vjezbe-01-dio-2-rjesenja

March 9, 2022

1 Vježbe 1 - drugi dio

[ ]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.datasets import load_wine
from sklearn import preprocessing
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
import seaborn as sns
```

1.1 Linearna regresija

1. Vaš zadatak za zadaću: implementacija modela linearne regresije
2. Primjena tog modela na podatke uz dodarno korištenje različitih alata s kojima smo se upoznali

1.1.1 Zadatak 1. Data exploatory analysis

Model linearne regresije primjenit ćemo na jednom skupu podataka. Prije toga, potrebno je učitati podatke, upoznati ih i analizirati.

1. Učitavanje podataka

[ ]:
```
df = pd.read_csv('./Podaci/winequality-white.csv')
df.head()
```

2. Informacije o skupu podataka Broj podataka, broj varijabli, tip podataka varijabli

[ ]:
```
df.info()
```

[ ]:
```
df.columns, df.dtypes
```

3. Analiza svake varijable posebno Ako je varijabla numerička - dobijemo relevantne kvantitativne mjere poput aritmetičke sredine, medijana i sl.

Varijabla *quality* unatoč tome što je numrička zapravo predstavlja ocjenu vina, a to je opet neka kategorija koja poprima vrijednosti $1-10$. Kod takvih varijabli je upitno koliko će nam ova analiza

imati smisla. Ali možemo saznati neke stvari poput toga da je najmanja ocjena 3, a najveća ocjena 9.

[ ]:
```
df['alcohol'].describe()
```

[ ]:
```
df['quality'].describe()
```

[ ]:
```
df['type'].describe()
```

4. Analiza svake varijable posebno - grafički Prethodno izračunate kvantitativne mjere je često *zanimljivije* vidjeti na grafu - histogram

[ ]:
```
sns.histplot(df['alcohol'])
```

[ ]:
```
# sns.histplot(df, x="quality")
sns.histplot(df['quality'])
```

[ ]:
```
sns.displot(df, x="alcohol", hue="quality", multiple="stack")
```

[ ]:
```
sns.displot(df, x="alcohol", hue="type", multiple="stack")
```

5. Veza između numeričkih varijabli Postoji li veza između količine alkohola u vinu i gustoće? Količine kiseline i ph-a? Za neke varijable očekujemo da su povezane.

Grafički koristeći *scatter_plot* možemo to ispitati.

[ ]:
```
sns.scatterplot(data=df, x="alcohol", y="density")
```

[ ]:
```
sns.scatterplot(data=df, x="residual sugar", y="free sulfur dioxide",␣
↪hue="type")
```

[ ]:
```
sns.scatterplot(data=df, x="alcohol", y="density", hue="type")
```

[ ]:
```
sns.scatterplot(data=df, x="pH", y="volatile acidity", hue="type")
```

[ ]:
```
sns.scatterplot(data=df, x="pH", y="fixed acidity")
```

6. Histogrami - usporedba varijabli (kategorija)

[ ]:
```
grid = sns.FacetGrid(df, row="type", col="quality", margin_titles=True)
grid.map(plt.hist, "alcohol", bins=np.linspace(0, 40, 15));
```

7. Boxplot

[ ]:
```
with sns.axes_style(style='ticks'):
    g = sns.catplot(data=df, x="quality", y="residual sugar", hue="type",␣
↪kind="box")
    g.set_axis_labels("Quality", "Residual sugar")
```

```
[ ]: with sns.axes_style(style='ticks'):
         g = sns.boxplot(x="type", y="residual sugar", data=df)
```

```
[ ]: sns.histplot(df[df["type"]=="white"],x="residual sugar")
```

```
[ ]: sns.histplot(df[df["type"]=="red"],x="residual sugar")
```

```
[ ]: sns.displot(df, x="residual sugar", col='type', bins=50)
```

8. Koreliranost podataka Naredba dataframe.corr() računa Pearsonov koeficijent korelacije između varijabli. Prisjetimo se, za zadani par slučajnih varijabli (X, Y) Pearsonov koeficijent korelacije definiramo kao

$$\rho_{x,y} = \frac{cov(X,Y)}{\sigma_X \cdot \sigma_Y},$$

gdje je $cov(X,Y)$ kovarijanca $= E[(X - E[X]) \cdot (Y - E[Y])]$, a σ_X je standardna devijacije slučajne varijable X, $\sigma_X = \sqrt{(Var(X))}$, $Var(X) = E(X - EX)^2$. - Ako su X, Y nezavisne, onda je Pearsonov koeficijent $\rho_{X,Y} = 0$ - ako je $Y = aX + b$ za neki koeficijent $a > 0$ i neki b onda je $\rho_{X,Y} = 1$ - ako je $Y = aX + b$ za neki koeficijent $a < 0$ i neki b onda je $\rho_{X,Y} = -1$

```
[ ]: df.corr()
```

```
[ ]: sns.heatmap(df.corr())
```

1.2 Zadatak 2: Priprema podataka

1. Uočili smo snažnu koreliranost između varijebli *density* i *alcohol*. Možemo li postaviti model linearne regresije koji ćemo naučiti da za zadanu vrijednost alkohola procijeni kolika je gustoća vina? Prvo nam trebaju podaci u toj formi.

Ulazna varijabla X je količina alkohola, a izlazna varijabla y je gustoća.

```
[ ]: X = df["alcohol"]
     y = df["density"]
```

2. Priprema podataka za treniranje: podjela, obrada - uobičajeno je podatke s kojima radimo podijeliti na 2 ili 3 dijela - kasnije nešto više o tome - za sada koristimo kao činjenicu da podatke želimo podijeliti na 2 dijela - jedan skup podataka (veći) nazvat ćemo trening skup, a drugi skup (manji) testni skup - podjela podataka može biti u omjeru npr. trening 70% - test 30% - to možemo napraviti ručno: odabrati koje točno podatke gdje želimo ili nasumično staviti neke podatke - osim toga možemo koristiti neke funkcije koje to rade - najjednostavnije je korištenje gotovih funkcija iz scikit-learna - https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.scale.html

```
[ ]: # rucni odabir
     X_train, y_train = X[:4500], y[:4500]
     X_test, y_test = X[4500:], y[4500:]
     print(X_train.shape, X_test.shape)

     # train_test_split funkcija
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3,␣
 ↪random_state=42)
print(X_train.shape, y_train.shape, X_test.shape, y_test.shape)

# algoritmu ucenja dajemo podatke spremljene u numpy array
print(type(X_train))
X_train = X_train.to_numpy().reshape(-1,1)
y_train = y_train.to_numpy().reshape(-1,1)
X_test = X_test.to_numpy().reshape(-1,1)
y_test = y_test.to_numpy().reshape(-1,1)
print(type(X_train))
```

[ ]:
```
scaler = preprocessing.StandardScaler()
scaler.fit(X_train, y_train)

X_train = scaler.transform(X_train)
X_test = scaler.fit_transform(X_test)
```

1.3 Zadatak 3: Upoznavanje modela linearne regresije iz scikit-learna

Model linearne regresije je implementiran kao klasa. Prilikom instanciranja možemo postaviti sljedeće parametre: - fit_intercept - bool - zadano je True - određuje treba li računati pomak ili ne - normalize: - bool - zadano je False - ako je fit_intercept False, ovaj parametar se ignorira - ako je True, podaci su normalizirani tako da se oduzme aritmetička sredina i podijeli sa L2 normom

Nakon što smo postavili model imamo dostupne sljedeće metode - fit(X, y) treniramo model - predict(X) vraća predikciju za neki podatak na temelju natreniranih težina - score(X,y) računa predikcije i uspoređuje sa stvarnim vrijednostima te vraća koeficijent (više na https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LinearRegression.html#sklearn.linear_model.LinearRe

I atribute: - coef_ - intercept_

1. Postavljanje i učenje modela

[ ]:
```
# Klasa
# LinearRegression(*, fit_intercept=True, normalize=False, copy_X=True,␣
 ↪n_jobs=None)
lr = LinearRegression(fit_intercept=True)
lr.fit(X_train, y_train)
lr.coef_, lr.intercept_
```

2. Testiranje modela

[ ]:
```
loss = 0
for i, prediction in enumerate(lr.predict(X_test)):
    loss += (prediction -  y_test[i])**2
print(loss / (2 * X_test.shape[0]))
```

[ ]:
```
# Razlikuje se od pogreske koju mi racunamo
# Dokumentacija: Return the coefficient of determination
```

```
lr.score(X_test,y_test)
```

3. Grafički prikaz rezultata modela (Za vježbu)

[ ]:
```
## Vaš kod ovdje
```