UNIVERZITET U NIŠU

ELEKTRONSKI FAKULTET

SEMINARSKI RAD

Transformacija podataka

Student
Stefan Jovanović 1959

Profesor
Prof. dr Aleksandar Stanimirović

Sadržaj

Uvod

Definicija preprocesiranja podataka

Preprocesiranje podataka predstavlja početnu i jednu od najvažnijih faza u procesu analize podataka i izgradnje modela mašinskog učenja. Njegova osnovna svrha je da pripremi sirove podatke tako da budu tačni, konzistentni, potpuni i pogodni za dalju analizu.

U praksi, podaci koji se prikupe iz različitih izvora, kao što su senzori, ankete, baze podataka ili internet, često su nepotpuni, duplirani ili u neujednačenim formatima. Takvi podaci mogu dovesti do pogrešnih zaključaka ili smanjene tačnosti modela. Preprocesiranjem se ti problemi rešavaju kroz niz sistematskih koraka kao što su čišćenje podataka, transformacija, normalizacija, kodiranje kategorija i smanjenje dimenzionalnosti.

Drugim rečima, preprocesiranje podataka obuhvata sve tehnike koje poboljšavaju kvalitet i upotrebljivost podataka kako bi se omogućila pravilna interpretacija i efikasna primena analitičkih i prediktivnih modela. U statistici se kaže da kvalitet ulaznih podataka određuje kvalitet izlaznih rezultata, zato se preprocesiranje smatra ključnim korakom koji direktno utiče na uspešnost svakog analitičkog projekta.

Uloga i značaj u procesu analize podataka i mašinskog učenja

Uloga preprocesiranja podataka u analizi i mašinskom učenju je osnovna i nezamenljiva, jer kvalitet ulaznih podataka direktno određuje kvalitet rezultata koji model može da postigne. Bez adekvatne pripreme, čak i najnapredniji algoritmi neće moći da pruže tačne ili pouzdane zaključke. U procesu analize podataka, preprocesiranje omogućava pretvaranje sirovih i neuređenih podataka u strukturisane i čiste skupove koji se mogu analizirati pomoću statističkih i analitičkih metoda. Time se eliminišu greške, nedoslednosti i nedostajuće vrednosti koje bi mogle iskriviti rezultate i dovesti do pogrešnih interpretacija.

U kontekstu mašinskog učenja, preprocesiranje ima još veći značaj. Algoritmi mašinskog učenja, poput regresije, klasifikacije ili klasterovanja, zahtevaju da podaci budu numerički, skalirani i homogeni, kako bi mogli da uče obrasce efikasno. Na primer, ako se ne izvrši normalizacija, osobina sa većim rasponom vrednosti može dominirati modelom, čime se smanjuje njegova tačnost. Pored toga, preprocesiranje doprinosi i povećanju efikasnosti modela. Kroz smanjenje dimenzionalnosti i uklanjanje nerelevantnih atributa postiže se brža obrada i manja verovatnoća pretreniranja.

Zbog svega navedenog, preprocesiranje se često naziva temeljem svakog projekta baziranog na podacima. Ono obezbeđuje da podaci budu kvalitetni, reprezentativni i spremni za analitičke i prediktivne procese, čime se značajno povećava vrednost i tačnost krajnjih rezultata.

Ciljevi preprocesiranja

Glavni cilj preprocesiranja podataka je da se poveća kvalitet, tačnost i pouzdanost podataka koji se koriste za analizu i mašinsko učenje. Kroz niz metoda i tehnika, preprocesiranje omogućava da se podaci očiste, standardizuju i prilagode tako da modeli mogu pravilno da uče i daju relevantne rezultate.

Najvažniji ciljevi preprocesiranja su:

1. Povećanje tačnosti modela

Kvalitetno preprocesirani podaci omogućavaju algoritmima mašinskog učenja da bolje prepoznaju obrasce i veze između promenljivih. Kada se uklone greške, duplikati i nedoslednosti, model uči iz realnih i reprezentativnih podataka, što vodi ka većoj preciznosti i boljoj generalizaciji na novim skupovima podataka.

2. Smanjenje šuma u podacima

Pod šumom se podrazumevaju slučajne ili pogrešne vrednosti koje ne predstavljaju stvarno stanje. Šum može nastati usled ljudske greške, tehničkih problema pri merenju ili neispravnih senzora. Preprocesiranjem se šum otkriva i uklanja pomoću statističkih metoda, filtriranja ili algoritama za detekciju outlajera, čime se smanjuje uticaj pogrešnih informacija na model.

3. Poboljšanje kvaliteta podataka

Podaci često sadrže nedostajuće vrednosti, neusklađene formate ili različite jedinice mere. Cilj preprocesiranja je da se takvi podaci očiste i usklade - popunjavanjem praznina, konverzijom formata i standardizacijom. Time se obezbeđuje da svi podaci budu kompatibilni i spremni za dalju obradu, analizu ili modelovanje.

U suštini, preprocesiranje je proces kojim se obezbeđuje da podaci postanu upotrebljivi, relevantni i tačni, čime se postavljaju temelji za donošenje ispravnih zaključaka i uspešnu primenu analitičkih i mašinskih modela.

Tipovi podataka i problemi u sirovim podacima

Vrste podataka

U procesu preprocesiranja podataka, jedan od prvih i najvažnijih koraka je prepoznavanje vrste podataka sa kojima se radi. Različite vrste podataka zahtevaju različite pristupe čišćenju, transformaciji i analizi.

Najčešće se podaci dele na numeričke, kategoričke, tekstualne i vremenske.

1. Numerički podaci

Numerički podaci predstavljaju brojčane vrednosti koje se koriste za kvantitativno izražavanje određenih osobina ili pojava. Primeri: visina, težina, temperatura, prihod, ocene.

Oni se mogu dalje podeliti na:

- **Diskretne podatke** - imaju ograničen broj mogućih vrednosti (npr. broj grešaka u kodu).
- **Kontinuirane podatke** - mogu poprimiti bilo koju vrednost unutar određenog intervala (npr. temperatura, težina).

Kod preprocesiranja numeričkih podataka često se primenjuju metode kao što su skaliranje (normalizacija, standardizacija), uklanjanje outlajera i popunjavanje nedostajućih vrednosti.

2. Kategorički podaci

Kategorički podaci opisuju kvalitativne osobine i sastoje se od kategorija ili oznaka koje ne predstavljaju brojčanu veličinu. Primeri: pol (muško/žensko), boja (crvena, plava, zelena).

Delimo ih na:

- **Nominalne** - redosled kategorija nije bitan (npr. boja).
- **Ordinalne** - kategorije imaju logičan redosled (npr. ocene: loše, srednje, dobro, odlično).

U preprocesiranju se kategorički podaci najčešće kodiraju u numerički oblik pomoću tehnika kao što su One-Hot Encoding ili Label Encoding, kako bi ih modeli mogli obraditi.

3. Tekstualni podaci

Tekstualni podaci sadrže neuređene reči, rečenice ili dokumente, najčešće prikupljene iz poruka, komentara, anketa ili društvenih mreža. Primeri: recenzije proizvoda, korisnički komentari, email poruke.

Za njihovo preprocesiranje koriste se tehnike iz oblasti obrade prirodnog jezika (NLP), kao što su:

- čišćenje teksta (uklanjanje znakova interpunkcije, brojeva i “stop” reči),
- tokenizacija (razdvajanje teksta na reči),
- lematizacija i stemovanje (svodenje reči na osnovni oblik).

4. Vremenski podaci (Time series)

Vremenski podaci predstavljaju niz vrednosti koje su zabeležene tokom određenog vremenskog perioda. Primeri: dnevna temperatura, cena akcija kroz vreme, broj korisnika po satu.

Ovi podaci se obrađuju pomoću metoda kao što su:

- konverzija formata datuma i vremena,
- resamplovanje (promena vremenskog koraka, npr. sa minuta na sate),
- detekcija trendova i sezonalnosti.

Svaka od ovih vrsta podataka zahteva specifične pristupe preprocesiranju, a pravilno prepoznavanje tipa podataka je ključni korak za dobijanje tačnih i pouzdanih rezultata u analizi i modelovanju.

Uobičajeni problemi u podacima

Podaci koji se prikupljaju iz različitih izvora retko su savršeni i često sadrže greške, praznine ili nepravilnosti koje mogu značajno uticati na rezultate analize ili modela mašinskog učenja. Pre nego što se pristupi analizi, neophodno je identifikovati i ispraviti ove probleme kako bi se obezbedila tačnost i pouzdanost podataka.

Najčešći problemi u podacima su sledeći:

1. Nedostajuće vrednosti

Nedostajuće vrednosti predstavljaju situacije kada određeni podaci nisu zabeleženi ili su izgubljeni tokom prikupljanja. Na primer, u tabeli podataka o korisnicima može nedostajati godina rođenja ili visina.

Ovaj problem se može rešiti na više načina:

- **Uklanjanjem redova ili kolona** sa previše nedostajućih vrednosti,
- **Popunjavanjem (imputacijom)** - npr. srednjom vrednošću, medijanom, najčešćom vrednošću ili pomoću modela koji predviđa nedostajuće podatke.

2. Duplikati

Duplikati predstavljaju ponovljene zapise u skupu podataka, što može dovesti do pristrasnih rezultata. Na primer, ako se isti korisnik pojavi više puta u bazi, to može iskriviti statistiku o broju kupaca.

Rešenje je detekcija i uklanjanje duplikata pomoću identifikacionih brojeva ili kombinacije više kolona koje zajedno jedinstveno opisuju red.

3. Šum i ekstremne vrednosti (outlajeri)

Šum predstavlja nasumične ili pogrešne podatke koji ne odražavaju stvarno stanje, dok su outlajeri vrednosti koje značajno odstupaju od većine ostalih podataka. Primer: ako su sve temperature u nizu između 20°C i 25°C, a jedna vrednost iznosi 60°C, to je verovatno greška u unosu.

Takve vrednosti mogu pogrešno uticati na prosek, varijansu i rezultate modela. Za njihovo otkrivanje koriste se metode kao što su Z-score, IQR (Interquartile Range) ili vizuelne tehnike poput box plot dijagrama.

4. Nedosledni formati i jedinice

Podaci često dolaze iz različitih izvora, pa se iste informacije mogu nalaziti u različitim formatima ili jedinicama. Na primer, datum može biti zapisan kao “2025-10-18” ili “18/10/2025”, dok se težina može izražavati u kilogramima (kg) ili funtama (lb). Ako se ne usaglase formati i jedinice, može doći do pogrešne interpretacije.

Rešenje je standardizacija formata i konverzija jedinica kako bi svi podaci bili u istom sistemu i spremni za analizu.

Faze preprocesiranja podataka

Čišćenje podataka (Data Cleaning)

Čišćenje podataka predstavlja prvi i najvažniji korak u procesu preprocesiranja, jer se u ovoj fazi otklanjaju sve nepravilnosti, greške i nedoslednosti koje mogu negativno uticati na dalju analizu ili obuku modela. Cilj čišćenja je da se podaci učine tačnim, potpunim i konzistentnim, kako bi predstavljali realno stanje posmatranog fenomena.

Najčešće operacije u procesu čišćenja podataka uključuju uklanjanje duplikata, popunjavanje ili uklanjanje nedostajućih vrednosti i detekciju i obradu outlajera.

1. Uklanjanje duplikata

Duplikati su ponovljeni redovi u skupu podataka koji mogu nastati prilikom spajanja više izvora ili grešaka u unosu. Njihovo prisustvo dovodi do iskrivljenih statističkih pokazatelja i netačnih rezultata analize.

U praksi, duplikati se prepoznaju po istim vrednostima u ključnim kolonama (npr. ID korisnika, datum, naziv proizvoda).

2. Popunjavanje ili uklanjanje nedostajućih vrednosti

Nedostajuće vrednosti (missing values) su čest problem u realnim skupovima podataka i mogu se pojaviti zbog grešaka u unosu, tehničkih problema ili neodgovorenih pitanja u anketama.

Postoje dva osnovna pristupa rešavanju ovog problema:

- Uklanjanje redova ili kolona koji sadrže previše praznih vrednosti (ako je gubitak informacija mali).
- Popunjavanje vrednosti (imputacija) pomoću metoda kao što su:
    - srednja vrednost (mean) ili medijana (median) za numeričke kolone,
    - najčešća vrednost (mode) za kategoričke kolone,
    - regresioni modeli ili algoritmi poput K-Nearest Neighbors (KNN) za napredniju imputaciju.

Cilj je očuvati što više korisnih podataka, a istovremeno izbeći unošenje pristrasnosti u dataset.

3. Detekcija i obrada outlajera

Outlajeri (ekstremne vrednosti) su podaci koji značajno odstupaju od ostalih i mogu ukazivati na greške u merenju, unosu, ili na retke, ali važne pojave. Ako se ne prepoznaju, mogu snažno uticati na proseke, varijanse i performanse modela mašinskog učenja.

Za njihovu detekciju koriste se statističke metode kao što su:

- Z-score metoda - identifikuje vrednosti koje odstupaju više od zadatog broja standardnih devijacija od proseka,
- IQR metoda (Interquartile Range) - koristi kvartile i raspon da bi pronašla ekstremne vrednosti,
- Vizuelne metode poput box plot dijagrama.

U zavisnosti od konteksta, outlajeri se mogu ukloniti, zameniti srednjom vrednošću, ili posebno analizirati ako nose važnu informaciju (npr. u detekciji prevara).

Efikasno čišćenje podataka obezbeđuje da dataset bude pouzdan, konzistentan i spreman za preciznu analizu i modelovanje, čime se postavlja čvrst temelj za sve naredne faze preprocesiranja.

Transformacija podataka (Data Transformation)

Transformacija podataka je proces u kojem se podaci menjaju, prilagođavaju ili pretvaraju u oblik koji je pogodan za analizu i modelovanje. Ova faza omogućava da svi atributi imaju slične razmere, formate i tipove vrednosti, čime se poboljšava rad algoritama mašinskog učenja i povećava tačnost modela.

Najčešće korišćene metode transformacije uključuju normalizaciju i standardizaciju, log-transformaciju i skaliranje, kao i kodiranje Kategoričkih podataka.

Normalizacija i standardizacija

Ove tehnike se koriste kako bi se numeričke vrednosti svodile na uporedive razmere, jer algoritmi poput linearne regresije, KNN-a ili neuronskih mreža mogu biti osetljivi na različite opsege podataka.

- **Normalizacija (Min-Max Scaling)**

  Podaci se transformišu tako da vrednosti budu u intervalu od 0 do 1. Ova metoda je korisna kada su podaci ograničeni i kada ekstremne vrednosti nisu dominantne.

$$x' = \frac{x - x_{min}}{x_{max} - x_{min}}$$

- **Standardizacija (Z-score Scaling)**

Podaci se transformišu tako da imaju srednju vrednost 0 i standardnu devijaciju 1. Standardizacija je pogodna kada podaci imaju različite raspone i kada se pretpostavlja da prate približno normalnu distribuciju.

$$x' = \frac{x - \mu}{\sigma}$$

Log-transformacija i skaliranje

Log-transformacija se koristi kada su podaci asimetrični ili imaju velike razlike između minimalnih i maksimalnih vrednosti (npr. cene, prihodi, broj klikova). Primena logaritma smanjuje uticaj ekstremno velikih vrednosti i stabilizuje varijansu, čime se obezbeđuje ravnomernija distribucija podataka.

Skaliranje podrazumeva prevođenje podataka u određeni opseg ili oblik koji odgovara zahtevima algoritma. Pored normalizacije i standardizacije, često se koriste i druge metode skaliranja poput RobustScaler, koji je otporan na outlajere.

Kodiranje Kategoričkih podataka

Većina algoritama mašinskog učenja može da radi samo sa numeričkim vrednostima, pa je neophodno pretvoriti kategoričke (tekstualne) promenljive u brojčane. To se postiže pomoću različitih metoda kodiranja:

- **Label Encoding** - svaka kategorija dobija jedinstven broj (npr. “muško”: 0, “žensko”: 1). Pogodno za ordinalne promenljive gde redosled ima značenje.
- **One-Hot Encoding** - svaka kategorija se predstavlja novom binarnom kolonom (npr. “boja”: crvena [1,0,0], plava [0,1,0], zelena [0,0,1]). Ova metoda se koristi za nominalne promenljive bez redosleda.
- **Ordinal Encoding** - koristi unapred definisan redosled kategorija (npr. “mali” = 1, “srednji” = 2, “veliki” = 3).

Diskretizacija (Binning)

Diskretizacija, poznata i kao binning, predstavlja proces pretvaranja kontinuiranih numeričkih vrednosti u diskretne (kategoričke) intervale ili klase. Cilj ove metode je da se smanji varijabilnost podataka, poboljša interpretacija i omogući jednostavnije analiziranje obrazaca, posebno u modelima koji bolje funkcionišu sa Kategoričkim podacima.

Na primer, umesto da se atribut starost posmatra kao neprekidna promenljiva (npr. 18, 23, 37, 64), on se može podeliti u intervale (npr. “mladi”, “odrasli”, “stari”).

Diskretizacija se može sprovesti na više načina, u zavisnosti od toga kako se definišu granice intervala (binova):

1. **Jednaka širina intervala (Equal-width binning)**
   - Vrednosti se dele u više intervala iste širine.
   - Granice se određuju prema minimalnoj i maksimalnoj vrednosti atributa.
   - Na primer, ako su vrednosti age od 0 do 100, podela na 5 binova daje intervale: [0–20), [20–40), [40–60), [60–80), [80–100].
   - Koristi se kada su podaci ravnomerno raspoređeni.

2. **Jednak broj elemenata (Equal-frequency binning)**
   - Svaki interval sadrži približno isti broj uzoraka.
   - Pogodno je za podatke koji nisu ravnomerno raspoređeni jer se tako izbegava prevelika koncentracija vrednosti u jednom intervalu.

3. **Diskretizacija na osnovu domena (Custom binning)**
   - Granice se definišu ručno prema stručnom znanju ili logici problema.

4. **Diskretizacija pomoću algoritama (npr. Decision Tree Binning)**
   - Koristi se automatska podela na osnovu graničnih vrednosti koje minimizuju grešku modela.
   - Ova metoda se često koristi u kreditnom skorovanju i prediktivnoj analitici.

Smanjenje dimenzionalnosti (Dimensionality Reduction)

Smanjenje dimenzionalnosti predstavlja proces u kojem se veliki broj atributa (karakteristika) u skupu podataka svodi na manji broj reprezentativnih promenljivih, uz što manji gubitak informacija. Ovaj korak je posebno važan kod kompleksnih skupova podataka sa desetinama ili stotinama kolona, gde preveliki broj atributa može izazvati pretežak model, duže vreme obrade i pojavu pretreniranja (overfittinga).

Cilj smanjenja dimenzionalnosti je da se zadrže najvažnije informacije, eliminišu redundantni ili slabo informativni podaci i time poboljša efikasnost i tačnost modela.

PCA (Principal Component Analysis)

Analiza glavnih komponenti (PCA) je jedna od najpoznatijih i najčešće korišćenih tehnika za smanjenje dimenzionalnosti. Ona funkcioniše tako što:

- Identifikuje pravce u prostoru podataka (tzv. glavne komponente) duž kojih je varijansa podataka najveća.
- Projektuje originalne podatke na te pravce, čime se dobijaju nove, međusobno nezavisne promenljive koje sadrže većinu informacija iz originalnog skupa.

Na primer, skup sa 10 atributa može se pomoću PCA svesti na 2 ili 3 glavne komponente koje i dalje zadržavaju najveći deo varijanse (npr. 90-95%). PCA se često koristi u vizualizaciji podataka, smanjenju kompleksnosti modela i uklanjanju šuma.

Prednosti PCA metode su:

- smanjenje broja promenljivih bez većeg gubitka informacija,
- lakša vizualizacija višedimenzionalnih podataka,
- poboljšanje performansi modela i smanjenje rizika od pretreniranja.

Feature selection i eliminacija irelevantnih atributa

Pored transformacionih metoda poput PCA, smanjenje dimenzionalnosti se može postići i odabirom najvažnijih atributa. Ova tehnika podrazumeva identifikovanje i zadržavanje samo onih karakteristika koje najviše doprinose predikciji ciljne promenljive, dok se irelevantni ili redundantni atributi uklanjaju.

Postoje tri glavna pristupa za izbor atributa:

- **Filter metode** - koriste statističke mere da procene važnost atributa nezavisno od modela.
- **Wrapper metode** - testiraju različite kombinacije atributa i biraju skup koji daje najbolje rezultate modela (npr. Recursive Feature Elimination - RFE).

- **Embedded metode** - biraju značajne atribute tokom same obuke modela (npr. Lasso regresija, Random Forest feature importance).

Eliminacijom irelevantnih atributa postiže se:

- manja kompleksnost modela,
- kraće vreme obrade,
- bolja interpretacija rezultata,
- smanjenje rizika od overfittinga.

Integracija i agregacija podataka

Integracija i agregacija podataka predstavljaju završne korake u procesu preprocesiranja, kojima se različiti izvori informacija povezuju u jedinstven i smislen skup podataka spreman za analizu. Ovi postupci omogućavaju da se iz različitih tabela, baza ili formata dobije koherentan skup sa objedinjenim i sažetim informacijama.

Spajanje više izvora podataka

U realnim situacijama, podaci često potiču iz različitih izvora, kao što su baze podataka, Excel fajlovi, CSV datoteke, senzori, API-ji ili web servisi. Ti podaci se obično međusobno dopunjuju, pa ih je potrebno integrisati (spojiti) u jednu celinu.

Integracija podataka podrazumeva spajanje više tabela na osnovu zajedničkih ključeva (npr. ID korisnika, broj porudžbine, datum), kako bi se dobio potpuniji pogled na analizirani fenomen.

U praksi se koriste sledeći tipovi spajanja:

- **Inner join** - zadržava samo podatke koji postoje u svim tabelama.
- **Left/Right join** - zadržava sve podatke iz jedne tabele, uz dopunu vrednostima iz druge ako postoje.
- **Outer join** - kombinuje sve podatke iz svih tabela, popunjavajući praznine tamo gde nema poklapanja.

Primer: kombinovanje baze kupaca i baze transakcija omogućava analizu potrošačkih navika, učestalosti kupovine i prosečne vrednosti porudžbina.

Integracija podataka obezbeđuje celovit uvid u poslovanje ili analizirani sistem, smanjuje redundanciju i omogućava bogatije analize.

Grupisanje i sumiranje vrednosti

Agregacija podataka podrazumeva grupisanje i sažimanje informacija radi boljeg razumevanja trendova, obrazaca i odnosa između podataka. Na primer, umesto da se analizira svaka pojedinačna transakcija, može se posmatrati ukupan promet po danu, prosečna vrednost po korisniku ili broj kupovina po regionu. Agregacija omogućava da se veliki i složeni skupovi podataka pretvore u jednostavne i razumljive pokazatelje, što je izuzetno važno za donošenje odluka, izveštavanje i vizualizaciju rezultata.

Detekcija i obrada anomalija

Detekcija i obrada anomalija predstavlja važan deo preprocesiranja podataka jer omogućava prepoznavanje neuobičajenih, nepravilnih ili neočekivanih vrednosti koje odstupaju od uobičajenog obrasca ponašanja podataka. Takve vrednosti mogu biti rezultat grešaka u merenju, tehničkih problema, ljudskih grešaka pri unosu, ali i indikatori važnih događaja, kao što su prevare u finansijskim transakcijama, kvarovi u industriji ili neuobičajene aktivnosti u mrežnom saobraćaju. Cilj ove faze nije uvek jednostavno "uklanjanje" anomalija, već njihovo prepoznavanje, razumevanje i pravilno tretiranje u zavisnosti od konteksta.

Vrste anomalija

Anomalije se obično dele na tri glavne grupe:

- **Globalne anomalije (point anomalies)** - pojedinačne vrednosti koje značajno odstupaju od celokupne distribucije podataka.
  Primer: Transakcija od 1.000.000 dinara u setu gde su prosečne vrednosti između 1.000 i 10.000 dinara.
- **Kontekstualne anomalije (contextual anomalies)** - vrednosti koje su neuobičajene u određenom kontekstu, ali ne nužno u celini.
  Primer: Temperatura od 30°C može biti normalna leti, ali je anomalija zimi.
- **Kolektivne anomalije (collective anomalies)** - skup podataka koji zajedno odstupaju od očekivanog obrasca.
  Primer: Neočekivano veliki broj neuspelih logovanja u kratkom vremenskom periodu.

Metode detekcije anomalija

Za prepoznavanje anomalija koriste se različite statističke i mašinske metode, u zavisnosti od tipa i kompleksnosti podataka:

- **Statističke metode**
  Koriste se kada su podaci raspoređeni prema poznatoj distribuciji (npr.

normalnoj). Anomalije se identifikuju pomoću Z-score ili Interquartile Range (IQR) metoda, gde se vrednosti koje su izvan određenog opsega smatraju sumnjivima.

- **Mašinsko učenje (unsupervised methods)**
  Kada nema unapred definisanih oznaka za “normalno” i “anomalno”, koriste se algoritmi koji uče obrasce u podacima, poput:
  - Isolation Forest - algoritam koji nasumično deli podatke i identifikuje vrednosti koje se izdvajaju jer se lako “izoluju”.
  - DBSCAN (Density-Based Spatial Clustering of Applications with Noise) - otkriva grupe (klastere) i vrednosti koje ne pripadaju nijednoj grupi.
  - Local Outlier Factor (LOF) - meri koliko je tačka udaljena od svojih suseda i označava izolovane kao anomalne.

Obrada anomalija

Nakon detekcije, potrebno je odlučiti kako tretirati anomalije, u zavisnosti od svrhe analize:

- Uklanjanje - ako su rezultat očigledne greške (npr. negativna starost osobe).
- Zamenjivanje (imputacija) - ako je vrednost važna, ali pogrešno zabeležena (npr. ispravka decimalne tačke).
- Zadržavanje - ako anomalije nose korisne informacije, npr. u analizi prevara ili detekciji kvarova.

Alati i biblioteke za preprocesiranje

U savremenoj analizi podataka, preprocesiranje se retko izvodi ručno. Danas postoje brojni programski alati i biblioteke koji omogućavaju brzo, tačno i efikasno sprovođenje svih potrebnih koraka pripreme podataka. Ovi alati olakšavaju čišćenje, transformaciju, kodiranje i skaliranje podataka, čime se značajno smanjuje vreme potrebno za pripremu kvalitetnog skupa podataka.

Razvoj specijalizovanih biblioteka za preprocesiranje podataka revolucionarizovao je oblast data science-a. Umesto da analitičari pišu stotine linija koda za osnovne operacije, danas mogu da koriste proverene, optimizovane i dobro dokumentovane funkcije koje pokrivaju gotovo sve scenarije preprocesiranja. Ove biblioteke nisu samo vremenski efikasne, već su one i dizajnirane da rade sa velikim količinama podataka, optimizovane

su za performanse i redovno se održavaju i unapređuju od strane aktivne zajednice developera.

Pandas

Biblioteka pandas je najčešće korišćeni alat za rad sa tabelarnim podacima u Pythonu. Njen naziv potiče od termina "panel data" koji se koristi u ekonometriji i statistici. Pandas je dizajniran da omogući intuitivnu i efikasnu manipulaciju strukturiranim podacima kroz dva glavna objekta: DataFrame (dvodimenzionalna tabela) i Series (jednodimenzionalni niz).

Ključne prednosti pandas biblioteke su:

- **Fleksibilnost u radu sa različitim formatima podataka** - pandas može da učita podatke iz CSV, Excel, JSON, SQL baza podataka, HTML tabela i mnogih drugih formata. Ova univerzalnost omogućava analitičarima da brzo integrišu podatke iz različitih izvora.
- **Intuitivna sintaksa** - pandas koristi sintaksu inspirisanu SQL-om i R-om, što olakšava transiciju analitičarima koji dolaze iz drugih okruženja. Operacije poput filtriranja, grupisanja i agregacije se izvršavaju prirodnim, čitljivim kodom.
- **Efikasno rukovanje nedostajućim vrednostima** - pandas ima ugrađenu podršku za označavanje i rukovanje sa nedostajućim podacima (NaN vrednosti), što je kritično za preprocesiranje realnih podataka koji retko dolaze u kompletnom obliku.

Najčešće korišćene funkcije za preprocesiranje su:

- **drop_duplicates()** - automatski identifikuje i uklanja redove koji se ponavljaju u skupu podataka. Može se primeniti na ceo DataFrame ili samo na određene kolone, što je korisno kada želimo da identifikujemo duplikate samo po ključnim atributima.
- **fillna()** - popunjava nedostajuće vrednosti prema različitim strategijama: konstantnom vrednošću, srednjom vrednošću, medijanom, najčešćom vrednošću ili metodom forward/backward fill koja koristi susedne vrednosti. Ova funkcija omogućava i napredne strategije kao što je popunjavanje različitim vrednostima za različite kolone istovremeno.
- **groupby()** - omogućava grupisanje podataka po jednoj ili više kolona i primenu agregacionih funkcija (zbir, prosek, broj elemenata) na svaku grupu. Ova funkcija je osnova za kreiranje novih atributa i razumevanje strukture podataka kroz stratifikaciju.
- **merge() i concat()** - spajaju različite DataFrame-ove vertikalno ili horizontalno. Merge funkcija omogućava različite tipove JOIN operacija (inner, outer, left, right) slične onima u SQL-u, što je neophodno kada se podaci nalaze u više tabela.

Dodatno, pandas pruža moćne funkcije kao što su pivot_table() za kreiranje pivot tabela, apply() za primenu custom funkcija na kolone ili redove, i str accessor za napredne operacije nad tekstualnim podacima.

Numpy

Biblioteka NumPy (Numerical Python) pruža osnovu za naučne proračune u Pythonu i predstavlja temelj na kom su izgrađene gotovo sve ostale biblioteke za analizu podataka. NumPy uvodi koncept ndarray (n-dimensional array), tj. homogeni multidimenzionalni niz koji omogućava vektorske operacije bez potrebe za eksplicitnim petljama.

Ključne karakteristike NumPy biblioteke:

- **Brzina izvršavanja** - NumPy operacije su implementirane u C jeziku, što ih čini do 100 puta bržim od ekvivalentnog Python koda sa listama. Ova brzina je kritična kada se radi sa velikim skupovima podataka koji mogu imati milione redova.
- **Vektorske operacije** - umesto pisanja petlji za primenu operacija na svaki element niza, NumPy omogućava primenu operacija na cele nizove odjednom. Na primer, `array * 2` automatski množi svaki element niza sa 2.
- **Broadcasting** - NumPy automatski prilagođava dimenzije nizova različitih veličina kako bi omogućio operacije između njih, što pojednostavljuje kod i povećava čitljivost.

NumPy se koristi za:

- **Matematičke transformacije** - primenu logaritamskih, eksponencijalnih, trigonometrijskih i drugih matematičkih funkcija na cele kolone podataka.
- **Normalizaciju i standardizaciju** - brzo izračunavanje proseka, standardne devijacije, minimuma, maksimuma i primenu formula za skaliranje podataka.
- **Rad sa matricama** - transponovanje, množenje matrica, računanje inverznih matrica i determinanti, što je osnova za mnoge algoritme mašinskog učenja.
- **Generisanje pseudo-slučajnih brojeva** - kreiranje kontrolisanih random vrednosti za testiranje algoritama, inicijalizaciju težina neuronskih mreža ili kreiranje sintetičkih podataka.
- **Logičke operacije i indeksiranje** - brzo filtriranje podataka na osnovu složenih logičkih uslova, kreiranje boolean maski, i indeksiranje višedimenzionalnih nizova.

Scikit-learn

Biblioteka scikit-learn (često skraćeno kao sklearn) je najkompletniji alat za mašinsko učenje u Python ekosistemu. Iako je prvenstveno poznata po algoritmima mašinskog učenja, scikit-learn sadrži izuzetno razvijen i obiman skup funkcija posebno dizajniranih za preprocesiranje podataka.

Filozofija scikit-learn biblioteke zasniva se na konceptu transformera, tj. objekata koji uče određenu transformaciju iz podataka (metodom fit()) a zatim tu transformaciju primenjuju na nove podatke (metodom transform()). Ovaj pristup omogućava konzistentnu primenu istih transformacija na trening i test skupove, što je kritično za izbegavanje data leakage problema. Najvažniji moduli za preprocesiranje:

sklearn.preprocessing - sadrži najširu paletu alata za skaliranje i transformaciju podataka:

- **StandardScaler** - standardizuje podatke tako da imaju srednju vrednost 0 i standardnu devijaciju 1. Ovo je najčešća metoda skaliranja za algoritme koji pretpostavljaju normalno distribuirane podatke.
- **MinMaxScaler** - skalira podatke u određeni opseg, obično [0,1]. Koristan kada algoritam zahteva da sve vrednosti budu u fiksnom opsegu.
- **RobustScaler** - otporan je na outliere jer koristi medijanu i interkvartilni opseg umesto srednje vrednosti i standardne devijacije.
- **LabelEncoder** i **OneHotEncoder** - kodiraju kategoričke varijable u numerički format. OneHotEncoder kreira binarne kolone za svaku kategoriju, dok LabelEncoder dodeljuje celobrojne vrednosti.
- **PolynomialFeatures** - generiše polinomske i interakcione atribute, što može poboljšati performanse modela kada postoje nelinearne veze između atributa.

sklearn.impute - specijalizovan modul za rukovanje nedostajućim vrednostima:

- **SimpleImputer** - popunjava nedostajuće vrednosti korišćenjem osnovnih strategija: srednja vrednost, medijana, najčešća vrednost ili konstanta.
- **KNNImputer** - koristi K najbližih suseda za procenu nedostajućih vrednosti, uzimajući u obzir vrednosti sličnih primera u skupu podataka.
- **IterativeImputer** - implementira MICE (Multivariate Imputation by Chained Equations) pristup, gde se svaki atribut sa nedostajućim vrednostima modeluje kao funkcija ostalih atributa.

sklearn.decomposition - implementira metode za smanjenje dimenzionalnosti:

- **PCA** (Principal Component Analysis) - pronalazi glavne komponente koje objašnjavaju najviše varijanse u podacima i omogućava projekciju podataka u prostor manjih dimenzija.
- **TruncatedSVD** - varijanta PCA koja je efikasnija za retke matrice (sparse matrices).
- **FactorAnalysis** i **FastICA** - druge metode za otkrivanje latentnih faktora u podacima.

sklearn.feature_selection - pruža algoritme za automatski odabir najrelevantnijih atributa:

- **SelectKBest** - bira K najboljih atributa na osnovu statističkih testova.
- **RFE** (Recursive Feature Elimination) - iterativno uklanja najmanje važne atribute.
- **VarianceThreshold** - eliminiše atribute sa malom varijansom koji ne doprinose raznovrsnosti podataka.

Dodatna prednost scikit-learn biblioteke je Pipeline klasa koja omogućava organizovanje više koraka preprocesiranja i modelovanja u jedan objekat. To znači da se svi koraci (od skaliranja preko transformacije do finalne predikcije) mogu izvršiti pozivom jedne metode, što garantuje konzistentnost i smanjuje mogućnost grešaka.

Komplementarnost ovih biblioteka

Ove tri biblioteke se retko koriste izolovano. U tipičnom projektu analitike podataka:

1. pandas se koristi za učitavanje podataka, inicijalno čišćenje, eksplorativnu analizu i kreiranje novih atributa
2. numpy se koristi za matematičke transformacije i operacije na nizovima unutar pandas DataFrame-ova
3. scikit-learn se koristi za standardizovane transformacije koje će se primeniti i na test skupu, kao i za kreiranje pipeline-a koji objedinjava preprocesiranje i modelovanje

Zajedno, ove biblioteke formiraju ekosistem koji pokriva kompletan životni ciklus preprocesiranja podataka – od sirovog učitavanja do finalno optimizovanog skupa spremnog za treniranje modela mašinskog učenja. Njihova dobra integracija, sveobuhvatna dokumentacija i aktivna zajednica korisnika čine ih nezamenjivim alatima u modernoj analizi podataka.

Zaključak

Preprocesiranje podataka predstavlja temelj svakog procesa analize i mašinskog učenja, jer od kvaliteta podataka direktno zavisi i kvalitet dobijenih rezultata. Sirovi podaci koje prikupljamo iz različitih izvora često su nekompletni, bučni i nedosledni, pa se pre bilo kakve analize moraju očistiti, transformisati i pripremiti.

Kroz različite faze preprocesiranja - čišćenje, transformaciju, integraciju, smanjenje dimenzionalnosti i diskretizaciju - podaci se dovode u stanje pogodno za obradu i modelovanje. Time se postiže veća tačnost modela, brža obrada, smanjenje šuma i bolja interpretacija rezultata. Poseban značaj imaju tehnike kao što su normalizacija, kodiranje Kategoričkih vrednosti, imputacija nedostajućih podataka, kao i smanjenje dimenzionalnosti koje eliminiše redundantne informacije i povećava efikasnost modela.

Upotreba alata i biblioteka kao što su pandas, numpy, scikit-learn omogućava da se ovi procesi sprovode tačno i ponovljivo, čime se olakšava rad i štedi vreme.

Zaključno, može se reći da preprocesiranje nije samo tehnička faza, već ključni korak u stvaranju vrednosti iz podataka. Ono obezbeđuje da analize budu zasnovane na pouzdanim, reprezentativnim i kvalitetno pripremljenim podacima, čime se postavlja čvrst osnov za donošenje ispravnih odluka, razvoj prediktivnih modela i uspešnu primenu data science tehnika u praksi.

Literatura

1. Han, J., Kamber, M., & Pei, J. (2012). *Data Mining: Concepts and Techniques* (3rd ed.). Morgan Kaufmann Publishers.
2. Géron, A. (2023). *Hands-On Machine Learning with Scikit-Learn, Keras & TensorFlow* (3rd ed.). O'Reilly Media.
3. Wirth, R., & Hipp, J. (2000). *CRISP-DM: Towards a Standard Process Model for Data Mining*. Proceedings of the 4th International Conference on the Practical Applications of Knowledge Discovery and Data Mining, 29–39.
4. Larose, D. T., & Larose, C. D. (2014). *Discovering Knowledge in Data: An Introduction to Data Mining* (2nd ed.). Wiley-Interscience.
5. Aggarwal, C. C. (2015). *Data Mining: The Textbook*. Springer.
6. Alpaydin, E. (2021). *Introduction to Machine Learning* (4th ed.). MIT Press.
7. Kuhn, M., & Johnson, K. (2019). *Feature Engineering and Selection: A Practical Approach for Predictive Models*. Chapman and Hall/CRC.
8. Raschka, S., & Mirjalili, V. (2022). *Python Machine Learning* (4th ed.). Packt Publishing.
9. Scikit-learn Developers. (2024). *Scikit-learn: Machine Learning in Python*. Preuzeto sa: https://scikit-learn.org/stable/
10. McKinney, W. (2017). *Python for Data Analysis: Data Wrangling with Pandas, NumPy, and IPython* (2nd ed.). O'Reilly Media.