CAR RESALE VALUE PREDICTION

TEAM ID: PNT2022TMID12591

In [141]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import pickle
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split, cross_val_score
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import r2_score
```

In [142]:
```
df=pd.read_csv('autos.csv',header=0,sep=',',encoding='Latin1')
```

```
c:\New folder\lib\site-packages\IPython\core\interactiveshell.py:3340: DtypeWar
ning: Columns (11) have mixed types.Specify dtype option on import or set low_m
emory=False.
  exec(code_obj, self.user_global_ns, self.user_ns)
```

In [143]:
```
df.head()
```

Out[143]:

	dateCrawled	name	seller	offerType	price	abtest	vehicleType	ye
0	24-03-2016 11:52	Golf_3_1.6	privat	Angebot	480.0	test	NaN	
1	24-03-2016 10:58	A5_Sportback_2.7_Tdi	privat	Angebot	18300.0	test	coupe	
2	14-03-2016 12:52	Jeep_Grand_Cherokee_"Overland"	privat	Angebot	9800.0	test	suv	
3	17-03-2016 16:54	GOLF_4_1_4__3TÜRER	privat	Angebot	1500.0	test	kleinwagen	
4	31-03-2016 17:25	Skoda_Fabia_1.4_TDI_PD_Classic	privat	Angebot	3600.0	test	kleinwagen	

In [144]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 371539 entries, 0 to 371538
Data columns (total 20 columns):
 #   Column               Non-Null Count   Dtype  
---  ------               --------------   -----  
 0   dateCrawled          371539 non-null  object 
 1   name                 371539 non-null  object 
 2   seller               371538 non-null  object 
 3   offerType            371538 non-null  object 
 4   price                371538 non-null  float64
 5   abtest               371538 non-null  object 
 6   vehicleType          333669 non-null  object 
 7   yearOfRegistration   371537 non-null  float64
 8   gearbox              351329 non-null  object 
 9   powerPS              371538 non-null  float64
 10  model                351054 non-null  object 
 11  kilometer            371538 non-null  object 
 12  monthOfRegistration  371537 non-null  float64
 13  fuelType             338151 non-null  object 
 14  brand                371537 non-null  object 
 15  notRepairedDamage    299477 non-null  object 
 16  dateCreated          371537 non-null  object 
 17  nrOfPictures         371537 non-null  float64
 18  postalCode           371537 non-null  float64
 19  lastSeen             371537 non-null  object 
dtypes: float64(6), object(14)
memory usage: 56.7+ MB
```

In [145]:
```
print(df['seller'].value_counts())
print(df['offerType'].value_counts())
```

```
privat        371534
gewerblich         3
golf               1
Name: seller, dtype: int64
Angebot    371525
Gesuch         12
150000          1
Name: offerType, dtype: int64
```

In [146]:
```
df.drop(columns=['seller','offerType','name','abtest','dateCrawled','nrOfPictures
```

In [147]:
```
df.head()
```

Out[147]:

	price	vehicleType	yearOfRegistration	gearbox	powerPS	model	kilometer	monthOfRegistra
0	480.0	NaN	1993.0	manuell	0.0	golf	150000	
1	18300.0	coupe	2011.0	manuell	190.0	NaN	125000	
2	9800.0	suv	2004.0	automatik	163.0	grand	125000	
3	1500.0	kleinwagen	2001.0	manuell	75.0	golf	150000	
4	3600.0	kleinwagen	2008.0	manuell	69.0	fabia	90000	

In [148]:
```
df.shape
```

Out[148]:
```
(371539, 11)
```

In [149]:
```
df=df[(df['powerPS']>50) & (df['powerPS']<90)]
```

In [150]:
```
df=df[(df['yearOfRegistration']>1949) & (df['yearOfRegistration']<2017)]
df=df[(df['price']>=100) & (df['price']<=150000)]
```

In [151]:
```
df.shape
```

Out[151]:
```
(78684, 11)
```

In [152]:
```
df.columns
```

Out[152]:
```
Index(['price', 'vehicleType', 'yearOfRegistration', 'gearbox', 'powerPS',
       'model', 'kilometer', 'monthOfRegistration', 'fuelType', 'brand',
       'notRepairedDamage'],
      dtype='object')
```

In [153]:
```
new_df=df.copy()
new_df=new_df.drop_duplicates(['price', 'vehicleType', 'yearOfRegistration', 'gea
```

In [154]:
```
new_df['gearbox'].replace(('manuell','automatik'),('manual','automatic'),inplace=
new_df['fuelType'].replace(('benzin','andere','elektro'),('petrol','others','elec
new_df['vehicleType'].replace(('kleinwagen','cabrio','kombi','andere'),('small ca
new_df['notRepairedDamage'].replace(('ja','nein'),('yes','no'),inplace=True)
```

In [155]:
```
new_df.to_csv('autos_preprocessed.csv')
```

In [156]:
```
labels=['gearbox','notRepairedDamage','model','brand','fuelType','vehicleType']
```

In [157]:
```
dicti={}
for feature in labels:
    dicti[feature]=LabelEncoder()
    dicti[feature].fit(new_df[feature])
    temp=dicti[feature].transform(new_df[feature])
    np.save(str('classes'+feature+'.npy'),dicti[feature].classes_)
    print(feature,':',dicti[feature])
    new_df.loc[:,feature+'_Labels']=pd.Series(temp,index=new_df.index)
labeled=new_df[['price','yearOfRegistration','powerPS','kilometer','monthOfRegist
```

```
gearbox : LabelEncoder()
notRepairedDamage : LabelEncoder()
model : LabelEncoder()
brand : LabelEncoder()
fuelType : LabelEncoder()
vehicleType : LabelEncoder()
```

In [158]:
```
labeled.columns
```

Out[158]:
```
Index(['price', 'yearOfRegistration', 'powerPS', 'kilometer',
       'monthOfRegistration', 'gearbox_Labels', 'notRepairedDamage_Labels',
       'model_Labels', 'brand_Labels', 'fuelType_Labels',
       'vehicleType_Labels'],
      dtype='object')
```

In [159]:
```
X=labeled.iloc[:,1:].values
Y=labeled.iloc[:,0].values
```

In [160]:
```
Y=Y.reshape(-1,1)
```

In [161]:
```
X_train,X_test,y_train,y_test=train_test_split(X,Y,test_size=.3)
```

In [162]:
```
regressor=RandomForestRegressor(n_estimators=1000,max_depth=10)
regressor.fit(X_train,np.ravel(y_train,order='C'))
```

Out[162]:
```
RandomForestRegressor(max_depth=10, n_estimators=1000)
```

In [163]:
```
prediction=regressor.predict(X_test)
```

In [164]:
```
r2_score(y_test,prediction)
```

Out[164]:
```
0.747891214137351
```

In [165]:
```
file='model.sav'
pickle.dump(regressor,open(file,'wb'))
```

In [172]:
```
from sklearn.linear_model import LogisticRegression
```

In [173]:
```
lr = LogisticRegression()
```

In [174]:
```
lr.fit(X_train, y_train)
```

```
c:\New folder\lib\site-packages\sklearn\utils\validation.py:993: DataConversion
Warning: A column-vector y was passed when a 1d array was expected. Please chan
ge the shape of y to (n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
c:\New folder\lib\site-packages\sklearn\linear_model\_logistic.py:814: Converge
nceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html (https://scikit-
learn.org/stable/modules/preprocessing.html)
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regressi
on (https://scikit-learn.org/stable/modules/linear_model.html#logistic-regressi
on)
  n_iter_i = _check_optimize_result(
```

Out[174]:
```
LogisticRegression()
```

In [175]:
```
pred = lr.predict(X_test)
```

In [176]:
```
r2_score(y_test,pred)
```

Out[176]:
```
-0.5266927414549378
```

In [ ]:
```

```