Podatki

Odkrivanje znanj iz podatkov se začne s podatki. Podatki so temelj gradnje vseh modelov, ki jih konstruirajo metode za podatkovno analitiko, in danes vseh metod, ki temeljijo na pristopih strojnega učenja in umetne inteligence. Brez podatkov ni mogoče učiti modelov, preverjati hipotez in izdelati napovedi. Podatki opisujejo sistem, ki ga opazujemo in za katerega bi radi ugotovili, kako deluje, kaj je v njem bolj ali manj važno, kako ga lahko izboljšamo in kako ga lahko uporabljamo.

Podatke najdemo v različnih oblikah, kot so številčne vrednosti v tabelah, besedila, slike, zaporedja ali pa jih dobimo bolj zapletenih oblikah, recimo, kot ontologije, omrežja, semantične mreže. Preden lahko podatke uporabimo, jih moramo ustrezno predstaviti, očistiti in pogosto tudi preoblikovati. V tem poglavju bomo orisali različne tipe podatkov, predstavili njihove osnovne oblike ter tehnike, s katerimi jih pripravimo za nadaljnjo obdelavo. Začeli bomo s tabelaričnimi podatki, ki predstavljajo najpogostejši in najbolj razumljiv primer strukturiranih podatkov.

Podatki v tabeli

Spodaj je primer tabelaričnih podatkov, ki beležijo karakteristike bolnišničnih preiskovancev:

ID osebe	Starost (leta)	Spol	Krvni tlak (mmHg)	Holesterol (mg/dL)	Sladkorna bolezen
P001	45	M	130/85	210	Ne
P002	52	Ž	140/90	250	Da
P003	37	M	125/80	190	Ne
P004	60	Ž	160/95	280	Da

Vsaka vrstica (razen glave tabele) predstavlja primer, v našem primeru torej posameznega preiskovanca. Vsak stolpec predstavlja značilko (Starost, Spol, Krvni tlak itd.). Značilkam v strojnem učenju pravijo tudi atributi, v statiki pa variate ali pa spremenljivke. Atributi so lahko numerični (Starost, Holesterol) ali kategorični (Spol, Ima sladkorno bolezen). Nekatere značilke nam služijo samo označevanju primerov. Takim pravimo, da so pomožne ali meta značilke.

Nekatere značilke (npr. Krvni tlak) bi morali ustrezno preoblikovati, da bi bile te primernejše za podatkovno analizo. Poznamo tudi značilke, katere vrednost bi radi napovedali na podlagi vseh ostalih značilk. V strojnem učenju takim značilkam pravimo razred, ali razredna spremenljivka, v statistiki pa uporabljamo izraz odvisna spremenljivka ter ostale uporabljene značilke poimenujemo kot neodvisne.

Formalna predstavitev tabelaričnih podatkov

Podatke, ki jih zapišemo v tabeli, lahko formalno predstavimo z matrično notacijo. Naj bo:

- X podatkovna množica, predstavljena kot matrika dimenzij $n \times m$, kjer:
  - n je število primerov (vrstic),
  - m je število značilk (stolpcev).
- $X_{i,j}$ predstavlja vrednost j-te značilke za i-ti primer.
- $\mathbf{x}_i$ je vektorska predstavitev značilk (vrstica) za i-ti primer:

$$\mathbf{x}_i = (X_{i,1}, X_{i,2}, \ldots, X_{i,m})$$

- $X_{\cdot,j}$ predstavlja stolpec j-te značilke (vse vrednosti te značilke za vse primere).

Če podatkovna množica vsebuje odvisno spremenljivko (ciljno vrednost), jo označimo kot:

- y je vektor dolžine n (ena vrednost na primer), ki predstavlja odvisno spremenljivko:

$$y = (y_1, y_2, \ldots, y_n)^T$$

  kjer y_i označuje ciljno vrednost za i-ti primer.

Za naš primer podatkovne množice preiskovancev lahko torej zapišemo:

$$X = \begin{bmatrix} 45 & \mathrm{M} & 130/85 & 210 \\ 52 & \check{\mathrm{Z}} & 140/90 & 250 \\ 37 & \mathrm{M} & 125/80 & 190 \\ 60 & \check{\mathrm{Z}} & 160/95 & 280 \end{bmatrix}$$

V primeru preiskovancev nas bi morda lahko zanimalo, ali lahko zgradimo model, ki bi ocenil, ali ima preiskovanec sladkorno bolezen. To stanje preiskovanca opišemo z vrednostjo odvisne spremenljivke y, ki jo za naš primer lahko predstavimo kot:

$$y = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \end{bmatrix}$$

Za posameznega preiskovanca lahko torej zapišemo:

- $y_1 = 0$, kar pomeni, da je preiskovanec P001 zdrav oziroma nima sladkorne bolezni,
- $y_2 = 1$, ki pove, da ima preiskovanec P002 sladkorno bolezen.

Tipi spremenljivk v tabelaričnih podatkih

V strojnem učenju lahko spremenljivke (ali značilke, variate, atribute) razvrstimo na dva načina:

1. Glede na njihovo vlogo v podatkovni množici (vhodne in izhodne, razredne in meta spremenljivke)
2. Glede na tip podatkov (realne vrednosti, diskretne, kategorične itd.)

Delitev spremenljivk glede na vlogo v podatkovni množici

Vhodne spremenljivke (značilke, kovariate ali neodvisne spremenljivke) so tiste, katerih vrednosti uporabljamo kot vhodne podatke za učenje modela. Matriko z njihovimi vrednostmi označujemo z X, pri čemer vrednost $X_{i,j}$ predstavlja vrednost značilke j za primer i.

Izhodna spremenljivka, ki jo imenujemo tudi ciljna spremenljivka, odziv, razred ali odvisna spremenljivka je tista, ki jo model skuša napovedati. Označimo jo z y, kjer je y_i ciljna vrednost za primer i. Narava te spremenljivke je odvisna od vrste problema: če gre za klasifikacijo, so vrednosti kategorične in predstavljajo razrede, kot na primer diagnoze bolezni; če gre za regresijo, pa je ciljna spremenljivka številska, torej izražena z realnimi vrednostmi. V posebnih slučajih imamo cel nabor takih spremenljivk, ki bi jih v tem primeru, podobno, kot vhodne spremenljivke, zapisali v matriki (\ Y).

Posebna vrsta spremenljivk je taka, ki jo pri strojnem učenju ne uporabljamo, lahko pa nam služi identifikaciji podatkov ali pa pri določanju konteksta. Imenujemo jih meta atributi. Take spremenljivke pogosto hranijo pomembne informacije o okoliščinah posameznega primera. Primeri vključujejo identifikacijsko številko pacienta, čas ali mesto meritve ali tip meritve.

Delitev spremenljivk glede na tip podatkov

Spremenljivke se lahko razlikujejo tudi po tipu vrednosti, ki jih lahko zavzamejo. Numerične ali kvantitativne spremenljivke izražajo količine in jih lahko razdelimo na realne in diskretne. Realne spremenljivke, imenovane tudi zvezne, lahko zavzamejo katerokoli vrednost na realni osi znotraj določenega območja. Tipični primeri vključujejo višino, težo ali temperaturo. Pri obdelavi takšnih podatkov se pogosto uporabljajo tehnike, kot sta normalizacija ali standardizacija, da prilagodimo razpone vrednosti.

Diskretne številčne spremenljivke zavzamejo le celoštevilske vrednosti. Primeri vključujejo število obiskov bolnišnice ali starost, če jo obravnavamo kot celo število, sta značilna primera. Kljub diskretni naravi jih pri modeliranju pogosto obravnavamo podobno kot realne spremenljivke.

Druga pomembna skupina so kategorične spremenljivke, ki predstavljajo razrede ali skupine. Nominalne spremenljivke nimajo naravnega vrstnega reda in zgolj razločujejo med različnimi kategorijami. Na primer, spremenljivka spola z vrednostmi »moški« in »ženska«, ali barve, kot so »rdeča«, »modra« in »zelena«, sodijo v to skupino. Za uporabo v modelih jih pogosto pretvorimo v binarne vektorje s pomočjo t. i. one-hot kodiranja ali pa jih predstavimo s številčnimi vektorji s transformacijami, ki smo se jih naučili iz podatkov.

Ordinalne spremenljivke imajo, za razliko od nominalnih, določen vrstni red, vendar razmiki med vrednostmi niso nujno enakomerni ali numerično smiselni. Primer bi bila stopnja tveganja, kot so oznake »nizko«, »srednje« in »visoko«. Pri pretvorbi v obliko, ki jo lahko uporabi model, jih pogosto preslikamo v cela števila ali numerične lestvice, vendar moramo pri tem paziti, da ne vnesemo napačnih predpostavk o razdaljah med kategorijami.

Posebno mesto imajo besedilne oziroma nizovne spremenljivke. Ko gre za prosto besedilo, kot so komentarji uporabnikov ali zdravniški zapisi, imamo opravka z nestrukturiranimi podatki, ki zahtevajo posebne pristope s področja obdelave naravnega jezika. Takšne vsebine pretvarjamo v vektorske predstavitve z uporabo modeli za vektorske vložitve, kot so word2vec ali BERT. Po drugi strani pa strukturirane nize – kot so poštne številke ali identifikacijske kode izdelkov – pogosto obravnavamo kot kategorične spremenljivke, čeprav na prvi pogled delujejo numerično. Včasih jih lahko pretvorimo tudi v številčne predstavitve, kadar to ustreza naravi naloge, še bolje pa za njih izdelamo modele za vložitve v vektorske predstavitve, ki pri vložitvah upoštevajo primerno semantiko; na primer, pri poštnih številkah geografsko lokacijo, ali pa tip naselja.

Številske predstavitvi kategoričnih spremenljivk

Pri uporabi strojnega učenja pogosto naletimo na podatke, ki jih je potrebno pred uporabo ustrezno pretvoriti v obliko, ki jo tehnika modeliranja lahko razume. To še posebej velja za diskretne podatke, kot so kategorije, besedila ali logične vrednosti, ki jih moramo prevesti v številčne predstavitve. Ena od najpogostejših metod za takšno pretvorbo je one-hot kodiranje. Vsaka možna vrednost kategorije se preslika v vektor binarnih vrednosti, pri čemer ima točno ena komponenta vrednost ena, vse ostale pa nič. Na primer, če imamo barvne oznake, kot so »rdeča«, »modra« in »zelena«, se vsaka izmed njih pretvori v svoj lastni vektor: »rdeča« postane (1, 0, 0), »modra« (0, 1, 0) in »zelena« (0, 0, 1). Ta pristop je posebej primeren za nominalne, torej neurejene spremenljivke. Njegova slabost pa je, da hitro poveča dimenzionalnost podatkov, kar lahko postane težavno, zlasti kadar imamo opravka s spremenljivkami, ki imajo veliko različnih možnih vrednosti – kot so poštne številke.

Drug pristop je ordinalno kodiranje, kjer vsaki kategoriji dodelimo celo število. To je smiselno pri spremenljivkah, kjer vrstni red med vrednostmi nosi pomen, na primer »nizek«, »srednji« in »visok«. Te bi lahko kodirali kot 1, 2 in 3. Čeprav je ta metoda učinkovita in enostavna, pa lahko povzroči težave, če model napačno interpretira razlike med številkami kot enako oddaljene oziroma linearne, kar ni vedno ustrezno.

Obstajajo tudi bolj kompleksne oblike kodiranja, ki jih pogosto imenujemo domensko kodiranje. V teh primerih poskušamo podatke organizirati na način, ki bolje izkorišča strukturo ali hierarhijo v podatkih. Na primer, namesto da bi poštne številke obravnavali kot golo številčno zaporedje ali zanje uvedli stotine binarnih spremenljivk, jih lahko najprej razvrstimo po pokrajinah, občinah ali mestih, in šele nato te skupine kodiramo s številkami. Tako združimo kategorije na bolj smiseln način in zmanjšamo dimenzionalnost.

Diskretizacija numeričnih podatkov

Zanimivo je, da v nekaterih primerih poteka pretvorba podatkov tudi v obratni smeri – iz številčnih vrednosti v diskretne kategorije. Takšna diskretizacija je koristna, kadar modeli bolje delujejo s kategoričnimi podatki, kar pogosto velja za metode, kot so odločitvena drevesa, gradnja pravil ali Bayesovih mrež. Ena izmed prednosti diskretizacije je, da olajša interpretacijo rezultatov. Namesto abstraktnih številskih vrednosti lahko uporabimo opisne oznake, kot so »nizek«, »srednji« in »visok«, kar olajša razlago napovedi in odločitev.

Diskretizacija ima tudi praktično vrednost pri zmanjševanju občutljivosti modela na manjše

spremembe v vhodnih podatkih. S tem se zmanjša vpliv šuma ali napak v meritvah, saj se majhne razlike, ki ne vplivajo na pripadnost diskretni kategoriji, pri analizi enostavno prezrejo. Poleg tega omogoča uporabo algoritmov, ki so posebej zasnovani za delo z diskretnimi podatki. Tako lahko številčne vrednosti, kot so starost ali temperatura, pretvorimo v razrede – na primer starostne skupine ali temperaturna območja – kar omogoča bolj pregledno in robustno modeliranje.

Eden od pristopov diskretizacije je razdelitev na enakomerno široke intervale. V tem primeru razdelimo območje numeričnih vrednosti v fiksno število razredov, kjer ima vsak interval enako širino. Na primer, starostne vrednosti lahko razdelimo v skupine »mlad«, »odrasel« in »starejši« glede na vnaprej določene meje. Matematično lahko to opišemo tako, da širino vsakega razreda izračunamo kot

$$w = \frac{x_{\max} - x_{\min}}{k},$$

kjer je k število razredov. Nato posamezno vrednost x_i uvrstimo v razred z

$$\text{razred}(x_i) = \left\lfloor \frac{x_i - x_{\min}}{w} \right\rfloor.$$

Drugačen pristop na primer lahko zahteva, da imajo vsi razredi približno enako število primerov. Tak pristop, znan kot diskretizacija z enako frekvenco, je še posebej uporaben, kadar so podatki nelinearno porazdeljeni. Z razvrščanjem vrednosti v kvartile, decile ali druge percentilne skupine poskrbimo, da nobena skupina ni preveč redka ali preveč pogosta, kar lahko koristi pri učenju modela.

Poleg teh enostavnih pristopov lahko za diskretizacijo uporabimo tudi metode gručenja, kjer strukturo podatkov odkrijemo z algoritmi, kot je na primer metoda voditeljev. Gručenje samodejno poišče skupine podatkovnih točk, ki so si med seboj podobne, in vsaki taki gruči dodelijo diskretno oznako. To je še posebej učinkovito pri podatkih z nelinearnimi ali kompleksnimi porazdelitvami.

Pri diskretizaciji spremenljivk se lahko zanašamo tudi na domensko znanje in podatke razdelimo ročno, glede na smiselne meje, znane iz prakse ali stroke. Temperatura je dober primer: njene vrednosti lahko pretvorimo v diskretne »mrzlo«, »zmerno« in »toplo«, pri čemer določimo prehode med kategorijami, ki so intuitivno razumljive in uporabne.

Nazadnje pa obstajajo tudi pristopi, ki pri diskretizaciji upoštevajo vrednost ciljne

spremenljivke oziroma razreda. Taki diskretizaciji lahko rečemo informirana, ali nadzorovana. Tu uporabimo metode, kot je učenje klasifikacijskega drevesa, ki na podlagi zmanjšanja entropije oziroma povečanja informacije razdeli numerične vrednosti v razrede. Tak pristop omogoča, da diskretizacijske meje niso določene zgolj na osnovi porazdelitve vhodnih podatkov, temveč tudi glede na njihovo napovedno vrednost za želen izid.

Pozveznenje

Pozveznenje pretvori kategorične ali diskretne spremenljivk v zvezne (numerične)vrednosti, tako, da jih lahko uporabimo pri tehnikah modeliranje, ki zahtevajo ali bolje delujejo s številčnimi vhodnimi podatki. Takšna pretvorba je nasprotje diskretizacije in je posebej uporabna pri algoritmih, ki temeljijo na metrikah ali predpostavkah o zveznosti, kot so linearna regresija, nevronske mreže ali metode, ki uporabljajo razdalje.

Najpogostejše oblike pozveznenja vključujejo:

- **Ordinalno kodiranje** z dodatno pretvorbo v zvezne vrednosti: ordinalne spremenljivke, kot so »nizek«, »srednji«, »visok«, se najprej kodirajo s celimi števili (1, 2, 3), nato pa po potrebi še skalirajo (npr. standardizirajo).
- **Učenje vektorskih vložitev** (angl. *embeddings*): pri bolj kompleksnih kategoričnih podatkih (npr. poštne številke, ID-ji izdelkov, SMILES nizi) uporabimo nevronske mreže za učenje gostih vektorskih predstavitev. Vsaki kategoriji priredimo vektor realnih vrednosti, ki ujame podobnosti med kategorijami na podlagi konteksta.
- **Frekvenčna kodiranja ali tarčno kodiranje**: vrednostim kategorij priredimo zvezno številko, izračunano na podlagi statistike iz učne množice – na primer povprečna vrednost ciljne spremenljivke za vsak razred. Na ta način vsaka kategorija dobi številčno oceno, ki lahko nosi informacijo o napovedni moči.

Pozveznenje ni vedno priporočljivo. Slabo izvedeno lahko v podatke vnese napačne predpostavke o razdaljah med kategorijami (npr. da je »rdeča« bliže »modri« kot »zeleni«). Zato je pomembno, da pri izbiri metode pozveznenja upoštevamo naravo podatkov, vlogo spremenljivke in zahteve modela.

Ocenjevanje razdalj

Ena od osnovnih nalog v podatkovnih vedah je ocenjevanje razdalj oziroma podobnosti med

podatkovnimi primeri. Takšna ocena omogoča, da v množici obstoječih primerov – na primer v učni množici – poiščemo tiste, ki so najbolj podobni danemu novemu primeru. Ta pristop je uporaben le, če so podatkovni primeri ustrezno pretvorjeni v številčno obliko, pri čemer mora biti izbrana tudi primerna metrika za merjenje razdalje ali podobnosti. Primeri uporabe vključujejo iskanje semantično podobnih besedil, prepoznavanje podobnih slik, iskanje pacientov s primerljivimi kliničnimi znaki, priporočanje izdelkov glede na pretekle izbire uporabnika, identifikacijo goljufivih transakcij ter razvrščanje dokumentov glede na vsebino. V vseh teh primerih je ključnega pomena, da lahko natančno in učinkovito primerjamo različne primere med seboj.

Metrike razdalj

Merjenje razdalj med primeri je osnova številnih metod v strojni analizi podatkov. V nalogah, kot so iskanje najbližjih sosedov, gručenje, iskanje podobnih dokumentov ali detekcija odstopanj, želimo količinsko oceniti, kako blizu ali daleč sta si dva primera. Različne metrike temeljijo na različnih predpostavkah o naravi podatkov in ponujajo različne matematične interpretacije razdalje.

Evklidska razdalja je najpogosteje uporabljena mera v prostoru realnih števil. Zanjo predpostavimo, da so vse spremenljivke neodvisne in enako pomembne. Če imamo dva vektorja $\mathbf{x}_i = (x_{i,1}, \ldots, x_{i,m})$ in $\mathbf{x}_j = (x_{j,1}, \ldots, x_{j,m})$, potem je Evklidska razdalja med njima:

$$d(\mathbf{x}_i, \mathbf{x}_j) = \sqrt{\sum_{k=1}^{m} (x_{i,k} - x_{j,k})^2}$$

Ta formula je naravna posplošitev razdalje med točkama v ravnini ali prostoru, vendar je občutljiva na različne lestvice vrednosti. Če imajo spremenljivke različne enote (npr. višina v cm in teža v kg), se lahko razdalja izkrivlja, zato je priporočljivo predhodno standardizirati podatke.

Razdalja Manhattan, znana tudi kot norma L1, nadomesti kvadrirane z absolutnimi vrednostmi. Določena je kot:

$$d(\mathbf{x}_i, \mathbf{x}_j) = \sum_{k=1}^{m} |x_{i,k} - x_{j,k}|$$

Uporablja se pri redkih vektorjih in je manj občutljiva na posamezne velike razlike. Njen geometrijski pomen ustreza razdalji, ki jo prevozi taksi po mreži ulic – vedno po pravokotnih poteh.

Pri obdelavi besedil in vektorskih vložitev pogosto uporabimo **kosinusno podobnost**, ki meri kot med dvema vektorjema:

$$S(\mathbf{x}_i, \mathbf{x}_j) = \frac{\mathbf{x}_i \cdot \mathbf{x}_j}{\|\mathbf{x}_i\| \cdot \|\mathbf{x}_j\|}$$

kjer je $\mathbf{x}_i \cdot \mathbf{x}_j$ skalarni produkt, $\| \cdot \|$ pa Evklidska norma. Namesto razdalje ocenjujemo podobnost: vrednost 1 pomeni, da sta vektorja vzporedna (identična usmerjenost), 0 pomeni, da sta pravokotna (nista podobna). Uporabna je pri TF-IDF predstavitvah, Word2Vec ali BERT embeddingih, kjer dolžina vektorja ni pomembna.

Jaccardova razdalja se uporablja pri množicah ali binarnih vektorjih. Zanjo štejemo delež skupnih in različnih elementov med množicama A in B:

$$d_J(A, B) = 1 - \frac{|A \cap B|}{|A \cup B|}$$

Če sta množici popolnoma enaki, je razdalja 0, če nimata nič skupnega, pa je 1. Primerna je za primerjavo naborov besed, fragmentov v DNA zaporedjih ali binarnih prstnih odtisov molekul.

Najbolj informativna – a tudi najzahtevnejša – je **Mahalanobisova razdalja**, ki upošteva kovariančno strukturo podatkov. Če je $\mathbf{x}_i$ podatkovna točka in $\boldsymbol{\mu}$ povprečje porazdelitve, razdaljo do povprečja izračunamo kot:

$$d_M(\mathbf{x}_i, \boldsymbol{\mu}) = \sqrt{(\mathbf{x}_i - \boldsymbol{\mu})^T S^{-1} (\mathbf{x}_i - \boldsymbol{\mu})}$$

kjer je S kovariančna matrika podatkov, S^{-1} pa njena inverzna. Mahalanobisova razdalja korigira dimenzije glede na njihovo razpršenost – če je neka spremenljivka zelo razpršena, ima njen prispevek k razdalji manjšo težo. Če so spremenljivke korelirane, bo razdalja med podobnimi točkami manjša, tudi če se razlikujejo v absolutni vrednosti.

Posebnost Mahalanobisove razdalje je v tem, da vodi do eliptičnih izolinij (geometrijskih mest točk z enako razdaljo), medtem ko Evklidska razdalja daje krožne izolinije. Če uporabimo enotsko kovariančno matriko, se Mahalanobisova razdalja zmanjša na običajno Evklidsko

razdaljo.

Različne metrike torej odražajo različne poglede na podobnost: Evklidska za fizično bližino, kosinusna za orientacijo, Jaccardova za deljeno vsebino, Mahalanobisova za statistično razdaljo. Uporaba prave metrike pogosto močno vpliva na uspešnost modela in interpretacijo rezultatov.

> **Opomba:** Področje, ki se je poglobljeno ukvarjalo z iskanjem podobnih primerov glede na razdaljo, je bilo sklepanje iz primerov (angl. *case-based reasoning*, CBR). Razvilo se je konec 1980-ih in bilo posebej aktivno v 1990-ih. V središču CBR je ideja, da lahko reševanje novih problemov temelji na preteklih izkušnjah oziroma podobnih primerih, pri čemer je ključna komponenta prav učinkovito merjenje razdalje med primeri. CBR se je razvijal predvsem na domenah s področja medicine, saj so bile tam že na voljo dobro dokumentirane baze primerov, poleg tega pa je bila možnost razlage odločitve na podlagi znanega podobnega primera zelo cenjena pri zdravnikih in strokovnjakih.

Prekletstvo visokih dimenzij

V nizkodimenzijskih prostorih, kjer so podatkovne točke dobro razporejene in relativno ločene, je Evklidska razdalja naravna in učinkovita metrika. Vendar pa se njena uporabnost z višanjem dimenzionalnosti močno zmanjša. Ta pojav imenujemo **prekletstvo dimenzionalnosti** in označuje, kako z rastjo števila dimenzij začnejo številne intuitivne lastnosti prostora razpadati.

Eden glavnih problemov je, da v zelo visokih dimenzijah razdalje med podatkovnimi točkami postanejo skoraj enake. Čeprav so točke še vedno numerično oddaljene, je relativna razlika med najbližjo in najbolj oddaljeno točko tako majhna, da postane nepomembna. Matematično povedano, če označimo najbližjo in najbolj oddaljeno razdaljo od neke točke $\mathbf{x}$ kot $d_{\min}$ in $d_{\max}$, potem velja:

$$\lim_{m \to \infty} \frac{d_{\max} - d_{\min}}{d_{\min}} \to 0$$

To pomeni, da izgubimo kontrast med podobnimi in različnimi primeri – nekaj, kar je za iskanje podobnosti ključno. Poleg tega vsaka nova dimenzija prinese dodatno variabilnost in šum. Če so dimenzije neodvisne, vsak nov atribut poveča pričakovano razdaljo med točkami, saj prispeva dodatni člen v vsoti kvadratov razlike:

$$d(\mathbf{x}_i, \mathbf{x}_j) = \sqrt{\sum_{k=1}^{m} (x_{i,k} - x_{j,k})^2}$$

Ob tem moramo biti previdni tudi pri enotah. Če so posamezne značilke izražene v zelo različnih lestvicah, lahko ena sama dimenzija močno prevlada in popolnoma izkrivi izračunano razdaljo. Zato je običajna praksa, da podatke predhodno standardiziramo – torej vsako značilko pretvorimo tako, da ima povprečje nič in standardni odklon ena.

Ko Evklidska razdalja postane nezanesljiva, pogosto preidemo na **kosinusno razdaljo**, ki meri kot med vektorjema, ne pa njihove absolutne razdalje. Kosinusna podobnost med dvema vektorjema $\mathbf{x}_i$ in $\mathbf{x}_j$ je definirana kot:

$$\cos(\theta) = \frac{\mathbf{x}_i \cdot \mathbf{x}_j}{\|\mathbf{x}_i\| \cdot \|\mathbf{x}_j\|}$$

in kosinusna razdalja kot:

$$d_{\cos}(\mathbf{x}_i, \mathbf{x}_j) = 1 - \cos(\theta)$$

Ker temelji na smeri in ne na dolžini vektorjev, je kosinusna razdalja bistveno manj občutljiva na redkost podatkov ali njihovo velikost. V visokodimenzionalnih prostorih, kjer so vektorji pogosto redki (večina komponent je nič), so koti med vektorji še vedno dovolj informativni za razlikovanje med primeri. Prav zaradi tega se kosinusna razdalja pogosto uporablja pri predstavitvah besedil, kot so TF-IDF, Word2Vec ali BERT embeddings, kjer vsak dokument ali beseda živi v prostoru z več sto ali tisoč dimenzijami.

Zato v visokodimenzijskih primerih ni vedno pomembno, kako daleč sta si točki po absolutni vrednosti, temveč kako usmerjeni sta – to je tisto, kar kosinusna metrika uspešno zajame.

Primer uporabe razdalj med primeri: iskanje osamelcev

Ko podatke predstavimo kot vektorje in definiramo metrik za računanje razdalj, lahko začnemo iskati osamelce – točke, ki so »drugačne« od večine. Osamelci pogosto kažejo na napake, redke dogodke ali zanimive vzorce v podatkih. Identifikacija osamelcev temelji bodisi na oddaljenosti točk, bodisi na njihovi lokalni gostoti.

Ena izmed osnovnih metod je identifikacija osamelcev na podlagi **razdalje do najbližjih sosedov**. Za vsako točko izračunamo povprečno razdaljo do njenih k najbližjih sosedov in

označimo tiste s posebej velikimi povprečnimi razdaljami kot osamelce. Če je $\mathbf{x}_i$ točka, potem definiramo:

$$d_k(\mathbf{x}_i) = \frac{1}{k} \sum_{\mathbf{x}_j \in kNN(\mathbf{x}_i)} d(\mathbf{x}_i, \mathbf{x}_j)$$

Točke z velikimi vrednostmi d_k so najverjetneje osamelci. Ta pristop je enostaven, vendar je lahko občutljiv na globalno porazdelitev podatkov. Uporabljene metrike razdalje so odvisne od narave podatkov – Evklidska za nizke dimenzije, kosinusna za redke visoko-dimenzionalne vektorje, Mahalanobisova za podatke z značilno korelacijo med spremenljivkami.

Za boljšo prilagoditev lokalni strukturi podatkov uporabimo **LOF – lokalni faktor osamelcev (Local Outlier Factor)**. Osnovna ideja je, da primerjamo gostoto točke z gostotami njenih sosedov. Če je gostota točke bistveno nižja, jo označimo kot osamelca.

Najprej definiramo **lokalno gostoto** kot inverzno povprečno dosegljivo razdaljo do sosedov:

$$lrd_k(\mathbf{x}_i) = \frac{k}{\sum_{\mathbf{x}_j \in kNN(\mathbf{x}_i)} \text{reach-dist}_k(\mathbf{x}_i, \mathbf{x}_j)}$$

kjer je dosegljiva razdalja definirana kot:

$$\text{reach-dist}_k(\mathbf{x}_i, \mathbf{x}_j) = \max\left\{\text{k-dist}(\mathbf{x}_j),\, d(\mathbf{x}_i, \mathbf{x}_j)\right\}$$

Tu $\text{k-dist}(\mathbf{x}_j)$ označuje razdaljo od točke $\mathbf{x}_j$ do njenega k-tega najbližjega soseda, s čimer zagotovimo robustnost glede na nenavadno bližnje sosede.

Končni **LOF** za točko $\mathbf{x}_i$ izračunamo kot razmerje povprečne gostote njenih sosedov in njene lastne gostote:

$$LOF_k(\mathbf{x}_i) = \frac{1}{k} \sum_{\mathbf{x}_j \in kNN(\mathbf{x}_i)} \frac{lrd_k(\mathbf{x}_j)}{lrd_k(\mathbf{x}_i)}$$

Če je ta količina večja od 1, potem ima točka bistveno nižjo gostoto kot njeni sosedi, kar kaže na osamljeno lego.

Ta metoda ima več prednosti: prilagaja se lokalni gostoti in ni občutljiva na neenakomerno porazdelitev podatkov. V regijah, kjer so podatki redkejši, točke niso avtomatsko obravnavane kot osamelci, razen če so redke tudi znotraj svojega lokalnega okolja.

Zaradi teh lastnosti je LOF priljubljeno orodje za odkrivanje anomalij v različnih podatkovnih domenah, vključno s financami, medicino in kibernetsko varnostjo.

Natabelarični podatki

V strojnem učenju poleg tabelaričnih, torej strukturiranih podatkov, pogosto obdelujemo tudi podatke, ki niso enotno organizirani. Sem spadajo nestrukturirani in polstrukturirani podatki, ki izhajajo iz vsakdanjih virov, kot so besedila, slike, zvoki, zaporedja ali omrežja. Vsaka od teh vrst podatkov ima posebnosti, ki vplivajo tako na predobdelavo kot tudi na izbiro ustreznih modelov.

Besedilni podatki, kot so novice, e-poštna sporočila, spletni komentarji ali zdravniška poročila, tipično nimajo vnaprej določene strukture in se lahko močno razlikujejo po dolžini in slogu. Njihova interpretacija zahteva razumevanje pomena in konteksta, zato jih najprej pretvorimo v numerično obliko. Klasične metode, kot sta vreča besed (angl. *bag-of-words*) in TF-IDF (angl. *term-frequency-inverse-document-frequency), štejejo pogostost besed ali tehtajo njihovo pomembnost. Naprednejši pristopi, ki besed predstavijo z številskimi vektorji (Word2Vec, GloVe) in transformacijski modeli (BERT, GPT), podpirajo predstavitve z globljim, semantično razumevanjem, ki izhajajo iz konteksta, v katerem se te besede uporabljajo. Tovrstne predstavitve so temelj za naloge, kot so strojno prevajanje, analiza sentimenta, iskanje po vsebini in interakcijo z uporabniki prek klepetalnih sistemov.

Slikovni podatki, kot so fotografije, medicinski posnetki ali satelitske slike, predstavljajo visoko-dimenzionalne matrike pikslov oziroma barvnih točk. Vsaka slika vsebuje številčne vrednosti za barvne komponente posameznih točk, pogosto organizirane v kanale R, G in B. Klasične metode so temeljile na ročno določenih značilnicah (kot sta SIFT in HOG), medtem ko danes prevladujejo konvolucijske nevronske mreže (CNN), ki se same naučijo prepoznavati pomembne vzorce. Uporaba slik v strojnem učenju obsega prepoznavanje predmetov, medicinsko diagnostiko, krmiljenje samovozečih vozil in analizo kompleksnih vizualnih vzorcev.

Posebna oblika strukturiranih podatkov so omrežja oziroma grafi, kjer vozlišča predstavljajo enote (npr. osebe, atome v molekulah, geografske lokacije), povezave med njimi pa odnose ali interakcije. Takšne podatke najdemo v socialnih omrežjih, molekularni biologiji, kemiji ali pri logističnih sistemih. Ker klasične metode analize ne delujejo neposredno na grafih, grafe in njihove podgrafe ter vozlišča predstavljamo z matrikami sosednosti ali uporabimo metode, ki

ustvarijo vektorske predstavitve vozlišč (npr. GraphSAGE, GCN). Ti pristopi omogočajo napovedovanje povezav ali pa tipa povezav, odkrivanje skupnosti in analizo kompleksnih odvisnosti.

Zaporedni oziroma sekvenčni podatki se pojavljajo tam, kjer je vrstni red pomemben – na primer pri časovnih vrstah, bioloških nizih ali signalih. Podatki, kot so vrednosti senzorjev, finančne meritve ali DNA zapisi pogosto vsebujejo odvisnosti med preteklimi in prihodnjimi vrednostmi. Za obdelavo takšnih podatkov se uporabljajo metode, kot so rekurentne nevronske mreže (RNN, LSTM) in transformerji, ki omogočajo učenje dolgoročnih vzorcev. Uporaba teh tehnik je ključna za napovedovanje, diagnostiko in razumevanje kompleksnih procesov v času.

Prostorski podatki vključujejo informacije o lokaciji in prostorskih odnosih, kot so geolokacije, kartografski podatki ali točke na slikah tipa lidar. Zahtevajo posebno obravnavo, saj vsebujejo prostorske zakonitosti, ki niso zajete z običajnimi značilkami. Podatke lahko pretvorimo v numerično obliko z ustreznimi koordinatnimi transformacijami ali prostorskimi vgraditvami. Uporabni so v urbanističnem načrtovanju, analizi gibanja, okoljskem modeliranju in kartiranju.

Nazadnje imamo še zvok in signalne podatke, ki vključujejo avdio posnetke, govorne signale ali fiziološke meritve, kot je EEG. Takšni podatki se pogosto analizirajo v frekvenčni domeni, pri čemer uporabljamo metode, kot so Fourierjeva transformacija ali MFCC koeficienti. Sodobni pristopi uporabljajo nevronske mreže, prilagojene za obdelavo zvoka (npr. WaveNet), ki omogočajo prepoznavanje govora, diagnostiko strojev ali analizo biofizikalnih signalov.

Vsaka od navedenih vrst podatkov ima svoj nabor izzivov in ustreznih metod za pretvorbo v obliko, ki jo modeli strojnega učenja lahko učinkovito obdelajo. Ključno je razumevanje strukture podatkov, saj to določa, kako jih lahko modeliramo, interpretiramo in uporabimo za napovedovanje ali odločanje.

Primer: analiza DNA niza in njegova pretvorba v numerično predstavitev s k-meri

Za ponazoritev metode kodiranja k-merov lahko uporabimo krajše DNA zaporedje, pri katerem nekatere možne kombinacije sploh ne nastopajo. To omogoča enostavnejšo predstavitev vektorske oblike, kjer bodo nekatere vrednosti nič.

Predpostavimo DNA zaporedje:

$$\text{DNA} = \text{”ATGCGT”}$$

Za $k = 3$ (trimere) dobimo naslednje k-mere:

Pozicija	3-mer
1–3	ATG
2–4	TGC
3–5	GCG
4–6	CGT

Če predpostavimo, da upoštevamo vse možne 3-mere (skupno jih je $4^3 = 64$), lahko vsak 3-mer predstavimo v vektorju dolžine 64. Uporabimo metodo **bag-of-k-mers**, kjer štejemo pojavitve:

- ATG: 1
- TGC: 1
- GCG: 1
- CGT: 1
- vse ostale: 0

Tako dobimo redko predstavitev, kjer je vektor $\mathbf{x}$ dolžine 64 in vsebuje le nekaj enic:

$$\mathbf{x} = (0, 0, \ldots, 1, \ldots, 1, \ldots, 0)$$

Konkretno bi vektor imel vrednost 1 na tistih mestih, ki ustrezajo pojavitvam ATG, TGC, GCG in CGT, vse ostale komponente pa bi bile enake 0. Takšna oblika predstavitve je še posebej primerna za algoritme, ki dobro obvladujejo redke (sparse) podatke.

Učenje vektorskih vložitev je ključno pri analizi bioloških zaporedij, saj omogoča, da vsak k-mer predstavimo kot vektor v večdimenzionalnem prostoru, pri čemer so si podobni k-meri tudi bližje. Dve glavni metodi za učenje takšnih predstavitev sta Continuous Bag-of-Words (CBOW) in Skip-gram, ki sta bili prvotno razviti za obdelavo naravnega jezika (Word2Vec), kasneje pa uspešno preneseni tudi na biološke zapise, kot je DNA (npr. DNA2Vec).

Metoda CBOW temelji na ideji, da lahko pomen besede oziroma k-mera razberemo iz njegovega konteksta. Model dobi več okoliških k-merov in iz njih poskuša napovedati

manjkajočega, torej tistega, ki se nahaja na sredini. V kontekstu naravnega jezika si to lahko predstavljamo tako, da v stavku "*Maček spi na okenski polici*" model dobi besede »Maček«, »na«, »okenski« in »polici« ter iz njih poskuša napovedati besedo »spi«. Matematično to pomeni, da pri zaporedju $w_1, w_2, ..., w_T$ model maksimizira verjetnost ciljne besede w_t, dane njenih sosednjih besed:

$$P(w_t|w_{t-n}, ..., w_{t-1}, w_{t+1}, ..., w_{t+n})$$

Podobno lahko metodo uporabimo tudi za k-mere v DNA zaporedjih. Če imamo zaporedje k-merov, na primer:

$$\text{”ATG CGA TGA ACC”}$$

in nastavimo širino okna na 2, potem bo model iz konteksta, ki vključuje »ATG« in »TGA«, napovedoval k-mer »CGA«. Na ta način model postopoma zgradi vektorske predstavitve, pri katerih so k-meri, ki se pojavljajo v podobnih kontekstih, v prostoru bliže drug drugemu.

CBOW je v primerjavi s Skip-gram hitrejši in primeren za velike količine podatkov, saj povpreči kontekstne vektorje in trenira model, da iz njih napove ciljni k-mer. Vendar pa ima slabšo ločljivost pri redkih k-merih, saj zgladi informacije in zato lahko slabše razlikuje redke pojavitve. Kljub temu ostaja priljubljena metoda za začetno učenje vložitev pri analizi bioloških nizov.

Druga pogosto uporabljena metoda za učenje vektorskih predstavitev k-merov je Skip-gram. V nasprotju s CBOW, kjer model iz konteksta napoveduje manjkajočo besedo, pri Skip-gram pristopu iz posamezne besede – ali v našem primeru k-mera – model napoveduje njegove okoliške elemente. Na ta način se uči, kateri k-meri se pogosto pojavljajo skupaj, kar omogoča, da so si v končnem vektorskem prostoru bliže tisti k-meri, ki nastopajo v podobnih kontekstih.

V primeru naravnega jezika si lahko ta pristop predstavljamo na stavku "*Maček spi na okenski polici*". Tukaj bi model dobil ciljno besedo »spi« in iz nje poskušal napovedati sosednje besede, kot so »Maček«, »na«, »okenski« in »polici«. Z drugimi besedami, model se uči verjetnost, da se določene besede pojavijo v okolici dane besede. Matematično je cilj maksimizirati:

$$P(w_{t-n}, ..., w_{t+n}|w_t)$$

V primeru bioloških nizov je logika enaka. Recimo, da imamo niz:

```
s = "ATG CGA TGA ACC"
```

in uporabimo Skip-gram z oknom velikosti 2. Potem bo za ciljni k-mer »CGA« model napovedoval njegove okoliške k-mere, torej »ATG« in »TGA«. Tako se nauči, da se »CGA« pogosto pojavlja skupaj z njima, kar se v procesu učenja prenese v podobne vektorske predstavitve.

Glavna prednost Skip-gram metode je, da je učinkovitejša pri učenju reprezentacij redkih k-merov, saj vsak primer omogoča več napovedi. To pomeni, da tudi manj pogosti zapisi prispevajo k učenju uporabne predstavitve. Vendar pa je cena za to večja računska zahtevnost, saj je treba za vsako ciljno besedo generirati več napovedi, kar model dela počasnejši v primerjavi s CBOW pristopom. Kljub temu je Skip-gram zaradi svoje natančnosti pri redkih pojavih pogosto prednostna izbira, kadar delamo z biološkimi podatki, ki so pogosto heterogeni in vključujejo številne redke k-mere.

Primer: kemijske strukture

Kemijske strukture lahko učinkovito predstavimo z uporabo SMILES zapisa (*Simplified Molecular Input Line Entry System*), ki omogoča, da molekulo opišemo kot niz znakov. Vsak simbol predstavlja atom, vez ali posebno funkcionalno skupino, kar omogoča enostavno shranjevanje, iskanje in obdelavo molekul v računalniških sistemih. Na primer, aspirin (acetilsalicilna kislina) je v SMILES zapisu predstavljen kot:

```
CC(=O)Oc1ccccc1C(=O)O
```

V tem zapisu se `CC(=O)O` nanaša na acetoksilno skupino, `c1ccccc1` predstavlja benzenski obroč, `C(=O)O` pa karboksilno skupino. SMILES zapis je priljubljen v kemijski informatiki, saj omogoča enostavno pretvorbo kemijske strukture v računalniško berljivo obliko, ki je primerna za nadaljnjo numerično analizo.

Za uporabo v strojnem učenju je treba SMILES zapise pretvoriti v numerične predstavitve. Ena izmed najpreprostejših metod je **one-hot kodiranje**, kjer vsak možni znak v zapisu dobi svoj položaj v binarnem vektorju. Če poznamo nabor 20 različnih znakov, se vsaka molekula pretvori v matriko, kjer vsaka vrstica ustreza enemu znaku v zapisu, stolpci pa predstavljajo posamezne znake iz nabora.

Naprednejši pristop so **molekularni prstni odtisi (fingerprints)**, kot sta **Morgan fingerprint (ECFP)** ali **MACCS ključne lastnosti**. Morgan fingerprint deluje tako, da se molekula obravnava kot graf, kjer se iz atomov izhaja krožno in beleži lokalne podstrukture. Tako nastane bitni vektor fiksne dolžine, kjer vsak bit označuje prisotnost določenega fragmenta. V primeru aspirina bi lahko v odtisu našli aktivne bite za benzenski obroč, karboksilno skupino, ester in ketonsko vez.

Drug pristop je pretvorba SMILES zapisa v molekularni graf, kjer so atomi predstavljeni kot vozlišča, vezi pa kot povezave med njimi. Takšne grafe lahko obdelamo z **grafovskimi nevronskimi mrežami (Graph Neural Networks, GNNs)**, ki se naučijo predstavitev molekul glede na njihovo topologijo in kemijske lastnosti.

Če SMILES obravnavamo kot zaporedje znakov, podobno kot pri naravnem jeziku, lahko uporabimo tudi **sekvenčne modele**, kot so **RNN**, **LSTM** ali **Transformerji**, ki omogočajo učenje kompleksnih zaporednih vzorcev in dolgoročnih odvisnosti v molekulskih nizih.

Primer za aspirin prikazuje, kako lahko z različnimi metodami ustvarimo numerične predstavitve:

1. V obliki one-hot matrike, kjer je vsak znak predstavljen z binarnim vektorjem.
2. Kot 1024-bitni ECFP fingerprint, kjer vsak bit označuje prisotnost specifične podstrukture.
3. Kot grafovna struktura, ki jo obdelamo z GNN modelom.

Morgan fingerprint je še posebej uporaben, saj poleg prisotnosti določenih fragmentov upošteva tudi njihov položaj in kontekst znotraj molekule. Na primer, za aspirin lahko med značilnimi fragmenti zaznamo benzen, karboksilno in ester skupino, etilni del in ketonsko vez. Vektor bo za aspirin vseboval nekaj aktivnih bitov (vrednost 1), ki označujejo prisotnost teh funkcionalnih skupin, medtem ko bodo vsi ostali biti enaki 0:

```
00001000100001000010001000000001000000000... (1024 bitov)
```

Takšni vektorji omogočajo učinkovito iskanje podobnosti med molekulami, napovedovanje lastnosti ali biološke aktivnosti spojin ter klasifikacijo v kemijskih bazah. Zaradi svoje učinkovitosti in robustnosti je Morgan fingerprint ena izmed najbolj uporabljenih tehnik za predstavitev molekul v aplikacijah strojnega učenja.

Primer: slike

Slike so eden najpogostejših virov podatkov v sodobnem strojnem učenju in računalniškem vidu. Za njihovo uporabo v algoritmih jih je treba pretvoriti v numerično obliko, ki omogoča matematično obdelavo. Osnovna predstavitev slike temelji na matrični zgradbi, kjer vsak element ustreza vrednosti svetlosti posameznega piksla.

Pri sivinskih slikah dobimo dvodimenzionalno matriko velikosti $H \times W$, kjer H in W predstavljata višino in širino slike, vsak element matrike pa je število med 0 in 255. Na primer:

$$I = \begin{bmatrix} 12 & 45 & 78 & 200 \\ 34 & 89 & 120 & 255 \\ 100 & 140 & 190 & 220 \end{bmatrix}$$

Pri barvnih slikah pa imamo tri take matrike – po eno za vsako barvno komponento: rdečo (R), zeleno (G) in modro (B). Tako dobimo tridimenzionalno strukturo velikosti $H \times W \times 3$, kjer je vsak piksel opisan s trojico števil.

Takšne podatke lahko nato razporedimo v enodimenzionalne vektorje, postopek imenujemo flattening. Tako sliko velikosti 28×28 pikslov pretvorimo v vektor dolžine 784. Ta oblika je primerna za algoritme, ki zahtevajo vektorje fiksne dolžine, vendar pogosto izgubi prostorske informacije, kar je pomembno pri analizi slik.

Zato slike pogosto raje predstavimo prek značilk, ki zajemajo pomembne informacije iz vsebine slike. Klasične metode vključujejo histogram orientiranih gradientov (HOG), ki zajema robove in strukture, ter metodo SIFT, ki zazna značilne točke neodvisno od merila in orientacije. Z razvojem globokega učenja pa so prevladale konvolucijske nevronske mreže (CNN), ki se same naučijo izluščiti pomembne značilke iz slik. Vsaka plast CNN-ja postopoma zajame vedno bolj abstraktne lastnosti, od robov do kompleksnih oblik.

Ena izmed uporabnih oblik predstavitve so vektorske vložitve slik v nižedimenzionalne prostore. Pri tem lahko uporabimo autoencoderje, ki sliko najprej stisnejo v kompaktno reprezentacijo, nato pa iz nje poskušajo rekonstruirati izvirno sliko. Dobljeni stisnjeni vektor vsebuje ključne informacije o sliki in je uporaben za klasifikacijo ali iskanje podobnosti. Podobno lahko uporabimo predtrenirane modele, kot sta ResNet ali VGG, ki slike pretvorijo v visokodimenzionalne vektorje, na primer dolžine 512 ali 2048. Ti vektorji ohranijo bogato semantično predstavitev slike in so široko uporabni v iskalnikih slik, detekciji objektov ali prenosu znanja na nove naloge.

Primer: besedila

Besedilni podatki so zaradi svoje narave nestrukturirani in zato zahtevajo pretvorbo v številčno obliko, da jih lahko obravnavamo z algoritmi strojnega učenja. Ta pretvorba se lahko izvede na več načinov, od preprostih pristopov, ki temeljijo na štetju besed, do naprednih modelov, ki zajamejo pomen in kontekst.

Ena od osnovnih metod je **one-hot kodiranje**, kjer vsaki besedi iz besednjaka dodelimo binarni vektor, v katerem ima ena sama komponenta vrednost 1, vse ostale pa 0. Na primer, če imamo stavek "*Maček spi na polici*", potem bo vsaka beseda predstavljena z vektorjem, ki označuje njen položaj v besednjaku. Slabost tega pristopa je, da se velikost vektorjev hitro povečuje s številom različnih besed, poleg tega pa vektorji ne vsebujejo nobenih informacij o pomenu ali sorodnosti med besedami.

Podoben, a bolj kompakten pristop je **bag-of-words (BOW)**, kjer ne zabeležimo samo prisotnosti besede, temveč tudi njeno pogostost. V besedilu "*Maček spi. Maček prede.*" bomo tako dobili vektor (2, 1, 1), kar pomeni, da se »Maček« pojavi dvakrat, »spi« enkrat in »prede« enkrat. BOW sicer zajame frekvenco, a še vedno ne upošteva vrstnega reda besed, zato izgubi sintaktične in semantične informacije.

Za boljšo razločevanje med pomembnimi in nepomembnimi besedami uporabimo metodo **TF-IDF (Term Frequency – Inverse Document Frequency)**. Ta poveča težo redkejših, a informativnih besed, in zmanjša vpliv pogosto uporabljenih, kot so vezniki ali predlogi. Vsaki besedi se tako dodeli utež, izračunana kot produkt pogostosti v danem dokumentu in logaritma obratne pogostosti v celotni zbirki dokumentov. Rezultat so bolj razločne predstavitve, ki so še vedno redke in linearne, a bolj informativne kot BOW.

Z napredkom v razumevanju jezika so se razvili **word embeddings**, kjer se vsaka beseda predstavi kot gost vektor v večdimenzionalnem prostoru. Modeli, kot so **Word2Vec**, **FastText** in **GloVe**, se naučijo takšnih vektorjev iz velikih besedilnih korpusov. Rezultat so vektorji, kjer so si pomensko podobne besede blizu, kar omogoča semantične operacije, kot je: *kralj - moški + ženska ≈ kraljica*. FastText še dodatno upošteva dele besed, kar omogoča boljše delo z redkimi ali neznanimi izrazi.

Sodobni pristopi pa temeljijo na **kontekstualnih vektorskih modelih**, kot so **BERT**, **GPT** ali **T5**, ki uporabljajo transformacijsko arhitekturo in besedam dodelijo vektorje glede na njihov pomen v stavku. Tako ima beseda »banka« v stavku "*Odšel sem v banko.*" drugačno

predstavitev kot v stavku "*Ribič sedi na bregu banke.*" Ti modeli upoštevajo celoten kontekst in so postali standard v nalogah, kjer je razumevanje pomena in sintakse ključno, kot so strojno prevajanje, razvrščanje besedil ali klepetalni agenti.

Različne metode torej ponujajo različno stopnjo kompleksnosti in izrazne moči. Enostavne metode, kot so one-hot kodiranje, BOW ali TF-IDF, so primerne za manjše projekte in osnovno analizo, medtem ko napredne metode, kot so word embeddings in kontekstualni modeli, omogočajo globlje semantično razumevanje in boljše rezultate pri kompleksnih nalogah.

Domensko predznanje

Posebna vrsta podatkov, ki močno prispevajo k razumevanju rezultatov in delovanju modelov strojnega učenja, so podatki, ki opisujejo vedenjske zakonitosti ali problemsko domeno. Ti **dodatni domenski podatki** vključujejo strukture znanja, besedne opise, slike in formalizirana pravila, ki ne služijo le kot vhodne značilke, temveč tudi kot razlaga modelov ali način za omejevanje napovedi v realno smiselne okvire.

Med najpomembnejšimi oblikami so **strukturni domenski podatki**, kot so ontologije in taksonomije. V biomedicini se pogosto uporabljajo specializirane ontologije, denimo Gene Ontology za funkcije genov ali SNOMED CT za klasifikacijo bolezni. Podobno omogočajo taksonomije, kot je Linnejeva klasifikacija vrst, organizirano združevanje podobnih entitet. Takšne strukture pomagajo modelom razumeti hierarhične odnose med pojmi: na primer, če model napove učinek nekega gena, lahko z ontološkim znanjem razložimo, kako ta vpliva na biološki proces. Poleg tega takšna znanja omogočajo filtriranje nedoslednih napovedi – če model predlaga rezultat, ki je neskladen z znanim ontološkim pravilom, ga lahko zavrnemo ali vsaj utemeljimo.

Pomembno vlogo igrajo tudi **besedni opisi** značilk, primerov ali napovedi. Če so posamezne spremenljivke v podatkovnem naboru opremljene z naravnojezikovnimi opisi, jih je lažje razložiti ne-strokovnjakom. Besedne razlage simptomov v medicinskih zapisih ali povzetki modelskih odločitev izboljšajo transparentnost postopka. Še posebej pri uporabi velikih jezikovnih modelov (LLM), kot je GPT, lahko ti opisi služijo kot vhod za generiranje samorazlagajočih se napovedi, kar premošča vrzel med številčnimi podatki in človeškim razumevanjem.

Tretji tip dopolnilnih podatkov predstavljajo **slikovni in vizualni opisi**, ki omogočajo razlago

modelskih odločitev v kontekstu vizualnih vzorcev. Na primer, pri klasifikaciji mikroskopskih slik celic ali satelitskih posnetkov model lahko označi, katera območja slike so bila ključna za napoved. Tehnike, kot je Grad-CAM pri konvolucijskih nevronskih mrežah, ustvarjajo toplotne zemljevide pozornosti, ki razložijo, zakaj je bila sprejeta določena odločitev. Če modele dopolnimo še z numeričnimi ali besedilnimi podatki, lahko ustvarimo multimodalne sisteme, ki razlagajo slike z besedami in številkami.

Poleg teh obstajajo še **pravilne omejitve in logična pravila**, ki modelom nalagajo formalne okvire. Pravila v obliki IF-THEN ali zapisane v jeziku Prolog ali OWL lahko nadzorujejo, da model ne predlaga rezultatov, ki so v nasprotju z medicinsko, fizikalno ali drugo strokovno logiko. Ta pristop ni le razložljiv, temveč tudi varen, saj modelu preprečuje nesmiselne sklepe – na primer napovedovanje negativne mase ali fiziološko nemogočih vrednosti.

Uporaba takšnih domenskih podatkov je posebej dragocena na področjih, kjer razlaga ni zgolj dodatek, temveč nuja – kot so medicina, biološke raziskave, pravni sistemi ali varnostno-kritični sistemi. S kombinacijo številskih, simbolnih in kontekstualnih informacij lahko modeli postanejo bolj razumljivi, bolj zanesljivi in bolj uporabni v realnem svetu.