Sodeinde Kehinde 23664826

This report aims to outline the steps for creating machine learning models that predict car prices using specific features.

1. Data Preprocessing for Machine Learning

General information about features, non-missing values count, data types, and the shape of the dataset.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 402005 entries, 0 to 402004
Data columns (total 12 columns):
 #   Column                Non-Null Count   Dtype
---  ------                --------------   -----
 0   public_reference      402005 non-null  int64
 1   mileage               401878 non-null  float64
 2   reg_code              370148 non-null  object
 3   standard_colour       396627 non-null  object
 4   standard_make         402005 non-null  object
 5   standard_model        402005 non-null  object
 6   vehicle_condition     402005 non-null  object
 7   year_of_registration  368694 non-null  float64
 8   price                 402005 non-null  int64
 9   body_type             401168 non-null  object
 10  crossover_car_and_van 402005 non-null  bool
 11  fuel_type             401404 non-null  object
dtypes: bool(1), float64(2), int64(2), object(7)
memory usage: 34.1+ MB
The data contains 402005rows and 12 columns
```

Dealing with Missing Values: Each machine learning model's pipeline was incorporated with the Simple Imputer algorithm to fit and transform the training and testing datasets, addressing missing values. Numeric features were imputed with their respective mean values, while categorical features were filled with the most frequently occurring value.

Dealing with Outliers: The price column has an average value of £17,341, minimum value of just £120 and a maximum value of £10,000,000. The frequency distribution is significantly skewed to the right towards the extreme values (outliers).

We decided to limit price values within a certain range defined by the 1st and 99th percentiles, solely to reduce the level of skewness and retain as much of the data rows as possible.

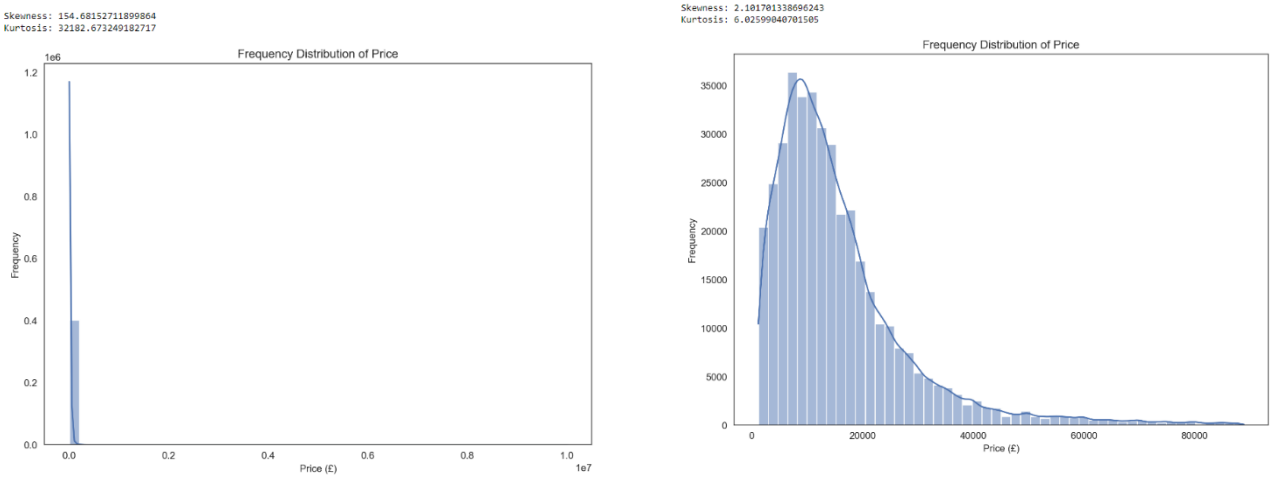
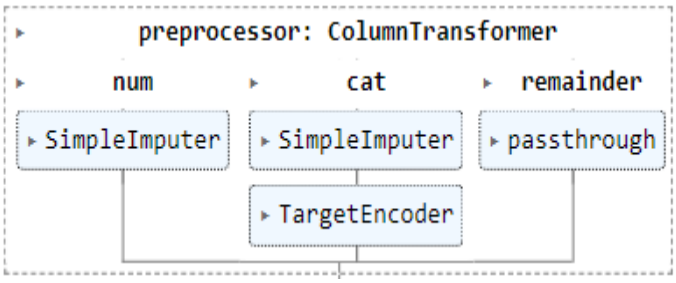
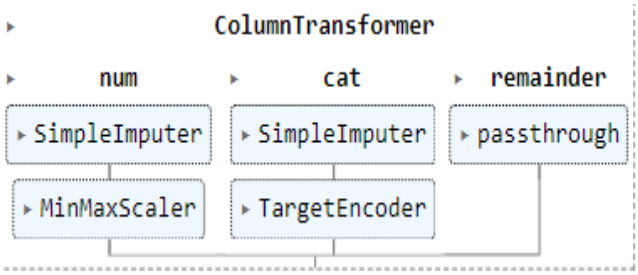


Figure1a. frquency distribution before capping price limit & Figure1b. Frequency distribution after capping limit

Data Transformation: In our machine learning pipeline, we implemented preprocessing steps for both numeric and categorical features. Specifically, we included a **MinMax Scaler** for numeric features, but only for the linear regression algorithm, as other predictive algorithms are not sensitive to feature scaling.

For categorical features, we employed a target encoder to convert them into numerical representations based on the target variable (price). This transformation ensures that categorical variables are compatible with machine learning models, which only understand numerical data.





The dataset was divided into predictors (independent features like **mileage, vehicle condition, make, and model**) and the target variable (**price**). This was followed by splitting the dataset into training and testing sets to assess the performance of the models.

```
##split te data set into independent features and target features
X = df_pp.drop(['price'],axis=1)
y= df_pp['price']

###split the dataset into train and test set
X_train, X_test, y_train, y_test = train_test_split(X, y,test_size=0.2,random_state=36)
print(X_train.shape, X_test.shape,y_train.shape,y_test.shape)
```

2. Feature Engineering

A thorough investigation revealed that the first 4 numbers of each unique value in the **public_reference** column signifies the particular year the car was placed on advert sales. the first 4 numbers were extracted to create a new feature called **advert_year**.

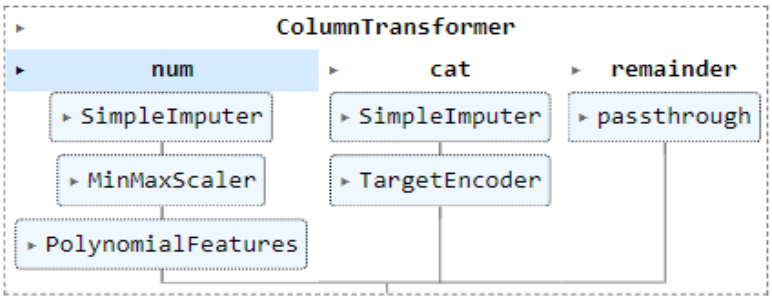
```
## assuming the first 4 public reference signifies the year the vehicle was up for advertisment sale, extract the first 4 num
df['advert_year'] = df['public_reference'].apply(lambda x: str(x)[0:4]).astype('int64')
```

We created another column called **car_age** which entails value of the difference between the year a car was ever first registered (**year_of_registration**) and the year it was placed on advert (**advert_year**), mainly to determine the age of the car as at when it was advertised.

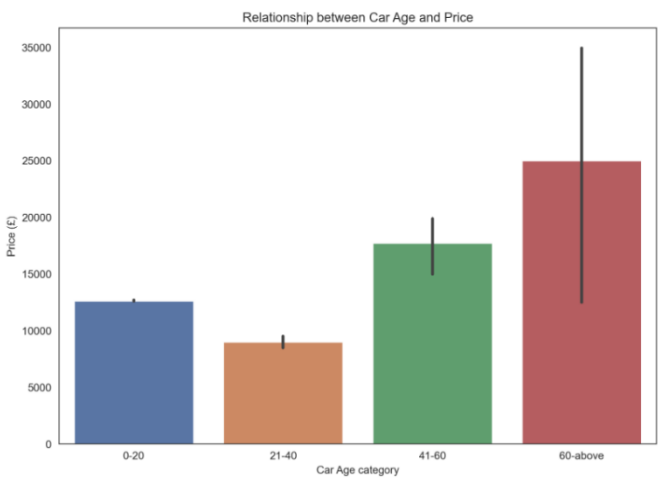
```
#age of the car as at when advertised is the advert year minus the year of registration
#convert year_of_registration to float and then to int
df['year_of_registration'] =df['year_of_registration'].astype('float64')
df['year_of_registration'] =df['year_of_registration'].astype('int64')
df['car_age'] = df['advert_year'] - df['year_of_registration']
#according to the table, some cars were registered after they've been advertised, get to absolute value of car_age column
df['car_age'] =abs(df['car_age'])
```

Polynomial Feature: The addition of polynomial features, specifically of degree 2, to the linear regression pipeline aims to enhance the model's flexibility, allowing it to better capture non-linear patterns, as real-world prediction problems are almost never exhibiting non-linear patterns.

Some of the new features created by Polynomial features includes **mileage^2, mileage crossover_car_and_van, mileage car_age, crossover_car_and_van^2, crossover_car_and_van car_age, car_age^2.**



3. Feature Selection and Dimensionality Reduction



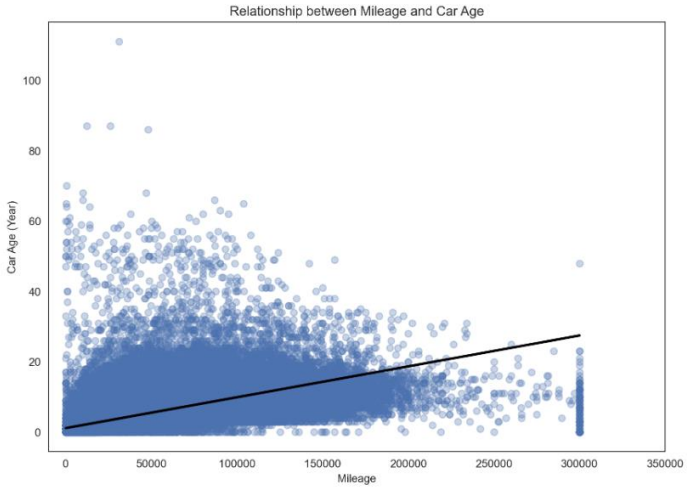


Figure 2a & 2b: Relationship between Car Age and price, Mileage and Price respectively.

While majority (99.5%) of cars falls between the age 0-20 years, Figure 2a, evidently shows that the median price of cars generally decreases within the age bracket of 21-40 years. However, beyond this age range, there is a noticeable upward trend in the median prices of cars. This shift may be attributed to the fact that certain cars older than 40 years are often considered vintage or classics, resulting in higher prices for such categories of cars.

Fig 2b showed an already presumed notion that the age of a car is directly proportional to its mileage, these are good predictors of price based on general domain knowledge.

Despite the wide disparity in distribution between new and used cars, 92.2% and 7.8 % respectively, the price distribution based on vehicle conditions as shown in Figure 3 clearly shows a wide range in price amongst new and used cars. This is definetly a strong predictor of price.

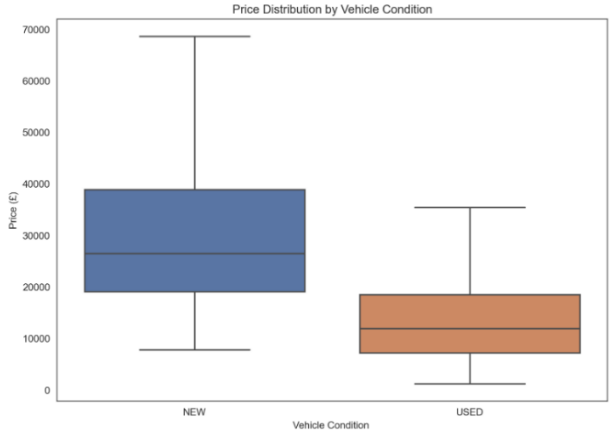


Figure 3: Boxplot distributuion of price categorised by vehicle condition

In consideration of the Pearson correlation coefficient analysis, conducted on both the relationships between features and target and also the correlation among the predictive features themselves, as presented in images above, and complemented by general domain knowledge, It is asserted that the following features play a crucial role in determining the price of a car: **mileage, standard_make, vehicle_condition, body, type, fuel_type, year of registration, car_age, crossover_car_and_van,** and **standard_model.**

These selected features were used to evaluate the performance of linear regression model with and without polynomial features, and **R-squared (R2)** test score was 0.797 and 0.783 respectively. indicating that 79.7% of the variability in price can be explained by the independent features in the test set for linear model with polynomial features, and 78.3% in the test set for linear model without polynomial features.

Despite the polynomial feature algorithm creating 14 features compared to the originally selected 8 features, the marginal difference in score does not justify the increased complexity introduced by the polynomial algorithm. As a result, the polynomial algorithm was excluded from the feature selection process.

```
regr_poly.fit(X_train, y_train)

train_pred = regr_poly.predict(X_train)
test_pred = regr_poly.predict(X_test)

print("LINEAR REGRESION WITH POLY")
print("TRAIN EVALUATION")
evaluation(y_train, train_pred)
print("TEST EVALUATION")
evaluation(y_test, test_pred)
```

```
LINEAR REGRESION WITH POLY
TRAIN EVALUATION
R2 score : 0.7978 | MAE score : 3679.1816
TEST EVALUATION
R2 score : 0.7972 | MAE score : 3684.9978
```

```
regr_1.fit(X_train, y_train)

train_pred = regr_1.predict(X_train)
test_pred = regr_1.predict(X_test)

print("LINEAR REGRESION WITHOUT POLY")
print("TRAIN EVALUATION")
evaluation(y_train, train_pred)
print("TEST EVALUATION")
evaluation(y_test, test_pred)
```

```
LINEAR REGRESION WITHOUT POLY
TRAIN EVALUATION
R2 score : 0.7822 | MAE score : 3785.0813
TEST EVALUATION
R2 score : 0.7830 | MAE score : 3778.2637
```

Recursive Feature Eliminator (RFE) was implemented for Linear Regression algorithm to select the best performing features by recursively removing features and building a model with the rest until target performance level is achieved. The initial manually selected 8 features were selected by the RFE algorithm. No feature selector was used with the random forest, gradient boosting and voting ensemble algorithm as they all perform automatic feature selection as part of model building.

4. Model Building

The following algorithms were used to build predictive models for this task, namely, Linear Regression, Random Forest Regressor, Gradient Boost Regressor and Voting Regressor.

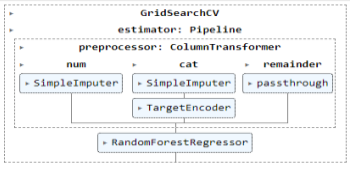
The algorithms were instantiated via preprocessor pipeline and tested on 20% of the entire dataset, using grid search cross validation to determine and select the best hyperparameters for optimized score.

	param_estimator__fit_intercept	mean_train_score	std_train_score	mean_test_score	std_test_score	rank_test_score
0	True	0.758272	0.001229	0.751449	0.006847	1
1	False	0.753266	0.001209	0.746090	0.006812	2

4.1 Linear Regression: The top estimator from the grid search has an average train score of 0.758 and an average test score of 0.751. The slight difference between these scores makes it great for making predictions. So therefore, the best pick for linear regression is the best estimator found in the linear regression grid search model.

Random Forest Regressor: The model had the best score when the hyperparameter; max_depth (maximum depth of each decision tree) and min_sample_leaf (minimum number of samples required to be at a leaf node)

```
rfr
kf = KFold(n_splits=6, shuffle= True, random_state=36)
param_grid = {'estimator__max_depth': [1, 2, 3, 4, 5, 6, 7],'estimator__min_samples_leaf': [5, 15, 20, 25, 30]}
rfr_cv = GridSearchCV(rfr, param_grid, cv=kf, return_train_score=True)
rfr_cv.fit(X_train,y_train)
```



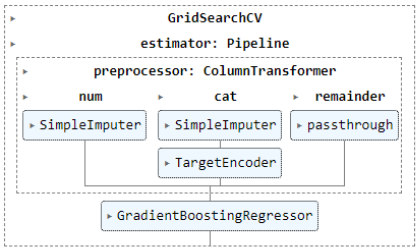

```
rfr_cv_df = pd.DataFrame(rfr_cv.cv_results_)
rfr_cv_df[['param_estimator__max_depth', 'param_estimator__min_samples_leaf', 'mean_train_score', 'std_train_score',
           'mean_test_score', 'std_test_score', 'rank_test_score']].sort_values('rank_test_score').head()
```

	param_estimator__max_depth	param_estimator__min_samples_leaf	mean_train_score	std_train_score	mean_test_score	std_test_score	rank_test_score
30	7	5	0.875442	0.002136	0.860160	0.005491	1
31	7	15	0.871848	0.002327	0.856930	0.005531	2
32	7	20	0.870125	0.002570	0.855805	0.005675	3
33	7	25	0.868450	0.002575	0.854360	0.005518	4
34	7	30	0.866627	0.002127	0.852790	0.005623	5

were 7 and 5 respectively. This random forest model had mean train R2 score of 0.875 and mean test R2 score of 0.860.

The mean standard deviation for the test score was 0.006, indicating how close an individual test score varies from the mean test score.

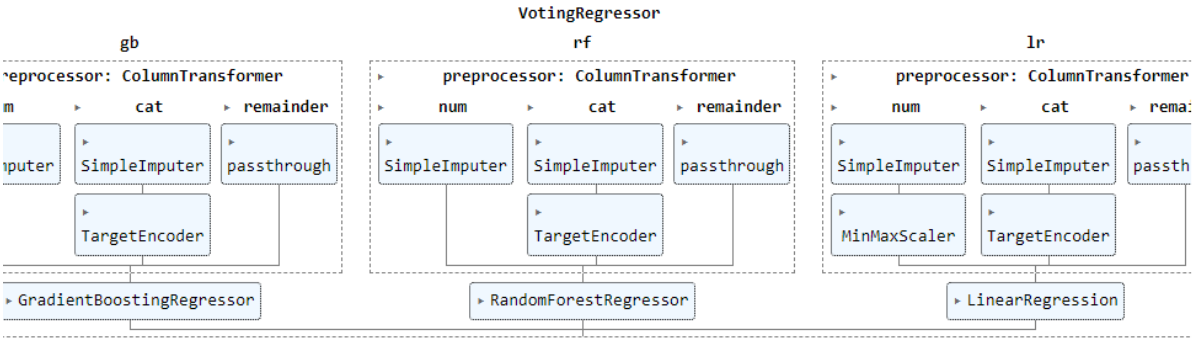
4.3 Gradient Boost Regressor: The model's best performing hyperparameters were max_depth =7 and min_samples_leaf =5, with a mean train R2 score of 0.949 and mean test R2 score of 0.927. This shows the model performed well on both the train test and test set, without any signs of overfitting.




```
gbr_cv_df = pd.DataFrame(gbr_cv.cv_results_)
gbr_cv_df[['param_estimator__max_depth', 'param_estimator__min_samples_leaf', 'mean_train_score', 'std_train_score',
           'mean_test_score', 'std_test_score', 'rank_test_score']].sort_values('rank_test_score').head()
```

	param_estimator__max_depth	param_estimator__min_samples_leaf	mean_train_score	std_train_score	mean_test_score	std_test_score	rank_test_score
30	7	5	0.949606	0.000792	0.927924	0.003893	1
31	7	15	0.943719	0.001335	0.925168	0.003388	2
32	7	20	0.942091	0.001118	0.924005	0.003351	3
33	7	25	0.940220	0.001368	0.922946	0.003689	4
34	7	30	0.938999	0.001127	0.922209	0.003779	5

4.4 Voting Regressor: Voting Regressor was instantiated by combining the 3 regressor models with the sole aim of aggregating their respectively target prediction and producing a model with even better predictive power. The train R2 score, 0.899 and test R2 score, 0.884 were lower than that of gradient boosting regressor, showing that depending on the task, voting regressors sometimes don't perform better than individual models.




```
Voting Regressor

Train
R2 score : 0.899 | MAE score : 2487.257
Test
R2 score : 0.884 | MAE score : 2537.954
```

5 Model Evaluation and Analysis

5.1 Overall Performance

The selected models from Grid Search cross validation were fitted, trained, and tested on the entire dataset. Table 1 below shows the evaluation of each model's performance in terms of R2 score, Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE).

While all the models performed relatively well, without any signs of overfitting, i.e. model performs well on the train set but poorly on the test set, The Gradient Boosting Regressor had the best R2 test score of 0.938, closely followed by voting regressor model with test score 0.906. Random Forest and Linear Regressor had a relatively lower test R2 score of 0.873 and 0.783 respectively.

	Model	Test R2 Score	Train R2 Score	Mean Absolute Error	Root Mean Squared Error
2	Gradient Boosting Regressor	0.938	0.944	1887.941	3146.879
3	Voting Regressor	0.907	0.924	2370.053	3876.933
1	Random Forest Regresor	0.873	0.877	2831.974	4516.887
0	Linear Regression	0.784	0.783	3767.036	5888.859

Table 1: Evaluation of models' performance

While the R2 score provides insight into model performance, it alone isn't sufficient for determining the best performing model. Therefore, Mean Absolute Error (MAE) was also used to assess the models' performance, offering a measure of the magnitude of differences between actual and predicted prices without indicating the direction of the difference.

A lower MAE indicates better model performance, and in this context, the Gradient Boosting Regressor model outperforms the others with a notably lower MAE of £1898. This trend is also observed in the Root Mean Squared Error (RMSE), where Gradient Boosting Regressor demonstrates superior performance by indicating how far predicted values deviate from the actual values, reinforcing its efficacy.

5.2 True Vs Predicted

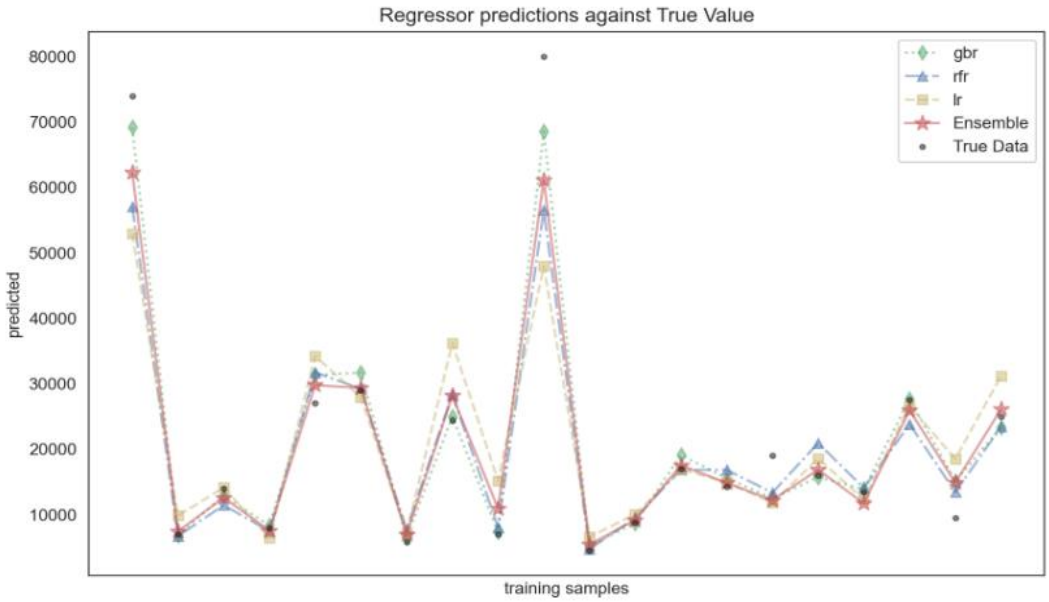


Figure 4: Predictions of individual models against True value

Linear Regression: Examining the instance-level errors reveals that the linear regression model successfully predicted car prices with absolute residual error ranging from as low as £0.05 to as high as £81,725. Approximately 11% of the predicted values have an absolute residual error of £500 or less.

Random Forest Regressor: The Random Forest Regressor model exhibits an absolute residual error as low as £0.11 and as high as £66,685. 15.5% of the predicted car prices have an absolute residual error of £500 or less.

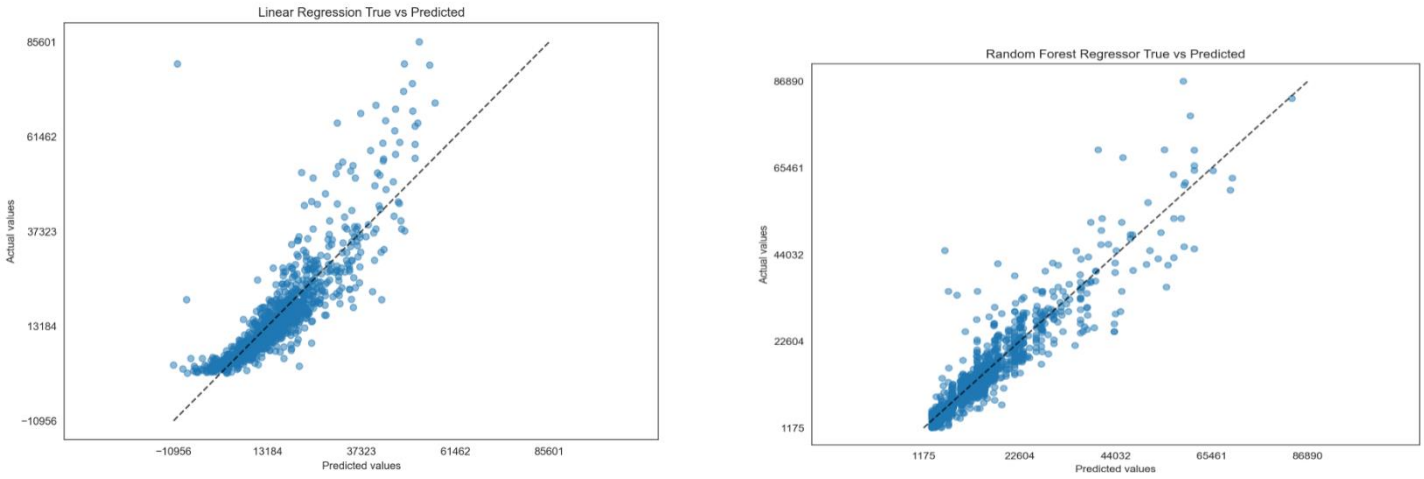


Figure 5a &5b: True vs Predicted regression plot; Linear Regression and Random Forest Regressor respectively.

Gradient Boost Regressor: Among the four selected models, Gradient Boost Regressor demonstrated superior performance. Upon examining the instance-level errors, the model successfully predicted values with a flawless absolute residual error of 0 and a maximum residual error of around £66,024. Notably, about 24.61% of the predicted values display an absolute residual error of £500 or less.

Voting Regressor: The Voting Regressor model also had flawless absolute residual error of 0 and a maximum residual error of high as £66735. 18.3% of the predicted car prices have an absolute residual error of £500 or less.

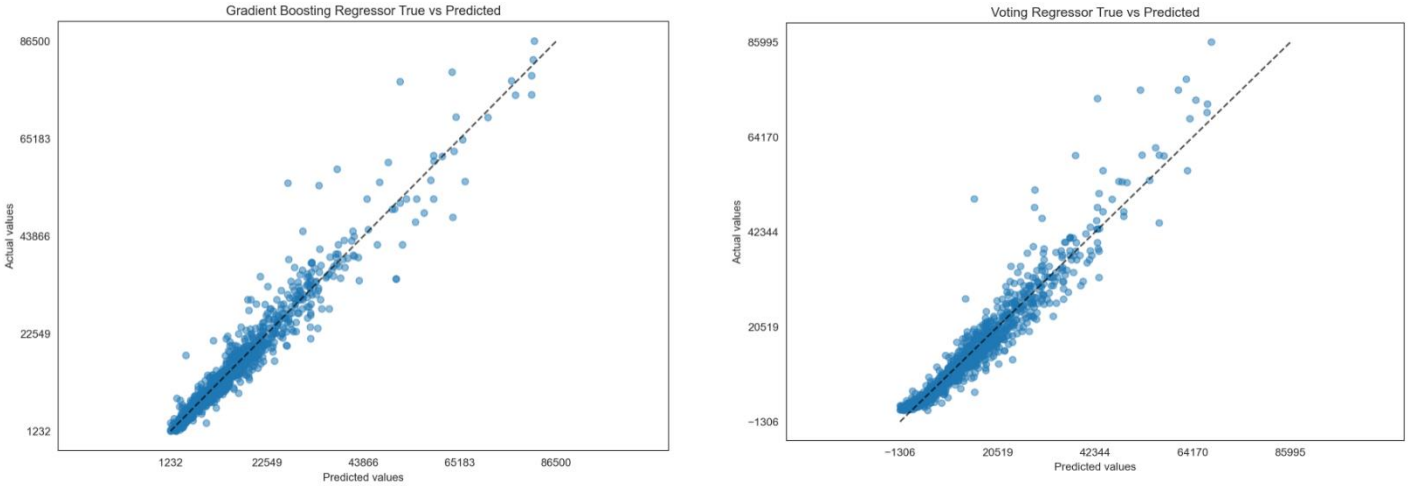


Figure 5c &5d: True vs Predicted regression plot; Gradient Boosting Regressor and Voting Regressor respectively.

Model Based Feature Importance

Linear Regression: In the linear regression model, only three features demonstrated notable impact on the price, as indicated by their regression coefficients—specifically**, car_age and mileage and crossover**. Mileage and car age both have a strong, significant negative correlation with the target, price.

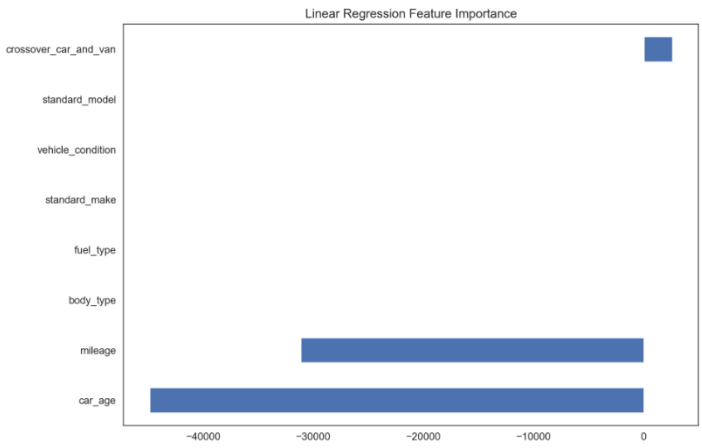


Figure 6a: Linear Regression Feature Importance

Random Forest Regressor and Gradient Regressor: Both models have a similar chart as seen in figure 6a and 6b below. **Standard_model, car_age, mileage, standard_make and vehicle_condition** all have significant influence on the target, price. The most important predictor variable is the standard model with a contribution of 60%.

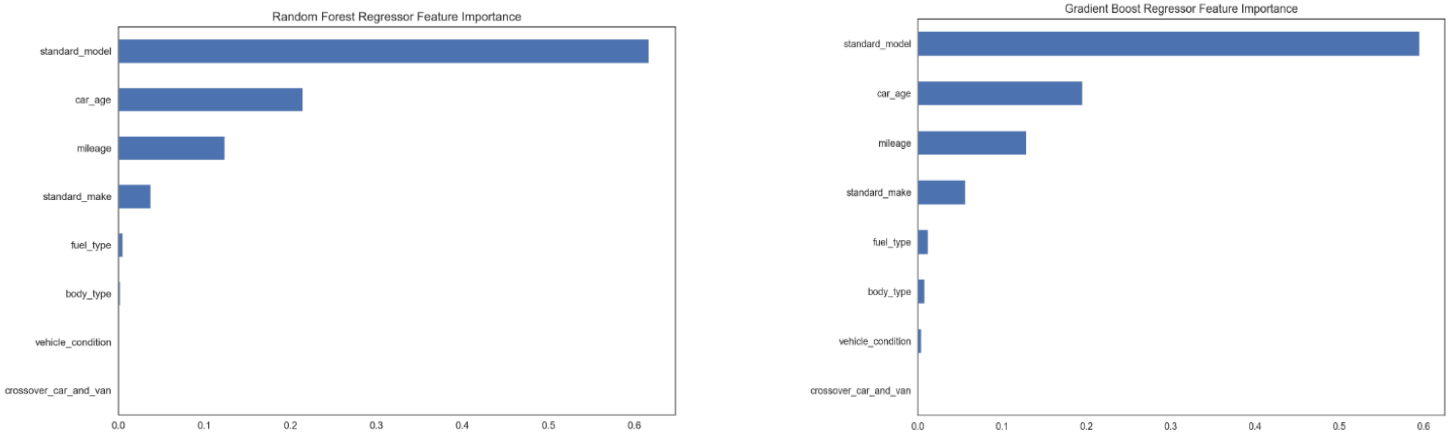


Figure 6b &6c. Feature Importance; Gradient Boosting Regressor and Voting Regressor respectively.

5.3 Global and Local Explanations with SHAP

Although model based feature importance tends to highlight features with significant influence on target, they lack interpretability. For example,How did the regressor model arrived at the particular predicted values?.

Shapley Additive exPlanations(SHAP) values are way of intepreting and expalining model output.

The local instance used in these shap explanations has a true value of £7,959.

Linear Regression Shap Explainer

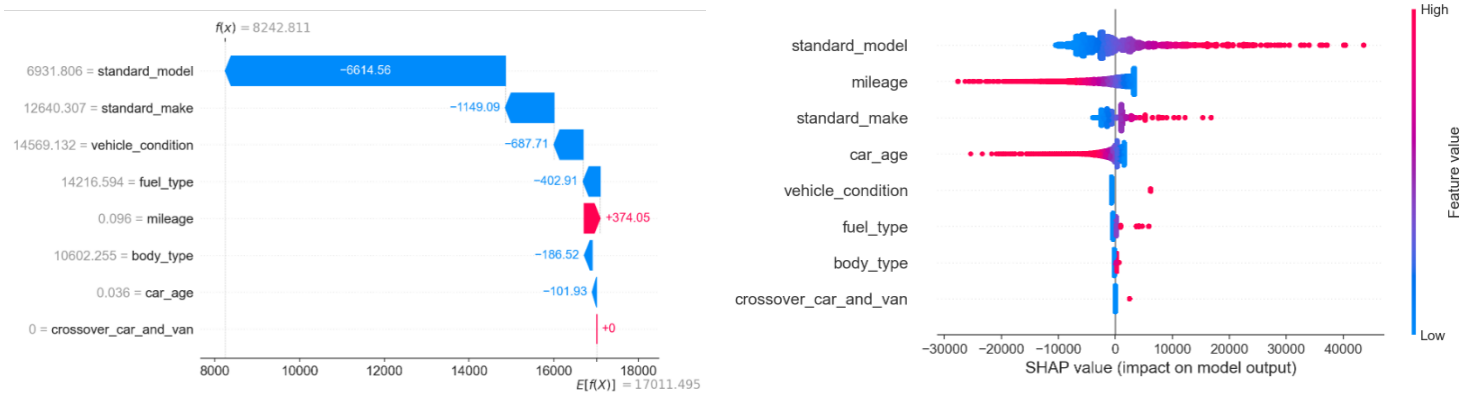


Figure 7a & 7b: Linear Regression Local and Global Explainer plots

Local Explanation:Figure 7a explains a predicted value of a single instance, with the base price being £17,011.The figure clearly shows the influence of each features as they affects the overall predicted price, with most of the features other than mileage having an inverse relationship with the predicted price value. Standard model have the most influence on this particular prediction value as it's value causes a £6614.58 value reduction in order to arrive at predicted price value being £8,242.811 for this paricular instance.

Global Explaination: Figure 7b explains the linear regression prediction for the entire test set. Vehicle condition, fuel type, body type and crossover car and van have little to no infleuence on the price values, mileage and car age havee an inverse relationship with price value, so strong that high values of these features could drastically reduce the expected price of cars by as much as 30,000. Standard make also have an direct proportional relationsip with car price.

While the infleunce of standard model tends to go both ways, the range of effect of high value of standard model is so pronounced that it could cause more than £40,000 increase in the predited price of a car.

Random Forest Regressor Shap Explainer

Local explainantion:Figure 7a shows the random forest regressor model explanation for a single instance.

Most of the features have no significant influence on this particlar instance, other than standard model causing a drastic reduction of £8455 in order to arrive at the predicted price value of £6485 .

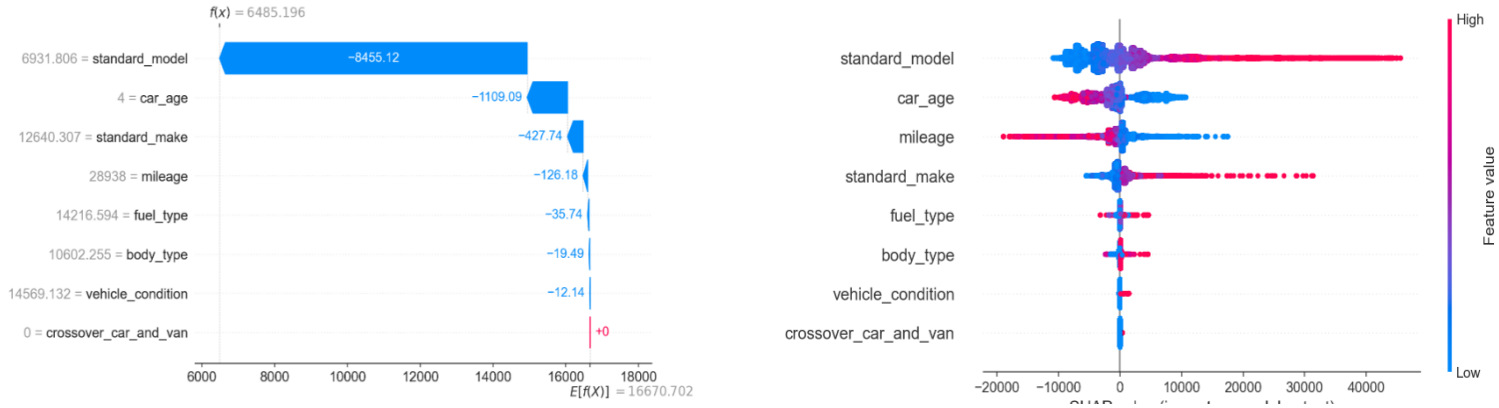


Figure 8a & 8b: Random Forest Regressor Local and Global Explainer plots

Global explanation: Figure 8b explains the random forest regressor prediction for the entire test set. Crossover car and van have no significant effect on determining the predicted value, fuel type and body type seems to have slight influence on price range seems balanced out. While the influence of car age and mileage appears to balanced out across the entire dataset, the direction of their influence on price of car is inverse i.e high value of these features causes decrease in predicted price of cars.

The standard make and standard model values are directly proportional to the predicted price values. High Standard make values have a wide range of effect on predicted price values as high as £35000 while that of standard model is as high as £45000.

Gradient Boosting Regressor Shap Explainer

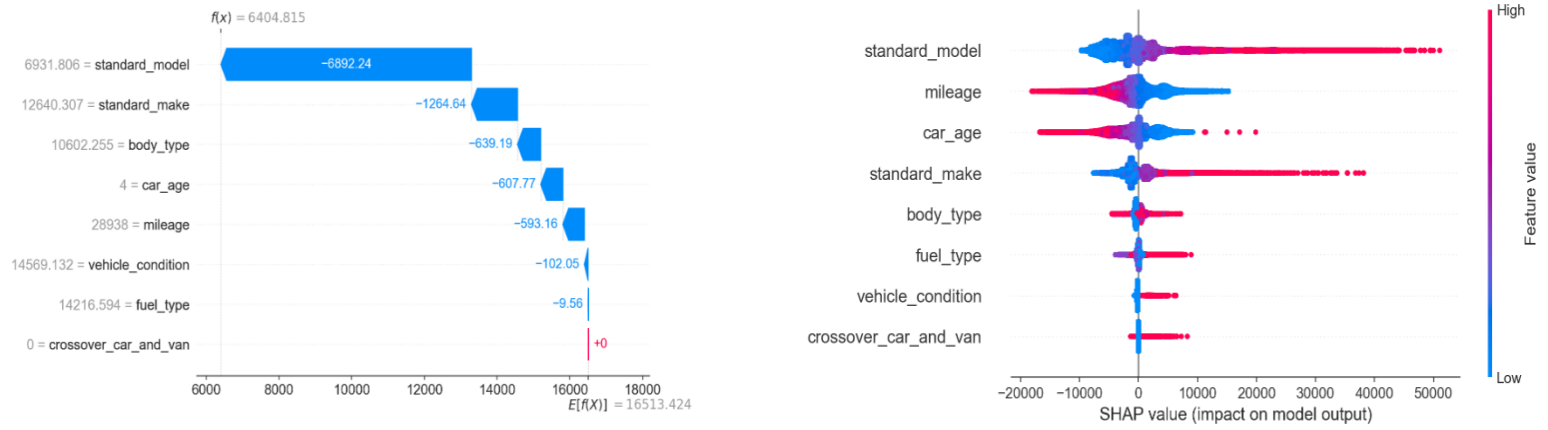


Figure 9a & 9b: Gradient Boosting Regressor Local and Global Explainer plots

Local explaination: Figure 9a shows the Gradient boosting regressor model explanation for a singlre instance Most of the features have no significant influence on this particlar instance, other than standard model causing a drastic reduction of £6892 in order to arrive at the predicted price value of £6,404.815.

Global explanation: Figure 9b explains the Gradient boosting regressor prediction for the entire test set. Crossover car and van, vehicle_condition, fueland body type had slight influence on determining the predicted value,with mostly high values of these aforementioned features leading to as much as £10,000 increase in price values. While the influence of car age and mileage appears to balanced out across the entire dataset, the direction of their influence on price of car is inverse i.e high value of these features causes decrease in predicted price of cars. It is also worth notting about 4 instances were the high car age caused an increase in their predicted values.

The standard make and standard model values are directly proportional to the predicted price values. High standard-make values have a wide range of effect on predicted price values as high as £40,000 while that of standard model is as high as £50,000.

5.4 Partial Dependency Plot

Partial dependency plot shows the effect of each features on the predicted target values, by making all other features constant, and also measurig the average marginal effect of the predictor on the predicted target value as the feature values changes.

Linear Regression

Partial dependency plot in linear regression always show a linear relationship between predictor values and corresponding target value. Figure 10. shows that body type, fuel type, crossover_ car and van, car age, vehicle condtion have a straight parallel line indicating that the effect changes in value of these features have no significant influence on the price of car. Mileage has a steady decline in the average marginal effect as it values increases. Standard Model has a sharp upward trend in its average marginal effect as it values increases, suggesting that for the linear regression model, standard model and mileage had the most signifcant effect on the price of the car

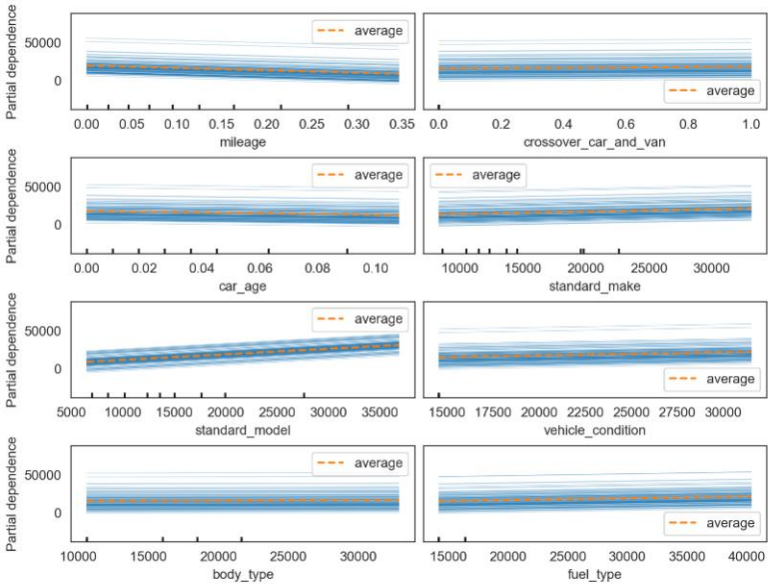


Figure 10. Linear Regression partial dependency plot

Random Forest Regressor

The mileage partial dependency plot shows that average marginal effect on car price reduces when the mileage value increases to 20000 and the average marginal effect was steady as mileage value increases until it got to 60000, then follow ed by anoher reduction in its effect on car price

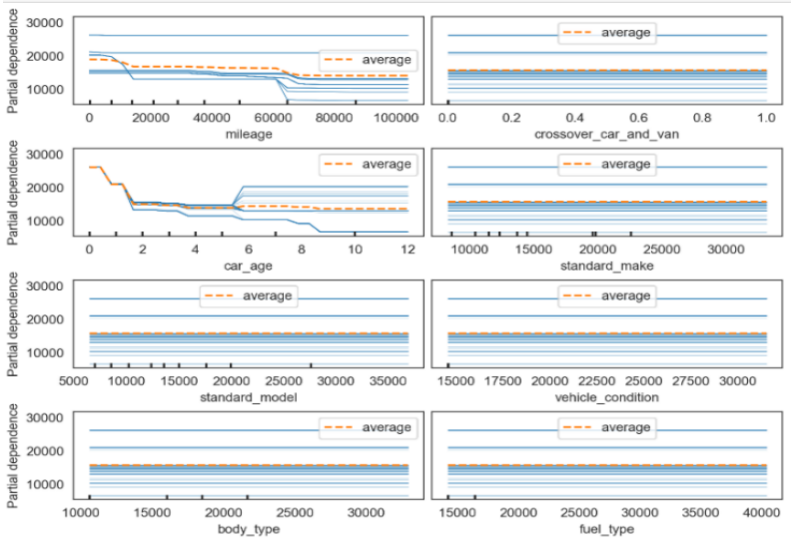


Figure 11. Random Forest Regressor partial dependency plot

The car age plot shows that between the age of 0 and 2, the average marginal effect of car age decreased drastically, but as the value of car age incrase beyond 2, there was no significant effect of in the changes in car age value on the price of car.

Figure 11. shows that body type, fuel type, crossover car and van, standard model, standard make, and vehicle condtion have a parallel average marginal effect line indicating that the effect of changes in values of these features have no significant influence on the price of car.

Gradient Boosting Regressor

For Gradient Boosting Regressor, the effect of mileage on price is similar to that in Linear RegressionFigure 11., with a steady decline in the average marginal effect as it values increases.

While increase in car age values also lead to a descrease in the average marginal effect, the effct wasn't steady as that of mileage.

Figure 12. shows that body type, fuel type, crossover car and van, standard model, standard make, and vehicle condtion have a parallel average marginal effect line indicating that the effect of changes in values of these features have no significant influence on the price of car.

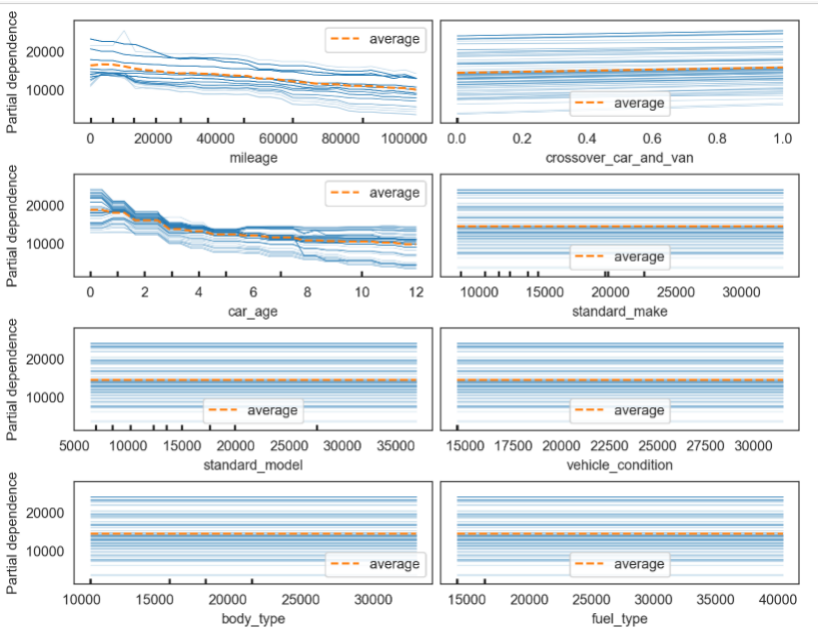


Figure 12. Gradient Boosting Regressor partial dependency plot