Before removing the outliers

Chart, treemap chart

Description automatically generated

Chart, treemap chart

Description automatically generated

Before removing the fuel type

Text

Description automatically generated

df\_4.drop(['Cylinders','Drive wheels','Airbags','Turbo','minDoors','Fuel type'],axis=1, inplace = True)

Text

Description automatically generated

df\_4.drop(['Cylinders','Drive wheels','Airbags','Turbo','minDoors','Fuel type','Levy'],axis=1, inplace = True)

Text

Description automatically generated

**Text

Description automatically generated**

**With no scaling**

**Text

Description automatically generated**

**Changing the mapping to automatic mapping**

**Background pattern

Description automatically generated**

**Text

Description automatically generated**

**After removing the scaling**

**Text

Description automatically generated**

**Removing Levy decreased the accuracy**

**Without**

**Text

Description automatically generated**

**With Levy**

**Text

Description automatically generated**

**By adding 'Cylinders','Drive wheels' to the data is a bit less accurate**

**Removing 'Manufacturer',’Mileage’ from the data make the model little bit less accurate**

**Adding 'minDoors' did nothing**

**Adding airbags**

**Text

Description automatically generated**

**Adding Turbo increased the accuracy**

**Text

Description automatically generated**

**Removing Prod. Year , Model decreased the accuracy**

**Removing Category increased the accuracy**

**Text

Description automatically generated**

**No scaling**

**Text

Description automatically generated** **Text

Description automatically generated**

**Standard scaler (no big impact)**

**Text

Description automatically generated Text

Description automatically generated**

**MinMax Scaler (++ve effect) inconsistent**

**Update: (no big impact)**

**Text

Description automatically generated Text

Description automatically generated RobustScaler (+ve effect)**

**Update: (no big impact)**

**Text

Description automatically generated** **Text

Description automatically generated**

**PowerTransformer (+ve effect)**

**Update: (no big impact)**

**Text

Description automatically generated** **Text

Description automatically generated**

**Trial to impute the milage:**

**TRIAL0012-0912 (no big impact)**

#Imputation of 0 km Mileage #TRIAL0012-0912

from sklearn.impute import KNNImputer

imp\_mileage = KNNImputer(n\_neighbors=3)

x\_train['Mileage'] = imp\_mileage.fit\_transform(x\_train[['Mileage','Prod. year','Levy','Engine volume','Gear box type']])

x\_test['Mileage'] = imp\_mileage.transform(x\_test[['Mileage','Prod. year','Levy','Engine volume','Gear box type']])

**Random under sampler**

**Text

Description automatically generated**

**Oversampling by smote: (only for classification models)**

**Adjusting the price bad\_data limits**

**Trial 1 (no big impact)**

**Text

Description automatically generated**

**Trial 2 (no big impact)**

**Text

Description automatically generated**

**Tial of changing the mean to sum (not a good idea)**

**Text

Description automatically generated**

**Tial of changing the mean to median (a bit less accurate than the mean)**

**Text

Description automatically generated**

**Utilizing the ID: (the accuracy increased) with Robustscaler or powertransformer**

**Text

Description automatically generated**

**Outlier limits 10 and 90 (the accuracy increased)**

**Text

Description automatically generated**

**Outlier limits 0 and 90 (No big chane)**

**Text

Description automatically generated**

**Changing the bad data from 4000 to 2000 (accuracy decreased)**

# Removing the records with a price above 90000 or below 1000

bad\_data = df\_2[(df\_2['Price']>200000) | (df\_2['Price']<2000)]

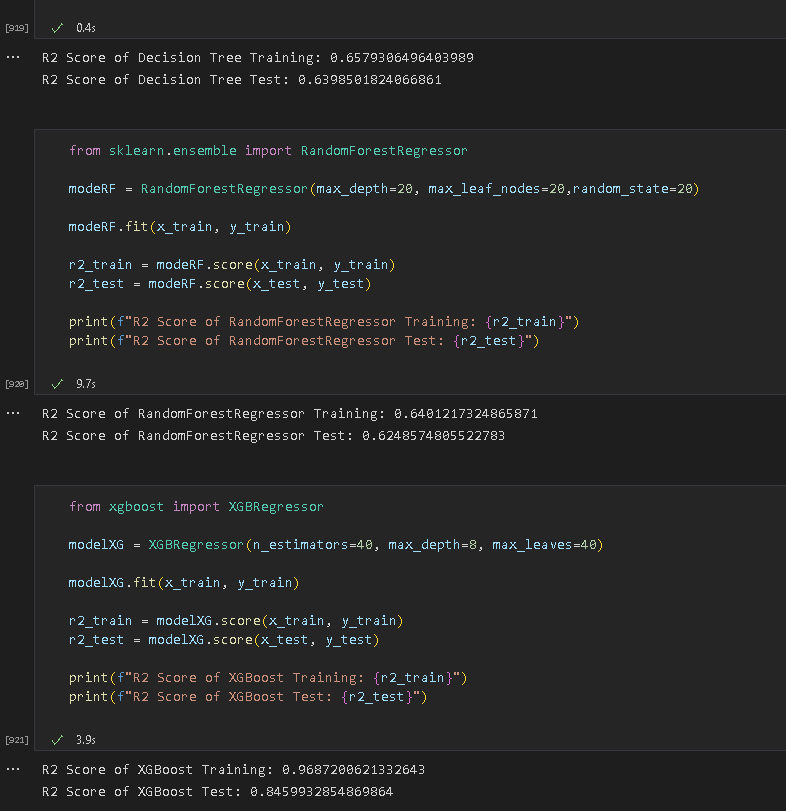
df\_2.drop(bad\_data.index,inplace=True)

**Changing the bad data from 200000 to 300000 with quantile 10 and 90 ()**

# Removing the records with a price above 90000 or below 1000

bad\_data = df\_2[(df\_2['Price']>200000) | (df\_2['Price']<2000)]

df\_2.drop(bad\_data.index,inplace=True)

****

**Check the rest of the outliers**

**Adding the age column = 2020 – Prod year (slightly increased the accuracy**

**Removing the outliers of Engine volume (accuracy increased)**

**Text

Description automatically generated**

**Removing the outliers of Levy (Increased the accuracy)**

**Text

Description automatically generated**

**Removing the outliers of Model (Decreased the accuracy)**

**Removing the outliers of Airbags (Decreased the accuracy)**

**Removing the outliers of Year (Decreased the accuracy)**

**Removing the outliers of Turbo (Decreased the accuracy)**

**Removing the outliers of Manufacturer (Increased the accuracy slightly)**

**Text

Description automatically generated**

|  |  |  |
| --- | --- | --- |
|  | **Variable** | **Effect on test accuracy from -2 to +2** |
| 1 | Mapping with values\_count | -2 |
| 2 | Mapping with mean price | +2 |
| 3 | Mapping with sum price | -1 |
| 4 | Mapping with median price | +1 |
| 5 | Dropping the Price outliers | +2 |
| 6 | Price outlers 10 to 90 % quantile | +2 |
| 7 | Price outlers 25 to 75 % quantile | +1 |
| 8 | Changing the bad data from 4000 to 2000 Georgian Lari | -1 |
| 9 | Dropping the outliers of Engine volume | +1 |
| 10 | Dropping the outliers of Levy | +1 |
| 11 | Dropping the outliers of Model | -1 |
| 12 | Dropping the outliers of Airbags | -1 |
| 13 | Dropping the outliers of Prod year | -1 |
| 14 | Dropping the outliers of Turbo | -1 |
| 15 | Dropping the outliers of Manufacturer | +1 |
| 16 | Excluding "Levy" | -2 |
| 17 | Excluding "Cylinders" | -1 |
| 18 | Excluding "Drive wheels" | +1 |
| 19 | Excluding "Manufacturer'" | -1 |
| 20 | Excluding "Mileage" | -1 |
| 21 | Excluding "minDoors" | 0 |
| 22 | Excluding "airbags" | -1 |
| 23 | Excluding "Turbo" | -1 |
| 24 | Excluding "Prod. Year" | -1 |
| 25 | Excluding "Category" | +1 |
| 26 | Standard scaler | -1 |
| 27 | MinMax Scaler | 0 |
| 28 | RobustScaler | -1 |
| 29 | PowerTransformer | -1 |
| 30 | Trial to impute the wrong milage bu KNN instead of dropping it | -1 |
| 31 | Dropping the irrational Mileage | +2 |

**The initial accuracy:**

**Train: 90.7 %**

**Test: 69.46 %**

**The best accuracy:**

**Train: 96.17 %**

**Test: 84.95 %**

**When tester.py is used to evaluate performance, precision and recall are both at least 0.3.**

**use val and Discuss validation and its importance.**

**The accuracy should be >90 %**

**use GridSearchCV)**

**Text

Description automatically generated**

**Explain the Project (the data cleaning) and show the variables that increase the accuracy.**

**A GitHub repo containing the following:**

**1- The project notebook/code**

**2- All datasets used**

**3- Video of screen recording explaining the project**

**4- Deployed Web Application (include the link for the application)**

**You must provide a link with your web application**

**Show visualizations.**