

4. Linearni modeli za regresiju. Vrednovanje rezultata

4.1 Cilj Vježbe

Upoznati se s problemom nadgledanog učenja te linearnim modelima za rješavanje regresijskih problema i načinom njihovog vrednovanja. Upoznati se sa Scikit-learn bibliotekom za strojno učenje.

4.2 Teorijska pozadina

4.2.1 Nadzirano učenje. Regresijski problem

U ovoj vježbi razmatra se problem **nadziranog učenja** (engl. supervised learning) gdje je cilj odrediti nepoznatu funkcionalnu ovisnost između m ulaznih veličina $X = [x_1, x_2, \dots, x_m]$ i izlazne veličine y na temelju podatkovnih primjera. Podatkovni primjeri mogu se predstaviti kao parovi koji se sastoje od vektora ulaznih veličina i odgovarajućih vrijednosti izlazne veličine. Stoga, i -ti podatkovni primjer se može prikazati kao uređeni par $(x^{(i)}, y^{(i)})$ pri čemu je vektor ulaznih jednak:

$$x^{(i)} = [x_1^{(i)}, x_2^{(i)}, \dots, x_m^{(i)}]^\top. \quad (4.1)$$

Podatkovni skup koji se sastoji od n raspoloživih primjera $(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \dots, (x^{(n)}, y^{(n)})$ može se zapisati u matričnom obliku:

$$X = \begin{bmatrix} x_1^{(1)}, x_2^{(1)}, \dots, x_m^{(1)} \\ x_1^{(2)}, x_2^{(2)}, \dots, x_m^{(2)} \\ \vdots \\ x_1^{(n)}, x_2^{(n)}, \dots, x_m^{(n)} \end{bmatrix}, y = \begin{bmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(n)} \end{bmatrix} \quad (4.2)$$

Ovakav skup se često naziva i označeni skup podataka (engl. *labelled dataset*) budući da svaki primjer ima vrijednost izlazne veličine odnosno svoju oznaku (engl. *label*). U ovoj vježbi promatra se slučaj kada je izlazna veličina y kontinuirana veličina odnosno vrijednosti vektora y su

iz skupa realnih brojeva. Ovakvi problemi nazivaju se regresijski problemi. Cilj nadziranog učenja je odrediti aproksimaciju \hat{f} funkcionalne ovisnosti f između ulaznih veličina i izlazne veličine. Ova aproksimacija se onda koristi za predikciju izlazne veličine za novi (nepoznati) uzorak ulaznih veličina \mathbf{x} :

$$\hat{y}(x) = \hat{f}(x) = h_{\theta}(x) \quad (4.3)$$

Dobivena aproksimacija obično se naziva **model** koji se često označava i s h_{θ} . Model se najčešće definira kao funkcija s konačnim brojem parametara θ . Pri tome se od svih mogućih funkcija odabire samo mali podskup mogućih funkcija, kao primjerice linearne funkcije, a **postupkom učenja** (engl. *training*) upravo se određuju ovi nepoznati parametri θ na temelju raspoloživih podataka.

Procjena nepoznatih parametara modela temelji se na dostupnim podacima za učenje. Pri tome, potrebno je definirati kriterij koji brojčano iskazuje koliko je dobar određeni model s nekim parametrima na danom skupu podataka za učenje. Kao kriterij koristi se empirijska pogreška modela i u slučaju regresijskih problema definira se kao:

$$J(\theta) = \frac{1}{2n} \sum_{i=1}^n (h_{\theta}(x^{(i)}) - y^{(i)})^2 \quad (4.4)$$

Često se kriterij 4.4 naziva i funkcija kvadratne pogreške. Optimalni parametri modela θ^* tada se dobivaju minimizacijom navedenog kriterija:

$$\theta^* = \operatorname{argmin}(J(\theta)) \quad (4.5)$$

4.2.2 Linearni modeli za regresiju

Kao aproksimacijsku funkciju moguće je koristiti linearu funkciju. Model je u tom slučaju oblika:

$$\hat{y}(x^{(i)}) = h_{\theta}(x^{(i)}) = \theta_0 + \theta_1 x_1^{(i)} + \theta_2 x_2^{(i)} + \dots + \theta_m x_m^{(i)} = \theta_0 + \sum_{j=1}^m \theta_j x_j^{(i)} \quad (4.6)$$

i naziva se višedimenzionalna linearna regresija (engl. *multiple linear regression*). Obično se vektor ulaznih veličina proširuje s dodatnom ulaznom veličinom x_0 koja je jednaka 1. U tom slučaju cijeli se model može jednostavnije zapisati:

$$\hat{y}(x^{(i)}) = h_{\theta}(x^{(i)}) = \sum_{j=0}^m \theta_j x_j^{(i)} = \theta^\top x^{(i)} \quad (4.7)$$

gdje je:

$$x^{(i)} = [x_0^{(i)}, x_1^{(i)}, x_2^{(i)}, \dots, x_m^{(i)}]^\top, \quad \theta = [\theta_0, \theta_1, \dots, \theta_m]^\top. \quad (4.8)$$

Vektor parametara θ određuje se prema 4.5. Rješenje optimizacijskog problema u ovom slučaju postoji u zatvorenoj formi:

$$\theta = (X^\top X)^{-1} X^\top y. \quad (4.9)$$

Drugi način rješavanja optimizacijskog problema 4.5 podrazumijeva korištenje iterativnog numeričkog postupka. Na primjer, gradijentnom metodom moguće je pronaći optimalne vrijednosti parametara modela 4.7:

Metoda 1 Metoda gradijentnog spusta za optimizaciju kriterija 4.5

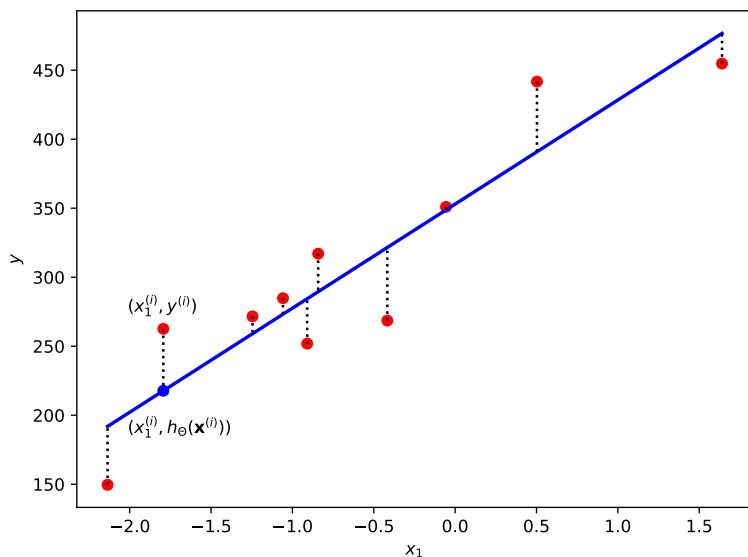
- 1: odaberi početnu vrijednost vektora parametara θ ; definiraj duljinu koraka α
- 2: simultano osvježi svaki element vektora θ :

$$\theta_j := \theta_j - \alpha \frac{\partial J(\theta)}{\partial \theta_j}, \text{ za } j = 0, \dots, m$$

$$\frac{\partial J(\theta)}{\partial \theta_j} = \frac{1}{n} \sum_{i=1}^n (h_\theta(x^{(i)}) - y^{(i)}) x_j^{(i)}$$

- 3: ako je zadovoljen kriterij zaustavljanja (npr. dostignut je maksimalni broj iteracija) zaustavi optimiranje; u suprotnom idи na 2. korak
-

Na slici 4.1 prikazan je regresijski problem s jednom ulaznom veličinom i 20 podatkovnih primjera. Izgrađeni linearni regresijski model prikazan je plavom bojom. Crne okomite linije predstavljaju pogrešku procjene modela za dani ulazni primjer.



Slika 4.1: Primjer regresijskog problema s jednom ulaznom veličinom.

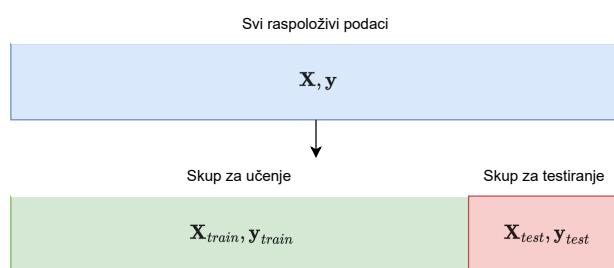
4.2.3 Podjela podataka na podatkovne skupove za učenje i testiranje

Izgradnja modela i testiranje modela na istom skupu podataka predstavlja metodološku pogrešku budući da je za uspješnu primjenu modela u praksi važno kakve će performanse model imati na

nepoznatim odnosno „neviđenim“ podacima. Stoga je uobičajena praksa podjela dostupnih podataka na skup podataka za učenje (trening skup) i skup podataka za testiranje (testni skup). Skup podataka za učenje služi za procjenu parametara modela dok se skup podataka za testiranje koristi tijekom vrednovanja modela kako bi se procijenile performanse izgrađenog modela. Konkretan omjer podjele ovisi o dostupnim podacima. Tipična podjela podataka na skup za učenje i skup za testiranje je u omjeru 80%-20%.

Primjer podjele podataka ilustriran je slikom 4.2. Neka su podaci pohranjeni u obliku 4.2 odnosno sastoje se od matrice X koja u retcima sadrži vrijednosti ulaznih veličina i izlaznog vektora y koji sadrži odgovarajuće vrijednosti izlazne veličine. Tada je podatke moguće podijeliti na skupove koji su opisani matricama/vektorima:

1. skup za učenje modela (trening skup) – X_{train}, y_{train}
2. skup za testiranje modela (testni skup) – X_{test}, y_{test}



Slika 4.2: Podjela podataka na skup za učenje i skup za testiranje modela.

4.2.4 Skaliranje ulaznih veličina

Većina algoritama strojnog učenja zahtijeva prethodno skaliranje ulaznih veličina. Skaliranjem ulaznih veličina se sve ulazne veličine svode na istu skalu. Na taj način se postiže brža konvergencija optimizacijskog postupka koji se temelji na gradijentnoj metodi. Dva su najčešća tipa skaliranja podataka: **min-max** i **standardizacija**.

Min-max skaliranje transformira vrijednosti ulazne veličine x_j na željeni raspon. Tipično se radi skaliranje na interval $[0, 1]$:

$$\tilde{x}_j = \frac{x_j - \min(x_j)}{\max(x_j) - \min(x_j)} \quad (4.10)$$

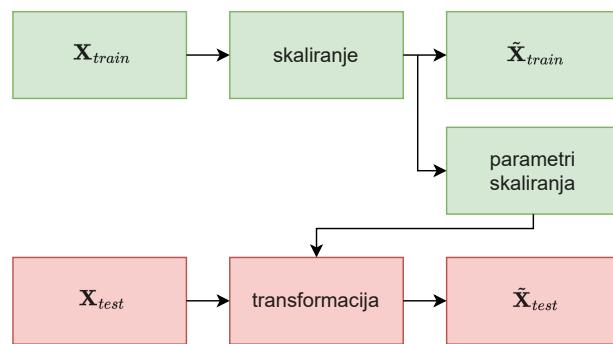
Standardizacija ulaznih veličina predstavlja skaliranje gdje podaci za svaku ulaznu veličinu imaju srednju vrijednost 0 i varijancu jednaku 1:

$$\tilde{x}_j = \frac{x_j - \bar{x}_j}{\sigma_j} \quad (4.11)$$

pri čemu su parametri skaliranja jednaki:

$$\bar{x}_j = \frac{1}{n} \sum_{i=1}^n x_j^{(i)} \quad \sigma_j = \sqrt{\frac{1}{n} \sum_{i=1}^n (x_j^{(i)} - \bar{x}_j)^2}. \quad (4.12)$$

Treba napomenuti kako se skaliranje provodi na skupu podataka za učenje pri čemu se dobivaju parametri skaliranja za svaku pojedinu veličinu (npr. u slučaju standardizacije su to \bar{x}_j i σ_j za svaku ulaznu veličinu). Nakon toga se pomoću dobivenih parametara transformiraju ulazne veličine skupa za testiranje kako prikazuje slika 4.3.



Slika 4.3: Skaliranje ulaznih veličina.

4.2.5 Kodiranje vrijednosti kategoričkih veličina

Kako bi se kategoričke veličine mogle koristiti kao ulazne veličine modela, potrebno ih je kodirati u diskrette numeričke veličine. Ako se radi o nominalnim veličinama, tada ne postoji odnos između mogućih vrijednosti i uobičajeno se koristi 1-od-K kodiranje (engl. *one-hot encoding*). Ako nominalna kategorička veličina ima K mogućih vrijednosti, tada se ova veličina kodira s K lažnih binarnih veličina (engl. *dummy variable*) pri čemu jedna lažna varijabla ima vrijednost 1, a sve ostale 0. U tablici ispod dan je primjer kodiranja nominalne kategoričke varijable koja predstavlja tip goriva vozila.

Fuel Type	Var_1	Var_2	Var_3	Var_4	Var_5
Diesel	1	0	0	0	0
Ethanol (E85)	0	1	0	0	0
Natural gas	0	0	1	0	0
Regular gasoline	0	0	0	1	0
Premium gasoline	0	0	0	0	1

Ako se radi o ordinalnim veličinama, tada se mora samostalno definirati mapiranje između kategoričkih vrijednosti i numeričkih vrijednosti kako bi se zadržao odnos koji postoji između kategoričkih vrijednosti.

4.2.6 Vrednovanje modela

Vrednovanje modela provodi se na skupu podataka za testiranje kako bi se procijenile predikcijske sposobnosti odnosno sposobnosti generalizacije izgrađenog modela. Neka ovaj skup podataka ima n uzoraka pri čemu je $y^{(i)}$ stvarna vrijednost izlazne veličine za i -ti uzorak ulaznih veličina, a $\hat{y}^{(i)}$ procjena izlazne veličine dobivena modelom. Postoje različite metrike za evaluaciju regresijskih modela, a najčešće su:

1. Srednja kvadratna pogreška (engl. *mean squared error* - MSE):

$$MSE = \frac{1}{n} \sum_{i=1}^n (y^{(i)} - \hat{y}^{(i)})^2$$

Naravno, modeli s dobrim generalizacijskim sposobnostima imaju manji iznos srednje kvadratne pogreške na testnim podacima.

2. Korijen iz srednje kvadratne pogreške (engl. *root mean squared error* - RMSE):

$$RMSE = \sqrt{MSE}$$

3. Srednja absolutna pogreška (engl. *mean absolute error* - MAE):

$$MAE = \frac{1}{n} \sum_{i=1}^n |y^{(i)} - \hat{y}^{(i)}|$$

4. Srednja absolutna postotna pogreška (engl. *mean absolute percentage error* - MAPE):

$$MAPE = \frac{1}{n} \sum_{i=1}^n \frac{|y^{(i)} - \hat{y}^{(i)}|}{\max(\epsilon, |y^{(i)}|)}$$

5. Koeficijent determinacije R^2 pokazuje koliko je varijacija u podacima obuhvaćeno modelom:

$$R^2 = 1 - \frac{\sum_{i=1}^n (y^{(i)} - \hat{y}^{(i)})^2}{\sum_{i=1}^n (y^{(i)} - \bar{y})^2}$$

gdje je $\bar{y} = \frac{1}{n_{test}} \sum_{i=1}^n y^{(i)}$. Najbolji model (savršeno procjenjuje vrijednosti izlazne veličine) za dane podatke ima vrijednost 1. Model oblika $h_\theta(x^{(i)}) = \bar{y}$ ima vrijednost $R^2 = 0$. Modeli koji lošije procjenjuju izlaznu veličinu od ovog konstantnog modela imaju negativan R^2 .

4.3 Scikit-learn biblioteka

Scikit-learn je open source Python biblioteka za strojno učenje koja se temelji na NumPy, SciPy i Matplotlib bibliotekama. Unutar biblioteke je implementiran velik broj metoda strojnog učenja za nadzirano i nenadzirano učenje, odabir modela, vizualizaciju i sl.

Podjela podataka na skup za učenje i skup za testiranje modela

Prilikom rada s bibliotekom, raspoloživi podatkovni skup je potrebno pohraniti u obliku dva numpy polja koja odgovaraju jednadžbi 4.2. Primjer 4.1 demonstrira podjelu podataka na skup podataka za učenje i skup podataka za testiranje u omjeru 80%-20% pomoću `train_test_split` funkcije. Za demonstraciju se koristi podatkovni skup dostupan u okviru scikit-learn biblioteke.

■ Primjer 4.1

```
from sklearn import datasets
from sklearn.model_selection import train_test_split

# ucitaj ugradeni podatkovni skup
X, y = datasets.load_diabetes(return_X_y=True)

# podijeli skup na podatkovni skup za ucenje i podatkovni skup za
# testiranje
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2,
                                                    random_state=1)
```

Skupovi za učenje i testiranje mogu se skalirati pomoću klase za min-max skaliraje (MinMaxScaler) i standardizaciju podataka. Primjer 4.2. predstavlja isječak koda koji ilustrira min-max skaliranje skupova podataka za učenje i testiranje.

■ Primjer 4.2

```
from sklearn.preprocessing import MinMaxScaler

# min-max skaliranje
sc = MinMaxScaler()
X_train_n = sc.fit_transform(X_train)
X_test_n = sc.transform(X_test)
```

Kako bi se kategoričke veličine mogle koristiti kao ulazne veličine modela, potrebno ih je kodirati. U scikit-learn dostupna je klasa OneHotEncoder s kojom je moguće izvršiti kodiranje nominalnih kategoričkih veličina kao što je prikazano u isječku koda u primjeru 4.3.

■ Primjer 4.3

```
from sklearn.preprocessing import OneHotEncoder

ohe = OneHotEncoder()
X_encoded = ohe.fit_transform(data[['Fuel Type']]).toarray()
```

U scikit-learn biblioteci modeli su implementirani u obliku klase koji implementiraju metodu za procjenu parametara modela na temelju danih podataka te metodu za izračunavanje izlaza modela na temelju ulaznih podataka:

- .fit(X,y)
- .predict(X)

Linearni regresijski model implementiran je u obliku klase:

```
class sklearn.linear_model.LinearRegression(*, fit_intercept=True, copy_X=True, n_jobs=None, positive=False)
```

Primjer 4.4. prikazuje isječak koda koji inicijalizira linearni regresijski model te se zatim procjenjuju parametri modela na temelju skupa za učenje.

■ Primjer 4.4

```
import sklearn.linear_model as lm

linearModel = lm.LinearRegression()
linearModel.fit(X_train_n, y_train)
```

U scikit-learn biblioteci dostupne su funkcije za evaluaciju modela. U primjeru 4.5. demonstrirano je kako se izračunava iznos srednje absolutna pogreške na skupu podataka za testiranje. Ostale metrike koje su dostupne unutar biblioteke mogu se vidjeti ovdje.

■ Primjer 4.5

```
from sklearn.metrics import mean_absolute_error

#predikcija izlazne velicine na skupu podataka za testiranje
y_test_p = linearModel.predict(X_test_n)

#evaluacija modela na skupu podataka za testiranje pomocu MAE
MAE = mean_absolute_error(y_test, y_test_p)
```

4.4 Priprema za vježbu

1. Proučite poglavlje 4.2.
2. Po potrebi dodatno proučite dokumentaciju:
 - a. Uvod u strojno učenje sa scikit-learn
 - b. Linearni regresijski model u scikit-learn
 - c. Metrike za evaluaciju regresijskih modela

4.5 Rad na vježbi

1. Isprobajte Python primjere iz poglavlja 4.3 u Visual Studio Code IDE.
2. Riješite dane zadatke.

Zadatak 4.5.1 Skripta `zadatak_1.py` učitava podatkovni skup iz `data_CO2_emission.csv`. Potrebno je izgraditi i vrednovati model koji procjenjuje emisiju CO2 plinova na temelju ostalih numeričkih ulaznih veličina. Detalje oko ovog podatkovnog skupa mogu se pronaći u 3. laboratorijskoj vježbi.

- a) Odaberite željene numeričke veličine specificiranjem liste s nazivima stupaca. Podijelite podatke na skup za učenje i skup za testiranje u omjeru 80%-20%.
- b) Pomoću matplotlib biblioteke i dijagrama raspršenja prikažite ovisnost emisije CO2 plinova o jednoj numeričkoj veličini. Pri tome podatke koji pripadaju skupu za učenje označite plavom bojom, a podatke koji pripadaju skupu za testiranje označite crvenom bojom.
- c) Izvršite standardizaciju ulaznih veličina skupa za učenje. Prikažite histogram vrijednosti jedne ulazne veličine prije i nakon skaliranja. Na temelju dobivenih parametara skaliranja transformirajte ulazne veličine skupa podataka za testiranje.
- d) Izgradite linearni regresijski modeli. Ispišite u terminal dobivene parametre modela i povežite ih s izrazom 4.6.
- e) Izvršite procjenu izlazne veličine na temelju ulaznih veličina skupa za testiranje. Prikažite pomoću dijagrama raspršenja odnos između stvarnih vrijednosti izlazne veličine i procjene dobivene modelom.
- f) Izvršite vrednovanje modela na način da izračunate vrijednosti regresijskih metrika na skupu podataka za testiranje.
- g) Što se događa s vrijednostima evaluacijskih metrika na testnom skupu kada mijenjate broj ulaznih veličina?

Zadatak 4.5.2 Na temelju rješenja prethodnog zadatka izradite model koji koristi i kategoričku varijable „Fuel Type“ kao ulaznu veličinu. Pri tome koristite 1-od-K kodiranje kategoričkih veličina. Radi jednostavnosti nemojte skalirati ulazne veličine. Komentirajte dobivene rezultate. Kolika je maksimalna pogreška u procjeni emisije CO2 plinova u g/km? O kojem se modelu vozila radi?

4.6 Izvještaj s vježbe

Kao izvještaj s vježbe prihvata se web link na repozitorij pod nazivom OSU_LV.