ECE 9063 Data Analytics Foundations

Assignment 1: Forecasting

Student Name: Jianping Ye

Student Number: 250887769

Instructor: Katarina Grolinger

Problem Statement

The used car market is a perfect place for finding cars in decent conditions and with fair prices. It is also the reason that the market has been growing in recent years. However, it is difficult to choose the opportune moment to buy or sell as the price fluctuates constantly. And there are many factors contributing to the price fluctuations. For instance, cars have diverse conditions and the market trend is not stationary all the time. It will be beneficial for both buyers and sellers if we could make a model to predict the value of cars such that they can make a more confident decision. With the help of a suitable model, buyers will be able to make sure the car is worthy of its price, and sellers can get a more accurate price estimation in accordance with other cars having similar conditions. In this report, the forecasting problem is defined as follow: predict the price of a used car in the current year given a set of relevant information.

Dataset Description

Link to the data: https://www.kaggle.com/adityadesai13/used-car-dataset-ford-and-mercedes

These datasets list scraped data of used cars in the British market and are separated into files specific for each car manufacturer. In this report, the dataset selected is “Audi.csv”. It contains 9 attributes and 10668 samples. The dataset is suitable for this assignment as it has adequate attributes and samples. With over 10,000 samples, it is easier to strike a balance between computational time and reliability of the model . The attributes are listed below:

- Model: The model code of the car
- Year: registration year of the car
- Price: price on the market
- Transmission: type of gearbox, either manual, automatic, or semi-auto
- Mileage: distance used so far
- fuelType: type of fuel the engine uses, either diesel, petrol, hybrid, or other
- tax: road tax
- mpg: miles per gallon
- engineSize: size of engine in litres

Noticeably, model, transmission, and fuelType have nominal data that needs to be transformed into numerical values. All the attributes in the dataset are considered in the model as they are all important factors while estimating the price of cars in the real-world.

Algorithms Overview

The first algorithm is support vector regression (SVR). Support vector regression adheres to the basic principle of support vector machine, which is the maximum margin characteristic, but it is used for regression instead. Linear kernel is used.

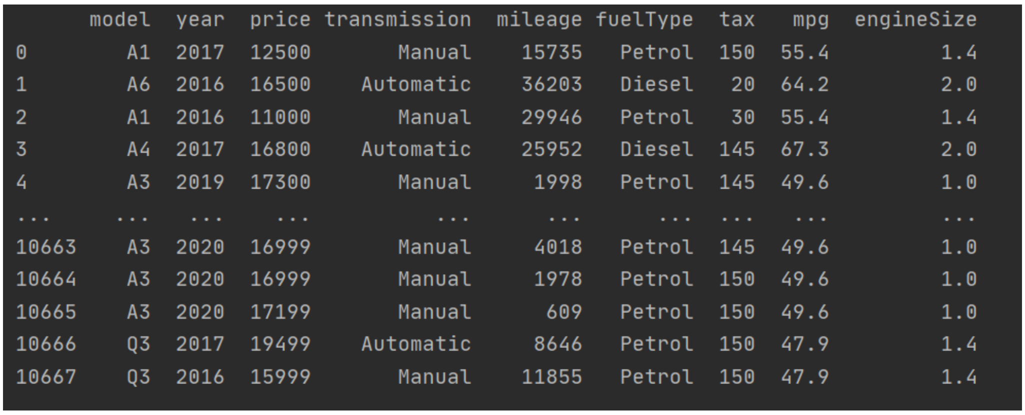
The second algorithm is decision tree regression. The algorithm proceeds incrementally as breaking down the data into smaller subsets and build the associated sub-trees from them. At the end, a tree structure with decision nodes and leaf nodes is constructed. However, one major issue with decision tree regression is that it is very prone to overfitting. At the result comparison section, we will inspect whether this problem arises.

The third algorithm is random forest regression. Random forest regression utilizes the idea of ensemble learning, which is a technique that can take advantages from multiple machine learning algorithms such that it can produce a more accurate prediction.

Before applying the above-mentioned algorithms, it is necessary to normalize our dataset as it is a common requirement of many machine learning algorithms and it is also considered good practice. The normalization technique used is standardization, which will make the data have zero mean and unit variance.

Detailed Procedures

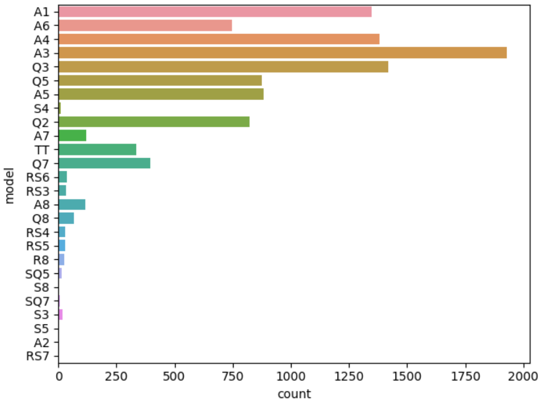
1. Exploratory Data Analysis

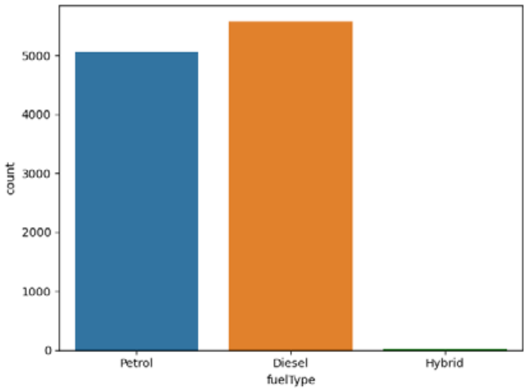


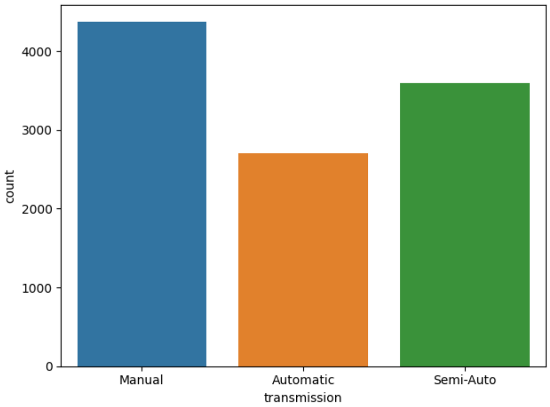
```
      model  year  price transmission  mileage fuelType  tax   mpg  engineSize
0        A1  2017  12500       Manual    15735   Petrol  150  55.4         1.4
1        A6  2016  16500    Automatic    36203   Diesel   20  64.2         2.0
2        A1  2016  11000       Manual    29946   Petrol   30  55.4         1.4
3        A4  2017  16800    Automatic    25952   Diesel  145  67.3         2.0
4        A3  2019  17300       Manual     1998   Petrol  145  49.6         1.0
...     ...   ...    ...          ...      ...      ...  ...   ...         ...
10663    A3  2020  16999       Manual     4018   Petrol  145  49.6         1.0
10664    A3  2020  16999       Manual     1978   Petrol  150  49.6         1.0
10665    A3  2020  17199       Manual      609   Petrol  150  49.6         1.0
10666    Q3  2017  19499    Automatic     8646   Petrol  150  47.9         1.4
10667    Q3  2016  15999       Manual    11855   Petrol  150  47.9         1.4
```

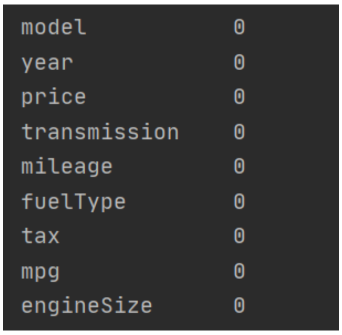
After importing the csv file, I printed the data to have an overview of its structure.

Clearly, model, transmission, and fuelType has nominal values. I plotted the counts of each type of observation versus that attribute.



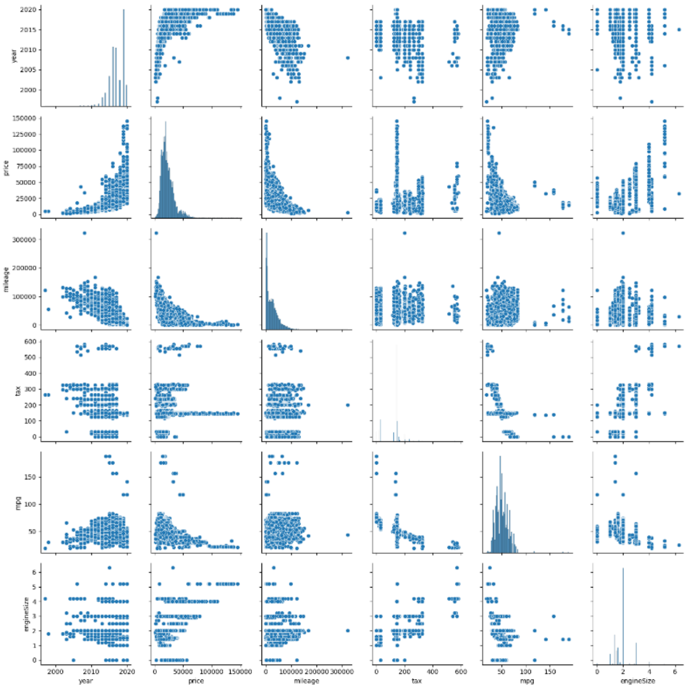






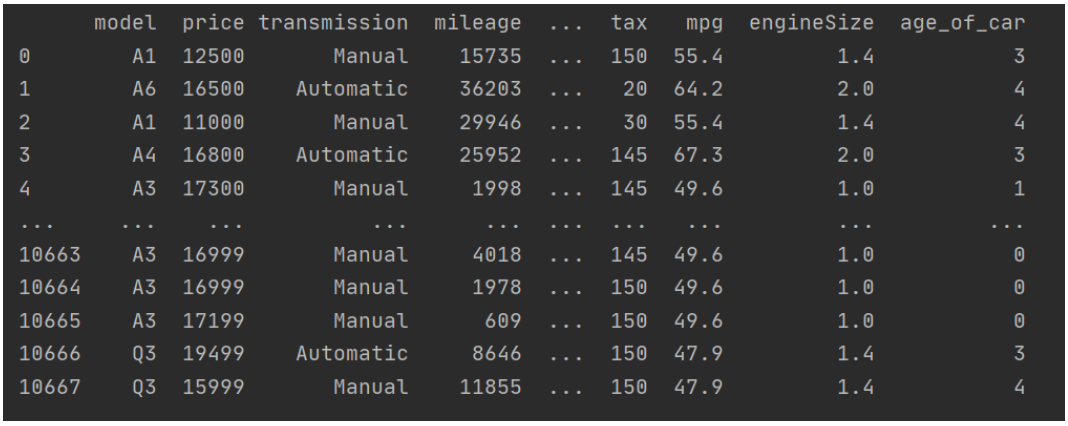
```
model           0
year            0
price           0
transmission    0
mileage         0
fuelType        0
tax             0
mpg             0
engineSize      0
```

It is observed that there are finite number of possibilities in each attribute, so one-hot encoding is selected as encoding scheme. And there is no null data in each attribute, which is very clean.



From the pair plot graph, we can see the data is dense (No zeros). Therefore, standardization will be used to normalize the data.

2. Preprocessing



```
       model  price transmission  mileage  ...  tax   mpg  engineSize  age_of_car
0         A1  12500       Manual    15735  ...  150  55.4         1.4           3
1         A6  16500    Automatic    36203  ...   20  64.2         2.0           4
2         A1  11000       Manual    29946  ...   30  55.4         1.4           4
3         A4  16800    Automatic    25952  ...  145  67.3         2.0           3
4         A3  17300       Manual     1998  ...  145  49.6         1.0           1
...      ...    ...          ...      ...  ...  ...   ...         ...         ...
10663     A3  16999       Manual     4018  ...  145  49.6         1.0           0
10664     A3  16999       Manual     1978  ...  150  49.6         1.0           0
10665     A3  17199       Manual      609  ...  150  49.6         1.0           0
10666     Q3  19499    Automatic     8646  ...  150  47.9         1.4           3
10667     Q3  15999       Manual    11855  ...  150  47.9         1.4           4
```

Firstly, compute a new attribute called “age_of_car” by subtracting 2020 from the ‘year’ attribute. The result is number of years that car has been used since its registration. I believe the ‘age of car’ will be more informative and intuitive than its original. Secondly, use one-hot encoding to encode categorical values. Now, we can separate the dataset into X (features) and Y (target) set. X set contains all the attributes except ‘price’. Y set is the price attribute. At this stage, we can split X and Y into X_train, Y_train, X_test, and Y_test. Finally, fit the standard scaler to X_train, then use the fitted scaler to transform both the training data and testing data (and new data in the future). This is to prevent data snooping.

3. Modeling

By using sklearn, we can easily create regression model to fit the training data. When creating the regression model instance, all the optional arguments are set to default to keep simplicity. All three algorithms will expect two arguments to pass into the function, one is X_train and the other is Y_train. This corresponds to the idea of train the model on the training set. After fitting the model, we can compute the fitted value of X_train by applying the learned parameters, then compare the fitted results with the target values, Y_train.

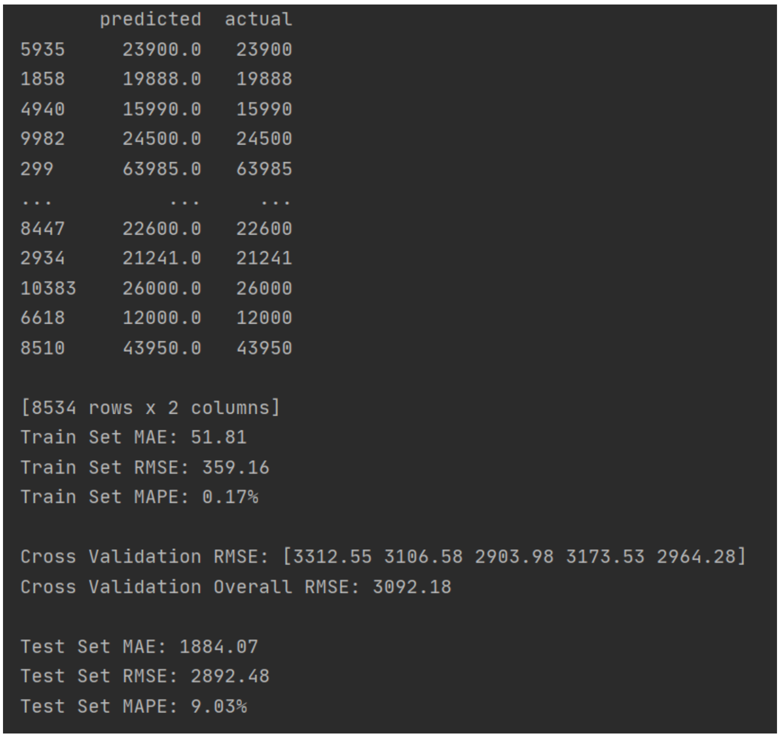
4. Accuracy & Evaluation

Different metrics are applied to reveal how well the models fit the data. Firstly, I will compute the mean absolute error (MAE). The rational is that MAE gives an intuitive measurement of the distance between predicted and actual values. To change the perspective, I also used the mean absolute percentage error (MAPE) to show the error as percentage values. However, these two metrics do not penalize errors that are bigger than others. Therefore, root mean squared error (RMSE) will be our primary error metric. In RMSE, bigger errors are penalized much heavier than the smaller errors as the error is squared.

The error metrics will be calculated on training set first. Then, 5-fold cross validation is used to validate the model before we launch the test set. The training data is split into 5 subsets and in each cross-validation iteration, one non-repeatable subset is selected as validation and the other 4 subsets is used for training. The validation error (measured in RMSE) in each iteration is stored to allow computation of overall error for the 5 folds by simply taking average value. After the cross validation, we can compute the generalization error on the test set.

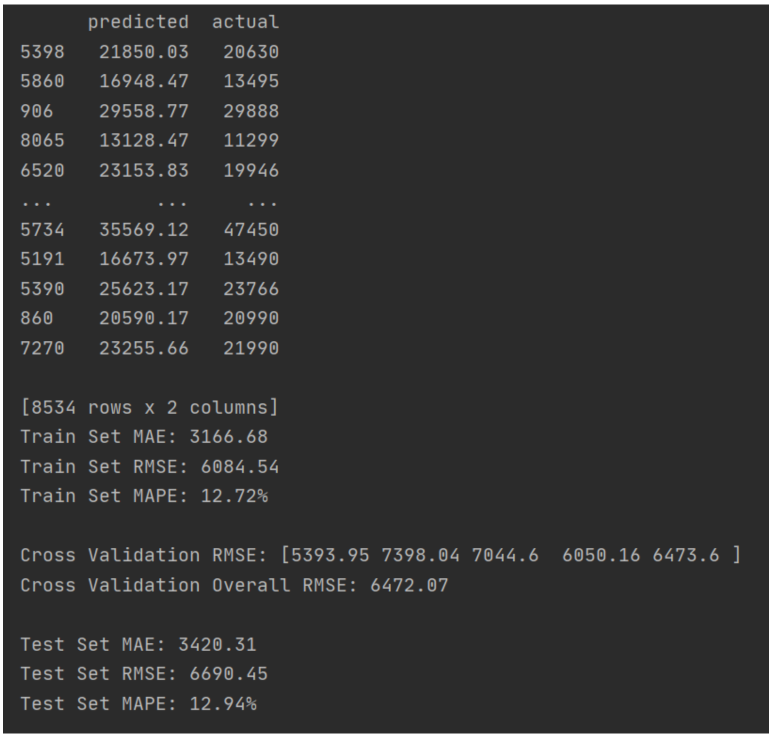
Generally, it is anticipated that the test set error will be larger than the training error. We will use test error as an indication of how well the model performs on unseen data.

Comparison of Results



```
       predicted  actual
5935     23900.0   23900
1858     19888.0   19888
4940     15990.0   15990
9982     24500.0   24500
299      63985.0   63985
...          ...     ...
8447     22600.0   22600
2934     21241.0   21241
10383    26000.0   26000
6618     12000.0   12000
8510     43950.0   43950

[8534 rows x 2 columns]
Train Set MAE: 51.81
Train Set RMSE: 359.16
Train Set MAPE: 0.17%

Cross Validation RMSE: [3312.55 3106.58 2903.98 3173.53 2964.28]
Cross Validation Overall RMSE: 3092.18

Test Set MAE: 1884.07
Test Set RMSE: 2892.48
Test Set MAPE: 9.03%
```

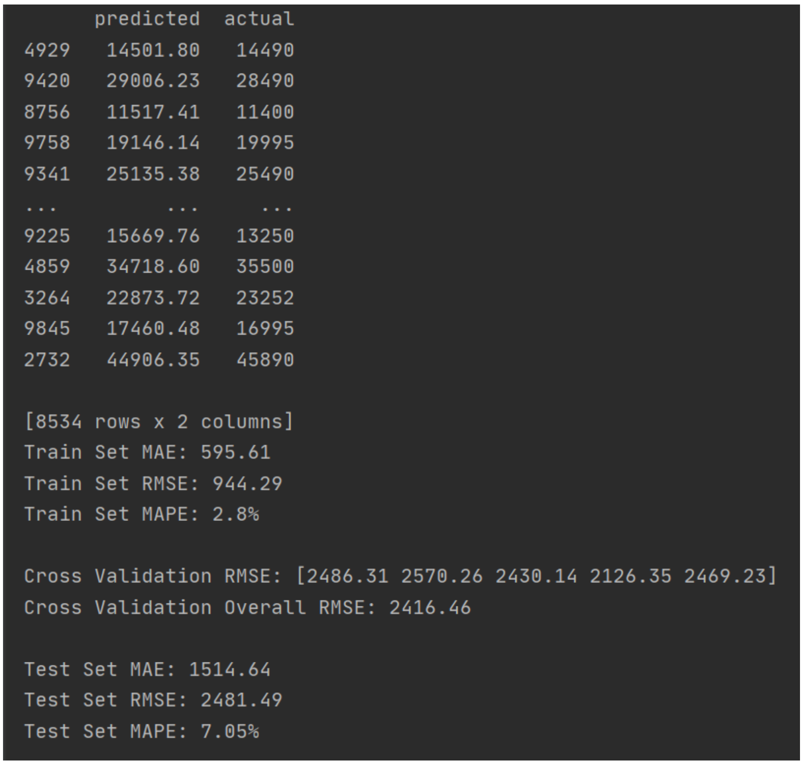
Decision Tree Regression



```
      predicted  actual
5398   21850.03   20630
5860   16948.47   13495
906    29558.77   29888
8065   13128.47   11299
6520   23153.83   19946
...         ...     ...
5734   35569.12   47450
5191   16673.97   13490
5390   25623.17   23766
860    20590.17   20990
7270   23255.66   21990

[8534 rows x 2 columns]
Train Set MAE: 3166.68
Train Set RMSE: 6084.54
Train Set MAPE: 12.72%

Cross Validation RMSE: [5393.95 7398.04 7044.6  6050.16 6473.6 ]
Cross Validation Overall RMSE: 6472.07

Test Set MAE: 3420.31
Test Set RMSE: 6690.45
Test Set MAPE: 12.94%
```

Support Vector Regression

Random Forest Regression



```
      predicted  actual
4929   14501.80   14490
9420   29006.23   28490
8756   11517.41   11400
9758   19146.14   19995
9341   25135.38   25490
...         ...     ...
9225   15669.76   13250
4859   34718.60   35500
3264   22873.72   23252
9845   17460.48   16995
2732   44906.35   45890

[8534 rows x 2 columns]
Train Set MAE: 595.61
Train Set RMSE: 944.29
Train Set MAPE: 2.8%

Cross Validation RMSE: [2486.31 2570.26 2430.14 2126.35 2469.23]
Cross Validation Overall RMSE: 2416.46

Test Set MAE: 1514.64
Test Set RMSE: 2481.49
Test Set MAPE: 7.05%
```

We can see that random forest regression has the lowest RMSE, MAE and MAPE in the test set. It also has the lowest overall RMSE in cross validation. The lowest RMSE indicates that large errors are fewer in random forest regression. In contrast, support vector regression has very high RMSE which means large errors are more prevalent.

Decision tree regression has stunningly low RMSE, MAE and MAPE (0.17%) in the training data. In comparison, RMSE in cross validation rises drastically to 9 times of that in training set. It is very likely that the model has overfitted the data. In the test set, the error becomes much larger and closer to results obtained from other two models.

As anticipated, all error metrics in test set are larger than those of the training set in all three algorithms. Since the model is not fitted to the test data, then it may not perform as good as it was in the training stage.

To sum up, random forest regression has the overall best performance. What’s better still, its cross-validation error is stable, which makes it a suitable model on this dataset and our forecasting problem. Support vector regression has very high RMSE as it is penalized more by making larger errors. And decision tree regression potentially suffers from overfitting, thus it needs more constraints on the model to make it useful on our problem.