

SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I
INFORMACIJSKIH TEHNOLOGIJA

Procjena cijene rabljenog automobila

Seminarski rad

Raspoznavanje uzoraka i strojno učenje

Igor Delić

Osijek, 2023.

Sadržaj

1. UVOD.....	2
2. NADZIRANO UČENJE	3
2.1. Regresija.....	4
3. ANALIZA PODATAKA	5
4. REZULTATI	14
4.1. Linearnu regresija	14
4.2. Random Forest Regressor	15
4.3. Stablo odlučivanja	16
4.4. K-najbližih susjeda.....	17
5. STREAMLIT APLIKACIJA	18
6. ZAKLJUČAK	22

1. UVOD

Procjena cijene automobila važan je zadatak u automobilske industriji i tržištu rabljenih vozila. Zbog raznolikosti karakteristika i stanja automobila, procjena cijene može biti složen proces koji zahtijeva visoku razinu stručnosti i iskustva. Srećom, danas postoji značajna količina podataka o prodaji rabljenih vozila, što olakšava izgradnju modela koji mogu procijeniti vrijednost automobila na temelju njihovih karakteristika. Stoga se strojno učenje i umjetna inteligencija sve više koriste u ovom području kako bi se poboljšale preciznost i učinkovitost u procjeni cijena automobila.

U ovom seminaru koristit ćemo strojno učenje za procjenu cijene automobila. Proučit ćemo ključne značajke koje utječu na cijenu automobila i kako se te značajke koriste za izgradnju modela. Također ćemo pregledati različite algoritme strojnog učenja koji se koriste za regresijsku analizu i kako se primjenjuju na skupu podataka automobila.

2. NADZIRANO UČENJE

Nadzirano učenje je vrsta strojnog učenja u kojoj se model uči na temelju ulaza i pripadnih izlaza ili oznaka. Cilj je naučiti model koji može predviđati izlaz za nove ulaze koji mu nisu bili prethodno poznati. U nadziranom učenju, skup podataka koji se koristi za treniranje modela obično se sastoji od ulaznih vrijednosti i odgovarajućih oznaka ili izlaza, a model koristi te podatke za učenje. Na primjer, u zadatku klasifikacije slika, model bi se trenirao na skupu podataka slika i pripadnih oznaka koje ih klasificiraju u određene klase. Ključna prednost nadziranog učenja je da modeli mogu biti vrlo precizni i da se mogu koristiti za predviđanje novih vrijednosti koje nisu bile prisutne u skupu podataka za treniranje. Međutim, to također znači da ovisimo o kvaliteti i raznolikosti podataka za treniranje, te se problemi mogu javiti ako skup podataka nije dovoljno velik ili reprezentativan za svrhu učenja. Nadzirano učenje koristi se u mnogim područjima, uključujući računalni vid, obradu prirodnog jezika, prepoznavanje govora, medicinsku dijagnostiku, financije i mnoge druge industrije i grane znanosti.

2.1. Regresija

Regresija je statistička metoda koja se koristi za predviđanje numeričkih vrijednosti (kontinuiranih varijabli) između dvije ili više varijabli. Cilj regresijske analize je pronalaženje matematičke funkcije koja najbolje odgovara odnosu između nezavisne varijable (ulaza) i zavisne varijable (izlaza). Ova funkcija se zatim može koristiti za predviđanje izlaza na temelju novih ulaznih vrijednosti. Regresija se koristi u mnogim područjima kao što su ekonomija, poslovna analiza, medicina, inženjerstvo, znanost o podacima i strojno učenje. Primjeri regresije uključuju predviđanje cijene nekretnina, prognoziranje prodaje proizvoda, predviđanje potrošnje energije, predviđanje broja prodanih proizvoda i tako dalje. Postoji mnogo algoritama koji se koriste u regresiji, a neki od najčešćih su linearna regresija, regresija drveća odluke, k-NN regresija, random forest regresija i gradient boosting regresija. Odabir algoritma ovisi o prirodi podataka i vrsti problema koji se rješava.

3. ANALIZA PODATAKA

Podaci za strojno učenje su skup podataka koji se koriste za treniranje, testiranje i evaluaciju modela strojnog učenja. Podaci mogu biti strukturirani ili nestrukturirani. Strukturirani podaci su podaci koji se nalaze u tabličnom formatu i lako se mogu obrađivati pomoću matematičkih funkcija. Ovi podaci se najčešće koriste u strojnom učenju za klasifikaciju i regresiju. Nestrukturirani podaci su podaci koji nemaju jasnu strukturu i teško se obrađuju.

Za potrebu izrade projektnog zadatka korišten je Dataset “car_price_prediction” spremljen kao CSV datoteka s atributima:

ID, Price, Levy, Manufacturer, Model, Prod. Year, Category, Leather interior, Fuel type, Engine volume, Mileage, Cylinders, Gear box type, Drive wheels, Doors, Wheel, Color i Airbags

```
df=pd.read_csv("/content/car_price_prediction.csv")
df.head()
```

Slika 3.1. Kod za učitavanje podataka

	ID	Price	Levy	Manufacturer	Model	Prod. year	Category	Leather interior	Fuel type	Engine volume	Mileage	Cylinders	Gear box type	Drive wheels	Doors	Wheel	Color	Airbags
0	45654403	13328	1399	LEXUS	RX 450	2010	Jeep	Yes	Hybrid	3.5	186005 km	6.0	Automatic	4x4	04-May	Left wheel	Silver	12
1	44731507	16621	1018	CHEVROLET	Equinox	2011	Jeep	No	Petrol	3	192000 km	6.0	Tiptronic	4x4	04-May	Left wheel	Black	8
2	45774419	8467	-	HONDA	FIT	2006	Hatchback	No	Petrol	1.3	200000 km	4.0	Variator	Front	04-May	Right-hand drive	Black	2
3	45769185	3607	862	FORD	Escape	2011	Jeep	Yes	Hybrid	2.5	168966 km	4.0	Automatic	4x4	04-May	Left wheel	White	0
4	45809263	11726	446	HONDA	FIT	2014	Hatchback	Yes	Petrol	1.3	91901 km	4.0	Automatic	Front	04-May	Left wheel	Silver	4

Slika 3.2. Prikaz prvih 5 učitanih podataka

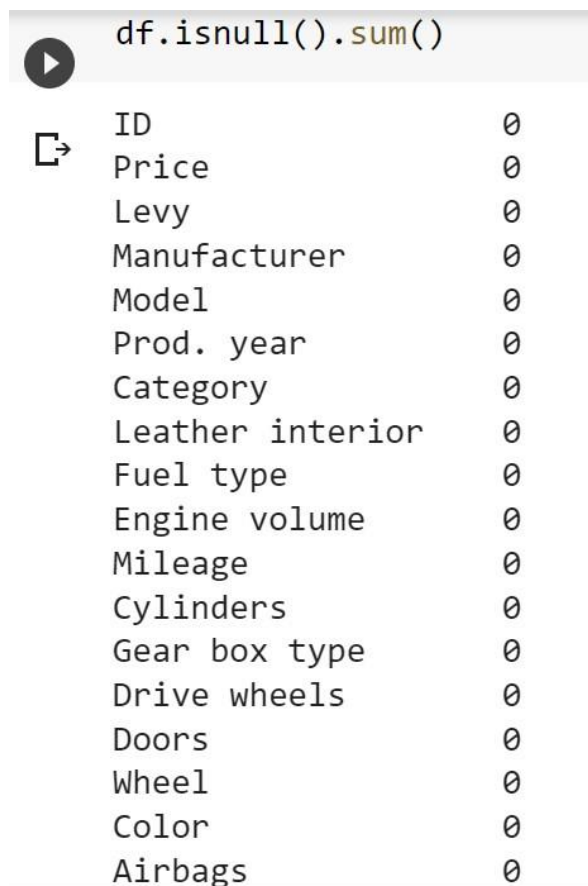


`df.info()`

```
>>> <class 'pandas.core.frame.DataFrame'>
RangeIndex: 19237 entries, 0 to 19236
Data columns (total 18 columns):
#   Column                Non-Null Count  Dtype
---  -
0   ID                    19237 non-null  int64
1   Price                 19237 non-null  int64
2   Levy                  19237 non-null  object
3   Manufacturer          19237 non-null  object
4   Model                 19237 non-null  object
5   Prod. year           19237 non-null  int64
6   Category              19237 non-null  object
7   Leather interior      19237 non-null  object
8   Fuel type             19237 non-null  object
9   Engine volume         19237 non-null  object
10  Mileage               19237 non-null  object
11  Cylinders             19237 non-null  float64
12  Gear box type         19237 non-null  object
13  Drive wheels          19237 non-null  object
14  Doors                 19237 non-null  object
15  Wheel                 19237 non-null  object
16  Color                 19237 non-null  object
17  Airbags               19237 non-null  int64
dtypes: float64(1), int64(4), object(13)
```

Slika 3.3. Prikaz općih informacija

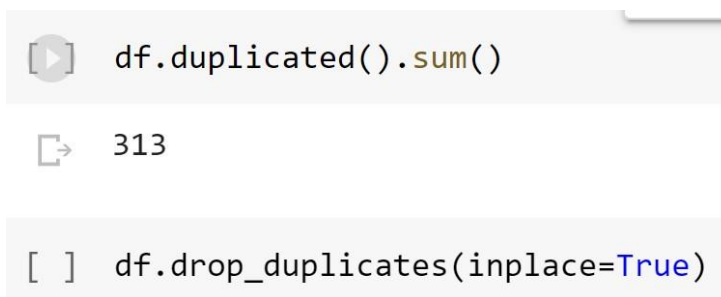
Porebno je odraditi provjeri nalazi li se u našim podacima neka null vrijednost i ako postoji, probrojati oliko ih ima. (Slika 3.4.)



▶	<code>df.isnull().sum()</code>
↳	ID 0
	Price 0
	Levy 0
	Manufacturer 0
	Model 0
	Prod. year 0
	Category 0
	Leather interior 0
	Fuel type 0
	Engine volume 0
	Mileage 0
	Cylinders 0
	Gear box type 0
	Drive wheels 0
	Doors 0
	Wheel 0
	Color 0
	Airbags 0

Slika 3.4.

Također je potrebno provjeriti nalaze li u našim podacima duplikati te ih je potrebno ukloniti.



```
▶ df.duplicated().sum()

↳ 313

[ ] df.drop_duplicates(inplace=True)
```

Slika 3.5.


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

HYUNDAI	3729
TOYOTA	3606
MERCEDES-BENZ	2043
FORD	1088
CHEVROLET	1047
...	
TESLA	1
PONTIAC	1
SATURN	1
ASTON MARTIN	1
GREATWALL	1

Slika 3.6. Proizvođači automobila

```
df['Category'].value_counts()
```

Sedan	8600
Jeep	5378
Hatchback	2799
Minivan	633
Coupe	528
Universal	361
Microbus	299
Goods wagon	229
Pickup	51
Cabriolet	35
Limousine	11

Slika 3.7. Kategorije automobila

```
df['Color'].value_counts()
```

Black	4944
White	4407
Silver	3729
Grey	2343
Blue	1376
Red	622
Green	321
Orange	252
Brown	185
Carnelian red	177
Golden	143
Beige	134
Sky blue	122
Yellow	105
Purple	39
Pink	25

Slika 3.8. Boje automobila

```
df['Gear box type'].value_counts()
```

Automatic	13282
Tiptronic	3065
Manual	1844
Variator	733

Name: Gear box type, dtype: int64

```
df['Drive wheels'].value_counts()
```

Front	12695
4x4	3969
Rear	2260

Slika 3.9. Vrsta mjenjača i pogona

Kako bi pripremili podatke za daljnju obradu bilo potrebno je napraviti neke izmjene. Uklonjen je stupac 'ID' jer nam je ID automobila nepotreban za analizu. Zamijenjeni su nedostajući podaci u stupcu 'Levy' s NaN i konvertiran je u tip float, tako da se njime može lakše raditi. U stupcu 'Engine volume' obrisani su svi stringovi 'Turbo' iz podataka te je konvertiran u tip float. U stupcu 'Mileage' obrisani su svi stringovi 'km' iz podataka te je konvertiran u tip int. U stupcu 'Doors', stringovi su zamijenjeni odgovarajućim brojevima, kako bi se omogućila daljnja obrada u modelu strojnog učenja.(Slika 3.10.)

```
df.drop('ID',axis=1,inplace=True)

df['Levy'].replace({'-':np.nan}, inplace = True)

df['Levy'] = df['Levy'].astype('float64')

df['Engine volume']=df['Engine volume'].apply(lambda x: x.replace('Turbo',''))
df['Engine volume']=df['Engine volume'].astype(dtype='float')

df['Mileage']=df['Mileage'].apply(lambda x: x.replace('km',''))
df['Mileage']=df['Mileage'].astype(dtype='int')

df['Doors']=df['Doors'].replace({'04-May':4, '02-Mar':2, '>5':5})
```

Slika 3.10. Prikaz opisanog koda

Slike prikazuju dio koda koji se koristi za identificiranje stupaca koji najviše utječu na cijenu automobila na temelju apsolutne vrijednosti korelacijskih koeficijenata. (Slika 3.11.)

```
df.corrwith(df['Price']).abs().sort_values(ascending=False)
```

Price	1.000000
Levy	0.065458
Doors	0.033387
Airbags	0.012709
Prod. year	0.012689
Engine volume	0.008888
Cylinders	0.007435
Mileage	0.001763

Slika 3.11.

```
df.corrwith(df['Price']).sort_values(ascending=False)
```

Price	1.000000
Prod. year	0.301996
Engine volume	0.136050
Cylinders	0.103306
Levy	0.081976
Doors	0.007072
Airbags	-0.024845
Mileage	-0.208223

Slika 3.12.

Nadalje podaci su podijeljeni u kategorije temeljem tipa podataka u svakom stupcu. Varijabla x sadrži sve podatke iz skupa podataka, osim prvog stupca koji je ciljna varijabla, dok je y ta ciljna varijabla. Stvaraju se dvije liste X_cats i X_nums koje se popunjavaju stvarnim vrijednostima podataka iz skupa podataka X koji odgovaraju imenima stupaca u odgovarajućim listama. X_cats sadrži sve kategoričke varijable, dok X_nums sadrži sve numeričke varijable.(Slika 3.13.)

```
X=df[df.columns[1:]]
y=df[df.columns[0]].values
X_cats=[]
X_nums=[]

for col in X.columns:
    if X[col].dtypes=='object':
        X_cats.append(col)
    else:
        X_nums.append(col)

X_nums=df[X_nums]
X_cats=df[X_cats]
```

Slika 3.13.

Prethodno obrađeni numerički i kategorički podaci kombiniraju se kako bi se stvorila konačna matrica značajki X. Num_pipeline je objekt Pipeline koji spaja dva transformatora: SimpleImputer i StandardScaler. (Slika 3.14.) Transformator SimpleImputer se koristi za zamjenu nedostajućih vrijednosti s medijanom respektivne značajke, a transformator StandardScaler koristi se za standardizaciju vrijednosti značajki oduzimanjem srednje vrijednosti i dijeljenjem s vrijednosti standardne devijacije. Nakon primjene num_pipeline na numeričke značajke u X_nums, rezultirajući niz se kombinira s one-hot enkodiranim kategoričkim značajkama u X_cats pomoću funkcije concat. Zatim se cijela matrica značajki X transformira pomoću num_pipeline. Konačno, funkcija train_test_split se koristi za podjelu podataka na skupove za učenje i testiranje, pri čemu se 20% podataka rezervira za testiranje. (Slika 3.15.)

```
num_pipeline=Pipeline([
    ('imputer',SimpleImputer(strategy='median')),
    ('std_scaler',StandardScaler())
])
```

Slika 3.14.

```
X=pd.concat([X_nums,X_cats],axis=1)
X=num_pipeline.fit_transform(X)
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.2,random_state=42)
```

Slika 3.15.

4. REZULTATI

U ovom poglavlju ćemo proći kroz rezultate učenja i vidjeti koji model je najuspješniji.

4.1. Linearnu regresija

Linearna regresija je jedna od najosnovnijih tehnika regresijskog strojnog učenja. Koristi se za modeliranje linearnih veza između ulaznih varijabli i izlazne varijable na temelju podataka za učenje. Cilj je pronaći linearnu funkciju koja najbolje opisuje odnos između ulaza i izlaza. Linearna regresija pokušava pronaći optimalne koeficijente koji minimiziraju kvadratnu razliku između predviđene vrijednosti i stvarne vrijednosti.

```
LR_model=LinearRegression()  
LR_model.fit(X_train,y_train)  
predict=LR_model.predict(X_test)  
  
print("Linear Regression")  
print(f"MSE:      ",np.sqrt(mean_squared_error(y_test,predict)))  
print(f"MAE:      ",mean_absolute_error(y_test,predict))  
print(f"R2 score:  ", r2_score(y_test,predict))  
  
Linear Regression  
MSE:      16379.067622103348  
MAE:      10367.885851579967  
R2 score: 0.29364881409135246
```

Slika 4.1.

Prvo se model trenira na podacima za obuku pomoću `LinearRegression()` funkcije, a zatim se model primjenjuje na testne podatke i predviđanja se uspoređuju s stvarnim vrijednostima. Imamo tri različite metrike za evaluaciju performansi regresijskog modela: Mean Squared Error (MSE), Mean Absolute Error (MAE) i R-squared (R2) vrijednosti. MAE je prosječna apsolutna vrijednost razlike između stvarnih vrijednosti i predviđenih vrijednosti, a MSE je prosječna kvadratna razlika između stvarnih vrijednosti i predviđenih vrijednosti. R2 je vrijednost koja predstavlja omjer varijabilnosti u predviđenim vrijednostima i stvarnim vrijednostima. R2 vrijednost je u rasponu od 0 do 1, gdje 1 označava savršeno prilagođeni model. (Slika 4.1.)

4.2. Random Forest Regressor

Random Forest Regressor utemeljen je na ansamblu stabala odlučivanja (decision trees) za zadatke regresije. Stvara više stabala odlučivanja na temelju različitih podskupova podataka za treniranje. Svako stablo se trenira na podskupu podataka te svako stablo daje svoju predikciju, a krajnja predikcija je prosjek svih predikcija pojedinačnih stabala. Popularan je zbog svoje jednostavnosti i fleksibilnosti, te dobre sposobnosti generalizacije i izbjegavanja problema overfittinga. Također, može rukovati velikim brojem ulaznih značajki i podataka te se može koristiti za rješavanje različitih problema regresije.

```
RF_model=RandomForestRegressor()  
RF_model.fit(X_train,y_train)  
predict=RF_model.predict(X_test)  
print('RandomForestRegressor')  
print(f"MSE:      ",np.sqrt(mean_squared_error(y_test,predict)))  
print(f"MAE:      ",mean_absolute_error(y_test,predict))  
print("R2 score: ", r2_score(y_test,predict))
```

```
RandomForestRegressor  
MSE:      9659.477320381407  
MAE:      4675.653969788997  
R2 score: 0.7543314395100621
```

Slika 4.2.

Stvaramo instancu modela regresije slučajnih šuma, koji se zatim trenira na `X_train` i `y_train` podacima pomoću metode `fit`. Nakon treniranja, model se koristi za predviđanje vrijednosti na `X_test` podacima pomoću metode `predict`, čije se vrijednosti uspoređuju s pravim vrijednostima `y_test` pomoću nekoliko metrika. Mean Squared Error (MSE), Mean Absolute Error (MAE) i R-squared (R2) vrijednosti. (Slika 4.2.)

4.3. Stablo odlučivanja

Stabla odlučivanja se koriste za predviđanje kontinuirane vrijednosti. Temelji se na konstrukciji stabla odlučivanja gdje svaki unutarnji čvor predstavlja testiranje na određenoj značajci, dok su listovi predviđanja vrijednosti. Tijekom predviđanja, algoritam prolazi kroz stablo odlučivanja s pitanjima koja testiraju značajke i na kraju dolazi do jednog od listova koji predstavlja predviđanu vrijednost. Ovaj algoritam ima tendenciju overfittinga, što znači da se može prilagoditi podacima za učenje vrlo precizno, ali može imati lošiju točnost na novim podacima.

```
DT_model=DecisionTreeRegressor()  
DT_model.fit(X_train,y_train)  
predict=DT_model.predict(X_test)  
print('DecisionTreeRegressor')  
print(f"MSE:      ",np.sqrt(mean_squared_error(y_test,predict)))  
print(f"MAE:      ",mean_absolute_error(y_test,predict))  
print("R2 score:  ", r2_score(y_test,predict))
```

```
DecisionTreeRegressor  
MSE:      13065.743696990658  
MAE:      5856.890915040004  
R2 score: 0.5505197682996765
```

Slika 4.3.

Na temelju zadanih podataka, gradi se stablo odlučivanja koje se sastoji od niza pitanja koja su postavljena kako bi se donijela konačna odluka. Stablo je izgrađeno na temelju atributa koji su najkorisniji u predviđanju tražene vrijednosti, a razgranavanje stabla ovisi o odgovorima na postavljena pitanja. Zatim, kada se stablo izgradi, podaci se mogu unijeti u stablo da bi se dobila predviđena vrijednost za traženu varijablu. Izvještaj prikazuje srednju kvadratnu pogrešku, apsolutnu pogrešku te koeficijent determinacije.(Slika 4.3.)

4.4. K-najbližih susjeda

KNeighborsRegressor je model regresije zasnovan na metodi k-najbližih susjeda (k-nearest neighbors). Ovaj model predviđa vrijednost za neku varijablu cilja (target variable) tako što traži k najbližih primjera (eng. instances) u skupu podataka i uzima prosječnu vrijednost ciljne varijable tih primjera kao predviđenu vrijednost za novi primjer. Metrika udaljenosti (eng. distance metric) se koristi za određivanje koje primjere uzeti u obzir kao k-najbliže primjere. U ovisnosti o odabranoj metrici i broju susjeda, model može biti osjetljiv na podatke i moguće imati tendenciju preprilagođavanja (eng. overfitting) na skup podataka za učenje. (Slika 4.4.)

```
KN_model=KNeighborsRegressor()
KN_model.fit(X_train,y_train)
predict=KN_model.predict(X_test)
print('KN_model')
print(f"MSE:      " ,np.sqrt(mean_squared_error(y_test,predict)))
print(f"MAE:      " ,mean_absolute_error(y_test,predict))
print("R2 score:  ",  r2_score(y_test,predict))
```

```
KN_model
MSE:      13003.013308026933
MAE:      6473.664716930507
R2 score:  0.5548254372092938
```

Slika 4.4.

5. STREAMLIT APLIKACIJA

Streamlit je Python biblioteka koja omogućuje jednostavno i brzo stvaranje web aplikacija za strojno učenje, vizualizaciju podataka i prototipiranje. Ova biblioteka nam omogućuje da u nekoliko linija koda izgradimo aplikaciju koja korisniku omogućuje interakciju s treniranim modelom. Kako bi smo koristili svoj trenirani model u streamlit aplikaciji, prvo moramo učitati model pomoću joblib biblioteke. Nakon toga, definiramo funkciju predict koja prima unos podataka i koristi prethodno učitani model za predviđanje izlaza za te podatke. Nakon što smo definirali funkciju, možemo koristiti streamlit komponente (npr. text_input, button, selectbox itd.) kako bismo omogućili korisniku interakciju s aplikacijom. Napravljena je streamlit aplikacija koja predviđa cijenu automobila na temelju nekoliko značajki. Nakon učitavanja modela, korisniku prikazujemo ulazne okvire (input-e) u koje može upisati nekoliko značajki kao što su godina proizvodnje, prijeđeni kilometri, broj vrata i sl. Nakon što korisnik unese podatke, može kliknuti na gumb za predviđanje (button), a aplikacija će koristiti funkciju predict za predviđanje cijene automobila na temelju unešenih podataka.

The screenshot shows a web application interface for predicting car prices. It contains three input sections: a numeric input for 'How much is your car worth?' with a value of 250000.00, a numeric input for 'Enter a Levy' with a value of 800.00, and a dropdown menu for 'Select a Manufacturer' with 'HYUNDAI' selected. Below these, a list of car models is displayed, including Jeep, Hatchback, Sedan, Microbus, Goods wagon, Universal, Coupe, and Minivan. The 'Jeep' model is currently selected in the dropdown.

How much is your car worth?

250000.00

You entered: 250000.0

Enter a Levy

800.00

You entered: 800.0

Select a Manufacturer

HYUNDAI

You selected: HYUNDAI

Jeep

Hatchback

Sedan

Microbus

Goods wagon

Universal

Coupe

Minivan

Jeep

You selected: Jeep

Slika 5.1. Streamlit aplikacija

Does it have leather interior?

Yes

You selected: Yes

Fuel type used?

Diesel

You selected: Diesel

Engine volume

3.5

You selected: 3.5

Enter car mileage

190000,00 - +

You entered: 190000.0

Cylinders

2.0

You selected: 2.0

Gear box type

Automatic

You selected: Automatic

Drive wheels

4x4

You selected: 4x4

Slika 5.2. Streamlit aplikacija

Drive wheels

4x4

4x4

Front

Rear

You selected: 04-May

Wheel

Left wheel

You selected: Left wheel

Color

Black

You selected: Black

Airbags

12

You selected: 12

You have entered good price value for your car

Slika 5.3. Streamlit aplikacija

```

model = joblib.load('/content/final.sav')

def app():
    data = pd.read_csv('/content/car_price_prediction.csv')
    value0 = st.number_input("How much is your car worth?")
    st.write('You entered:', value0)

    value1 = st.number_input("Enter a Levy")
    st.write('You entered:', value1)

    column_values2 = data['Manufacturer'].unique()
    value2 = st.selectbox('Select a Manufacturer', column_values2)
    st.write('You selected:', value2)

    column_values3 = data['Model'].unique()
    value3 = st.selectbox('Select a value', column_values3)
    st.write('You selected:', value3)

    value4 = st.number_input("Enter a prod. year")
    st.write('You entered:', value4)

```

Slika 5.4. Kod streamlit aplikacije

```

column_values16 = data['Airbags'].unique()
value16 = st.selectbox('Airbags', column_values16)
st.write('You selected:', value16)

if st.sidebar.button('Predict'):
    input_row = pd.DataFrame({
        'Price': [value0],
        'Levy': [value1],
        'Prod. year': [value4],
        'Category': [value5],
        'Leather interior': [value6],
        'Fuel type': [value7],
        'Engine volume': [value8],
        'Mileage': [value9],
        'Cylinders': [value10],
        'Gear box type': [value11],
        'Drive wheels': [value12],
        'Doors': [value13],
        'Wheel': [value14],
        'Color': [value15],
        'Airbags': [value16]
    })

    prediction = predict(input_row)

    if prediction>=0.5:
        st.write("True")
    else:
        st.write("False")

```

Slika 5.4. Kod streamlit aplikacije

6. ZAKLJUČAK

Na temelju navedenih mjera evaluacije, možemo zaključiti da model Random Forest Regressor ima najmanju vrijednost MSE i MAE, što znači da pokazuje najbolju točnost u predviđanju cijene automobila među ovim modelima. Također, R2 ocjena od 0,75 pokazuje da je model uspješan. Model linearne regresije ima najlošije rezultate u usporedbi s drugim modelima, s najvećim vrijednostima MSE i MAE te najnižom R2 ocjenom dok K-najbližih susjeda i stablo odlučivanja imaju slične vrijednosti.