Car Price Prediction of used Cars

Introduction

Problem Statement

New cars cannot be afforded by most of the people due to various factors, so all these customers opt for a used car with best features due to which the demand for used cars is always increasing. To stay in the market competition we should always offer a best price which is best to both to customer and the owner. The model we are going to create would help us in identifying the price of the car based the parameters that we would be passing would help us tag a better price on the car.

Technical Approach

The Cross-Industry Standard Process for Data Mining (CRISP-DM). This process model has 6 phases that naturally describe the data science life cycle for this project.

1. Business Understanding
2. Data Understanding
3. Data Preparation
4. Modeling
5. Evaluation
6. Deployment

During every phase of this project lifecycle, we might discover new aspects/finding which we will incorporate them in ways to improve the efficiency of our model.

Data Sources

The dataset that is selected has all the details that are needed for a model to identify the car price

- Model
- Year(Manufacturing year)
- Price
- Transmission(Automatic/Manual)
- Mileage
- FuelType(Petrol/Diesel)
- Tax
- MPG(Miles Per Gallon)
- EngineSize
- Make(Manufacturer)

Analysis

We are planning to perform feature reduction and dimensionality reduction to select the most relevant variables for our model. Many predictive algorithms assume the model variables follow a normal distribution. There are inherent advantages to using normally distributed variables, so our approach will focus on columns that closely follow this distribution. We will determine which variables are normally distributed by conducting the following analyses:

- Summary statistics on all variables: concentrating on mean and SD values.
- Identify the best model.

Requirement Development

For this project, we are planning to use python. To complete this project in python the following are required for our development:

IDE: Jupyter Notebook

Libraries:

1. Numpy – extensively used for data analysis to handle multidimensional arrays.
2. Pandas – high-performance data structures and analysis tools for the labeled data.
3. Matplotlib –powerful visualizations which create several stories with the data visualized.
4. SciPy – used for high-level technical computations.
5. Seaborn – interface for drawing attractive and informative statistical graphics.

Model Deployment

We will use feature selection techniques to finalize our feature list for models. Using the results from this step, we will build a couple of classification models and evaluate their performance. Below are some example models.

- **Random forest:** It builds multiple decision trees and merges them to get a more accurate and stable prediction. One big advantage of random forest is that it can be used for both classification and regression problems, which form the majority of current machine learning systems.
- **Decision Tree:** It belongs to the family of supervised learning algorithms. It can be used for solving regression and classification problems. The goal of using a Decision Tree is to create a training model that can use to predict the class or value of the target variable by learning simple decision rules inferred from training data. In Decision Trees, for

predicting a class label for a record we start from the root of the tree. We compare the values of the root attribute with the record's attribute. Based on the comparison, we follow the branch corresponding to that value and jump to the next node.

Testing and Evaluation

We will be splitting the data into 70% training and 30% test dataset. Using the test dataset, we will test the model. confusion matrix, AUC, and F1 scores will be used for the evaluation. cross-validation will also be used to decide the best model.

Expected Results

Using this model to determine the car price.

Ethical Considerations

Ethical considerations in research are a set of principles that guide our research designs and practices. We must always adhere to a certain code of conduct when collecting data from people. The goals of human research often include understanding real-life phenomena, studying effective treatments, investigating behaviors, and improving lives in other ways. What you decide to research and how you conduct that research involve key ethical considerations.

These considerations work to

- Protect the rights of research participants
- Enhance research validity
- Maintain scientific integrity
- Should not share the data provided by the customers to third parties without his consent

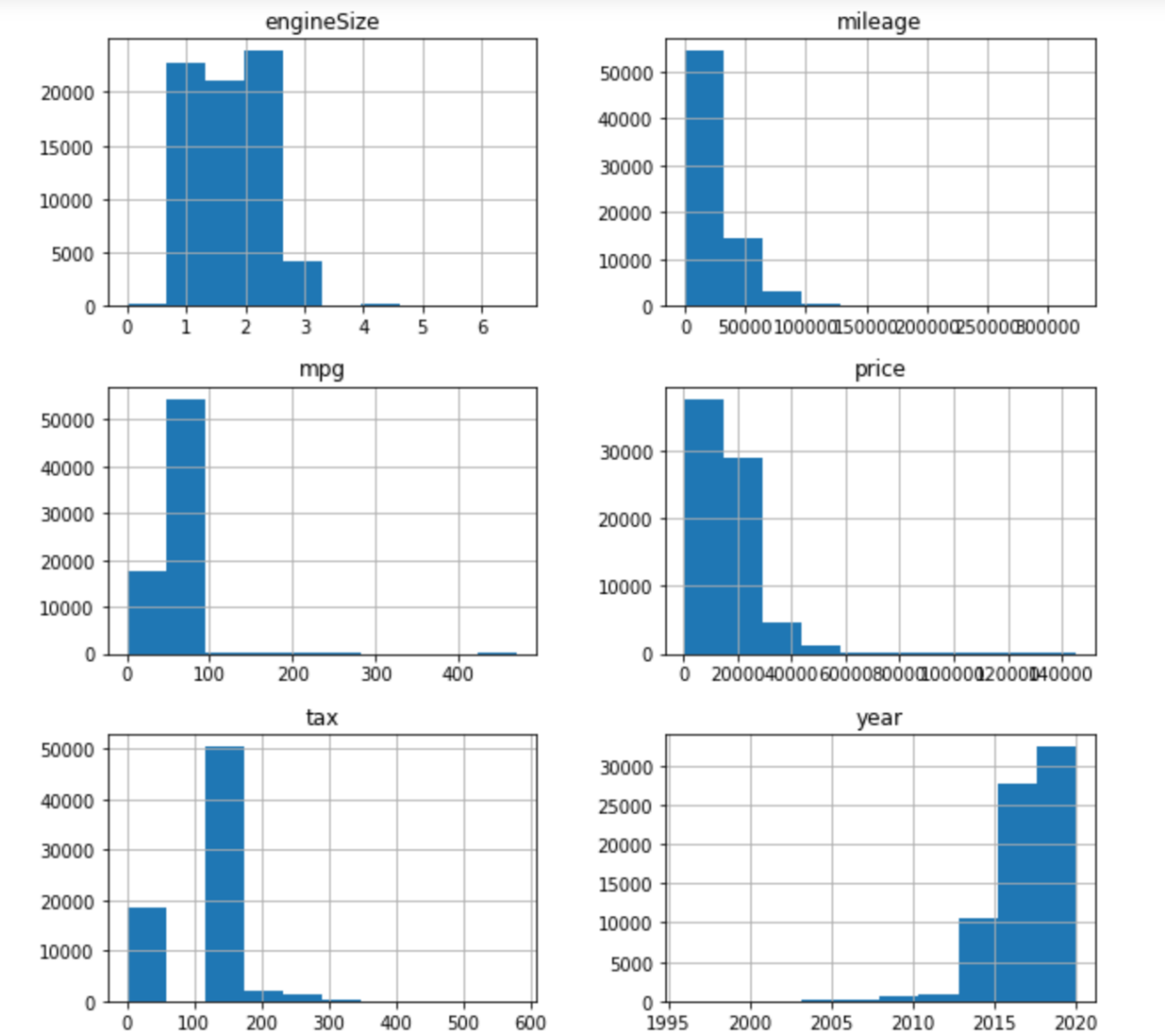
Challenges/Issues

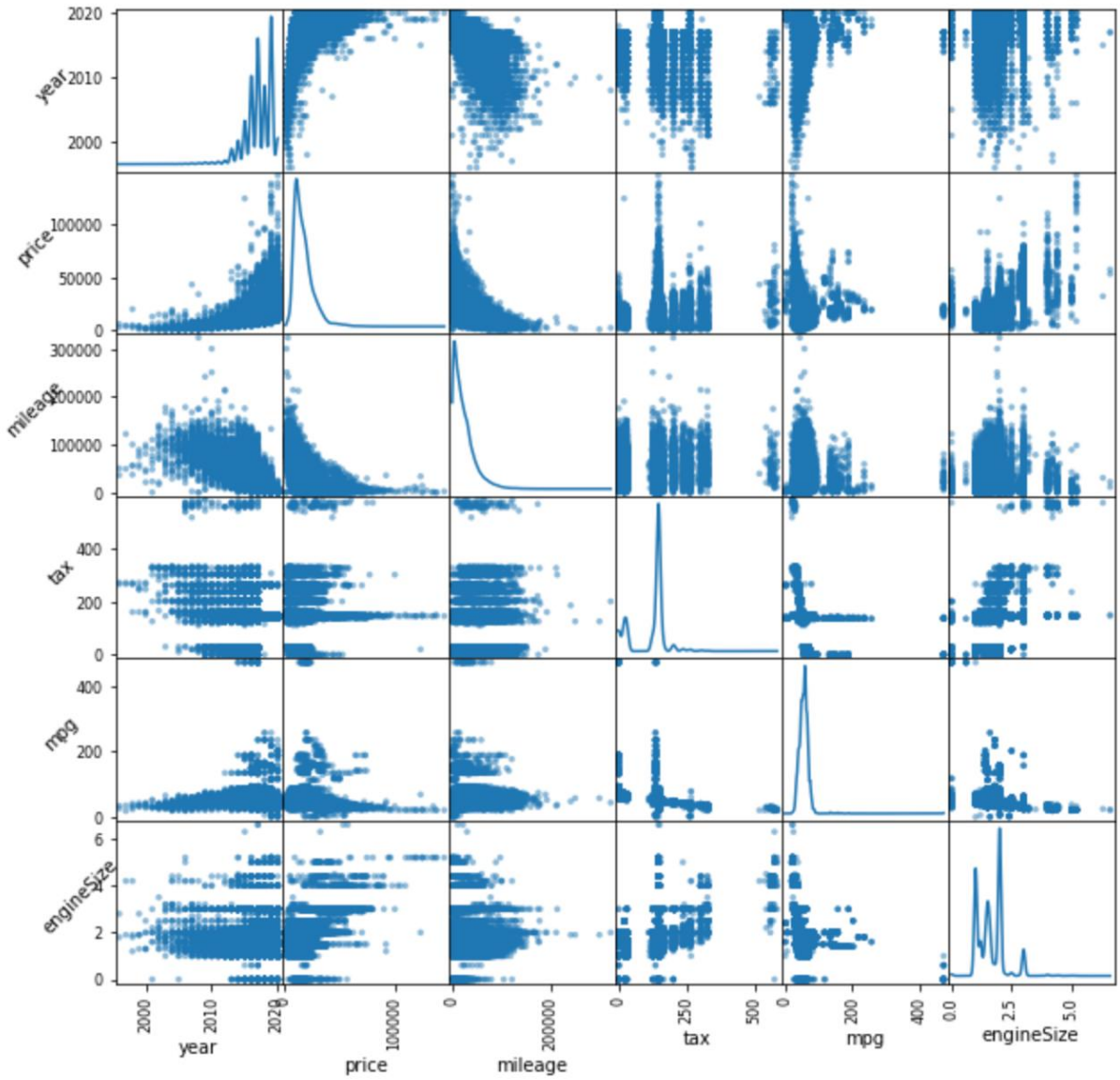
The biggest challenge is remove the features that are not required to improve the efficiency of the model.

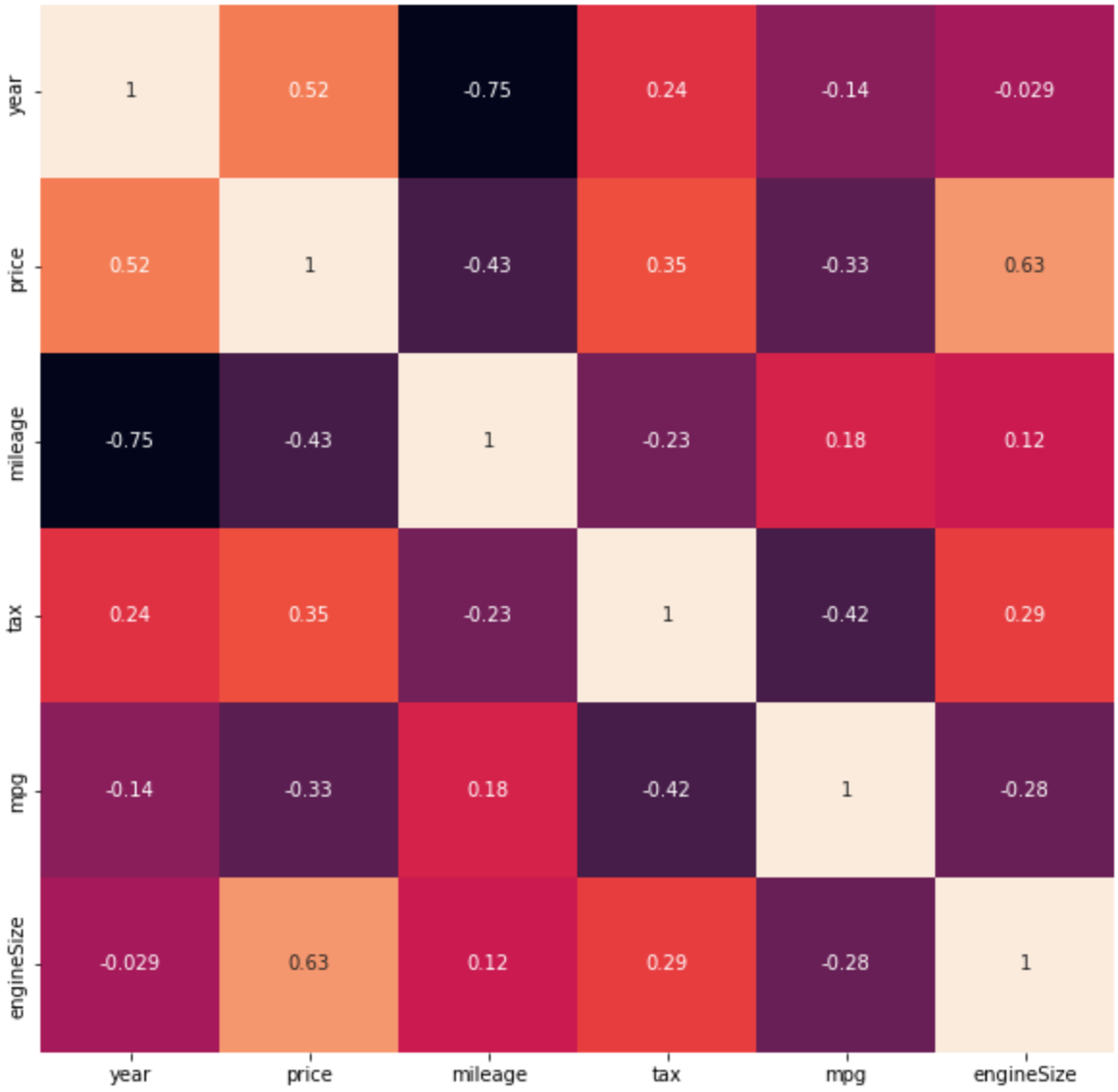
Reference:

- https://www.kaggle.com/aishwaryamuthukumar/cars-dataset-audi-bmw-ford-hyundai-skoda-vw
- https://www.scribbr.com/methodology/research-ethics/
- https://www.carmax.com/
- https://www.carvana.com/

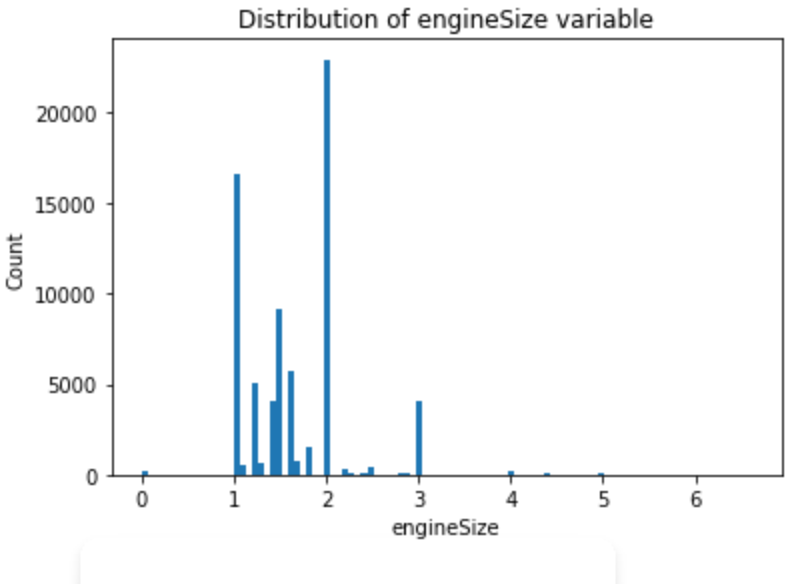
Illustrations:




year
price
mileage
tax
mpg
engineSize
2020
2010
2000
100000
50000
0
300000
200000
400
200
6
4
2
250
500
2.5
5.0


year
price
mileage
tax
mpg
engineSize
1
0.52
-0.75
0.24
-0.14
-0.029
0.52
1
-0.43
0.35
-0.33
0.63
-0.75
-0.43
1
-0.23
0.18
0.12
0.24
0.35
-0.23
1
-0.42
0.29
-0.14
-0.33
0.18
-0.42
1
-0.28
-0.029
0.63
0.12
0.29
-0.28
1
year
price
mileage
tax
mpg
engineSize

```
plt.hist(cars_df['engineSize'], bins=100)
plt.ylabel('Count')
plt.xlabel('engineSize')
plt.title('Distribution of engineSize variable');
```



We can see there is data where enginesize is 0 as based on the above histogram. A car without enginesize dosent make any sense. So, we can delete the data where enginesize is 0

Removing the 0 engine size which doesnt make any sense

```
]: cars_df = cars_df.drop(cars_df.loc[cars_df['engineSize'] < 1].index)
```

```
]: np.sum(cars_df['engineSize'] < 1)
```

```
t[10]: 0
```

We are reducing the size of the dataset based on the column year since the data is huge. We took a random sample from the dataset that we have in hand and dropping all the NA values from the dataset

```
sampleDataSetMain = pd.DataFrame()
sampleDataSet = pd.DataFrame()
count = 0

years = sorted(cars_df.year.unique())
for i in years:
    #print(i)
    count = sum(cars_df['year'].eq(i))
    if count > 75 :
                sam = 75
    else:
                sam = count
    #print('hi')
    sampleDataSet = cars_df[cars_df['year'].eq(i)].sample(n=sam, replace=True)
    #print(sampleDataSet)
    sampleDataSetMain = sampleDataSetMain.append(sampleDataSet)
    #print(sampleDataSetMain)

print(sampleDataSetMain.shape)
print(sampleDataSetMain.head())

sampleDataSetMain=sampleDataSetMain.dropna()
```

Our model is to predict the price of the used cars based on the dataset. So, price is target variable in our model and we are using year, mileage, tax, mpg, engineSize which are dependent variables for the model

```
#y = cars_df['price']
#x = cars_df.drop(columns=['price','model'])
#print(x)
y = sampleDataSetMain['price']
x = sampleDataSetMain.drop(columns=['price','model','fuelType','transmission', 'Make'])
print(x)
```

We split the train and test data with a sample size of 0.2

```
In [16]: x_train, x_test, y_train, y_test = train_test_split(x,y,test_size=0.2,random_state=42)
```

Below are the different methods we thought of to build a model. So, we will calculate the r2 score to determine which model has the best score and can be used for our project.

Model Evaluation

```
# define the data preparation and modeling pipeline
def pipeline_model(model):
    pipeline = Pipeline(steps=[('prep',col_transform), ('model', model)])
    return(pipeline)
```

```
cv = KFold(n_splits=3, shuffle=True, random_state=1)

models = [RandomForestRegressor(),
          SVR(),
          Ridge(),
          LinearRegression(),
          Lasso(),
          ElasticNet()
         ]

score_list = []
for _, model_ in enumerate(models):
    model = TransformedTargetRegressor(regressor=pipeline_model(model_),
                                       func=np.log1p, inverse_func=np.expm1)

    scores = cross_val_score(model, x, y, scoring='r2', cv=cv, n_jobs = -1, verbose = 2)

    final_score = np.mean(scores)

    score_list.append(final_score)
```

Random Forest Regressor is the best within all the models that we chose. So, we proceed with Random Forest Regressor

```
model_name_list = ['Random Forest Regressor',
           'Support Vector Machine - Regressor',
           'Ridge Regressor',
           'Linear Regression',
           'Lasso Regression',
           'ElasticNet Regression']

results = pd.DataFrame(
    {'Model type': model_name_list,
     'Mean Score (R^2)': score_list})

results
```

	Model type	Mean Score (R^2)
0	Random Forest Regressor	0.885987
1	Support Vector Machine - Regressor	0.877373
2	Ridge Regressor	0.851423
3	Linear Regression	0.850841
4	Lasso Regression	-0.104605
5	ElasticNet Regression	0.043313

With the dataset that we have here is providing good results with a score of 90% using the Random Forest Regressor.

```
pipeline = Pipeline(steps=[('prep',col_transform), ('model', RandomForestRegressor())])

model = TransformedTargetRegressor(regressor=pipeline, func=np.log1p, inverse_func=np.expm1)

model.fit(x_train, y_train)
```

```
y_pred = model.predict(x_test)
y_test = y_test.to_numpy()

r2_score(y_pred, y_test)
```

```
0.9024168512589992
```

Since the accuracy score is on the higher side we can use this model to predict the car price of the used cards with a given data.

Appendix:

- **Random forest:** It builds multiple decision trees and merges them to get a more accurate and stable prediction. One big advantage of random forest is that it can be used for both classification and regression problems, which form the majority of current machine learning systems.
- **Logistic regression:** It is the appropriate regression analysis to conduct when the dependent variable is binary. Logistic regression is used to describe data and to explain the relationship between one dependent binary variable and one or more nominal, ordinal, interval, or ratio-level independent variables.
- **Linear Regression:** Linear regression is a linear approach for modelling the relationship between a scalar response and one or more explanatory variables. The case of one

explanatory variable is called simple linear regression; for more than one, the process is called multiple linear regression.

- **Random Forest Regression:** Random Forest Regression is a supervised learning algorithm that uses ensemble learning method for regression. A Random Forest operates by constructing several decision trees during training time and outputting the mean of the classes as the prediction of all the trees.
- **Decision Tree:** It belongs to the family of supervised learning algorithms. It can be used for solving regression and classification problems. The goal of using a Decision Tree is to create a training model that can use to predict the class or value of the target variable by learning simple decision rules inferred from training data. In Decision Trees, for predicting a class label for a record we start from the root of the tree. We compare the values of the root attribute with the record's attribute. Based on the comparison, we follow the branch corresponding to that value and jump to the next node.
- Pandas DataFrame is two-dimensional size-mutable, potentially heterogeneous tabular data structure with labeled axes (rows and columns). A Data frame is a two-dimensional data structure, i.e., data is aligned in a tabular fashion in rows and columns. Pandas DataFrame consists of three principal components, the data, rows, and columns.
- A numpy array is a grid of values, all of the same type, and is indexed by a tuple of nonnegative integers. The number of dimensions is the rank of the array; the shape of an array is a tuple of integers giving the size of the array along each dimension.