Laboratorijska vježba 5 - Nelinearna jednostavna regresija i višestruka regresija

Metode statističke analize podataka

2025./2026.

1 Cilj vježbe

Cilj ove laboratorijske vježbe je upoznati se s:

- nelinearnom jednostavnom regresijom na primjeru **logističke regresije** (binarni ishod),
- **višedimenzionalnom linearnom regresijom** (više regresora),
- osnovnim pretpostavkama, interpretacijom koeficijenata i osnovnim dijagnostičkim mjerama modela.

Studenti nakon vježbe trebaju:

- znati prepoznati situacije kada linearna regresija nije primjeren model za zavisnu varijablu,
- razumjeti zašto koristimo logističku funkciju (sigmoid) za modeliranje vjerojatnosti,
- moći zapisati i interpretirati model višestruke linearne regresije,
- poznavati ulogu koeficijenta determinacije R^2, prilagođenog R^2 i F-testa,
- biti svjesni problema multikolinearnosti i pojma VIF (Variance Inflation Factor).

2 Nelinearna jednostavna regresija: logistička regresija

2.1 Motivacija

Kod klasične jednostavne linearne regresije pretpostavljamo model oblika

$$Y = \beta_0 + \beta_1 X + \varepsilon,$$

gdje je Y kvantitativna (numerička) zavisna varijabla.

Međutim, u mnogim primjenama zavisna varijabla je **binarna**, npr.:

- uspjeh / neuspjeh (1 / 0),
- kupio proizvod / nije kupio,
- bolestan / zdrav,
- klik na oglas / nema klika.

U takvim slučajevima:

- Y poprima samo vrijednosti 0 ili 1,
- zanima nas **vjerojatnost** da se događaj dogodi, tj. $P(Y = 1 \mid X)$,
- linearnim modelom $\beta_0 + \beta_1 X$ ne možemo direktno modelirati vjerojatnost, jer vrijednosti mogu biti manje od 0 ili veće od 1.

Zato uvodimo **logističku regresiju** kao nelinearni model: transformiramo linearni izraz

$$z = \beta_0 + \beta_1 X$$

logističkom (sigmoidnom) funkcijom u interval $(0, 1)$.

2.2 Logistička funkcija i definicija modela

Logistička funkcija (sigmoid) ili logaritam omjera šansi (engl. *log-odds*) definirana je kao

$$\sigma(z) = \frac{1}{1 + e^{-z}}.$$

Za svaku realnu vrijednost $z \in \mathbb{R}$ vrijedi:

$$0 < \sigma(z) < 1, \quad \lim_{z \to -\infty} \sigma(z) = 0, \quad \lim_{z \to +\infty} \sigma(z) = 1.$$

Model logističke regresije za binarnu zavisnu varijablu $Y \in \{0, 1\}$ i jednu nezavisnu varijablu X zapisujemo kao

$$P(Y = 1 \mid X) = \pi(X) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 X)}}. \tag{1}$$

Gdje su:

- $P(Y = 1 \mid X)$ – vjerojatnost da se događa “pozitivni” ishod (npr. uspjeh, bolest prisutna, kupnja proizvoda),
- β_0 – presjek (intercept),
- β_1 – koeficijent koji opisuje utjecaj X na logit vjerojatnosti.

2.3 Logit, izgledi i linearizacija modela

Umjesto da izravno modeliramo vjerojatnost $\pi(X)$, često radimo s tzv. **izgledima** (odds):

$$\text{odds}(X) = \frac{\pi(X)}{1-\pi(X)}.$$

Ako u logistički model uvrstimo definiciju $\pi(X)$ (Eq. 1), dobivamo:

$$\frac{\pi(X)}{1-\pi(X)} = e^{\beta_0+\beta_1 X}.$$

Logit funkcija je logaritam izgleda:

$$\text{logit}(\pi(X)) = \log\left(\frac{\pi(X)}{1-\pi(X)}\right).$$

U logističkoj regresiji logit je **linearna funkcija** od X:

$$\log\left(\frac{\pi(X)}{1-\pi(X)}\right) = \beta_0 + \beta_1 X.$$

Ovim smo postigli:

- na razini vjerojatnosti model je nelinearan (sigmoid),
- na razini logita model je **linearan** u parametrima β_0, β_1.

2.4 Interpretacija parametara u logističkoj regresiji

Presjek β_0. Kada je $X = 0$, logit vjerojatnosti je jednak β_0:

$$\log\left(\frac{\pi(0)}{1-\pi(0)}\right) = \beta_0.$$

To znači da β_0 predstavlja log-izglede za ishod $Y = 1$ kada je $X = 0$. U praksi $X = 0$ ponekad nema smisla (npr. dob = 0 godina), ali je nužan za određivanje položaja sigmoidne krivulje.

Koeficijent β_1. Koeficijent β_1 opisuje promjenu log-izgleda pri promjeni X za jednu jedinicu:

$$\log\left(\frac{\pi(X+1)}{1-\pi(X+1)}\right) - \log\left(\frac{\pi(X)}{1-\pi(X)}\right) = \beta_1.$$

Primjenom eksponencijalne funkcije na obje strane jednadžbe, dobivamo omjer:

$$\frac{\text{odds}(X+1)}{\text{odds}(X)} = e^{\beta_1}. \tag{2}$$

Dakle:

- ako je $\beta_1 > 0$, tada $e^{\beta_1} > 1$ i izgledi za $Y = 1$ rastu s povećanjem X,
- ako je $\beta_1 < 0$, tada $e^{\beta_1} < 1$ i izgledi za $Y = 1$ padaju s povećanjem X,
- ako je $\beta_1 = 0$, tada $e^{\beta_1} = 1$ i X nema utjecaja na vjerojatnost ishoda.

2.5 Procjena parametara: metoda maksimalne vjerodostojnosti

Za razliku od linearne regresije (gdje se parametri obično procjenjuju metodom najmanjih kvadrata), u logističkoj regresiji koristimo **metodu maksimalne vjerodostojnosti** (engl. Maximum Likelihood Estimation, MLE).

Ideja:

- pretpostavimo da su opažanja (x_i, y_i) nezavisna,
- za svaki par vrijedi

$$P(Y_i = 1 \mid X_i) = \pi_i, \quad P(Y_i = 0 \mid X_i) = 1 - \pi_i,$$

  gdje je

$$\pi_i = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x_i)}}.$$

- funkcija vjerodostojnosti je umnožak vjerojatnosti za sve opažene ishode:

$$L(\beta_0, \beta_1) = \prod_{i=1}^{n} \pi_i^{y_i} (1 - \pi_i)^{1 - y_i},$$

- tražimo vrijednosti β_0, β_1 koje **maksimiziraju** $L(\beta_0, \beta_1)$, odnosno log-vjerodostojnost

$$\ell(\beta_0, \beta_1) = \log L(\beta_0, \beta_1).$$

Analitičko zatvoreno rješenje u pravilu ne postoji, pa se u računalnim paketima koriste **numeričke optimizacijske metode** (npr. Newton–Raphson, iterativna regrupacija težina i sl.).

2.6 Zašto ne koristiti običnu linearnu regresiju za binarnu Y?

Ako bismo za binarni ishod pokušali koristiti model

$$Y = \beta_0 + \beta_1 X + \varepsilon,$$

dobili bismo sljedeće probleme:

- predviđene vrijednosti $\hat{Y}$ ne bi bile ograničene na interval $[0, 1]$, nego bi mogle biti npr. $-0{,}3$ ili $1{,}4$,
- varijanca pogreške nije konstantna (varijanca Bernoullijeve varijable ovisi o π),
- pretpostavke linearne regresije (normalnost reziduala, homoskedastičnost) teško su zadovoljene.

Logistička regresija:

- daje predviđene vjerojatnosti u intervalu $(0, 1)$,
- bolje odražava prirodu binarne zavisne varijable,
- ima jasnu interpretaciju u terminima izgleda (odds) i logita.

3 Numerički primjer logističke regresije

Razmotrimo jednostavan primjer u kojem želimo modelirati vjerojatnost da će student položiti ispit ($Y = 1$) na temelju vremena provedenog u učenju (X). Pretpostavimo da su opažanja sljedeća:

Sati učenja X	**Ishod** Y
1	0
2	0
3	0
4	1
5	1
6	1

Želimo procijeniti logistički model:

$$P(Y = 1 \mid X) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 X)}}.$$

Numeričko rješenje (pojednostavljeno)

Pretpostavimo da metoda maksimalne vjerodostojnosti daje sljedeće procjene:

$$\hat{\beta}_0 = -6.5, \qquad \hat{\beta}_1 = 1.3.$$

Model postaje:

$$\hat{P}(Y = 1 \mid X) = \frac{1}{1 + e^{-(-6.5 + 1.3X)}}.$$

Izračunajmo procijenjene vjerojatnosti za nekoliko vrijednosti:

$$\hat{P}(Y = 1 \mid X = 3) = \frac{1}{1 + e^{-(-6.5+3.9)}} = \frac{1}{1 + e^{2.6}} \approx 0.069.$$

$$\hat{P}(Y = 1 \mid X = 5) = \frac{1}{1 + e^{-(-6.5+6.5)}} = \frac{1}{1 + e^{0}} = 0.5.$$

$$\hat{P}(Y = 1 \mid X = 6) = \frac{1}{1 + e^{-(-6.5+7.8)}} = \frac{1}{1 + e^{-1.3}} \approx 0.785.$$

Interpretacija

- Nakon oko 5 sati učenja vjerojatnost prolaza ispita iznosi oko 0.5.
- Za svaki dodatni sat učenja, log-izgledi se povećavaju za $\hat{\beta}_1 = 1.3$.
- Budući da je $e^{1.3} \approx 3.67$, izgledi prolaska *povećavaju se 3.67 puta* za svaki dodatni sat učenja.

4 Primjer u Pythonu

U nastavku slijedi primjer prilagodbe logističkog modela u Pythonu koristeći paket `statsmodels`.

```
import numpy as np
import pandas as pd
import statsmodels.api as sm

# Podaci
X = np.array([1,2,3,4,5,6])
y = np.array([0,0,0,1,1,1])

df = pd.DataFrame({"Hours": X, "Pass": y})

# Dodati konstantu
X_sm = sm.add_constant(df["Hours"])

# Logisticki model
model = sm.Logit(df["Pass"], X_sm).fit()
print(model.summary())

# Predvidjanje vjerojatnosti
x_new = pd.DataFrame({"const": 1, "Hours": [3, 5, 6]})
pred_probs = model.predict(x_new)

print("Predviđene vjerojatnosti:")
print(pred_probs)
```

Očekivani ispis

Model će procijeniti koeficijente β_0 i β_1 te dati:

- procijenjene parametre,
- standardne pogreške,
- z-statistike i p-vrijednosti,
- pseudo R^2,
- predviđene vjerojatnosti za nove podatke.

Tipični rezultat predviđanja

Primjer ispisa:

```
X = 3 → P(Y=1) = 0.07
X = 5 → P(Y=1) = 0.50
X = 6 → P(Y=1) = 0.79
```

Što se poklapa s našim numeričkim primjerom.

5 Višedimenzionalna linearna regresija

5.1 Motivacija i definicija modela

U jednostavnoj linearnoj regresiji imamo:

$$Y = \beta_0 + \beta_1 X + \varepsilon.$$

U praksi, Y vrlo često ovisi o više od jedne nezavisne varijable. Primjeri:

- potrošnja goriva vozila ovisi o masi, snazi motora, vrsti prijenosa, zapremini motora,
- jačina povlačenja žice ovisi o duljini žice i visini čipa,
- vrijeme izvršavanja algoritma o veličini ulaza i broju dretvi (threadova).

Tada modeliramo:

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_k X_k + \varepsilon.$$

Gdje je:

- Y – zavisna (odgovorna) varijabla,
- $X_1, \ldots, X_k$ – nezavisne (prediktorske) varijable,
- β_0 – presjek (intercept),
- β_j – koeficijent povezan s varijablom X_j,
- ε – slučajna pogreška (neobjašnjena varijabilnost).

5.2 Matrični zapis modela

Za n opažanja možemo model zapisati u matričnom obliku:

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon},$$

gdje je:

- $\mathbf{y}$ vektor dimenzije $n \times 1$ (opažanja Y_i),
- $\mathbf{X}$ matrica dimenzije $n \times p$ ($p = k + 1$), gdje prvi stupac sadrži jedinice (za β_0), a preostali stupci vrijednosti regresora $X_1, \ldots, X_k$:

$$\mathbf{X} = \begin{bmatrix} 1 & x_{11} & x_{12} & \ldots & x_{1k} \\ 1 & x_{21} & x_{22} & \ldots & x_{2k} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n1} & x_{n2} & \ldots & x_{nk} \end{bmatrix},$$

- $\boldsymbol{\beta}$ vektor parametara dimenzije $p \times 1$:

$$\boldsymbol{\beta} = \begin{bmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_k \end{bmatrix},$$

- $\boldsymbol{\varepsilon}$ vektor pogrešaka dimenzije $n \times 1$.

5.3 Pretpostavke modela višestruke linearne regresije

Standardne pretpostavke su:

1. **Linearnost**: očekivana vrijednost $E(Y \mid X_1, \ldots, X_k)$ linearna je funkcija regresora:
$$E(Y \mid X_1, \ldots, X_k) = \beta_0 + \beta_1 X_1 + \cdots + \beta_k X_k.$$
2. **Nezavisnost pogrešaka**: pogreške ε_i su međusobno nezavisne.
3. **Jednakost varijanci (homoskedastičnost)**: $Var(\varepsilon_i) = \sigma^2$ za sve i.
4. **Normalnost pogrešaka**: $\varepsilon_i \sim N(0, \sigma^2)$.
5. **Nema jake multikolinearnosti**: regresori nisu gotovo linearno zavisni (npr. nema gotovo savršene korelacije između X_1 i X_2).

Te pretpostavke se u praksi provjeravaju analizom reziduala, grafovima i dodatnim testovima.

5.4 Procjena parametara metodom najmanjih kvadrata

Procjenitelj metode najmanjih kvadrata minimizira sumu kvadrata reziduala:

$$S(\boldsymbol{\beta}) = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 = \sum_{i=1}^{n} (y_i - \beta_0 - \beta_1 x_{i1} - \cdots - \beta_k x_{ik})^2.$$

U matričnom obliku:

$$S(\boldsymbol{\beta}) = (\mathbf{y} - \mathbf{X}\boldsymbol{\beta})^T (\mathbf{y} - \mathbf{X}\boldsymbol{\beta}).$$

Minimizacijom po $\boldsymbol{\beta}$ dobivamo normalne jednadžbe, a rješenje (ako $\mathbf{X}^T\mathbf{X}$ ima inverz):

$$\hat{\boldsymbol{\beta}} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}.$$

Procjene $\hat{\beta}_j$ zatim interpretiramo kao:

- $\hat{\beta}_0$ – očekivana vrijednost Y kada su svi regresori jednaki nuli (često samo tehnički parametar),
- $\hat{\beta}_j$ za $j \geq 1$ – promjena u očekivanoj vrijednosti Y kada X_j poraste za jednu jedinicu, uz **fiksne** ostale regresore.

5.5 Dekompozicija sume kvadrata i koeficijent determinacije

Ukupnu varijabilnost Y možemo rastaviti na:

$$SST = SSR + SSE,$$

gdje je:

- SST – totalna suma kvadrata (total sum of squares),
- SSR – suma kvadrata regresije (sum of squares due to regression),
- SSE – suma kvadrata pogreške (sum of squares for error / residual).

Koeficijent determinacije je definiran kao:

$$R^2 = \frac{SSR}{SST} = 1 - \frac{SSE}{SST}.$$

On mjeri koji dio varijabilnosti zavisne varijable Y je objašnjen modelom (regresorima):

- R^2 je u intervalu $[0, 1]$,
- veća vrijednost znači da model bolje prilagođava podatke,
- međutim, R^2 **uvijek raste** dodavanjem novih regresora, čak i ako oni nisu stvarno važni.

Zato koristimo i **prilagođeni koeficijent determinacije**:

$$R^2_{\text{adj}} = 1 - \frac{SSE/(n-p)}{SST/(n-1)},$$

gdje je p broj parametara (uključujući presjek). Prilagođeni R^2 *kažnjava* dodavanje irelevantnih regresora.

5.6 Testiranje značajnosti modela: F-test

Želimo testirati hipotezu:

$$H_0 : \beta_1 = \beta_2 = \cdots = \beta_k = 0$$

(odnosno model nema objašnjavačku moć, jedini parametar je presjek) protiv

$$H_1 : \text{barem jedan od } \beta_j \text{ je različit od nule.}$$

Test statistika:

$$F_0 = \frac{SSR/k}{SSE/(n-p)}.$$

Ako su zadovoljene pretpostavke modela, F_0 slijedi F-razdiobu s k i $n-p$ stupnjeva slobode pod H_0.

- Ako je p-vrijednost $\leq \alpha$ (npr. $\alpha = 0{,}05$), odbacujemo H_0 i zaključujemo da model kao cjelina ima objašnjavačku moć.
- Ako je p-vrijednost $> \alpha$, nemamo dovoljno dokaza da model objašnjava Y bolje od modela samo s presjekom.

5.7 Testiranje pojedinačnih koeficijenata: t-test

Za svaki koeficijent β_j (obično $j = 1, \ldots, k$) testiramo:

$$H_0 : \beta_j = 0, \quad H_1 : \beta_j \neq 0.$$

Test statistika ima oblik:

$$t_0 = \frac{\hat{\beta}_j}{se(\hat{\beta}_j)},$$

gdje je $se(\hat{\beta}_j)$ procijenjena standardna pogreška koeficijenta $\hat{\beta}_j$.

Pod H_0, uz pretpostavke modela, t_0 slijedi t-razdiobu s $n - p$ stupnjeva slobode. Na temelju p-vrijednosti zaključujemo je li pojedini regresor statistički značajan uz ostale u modelu.

5.8 Dva načina implementacije višestruke linearne regresije u Pythonu

Višestruka linearna regresija matematički se zapisuje kao:

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_3 + \varepsilon,$$

što se u statističkoj notaciji često prikazuje putem formule:

$$Y \sim X_1 + X_2 + X_3.$$

U Pythonu (biblioteka `statsmodels`) postoje **dva standardna načina** izgradnje takvog modela.

5.8.1 Formula API ($Y \sim X1 + X2 + X3$)

Ovaj pristup automatski:

- kreira matricu prediktora X,
- dodaje konstantu,
- prepoznaje dummy varijable prema nazivima stupaca.

```
import statsmodels.formula.api as smf

model = smf.ols("Y~X1+X2+X3", data=df).fit()
print(model.summary())
```

5.8.2 Klasični OLS zapis (`sm.OLS(y, X)`)

U ovom pristupu korisnik ručno definira matricu prediktora X i mora eksplicitno dodati konstantu.

```
import statsmodels.api as sm

y = df["Y"]
X = df[["X1", "X2", "X3"]]
X_const = sm.add_constant(X)

model = sm.OLS(y, X_const).fit()
print(model.summary())
```

Razlika između dva pristupa

- **Formula API** koristi zapis nalik na R-u (npr. "$Y \sim X1+X2$") i automatski obavlja pripremu podataka. Idealan je za brzu specifikaciju modela i lakše čitanje koda.
- **Čisti OLS pristup** zahtijeva ručno konstruiranje matrice X, uključujući dodavanje konstante. Daje veću fleksibilnost i veću kontrolu nad konstrukcijom modela (npr. kada se koriste skalirane, transformirane ili ručno kreirane značajke).
- Statistički i numerički modeli dobiveni jednim i drugim pristupom **identični su** — razlika je isključivo u načinu specificiranja modela.

5.9 Koeficijent determinacije

Koeficijent determinacije definiran je kao omjer sume kvadrata objašnjenog dijela i ukupne sume kvadrata, te poprima vrijednosti na intervalu $[0, 1]$. Koristi se kao deskriptivno obilježje snage linearnog odnosa između nezavisnih varijabli i zavisne varijable, te kao kriterij usporedbe modela na istim uzorcima.

Kvalitetni modeli imaju R^2 vrijednost blizu 1.

5.10 Skaliranje podataka

Skaliranje podataka prilagođava vrijednosti varijabli na određeni raspon, najčešće $[0, 1]$ ili $[-1, 1]$. Korisno je za modele osjetljive na skalu varijabli (k-NN, SVM, neuronske mreže).

Standardizacija podrazumijeva transformaciju varijabli na sredinu 0 i standardnu devijaciju 1. Potrebna je kada varijable imaju različite skale i kada želimo stabilnije koeficijente u regresijskom modelu.

Kod jednostavne linearne regresije oblika:

$$y = \beta_0 + \beta_1 x,$$

standardizacija nije nužna.

Interpretacija bez standardizacije zadržava mjerne jedinice, što olakšava tumačenje parametara.

Prednosti:

- koeficijenti β_0 i β_1 imaju stvarno značenje u kontektstu problema;
- podaci zadržavaju svoje mjerne jedinice

Ograničenja:

- ako x i y imaju vrlo različite skale, može nastati numerička nestabilnost.

Standardizacija je korisna:

- za usporedbu prediktora u višestrukoj regresiji (dob, prihod, razina obrazovanja) - standardizacija omogućuje lakše uspoređivanje njihovog relativnog utjecaja jer svi predikotri imaju istu sklau (srednja vrijednost 0, standardna devijacija 1)
- kada se koriste različiti skupovi podataka,
- kod optimizacijskih metoda (npr. gradijentni spust - standardizacija može poboljšati stabilnost i brzinu konverencije algoritma).

5.11 Kako standardizirati

Biblioteka scikit-learn sadrži klase:

(a) StandardScaler Standardizacija ili normalizacija z-vrijednosti, transformira podatke tako da im je ar.sredina jednaka nula i standardna devijacija 1. Ova metoda je korisna kada varijable imaju različite skale i mjerne jedinice.

```
from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()
scaled_data = scaler.fit_transform(data)
```

(b) MinMaxScaler Normalizacija ili skaliranje podataka na fiksni raspon, obično između 0 i 1. Ova tehnika je korisna kada varijable imaju različite raspone i želi se sačuvati izvorna distribucija podataka.

```
from sklearn.preprocessing import MinMaxScaler

scaler = MinMaxScaler()
scaled_data = scaler.fit_transform(data)
```

(c) RobustScaler Normalizacija medijana i kvantila, mjeri podatke na temelju robusnih procjena lokacije i razmjera. Ova metoda je korisna kada podaci sadrže ekstremne vrijednosti (otporan na outliere) ili kada distribucija ne prati Gaussovu razdiobu.

```
from sklearn.preprocessing import RobustScaler

scaler = RobustScaler()
scaled_data = scaler.fit_transform(data)
```

Napomena: Više o ovome možete pronaći na linku.

5.12 Pretvaranje kategorijskih značajki u numeričke

Najčešće se koriste dvije metode: **Label Encoding** i **One-Hot Encoding**, ali se moram paziti kada se koja koristi.

5.12.1 Label Encoding

Kodiranje oznakama - pretvara svaku kategoriju u jedinstveni cijeli broj, ali ako ima više katerogrija, onda uvodi “redoslijed” među kategorijama, što nije prikladno u većini slučajeva s više kategorija.

```
from sklearn.preprocessing import LabelEncoder

label_encoder = LabelEncoder()
df["mainroad"] = label_encoder.fit_transform(df["mainroad"])
```

5.12.2 One-Hot Encoding

Koristi se kod varijabli koje imaju više kategorija. Ova metoda stvara novi binarni stupac za svaku kategoriju.

Prednosti: Uklanja problem "redoslijeda" među kategorijama; prikladno za značajke s više kategorija. Nedostaci: Može stvoriti veliki broj novih kolona, što povećava dimenzionalnost podataka.

Primjer 5.1. Pretpostavimo da imamo dataset s kategorijskom značajkom City koja ima tri različite vrijednosti: London, Paris, i Berlin.

```
import pandas as pd

data = {"City": ["London", "Paris", "Berlin", "London", "
    Berlin"]}
df = pd.DataFrame(data)

df_encoded = pd.get_dummies(df, columns=["City"], drop_first=
    False)
print(df_encoded)
```

Koristimo `pd.get_dummies()` za pretvaranje kategorijskih vrijednosti u binarne stupce.

Objašnjenje rezultata: Svaka jedinstvena vrijednost u značajki City postaje zaseban stupac.

Vrijednosti u stupcima su:

1 ako redak pripada toj kategoriji.

0 inače.

Uklanjanje prvog stupca (`drop_first=True`): Da biste izbjegli višestruku kolinearnost (redundantne informacije), možete ukloniti jedan stupac koristeći `drop_first=True`.

```
df_encoded = pd.get_dummies(df, columns=["City"], drop_first=
    True)

print("\nPodaci s uklonjenom prvom kategorijom:")
print(df_encoded)
```

Berlin nije obrisan, već je implicitno predstavljen kao "osnovna kategorija". To znači da kada su sve ostale kategorije (`City_London` i `City_Paris`) 0, tada znamo da je pripadajuća kategorija `Berlin`.

Ovo je poznato kao referentna kategorija i koristi se za izbjegavanje višestruke kolinearnosti u regresijskim modelima. Kada koristimo `drop_first=True`, uklanjamo jednu kategoriju (u ovom slučaju, `City_Berlin`) i time smanjujemo redundantnost podataka.

5.12.3 Binary_map za binarne kategorijske značajke

Za pretvaranje binarnih kategorijskih značajki u numeričke (0 i 1) može se koristiti i funkcija `binary_map()`, što predstavlja brži i jednostavniji pristup od One-Hot Encodinga kada se radi o dvjema kategorijama.

- **Jednostavnost:**
  - Ako je kategorijska značajka već binarna (npr. `yes/no`, `true/false`, `male/female`), `binary_map()` izravno mapira vrijednosti na brojeve:
    - `yes` $\rightarrow 1$, `no` $\rightarrow 0$
    - `True` $\rightarrow 1$, `False` $\rightarrow 0$
    - `Male` $\rightarrow 1$, `Female` $\rightarrow 0$
- **Brzina i efikasnost:**
  - Kod binarnih značajki nije potrebno stvarati nove stupce kao kod One-Hot Encodinga.
  - Mapiranje se izvodi izravno nad postojećim stupcem, što smanjuje memorijsku potrošnju i ubrzava obradu.
- **Lakša interpretacija:**
  - Rezultirajuća vrijednost je intuitivna: 1 predstavlja jednu kategoriju, a 0 drugu.
- **Implementacija:**
  - Najčešće se implementira kao prilagođena Python funkcija koja koristi metode `map()` ili `replace()`, primjerice:

```
def binary_map(feature):
return feature.map({"yes": 1, "no": 0})
```

Za binarne značajke (yes/no, male/female):

```
def binary_map(feature):
return feature.map({"yes": 1, "no": 0})

df["mainroad"] = binary_map(df["mainroad"])
```

ili:

```
df["mainroad"] = df["mainroad"].replace({"yes": 1, "no": 0})
```

Za True/False:

```
housing["semi-furnished"] = housing["semi-furnished"].map(
{True: 1, False: 0}
)
```

Kada NE koristiti:

- **One-Hot Encoding** — nepotrebno za binarne varijable,
- **Label Encoding** — uvodi umjetni poredak.

5.13 Multikolinearnost

Variance Inflation Factor (VIF) kvantificira multikolinearnost među prediktorima.

Što je multikolinearnost?

Multikolinearnost nastaje kada su dvije ili više nezavisnih varijabli u regresijskom modelu **međusobno snažno povezane**.

Posljedice:

- koeficijenti postaju numerički nestabilni (male promjene u podacima mogu uzrokovati velike promjene u procijenjenim koeficijentima),
- standardne pogreške koeficijenata rastu,
- t-testovi za pojedine koeficijente mogu pokazivati neznačajnost iako model kao cjelina ima dobru objašnjavačku moć,
- teško je odvojiti utjecaj pojedinih prediktora.

Variance Inflation Factor (VIF)

VIF kvantificira razinu multikolinearnosti za svaku varijablu X_i:

Formula:

$$VIF_i = \frac{1}{1 - R_i^2} \tag{3}$$

gdje je R_i^2 koeficijent determinacije dobiven regresijom varijable X_i na sve ostale prediktore.

Interpretacija

- VIF = 1: nema multikolinearnosti,
- VIF > 5: visoka multikolinearnost,
- VIF > 10: ekstremna multikolinearnost.

Primjer 5.2.

```
from statsmodels.stats.outliers_influence import
    variance_inflation_factor

data["Gender"] = data["Gender"].map({"Male": 0, "Female"
    : 1})
X = data[["Gender", "Height", "Weight"]]

vif_data = pd.DataFrame()
vif_data["feature"] = X.columns
vif_data["VIF"] = [
variance_inflation_factor(X.values, i)
      for i in range(len(X.columns))
]
print(vif_data)
```

Napomena: više na linku.

6 Zadaci za pripremu

Opis vježbe

U ovoj pripremi za laboratorijsku vježbu koristiti skup podataka *Airlines Customer Satisfaction* (korisnička zadovoljstva u zračnom prijevozu).

Ciljevi pripreme za vježbu su:

- upoznati se s osnovama rada s kategorijskim i numeričkim varijablama u Pandas DataFrameu,
- izračunati i interpretirati koeficijente korelacije između numeričkih varijabli,
- izgraditi **logistički regresijski model** za predikciju **zadovoljstva putnika**,
- interpretirati koeficijente logističke regresije (znak, veličina, odds ratio),
- izgraditi višestruku regresiju

Opis skupa podataka

Skup podataka sadrži zapise o putnicima i njihovim ocjenama usluge. Neke od varijabli su:

- `Gender` – spol putnika (Male/Female),
- `Customer Type` – tip putnika (Loyal customer / disloyal),
- `Age` – dob putnika,
- `Type of Travel` – vrsta putovanja (Business / Personal),
- `Class` – klasa putovanja (Business, Eco, Eco Plus),
- `Flight Distance` – udaljenost leta,

- niz ocjena usluge (skala 1–5): `Seat comfort`, `Food and drink`, `Inflight wifi service`, `Inflight entertainment`, `On-board service`, `Cleanliness`, `Baggage handling`, `Leg room service`, ...
- `Departure Delay in Minutes`, `Arrival Delay in Minutes`,
- `Satisfaction` – ciljna varijabla (Satisfied / dissatisfied).

6.1 Zadatak: Učitavanje i osnovno čišćenje podataka

1. Uvezite potrebne biblioteke (`pandas`, `numpy`, `matplotlib`, `seaborn`, `statsmodels`, `sklearn` po potrebi).
2. Učitajte CSV datoteku s podacima, npr.:

```
import pandas as pd
df = pd.read_csv("Naziv_dokumenta.csv")
```

3. Ispitajte osnovne informacije o DataFrameu:

```
df.info()
df.describe()
```

4. Provjerite nedostajuće vrijednosti i po potrebi ih uklonite ili zamijenite:

```
df.isna().sum()
df = df.dropna()
```

6.2 Zadatak: Priprema varijabli za analizu

1. Kreirajte binarnu varijablu zadovoljstva, jer ima samo dvije vrijednosti:

```
df["Satisfied"] = df["satisfaction"].map({
 "jedna_kategorija": 1,
 "druga_kategorija": 0
})
```

Provjeriti je li dobro kategorizirano.

```
df["Satisfied"].unique()
```

Ako je sve u redu, onda rješenje treba izgledati: `array([1,0])`

Možete provjeriti i s:

```
df["Satisfied"].value_counts()
```

2. Odaberite skup numeričkih varijabli za analizu korelacije, npr.:
   - `Age`,
   - `Flight Distance`,
   - `Seat comfort`,
   - `Inflight entertainment`,
   - `On-board service`,
   - `Cleanliness`,
   - `Departure Delay in Minutes`,
   - `Arrival Delay in Minutes`.

```
num_cols = ["Age", "Flight Distance",
"Seat comfort", "Inflight entertainment",
"On-board service", "Cleanliness",
"Departure Delay in Minutes",
"Arrival Delay in Minutes",
"Satisfied"]
df_num = df[num_cols]
```

6.3 Zadatak: Odabir prikladnog koeficijenta korelacije

U nastavku je naveden skup varijabli iz baze *Airlines Customer Satisfaction*. Cilj zadatka je odrediti koje se varijable smiju koristiti s Pearsonovim koeficijentom korelacije, a koje zahtijevaju alternativne koeficijente (Spearman, Kendall, Phi, Cramer's V).

Varijabla	**Tip varijable**
Age	numerička (kvantitativna)
Flight Distance	numerička (kvantitativna)
Seat comfort (1–5)	ordinalna (ocjena)
Inflight entertainment (1–5)	ordinalna (ocjena)
Cleanliness (1–5)	ordinalna (ocjena)
Departure Delay in Minutes	numerička (kvantitativna)
Gender	nominalna (M/F)
Customer Type	nominalna (Loyal/Disloyal)
Type of Travel	nominalna (Business/Personal)
Class	nominalna (tri kategorije)
Satisfaction	nominalna (zadovoljan/nezadovoljan)

6.3.1 1. Varijable prikladne za Pearsonovu korelaciju

Pearsonov koeficijent koristi se za ispitivanje linearne povezanosti između **kvantitativnih varijabli** (intervalne ili omjerne skale). Primjeri varijabli koje se mogu koristiti:

- Age,
- Flight Distance,
- Departure Delay in Minutes,
- Arrival Delay in Minutes (ako postoji u odabranom skupu).

Ordinalne varijable na skali 1–5 ponekad se koriste s Pearsonom kao aproksimacija numeričke skale, ali teoretski su prikladnije za Spearmanov koeficijent (vidi iduću sekciju).

6.3.2 2. Varijable prikladne za Spearmanov ili Kendallov koeficijent

Spearmanov ρ i Kendallov τ koriste se za:

- ordinalne varijable (rangovi),
- distribucije koje ne prate normalnu razdiobu,
- monotone, ali ne nužno linearne odnose.

Za ove varijable prikladniji je Spearman ili Kendall:

- Seat comfort,
- Inflight entertainment,
- Cleanliness,
- On-board service,
- sve ostale ocjene usluge (1–5).

6.3.3 3. Varijable koje zahtijevaju nominalne koeficijente

Nominalne varijable nemaju prirodan poredak te se ne smiju analizirati Pearsonovim koeficijentom. Umjesto njega koriste se:

- **Phi koeficijent** — kada su obje varijable binarne (0/1),
- **Cramer's V** — za nominalne varijable s dvije ili više kategorija.

Primjeri varijabli koje zahtijevaju nominalne koeficijente:

- Gender (M/F),
- Customer Type,
- Type of Travel,
- Class,
- Satisfaction (ako je kategorijska, prije binarizacije).

Zadatak za pripremu:

1. Razvrstajte sve navedene varijable u tri skupine:
   (a) varijable za Pearson,
   (b) varijable za Spearman/Kendall,
   (c) varijable za Phi/Cramer's V.
2. Izračunajte Pearsonovu korelaciju samo za skup kvantitativnih varijabli koje ste odabrali i komentirajte dobivenu matricu korelacije.
3. Izračunajte Spearman/Kendall korelaciju za skup ordinalnih varijabli koje ste odabrali i komentirajte dobienu matricu korelacije
4. Odaberite dvije nominalne varijable i izračunajte Phi ili Cramer's V Protumačite dobivenu vrijednost.

6.4 Zadatak: Koeficijenti korelacije

1. Izračunajte **Pearsonovu korelacijsku matricu**, primjer koda je sljedeći:

```
corr_pearson = df[cols_pearson].corr(method="pearson")
print(corr_pearson)
```

   (a) Vizualizirajte matricu korelacije pomoću `seaborn.heatmap`.
   (b) Identificirajte:
      - dvije varijable s najjačom pozitivnom korelacijom,
      - dvije varijable s najjačom negativnom korelacijom,
      - koje varijable su najjače povezane s varijablom `Satisfied`.
   (c) Komentirajte dobivene rezultate: npr. ima li smisla da kvaliteta sjedala ili čistoća budu povezane s zadovoljstvom putnika.
2. Izračunajte **Spearmanovu korelaciju matricu**, primjer koda je sljedeći:

```
corr_spearman = df[cols_spearman].corr(method="spearman")
print(corr_spearman)
```

   (a) prvo odabrati barem pet ordinalnih varijabli (ocjena usluge).
   (b) Vizualizirajte matricu korelacije pomoću `seaborn.heatmap`

```
import seaborn as sns
import matplotlib.pyplot as plt

plt.figure(figsize=(10, 8))
sns.heatmap(corr_spearman, annot=True,
  cmap="coolwarm", fmt=".2f")
plt.title("Spearmanova korelacijska matrica")
plt.show()
```

(c) Na temelju dobivene matrice i vizualizacije:

- Identificirajte parove varijabli s **najvišom pozitivnom** korelacijom.
- Identificirajte varijable s **najslabijom** ili **negativnom** korelacijom.
- Komentirajte imaju li rezultati smisla (npr. očekivano je da su *Seat comfort* i *Leg room service* snažno pozitivno korelirani).

3. Izračunajte **Phi koeficijent** za par binarnih varijabli. Primjer koda je sljedeći:

```
from scipy.stats import chi2_contingency
import numpy as np

cont = pd.crosstab(df[var1], df[var2])
chi2, p, dof, exp = chi2_contingency(cont)
phi = np.sqrt(chi2 / df.shape[0])
print("Phi koeficijent =", phi)
```

(a) Najprije odaberite dvije **binarne varijable**, npr.:

- `Male` (1 = muškarac, 0 = žena),
- `Satisfied` (1 = zadovoljan, 0 = nezadovoljan),
- `BusinessTravel` (1 = poslovno putovanje, 0 = osobno).

(b) Kreirajte kontingencijsku tablicu za odabrane varijable:

```
cont = pd.crosstab(df[var1], df[var2])
print(cont)
```

(c) Izračunajte Phi koeficijent prema prikazanom kodu.

(d) Na temelju dobivene vrijednosti Phi koeficijenta:

- Identificirajte postoji li **pozitivna**, **negativna** ili **vrlo slaba** povezanost između varijabli.
- Usporedite dobivenu vrijednost s uobičajenim interpretacijskim kriterijima:
  - 0.00–0.10: vrlo slaba povezanost,
  - 0.10–0.20: slaba povezanost,
  - 0.20–0.40: umjerena povezanost,
  - > 0.40: jaka povezanost.
- Komentirajte imaju li dobiveni rezultati smisla. Primjer: očekivano je da je *BusinessTravel* pozitivno povezan sa zadovoljstvom, jer poslovni putnici često imaju bolju uslugu.

6.4.1 Identifikacija i uklanjanje značajki s visokom multikolinearnošću

Prije izgradnje regresijskih modela potrebno je provjeriti postoje li varijable koje su međusobno snažno povezane, jer multikolinearnost može dovesti do:

- nestabilnih procjena koeficijenata,
- rasta standardnih pogrešaka,
- problema u interpretaciji modela.

Analiza se provodi u dvije faze.

(a) Izračun korelacijskih koeficijenata Na temelju gore izračunatih matrica korelacije potrebno je:

- identificirati varijable s vrlo visokom korelacijom ($|r| > 0.8$),
- procijeniti koje je varijable poželjno izostaviti ili kombinirati,
- odabrati reprezentativne značajke koje unose najviše informacije u model.

(b) Variance Inflation Factor (VIF) VIF se koristi za kvantitativno određivanje multikolinearnosti među prediktorima.

$$VIF_j = \frac{1}{1 - R_j^2}$$

gdje je R_j^2 koeficijent determinacije iz pomoćnog OLS modela u kojem se promatrana značajka regresira na sve ostale značajke.

```
from statsmodels.stats.outliers_influence import
    variance_inflation_factor
import pandas as pd

X = df[selected_features] # numericke i dummy varijable
X_const = sm.add_constant(X)

vif_data = pd.DataFrame()
vif_data["feature"] = X.columns
vif_data["VIF"] = [
variance_inflation_factor(X_const.values, i+1) # preskace
    konst., i+1
for i in range(len(X.columns))
]

print(vif_data)
```

Vrijednosti VIF-a tumače se na sljedeći način:

- $VIF < 5$ — prihvatljiva razina multikolinearnosti,
- $5 \leq VIF < 10$ — visoka multikolinearnost, **moguće uklanjanje varijable** - provjeriti koeficijente korelacije prije brisanja,
- $VIF \geq 10$ — vrlo visoka multikolinearnost, **uklanjanje ili transformacija varijable preporučena**.

Na temelju dane baze podataka, sređenih podataka, možemo na primjer analizirati VIF:

```
from statsmodels.tools import add_constant
from statsmodels.stats.outliers_influence import
   variance_inflation_factor
import numpy as np


df_encoded = pd.get_dummies(df, columns = ['Class', 'Type of Travel'
   ], drop_first= True)

df_encoded[["Class_Eco", "Class_Eco Plus",
"Type of Travel_Personal Travel"]].astype(int)

X = add_constant(df_encoded[features])

# 1) Pretvori bool u 0/1 i sve u float
bool_cols = X.select_dtypes(include='bool').columns
X[bool_cols] = X[bool_cols].astype(int)
X = X.astype(float)

# 2) Ukloni NaN i inf za svaki slucaj
X = X.replace([np.inf, -np.inf], np.nan).dropna()

# 3) VIF s indeksima i (bez i+1, jer je const unutra)
vif = pd.DataFrame()
vif["Feature"] = X.columns
vif["VIF"] = [variance_inflation_factor(X.values, i)
for i in range(X.shape[1])]

print(vif)
```

Napomena: Za pripremu za LV isprobati druge varijable.

6.4.2 Normalizacija ili standardizacija numeričkih značajki

Ako numeričke varijable koje se koriste u regresijskom modelu imaju različite raspona vrijednosti (npr. `Age`, `Flight Distance`, `Delay Minutes`), preporučuje se primjena standardizacije kako bi se:

- poboljšala stabilnost optimizacijskog algoritma u logističkoj regresiji,
- spriječilo da varijable s velikim rasponom dominiraju nad ostalima,
- omogućila korektna interpretacija koeficijenata.

Za standardizaciju se koristi `StandardScaler`:

```
    from sklearn.preprocessing import StandardScaler

    scaler = StandardScaler()

```

```
    numerical = df[numeric_cols]
    scaled = scaler.fit_transform(numerical)

    df_scaled = df.copy()
    df_scaled[numeric_cols] = scaled
```

Standardizacija transformira svaku značajku prema formuli:

$$z = \frac{x - \bar{x}}{s_x}$$

Napomena: Standardiziraju se samo:

- numeričke varijable,
- ne standardiziraju se dummy varijable (0/1),
- ne standardiziraju se već binarizirane kategorije.

6.5 Zadatak: Logistička regresija

U ovom zadatku želimo modelirati vjerojatnost da je putnik zadovoljan (`Satisfied` = 1) na temelju nekih od ocjena usluge i karakteristika leta.

1. Odaberite nekoliko prediktora, npr.:
   - `Seat comfort`,
   - `Inflight entertainment`,
   - `On-board service`,
   - `Cleanliness`,
   - `Flight Distance`.

   ```
   X = df[["Seat comfort",
   "Inflight entertainment",
   "On-board service",
   "Cleanliness",
   "Flight Distance"]]
   y = df["Satisfied"]
   ```

2. Po potrebi standardizirajte numeričke prediktore (nije obavezno za ovu vježbu, ali može pomoći pri interpretaciji i numeričkoj stabilnosti).
3. Dodajte konstantu i izgradite logistički model pomoću `statsmodels`:

   ```
   import statsmodels.api as sm

   X_const = sm.add_constant(X)
   logit_model = sm.Logit(y, X_const).fit()
   print(logit_model.summary())
   ```

4. Iz sažetka modela zapišite procijenjene koeficijente $\hat{\beta}_j$, pripadne p-vrijednosti i komentirajte:
   - koji prediktori su statistički značajni,
   - imaju li koeficijenti očekivani znak (npr. viša ocjena udobnosti sjedala povećava vjerojatnost zadovoljstva).
5. Izračunajte **odds ratio** za svaki koeficijent:

```
params = logit_model.params
odds_ratios = params.apply(lambda b: np.exp(b))
print(odds_ratios)
```

6. Interpretirajte barem jedan koeficijent, npr. za `"Seat comfort"`: objasnite koliko puta se mijenjaju izgledi za zadovoljstvo pri povećanju ocjene za jednu jedinicu.

6.6 Višestruka linearna regresija

U ovom zadatku potrebno je primijeniti sve korake obrade podataka i izgradnje modela višestruke linearne regresije.

Postavljanje višestruke linearne regresije

Korak 4.1: Definiranje ciljnih i ulaznih varijabli

Odaberite:

- ciljnu varijablu y (zavisna varijabla),
- skup značajki X (nezavisne varijable) koje uključuju numeričke i prethodno enkodirane kategorijske varijable.

```
y = df["TargetVariable"]
X = df[["Feature1", "Feature2", "Feature3"]]
X = sm.add_constant(X)
```

Korak 4.2: Izgradnja modela korištenjem statsmodels

- Importirati funkciju `OLS` iz biblioteke `statsmodels.api`.
- Izgraditi model i prikazati rezultate pomoću `summary()`.

```
import statsmodels.api as sm
import statsmodels.formula.api as smf

model = sm.OLS(y, X).fit()

```

```
    # ili

    model = smf.ols("y ~ Feature1 + Feature2 + Feature3", data=df)
        .fit()

    print(model.summary())
```

5. Analiza rezultata modela

(a) Interpretacija koeficijenata

Promotrite izlaz modela i odgovorite na sljedeća pitanja:

- Koje značajke imaju najveći utjecaj na ciljnu varijablu?
- Jesu li koeficijenti statistički značajni (p-vrijednost < 0.05)?
- Ima li značajki čiji je učinak suprotan očekivanom?

(b) Kvaliteta modela

Analizirajte sljedeće mjere:

- R^2 (koeficijent determinacije) — koliki dio varijabilnosti ciljna varijabla objašnjava model?
- Rezidualni standardni error — je li model precizan?
- Statistika F-testa — ima li model kao cjelina objašnjavačku moć?

(c) Analiza reziduala

Korištenjem predviđenih vrijednosti iz modela, izračunajte reziduale:

```
    df["pred"] = model.fittedvalues
    df["resid"] = model.resid
```

Odgovorite:

- Postoje li prepoznatljivi obrasci u rezidualima?
- Jesu li reziduali približno simetrično raspodijeljeni oko nule?
- Ukazuju li reziduali na heteroskedastičnost?

6. Vizualizacija rezultata

(a) Stvarne naspram predviđenih vrijednosti

```
y_pred = model.predict(X)

plt.scatter(y, y_pred)
plt.xlabel("Stvarne vrijednosti (y)")
plt.ylabel("Predvidjene vrijednosti ()")
plt.title("Stvarne vs. predvidjene vrijednosti")
plt.plot([y.min(), y.max()], [y.min(), y.max()], "r--") # idealna
   dijagonala
plt.show()
```

(b) Vizualizacija reziduala

```
y_pred = model.predict(X)
resid = y - y_pred

plt.scatter(y_pred, resid)
plt.axhline(0, color="red", linestyle="--")
plt.xlabel("Predvidjene vrijednosti ()")
plt.ylabel("Reziduali")
plt.title("Dijagnosticki graf reziduala")
plt.show()
```

Zaključak

Odgovorite na sljedeća pitanja:

- Koji su ključni faktori koji utječu na ciljnu varijablu?
- Jesu li rezultati u skladu s očekivanjima (pozitivan/negativan utjecaj, značajnost)?
- Kako se višestruka linearna regresija može koristiti za interpretaciju odnosa među varijablama?
- Koja biste poboljšanja predložili (npr. uklanjanje značajki, transformacija varijabli, dodavanje novih prediktora, standardizacija)?

7 Sažetak zadataka za pripremu

7.1 Učitavanje i početni pregled podataka

- učitati skup podataka i prikazati osnovne informacije: broj redaka, tipovi varijabli, statistički opis,
- provjeriti postojanje nedostajućih vrijednosti i, po potrebi, ukloniti ili imputirati retke.

7.2 Priprema binarne ciljne varijable

- varijablu `satisfaction` potrebno je pretvoriti u binarni oblik, pri čemu vrijednost *satisfied* poprima vrijednost 1, a *neutral or dissatisfied* vrijednost 0,
- provjeriti ispravnost kreirane varijable (jedinstvene vrijednosti i distribucija).

7.3 Odabir numeričkih i ordinalnih varijabli

- odabrati numeričke varijable (npr. dob, udaljenost leta, kašnjenja),
- izdvojiti ordinalne varijable (ocjene usluge na skali 1–5).

7.4 Korelacijska analiza

- izračunati Pearsonovu korelaciju za numeričke varijable i interpretirati najjače pozitivne i negativne veze,
- izračunati Spearmanovu korelaciju nad ordinalnim varijablama i komentirati logičnost pronađenih odnosa,
- za odabrane binarne varijable izračunati Phi koeficijent i interpretirati jačinu povezanosti prema standardnim kriterijima.

7.5 Provjera multikolinearnosti

- pregledati korelacijsku matricu i identificirati parove varijabli s visokom međusobnom korelacijom,
- izračunati VIF (Variance Inflation Factor) za odabrane značajke,
- komentirati koje varijable imaju previsoku kolinearnost te ponuditi rješenja (uklanjanje, spajanje ili transformacija).

7.6 Priprema podataka za modele

- pretvoriti kategorijske varijable u dummy varijable (npr. one-hot encoding),
- po potrebi standardizirati numeričke varijable radi usporedbe koeficijenata.

7.7 Logistička regresija

- definirati ciljnu varijablu i odabrati prediktore,
- izgraditi logistički regresijski model,
- interpretirati koeficijente i p-vrijednosti,
- izračunati omjere izgleda (odds ratios) i objasniti njihov praktični smisao.

7.8 Višestruka linearna regresija (priprema za LV6)

- definirati ciljnu i ulazne varijable,
- izgraditi OLS model i interpretirati dobivene parametre,
- analizirati kvalitetu modela putem R^2, prilagođenog R^2 i F-testa,
- grafički prikazati stvarne i predviđene vrijednosti te analizirati reziduale.

7.9 Zaključak

Student mora opisati:

- koji su prediktori najvažniji,
- kakvi su odnosi među varijablama,
- kvaliteta prediktivnih modela i mogućnosti njihovog poboljšanja.

8 LV5 – Korelacija i višestruka linearna regresija

Analiza cijena nekretnina

Cilj vježbe: Studenti će analizirati dani dataset o nekretninama primjenom deskriptivne statistike, koeficijenata korelacije i višestruke linearne regresije. Naučit će odabrati značajke, postaviti i procijeniti regresijski model te interpretirati koeficijente i prikladnost modela.

Dataset: Housing.csv (Kaggle housing-dataset). Ciljna varijabla je `price`.

1. **Učitavanje i pregled podataka**
   (a) Učitati dataset u Pythonu pomoću paketa `pandas`.
   (b) Ispisati broj redaka i stupaca te kratko objasniti što predstavljaju.
   (c) Analizirati tipove podataka po stupcima i razlikovati numeričke i kategoričke varijable.
   (d) Provjeriti postoje li nedostajuće vrijednosti.
   (e) Prikazati prvih nekoliko redaka dataseta i ukratko opisati sadržaj.
2. **Deskriptivna analiza i korelacija**
   (a) Izračunati osnovne statistike (srednja vrijednost, medijan, standardna devijacija, min, max, kvartili) za numeričke varijable (`area`, `bedrooms`, `bathrooms`, `stories`, `parking`, `price`) i ukratko ih interpretirati.
   (b) Nacrtati scatter dijagrame između `price` i svake numeričke varijable te komentirati uočene odnose.

(c) Izračunati Pearsonovu korelacijsku matricu za numeričke varijable. Identificirati varijable koje imaju najveću pozitivnu korelaciju s `price` te eventualno varijable sa slabom korelacijom. Kratko interpretirati dobivene koeficijente i objasniti zašto korelacija ne podrazumijeva uzročnost.

3. **Priprema podataka za višestruku regresiju**

   (a) Definirati ciljnu varijablu `y = price` te početni skup ulaznih varijabli (numeričke i kategoričke).

   (b) Binarne kategoričke varijable mapirati na 0/1, a višeklasne varijable kodirati metodom one-hot (npr. `pd.get_dummies` s `drop_first=True`).

   (c) Kreirati DataFrame `data_selected` s ciljnom varijablom i svim odabranim značajkama te provjeriti da ne postoje nedostajuće vrijednosti.

   (d) Izračunati VIF (Variance Inflation Factor) za sve značajke. Identificirati značajke s visokim VIF-om i po potrebi ih ukloniti iz modela te ukratko obrazložiti izbor.

4. **Postavljanje modela višestruke linearne regresije (statsmodels)**

   (a) Definirati `y` kao `price`, a `X` kao matricu značajki te dodati stupac konstante.

   (b) Procijeniti model korištenjem funkcije `OLS` iz paketa `statsmodels` i ispisati sažetak rezultata (`summary()`).

   (c) Zapisati jednadžbu regresijskog modela s procijenjenim koeficijentima.

   (d) Identificirati statistički značajne varijable (p-vrijednost < 0.05) i ukratko interpretirati znak i veličinu njihovih koeficijenata.

   (e) Interpretirati vrijednosti R^2, prilagođenog R^2 i F-statistike te procijeniti kvalitetu modela.

5. **Provjera pretpostavki i vizualizacija rezultata**

   (a) Izračunati predviđene vrijednosti i reziduale modela.

   (b) Nacrtati scatter dijagram stvarnih naspram predviđenih vrijednosti ciljane varijable i komentirati dobivene rezultate.

   (c) Nacrtati graf reziduala naspram predviđenih vrijednosti i komentirati zadovoljava li se pretpostavka homoskedastičnosti.

   (d) Prikazati histogram i/ili QQ-plot reziduala i ukratko komentirati odstupanja od normalne distribucije.

6. **Primjena modela i zaključak**

   (a) Odabrati ili konstruirati barem jedan primjer kuće s konkretnim vrijednostima značajki te uz pomoć procijenjenog modela izračunati predviđenu cijenu. (**dodatno**)

   (b) U nekoliko rečenica zaključiti koji su ključni faktori koji utječu na cijenu nekretnine prema dobivenom modelu i može li se model dodatno poboljšati.