CAR PRICE PREDICTION

Submitted by:

Komal Vijay Ghatvilkar

ACKNOWLEDGMENT

The success & outcome of this project were possible by the guidance and support from FlipRobo.

It was not possible to done without research from different machine learning sites on Google.

I referred DataTrained material for more information that helped me completion of the project.

Thank you.....!!!

The price of a new cars in the industry is fixed by the manufacturer with some additional costs incurred by the Government in the form of taxes. So, customers buying a new car can be assured of the money they invest to be worthy. But, due to the increased prices of new cars and the financial incapability of the customers to buy them, Used Car sales are on a global increase. Therefore, there is a need for a Used Car Price Prediction system which effectively determines the worthiness of the car using a variety of features.

Existing System includes a process where a seller decides a price randomly and buyer has no idea about the car and it's value in the present day scenario. In fact, seller also has no idea about the car's existing value or the price he/she should be selling the car at.

To overcome this problem we have developed a model which will be highly effective. Regression Algorithms are used to predict the actual price a car rather than the price range of a car.

Determining whether the listed price of a used car is a challenging task, due to the many factors that drive a used vehicle's price on the market. The focus of this project is developing machine learning models that can accurately predict the price of a used car based on its features, in order to make informed purchases. We implement and evaluate various learning methods on a dataset consisting of the sale prices of different makes and models . Depending on various parameters we will determine the price of the car. Regression Algorithms are used because they provide us with continuous value as an output and not a categorized value because of which it will be possible to predict the actual price a car rather than the price range of a car.

I have collected the data from one of the sites Cars24 which deals in car selling and purchasing activities for all over India.

We have done the following analysis of the dataset where we Imported necessary libraries so that we can work on datasets with the Jupyter notebook.

```
- import numpy as np
- import pandas as pd
- import matplotlib.pyplot as plt
- import seaborn as sns
- import warnings
- warnings.filterwarnings('ignore')
```

Data contains 5765 entries each having 11 variables. After reading the dataset I proceed with the EDA.

```
- df=pd.read_excel(r'C:\Users\HP\Desktop\CarPricePredictionData.xlsx')
```

I checked the description of data with .info() method & find null values contains in dataset.

```
- df.info()
- df.isnull().sum()
```

I removed unwanted or unnecessary columns to perform further tasks.

```
- df=df.drop(['Unnamed: 0','Downpayment'],axis=1)
```

To perform further task tried with conversion of String data into Integers for equalize the data type for process with LabelEncoder.

```
- from sklearn.preprocessing import LabelEncoder
- LE=LabelEncoder()
- df['Brand']=LE.fit_transform(df['Brand'])
- df['Model']=LE.fit_transform(df['Model'])
- df['Transmission']=LE.fit_transform(df['Transmission'])
- df['Owner']=LE.fit_transform(df['Owner'])
- df['Feul']=LE.fit_transform(df['Feul'])
- df['Location']=LE.fit_transform(df['Location'])
```

After that I removed unwanted strings from some of the columns and converted them into Integer as showing below.

```
- df["Kilometers Drive Till Now"]=df["Kilometers Drive Till Now"].str.replace(" km","")
- df["EMI"]=df["EMI"].str.replace("/month","")
- df["EMI"]=df["EMI"].str.replace(",","")
- df["Price Of Car"]=df["Price Of Car"].str.replace(",","")
- df["Kilometers Drive Till Now"]=df["Kilometers Drive Till Now"].str.replace(",","")
- df['Kilometers Drive Till Now']=df['Kilometers Drive Till Now'].astype(str).astype(int)
- df['EMI']=df['EMI'].astype(str).astype(int)
- df['Price Of Car']=df['Price Of Car'].astype(str).astype(int)
```

Found some null values so performed the task and convert or remove the Nan values with mean.

```
- df['Transmission']=df['Transmission'].fillna(df['Transmission'].mean())
```

After .describe() done found the statistical description of data & with the correlation among all the columns checked the correlation and found some of the data is positively correlated and some is negatively correlated with each other.

```
- df.describe()
- df.corr()
```

In data visualization done the following visualizations :

First used Correlation Matrix for showing the correlation between all columns with Heatmap.

From the output of correlation matrix, we can see that it is symmetrical i.e. the bottom left is same as the top right.

```
corr_mat=df.corr()
# Size of the canvas
plt.figure(figsize=[20,20])
#Plot Correlation Matrix
sns.heatmap(corr_mat,annot=True) # annot represnts each value encoded in heatmap
plt.title('Correlation Matrix')
plt.show()
```

The result of the Correlation Matrix is on following GitHub link.

https://github.com/komalghatvilkar/Internship/blob/main/Car%20Price%20Prediction%20Project/Correlation%20Matrix.png

Second, used Density plot & Histogram both for all dataset for visualizing the data individually.

The visualization plot shows that each variable distributed differently and as we can see some data has categorical values, so used histogram also for better understanding to show the distribution.

```
df.plot(kind='density',subplots=True,layout=(10,15),sharex=False,fontsize=1,figsize=(40,30))
plt.show()
df.hist(bins=30,figsize=(40,40))
plt.show()
```

The result of the Density plot & Histogram is on following GitHub link.

https://github.com/komalghatvilkar/Internship/blob/main/Car%20Price%20Prediction%20Project/Density%20Plot.png

https://github.com/komalghatvilkar/Internship/blob/main/Car%20Price%20Prediction%20Project/Histogram.png

Then I checked the skewness of the data. After that checked the outliers if any, found many so tried to remove the outliers. I used boxplot to check the outliers in dataset.

```
# Checking Skweness
df.skew().sort_values(ascending=False) # For descending
# Checking Outliers in data
# Plot boxplot
df.boxplot(figsize=[10,10])
plt.subplots_adjust(bottom=0.25)
plt.show()
from scipy.stats import zscore
z=np.abs(zscore(df))
z.shape
threshold=2.5
print(np.where(z>2.5))
df_new=df[(z<2.5).all(axis=1)]
print(df.shape)
print(df_new.shape)
# % data loss
```

```
- loss_percent=(5765-5162)/5765*100
- print(loss_percent)
- # Checking Outliers in data
- # Plot boxplot
- df_new.boxplot(figsize=[20,15])
- plt.subplots_adjust(bottom=0.25)
- plt.show()
```

The result of the Boxplot is on following GitHub link.

https://github.com/komalghatvilkar/Internship/blob/main/Car%20Price%20Prediction%20Project/Outliers.png

After checking the outliers We split the data into x & y. I have taken the output as the column “Price Of Car” because our problem statement is to find the actual prices of cars.

```
- x=df_new.drop(['Price Of Car'],axis=1)
- y=df_new['Price Of Car']
```

And then did the Train Test Split and find the best accuracy & random state which gives me the following results :-

```
- # Training process
- # Min-max scaler
- from sklearn.preprocessing import MinMaxScaler
- mms=MinMaxScaler()
- from sklearn.linear_model import LinearRegression
- lr=LinearRegression()
- from sklearn.metrics import r2_score
- from sklearn.model_selection import train_test_split
- for i in range(0,100):
-
  x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=.20,random_state=i)
-         lr.fit(x_train,y_train) # Fitting the data will train the model
-         pred_train=lr.predict(x_train) # Predicting the data # Predicted traget variable
-         pred_test=lr.predict(x_test)
-         print(f'At Random State {i}, the training accuracy is :- {r2_score(y_train,pred_train)}')
-         print(f'At Random State {i}, the training accuracy is :- {r2_score(y_test,pred_test)}')
-         print("\n")
```

At Random State 97, the training accuracy is :- 0.9999999965034685 At Random State 97, the training accuracy is :- 0.9999999964589724

At Random State 98, the training accuracy is :- 0.9999999964836568 At Random State 98, the training accuracy is :- 0.9999999965404449

At Random State 99, the training accuracy is :- 0.9999999964487212 At Random State 99, the training accuracy is :- 0.9999999966792197

With the best suitable model building for regression dataset that is Linear Regression performed The task.

Next did Cross Validation with “cross_val_score” for the models used & it shows the output :-

```
# Cross Validation
Train_accuracy = r2_score(y_train,pred_train)
Test_accuracy = r2_score(y_test,pred_test)
from sklearn.model_selection import cross_val_score
for j in range(5,20):
    cv_score=cross_val_score(lr,x,y,cv=j)
    cv_mean=cv_score.mean()
    print(f'At cross fold {j} the cv score is  {cv_mean} and accuracy score for training is {Train_accuracy} and accuracy score for testing is {Test_accuracy}')
    print('\n')
```

At cross fold 17 the cv score is 0.9999999957436349 and accuracy score for training is 0.9999999964487212 and accuracy score for testing is 0.9999999966792197

At cross fold 18 the cv score is 0.9999999957911245 and accuracy score for training is 0.9999999964487212 and accuracy score for testing is 0.9999999966792197

At cross fold 19 the cv score is 0.9999999957019833 and accuracy score for training is 0.9999999964487212 and accuracy score for testing is 0.9999999966792197

Then with the matplotlib.pyplot showed the distribution of data & the result shows best fit line & relationship between two variables.

```
import matplotlib.pyplot as plt
plt.figure(figsize=(8,6))
plt.scatter(x=y_test,y=pred_test,color='blue')
plt.plot(y_test,y_test,color='yellow')
plt.xlabel('Actual price', fontsize=14)
plt.ylabel('Predicted price', fontsize=14)
plt.title('Linear Regression', fontsize=18)
plt.show()
```

Then checked the regularization with GridSearchCV & With Lasso technique to perform regularization in order to enhance the prediction accuracy and interpretability of the resulting statistical model and found

```
# Regularization
from sklearn.model_selection import GridSearchCV
from sklearn.model_selection import cross_val_score
from sklearn.linear_model import Lasso
parameters={'alpha':[.0001,.001,.01,.1,1,10],'random_state':list(range(0,20))}
```

- ls=Lasso()
- clf=GridSearchCV(ls,parameters)
- clf.fit(x_train,y_train)
- print(clf.best_params_)
- ls=Lasso(alpha=10,random_state=0)
- ls.fit(x_train,y_train)
- ls.score(x_train,y_train)
- pred_ls=ls.predict(x_test)
- lss=r2_score(y_test,pred_ls)
- Lss
- cv_score=cross_val_score(ls,x,y,cv=5)
- cv_mean=cv_score.mean()
- cv_mean*100

With AdaBoostRegressor assemble technique to achieve best accuracy of dataset.

- # Ensemble Technique
- from sklearn.ensemble import AdaBoostRegressor
- AD = AdaBoostRegressor()
- AD.fit(x_train,y_train)
- AD.score(x_test,y_test)

Finally came to the Conclusion as per the results found those are the model is showing the exact result for the dataset with 99% accuracy.

Please find the GitHub links for Pyplot of Linear regression to refer.

https://github.com/komalghatvilkar/Internship/blob/main/Car%20Price%20Prediction%20Project/Regression%20plot.png

Please find the GitHub links for Jupyter Notebook Solution of web scraping of data collected to refer.

https://github.com/komalghatvilkar/Internship/blob/main/Car%20Price%20Prediction%20Project/Car%20Price%20Prediction%20-%20Data%20Collection.ipynb

Please find the GitHub links for Jupyter Notebook Solution of dataset to refer.

https://github.com/komalghatvilkar/Internship/blob/main/Car%20Price%20Prediction%20Project/Car%20Price%20Prediction%20Project.ipynb

The results shows that the dataset is correct 99% & we can proceed with the data accordingly.

Thank you...!!