CAPSTONE PROJECT

FINAL REPORT

Post Graduate Program in Data Science Engineering Location: Bangalore Batch :

PGP DSE-APR 23

PREDICITING PRICE OF USED CARS

BY GROUP 6:

ADARSH B

ALKA CHAUHAN R

DINESH S

GAGAN

SREEDEVI KSS

Mentor:

Chandran Venkatesan

Step-1 PROBLEM STATEMENT:

Background:

The used car market is inherently complex, influenced by an array of factors such as make, model, year, mileage, condition, and market trends. Traditionally, pricing decisions have been subjective, often relying on intuition and personal judgment. This approach, however, can lead to suboptimal outcomes, including overpricing, which may result in extended inventory turnover, or underpricing, causing potential revenue loss for sellers. Moreover, the lack of transparency in pricing can leave buyers in a disadvantaged position during negotiations.

In the dynamic and complex market of used cars, predicting accurate prices for pre-owned vehicles is crucial for facilitating fair transactions, optimizing inventory management, and enhancing the overall efficiency of the automotive ecosystem. The lack of a reliable method for predicting used car prices poses significant challenges for both buyers and sellers, leading to suboptimal pricing decisions, increased market uncertainty, and potentially inefficient resource allocation.

Business problem

The business problem involves predicting accurate prices for used cars based on various attributes. The objective is to provide a model that estimates resale values, offering buyers and sellers a fair pricing benchmark and enhancing transparency in the market. The primary objective is to build a predictive model capable of estimating used car prices with precision. This empowers buyers to gauge value and assists sellers in setting reasonable prices. Ultimately, the aim is to foster trust and confidence in used car transactions.

About this dataset

The autos.csv dataset is a comprehensive collection of valuable data about used cars, and provides insight into how the cars are being sold, what price they are being sold for, and all the details about their condition. Each ad contains information such as date Crawled (the date the ad was first seen), name of the car, seller type (private or dealer), offer type, price, A/B testing information , vehicle type, year of registration (at which year was the car first registered) , gearbox type, power output in PS (horsepower) , model of car , how many kilometers has it driven so far , month of registration(when it was first registered)(essentially giving us an idea about its age), fuel type utilized by it(petrol/diesel /electricity/ lpg etc.), brand name to which it belongs to notRepairedDamage - if there is any damage on the vehicle that has not been repaired yet. Date Created gives us information when this particular advertisement was created in e-bay or other place where these cars can be posted.

Step 2- Overview of the final process

The method uses an organized workflow:

DATA EXPLORATION
DATA PREPROCESSING
FEATURE ENGINEERING
MODEL TRAINING
MODEL EVALUATION

- Data Exploration
- Data Preprocessing
- Feature Engineering
- Model Training
- Model Evaluation

Data Pre-processing steps:

- Handling missing values.
- Dealing with outliers
- Feature Engineering
- Encoding categorical variables
- Scaling numerical features

Algorithms used: Regression analysis

- Linear regression
- Decision tree regression
- Random forest regressor
- Gradient boost
- XGBoost
- Support vector regressor

Step 3- Step by step walk through of the solution

Stage 1- Data exploration and pre-preprocessing:

Stage- 2

- Missing data imputation – We have dropped the missing data because we had very few missing data
- Categorical encoding - N_1 dummy encoding
- Variable transformation – Power Transformer
- Outlier engineering
- Date and time engineering

Stage-3 - Model training and Evaluation

We have used train_test_split to split the data and made various models using Grid Search CV, cross validation score, sequential feature selector, and used r2_score, mean_squared_error , mean absolute percentage error for evaluation.

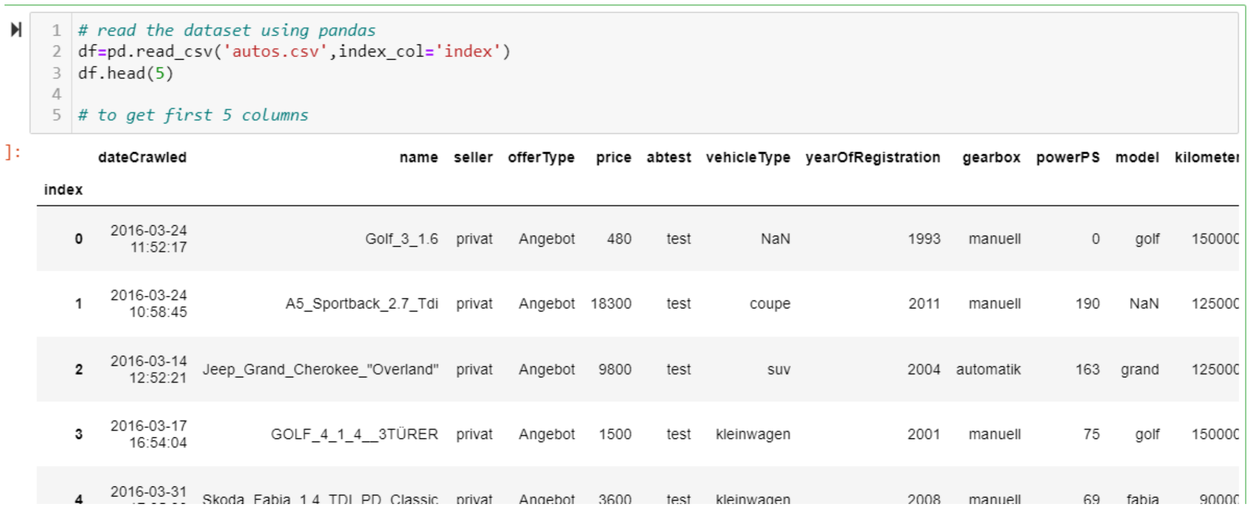
IN DETAIL:

3.1. Initial Exploration we checked the shape , data types of the variables, then explored the dataset using descriptive statistics, uni-variate analysis, bivariate analysis, pair plots, box plot and correlation heat maps. This allowed us to understand the 5-point summary, distribution of data, identify outliers, and find correlations between features.

Features of the Dataset

- dateCrawled: Date the car advertisement was crawled. (Date)
- name: Name of the car. (String)
- seller: Type of seller (private or dealer). (String)
- offerType: Type of offer (e.g. sale, repair, etc.). (String)
- price: Price of the car. (Integer)
- abtest: Test type (A or B). (String)
- vehicleType: Type of vehicle (e.g. SUV, sedan, etc.). (String)
- yearOfRegistration: Year the car was registered. (Integer)
- gearbox: Type of gearbox (manual or automatic). (String)
- powerPS: Power of the car in PS. (Integer)
- model: Model of the car. (String)
- kilometer: Kilometers the car has been driven. (Integer)
- monthOfRegistration: Month the car was registered. (Integer)
- fuelType: Type of fuel (e.g. diesel, petrol, etc.). (String)
- brand: Brand of the car. (String)
- notRepairedDamage: Whether or not the car has any damage that has not been repaired. (String)
- dateCreated: Date the car advertisement was created. (Date)
- nrOfPictures: Number of pictures of the car. (Integer)
- postalCode: Postal code of the car. (Integer)
- lastSeen: Date the car was last seen. (Date)

Data View

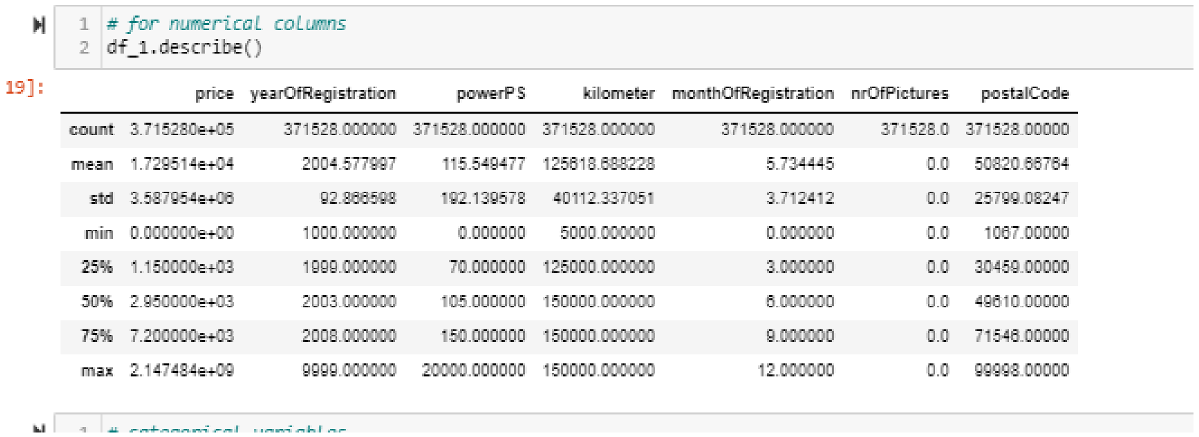


```
# read the dataset using pandas
df=pd.read_csv('autos.csv',index_col='index')
df.head(5)

# to get first 5 columns
```

]:

index	dateCrawled	name	seller	offerType	price	abtest	vehicleType	yearOfRegistration	gearbox	powerPS	model	kilometer
0	2016-03-24 11:52:17	Golf_3_1.6	privat	Angebot	480	test	NaN	1993	manuell	0	golf	150000
1	2016-03-24 10:58:45	A5_Sportback_2.7_Tdi	privat	Angebot	18300	test	coupe	2011	manuell	190	NaN	125000
2	2016-03-14 12:52:21	Jeep_Grand_Cherokee_"Overland"	privat	Angebot	9800	test	suv	2004	automatik	163	grand	125000
3	2016-03-17 16:54:04	GOLF_4_1_4__3TÜRER	privat	Angebot	1500	test	kleinwagen	2001	manuell	75	golf	150000
4	2016-03-31	Skoda_Fabia_1.4_TDI_PD_Classic	privat	Angebot	3600	test	kleinwagen	2008	manuell	69	fabia	90000

5 –Point summary



```
# for numerical columns
df_1.describe()
```

19]:

	price	yearOfRegistration	powerPS	kilometer	monthOfRegistration	nrOfPictures	postalCode
count	3.715280e+05	371528.000000	371528.000000	371528.000000	371528.000000	371528.0	371528.00000
mean	1.729514e+04	2004.577997	115.549477	125618.688228	5.734445	0.0	50820.66764
std	3.587954e+06	92.866598	192.139578	40112.337051	3.712412	0.0	25799.08247
min	0.000000e+00	1000.000000	0.000000	5000.000000	0.000000	0.0	1067.00000
25%	1.150000e+03	1999.000000	70.000000	125000.000000	3.000000	0.0	30459.00000
50%	2.950000e+03	2003.000000	105.000000	150000.000000	6.000000	0.0	49610.00000
75%	7.200000e+03	2008.000000	150.000000	150000.000000	9.000000	0.0	71546.00000
max	2.147484e+09	9999.000000	20000.000000	150000.000000	12.000000	0.0	99998.00000

- yearOfRegistration - min is 1000 and max is 9999 , its incorrect as year cant be such numbers
- powerPS - mean is 105,max is very high ,i,e 20000
- Mininum nuber of kilometers the car has ran is 5000kms and maximum is 150000kms
- no of picttures feature has no values, it has only 0, its s a insignificant variable

Changing datatype

- DateTime
  Data type of datecrawled, lastseen, and datecreated columns is object --> Convert to DateTime

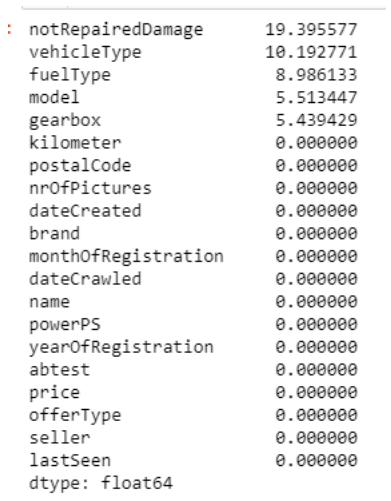
Removing Irrelavent data

- price has 10778 (2.9%) zeros
- powerPS has 40820 (11.0%) zeros
- monthOfRegistration has 37675 (10.1%) zeros
- We have removed the irrelavent data from these features by taking a price greater than 0 and considering only valid years for analysis

Missing Values:

While we have worked to treat missing values in our dataset, we have taken a different approach when it comes to outliers, given the type of data available and the scope of analysis that we aim to undertake. Instead of removing outliers , we did capping method to treat outliers so that some records are not lost in our dataset.

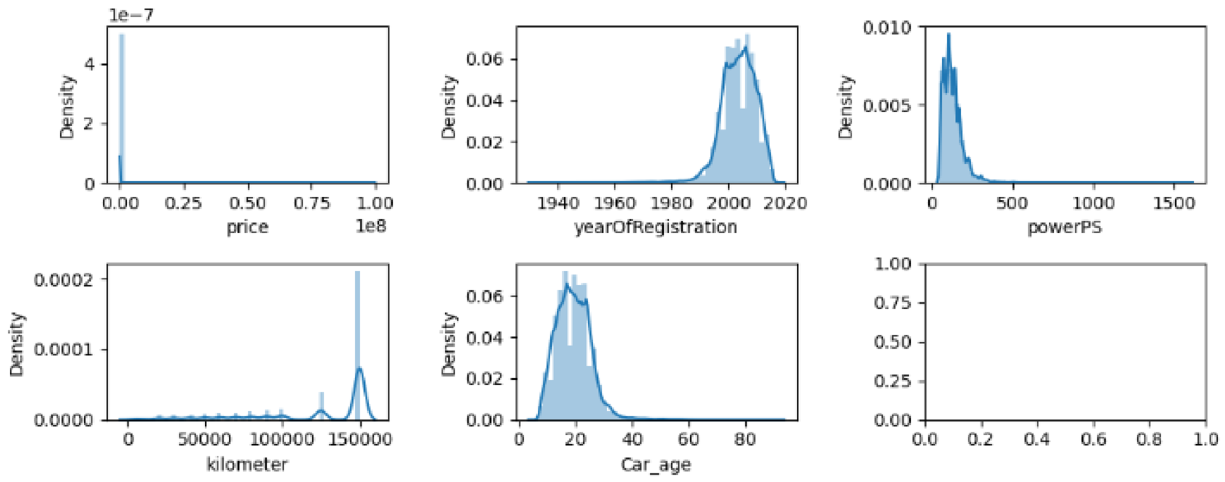
Winsorization can be a good approach to handle a large number of outliers, particularly because the outliers are genuine extreme values and removing them would significantly alter the results of the analysis. Winsorization is a data preprocessing technique that involves replacing extreme values in a dataset with less extreme values, typically the highest or lowest non-extreme values. This approach can be useful for handling outliers because it preserves the original data distribution while reducing the influence of extreme values. Winsorization can also be used to reduce the impact of measurement errors, which can cause extreme values in the data



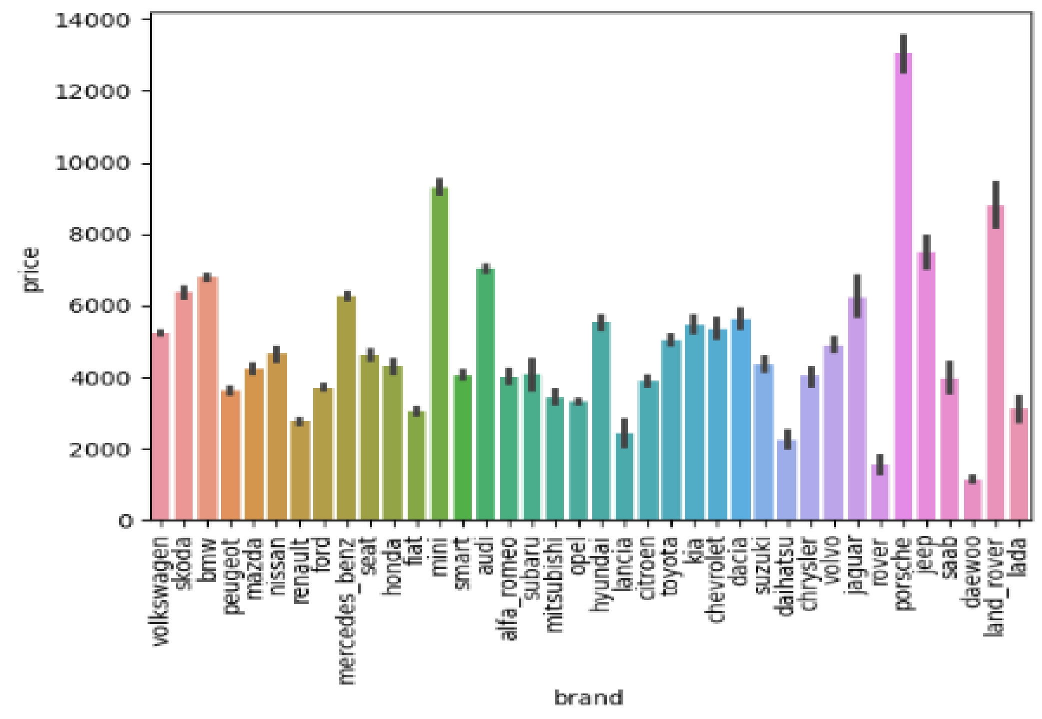
```
notRepairedDamage      19.395577
vehicleType            10.192771
fuelType                8.986133
model                   5.513447
gearbox                 5.439429
kilometer               0.000000
postalCode              0.000000
nrOfPictures            0.000000
dateCreated             0.000000
brand                   0.000000
monthOfRegistration     0.000000
dateCrawled             0.000000
name                    0.000000
powerPS                 0.000000
yearOfRegistration      0.000000
abtest                  0.000000
price                   0.000000
offerType               0.000000
seller                  0.000000
lastSeen                0.000000
dtype: float64
```

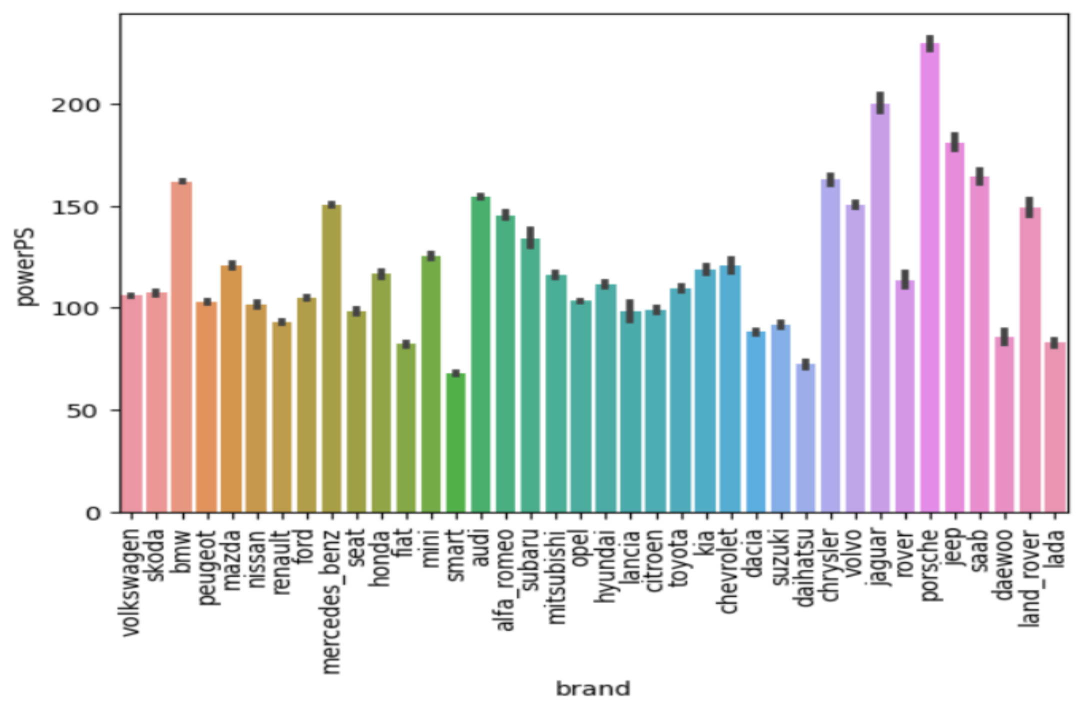
UNIVARIATE ANALYSIS

Our Data is not normally distributed, we will be transforming it



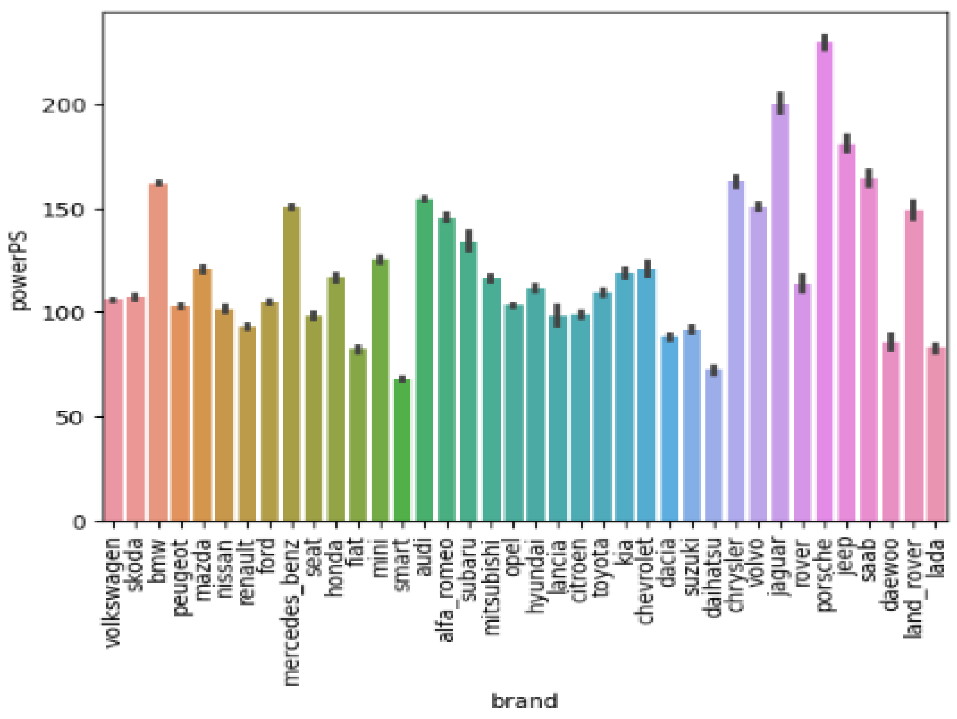
BIVARIATE ANALYSIS





```
sns.barplot(data=df_rem,x='brand',y='powerPS')
plt.xticks(rotation=90)
```

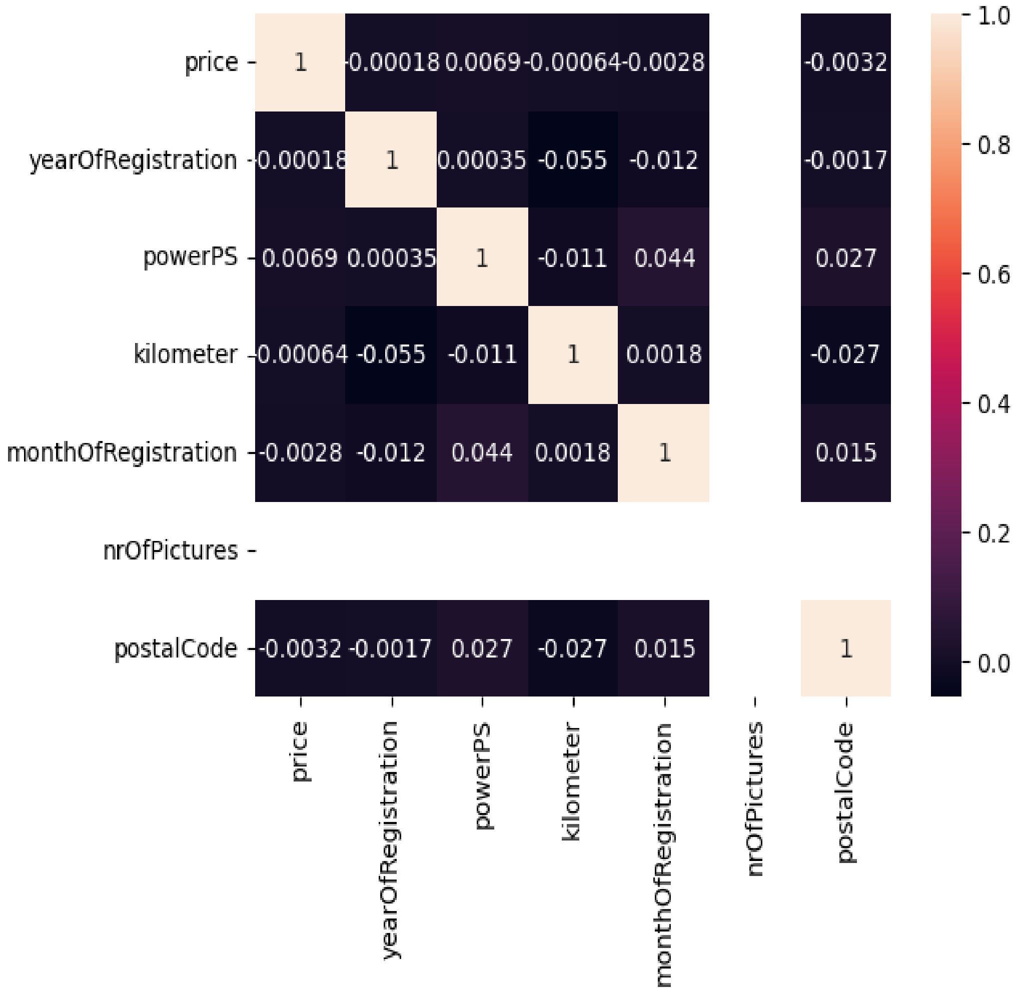
```
73]: (array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16,
        17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33,
        34, 35, 36, 37]),
 [Text(0, 0, 'volkswagen'),
  Text(1, 0, 'skoda'),
  Text(2, 0, 'bmw'),
  Text(3, 0, 'peugeot'),
  Text(4, 0, 'mazda'),
  Text(5, 0, 'nissan'),
  Text(6, 0, 'renault'),
  Text(7, 0, 'ford'),
  Text(8, 0, 'mercedes_benz'),
  Text(9, 0, 'seat'),
  Text(10, 0, 'honda'),
  Text(11, 0, 'fiat'),
  Text(12, 0, 'mini'),
  Text(13, 0, 'smart'),
  Text(14, 0, 'audi'),
  Text(15, 0, 'alfa_romeo'),
  Text(16, 0, 'subaru'),
  Text(17, 0, 'mitsubishi'),
  Text(18, 0, 'opel'),
  Text(19, 0, 'hyundai'),
  Text(20, 0, 'lancia'),
  Text(21, 0, 'citroen'),
  Text(22, 0, 'toyota'),
  Text(23, 0, 'kia'),
  Text(24, 0, 'chevrolet'),
  Text(25, 0, 'dacia'),
  Text(26, 0, 'suzuki'),
  Text(27, 0, 'daihatsu'),
  Text(28, 0, 'chrysler'),
  Text(29, 0, 'volvo'),
  Text(30, 0, 'jaguar'),
  Text(31, 0, 'rover'),
  Text(32, 0, 'porsche'),
  Text(33, 0, 'jeep'),
  Text(34, 0, 'saab'),
  Text(35, 0, 'daewoo'),
  Text(36, 0, 'land_rover'),
  Text(37, 0, 'lada')])
```



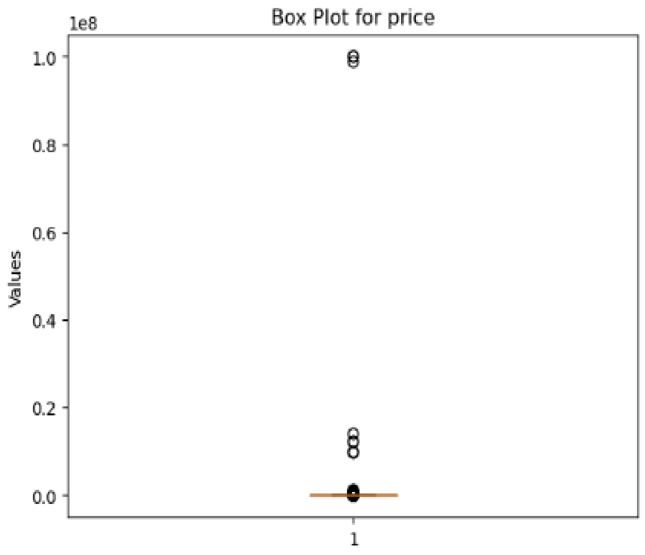
.

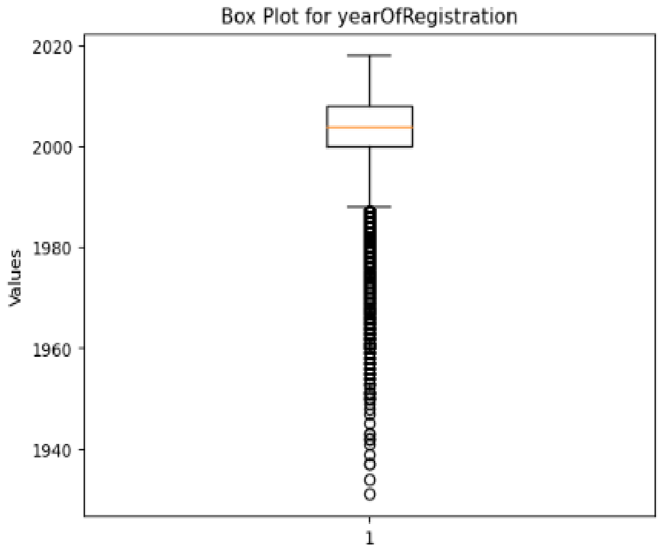
Correlation between features – Heatmap

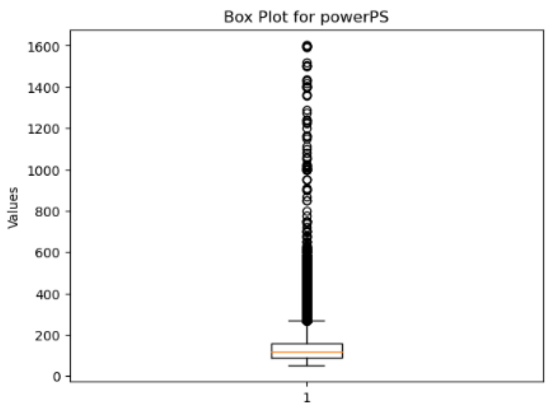
There is no High correlation between the variables in our dataset.Picture column is constant, we have only one value in it.

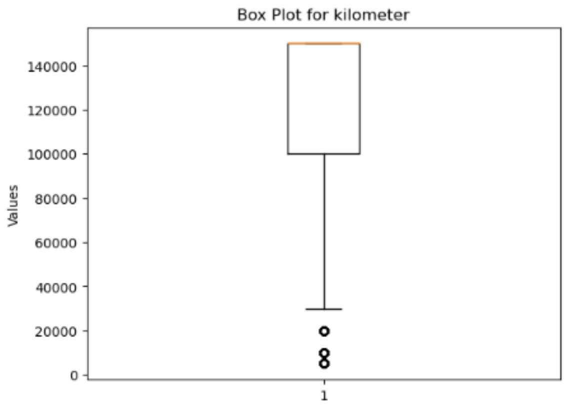


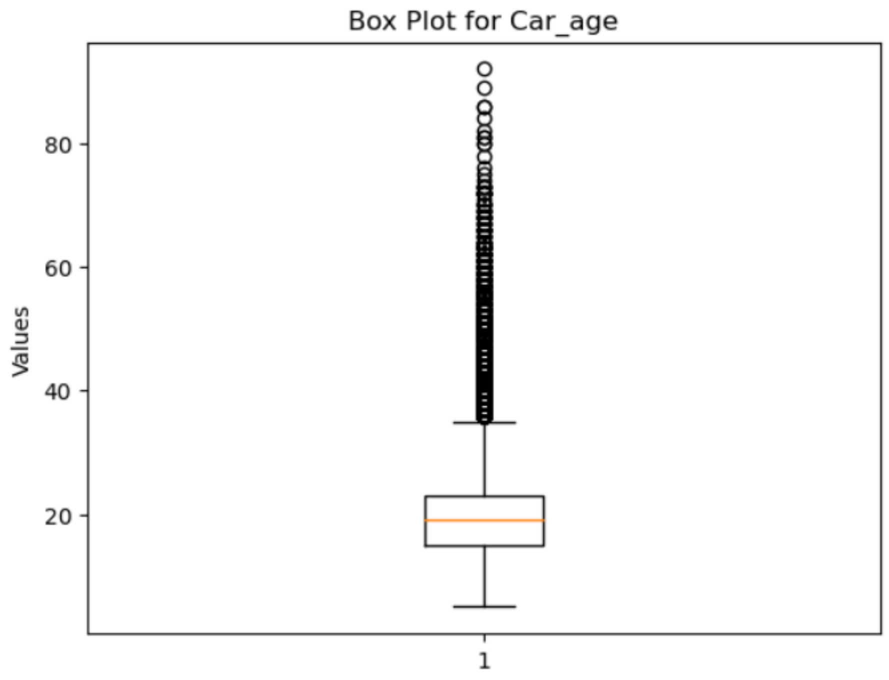
Dealing with outliers - We have dropped the outliers (Because we many extreme outliers)





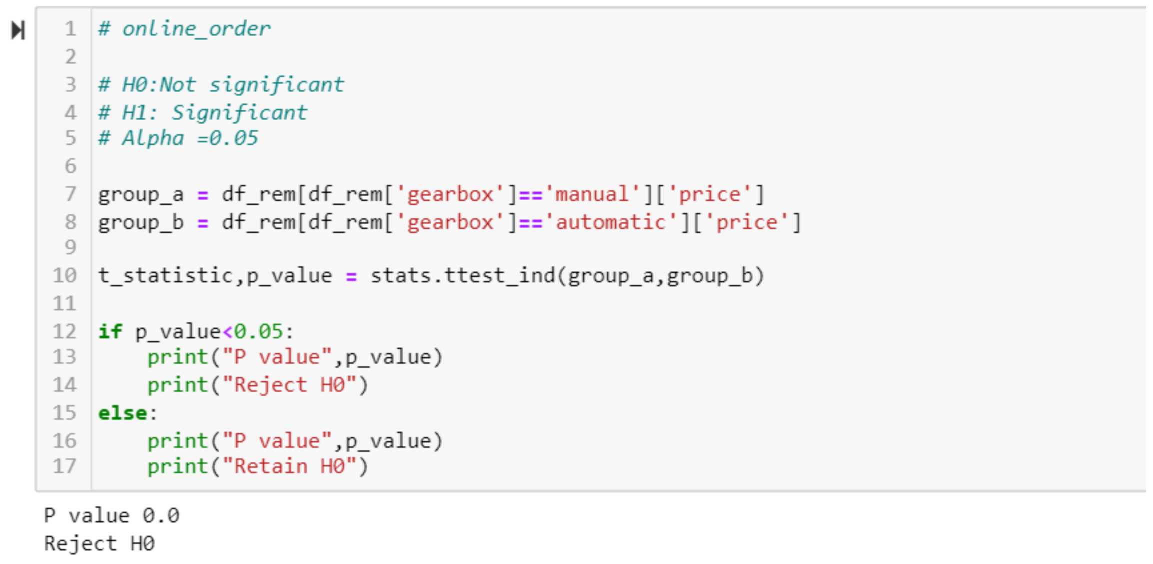

Box Plot for powerPS
Values
1600
1400
1200
1000
800
600
400
200
0
1


Box Plot for kilometer
Values
140000
120000
100000
80000
60000
40000
20000
0
1


Box Plot for Car_age
Values
80
60
40
20
1

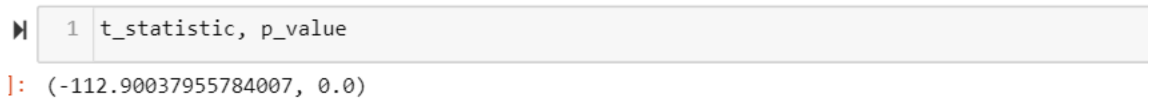
Statistical test

A. To check if gearbox type affecting price



```
# online_order

# H0:Not significant
# H1: Significant
# Alpha =0.05

group_a = df_rem[df_rem['gearbox']=='manual']['price']
group_b = df_rem[df_rem['gearbox']=='automatic']['price']

t_statistic,p_value = stats.ttest_ind(group_a,group_b)

if p_value<0.05:
    print("P value",p_value)
    print("Reject H0")
else:
    print("P value",p_value)
    print("Retain H0")
```

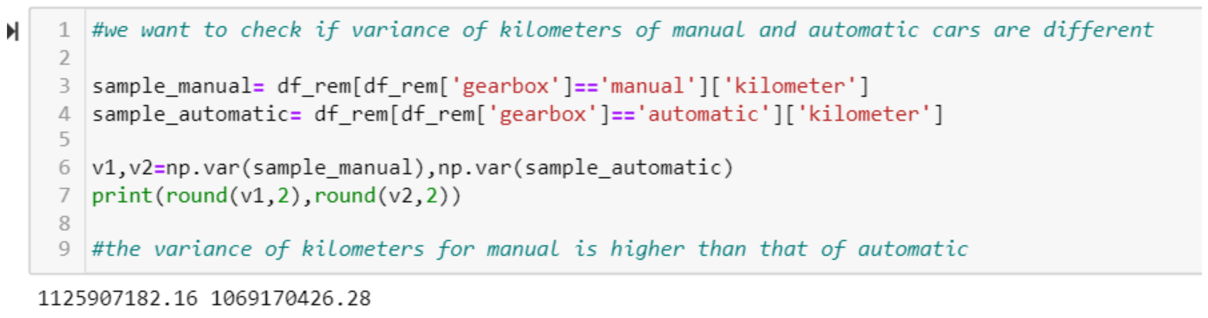
```
P value 0.0
Reject H0
```



```
t_statistic, p_value
```

```
]: (-112.90037955784007, 0.0)
```

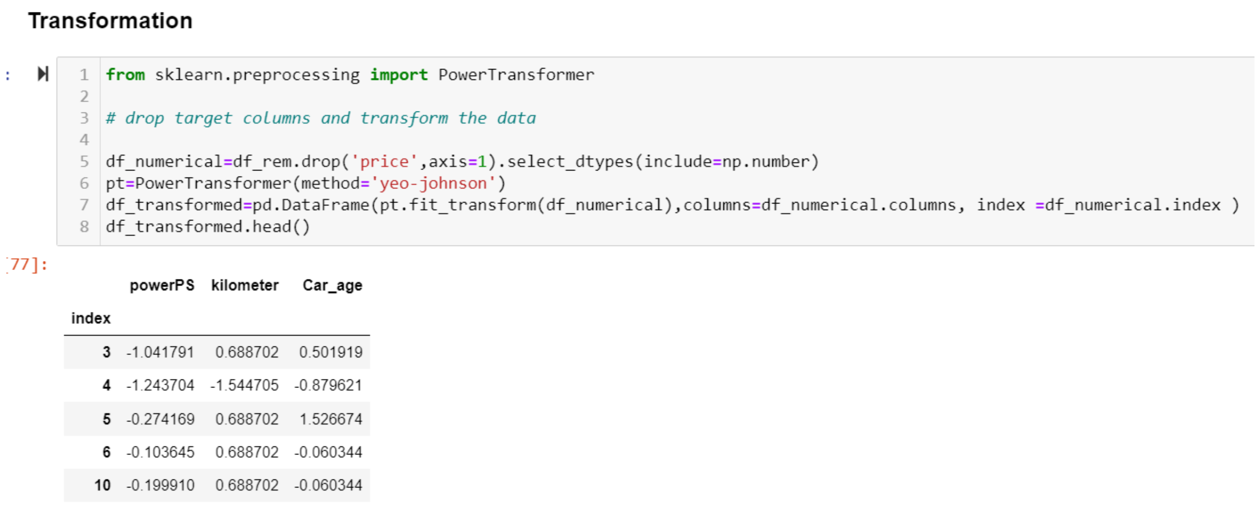
B. Check if variance of kilometers of manual and automatic cars are different



```
#we want to check if variance of kilometers of manual and automatic cars are different

sample_manual= df_rem[df_rem['gearbox']=='manual']['kilometer']
sample_automatic= df_rem[df_rem['gearbox']=='automatic']['kilometer']

v1,v2=np.var(sample_manual),np.var(sample_automatic)
print(round(v1,2),round(v2,2))

#the variance of kilometers for manual is higher than that of automatic
```

```
1125907182.16 1069170426.28
```

3.2. Preprocessing we cleaned the data by handling missing values. For encoding the categorical variables, we have used one-hot encoding . For numerical variables, we applied power transformer (yeo-johnson method) to scale the data to a specific range.

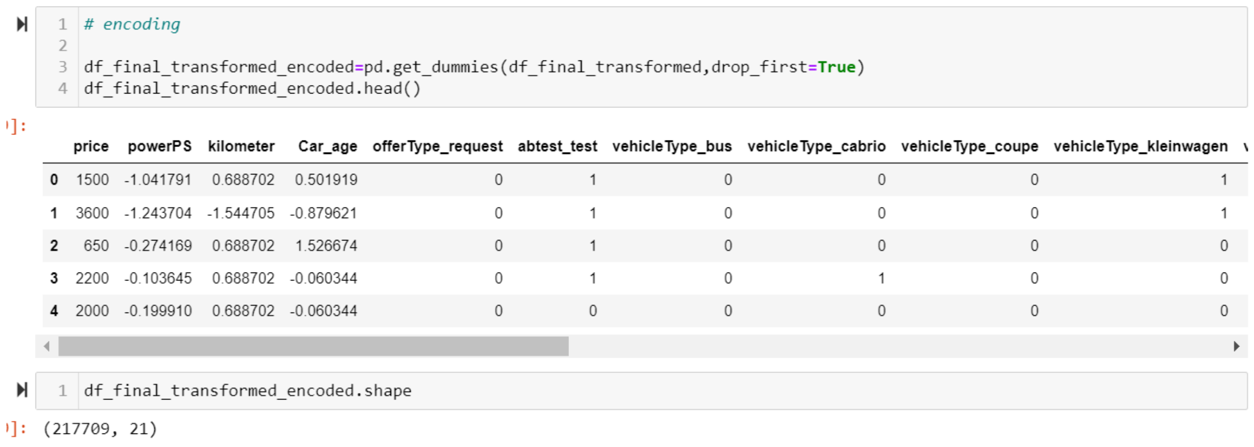


Transformation

```python
from sklearn.preprocessing import PowerTransformer

# drop target columns and transform the data

df_numerical=df_rem.drop('price',axis=1).select_dtypes(include=np.number)
pt=PowerTransformer(method='yeo-johnson')
df_transformed=pd.DataFrame(pt.fit_transform(df_numerical),columns=df_numerical.columns, index =df_numerical.index )
df_transformed.head()
```

index	powerPS	kilometer	Car_age
3	-1.041791	0.688702	0.501919
4	-1.243704	-1.544705	-0.879621
5	-0.274169	0.688702	1.526674
6	-0.103645	0.688702	-0.060344
10	-0.199910	0.688702	-0.060344

Transformed and Encoded Data



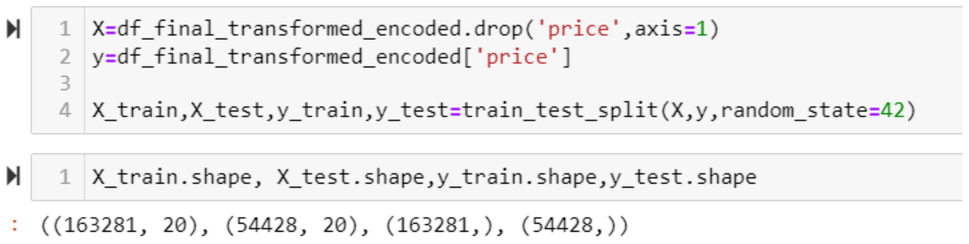
```python
# encoding

df_final_transformed_encoded=pd.get_dummies(df_final_transformed,drop_first=True)
df_final_transformed_encoded.head()
```

	price	powerPS	kilometer	Car_age	offerType_request	abtest_test	vehicleType_bus	vehicleType_cabrio	vehicleType_coupe	vehicleType_kleinwagen
0	1500	-1.041791	0.688702	0.501919	0	1	0	0	0	1
1	3600	-1.243704	-1.544705	-0.879621	0	1	0	0	0	1
2	650	-0.274169	0.688702	1.526674	0	1	0	0	0	0
3	2200	-0.103645	0.688702	-0.060344	0	1	0	1	0	0
4	2000	-0.199910	0.688702	-0.060344	0	0	0	0	0	0

```python
df_final_transformed_encoded.shape
```

```
(217709, 21)
```

3.3. Training and Testing we split the dataset into training and testing sets. Taking 30% as test set. We trained the XGB model using the training data.



```
X=df_final_transformed_encoded.drop('price',axis=1)
y=df_final_transformed_encoded['price']

X_train,X_test,y_train,y_test=train_test_split(X,y,random_state=42)
```

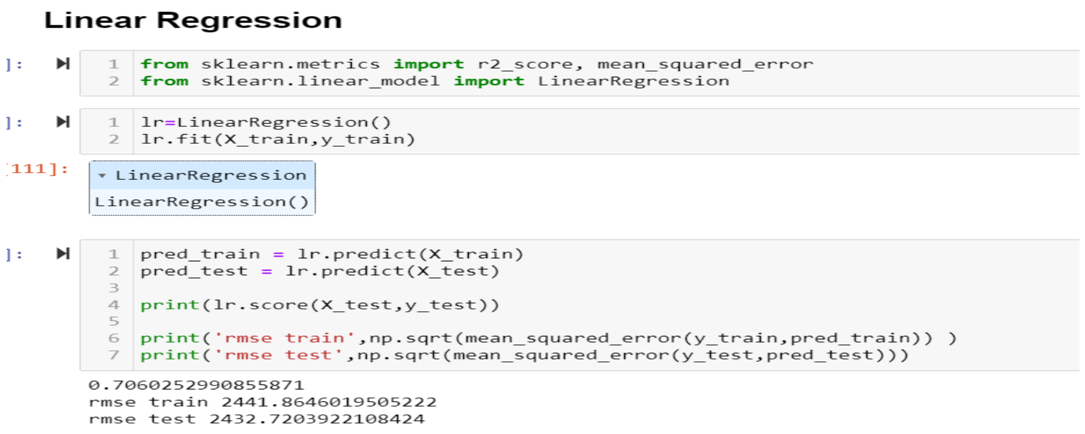
```
X_train.shape, X_test.shape,y_train.shape,y_test.shape
```

```
((163281, 20), (54428, 20), (163281,), (54428,))
```

3.4. We tried models like linear regression, decision tree regression, random forest regressor, gradient boost, XGBoost, support vector regressor.

Model Selection - we decided to use a XGBoost model as our final model due to its ability to give high performance, has built in pruning algorithm that removes splits and add little predictive power.

Base Model



Linear Regression

```
from sklearn.metrics import r2_score, mean_squared_error
from sklearn.linear_model import LinearRegression
```

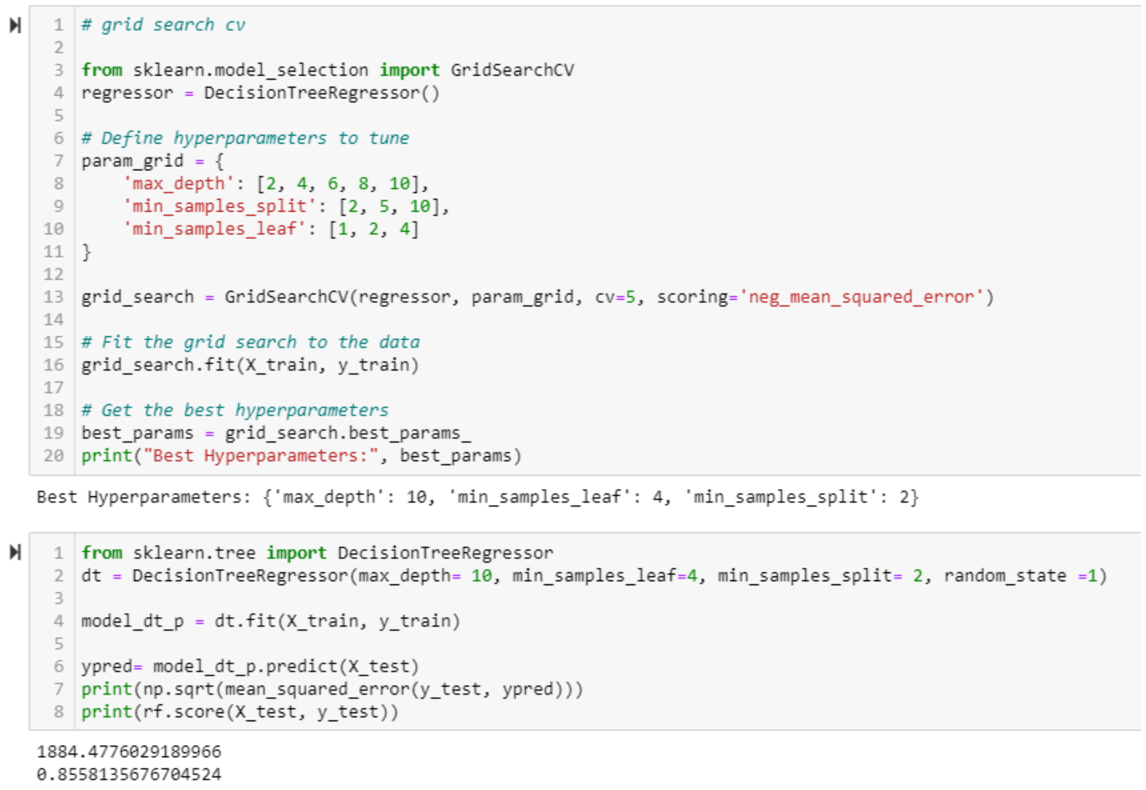
```
lr=LinearRegression()
lr.fit(X_train,y_train)
```

```
[111]:
▾ LinearRegression
LinearRegression()
```

```
pred_train = lr.predict(X_train)
pred_test = lr.predict(X_test)

print(lr.score(X_test,y_test))

print('rmse train',np.sqrt(mean_squared_error(y_train,pred_train)) )
print('rmse test',np.sqrt(mean_squared_error(y_test,pred_test)))
```

```
0.7060252990855871
rmse train 2441.8646019505222
rmse test 2432.7203922108424
```

Grid Search CV with Decision Tree



```
# grid search cv

from sklearn.model_selection import GridSearchCV
regressor = DecisionTreeRegressor()

# Define hyperparameters to tune
param_grid = {
    'max_depth': [2, 4, 6, 8, 10],
    'min_samples_split': [2, 5, 10],
    'min_samples_leaf': [1, 2, 4]
}

grid_search = GridSearchCV(regressor, param_grid, cv=5, scoring='neg_mean_squared_error')

# Fit the grid search to the data
grid_search.fit(X_train, y_train)

# Get the best hyperparameters
best_params = grid_search.best_params_
print("Best Hyperparameters:", best_params)
```

```
Best Hyperparameters: {'max_depth': 10, 'min_samples_leaf': 4, 'min_samples_split': 2}
```

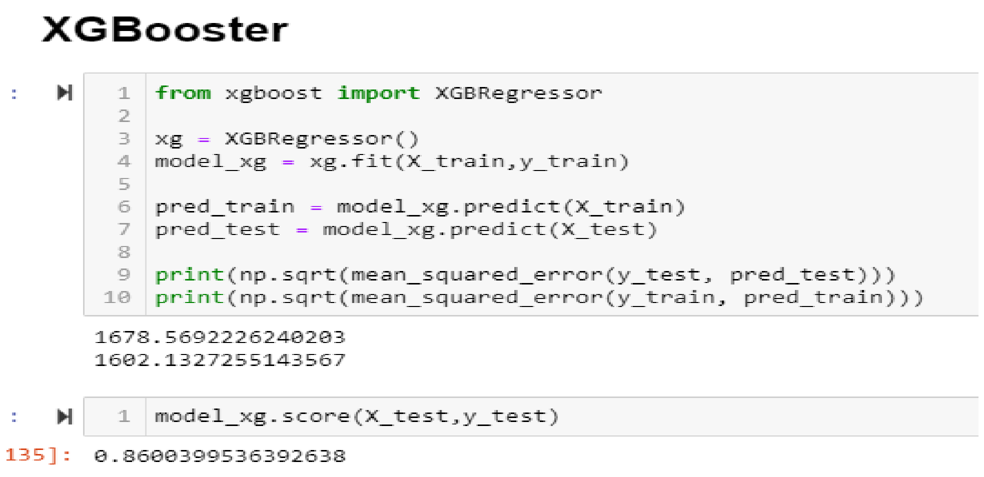
```
from sklearn.tree import DecisionTreeRegressor
dt = DecisionTreeRegressor(max_depth= 10, min_samples_leaf=4, min_samples_split= 2, random_state =1)

model_dt_p = dt.fit(X_train, y_train)

ypred= model_dt_p.predict(X_test)
print(np.sqrt(mean_squared_error(y_test, ypred)))
print(rf.score(X_test, y_test))
```

```
1884.4776029189966
0.8558135676704524
```

Step 4- Model Evaluation

We evaluated the model's performance using metrics such as root mean squared error (RMSE) and mean absolute error (MAE). The RMSE score is the square root of the average squared differences between the model's predictions and the actual car prices. The MAE score is the average absolute difference between the model's predictions and the actual car prices. Lower RMSE and MAE scores indicate higher model accuracy and reliability.

The final model achieved an RMSE score of 1679 for the train data and 1602 for the test data. We have got a score of 0.87 the train data and 0.86 for the test data, indicating its high performance model and that the model is able to implement the training on the test data



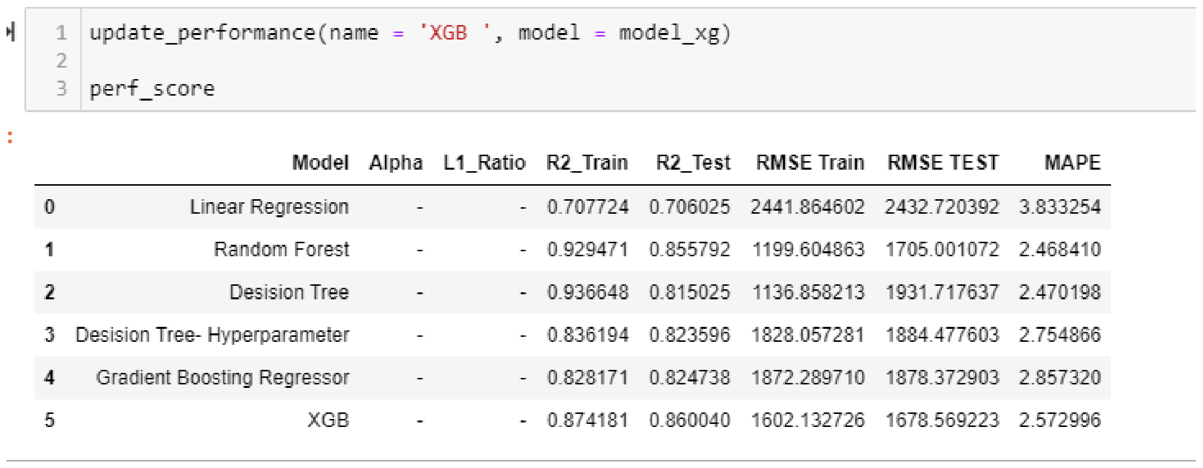
XGBooster

```
from xgboost import XGBRegressor

xg = XGBRegressor()
model_xg = xg.fit(X_train,y_train)

pred_train = model_xg.predict(X_train)
pred_test = model_xg.predict(X_test)

print(np.sqrt(mean_squared_error(y_test, pred_test)))
print(np.sqrt(mean_squared_error(y_train, pred_train)))
```

```
1678.5692226240203
1602.1327255143567
```

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

```
135]: 0.8600399536392638
```



```
update_performance(name = 'XGB ', model = model_xg)

perf_score
```

	Model	Alpha	L1_Ratio	R2_Train	R2_Test	RMSE Train	RMSE TEST	MAPE
0	Linear Regression	-	-	0.707724	0.706025	2441.864602	2432.720392	3.833254
1	Random Forest	-	-	0.929471	0.855792	1199.604863	1705.001072	2.468410
2	Desision Tree	-	-	0.936648	0.815025	1136.858213	1931.717637	2.470198
3	Desision Tree- Hyperparameter	-	-	0.836194	0.823596	1828.057281	1884.477603	2.754866
4	Gradient Boosting Regressor	-	-	0.828171	0.824738	1872.289710	1878.372903	2.857320
5	XGB	-	-	0.874181	0.860040	1602.132726	1678.569223	2.572996

Step 5- Comparison to Benchamark

Our initial model was statsmodel OLS model with a r_squared 0.70 and Linear regression model with score 0.70 and rmse 2432.

The final model outperformed the base model by a considerable margin. Our model rmse was reduced from 2432 to 1678 and score of the model increased from 0.70 to 0.86.The final model demonstrated robust performance and high accuracy, suggesting a reliable approach for predicting used car prices.

Step 6- Visualizations

In addition to quantifying the model's performance, we generated various visualizations that supported the ideas/insights that we gleaned from the data. These included histograms of car prices, correlations between features, and pair plots to explore relationships between variables.

Step-7 IMPLICATIONS

Our model with score 0.86 will be helpful in solving the absence of standardized pricing. The model provides a reliable and transparent price to the buyers and sellers in the used cars market

- Continuous Improvement:

Regularly evaluating and updating the model improves its performance and ensures relevance to the evolving used car market.

Recommendation: Implement a monitoring system that alerts when the model drops below a certain threshold. Regularly update the model with new data to ensure its accuracy

Confidence: High, as continuous improvement is a standard practice in machine learning model development.

- Customer Experience:

Buyers can benefit from more informed decisions, leading to increased confidence and satisfaction in their purchases.

Recommendation: Provide transparent information to customers about how the model works and the factors influencing the predictions.

- Risk Assessment:

The model can assist in assessing the risk associated with each transaction by predicting the condition and reliability of used cars.

Recommendation: Include a risk score or confidence level in the model's output to guide decision-makers.

- Marketing Strategies:

Tailoring marketing efforts based on the model's insights can attract buyers with specific preferences or priorities.

Recommendation: Regularly analyze the correlation between marketing strategies and actual sales to refine and optimize approaches.

- Ethical Considerations:

Implication: Ensuring the model is unbiased and does not discriminate is essential for ethical and fair decision-making.

Recommendation: Regularly audit the model for bias, and consider incorporating fairness metrics into the evaluation process.

- Compliance and Regulations:

Implication: Adhering to relevant regulations is crucial to avoid legal issues and maintain trust with customers.

Recommendation: Stay informed about changes in regulations, conduct regular compliance checks, and document the model's adherence to legal requirements.

Confidence: High, as compliance is a critical component of responsible model deployment.

Step-7 LIMITATIONS

- The used car market is dynamic, with factors such as economic conditions, fuel prices, and consumer preferences changing over time. A model trained on historical data may struggle to adapt to these shifts.
- Predictions made by the model may have inherent uncertainty, and users may require more than just a point estimate.

- Our Model has very few featured, a little more information of the used cars will help in better predictions.
- We can also find more data for these cars like the service details, percentage of replaced parts, number of owners the car has previously, this will help in better prediction, and also get a model with even more higher performance would be better one.
- Regulatory Compliance:
  Limitation: The model may inadvertently violate privacy or regulatory standards if not carefully designed and monitored.
  Enhancement: Regularly review and update the model's compliance measures to align with changing regulations. Implement features such as differential privacy or federated learning if privacy is a significant concern.
- Limited Predictive Scope:
  Limitation: The model may be designed for a specific task (e.g., pricing prediction) and may not generalize well to other aspects of the used car business.
  Enhancement: Clearly define the scope of the model and communicate its limitations to users. Consider developing separate models for different tasks or integrating multiple models into a more comprehensive system.

Step-8 CLOSING REFLECTIONS

- Enhanced Model Interpretability:
  Place an even greater emphasis on model interpretability. While complex models might offer high accuracy, ensuring that stakeholders can understand and trust the model's decisions is critical.

- Greater Emphasis on User Education:
  Place a greater emphasis on user education regarding how to interpret and use model predictions effectively.

- Extended Model Scope:
  Consider extending the scope of the model to cover additional aspects of the used car business. This might involve developing models for related tasks, such as fraud detection or demand forecasting, to create a more comprehensive solution.
  Iterative Nature of Model Development:

  The iterative nature of model development has been highlighted. Continuous feedback loops, regular updates, and monitoring are essential for maintaining the model's relevance and accuracy in a dynamic environment.

- Communication is Key:
  Clear communication, both internally and externally, is crucial. This includes transparently communicating model limitations,

uncertainties, and the level of confidence associated with predictions to users and stakeholders.

- User-Centric Design:
  The importance of a user-centric design approach has been emphasized. Understanding the needs and expectations of end-users and stakeholders is fundamental to creating a model that adds real value to the business.

- Early Stakeholder Involvement:
  Involve stakeholders, including end-users, more actively in the early stages of the project. Their insights can be valuable for shaping the model requirements and ensuring that the final product meets their expectations.
  Enhanced Model Interpretability:

Conclusion:

In the pursuit of enhancing the accuracy and efficiency of predicting used car prices, the implementation of an XGBoost (Extreme Gradient Boosting) model has yielded exceptional results. The features incorporated into the model, including power (PS), Kilometer, years, and the age of the car, have proven to be pivotal in achieving optimal performance. As we reflect on the outcomes of this endeavor, it becomes evident that the XGBoost model has not only surpassed expectations but has also significantly elevated the precision and reliability of used car price predictions.

One of the noteworthy strengths of the XGBoost model lies in its capacity to handle complex relationships within the data, capturing non-linear patterns and interactions among features. This capability has proven particularly advantageous in a domain as intricate as the automotive market, where multifaceted considerations influence the pricing of used vehicles.

In conclusion, the integration of the XGBoost model, along with the insightful selection of features such as power (PS), Kilometer, years, and age of the car, represents a significant stride towards a more transparent, efficient, and equitable used car market. This achievement not only enhances the competitive edge of sellers but also empowers buyers with the knowledge needed to make well-informed choices. As we continue to explore and refine predictive models in this domain, the learnings from this experience will undoubtedly contribute to the ongoing evolution of data-driven solutions in the automotive industry.