv i rađa žive mlade. On aktivira pravilo P_3 i, shodno tome, biva klasifikovan kao sisar.
- Drugi primer, kornjača, aktivira pravila P_4 i P_5. Pošto klase koje ova pravila predviđaju nisu saglasne (reptil nasuprot amfibiji), dolazi do konflikta koji mora biti razrešen odgovarajućom strategijom izbora pravila.
- Na treći primer, psa ajkulu, nijedno pravilo se ne može primeniti. U tom slučaju potrebno je odrediti koju klasu dodeliti takvoj test instanci.

Naziv	Tip krvi	Živorođeno	Može da leti	Živi u vodi	Klasa
čovek	topla	da	ne	ne	sisar
piton	hladna	ne	ne	ne	reptil
losos	hladna	ne	ne	da	riba
kit	topla	da	ne	da	sisar
žaba	hladna	da	ne	povremeno	amfibija
komodo	hladna	ne	ne	ne	reptil
slepi miš	topla	da	da	ne	sisar
golub	topla	ne	da	ne	ptica
mačka	topla	da	ne	ne	sisar
leopard ajkula	hladna	da	ne	da	riba
kornjača	hladna	ne	ne	povremeno	reptil
pingvin	topla	ne	ne	povremeno	ptica
bodljikavo prase	topla	da	ne	ne	sisar
jegulja	hladna	ne	ne	da	riba
salamander	hladna	ne	ne	povremeno	amfibija
gušter gila	hladna	ne	ne	ne	reptil
kljunar	topla	ne	ne	ne	sisar
sova	topla	ne	da	ne	ptica
delfin	topla	da	ne	da	sisar
orao	topla	ne	da	ne	ptica

Tabela 7.3: Skup podataka koji sadrži kičmenjake i klase kojima pripadaju

Naziv	Tip krvi	Živorođeno	Može da leti	Živi u vodi	Klasa
lemur	topla	da	ne	ne	?
kornjača	hladna	ne	ne	povremeno	?
pas ajkula	hladna	da	ne	da	?

Tabela 7.4: Primer test instanci za klasifikator zasnovan na pravilima

7.1.2 Karakteristike skupa pravila

Postoje dve važne karakteristike skupa pravila generisanog od strane klasifikatora zasnovanih na pravilima.

Međusobna isključivost Kažemo da su pravila klasifikatora su međusobno isključiva ako nijedna instanca ne aktivira više od jednog pravila. Ovo svojstvo osigurava da je svaka instanca pokrivena *najviše jednim* pravilom.

Potpuna pokrivenost Kažemo da skup pravila ima potpunu pokrivenost ako sadrži kombinaciju pravila za sve moguće vrednosti atributa. Ovo svojstvo osigurava da je svaka instanca pokrivena *bar jednim* pravilom, tj. za svaku instancu klasifikator može da dâ odgovor kojoj klasi pripada.

Zajedno, ovim dvama karakteristikama se osigurava da je svaka instanca pokrivena *tačno jednim* pravilom. Idealno je da je skup pravila uzajamno isključiv i da ima potpunu pokrivenost. Jedan primer takvog skupa je prikazan na tabeli 7.5.

r_1: (Tip krvi = topla) $\wedge$ (Živi u vodi = ne) $\rightarrow$ sisar
r_2: (Tip krvi = topla) $\wedge$ (Može da leti = da) $\rightarrow$ ptica
r_3: (Tip krvi = hladna) $\wedge$ (Živorođeno = ne) $\rightarrow$ reptil

Tabela 7.5: Primer skupa pravila koja nisu međusobno isključiva

Često skupovi pravila nemaju ovakva svojstva, uključujući i skup pravila prikazan na tabeli 7.1. U nastavku govorimo o načinima za prevazilaženje ovakvih nedostataka.

Ako skup pravila nema potpunu pokrivenost, onda mu se može dodati pravilo oblika $r_d : () \longrightarrow y_d$, koje se naziva podrazumevano pravilo (engl. *default rule*) i koje ima ulogu da pokrije preostale slučajeve. Podrazumevano pravilo se aktivira kada su sva preostala pravila u skupu promašena. Klasa y_d se naziva podrazumevana klasa (engl. *default class*) i za podrazumevanu klasu se tipično bira ona klasa koja se javlja najčešće, čime se dobija najmanja greška u slučaju nepristrasnog uzorka.

Ako pravila nisu međusobno isključiva, tada jedna instanca može da aktivira više pravila i može doći do predviđanja različitih klasa i potencijalnog konflikta. Jedna mogućnost je da se svakom pravilu dodeli prioritet. Prioritet se može definisati na različite načine, na primer na osnovu tačnosti. U ovom slučaju se pravila poređaju u opadajućem redosledu po prioritetu. Tako uređeni skup pravila se još naziva i lista odlučivanja (engl. *decision list*). Kada dobijemo test instancu, klasifikujemo je pomoću najviše rangiranog pravila koje pokriva tu instancu.

Drugi način je glasanje gde se redosled unutar liste pravila ne menja. Ovaj pristup podrazumeva da se primene sva pravila za koja instanca ispunjava preduslove i da posledicu svakog od tih pravila koristimo kao glas za tu klasu. Instanci se dodeljuje klasa koja ima najviše glasova. Uz to, možemo dodati težinu glasovima prema tačnosti ili pokrivenosti pravila.

Korišćenje neuređenih skupova pravila za izgradnju klasifikatora zasnovanih na pravilima ima i prednosti i nedostatke. Neuređeni skupovi pravila manje su podložni greškama nastalim usled pogrešnog izbora pravila za klasifikaciju test instance u poređenju sa klasifikatorima zasnovanim na uređenim skupovima pravila, koji su osetljivi na kriterijum rangiranja pravila. Sa druge strane, izgradnja modela je često jeftinija jer se pravila ne moraju čuvati u uređenom redosledu. Ipak, klasifikacija test instance može biti prilično skupa jer je potrebno uporediti atribute test instance sa preduslovima svakog pravila u skupu pravila.

7.1.3 Direktne metode za izdvajanje pravila

Da bismo ilustrovali direktne metode, razmotrićemo široko korišćen algoritam za formiranje pravila pod nazivom RIPPER (Repeated Incremental Pruning to Produce Error Reduction). Složenost ovog algoritma je gotovo linearna u odnosu na broj trening instanci i posebno je pogodan za izgradnju modela na skupovima podataka sa neuravnoteženom raspodelom klasa. RIPPER takođe pokazuje dobre rezultate na skupovima podataka sa šumom, budući da koristi validacioni skup kako bi se sprečilo preprilagođavanje modela.

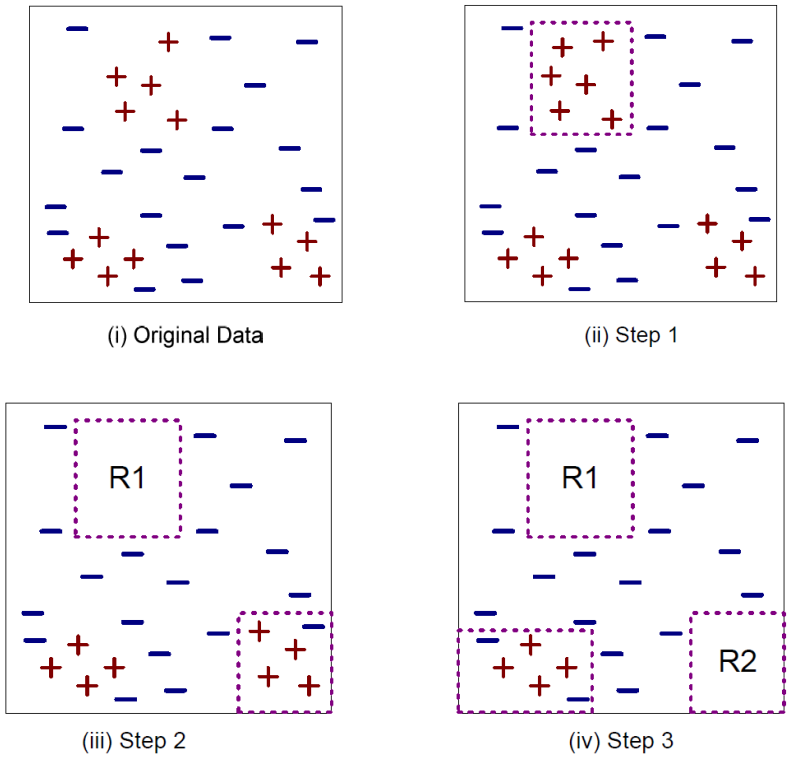
Sekvencijalno pokrivanje

RIPPER koristi algoritam sekvencijalnog pokrivanja za izdvajanje pravila direktno iz podataka. Pravila se generišu pohlepno, u svakom koraku za jednu klasu. Kod problema binarne klasifikacije, RIPPER bira većinsku klasu kao podrazumevanu klasu i uči pravila za prepoznavanje instanci koje pripadaju manjinskoj klasi. Kod višeklasnih problema, klase se najpre uređuju prema svojoj zastupljenosti u trening skupu. Neka je $(y_1, y_2, \ldots, y_k)$ uređena lista klasa, gde je y_1 *najmanje zastupljena* klasa, a y_k *najzastupljenija* klasa. Sve instance trening skupa koje pripadaju klasi y_1 se inicijalno označavaju kao pozitivni primeri, dok se instance koje pripadaju ostalim klasama označavaju kao negativni primeri. Algoritam sekvencijalnog pokrivanja zatim, na poseban način, uči skup pravila kojim se razlikuju pozitivni od negativnih primera, a iz trening skupa se izbacuju sve instance koje pripadaju y_1. Dalje, sve instance koje pripadaju klasi y_2 označavaju se kao pozitivni primeri, dok se instance iz klasa $y_3, y_4, \ldots, y_k$ označavaju kao negativni primeri. Algoritam zatim uči sledeći skup pravila za razlikovanje klase y_2 od preostalih klasa. Ovaj postupak se ponavlja sve dok ne ostane samo jedna klasa, najzastupljenija y_k, koja se proglašava za podrazumevanu klasu.

Sekvencijalno pokrivanje prikazano je u algoritmu 1. Algoritam započinje sa praznom listom odlučivanja R i izdvajanje pravila vrši za svaku klasu na osnovu redosleda određenog zastupljenošću klasa u trening skupu. Pravila za datu klasu y iterativno se izdvajaju primenom funkcije *Learn-One-Rule*. Kada se takvo pravilo pronađe, sve instance iz trening skupa koje su pokrivene tim pravilom uklanjaju se iz trening skupa. Novo pravilo se zatim dodaje na kraj liste odlučivanja R. Ovaj postupak se ponavlja sve dok nije zadovoljen kriterijum zaustavljanja, nakon čega algoritam nastavlja sa generisanjem pravila za sledeću klasu.

Algorithm 1 Sekvencijalno pokrivanje

1: Neka je E trening skup i A skup parova atribut-vrednost, $\{A_j, v_j\}$
2: Neka je Y_o lista klasa $\{y_1, y_2, \ldots, y_k\}$ uređena rastuće prema učestalosti u E
3: Neka je $R = \{\}$ inicijalni skup pravila
4: **for all** $y \in Y_o \setminus \{y_k\}$ **do**
5: **while** kriterijum zaustavljanja nije dostignut **do**
6: $r \leftarrow$ Learn-One-Rule(E, A, y)
7: Ukloni iz E trening instance koje su pokrivene pravilom r
8: $R \leftarrow R \vee r$ ▷ Dodaj r na kraj skupa pravila
9: **end while**
10: **end for**
11: $R \leftarrow R \vee (\{\} \longrightarrow y_k)$ ▷ Dodaj podrazumevano pravilo na kraj skupa pravila



Slika 7.1: Primer algoritma sekvencijalno pokrivanje

Na slici 7.1 prikazan je rad algoritma sekvencijalnog pokrivanja nad skupom podataka koji sadrži kolekciju pozitivnih i negativnih primera. Pravilo R_1, čija je pokrivenost prikazana na slici 7.1(ii), izdvaja se prvo jer pokriva najveći deo pozitivnih primera. Sve instance iz trening skupa koje su pokrivene pravilom R_1 zatim se uklanjaju, a algoritam nastavlja potragu za sledećim najboljim pravilom, koje je označeno kao R_2.

Funkcija *Learn-One-Rule*

Pronalaženje optimalnog pravila je računski zahtevan zadatak zbog eksponencijalno velikog prostora pretrage. Funkcija *Learn-One-Rule* rešava ovaj problem tako što gradi pravila na pohlepan način. Proces započinje generisanjem početnog pravila $r : \{\} \rightarrow +$, gde leva strana predstavlja prazan skup, a desna strana odgovara pozitivnoj klasi. Pravilo se zatim postupno usavršava dodavanjem konjunkata sve dok nije zadovoljen određeni kriterijum zaustavljanja.

RIPPER koristi FOIL (First Order Inductive Learner) informacioni dobitak kako bi izabrao najbolju konjunkciju koja se dodaje u uslov pravila. Ova mera uzima u obzir kako povećanje tačnosti, tako i podršku pravila-kandidata, pri čemu je podrška definisana kao broj pozitivnih primera koje pravilo pokriva. Na primer, pretpostavimo da pravilo $r : A \rightarrow +$ u početku pokriva p_0 pozitivnih i n_0 negativnih primera. Nakon dodavanja nove konjunkcije B, prošireno pravilo $r' : A \wedge B \rightarrow +$ pokriva p_1 pozitivnih i n_1 negativnih primera. FOIL informacioni dobitak proširenog pravila računa se na sledeći način:

$$\text{FOIL informacioni dobitak} = p_1 \times \left(\log_2 \frac{p_1}{p_1 + n_1} - \log_2 \frac{p_0}{p_0 + n_0} \right).$$

RIPPER bira onu konjunkciju koja ima najveći FOIL informacioni dobitak za proširenje pravila, kao što je ilustrovano u narednom primeru.

Razmotrimo nešto proširen trening skup za problem klasifikacije kičmenjaka prikazan u tabeli 7.6. Pretpostavimo da je ciljna klasa za funkciju *Learn-One-Rule* klasa sisara. U početku, uslov pravila $\{\} \rightarrow$ sisar pokriva 5 pozitivnih i 10 negativnih primera, pa je tačnost pravila jednaka 0.333.

Naziv	**Tip krvi**	**Pokrivač kože**	**Živorođeno**	**Živi u vodi**	**Može da leti**	**Ima noge**	**Hibernira**	**Klasa**
čovek	topla	dlake	da	ne	ne	da	ne	sisar
piton	hladna	krljušt	ne	ne	ne	ne	da	reptil
losos	hladna	krljušt	ne	da	ne	ne	ne	riba
kit	topla	dlake	da	da	ne	ne	ne	sisar
žaba	hladna	nema	ne	povremeno	ne	da	da	amfibija
komodo zmaj	hladna	krljušt	ne	ne	ne	da	ne	reptil
slepi miš	topla	dlake	da	ne	da	da	da	sisar
golub	topla	perje	ne	ne	da	da	ne	ptica
mačka	topla	krzno	da	ne	ne	da	ne	sisar
gupi	hladna	krljušt	da	da	ne	ne	ne	riba
aligator	hladna	krljušt	ne	povremeno	ne	da	ne	reptil
pingvin	topla	perje	ne	povremeno	ne	da	ne	ptica
bodljikavo prase	topla	bodlje	da	ne	ne	da	da	sisar
jegulja	hladna	krljušt	ne	da	ne	ne	ne	riba
salamander	hladna	nema	ne	povremeno	ne	da	da	amfibija

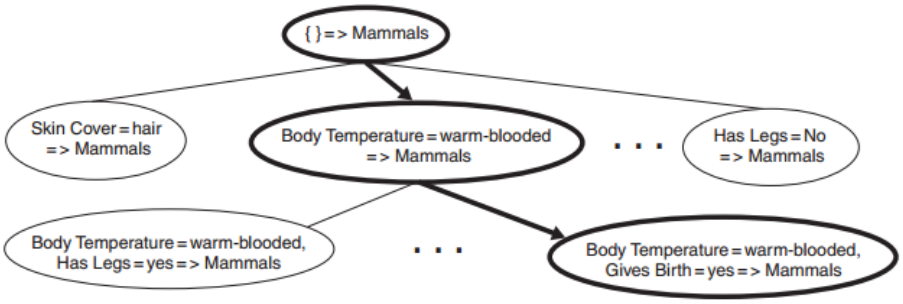
Tabela 7.6: Skup podataka o kičmenjacima za problem klasifikacije

Zatim razmotrimo sledeće tri kandidata za konjunkcije koje mogu biti dodate u uslov pravila: *Pokrivač kože* = dlake, *Tip krvi* = topla i *Ima noge* = ne. Broj pozitivnih i negativnih primera koje pravilo pokriva nakon dodavanja svake konjunkcije, zajedno sa odgovarajućom tačnošću i FOIL informacionim dobitkom, prikazani su u narednoj tabeli.

Kandidati za konjunkciju	p_1	n_1	**Tačnost**	**FOIL informacioni dobitak**
{Pokrivač kože = dlake} $\rightarrow$ sisar	3	0	1.000	4.755
{Tip krvi = topla} $\rightarrow$ sisar	5	2	0.714	5.498
{Ima noge = ne} $\rightarrow$ sisar	1	4	0.200	-0.737

Tabela 7.7: FOIL informacioni dobitak za različite kandidate za konjunkcije

Iako konjunkcija *Pokrivač kože* = dlake ima najveću tačnost među tri kandidata, konjunkcija *Tip krvi* = topla ima najveći FOIL informacioni dobitak. Zbog toga se ona bira za proširenje pravila (videti sliku 7.2). Ovaj postupak se nastavlja sve dok dodavanje novih konjunkcija više ne dovodi do povećanja vrednosti FOIL informacionog dobitka.



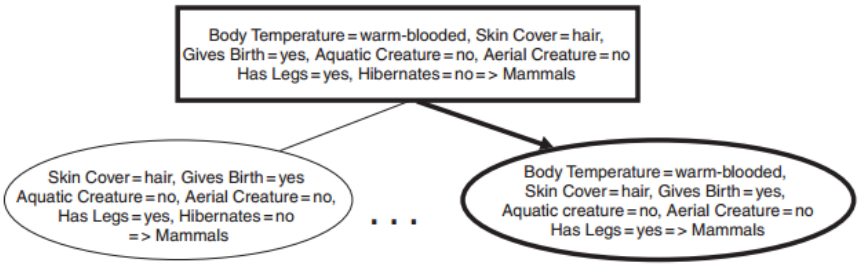
Slika 7.2: Formiranje pravila od opštijeg ka specifičnijem.

Potkresivanje pravila Algoritam RIPPER zasniva se na pohlepnoj pretrazi što znači da se u svakom koraku donosi lokalno optimalna odluka. U procesu izgradnje pravila postoji više mogućih konjunkcija koje se mogu dodati u uslov pravila, a algoritam u svakom trenutku bira onu koja prema izabranoj heurističkoj meri deluje kao najbolja. Nakon toga se postupak ponavlja pri izboru sledeće konjunkcije, čime se pravilo postupno proširuje dodavanjem novih konjunkcija. Na ovaj način dobija se pravilo koje sadrži niz konjunkcija izabranih pohlepno, bez razmatranja svih mogućih kombinacija u prostoru pretrage.

Zbog ovakve pohlepne strategije, može se dogoditi da neka konjunkcija bude odbačena u ranijoj fazi jer u datom trenutku ne deluje obećavajuće, iako bi u kombinaciji sa kasnije dodatim konjunkcijama mogla dovesti do boljeg pravila od onog koje je tada izabrano kao lokalno optimalno. Kao posledica toga, pohlepna pretraga često proizvodi pravila koja su nepotrebno dugačka ili previše specifična, odnosno sadrže konjunkcije koje negativno utiču na sposobnost generalizacije modela. Kako bi se smanjila greška generalizacije i ublažili nedostaci pohlepne konstrukcije pravila, RIPPER primenjuje postupak *potkresivanja* (eng. *pruning*) pravila generisanih funkcijom *Learn-One-Rule* (slika 7.3).

Potkresivanje se zasniva na proceni performansi pravila na validacionom skupu. Za odlučivanje da li je potkresivanje poželjno koristi se mera $(p - n)/(p + n)$, gde p i n označavaju broj pozitivnih, odnosno negativnih primera iz validacionog skupa koje pravilo pokriva. Ova mera je proporcionalna tačnosti pravila na validacionom skupu, te veća vrednost ukazuje na bolje generalizacione sposobnosti pravila. Postupak potkresivanja odvija se iterativno, pri čemu se redom razmatra uklanjanje konjunkcija iz uslova pravila. Potkresivanje započinje od poslednje dodate konjunkcije. Na primer, za pravilo oblika $ABCD \rightarrow y$, RIPPER najpre proverava da li uklanjanje konjunkcije D poboljšava vrednost navedene mere. Ako se performanse poboljšaju, konjunkcija D se trajno uklanja iz uslova pravila i pokušava se uklanjanje skupa konjunkcija CD, zatim BCD, i tako redom. U suprotnom, konjunkcija D se vraća, a algoritam nastavlja razmatranjem uklanjanja neke od preostalih konjunkcija. Važno je napomenuti da, iako originalno pravilo može pokrivati isključivo pozitivne primere iz trening skupa, potkresano pravilo može pokriti i određeni broj negativnih primera. Ovo je prihvatljivo ukoliko ukupne performanse pravila na validacionom skupu postanu bolje, jer je cilj potkresivanja da se postigne bolja generalizacija, a ne nužno savršeno prilagođavanje trening podacima.

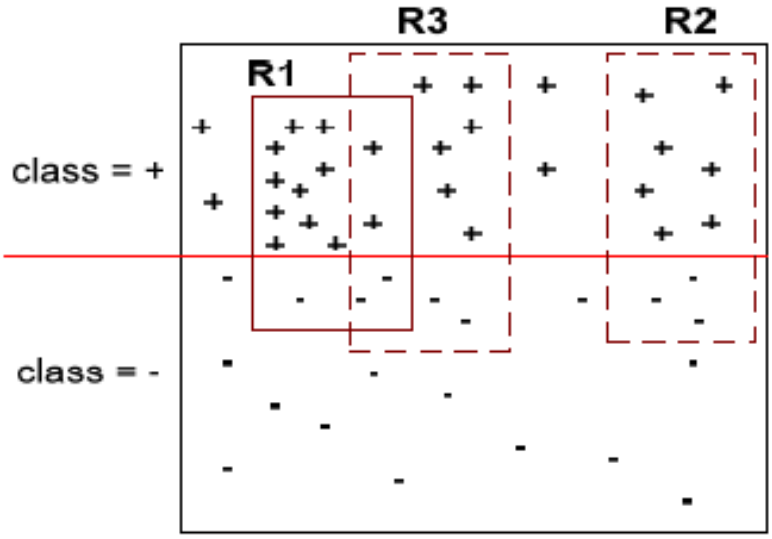
Izgradnja skupa pravila Nakon generisanja pravila, svi pozitivni i negativni primeri koje pravilo pokriva eliminišu se iz daljeg razmatranja. Pravilo se zatim dodaje u skup pra-



Slika 7.3: Formiranje pravila od specifičnijeg ka opštijem (potkresivanje).

vila dogod nije ispunjen uslov zaustavljanja koji je zasnovan na principu minimalne dužine opisa (eng. *minimal description lenght*, MDL). Naime, ako novo pravilo poveća ukupnu dužinu opisa skupa pravila za najmanje d bitova, tada RIPPER prestaje sa dodavanjem novih pravila u skup pravila (podrazumevano, d se postavlja na 64 bita). Još jedan uslov zaustavljanja koji koristi RIPPER jeste da stopa greške pravila na validacionom skupu ne sme prelaziti 50%.

Uklanjanje instanci Uklanjanje instanci se vrši, pre svega, zato što ako bi ostale sve instance, uvek bismo izdvajali isto pravilo. Mogli bismo da tražimo drugo rangirano pravilo, ali time se stvari dodatno komplikuju. Ako pogledamo Sliku 7.4, vidimo zašto je bitno da se instance uklanjaju.



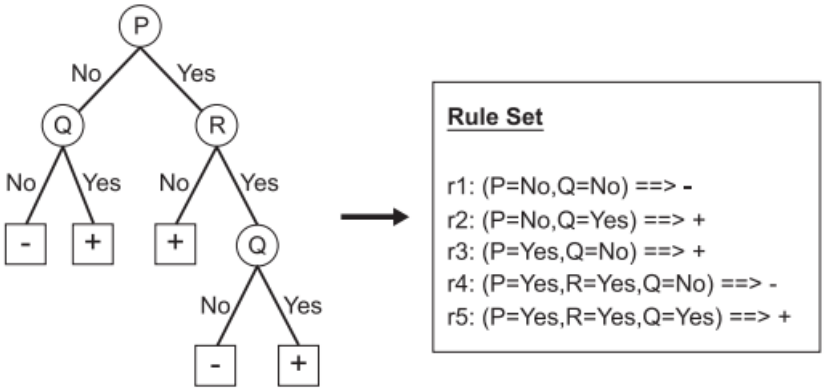
Slika 7.4: Uklanjanje instanci trening skupa sekvencijalnim pokrivanjem. $R1$, $R2$ i $R3$ predstavljaju pravila koja pokrivaju odgovarajuće podskupove instanci.

Slika prikazuje tri moguća pravila, $R1$, $R2$ i $R3$, koja su izvučena iz skupa podataka koji sadrži 29 pozitivnih instanci i 21 negativnu instancu. Pravilo $R1$ ima najveću preciznost (12/15 instanci je pozitivno, tj. 80%), zatim $R2$ (7/10, odnosno, 70%) i na kraju $R3$ (8/12, tj. 66.7%). Pravilo $R1$ je izdvojeno prvo s obzirom da ima najveću preciznost. Nakon toga, jasno je da se pozitivne instance moraju ukloniti kako bi sledeće pravilo bilo različito od $R1$. Postavlja se pitanje da li treba ukloniti samo pozitivne primere,

samo negativne primere ili oba. Da bismo odgovorili na ovo pitanje, pretpostavimo da algoritam nakon pravila R_1 mora da bira između generisanja pravila R_2 ili R_3. Iako R_2 ima veću tačnost od R_3 (70% naspram 66.7%), uočava se da je oblast pokrivena pravilom R_2 disjunktna u odnosu na R_1, dok se oblast pokrivena pravilom R_3 preklapa sa R_1. Kao posledica toga, pravila R_1 i R_3 zajedno pokrivaju 18 pozitivnih i 5 negativnih primera (što rezultuje ukupnom tačnošću od 78.3%), dok pravila R_1 i R_2 zajedno pokrivaju 19 pozitivnih i 6 negativnih primera (što rezultuje manjom ukupnom tačnošću od 76%). Ako se pozitivni primeri koje pokriva R_1 ne uklone, može doći do precenjivanja efektivne tačnosti pravila R_3. Ako se negativni primeri koje pokriva R_1 ne uklone, može doći do potcenjivanja tačnosti pravila R_3. U tom slučaju, mogli bismo neopravdano dati prednost pravilu R_2 u odnosu na R_3, iako je polovina lažno pozitivnih grešaka koje čini R_3 već obuhvaćena prethodnim pravilom R_1. Ovaj primer pokazuje da efektivna tačnost nakon dodavanja pravila R_2 ili R_3 u skup pravila postaje očigledna tek kada se uklone i pozitivni i negativni primeri koje pokriva R_1.

7.1.4 Indirektne metode za izdvajanje pravila

Listu pravila možemo dobiti iz stabla odlučivanja. U čvorovima su postavljana pitanja i odgovaranjem na ta pitanja dolazi se do listova gde dobijamo klasu. Upravo ta pitanja mogu da posluže kao pravila, odnosno, pravila čine skup pitanja i odgovora (put od korena do lista u stablu) koji vodi do konkretne klase. Test uslovi na koje nailazimo u putu kroz stablo čine preduslov, dok oznaka klase u listu čini posledicu pravila (slika 7.5). Kvalitet ovog pristupa je što se dobija skup pravila koji je međusobno isključiv i sa potpunom pokrivenošću. Mana je što možemo dobiti veliki broj pravila koji dele veliki broj uslova, a sa minimalnom razlikom među njima. Ipak, neki uslovi se mogu uprostiti, što ilustrujemo narednim primerom.



Slika 7.5: Generisanje pravila na osnovu stabla odlučivanja

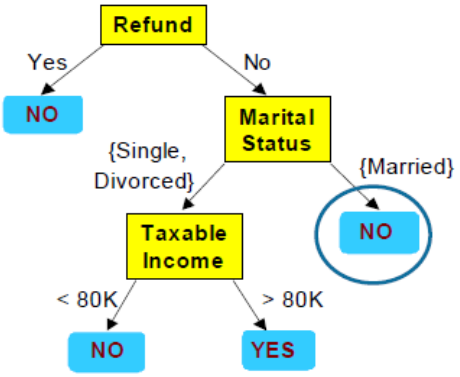
Primer. Na slici 7.6 vidimo da je, prema stablu odlučivanja, jedno od pravila koja važe za klasu NO sledeće:

$$(\text{Refund=No}) \wedge (\text{Status=Married}) \longrightarrow \text{NO}.$$

Međutim, u tabeli podataka vidimo da je svim instancama kojima je vrednost atributa Status jednaka Married pridružena klasa NO i da atribut Refund uopšte ne utiče na krajnji

ishod, pa se pravilo može uprostiti, čime se dobija pravilo

$$(\text{Status}{=}\text{Married}) \longrightarrow \text{NO}.$$



Tid	Refund	Marital Status	Taxable Income	Cheat
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

Slika 7.6: Test skup i odgovarajuće stablo odlučivanja

7.1.5 Prednosti klasifikatora zasnovanih na pravilima

Ukratko ćemo navesti neke dobre strane klasifikatora zasnovanih na pravilima, bez dubljeg ulaženja u detalje. Prvo, ovi klasifikatori imaju istu izražajnu moć kao i stabla odlučivanja. Drugo, njihova interpretacija je jednostavna, sve dok broj pravila nije veliki. Treće, njihovo formiranje je jednostavno.

7.2 Modeli zasnovani na instancama

Kod stabala odlučivanja, stablo sa test uslovima predstavlja eksplicitan model. Nasuprot tome, kod modela zasnovanih na instancama ne postoji eksplicitan model već se instance iz trening skupa koriste za davanje odgovora na pitanje kojoj klasi pripada nova instanca. Cena treninga ne postoji jer nema modela, ali cena predviđanja je velika. Kod stabala odlučivanja cena treninga je bila veća, ali proveravanje instance je trivijalno. Kod modela zasnovanih na instancama prilikom svake klasifikacije se analizira ceo trening skup i to je skupo.

Primeri modela zasnovanih na instacama su učenje napamet i metod najbližih suseda. Učenje napamet (engl. *Rote classifier*) čuva celokupan skup instanci za trening i sprovodi klasifikaciju samo ako se atributi novih instanci potpuno poklope sa atributima trening instanci. Metod najbližih suseda ili k-najbližih suseda (engl. *k-nearest neighbors*, kNN), kao što mu ime sugeriše, ne koristi ceo trening skup, već samo k tačaka najbližih instanci koja se klasifikuje (u nastavku govorimo o tome šta znači da je neka tačka najbliža drugoj tački) za obavljanje klasifikacije.

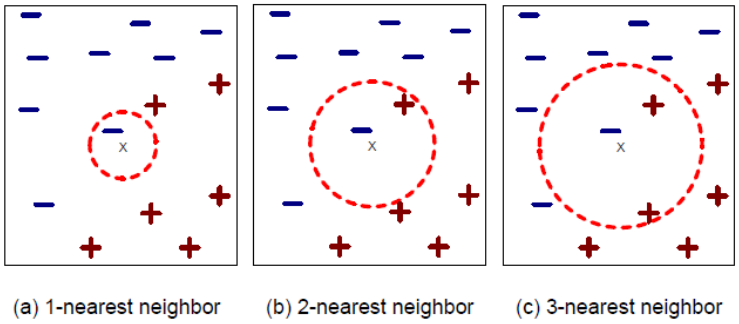
7.2.1 Klasifikacija pomoću najbližih suseda

Osnovna ideja je da na osnovu k najbližih suseda date instance odredimo klasu te instance. Vrši se određivanje suseda, a zatim se prebroji koliko suseda pripada kojoj klasi (u literaturi se ovaj postupak često naziva glasanjem (eng. *voting*)) i test primeru se dodeli najbrojnija klasa. Algoritam kNN je formalno predstavljen algoritmom 2 .

Za klasifikatore kažemo su lenji (engl. *lazy*) ukoliko se proces modeliranja podataka za trening odlaže do procesa klasifikacije podataka za test. Klasifikatori zasnovani na metodu k-najbližih suseda su primer lenjih klasifikatora.

Algorithm 2 Metod k najbližih suseda

1: Neka je k broj najbližih suseda, a D skup trening primera.
2: **for** svaki test primer $z = (x', y')$ **do**
3: Izračunaj $d(x', x)$, rastojanje između z i svakog primera $(x, y) \in D$.
4: Izdvoj $D_z \subseteq D$, skup od k trening primera koji su najbliži primeru z.
5: $y' = \arg\max_v \sum_{(x_i, y_i) \in D_z} I(v = y_i)$ (glasanje)
6: **end for**



Slika 7.7: Skup najbližih suseda instance x za $k \in \{1, 2, 3\}$.

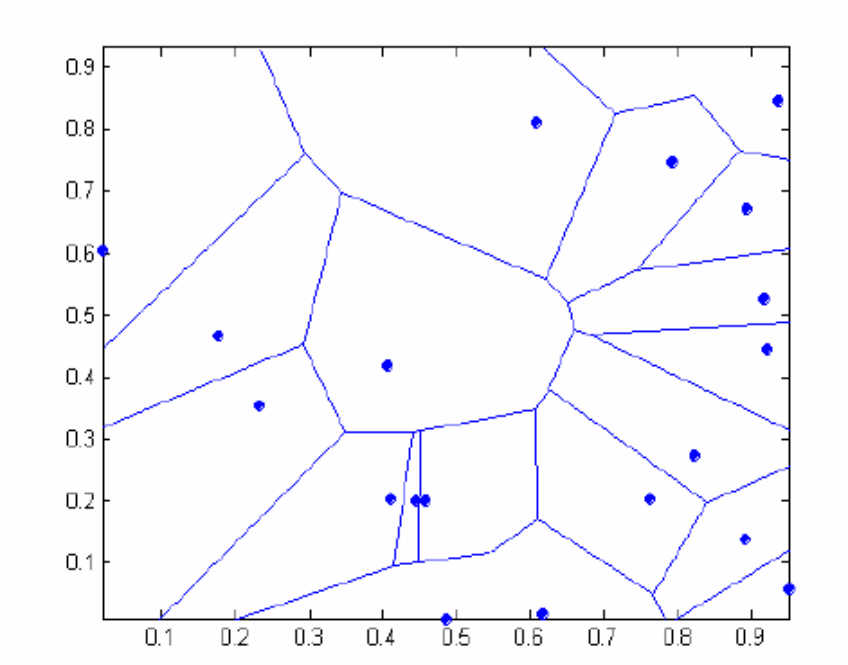
Da bismo razmotrili koliko je važno dobro izabrati k, posmatrajmo primer na slici 7.7. Želimo da klasifikujemo instancu x. Ukoliko uzmemo jednog najbližeg suseda, vidimo da joj je najbliža tačka koja pripada klasi minus, pa zaključujemo da je to klasa instance x. Međutim, ako uzmemo dva suseda, onda nismo sigurni koja je klasa jer imamo isti broj instanci obe klase (plus i minus). Na kraju, ako uzmemo tri suseda, onda nekako više liči da je klasa plus, a ne minus. Dakle, bitno je dobro odrediti k.

Biranje vrednosti za k se najlakše postiže isprobavanjem različitih vrednosti na validacionom skupu koji je disjunktan trening i test skupovima. Za različite vrednosti k, testiramo klasifikator za sve instance validacionog skupa. Na osnovu odabrane mere kvaliteta (tačnost, F-mera ili slično), izaberemo vrednost k za koju je klasifikator imao najbolje performanse.

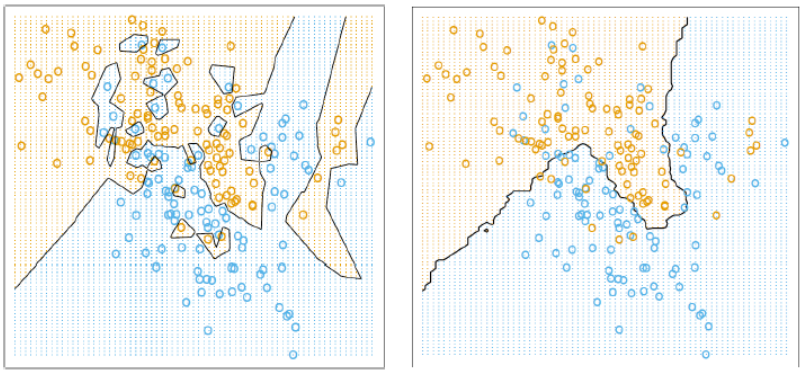
Što je k veće, model je manje fleksibilan. Razmotrimo šta fleksibilnost označava u metodu k-najbližih suseda. Vratimo se na primer sa Slike 7.7. Ako bismo uzeli $k = 14$, odgovor za bilo koju instancu bi uvek bio minus jer imamo ukupno 14 instanci u skupu,

a negativnih instanci ima više nego pozitivnih. Sa druge strane, manje vrednosti broja k povećavaju fleksibilnost modela.

Već smo rekli da se ne izgrađuju eksplicitni modeli, ali grade se implicitni modeli. U primeru na Slici 7.8, implicitan model je upravo jedan od poligona. Svaka tačka u okviru nekog poligona najbliža je plavoj tački u tom poligonu. Ovakav, implicitan model prikazan na slici se naziva Voronoj dijagram. Za dati skup tačaka Voronoj dijagram je podela prostora u regione unutar kojih su sve tačke bliže nekom pojedinačnom čvoru nego bilo kom drugom čvoru. Jasno je da je Voronoj dijagram primer metoda 1-najbližeg suseda. Na Slici 7.9 prikazan je primer tačaka u ravni koji se klasifikuju metodom k-najbližih suseda za dve vrednosti k; levo, za $k = 1$ i desno, za $k = 10$. Uočimo oblik granice za levi klasifikator i primetimo razliku u fleksibilnosti u odnosu na desni.



Slika 7.8: Voronoj dijagram ($k = 1$).



Slika 7.9: Podela na regione za $k = 1$ (levo) i $k = 10$ (desno).

Najbliže susede možemo izabrati, na primer, u odnosu na euklidsko rastojanje. Ipak, treba voditi računa o sledećem: da bismo koristili euklidsko rastojanje, moramo da standardizujemo podatke, odnosno, merne jedinice moraju biti uporedive (na primer, visina

i težina bi se morale normalizovati). Podsetimo se da se za računanje sličnosti kod dokumenata koristi kosinusna sličnost.

Već smo napomenuli da se prilikom određivanja klase nove instance za $k > 1$ koristi pravilo glasanja (korak 5 u algoritmu 2). Često će nam biti od značaja da razlikujemo glasove, na primer, prvog najbližeg i petog najbližeg suseda. Korisna modifikacija glasanja jeste da se koristi težinski faktor koji utiče na težinu ili vrednost glasa svakom suseda, a samim tim i smanjuje osetljivost algoritma na odabir broja k. Često se za težinu koristi izraz

$$w_i = \frac{1}{d(x', x_i)^2},$$

gde je x' nova instanca, x_i i-ti sused nove instance, a $d(x', x_i)$ rastojanje između nove instance i njenog i-tog suseda. Tada bi korak 5 bio modifikovan sledećom formulom:

$$y' = \arg\max_v \sum_{(x_i, y_i) \in D_z} w_i \cdot I(v = y_i)$$

Prednosti i mane

I u ovom delu ćemo se kratko zadržati na prednostima i manama. Prednosti metoda k-najbližih suseda su: jednostavnost, lokalnost (nije potrebno graditi model za ceo skup podataka u kojem možda ne važi isti trend), kao i pojava proizvoljnih oblika granica između klasa. Ovaj metod ima i svojih mana, a neke od njih su: standardizacija podataka je neophodna, neotporan je na redundantne i irelevantne atribute, kao i na nedostajuće vrednosti, a nepostojanje modela donosi i nedostatak interpretabilnosti.

7.3 Bajesov klasifikator

U mnogim aplikacijama, veza između skupa atributa i klasa nije deterministička. Drugim rečima, oznaka klase test instanci se ne može predvideti sa sigurnošću iako je skup atributa identičan nekim instancama trening skupa. Ovo se može desiti zbog šumova u podacima ili zbog prisustva nekih faktora koji utiču na klasifikaciju, a nisu uključeni u trening skup. Na primer, razmotrimo predviđanje da li je čovek u opasnosti da oboli od srčane bolesti na osnovu njegove ishrane i fizičke aktivnosti. Iako većina ljudi koji jedu zdravo i redovno vežbaju imaju manje šanse da obole od srčanih bolesti, one i dalje mogu biti izazvane nekim drugim faktorima kao što je nasledni faktor, prekomerno pušenje ili konzumiranje alkohola.

U ovoj sekciji govorimo o modeliranju probabilističkih veza između skupa atributa i klasa. Neka su X i Y par slučajno odabranih promenljivih. Njihova zajednička verovatnoća (engl. *joint probability*), u oznaci $P(X = x, Y = y)$, odnosi se na verovatnoću da će X uzeti vrednost x i da će Y uzeti vrednost y. Uslovna verovatnoća je verovatnoća da će slučajna promenljiva uzeti zadatu vrednost kada je vrednost druge promenljive poznata. Na primer, uslovna verovatnoća $P(Y = y|X = x)$ odnosi se na verovatnoću da će Y uzeti vrednost y onda kada X ima vrednost x. Zajednička i uslovna verovatnoća su povezane na sledeći način:

$$P(X, Y) = P(Y|X) \cdot P(X) = P(X|Y) \cdot P(Y).$$

Iz ove jednačine dobijamo formulu, koja je poznata kao Bajesova teorema:

$$P(Y|X) = \frac{P(X|Y) \cdot P(Y)}{P(X)}.$$

Bajesova teorema može se koristiti za rešavanje problema klasifikacije.

7.3.1 Korišćenje Bajesove teoreme za klasifikaciju

Za potrebe klasifikacije, interesuje nas izračunavanje verovatnoće klase y za instancu datu njenim skupom atributa $\mathbf{x}$. Ovo se može zapisati kao $P(y \mid \mathbf{x})$, što nazivamo **posteriornom verovatnoćom** ciljne klase. Korišćenjem Bajesove teoreme, posteriornu verovatnoću možemo predstaviti kao

$$P(y \mid \mathbf{x}) = \frac{P(\mathbf{x} \mid y)\, P(y)}{P(\mathbf{x})}$$

Primetimo da brojilac prethodne jednačine sadrži činioce $P(\mathbf{x} \mid y)$ i $P(y)$, koji oba doprinose posteriornoj verovatnoći $P(y \mid \mathbf{x})$. U nastavku opisujemo oba ova člana.

Prvi član $P(\mathbf{x} \mid y)$ nazivamo **uslovnom verovatnoćom klase** (eng. *class-conditional probability*) za vektor atributa sa datom oznakom klase. $P(\mathbf{x} \mid y)$ meri verovatnoću pojavljivanja vektora $\mathbf{x}$ u skupu instanci koje pripadaju klasi y. Ako $\mathbf{x}$ zaista pripada klasi y, tada treba očekivati da je $P(\mathbf{x} \mid y)$ visoka. Sa ove tačke gledišta, upotreba uslovnih verovatnoća klase nastoji da obuhvati distribuciju iz kog je generisan trening skup.

Drugi član u brojiocu jednačine nazivamo **priornom verovatnoćom** $P(y)$. Priorna verovatnoća obuhvata naša prethodna uverenja o raspodeli klasa, nezavisno od posmatranih vrednosti atributa. Na primer, možemo imati prethodno uverenje da je verovatnoća da neka osoba oboli od srčane bolesti jednaka α, bez obzira na njenu zdravstvenu dokumentaciju. Priorna verovatnoća može se dobiti korišćenjem ekspertskog znanja ili zaključiti na osnovu prethodnog iskustva sa raspodelom klasa.

Imenilac u jednačini je verovatnoća $P(\mathbf{x})$ koju nazivamo **verovatnoćom dokaza** (eng. *probability of evidence*). Primetimo da ovaj član ne zavisi od klase i da se stoga može tretirati kao normalizaciona konstanta pri izračunavanju posteriornih verovatnoća.

Dakle, Bajesova teorema nam omogućava da klasifikujemo instancu $\mathbf{x}$ izračunavanjem za svako y posteriorne verovatnoće $P(y \mid \mathbf{x})$ na osnovu uslovne verovatnoće po klasi $P(\mathbf{x} \mid y)$, priorne verovatnoće $P(y)$ i verovatnoće dokaza $P(\mathbf{x})$. Da vidimo kako se svaka od ovih vrednosti može izračunati:

- Verovatnoća dokaza $P(\mathbf{x})$ za dati trening skup može se izračunati kao

$$P(\mathbf{x}) = \sum_i P(\mathbf{x}, y_i) = \sum_i P(\mathbf{x} \mid y_i) P(y_i)$$

- Priorna verovatnoća $P(y)$ može se lako proceniti iz trening skupa izračunavanjem udela trening instanci koje pripadaju svakoj klasi.

- Za izračunavanje uslovnih verovatnoća po klasi $P(\mathbf{x} \mid y)$, jedan pristup je razmatranje udela trening instanci date klase za svaku moguću kombinaciju vrednosti atributa. Na primer, pretpostavimo da postoje dva atributa X_1 i X_2 koji mogu poprimiti diskretne vrednosti od c_1 do c_k. Neka n^0 označava broj trening instanci koje pripadaju klasi 0, od kojih n_{ij}^0 instanci ima $X_1 = c_i$ i $X_2 = c_j$. Uslovna verovatnoća po klasi tada se može dati kao

$$P(X_1 = c_i, X_2 = c_j \mid Y = 0) = \frac{n_{ij}^0}{n^0}.$$

  Ovaj pristup može lako postati računarski neizvodljiv kako se povećava broj atributa, usled eksponencijalnog rasta broja kombinacija vrednosti atributa. Na primer, ako svaki atribut može da uzme k diskretnih vrednosti, tada je broj kombinacija vrednosti atributa jednak k^d, gde je d broj atributa. Veliki broj kombinacija vrednosti atributa takođe može dovesti do loših procena uslovnih verovatnoća po klasi, budući da će svaka kombinacija imati manji broj trening instanci kada je veličina trening skupa mala. Stoga, za računanje uslovne verovatnoće klasa se koristi takozvana naivna Bajesova pretpostavka o kojoj govorimo u narednom poglavlju.

7.3.2 Naivna Bajesova pretpostavka

Naivni Bajesov klasifikator pretpostavlja da se uslovna verovatnoća po klasi za sve atribute $\mathbf{x}$ može faktorisati kao proizvod uslovnih verovatnoća po klasi za svaki atribut x_i, kao što je prikazano u sledećoj jednačini:

$$P(\mathbf{x} \mid y) = \prod_{i=1}^{d} P(x_i \mid y),$$

gde se svaka instanca $\mathbf{x}$ sastoji od d atributa, $\{x_1, x_2, \ldots, x_d\}$. Osnovna pretpostavka koja stoji iza prethodne jednačine jeste da su vrednosti atributa x_i **uslovno nezavisne** jedna od druge, pod uslovom da je data klasa y. To znači da atributi zavise isključivo od ciljne klase i da, ukoliko znamo koja je to klasa, možemo smatrati atribute međusobno nezavisnim. Pojam uslovne nezavisnosti može se formalno izraziti na sledeći način.

Uslovna nezavisnost

Neka X_1, X_2 i Y označavaju tri skupa slučajnih promenljivih. Kaže se da su promenljive u skupu X_1 uslovno nezavisne od promenljivih u skupu X_2, pod uslovom da je data promenljiva Y, ako važi sledeći uslov:

$$P(X_1 \mid X_2, Y) = P(X_1 \mid Y). \tag{6.16}$$

Ovo znači da, uslovljeno promenljivom Y, raspodela promenljive X_1 nije pod uticajem ishoda promenljive X_2, te je stoga uslovno nezavisna od X_2. Da bismo ilustrovali pojam uslovne nezavisnosti, razmotrimo odnos između dužine ruke neke osobe (X_1) i njene veštine čitanja (X_2). Može se primetiti da osobe sa dužim rukama imaju tendenciju bolje čitaju, pa bi se moglo smatrati da su X_1 i X_2 međusobno povezani. Međutim, ovaj odnos se može objasniti drugim faktorom, a to je starost osobe (Y). Malo dete obično ima kratke ruke i ne čita toliko dobro kao odrasle osobe. Ako je starost osobe fiksirana, tada uočeni odnos između dužine ruke i veštine čitanja nestaje. Stoga možemo zaključiti da

dužina ruke i čitalačke sposobnosti nisu direktno povezane i da su uslovno nezavisne kada je promenljiva starosti fiksirana - tada jedno o drugom ništa ne može da kaže što nije sadržano u godinama. Poenta nezavisnosti je da jedna promenljiva ne može da kaže ništa o drugoj, a poenta uslovne nezavisnosti da jedna promenljiva ne može da kaže ništa o drugoj za fiksiranu vrednost uslova.

Kako funkcioniše naivni Bajesov klasifikator

Korišćenjem naivne Bajesove pretpostavke, potrebno je da se uslovna verovatnoća svakog atributa x_i za dato Y, proceni zasebno, umesto da se izračunava uslovna verovatnoća po klasi za svaku kombinaciju vrednosti atributa. Na primer, ako n_i^0 i n_j^0 označavaju broj trening instanci koje pripadaju klasi 0 sa $X_1 = c_i$, odnosno $X_2 = c_j$, tada se uslovna verovatnoća po klasi može proceniti kao

$$P(X_1 = c_i, X_2 = c_j \mid Y = 0) = \frac{n_i^0}{n^0} \times \frac{n_j^0}{n^0}.$$

U prethodnoj jednačini potrebno je samo prebrojati broj trening instanci za svaku od k vrednosti jednog atributa X, nezavisno od vrednosti ostalih atributa. Shodno tome, broj parametara potrebnih za učenje uslovnih verovatnoća po klasi smanjuje se sa d^k na $d \cdot k$. Ovo u velikoj meri pojednostavljuje izraz za uslovnu verovatnoću po klasi, ubrzava trening i testiranje i omogućava rad sa visokodimenzionalnim skup podataka.

Naivni Bajesov klasifikator izračunava posteriornu verovatnoću za test instancu $\mathbf{x}$ korišćenjem sledeće jednačine:

$$P(y \mid \mathbf{x}) = \frac{P(y) \prod_{i=1}^{d} P(x_i \mid y)}{P(\mathbf{x})}. \tag{6.18}$$

Pošto je $P(\mathbf{x})$ fiksnirano za svaku klasu y i ima ulogu normalizacione konstante koja obezbeđuje da $P(y \mid \mathbf{x}) \in [0, 1]$, važi

$$P(y \mid \mathbf{x}) \propto P(y) \prod_{i=1}^{d} P(x_i \mid y).$$

S obzirom na to, problem klasifikacije se svodi na problem maksimizacije izraza

$$P(y) \prod_{i=1}^{d} P(x_i \mid y)$$

.

Jedna od korisnih osobina naivnog Bajesovog klasifikatora jeste to što može jednostavno da radi sa nepotpunim informacijama unutar trening skupa, kada za svaku instancu posmatramo samo podskup atributa. Na primer, ako posmatramo samo p od ukupno d atributa u datoj instanci, tada i dalje možemo izračunati $P(y) \prod_{i=1}^{p} P(x_i \mid y)$ koristeći tih p atributa i izabrati klasu sa najvećom vrednošću. Naivni Bajesov klasifikator na taj način prirodno može da se nosi sa nedostajućim vrednostima u test instancama. Zapravo, u ekstremnom slučaju kada se ne posmatra nijedan atribut, i dalje možemo koristiti priornu verovatnoću $P(y)$ kao procenu posteriorne verovatnoće. Kako posmatramo sve

veći broj atributa, možemo dodatno kalibrisati posteriornu verovatnoću kako bi ona bolje odražavala verovatnoću pojavljivanja instance iz trening skupa.

U naredne dve podsekcije opisujemo nekoliko pristupa za procenu uslovnih verovatnoća $P(x_i \mid y)$ za kategoričke i kontinualne atribute na osnovu trening skupa.

Procena uslovnih verovatnoća za kategoričke atribute

Za kategorički atribut X_i, uslovna verovatnoća $P(X_i = c \mid y)$ procenjuje se na osnovu udela trening instanci u klasi y za koje atribut X_i ima određenu kategoričku vrednost c:

$$P(X_i = c \mid y) = \frac{n_c}{n},$$

gde je n ukupan broj trening instanci koje pripadaju klasi y, dok je n_c broj instanci za koje važi $X_i = c$. Na primer, u trening skupu prikazanom u tabeli 7.8, sedam osoba ima oznaku klase *Neizmireni dužnik = Ne*, pri čemu tri osobe imaju *Vlasnik nekretnine = Da*, dok preostale četiri osobe imaju *Vlasnik nekretnine = Ne*. Kao rezultat toga, uslovna verovatnoća P(Vlasnik nekretnine = Da | Neizmireni dužnik = Ne) iznosi 3/7. Slično tome, uslovna verovatnoća za neizmirene dužnike sa bračnim statusom *Sam* data je kao P(Bračni status = Sam | Neizmireni dužnik = Da) = 2/3. Imajmo u vidu da je zbir uslovnih verovatnoća po svim mogućim vrednostima atributa X_i jednak jedinici, tj. $\sum_c P(X_i = c \mid y) = 1$.

Tid	**Vlasnik nekretnine** *binarni*	**Bračni status** *kategorički*	**Godišnji prihod** *kontinualni*	**Neizmireni dužnik** *klasa*
1	Da	Sam	125K	Ne
2	Ne	Oženjen	100K	Ne
3	Ne	Sam	70K	Ne
4	Da	Oženjen	120K	Ne
5	Ne	Razveden	95K	Da
6	Ne	Oženjen	60K	Ne
7	Da	Razveden	220K	Ne
8	Ne	Sam	85K	Da
9	Ne	Oženjen	75K	Ne
10	Ne	Sam	90K	Da

Tabela 7.8: Trening skup za primer neizmirenog zajma

Procena uslovnih verovatnoća za kontinualne atribute

Postoje dva načina za procenu uslovnih verovatnoća po klasi za kontinualne atribute:

1. Možemo diskretizovati svaki kontinualni atribut, a zatim zameniti kontinualne vrednosti odgovarajućim diskretnim intervalima. Ovaj pristup transformiše kontinualne atribute u redne atribute, nakon čega se može primeniti jednostavna metoda opisana ranije za izračunavanje uslovnih verovatnoća kategoričkih atributa. Treba imati u vidu da greška procene ove metode zavisi od strategije diskretizacije, kao i od broja diskretnih intervala. Ako je broj intervala prevelik, svaki interval može sadržati

nedovoljan broj trening instanci da bi se obezbedila pouzdana procena verovatnoće $P(X_i \mid Y)$. S druge strane, ako je broj intervala premali, proces diskretizacije može dovesti do gubitka informacija o stvarnoj raspodeli kontinualnih vrednosti, a samim tim i do loših predviđanja.

2. Možemo pretpostaviti određeni oblik raspodele verovatnoće za kontinualnu promenljivu i proceniti parametre te raspodele korišćenjem trening podataka. Na primer, možemo koristiti Gausovu raspodelu kako bismo predstavili uslovnu verovatnoću kontinualnih atributa. Gausova raspodela karakterisana je sa dva parametra: srednjom vrednošću μ i varijansom σ^2. Imajući u vidu definiciju gustine verovatnoće normalne raspodele, Za svaku klasu y_j, uslovna verovatnoća po klasi za atribut X_i data je kao

$$P(X_i = x_i \mid Y = y_j) = \frac{1}{\sqrt{2\pi\sigma_{ij}^2}} \exp\left[-\frac{(x_i - \mu_{ij})^2}{2\sigma_{ij}^2}\right]. \tag{6.19}$$

Parametar μ_{ij} može se proceniti kao aritmetička sredina vrednosti atributa X_i za sve trening instance koje pripadaju klasi y_j. Slično tome, σ_{ij}^2 može se proceniti na osnovu varijanse takvih trening instanci. Na primer, razmotrimo atribut godišnjeg prihoda prikazan u tabeli 7.8. Srednja vrednost i varijansa ovog atributa u odnosu na klasu *Ne* iznose

$$\bar{x} = \frac{125 + 100 + 70 + \cdots + 75}{7} = 110,$$

$$s^2 = \frac{(125 - 110)^2 + (100 - 110)^2 + \cdots + (75 - 110)^2}{6} = 2975,$$

$$s = \sqrt{2975} = 54.54.$$

Za test instancu sa oporezivim prihodom jednakim 120K, možemo koristiti sledeću vrednost kao uslovnu verovatnoću datu za klasu *Ne*:

$$P(\text{Income} = 120 \mid \text{No}) = \frac{1}{\sqrt{2\pi}(54.54)} \exp\left(-\frac{(120 - 110)^2}{2 \times 2975}\right) = 0.0072.$$

Primer

Razmotrimo skup podataka prikazan u tabeli 7.8, gde je ciljna klasa *Neizmireni dužnik*, koja može imati dve vrednosti: *Da* i *Ne*. Možemo izračunati uslovne verovatnoće po klasi za svaki kategorički atribut, kao i srednju vrednost i varijansu za kontinualni atribut, što je sumarno prikazano u tabeli 7.9.

Želeli bismo da odredimo klasu sledećeg test primera:

$$\mathbf{x} = (\text{Vlasnik nekretnine} = \text{Ne},\ \text{Bračni status} = \text{Oženjen},\ \text{Godišnji prihod} = \$120\text{K}).$$

Da bismo to učinili, najpre izračunavamo priorne verovatnoće prebrojavanjem broja trening instanci koje pripadaju svakoj klasi. Na taj način dobijamo

$$P(\text{Da}) = 0.3 \quad \text{i} \quad P(\text{Ne}) = 0.7.$$

Atribut / Vrednost	**Klasa Ne**	**Klasa Da**
Vlasnik nekretnine = Da	3/7	0
Vlasnik nekretnine = Ne	4/7	1
Bračni status = Sam	2/7	2/3
Bračni status = Razveden	1/7	1/3
Bračni status = Oženjen	4/7	0
Srednja vrednost (μ)	110	90
Varijansa (σ^2)	2975	25

Tabela 7.9: Uslovne verovatnoće za kategoričke atribute i parametri Gausove raspodele za godišnji prihod

Zatim možemo izračunati uslovne verovatnoće po klasi na sledeći način:

$$\begin{aligned} P(\mathbf{x} \mid \text{Ne}) &= P(\text{Vlasnik nekretnine} = \text{Ne} \mid \text{Ne}) \\ &\quad \times P(\text{Bračni status} = \text{Oženjen} \mid \text{Ne}) \\ &\quad \times P(\text{Godišnji prihod} = \$120\text{K} \mid \text{Ne}) \\ &= \frac{4}{7} \times \frac{4}{7} \times 0.0072 = 0.0024. \end{aligned}$$

$$\begin{aligned} P(\mathbf{x} \mid \text{Da}) &= P(\text{Vlasnik nekretnine} = \text{Ne} \mid \text{Da}) \\ &\quad \times P(\text{Bračni status} = \text{Oženjen} \mid \text{Da}) \\ &\quad \times P(\text{Godišnji prihod} = \$120\text{K} \mid \text{Da}) \\ &= 1 \times 0 \times 1.2 \times 10^{-9} = 0. \end{aligned}$$

Primetimo da je uslovna verovatnoća po klasi *Da* jednaka nuli, jer u trening skupu ne postoji nijedna instanca koja pripada klasi *Da* i ima bračni status *Oženjen*. Na osnovu ovih uslovnih verovatnoća, možemo proceniti posteriorne verovatnoće kao

$$P(\text{Ne} \mid \mathbf{x}) = \frac{0.7 \times 0.0024}{P(\mathbf{x})} = 0.0016\,\alpha,$$

$$P(\text{Da} \mid \mathbf{x}) = \frac{0.3 \times 0}{P(\mathbf{x})} = 0,$$

gde je $\alpha = 1/P(\mathbf{x})$ normalizaciona konstanta. Pošto važi

$$P(\text{Ne} \mid \mathbf{x}) > P(\text{Da} \mid \mathbf{x}),$$

test instanca se klasifikuje u klasu *Ne*. ■

Obrada nultih uslovnih verovatnoća

Prethodni primer ilustruje potencijalni problem koji se javlja prilikom korišćenja naivne Bajesove pretpostavke pri proceni uslovnih verovatnoća po klasi. Ukoliko je uslovna verovatnoća za neki od atributa jednaka nuli, tada čitav izraz za uslovnu verovatnoću po klasi postaje jednak nuli. Imajmo u vidu da nulte uslovne verovatnoće možemo posmatrati u slučajevima kada je broj trening instanci mali, a broj mogućih vrednosti atributa veliki. Tada se može dogoditi da neka kombinacija vrednosti atributa i klase nije u datom trening skupu, što rezultira nultom uslovnom verovatnoćom.

U ekstremnijem slučaju, ako trening instance ne pokrivaju određene kombinacije vrednosti atributa i klasa, tada možda nećemo biti u mogućnosti da klasifikujemo neke test instance jer su sve uslovne verovatnoće po klasi jednake nula. Na primer, ako je

$$P(\text{Bračni status} = \text{Razveden} \mid \text{Ne}) = 0$$

umesto 1/7, tada instanca data vektorom atributa

$$\mathbf{x} = (\text{Vlasnik nekretnine} = \text{Da},\ \text{Bračni status} = \text{Razveden},\ \text{Godišnji prihod} = \$120\text{K})$$

ima sledeće uslovne verovatnoće po klasi:

$$P(\mathbf{x} \mid \text{Ne}) = \frac{3}{7} \times 0 \times 0.0072 = 0,$$

$$P(\mathbf{x} \mid \text{Da}) = 0 \times \frac{1}{3} \times 1.2 \times 10^{-9} = 0.$$

Pošto su obe uslovne verovatnoće po klasi jednake nuli, naivni Bajesov klasifikator nije u stanju da klasifikuje ovu instancu. Da bi se ovaj problem prevazišao, neophodno je prilagoditi procene uslovnih verovatnoća tako da one ne budu previše osetljive na prosto korišćenje razlomaka trening instanci. Ovo se može postići primenom sledećih alternativnih procena uslovne verovatnoće:

$$\text{Laplasova procena:} \quad P(X_i = c \mid y) = \frac{n_c + 1}{n + v},$$

$$m\text{-procena:} \quad P(X_i = c \mid y) = \frac{n_c + mp}{n + m},$$

gde je n broj trening instanci koje pripadaju klasi y, n_c broj trening instanci za koje važi $X_i = c$ i $Y = y$, v ukupan broj mogućih vrednosti atributa X_i, p početna procena $P(X_i = c \mid y)$ koja je poznata unapred, a m hiperparametar koji određuje stepen poverenja u procenu p u slučajevima kada je udeo trening instanci suviše mali. Obično ako je veliki trening skup, onda biramo malo m, a ako je mali trening skup, onda uzimamo veliko m jer je mali skup nepouzdan.

Intuicija iza Laplasove procene je da, bez obzira što se nisu pojavile sve vrednosti atributa X_i uz klasu y, znamo da su one moguće. Stoga dodajemo $+1$ u brojilac (to je jedna vrednost (c) atributa X_i) i $+$v u imenilac (to su sve moguće vrednosti (njih m) atributa X_i).

Ako je $n_c = 0$, i Laplasova procena i m-procena daju nenulte vrednosti uslovnih verovatnoća. Samim tim, ove metode izbegavaju problem nestajanja uslovnih verovatnoća po klasi.

Prednosti i mane

Prednosti naivnog Bajesovog klasifikatora su razne. Prvo, efikasni su trening (prebrojavanje) i predviđanje (imamo sve verovatnoće samo izračunamo verovatnoću po formuli). Ovaj klasifikator nije osetljiv na prisustvo irelevantnih atributa (imaju iste raspodele u svim klasama). Ne zavisi od vrste atributa (kategorički ili neprekidni). Ipak, lakše je računati za kategoričke jer njih jednostavno brojimo. Neprekidne prvo diskretizujemo, pa

prebrojavamo ili pronađemo normalnu raspodelu ili neku drugu raspodelu koja im odgovara. Takođe, lako se ažurira kako pristižu podaci (ne mora da se broji ispočetka).

Naravno, i ovaj klasifikator ima i svoje mane. Naime, naivni Bajesov klasifikator pretpostavlja uslovnu nezavisnost atributa. Takođe, ako se neke vrednosti atributa ne pojavljuju u trening skupu, klasifikator im dodeljuje verovatnoću 0, što nije realistično. Poslednji problem se ipak može rešiti.

Glava 8

Procena kvaliteta i izbor modela

U prethodnim poglavljima smo videli da uz pomoć različitih algoritama dobijamo različite modele, kao što su stablo odlučivanja ili Bajesov model. Procena kvaliteta i izbor modela imaju ogroman značaj u istraživanju podataka. Na prvi pogled mogu delovati jednostavno, ali ispostavlja se da ovo nije trivijalan posao.

Procena kvaliteta modela se bavi ocenom greške predviđanja modela. Izbor modela se bavi izborom jednog od više mogućih modela.

Izbor modela se zasniva na proceni kvaliteta modela. Preciznije, biraćemo modele koji su najkvalitetniji u zavisnosti od procene kvaliteta koju smo izvršili. Ipak, veza ne mora biti trivijalna zbog raznih praktičnih problema. Upravo zbog netrivijalnosti veze između procene kvaliteta i izbora dolazi do prethodno pomenutih grešaka.

Procena kvaliteta i izbor modela se zasnivaju na merama kvaliteta (na primer, preciznost) i tehnikama evaluacije i izbora (na primer, unakrsna validacija). Vrlo često se pravi greška pri izjednačavanju tehnika evaluacije i tehnika izbora. Mi ćemo praviti razliku između njih u daljem tekstu. S obzirom da smo diskutovali o merama kvaliteta u Podsekciji **??**, na ovom mestu ćemo se samo podsetiti da mere kvaliteta zavise od problema. Na primer, za problem klasifikacije možemo koristiti preciznost ili F-meru, dok za regresiju imamo srednjekvadratnu grešku i koeficijent determinacije (R^2). Pozabavimo se detaljnije tehnikama evaluacije i izbora.

8.1 Tehnike evaluacije

Tehnike evaluacije variraju po složenosti. Varijabilnost zavisi od: (1) konfigurabilnosti algoritma i (2) željenog kvaliteta ocene. Konfigurabilnost algoritma se odnosi na menjanje parametara od kojih zavisi odabir konkretnog algoritma. Na primer, broj k u algoritmu k-najbližih suseda, maksimalna dubina ili odabir mere nečistoće kod stabala odlučivanja, i sl. Što se tiče željenog kvaliteta ocene, naravno da bismo uvek želeli najbolji kvalitet ocene. Međutim, vrlo česta je situacija da jednostavniji modeli ne daju dovoljnu pouzdanost ocene. Ponavljanjem eksperimenta nad malo izmenjenim podacima dobijaju se greške koje jako variraju, tj. nisu stabilne.

Primarno načelo procene kvaliteta Glavno načelo procene kvaliteta modela glasi: ,,Podaci korišćeni za procenu kvaliteta modela ni na koji način ne smeju biti upotrebljeni prilikom treninga". Deluje jednostavno, ali se u praksi ispostavlja kao vrlo pipavo. Pre

svega zbog toga što postoje načini na koje je moguće prekršiti ovo načelo, a da uopšte nismo svesni da smo to uradili.

8.1.1 Nekonfigurabilan slučaj

Pretpostavlja se da algoritam nije konfigurabilan ili da je konfiguracija fiksirana. Jedan primer bi bilo stablo odlučivanja sa unapred određenom topologijom. Već nam je jasno da nekonfigurabilan algoritam nije realističan scenario. Ako se naučeni model pokaže loše, u iskušenju smo da vršimo neke izmene i u tom slučaju naredni metodi procene kvaliteta nisu validni, i to predstavlja osnovnu grešku, odnosno, korišćenje tehnika za nekonfigurabilne algoritme kod konfigurabilnih algoritama je zabranjeno.

Izbor modela Pošto nema različitih konfiguracija, nema ni većeg broja modela iz kojeg se može birati, pa je izbor praktično trivijalan. Jednostavno, dajemo celokupan skup podataka kao ulaz u algoritam i dobijamo model M. On ima svoju grešku E koju ćemo probati u fazi evaluacije da aproksimiramo. Ipak, postavlja se pitanje na kojim podacima treba trenirati?

Procena kvaliteta Podaci se dele na dva skupa: trening skup i test skup (videti Tabelu 8.1). Trening skup, pošto je blizak ukupnim podacima, koristi se za treniranje modela M_0 koji služi kao aproksimacija modela M. Test skup se koristi za procenu greške E_0 koju model M_0 pravi prilikom predvidanja. Smatra se da je ta greška dobra aproksimacija greške E modela M.

Tabela 8.1: Podela skupa podataka na trening skup (plavo) i test skup (crveno).

x_1	x_2	x_3	y
1	9	0	8
0	6	2	1
1	3	1	5
4	9	7	6
1	1	6	7
7	2	3	4
2	9	9	9
3	3	4	6
7	2	1	7
6	5	1	5

Problemi vezani za procenu pomoću trening i test skupa Sve je u redu ukoliko je skup podataka vrlo veliki i reprezentativan, ali u suprotnom, postavljaju se pitanja poput ,,Kako izvršiti podelu?", ,,Šta ako su raspodele trening i test skupa različite?" i ,,Šta raditi usled velike varijanse ocene greške".

Procena kvaliteta K-strukom unakrsnom validacijom

Procena kvaliteta modela se može izvršiti tehnikom unakrsne validacije (engl. *cross validation*). Algoritam 3 prikazuje ovu tehniku.

Algorithm 3 K-struka unakrsna validacija.

```
1: procedure UnakrsnaValidacija(k)
2:     Podaci se dele na k slojeva                ▷ Slojevi su disjunktni podskupovi
3:     for all sloj ∈ slojevi do
4:         Trenira se model na prestalih k − 1 slojeva
5:         Vrše se predviđanja dobijenim modelom na izabranom sloju
6:     end for
7:     Računa se ocena greške
8: end procedure
```

Tabela 8.2 predstavlja primer podele skupa podataka u prvom koraku Algoritma 3. Nadalje, postupak se može opisati sledećim: odaberemo prvi crveni sloj, izvučemo ga iz skupa podataka, treniramo korišćenjem preostalih slojevima (žuti, plavi, zeleni i braon), izvršimo predviđanje korišćenjem crvenog sloja, vratimo nazad crveni sloj. Ovaj postupak (za crveni sloj) se ponavlja za slojeve žuti, plavi, zeleni i braon. Možemo primetiti da smo ovim postupkom, za razliku od podele na trening i test skupove, za treniranje i testiranje koristili sve podatke, ali uz veoma bitnu napomenu da smo svaki put razdvajali trening skup i test skup, tako da nismo prekršili primarno načelo procene kvaliteta.

Tabela 8.2: Primer podele skupa podataka unakrsnom validacijom za $k = 5$.

x_1	x_2	x_3	y
1	9	0	8
0	6	2	1
1	3	1	5
4	9	7	6
1	1	6	7
7	2	3	4
2	9	9	9
3	3	4	6
7	2	1	7
6	5	1	5

Problemi vezani za procenu unakrsnom validacijom Pre svega, posmatrajući Algoritam 3 trebalo bi odmah da dođemo do zaključka da je taj algoritam računski veoma zahtevan. Drugo pitanje koje se postavlja je biranje broja slojeva. U praksi se najčešće koriste vrednosti $k = 5$ i $k = 10$. Preporuka je da se ne koriste jednočlani slojevi (engl. *leave one out*), pošto je takva ocena greške optimistična. Dodatno, ne računati ocene greške za svaki sloj, pa ih uprosečavati (ne radi za nelinearne mere poput R^2). Umesto toga, treba dati sva predviđanja, pa onda računati grešku. Konačno, kao što smo primetili, sve instance se koriste u proceni kvaliteta, pa tehnika jeste pouzdanija, ali i dalje jedan sloj ne mora imati istu raspodelu kao preostali slojevi. Poslednji problem će biti detaljnije razmatran u narednom tekstu.

Stratifikacija

Sekundarno načelo procene kvaliteta Još jedno načelo procene kvaliteta modela glasi: „Trening i test skup treba da imaju istu raspodelu kao i buduća opažanja”. Iz

ovoga je jasno da trening i test skup među sobom treba da imaju istu raspodelu. Deluje pipavo i jeste pipavo. Ublaženje ovog problema može se postići korišćenjem velike količine podataka, korišćenjem naprednijih tehnika uzorkovanja i stratifikacijom.

Stratifikacija podrazumeva da se prilikom deljenja podataka obezbeđuje da delovi imaju istu raspodelu kao i ceo skup podataka.

Stratifikacija celokupnog skupa podataka je teška za male skupove podataka. Zbog toga se može koristiti njena pojednostavljena varijanta, odnosno, cilj je očuvati raspodelu ciljne promenljive. Algoritam 4 pokazuje kako se vrši stratifikacija u odnosu na ciljnu promenljivu.

Algorithm 4 Stratifikacija u odnosu na ciljnu promenljivu.

```
1: procedure STRATIFIKACIJA(k)
2:     Sortirati podatke u odnosu na ciljnu promenljivu
3:     for all i ← 1, k do
4:         for all j ← 1, ... do
5:             P_i ← P_i ∪ (i + j · k)          ▷ P_i je i-ti deo u stratifikaciji
6:         end for
7:     end for
8:     return P
9: end procedure
```

Tabela 8.3 prikazuje primer rezultata Algoritma 4 za $k = 5$. Primećujemo da su vrednosti atributa y (koja predstavlja ciljnu promenljivu) sortirane. Samim tim, slojevi se izgrađuju tako što redom uzimamo svaku k-tu instancu (u ovom slučaju, svaku petu) i umećemo je u odgovarajući sloj. Time dobijamo da su instance u svakom sloju približno isto raspoređene.

Tabela 8.3: Primer rezultata stratifikacije u odnosu na ciljnu promenljivu za $k = 5$.

x_1	x_2	x_3	y
0	6	2	1
7	2	3	4
1	3	1	5
6	5	1	5
4	9	7	6
3	3	4	6
1	1	6	7
7	2	1	7
1	9	0	8
2	9	9	9

Dodatno treba razmotriti slučaj kada je ciljna promenljiva kategorička (tj. kada je klasa). Rešenje je jednostavno: različitim klasama dodeliti proizvoljne numeričke vrednosti, sortirati po dodeljenim vrednostima, pa zatim ponovo uraditi kao u slučaju numeričke ciljne promenljive.

Napomenimo da stratifikacija ne rešava problem reprezentativnosti budućih predviđanja, već da raspodele slojeva budu približno iste.

8.1.2 Konfigurabilan slučaj

Algoritmi mašinskog učenja se obično navode vrednostima metaparametara. Navedimo nekoliko primera metaparametara. Linearna i logistička regresija imaju regularizacioni parametar. Metod potpornih vektora ima cenu grešaka, parametre kernela, itd. Neuronske mreže imaju regularizacioni parametar, parametar vezan za metodu inercije, itd. Pre učenja, moguće je izabrati podskup atributa. Neuronske mreže mogu imati različite arhitekture. Algoritam k najbližih suseda može koristiti različite distance. Dakle, postoje različiti načini za konfigurisanje različitih algoritama, stoga uzimamo u obzir algoritamske konfiguracije (vrednosti metaparametara, atributi, arhitekture, kerneli, itd.).

Skup svih algoritamskih konfiguracija je određen svim mogućim vrednostima svakog od parametara nekog algoritma (odnosno, svakog od delova algoritma koji može da varira). Algoritamska konfiguracija je jedna konkretna torka iz tog skupa.

Izbor modela Različite konfiguracije kada se primene da iste podatke daju različite modele. Primetimo sledeće: izborom konfiguracije se bira model. Dakle, na izbor modela utiču podaci i algoritam koji je vođen konfiguracijom.

Pitanje je ,,Kako izabrati adekvatnu konfiguraciju, a time i model?". Odgovor je – jednostavno! Izvršiti evaluaciju modela dobijenih za različite konfiguracije i izabrati najbolji. Postavlja se pitanje izbora ocena greške predvidanja tog modela. Ovaj problem nije jednostavan.

Procena kvaliteta i izbor modela na pogrešan način Prikažimo prvo pogrešan pristup koji predstavlja jedan od osnovnih načina na koje ljudi greše kada se bave ovim problemom. Pristup glasi: evaluira se svaka konfiguracija unakrsnom validacijom. Zatim, bira se najbolja konfiguracija i prijavljuje se upravo dobijena procena kvaliteta modela. Konačno, trenira se finalni model pomoću najbolje konfiguracije na celom skupu podataka. Pokažimo ovaj pristup na jednostavnom primeru. Neka imamo algoritam čija konfiguracija se sastoji samo od jednog parametra λ. Za svaku od izabranih vrednosti λ_i parametra λ dobijamo greške E_i evaluiranjem na test skupu:

$$\begin{array}{lccccc} \lambda: & 10^{k_1} & \cdots & 10^{k_i} & \cdots & 10^{k_n} \\ E: & E_1 & \cdots & E_i & \cdots & E_n \end{array}$$

Neka je $E_i = \min E_k, k = \overline{1,n}$. Dakle, treba da izaberemo parametar $\lambda_i = 10^{k_i}$ na osnovu čega dobijemo finalni model, treniran na osnovu celog skupa podataka, koji ima grešku E_i.

U čemu je greška? Uobičajeno opravdanje datog postupka bi bilo: pošto se koristi unakrsna validacija, nikad se ne vrši trening na instancama koji se koriste za testiranje. Međutim, da li je zaista tako? Vidimo da u izboru nije greška, unakrsna validacija radi dobro, onako kako smo je opisali. To znači da bi greška mogla da se potkrade u evaluaciji. Ako malo bolje razmotrimo opisani pristup, dolazimo do sledećeg: Prilikom izbora najbolje konfiguracije, oslonili smo se na informaciju dobijenu korišćenjem celog skupa podataka, a izbor najbolje konfiguracije je deo treninga, pošto se direktno odražava na rezultujući model! Iz prethodnog primera, biranje parametra λ_i je zapravo deo treninga, a mi smo taj odabir vršili na osnovu grešaka E_i, koje smo dobili iz test skupa. Eto preklapanja trening i test podataka.

Pojam validacionog skupa

Kao rešenje prethodno opisanog problema, javlja se potreba za novim skupom koji će biti disjunktan i trening i test skupu. Takav skup se naziva validacioni skup (engl. *validation set*). Pogledajmo kako se menja izbor modela i ocena greške uvođenjem validacionog skupa.

Izbor modela pomoću validacionog skupa Podaci se dele na trening i validacioni skup. Za svaku konfiguraciju se radi sledeće: (1) trenira se model za tu konfiguraciju na trening skupu i (2) vrši se ocena greške predvidanja modela na validacionom skupu. Zatim, bira se konfiguracija koja daje najmanju grešku na validacionom skupu. Konačno, trenira se finalni model pomoću najbolje konfiguracije na celom skupu podataka.

Tabela 8.4: Podela skupa podataka na trening skup (plavo), validacioni skup (zeleno) i test skup (crveno).

x_1	x_2	x_3	y
1	9	0	8
0	6	2	1
1	3	1	5
4	9	7	6
1	1	6	7
7	2	3	4
2	9	9	9
3	3	4	6
7	2	1	7
6	5	1	5

Ocena greške pomoću validacionog i test skupa Podaci se dele na trening i test skup (u okviru trening skupa se pravi podela na skup za treniranje i validacioni skup; videti Tabelu 8.4). Na celom trening skupu se izvrši izbor modela i pridružene konfiguracije pomoću validacionog skupa. Za tu konfiguraciju se trenira model na celom trening skupu. Vrši se ocena greške tog modela na test skupu i ona se prijavljuje kao ocena kvaliteta modela.

Izbor modela pomoću unakrsne validacije Za svaku konfiguraciju se vrši ocena greške predvidanja modela unakrsnom validacijom. Zatim, bira se konfiguracija koja daje najmanju grešku pri unakrsnoj validaciji. Konačno, trenira se finalni model pomoću najbolje konfiguracije na celom skupu podataka.

Procena kvaliteta ugnežđenom K-strukom unakrsnom validacijom

Uopštenje Algoritma 3 u slučaju nekonfigurabilnog algoritma dato je Algoritmom 5 za konfigurabilne algoritme.

Algorithm 5 Ugnežđena K-struka unakrsna validacija.

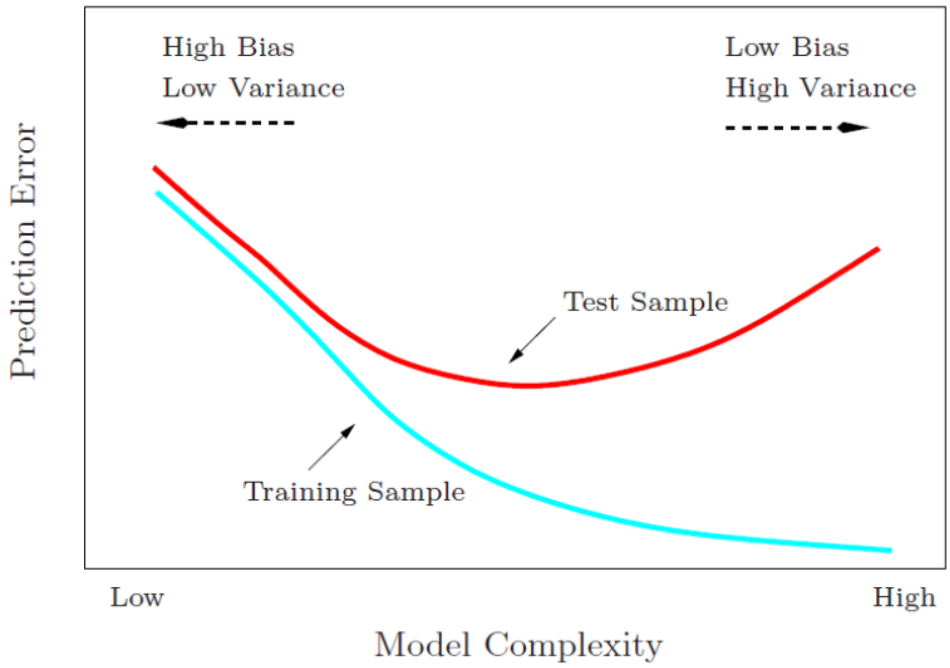
```
1: procedure UgnežđenaUnakrsnaValidacija(k)
2:     Podeliti podatke na k slojeva
3:     for all sloj ∈ slojevi do
4:         for all konfiguracija ∈ konfiguracije do
5:             Vršiti ocenu greške konfiguracije na preostalih k − 1 slojeva Algoritmom 3
6:         end for
7:         Odabrati konfiguraciju sa najmanjom greškom
8:         Trenirati model pomoću te konfiguracije na preostalih k − 1 slojeva
9:         Izvršiti predviđanje dobijenim modelom na izabranom sloju
10:    end for
11:    Računati ocenu greške
12: end procedure
```

8.1.3 Pitanje kvaliteta modela

Uzroci niskog kvaliteta modela mogu biti razni. Jedno osnovno pitanje je da li je dobro odrađena stratifikacija prilikom evaluacije. Ako smo ustanovili da je stratifikacija dobro odrađena, onda može doći do: (1) nedovoljne prilagodenosti modela i (2) preprilagodenosti modela. Nedovoljna prilagođenost modela je, na primer, ukoliko podaci dolaze iz funkcije $f(x) = x^2$, a mi ih ukalupljujemo pravom $f(x) = k \cdot x + n$. Preprilagođenost modela je ako bismo tačke koje imaju linearnu zavisnost ukalupljivali u interpolacioni polinom kroz te tačke.

Nagodba između potprilagođenosti i preprilagođenosti Ako je model nedovoljno prilagođen, onda neki od načina rešavanja problema su: koristiti prilagodljivije modele, koristiti niže vrednosti regularizacionog parametra i konstruisati nove atribute. Sa druge strane, ako je model preprilagođen, onda treba: koristiti manje prilagodljive modele, koristiti tehnike za izbor atributa, koristiti više vrednosti regularizacionog parametra i koristiti više podataka. Slika 8.1 zavisnost greške predviđanja od kompleksnosti modela.

Nedostatak informativnih atributa Ukoliko atributi nisu dovoljno informativni, nijedan algoritam učenja ne može dati pogodne rezultate. Na primer, kod klasifikacije, treba proveriti koje klase se međusobno mešaju i proveriti da li se može očekivati da postojeći atributi diskriminišu između njih. Takođe, na primeru regresije, treba proveriti da li su atributi korelirani sa ciljnom promenljivom pomoću koeficijenta korelacije i grafika vrednosti ciljne promenljive naspram vrednosti atributa.



Slika 8.1: Nagodba između potprilagođenosti i preprilagođenosti.