LV5 –VIŠESTRUKA LINEARNA REGRESIJA – kratke smjernice

Koeficijent determinacije je omjer sume kvadrata protumačenog dijela i ukupne sume kvadrata, te poprima vrijednosti na intervalu [0, 1]. Ovaj koeficijent se koristi kao deskriptivno obilježje kojim se opisuje snaga linearnog odnosa izmedu nezavisnih varijabli i zavisne varijable, a može se koristiti za usporedbu modela koji obuhvaćaju jednake uzorke opažanja. Kvalitetni modeli imaju R2 vrijednost blizu 1

Skaliranje podataka

Skaliranje podataka prilagođava vrijednosti varijabli na određeni raspon, obično [0,1] ili [−1,1], ovisno o modelu i potrebi. Skaliranje je korisno za metode osjetljive na razliku u rasponu vrijednosti, poput k-NN, SVM-a, i neuronskih mreža.

Standardizacija podataka može biti korisna, ali nije uvijek nužna. Standardizacija podrazumijeva transformaciju varijabli tako da imaju srednju vrijednost 0 i standardnu devijaciju 1. Ovo je posebno važno kada varijable imaju različite skale, što može utjecati na procjenu koeficijenata u regresijskom modelu.

Kod jednostavne linearne regresije oblika: y=b0 + b1x nije nužna standardizacija varijabli.

Na primjer, ako je cilj utvrditi kako se ušteđevina (y) mijenja s obzirom na dob (x) koristeći linearni model, tada b0 (intercept) odgovara procijenjenoj ušteđevini za osobu s dobi x=0, a b1 (nagib pravca) predstavlja Koliko se ušteđevine povećava (ili smanjuje) za svaku dodatnu godinu dobi.

Interpretacija bez standardizacije – vrijednost x i y ostaju u svojim mjernim jedinicama

Prednost: koeficijenti b0 i b1 imaju stvarno značenje u kontektstu problema I podaci zadržavaju svoje mjerne jedinice

Ograničenja: ako je raspon x širok ilia ko vrijednosti x I y imaju vrlo različite skale može doći do problema s numeirčkom stabilnošću

Kada je bolje standardizirati? Za usporedbu različitih značajki 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). Standardizacija također može pomoći ako se koriste različiti skupovi podakta s vrlo različitim skalama. Ako se koristi metoda optimizacije poput gradijentnog spusta, standardizacija može poboljšati stabilnost i brzinu konverencije algoritma.

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

# Create an instance of the StandardScaler class
scaler = StandardScaler()

# Fit the scaler to the data and transform the data
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 podatak

```
python
from sklearn.preprocessing import MinMaxScaler

# Create an instance of the MinMaxScaler class
scaler = MinMaxScaler()

# Fit the scaler to the data and transform the data
scaled_data = scaler.fit_transform(data)
```

(c) Robusno skaliranje – normalizacija medijana i kvantila, mjeri podatke na temelju robusnih procjena lokacije I razmjera. Ova metoda je korisna kada podaci sadrže ekstremne vrijednosti ili kada distribucija ne prati Gaussovu razdiobu.

```
python
from sklearn.preprocessing import RobustScaler

# Create an instance of the RobustScaler class
scaler = RobustScaler()

# Fit the scaler to the data and transform the data
scaled_data = scaler.fit_transform(data)
```

Dodatno u skraćenoj verziji mžete pronaći na linku.

Pretvaranje kategorijskih značajki u numerički

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

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

# Pretpostavimo da je 'mainroad' kategorijska značajka
label_encoder = LabelEncoder()
df['mainroad'] = label_encoder.fit_transform(df['mainroad'])
```

2. **One-Hot Encoding (Jednostruko kodiranje)**

   Koristi se kod varijabli koje imaju više kategorija. Ova metoda stvara novu binarnu kolonu za svaku kategoriju. Na primjer, za značajku `prefarea` s kategorijama `Urban`, `Suburban` i `Rural`:

   Prednosti: Uklanja problem "redoslijeda" među kategorijama; prikladno za značajke s više kategorija.

   Nedostaci: Može stvoriti veliki broj novih kolona, što povećava dimenzionalnost podataka.

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

```
import pandas as pd

# Primjer podataka

data = {'City': ['London', 'Paris', 'Berlin', 'London',
'Berlin']}

df = pd.DataFrame(data)

print("Originalni podaci:")

print(df)
```

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

```
# Primjena One-Hot Encoding-a
df_encoded = pd.get_dummies(df, columns=['City'],
drop_first=False)

print("\nPodaci nakon One-Hot Encoding-a:")
print(df_encoded)
```

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 vrijednost kategorije `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.

3. Binary_map()

Za pretvaranje binarnih kategorijskih značajki u numeričke (0 i 1) može se koristiti i **Binary_map()** što je brži i jednostavniji način.

- **Jednostavnost:**
    - Ako je kategorijska značajka već binarna (npr., `yes/no`, `true/false`, `male/female`), `binary_map()` izravno mapira vrijednosti na brojeve:
        - `Yes` → 1, `No` → 0
        - `True` → 1, `False` → 0
        - `Male` → 1, `Female` → 0
- **Brzina i efikasnost:**
    - Kod binarnih značajki nije potrebno stvarati nove stupce kao kod **One-Hot Encoding**.
    - Radi direktno na postojećim podacima, smanjujući memorijsku potrošnju.
- **Lakša interpretacija:**
    - Rezultat je odmah čitljiv: 1 predstavlja jednu kategoriju, a 0 drugu.

- Najčešće se implementira kao prilagođena funkcija (custom function) koja koristi Pythonove metode poput `map()` ili `replace()`.

- **Primjer s `binary_map()`**

  Pretpostavimo da imate značajku `mainroad` s vrijednostima `yes` i `no`. Funkcija `binary_map()` mogla bi izgledati ovako:

```
def binary_map(feature):

    return feature.map({'yes': 1, 'no': 0})

# Primjena na podatke

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

- Ako ne želite definirati zasebnu funkciju, možete direktno koristiti metodu `replace()`

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

Kada ne koristiti druge metode?

- **One-Hot Encoding:** nepotrebno za binarne značajke jer dodaje višak dimenzija (stvara dva stupca).

- **Label Encoding:** može biti previše općenit i dodati redoslijed koji nije potreban za jednostavne binarne značajke.

Multikolinearnost

Variance Inflation Factor (VIF) je mjera koja se koristi u regresijskoj analizi kako bi se otkrila i kvantificirala **multikolinearnost** među neovisnim varijablama.

Što je multikolinearnost (engl. Multicollinearity)?

- **Multikolinearnost** nastaje kada su dvije ili više neovisnih varijabli visoko međusobno povezane.
- To može uzrokovati probleme u linearnim modelima jer:
  - Teško je odvojiti učinak svake varijable na ciljnu varijablu.
  - Koeficijenti postaju nestabilni i netočni.
  - Model može biti osjetljiv na male promjene u podacima.

Što je VIF?

- **Variance Inflation Factor** mjeri koliko je varijanca koeficijenta određene varijable uvećana zbog multikolinearnosti s drugim varijablama.

Formula za VIF:

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

Gdje je:

- R_i^2 = koeficijent determinacije kada se varijabla Xi regresira na sve ostale neovisne varijable.
- R2 je koeficijent determinacije koji mjeri koliko dobro nezavisne varijable (X2,X3) objašnjavaju varijaciju u zavisnoj varijabli (X1).

Interpretacija VIF-a:

- **VIF = 1**: Nema multikolinearnosti (idealno).
- **VIF > 5**: Visoka multikolinearnost (treba pažljivo razmotriti uklanjanje te varijable).
- **VIF > 10**: Ekstremna multikolinearnost (varijablu treba ukloniti).

Zaljučak:
Analizirati odnos između zavisne varijable (Xi) i svih ostalih nezavisnih varijabli (X1,X2,…, osim Xi).

- Dobiti R2 kao mjeru korelacije između zavisne varijable i prediktora.
- Na temelju R2 izračunati VIF i otkriti redundantne varijable.

Pogledati više na linku