

Manchester
Metropolitan
University

Advanced Machine Learning

6G7V0017

VEHICLE PRICE PREDICTION
AUTOTRADER -DATA

Mitali Patle
Date- 12 May 2023

1. Data Processing for Machine Learning

The dataset used in this assignment is from Auto-trader, which contains more than 4millions entries of vehicle details (including features and price). In this task we will be predicting the price of the vehicle considering the features using various regressor models.

Data processing is the most important step before proceeding with the building of any Machine Learning model. In this section, we have identified that the shape of the data is 402005 rows with 12 columns. Provided the dataset had 12 columns out of which 4 columns have numeric values and other 8 have datatypes as Object.

In this part, we have visually analyzed the data and all the features, considering vehicle condition- 92.23% of vehicles are in used condition and only 7.77% of vehicles are New. On the other hand, most of the vehicles have year of registration near to 2000's but after visualizing the data we can observe that there are few invalid entries with year of registration near to 1000's.

We have also, worked on the null values and the missing values in the data. Using `isna()` function we have identified that there are null entries in mileage, standard color, year of registration, body type and fuel type. As we cannot ignore these entries, so we have filled these null values with the mean of the value for all the numeric features, for e.g.,

```
advert['year_of_registration']=advert['year_of_registration'].fillna(advert['year_of_registration'].mean())  
advert['year_of_registration']=advert['year_of_registration'].astype(int)
```

For categorical data, the null values are replaced with the one occurring maximum time.

The range of the numeric data is widely speeded e.g., mileage varies from 0 to 999999, price varies from 120 to 9999999 and year of registration from 999 to 2020, it is clearly visible that these features have outlier as there are many values away from the standard deviation. IQR method is being used to remove these outliers, where we find the standard deviation of the feature and remove the values which are away from upper and lower quartile.

```
def quantile(x):  
    global Q1  
    global Q3  
    Q1=x.quantile(0.25)  
    Q3=x.quantile(0.75)  
    print('Q1=',Q1,'\nQ3=',Q3)  
    global IQR SS  
    IQR = Q3 -Q1  
    print ('IQR=', IQR)  
    global Lowerlimit  
    Lowerlimit= Q1 - 1.5*IQR  
    global Upperlimit  
    Upperlimit= Q3 + 1.5*IQR  
    print ("Lower limit=", Lowerlimit, "\nUpper limit=", Upperlimit)
```

For categorical data, the features which have very less entries are classified collectively as other to reduce the widespread range.

After removing all the outliers, noise and null values from the data, now the size of the dataset is reduced to (357310, 10), i.e. 357310 rows and 10 columns.

Later, Ordinal encoding technique is used for converting categorical data to numeric data for further processing.

```
oe = OrdinalEncoder() #Ordinal Encoding
```

In the next step, the data is rescaled, as all the feature's values ranges are different, for precise prediction of the model, all the data (both features and target) requires scaling. The learning process can be unstable when the target variable has a wide spread of values, resulting in large error gradient values. In this case, the standard scaling method is used to normalize the data, where each feature is transformed so that it has mean of 0 and standard deviation of 1.

The last step of data processing is to split the data into predictors and target and then into train and test. 25% of complete data is used for testing and 75% data is used for training. All the other processing of data will be done on train data only.

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25)
```

X_train shape- (267982, 9) and y_train shape – (267982,)

2. Feature Engineering

As, the dataset contains the following columns of the vehicle-

'standard colour', 'standard make', 'standard model', 'body type', 'fuel type', 'vehicle condition', 'crossover car and van', 'mileage', 'year of registration', 'reg_code', 'Reference_number', 'price'

As we have to detect the price, we will obtain the features from all the other columns. We will drop reference number and registration code columns as this does not contribute to the price. Now, we have 9 columns as a feature for prediction.

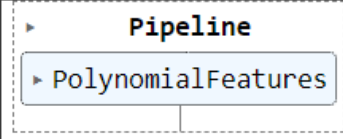
Also, combining two features together can help us predict the price of the vehicle more precisely. It is useful in capturing the nonlinear relationship between the feature and the target, for example, considering the mileage and year of registration can help us predict the price of the vehicle. The older the vehicle, the less the price.

Pipeline is created to process the training data and fit into the model all at once, pipeline helps to reduce the number of steps.

To combine the features, polynomial feature technique is used to generate a new feature from a existing one, here degree 2 polynomial feature technique is used to generate all the possible interactions between the features.

```
[107] preprocessing_pipe = Pipeline([
      ('poly', PolynomialFeatures(interaction_only=True, include_bias=False))
    ]).set_output(transform="pandas")

[82] preprocessing_pipe
```



```
graph TD
    Pipeline[Pipeline] --> PolynomialFeatures[PolynomialFeatures]
```

By fitting and transforming the pipe, we can check the number of features got created-

```
X_pp = preprocessing_pipe.fit_transform(X)
X_pp.head()
```

The shape of X_pp is now- (357310, 45), which means that there were 9 features and after applying polynomial feature technique there are now 45 features available to predict the target.

	standard_colour	standard_make	standard_model	body_type	fuel_type	vehicle_condition	crossover_car_and_van	mileage	year_of_registration	standard_colour standard_make	...	fuel_type vehicle_condition
0	-1.027410	-0.483252	1.706802	1.503360	-1.216202	-0.261965	-0.058893	2.430460	-1.403218	0.496497	...	0.31860
1	-0.261762	0.698867	1.746754	1.195387	0.766041	-0.261965	-0.058893	-0.943023	0.477229	-0.182937	...	-0.20067
2	-0.772194	1.171714	0.513948	-0.652448	-1.216202	-0.261965	-0.058893	0.306540	0.163821	-0.904791	...	0.31860
3	-0.261762	-0.305934	0.919176	1.195387	-1.216202	-0.261965	-0.058893	0.944757	-0.149587	0.080082	...	0.31860
4	-1.027410	-1.606264	1.039032	-1.576366	0.766041	-0.261965	-0.058893	-0.667582	0.477229	1.650292	...	-0.20067

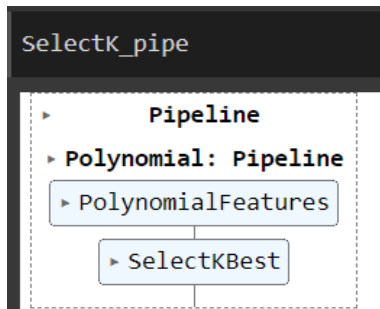
5 rows x 45 columns

3. Feature Selection and Dimensionality Reduction

There are now 45 features available for input and not all the features are important or will help in predicting the price of a vehicle. To reduce these features there are several techniques, we have used the following feature selection methods-

- a) SelectKBest
- b) PCA (Principal Component Analysis)
- c) RFE (Recursive feature elimination)

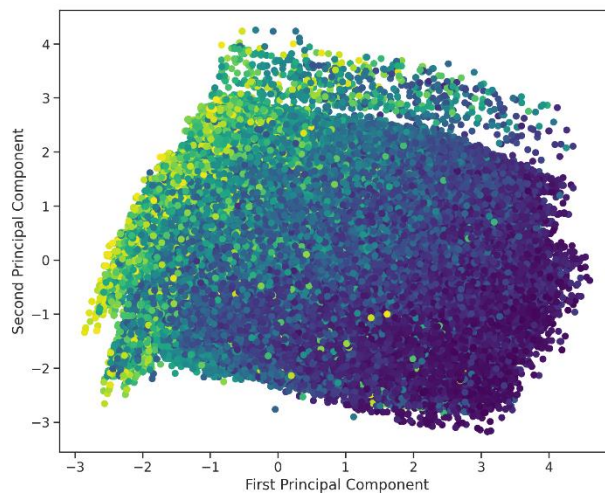
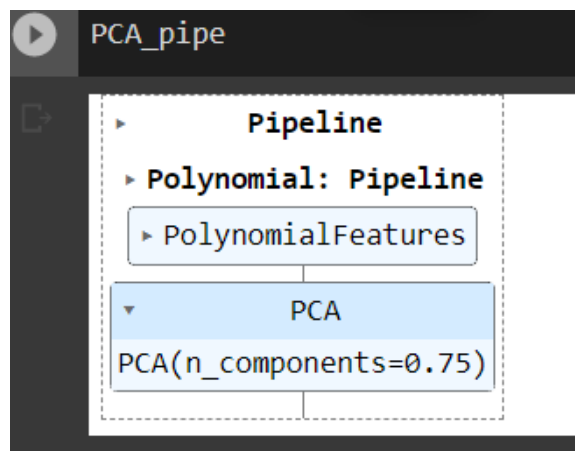
SelectKBest- This method selects the top k features by evaluating each feature independently and measuring the strength of relationship between feature and target. K is user defined value, here we have used k=10, which means, this function will give us 10 best features out of 45.



The best 10 features which are selected- *“standard_make, body_type, vehicle_condition, mileage, year_of_registration, standard_make vehicle_condition, fuel_type vehicle_condition, vehicle_condition mileage, vehicle_condition year_of_registration, mileage year_of_registration”*

PCA (Principal Component Analysis)- This technique is a dimensionality reduction technique which reduces the number of features and retains the most important information from the data. In PCA () we can either pass the number of features we wanted to retain or the percentage of original data we want to retain-

In this case we tried to retain 75% of the original data. The shape of X after fit and transform changed to `-(267982, 21)`, which shows that the number of columns is reduced to 21 which were earlier 45.



PCA Scatter Plot

RFE (Recursive feature elimination)- Recursive feature elimination technique recursively selects the best feature using the feature importance score, after every iteration the feature with least score is reduced and model is again trained with the remaining features. These steps are repeated until the best score is achieved.

We can mention the number of features we want to select by using parameter- `n_features_to_select=10`.

With RFE -linear model scores: train score = 0.515, test score = 0.515 with RMSE = 0.694.

4. Model Building

There are several types of models that can be used to make predictions of the target with input features. In this assignment we have used 3 different models for our prediction.

1. Linear regression
2. Random Forest Regressor
3. Gradient Boosted Tree

4.1 LINEAR REGRESSION

Linear regressor is the most basic regressor model, that makes prediction based on the input features. In this assignment we have considered the linear model as the base model and tried all the three types of feature selection methods (mentioned above), the following are the result and the scores of Linear Model

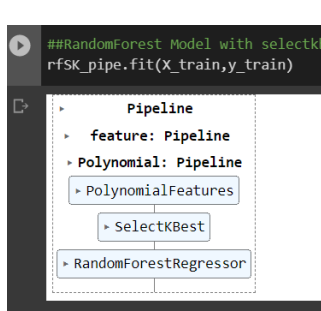
	Linear Model - Feature Selection	Test Accuracy Score	Train Accuracy score	RMSE- test	RMSE- train	MAE
1.	SelectKBest	0.490	0.491	0.711	0.714	0.535
2.	PCA	0.415	0.411	0.761	0.768	0.589
3.	RFE	0.515	0.515	0.6933	0.696	0.516

From the above table it is observable that linear model works best with the RFE method. The difference in training error and testing error is very less which shows us that there is underfitting or overfitting of the model. However, the train and test accuracy score are ~50% which is very less.

4.2 RANDOM FOREST REGRESSOR

Random forest regressor uses predictions from different decision tree models created with different subsets of original data. All the predictions are then combined to give a final prediction, random forest regressor models are best for large data like we are using and are less prone to overfitting of model.

In our case we have trained our model with features selected from SelectKBest method, fit out data in the regressor model with parameter `max_depth=10`. The results are as follows-



RANDOM FOREST REGRESSOR: MAX_DEPTH =10	
TEST ACCURACY SCORE	0.798
TRAIN ACCURACY SCORE	0.804
RSME	0.448
MSE	0.20
MAE	0.30

There are certain parameters which can help to make our model predictions more accurate, we can find these parameters using Grid Search CV

GRID PARAM SERACH- We will try to evaluate the best parameters for our model

```
grid_param = { 'n_estimators': [50,100], 'max_depth': [2,8, 5, 10, 20, None], 'min_samples_split': [2, 5, 10], 'min_samples_leaf': [1, 2, 4] }
```

Grid search CV gave us the best parameters as follows- “`{'max_depth': 20, 'min_samples_leaf': 1, 'min_samples_split': 5, 'n_estimators': 100}`”

After getting the best parameters we will now use these parameters in our model to fit the train data and will check our accuracy and error score on test data-

RANDOM FOREST REGRESSOR: BEST PARAMETERS	
TEST ACCURACY SCORE	0.84
TRAIN ACCURACY SCORE	0.92
RSME	0.44
MSE	0.15
MAE	0.26

It's clearly visible that with best parameters from grid search CV, we got better accuracy and lesser error values.

4.3 GRADIENT BOOSTED TREE

Gradient Boosted tree is similar to Random Forest Regressor, however, in case of GBT, there are sequence of Decision Tree in which first decision tree is trained on the complete data and rest trees are trained on the errors of previous tree. We can use different parameters for best prediction of tree, initially we have used- `_estimators=100`, `learning_rate=0.1`, `max_depth=3`. The results are as follows-

GBT: ESTIMATOR=100, LEARNING RATE=0.1, MAX DEPTH=3	
TEST ACCURACY SCORE	0.76
TRAIN ACCURACY SCORE	0.76
RSME	0.48
MSE	0.33
MAE	0.2

Here the train score and the test score are the same with very less RMSE, MSE and MAE.

Next, we will use Grid search CV to find the best parameter for the gradient boosted tree.

GRID SEARCH CV- `param_grid = { 'learning_rate': [0.01, 0.1, 1], 'n_estimators': [20, 50, 100], 'max_depth': [3, 5, 7], 'min_samples_split': [2, 5, 10], 'min_samples_leaf': [1, 2, 4] }`

Grid search cv gave us the best param as follows- `{'max_depth': 10, 'min_samples_leaf': 1, 'min_samples_split': 10, 'n_estimators': 100}`

As, we have best parameters now, we fit our model with these parameters to predict the target and will evaluate the scores and the error values produced-

GRADIENT BOOSTED TREE: BEST PARAMETERS	
TEST ACCURACY SCORE	0.85
TRAIN ACCURACY SCORE	0.85
RSME	0.38
MSE	0.14
MAE	0.26

We could observe that the test and train score got increased and errors got decreased with these parameters.

We could observe from the three models that Random Forest regressor gave us the best accuracy with least error values. Also, the test data errors are very which shows that the model is not under/over fitted.

4.4 VOTING ENSEMBLE

In voting ensemble, we will combine all our three models to make predictions on a smaller data set separately and then predictions of these three models will be combined to give a final prediction.

Machine learning uses voting classifiers to perform classification tasks. Voting Classifiers includes multiple models of different machine-learning algorithms. All models are fed the whole data set, and every algorithm predicts once trained on it. The final prediction is derived from the most frequent strategy once all the models have predicted the sample data. In this case, the final prediction of the model is the category most predicted by the multiple algorithms.

We will first create a list of all three models –

`ensembled = [lmSK_pipe, rfSK_pipe, gbrSK_pipe]` and then will fit the train data in all three models at once.

After this we will use Voting Regressor model to predict the target and find the which model gives best negative root mean square error

Model	Negative RMSE- mean	Negative RMSE- std deviation
Linear Model	0.7137944954855686	0.0025702204403095967
Random Forest Regressor	0.4517138686544963	0.0025519477248256415
Gradient Boosted Tree	0.4853960136480436	0.003342222069978418
Voting Regressor	0.5030386130148161	0.0025198758024962014



The above plot compares the prediction of all 4 models, Linear Model, Random Forest Regressor, Gradient Boosted tree and Voting Regressor with the true data. And it is observable that the true data coincides with the Ensemble model and Random Forest model.

Also, the mean and standard deviation for negative root mean squared error is best for Random Forest regressor.

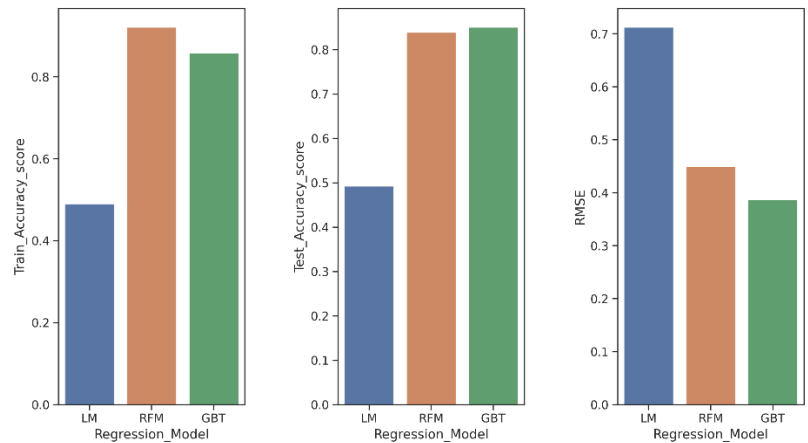
5. MODEL EVALUATION

Now, we have fitted and transformed our data into three different regressor models and made the predictions. As we observed in the previous part that Random Forest regressor performed best with our data. In this section we will perform fine grained evaluation of these models to prove which model performed best.

- **Overall performance with cross validation**

This bar plot shows that the Gradient boosted Tree model had the best test score and the least RMSE value. We decide the prediction is best on the test data as the test data is unseen data for our model and if the model is giving better accuracy on test data, then our model's performance is better.

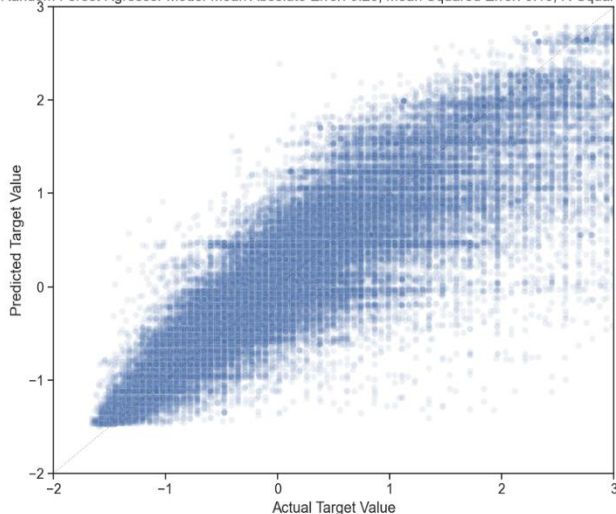
Model Cross Validation Scores(N-RMSE)		
Models	Mean	Std
Linear Model	0.7091	0.014
Random Forest	0.4481	0.010
Gradient Boosted tree	0.407	0.009



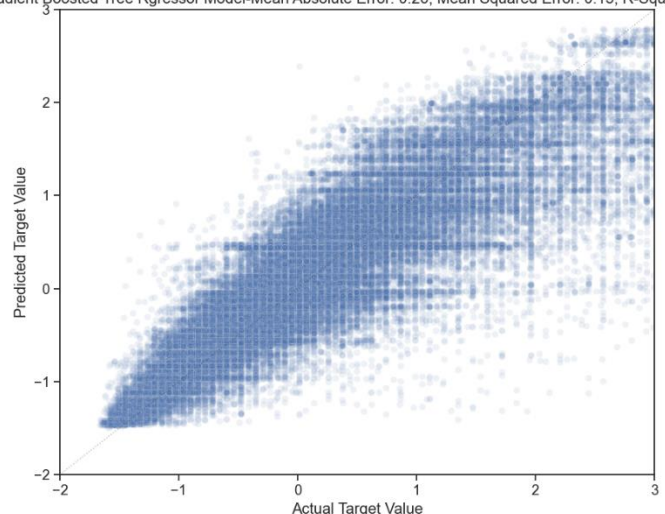
- **Actual VS predict Plot.**

The actual vs predict plot is a scatter plot which shows the distribution of the predicted values. The line passing through the axis is the true value (y_{test}) and the points show the distribution of the predicted values. When the distribution of the plots are closer to the true line, this shows us that our true value and the predicted value are very close to each other and the erroneous gradient is also less.

Random Forest Rgessor Model-Mean Absolute Error: 0.26, Mean Squared Error: 0.16, R-Squared: 0.84



Gradient Boosted Tree Rgessor Model-Mean Absolute Error: 0.26, Mean Squared Error: 0.15, R-Squared: 0.85

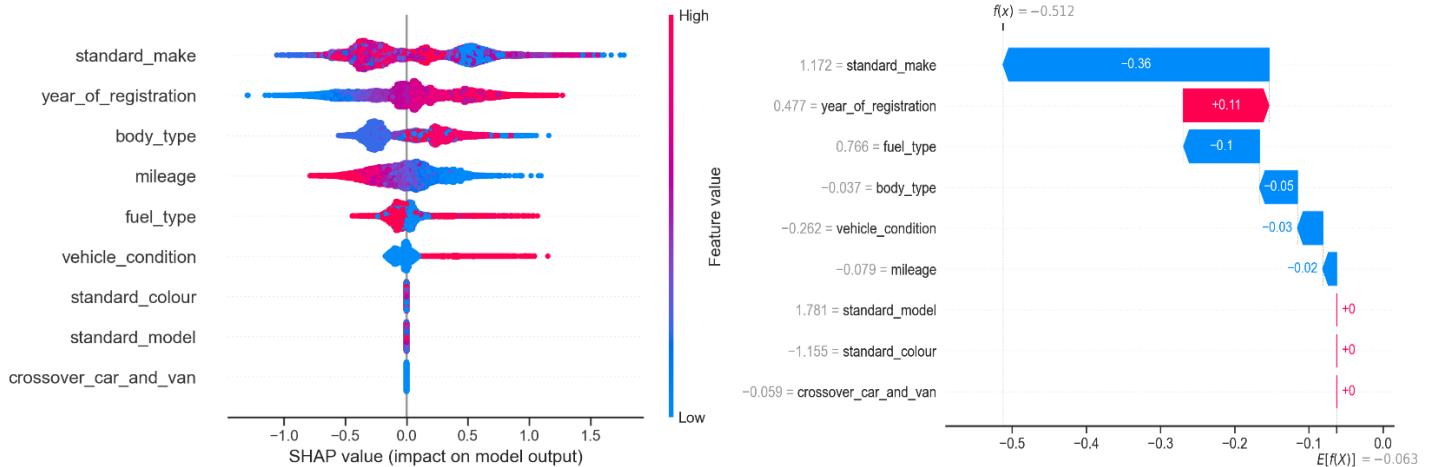


As our predictions are better with both Random Forest Model and Gradient Boosted Tree, the scatter plots are very near to the true data line.

- **Global and Local Explanation with SHAP**

SHAP-Shapley Additive Explanation is a technique which explains the feature importance in the model prediction. We will use waterfall plot to visualize the impact of each feature on the model prediction.

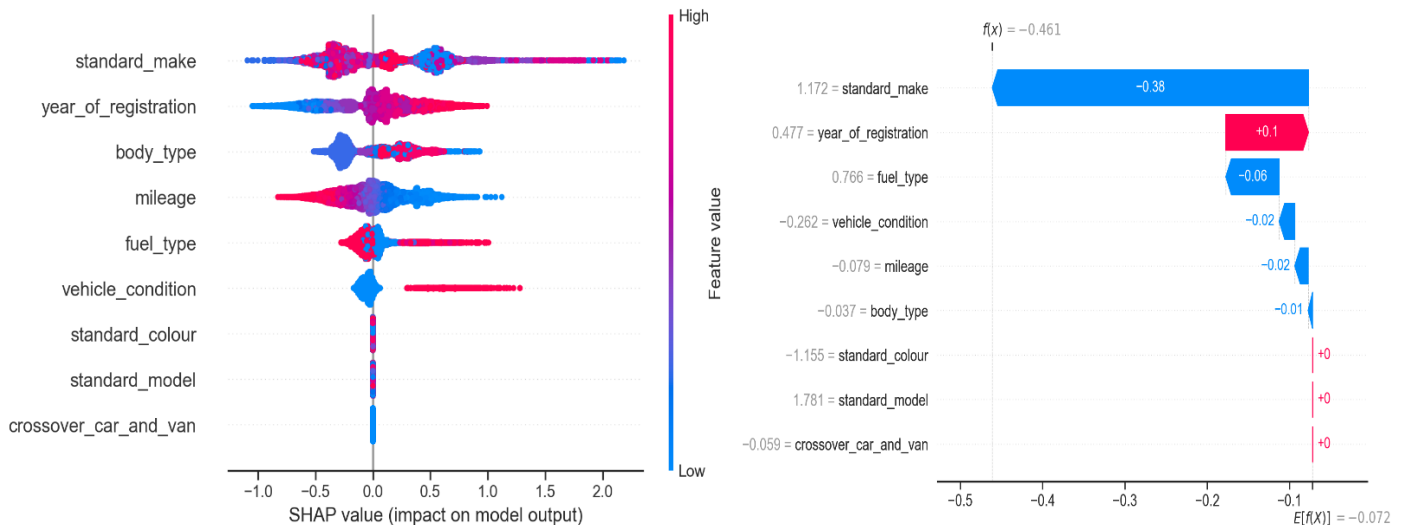
a) SHAP Plot for Gradient Boosted Tree



The features sequence in the waterfall plot are lined up according to the contribution of the feature towards the prediction. The negative value indicates the feature is pushing lower prediction and the positive value indicates opposite.

Here, for Gradient Boosted Tree, year of registration is pushing the prediction higher, while standard make is pushing lower prediction, this can help us for feature selection.

b) SHAP Plot for Random Forest regressor



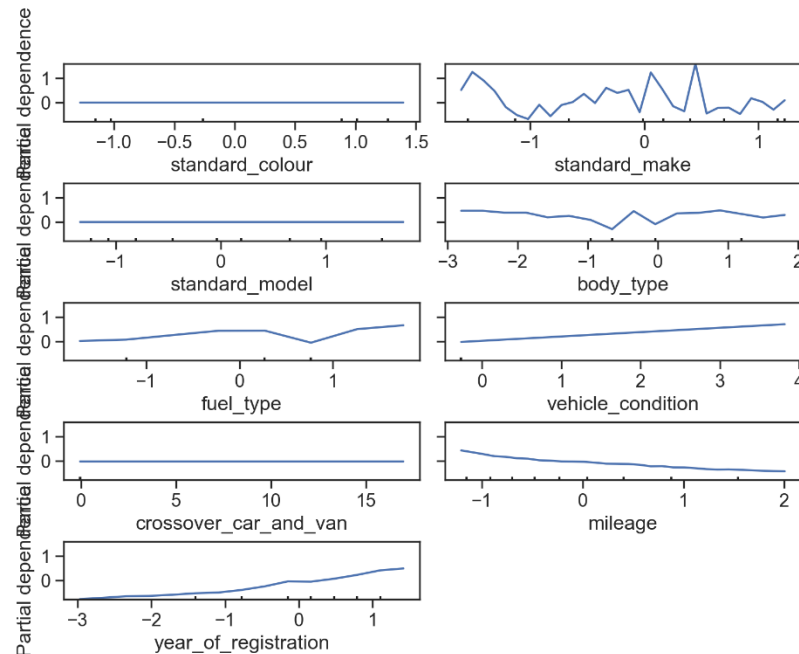
Similarly, for Random Forest Regressor, the contribution of features are similar to that of GBR. Standard colour, standard model and cross over car and van have the least contribution towards the prediction of price of Vehicle.

- **Partial Dependency Plot**

Partial Dependency Plot shows the marginal effect of a feature on the predicted data of the model while keeping the other features constant.

PDP shows the relationship between the feature and the predicted output of the model, e.g., if the relation ship is linear or nonlinear or no relation.

X- axis shows the feature while the Y-axis shows the predicted output of the model.

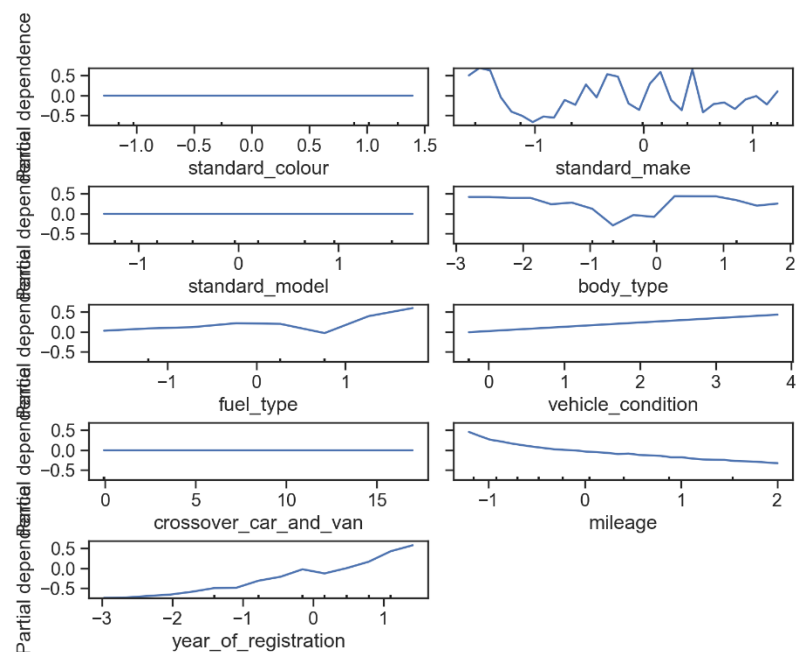


Gradient Boosted Tree -PDP

These Plot shows that the vehicle condition, mileage, year of registration and fuel type have linear relationship with the predicted outcome.

However, on the other hand standard make and body type have a non-linear relationship.

Standard Colour, standard model and crossover car and van have no relation with the predicted output.



Random Forest Regressor- PDP

Similarly, for RFR as well, these Plot shows that the vehicle condition, mileage, year of registration and fuel type have linear relationship with the predicted outcome.

Standard make and body type have a non-linear relationship.

Standard Colour, standard model and crossover car and van have no relation to the predicted output.