MACHINE LEARNING CONCEPTS

(6G7V0015_2425_9F) SUBMISSION BY SHAMILI GOVINDARAJ [24834013]
CONTACT: 24834013@STU.MMU.AC.UK

INTRODUCTION:

This report gives the comprehensive analysis and implementation of a machine learning pipeline for predicting car selling price using adverts dataset provided by AutoTraders. The main goal of this assignment is to build, evaluate, and analyse models to predict car prices based on their attributes.

Three machine learning algorithms-k-Nearest Neighbors, Decision Trees, and Linear Regression are utilized to build models.

1. DATA/DOMAIN UNDERSTANDING AND EXPLORATION

This section focuses on understanding the dataset, its structure, and the features present for building predictive models.

1.1. MEANING AND TYPE OF FEATURES; ANALYSIS OF DISTRIBUTIONS:

The given dataset consists of 12 columns with 402004 entries, each column represents a feature of the of a car with its corresponding selling price. The features include both categorical and numerical types.

```
car_data.info()
```

```
<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
```

	public_reference	mileage	year_of_registration	price
count	4.020050e+05	401878.000000	368694.000000	4.020050e+05
mean	2.020071e+14	37743.595656	2015.006206	1.734197e+04
std	1.691662e+10	34831.724018	7.962667	4.643746e+04
min	2.013072e+14	0.000000	999.000000	1.200000e+02
25%	2.020090e+14	10481.000000	2013.000000	7.495000e+03
50%	2.020093e+14	28629.500000	2016.000000	1.260000e+04
75%	2.020102e+14	56875.750000	2018.000000	2.000000e+04
max	2.020110e+14	999999.000000	2020.000000	9.999999e+06

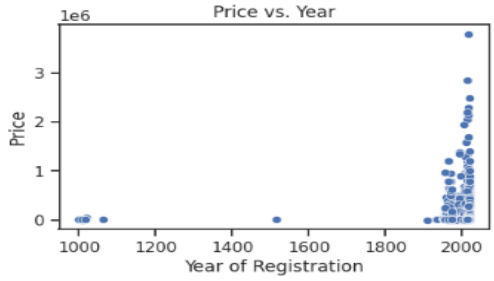
Numerical features include public reference, mileage, year of registration and 'price 'which is the target variable to be predicted. These features are represented as integer and float data types. On the other hand, the categorical features include reg code, standard colour, standard make, standard model, vehicle condition, body type, crossover, fuel type, all of which are of object data type.

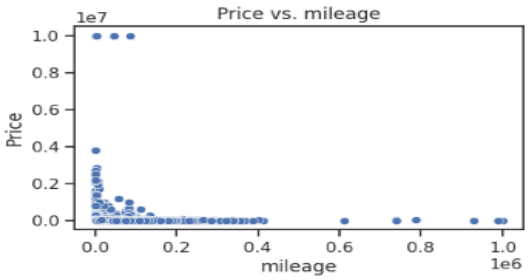
The statistical summary of the dataset represents the key characteristics of numerical features. It is noted that there is a vast variation in the mileage from 0 to 999,999 with the mean of the data set being 37743.6 and the high standard deviation indicating significant variability and the presence of outliers. The year of registration ranges from 999 to 2020, approximately at mean value of 2015 indicating the need for data cleaning. These observations show the importance of preprocessing steps like outliers removal and scaling for effective model performance. Finally, some of the column names were renamed to make them less complex and more easily understood, for instance public reference became ref_id, year of registration to year. This makes the dataset easier to manipulate during analyses as we can understand the values better.

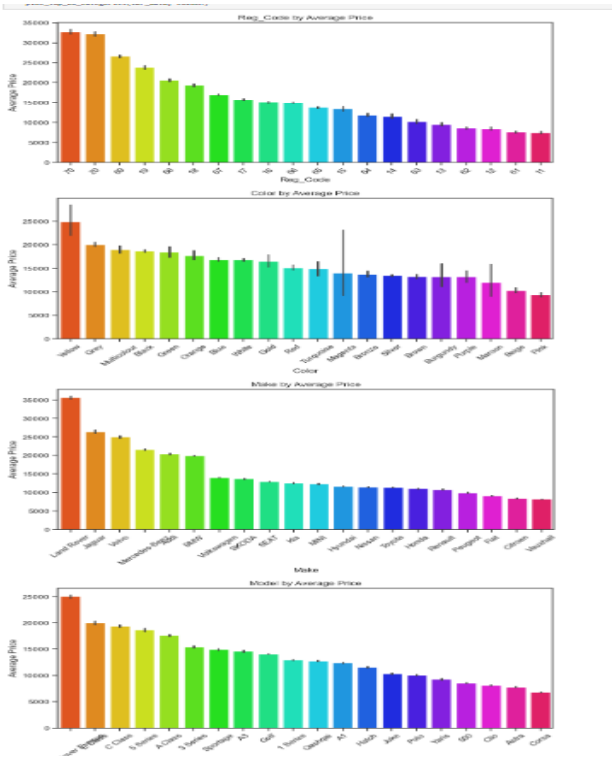
ref_id	mileage	reg_code	color	make	model	vehicle_condition	year	price	body_type	is_crossover	fuel_type
202006039777689	0.0	NaN	Grey	Volvo	XC90	NEW	NaN	11.211428	SUV	False	Petrol Plug-in Hybrid
202007020778260	108230.0	61	Blue	Jaguar	XF	USED	2011.0	8.853808	Saloon	False	Diesel
202007020778474	7800.0	17	Grey	SKODA	Yeti	USED	2017.0	9.546884	SUV	False	Petrol
202007080986776	45000.0	16	Brown	Vauxhall	Mokka	USED	2016.0	8.986697	Hatchback	False	Diesel
202007161321269	64000.0	64	Grey	Land Rover	Range Rover Sport	USED	2015.0	10.203444	SUV	False	Diesel

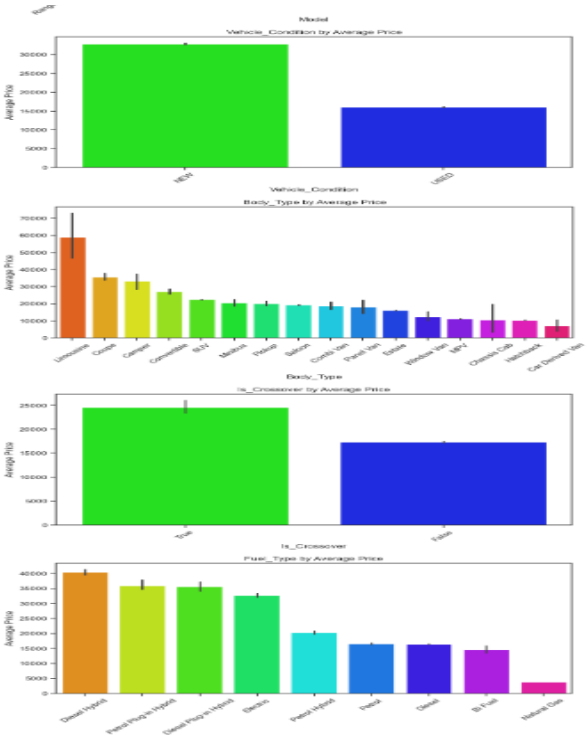
1.2. Analysis of Predictive Power of Features:

To analyze the predictive power of features, the distribution of the numerical features against the target variable `price' was done using a scatter plot. Specifically, mileage and year of registration were compared against price to understand their correlation. The scatter plot for mileage against price shows an inverse relationship, where cars with higher mileage generally have lower prices. This is similarly observed in the scatter plot of year of registration and price, establishing a positive relationship, where newer cars tend to have higher prices, though a few old car prices are also high, potentially due to their premium nature.









For Categorical features, the bar plot was generated for features such as make, model, body type, fuel type, vehicle condition, and others against the target feature price. In Categorical features, major columns have large number of unique categories, this was an issue for visualization. To resolve this, the analysis was refined by focusing on the top 20 categories for each feature.

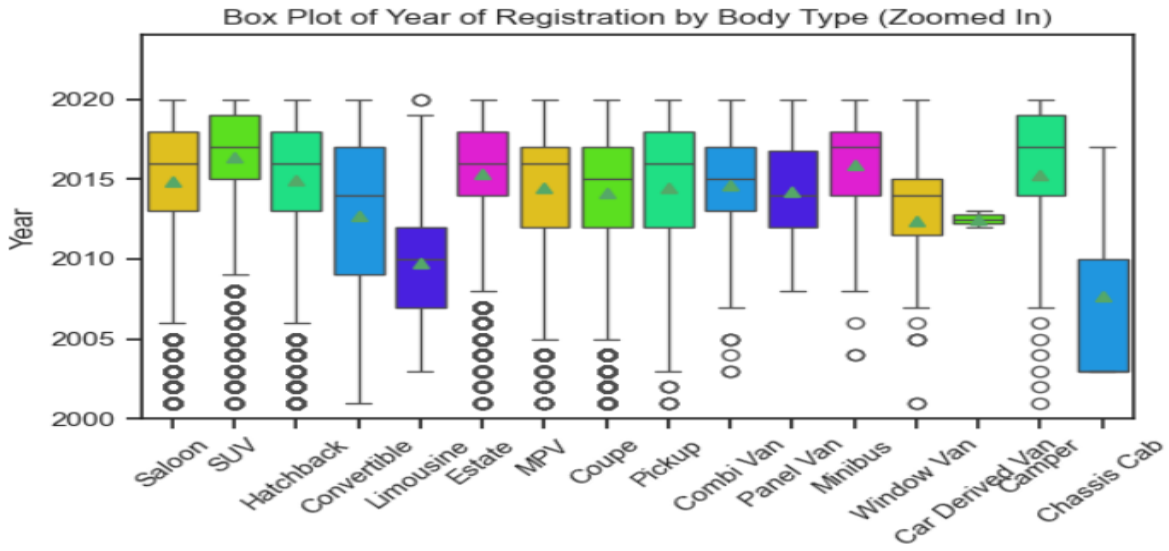
The comparison of price within categorical features indicated that the average price had strong differences depending on the category. From the plot, it shows that popular brands and unpopular brands had lower averages, and the highest prices were for luxury brands with models such as Land Rover Range Rover Sport. Sedan types like the limousines and coupes were relatively expensive than hatch backs and saloons. Hybrid and electric operated vehicles cost a premium to the regular fuel types like petrol and diesel. New vehicles and crossovers were also priced higher on average. Indeed, it can be concluded that the features within the best categories play a significant role in price estimation as it is supported by focused analysis below.

1.3. Data Processing for Data Exploration and Visualisation:

For the data processing, exploration and visualization step, the preprocessing was performed to prepare the dataset for effective exploration and to identify meaningful relationships between features. Firstly, the column Reference Id is transformed into a serial number format, this made the dataset more manageable and comprehensible by assigning each record with a unique, sequential identifier.

ref_id	mileage	reg_code	color	make	model	vehicle_condition	year	price	body_type	is_crossover	fuel_type
202006039777689	0.0	NaN	Grey	Volvo	XC90	NEW	NaN	73970	SUV	False	Petrol Plug-in Hybrid
202007020778260	108230.0	61	Blue	Jaguar	XF	USED	2011.0	7000	Saloon	False	Diesel
202007020778474	7800.0	17	Grey	SKODA	Yeti	USED	2017.0	14000	SUV	False	Petrol
202007080986776	45000.0	16	Brown	Vauxhall	Mokka	USED	2016.0	7995	Hatchback	False	Diesel
202007161321269	64000.0	64	Grey	Land Rover	Range Rover Sport	USED	2015.0	26995	SUV	False	Diesel

Next, the dataset was filtered to include only records where the year was greater than 2000. This step of filtering made it possible to focus only on recent and most relevant data. The filtered data is used to explore the relationship between numerical feature and the categorical feature. For example, the below graph is plotted between Year and Body type. This plot revealed trends in car registrations, that newer generations of registrations were more likely to include new body types and that older generations were likely to include specific old pattern body types.



Another visualization compared mileage with fuel type, shows how mileage varies across different fuel categories. The box plot highlighted the fact that hybrid & electric vehicles have comparatively lower mileage, and much fluctuating mileage rates in contrast to petrol & diesel vehicles. These comparisons provided insights into features which probably plays a significant role for predicting car prices.

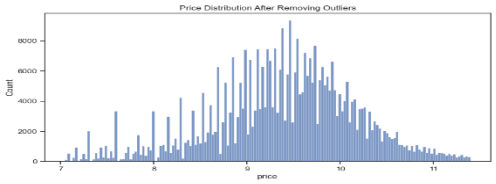
```
mileage              127
reg_code           31857
color               5378
make                   0
model                  0
vehicle_condition      0
year               33311
price                  0
body_type            837
is_crossover           0
fuel_type            601
dtype: int64
```

Additionally, missing value analysis was performed. It was observed that most of the columns contain missing values; the highest number was observed in reg code and year. Addressing these missing values is crucial for maintaining data quality and ensuring accurate model predictions in subsequent steps. Overall, all these preprocessing and visualization steps enabled to gain insights into the obtained dataset, understanding major connections between chosen features, as well as recognizing potential challenges which needs subsequent data cleaning and preparation.

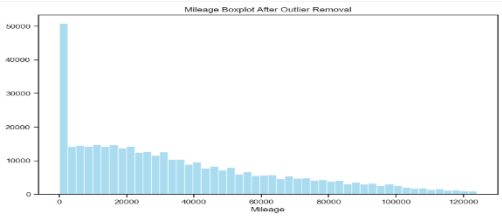
2. Data Processing for Machine Learning

Several data preprocessing operations were performed to prepare a good ML dataset including methods to handle missing values, outliers, noise and to derive some new features best suited for the model.

2.1. Dealing with Missing Values, Outliers, and Noise:



Outliers: Outliers in features like Price and mileage, are identified and addressed. For the Price distribution, a log transformation was applied, which normalized the distribution. Then the outliers were removed by filtering the data set based on percentile values.



Secondly for mileage, outliers are detected and handled by the interquartile range (IQR). This approach was useful in eliminating outside values beyond the computed upper and lower limits. The results were further checked using box plots, confirming that the majority of meaningful data was retained while extreme outliers being removed.

Missing Values and Noise:

To address the missing values, the data is first analysed to calculate the count and percentage of missing entries across all features. From the observation it was evident that reg code and year has the highest proportion of missing values, needs focused handling. This missing value of year was filled using two approaches, first for car marked as vehicle condition "NEW" the year was set to 2020. This prediction was based on the reference ID, which starts with 2020 for these entries, indicating the current year of registration. Secondly the missing years of other cars are estimated using reg code A formula was applied where, if the reg code was lesser than 50 the year was calculated as reg code + 2000. For larger values of reg code (i.e. greater than 50) the year was calculated as (reg code - 50) + 2000. By these two approaches the missing values in year are filled efficiently.

	Missing Values	Percentage
mileage	0	0.000000
reg_code	31167	8.075315
color	4886	1.265954
make	0	0.000000
model	0	0.000000
vehicle_condition	0	0.000000
year	32565	8.437534
price	0	0.000000
body_type	766	0.198469
is_crossover	0	0.000000
fuel_type	550	0.142504

Parallelly the missing reg code were also filled effectively, while handling year. The next step focused on handling the color feature which had the second highest percentage of missing data after the reg code. Its percentage is 1.27%, though relatively small, addressing these missing values was essential to maintain the dataset's integrity. To fill the missing values in the color, the most popular color of the feature was employed, which is obtained from the mode. This method ensured that the imputation did not disrupt the natural distribution of the data. Afterward, to check for the quality of the imputation of colors, a box plot was constructed to compare the distribution of color levels before and after imputation. After addressing missing values in color, other features with minimal missing values, such as fuel type, body type and mileage, were cleaned by dropping rows with missing data. The low percentage of missing values in these features made this approach feasible without reducing the size and diversity of the dataset.

	Missing Values	Percentage
mileage	0	0.0
reg_code	0	0.0
color	0	0.0
make	0	0.0
model	0	0.0
vehicle_condition	0	0.0
year	0	0.0
price	0	0.0
body_type	0	0.0
is_crossover	0	0.0
fuel_type	0	0.0
age	0	0.0
annual_mileage	0	0.0

After applying this method, all the missing values in this dataset were imputed and the noises including inconsistent entries or unwanted inputs were removed. These cleaning efforts resulted in a fully pre-processed dataset, free from missing values and noise, ensuring it was optimized for training robust machine learning models. This is because such thorough cleaning improves the dataset's completeness and quality for further analysis and modelling.

2.2. Feature Engineering, Data Transformations, Feature Selection:

Feature Engineering: In the feature engineering process, two more features were derived to better capture the characteristics of the cars. The first feature age was calculated using the formula: age=current year- year of registration, where current is predicted and set to 2020. This feature gives the straightforward representation of car age which is essential for understanding its depreciation and overall value. Based upon this feature, the next feature annual mileage of the car is derived using the formula: annual mileage = $\frac{mileage}{age}$. These features were visualized using box plots to check their distribution and make sure there were no outliers, or any unreasonable pre-processing done. Both features add meaningful insights to the dataset, with age representing the car's life span and annual mileage capturing usage trends that can significantly impact car pricing.

ref_id	age	annual_mileage
202007020778260	9.0	12025.555556
202007020778474	3.0	2600.000000
202007080986776	4.0	11250.000000
202007161321269	5.0	12800.000000
202009304412074	3.0	5333.333333
202007080998445	3.0	8025.000000
202009244143980	7.0	14142.857143
202010014442611	12.0	9269.666667
202006230431327	1.0	9500.000000
202007151278313	10.0	7700.000000

. 1.created features

```
<class 'pandas.core.frame.DataFrame'>
Index: 358023 entries, 202007020778260 to 201512149444029
Columns: 992 entries, mileage to color_Yellow
dtypes: float64(992)
memory usage: 2.6 GB
```

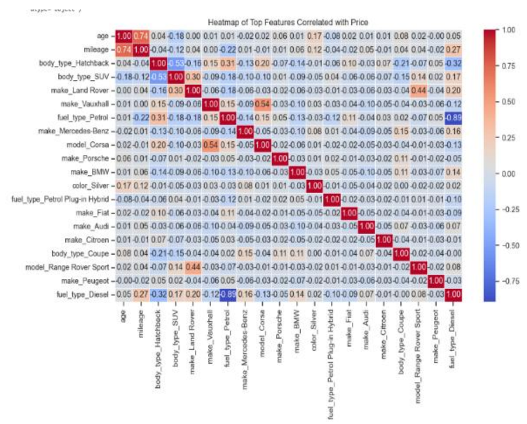
```
f_car_data_encoded.head()
```

make_AK	make_Abarth	make_Aixam	make_Alfa Romeo	make_Alpine	make_Ariel	...	color_Multicolour	color_Navy	color_Orange	color_Pink	color_Purple	color_Red	colo
0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	
0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	
0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	
0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	
0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	

2. Encoded Data set

Data Transformation: For data transformation, categorical features such as `make`, `model`, `fuel type`, `body type`, and `color` were not easily understandable by the machine learning models. To address this, one-hot encoding was applied, which transformed these categorical features into numerical values by creating binary columns for each unique category. Each category was represented as a column, with values of 0 or 1 of datatype float indicating the presence of data. After applying one-hot encoding, the dataset expanded to 992 columns, with all numerically converted data. One hot encoder is useful in coding the categorical data appropriately for the machine learning algorithms keeping the interdependence of the categories while excluding the dependence on the order. This initial step was important in creating a suitable format of the dataset for model training.

Feature selection: The next step of data transformation is feature selection, for this process a corelation maatrix was used to analyze the relationships between all features and the target variable, price. This helps to identify features with highest predictive powers.As per the analysis, first, 20 more features related to the selected variable with the most significant value of correlation coefficients were chosen.



So based on these results the finest top 10 features are selected to train and test the model. These features included both numerical and encoded categorical variables, ensuring a different set of predictors that capture essential information about the cars. Numerical features such as mileage, age, and specific categories like make Land Rover and body type SUV stood out as highly correlated with price. The desired features are then selected and 15% of overall data was retained for training and testing purpose. This reduced dataset retained the most relevant features and rows, ensuring that the model training process would focus only on the most meaningful predictors. This set of data was then pre-processed for training, optimizing the model's performance by minimizing noise and redundancy while preserving critical information.

3. MODEL BUILDING

This section covers the process of model building, including the selection of algorithms, data preparation, model configuration, training and evaluation.

3.1. ALGORITHM SELECTION, MODEL INSTANTIATION AND CONFIGURATION:

In this section, three different regression models were implemented: **K-Nearest Neighbors (KNN) Regressor**, **Decision Tree Regressor**, and **Linear Regressor**. The process for each model configuration and evaluation was structured as follows:

Feature Setting and Data Splitting: Feature selected in the previous process are identified as X and Y, where X are used as predictors and Y is the target variable "price". This dataset was split into training and testing sets using an 80-20 ratio, ensuring a robust evaluation setup.

Scaling and Pre-processing: Models such as KNN and Linear regressor, are sensitive to feature scales, So MinMaxScalers was applied to normalize the feature values between 0 and 1, which enhanced model accuracy and stability. Whereas, the Decision Tree Regressor, does not require scaling, and the data was used as-is.

```
# Scaling the Data
scaler = MinMaxScaler().set_output(transform='pandas')
scaler.fit(X_train)
```

```
MinMaxScaler
MinMaxScaler()
```

Model instantiation and Fitting: Default versions of KNN Regressor, Decision Tree Regressor, and Linear Regressor were instantiated using Scikit-learn. KNN Regressor was initialized with default hyperparameters such as the n_neighbours_7 and fit on the scaled training data. Similarly, the Decision Tree Regressor was configured with default parameters(path) which enabled the tree to fully grow, to test its capabilities. The Linear Regressor was instantiated using its default configuration to allow for a comparison with the proposed models.

Cross Validation: Each model was validated using cross-validation for making sure consistent performance across different data splits. To evaluate the performance of the model, mean of training and test data set, standard deviations of the training and test scores were calculated. As an example, the Decision Tree Regressor, gave a train score of 0.7057 and cross validation score of 0.6987 with a low Time standard deviation across the fold, hence moderate variation among the models. Similarly, the other model scores are given in below image:

```
print("\nMean Train Score:", scores_df['train_score'].mean())
print("Train Score Std Dev:", scores_df['train_score'].std())
```

```
Mean Train Score: 0.8366785388947285
Train Score Std Dev: 0.0003130988776570 53
```

```
print("\nMean Test Score:", scores_df['test_score'].mean())
print("Test Score Std Dev:", scores_df['test_score'].std())
```

```
Mean Test Score: 0.7816642944731163
Test Score Std Dev: 0.0017703913228947282
```

```
print("\nMean Train Score:", scores_df['train_score'].mean())
print("Train Score Std Dev:", scores_df['train_score'].std())
```

```
Mean Train Score: 0.7057996180271525
Train Score Std Dev: 0.001285513379294306
```

```
print("\nMean Test Score:", scores_df['test_score'].mean())
print("Test Score Std Dev:", scores_df['test_score'].std())
```

```
Mean Test Score: 0.6987561086231597
Test Score Std Dev: 0.0041714473793074336
```

```
# Mean and Std of Train/Test Scores for Linear regressor
print("\nMean Train Score:", scores_df['train_score'].mean())
print("Train Score Std Dev:", scores_df['train_score'].std())
```

```
Mean Train Score: 0.7740835282561142
Train Score Std Dev: 0.0006960882771531995
```

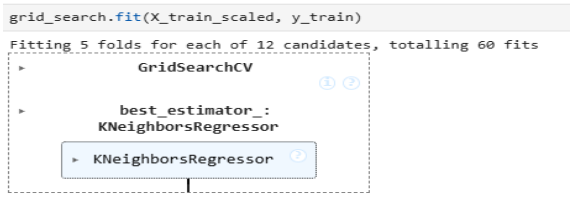
```
print("\nMean Test Score:", scores_df['test_score'].mean())
print("Test Score Std Dev:", scores_df['test_score'].std())
```

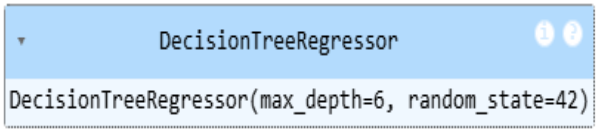
```
Mean Test Score: 0.7739096619501721
Test Score Std Dev: 0.0027602868767334907
```

Evaluation and scores: For model performance, the measures including R^2, Mean Squared Error (MSE), and Mean Absolute Error (MAE) have been computed for both training and testing data sets. R^2 (Coefficient of Determination): Shows to what extent the variation in the target variable is covered by the model. Mean Squared Error (MSE): Measures the average squared deviation of each value from the mean value or value which was predicted by the model. Mean Absolute Error (MAE): The other is used to estimate the average magnitude of the errors in the predictions.

3.2. GRID SEARCH, AND MODEL RANKING AND SELECTION:

For enhancing the model performance, hyperparameter tunning is done. For Knn Regressor, the hyperparameter tuning is done using grid search, where a set of values called neighbors, ranging from 1 to 25 was tested. The optimal value of n_neighbors was identified, in order to improve the model performance. Then the tuned Knn model is used for prediction and evaluation.





Similarly, for decision tree regressor the following are considered as hyperparameters for tuning: max_depth and random_state to control the model's complexity and its capacity for generalization. In case of Linear regressor, the model was assessed with no parameter tuning, due to its simplicity and straightforward nature, it should not be a problem. With these all these tuning and implementation techniques the models are built to be a good predictor for the selling prices of car.

MODEL RANKING: The model ranking is done based on their performance metrices like R^2 (Coefficient of Determination), Mean Squared Error (MSE), Mean Absolute Error (MAE). They help to conclude how close the model predicted values are to the actual values and also measures to which extend the model minimizes the error.

	Model	R^2 Score	Mean Squared Error (MSE)	Mean Absolute Error (MAE)
0	KNN	0.7957	0.1129	0.2441
1	Decision Tree	0.7063	0.1622	0.3043
2	Linear Regression	0.7746	0.1245	0.2567

Based on this above its clear that the KNN Regressor model was the most effective one – it has the highest R^2 (0.7957) meaning it indicates 79.57% of the target variability. Also, the model possessed the least MSE of 0.1129 and MAE of 0.2441 which indicates the model possessed good predictability. This is credited to the hyperparameters tuning through the grid search that was conducted.

The Linear Regressor ranked second highest with a R^2 score of 0.7746. Although its MSE as 0.1245 and MAE as 0.2567 were somewhat higher than the KNN Regressor, it offered consistently satisfactory base measures.

The Decision Tree Regressor, although capturing non-linear relationships, ranked last with an R^2 score of 0.7063 and the highest error metrics (MSE: 0.1622, MAE: 0.3043). It is seen that it has relatively higher error values that refer to lower generalization accuracy than other models.

MODEL SELECTION: The model selection process is carried out by evaluating the model's performance. Calculation of performance is done using the metrices like R^2, MSE and MSA. Comparing all three algorithms, the KNN Regressor was revealed to be the most accurate model with the highest R^2 coefficient of 0.7957, and the lowest of MSE at 0.1129 and the MAE of 0.2441. The Linear Regressor came next because such a model was dependable; however, the Decision Tree Regressor was third mainly because of the higher errors and slightly lower efficiency of the predictions made. In general, the KNN Regressor was chosen as the final model of predicting car prices because the model has a robust capacity to generalize across the given data set.

```
The best model for predicting car selling price is : KNN Regressor
```

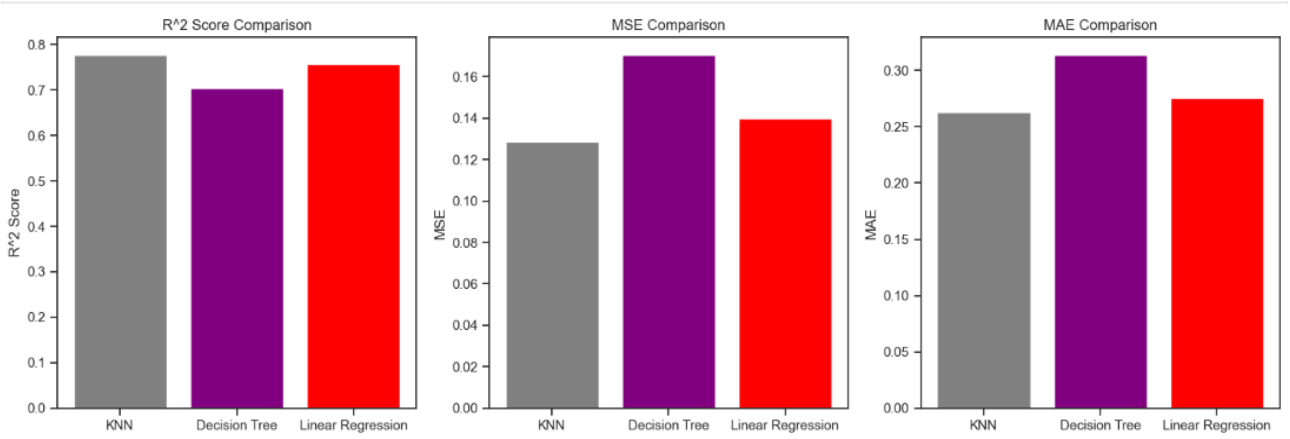
4. Model Evaluation and Analysis

4.1. Coarse-Grained Evaluation/Analysis:

The coarse-grained evaluation focused on assessing the overall performance of the models using three key metrics: R^2, MSE and MSA. To make a side-by-side comparison of these metrics, a summary table was made showing the observations of each of these models- KNN Regressor, Decision Tree Regressor and Linear Regressor. The KNN Regressor has the highest R^2 while having the lowest values of error. To validate these results, all the above three models were cross verified, producing a table of mean and standard deviation for both train and test scores. It was done to make sure that the models will perform equally well no matter what condition of the data is used.

	Model	Train Mean Score	Train Std Dev	Test Mean Score	Test Std Dev
0	KNN	0.8366	0.0003	0.7816	0.0017
1	Decision Tree	0.7057	0.0012	0.6987	0.0041
2	Linear Regression	0.7740	0.0006	0.7739	0.0027

In addition to tabular analysis, bar plots were created to visualize the R^2 Score, MSE, and MAE for all three models. These visuals exposed the performance comparison and provided an intuitive understanding of the metrics. It establishes that KNN Regressor had the best performance. Linear Regressor has demonstrated fair performance turning in the second overall result which is less significant than KNN. The Decision Tree Regressor though effective in capturing non-linear relationships, gave the lowest R^2 and highest errors.



To further analyse the model behaviour an overfitting and underfitting analysis was done through the comparison of scores between the training sets and testing sets. If the train score was significantly higher than the test score, the model was considered overfitting. Conversely, a much lower train score compared to the test score indicated underfitting. In this evaluation, the KNN Regressor shows a good performance with small overfitting or underfitting, which also indicates why the KNN Regressor was chosen as the most balanced model. The below picture shows that three models are balanced with the definite train and test scores. The **KNN Regressor** achieved the highest consistency with a train score of **0.8135** and a test score of **0.7511**, indicating the superiority in predicting accurate values. The decision tree and linear regressor shows consistent scores, but both are lower than that of KNN regressor.

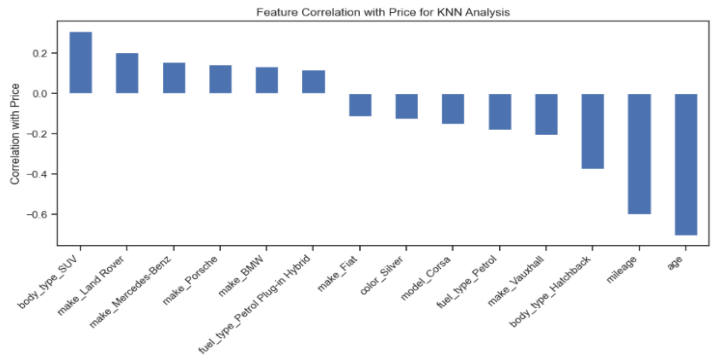
```
Overfitting/Underfitting Analysis:
KNN: Balanced (Train Score: 0.8366, Test Score: 0.7816)
Decision Tree: Balanced (Train Score: 0.7057, Test Score: 0.6987)
Linear Regression: Balanced (Train Score: 0.774, Test Score: 0.7739)
```

These gave a detailed understanding of their efficiency and accuracy of predictions, and the chosen model was the KNN Regressor due to its high comparative performance.

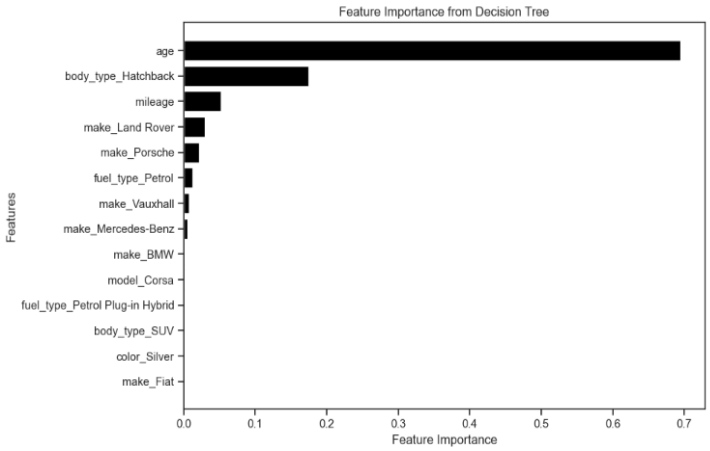
4.2. Feature Importance:

Feature importance evaluation was conducted to understand the influence of various features on car price prediction. This analysis was done separately for each of the three models- the KNN Regressor, decision tree regressor and the linear regressor, using the methods applicable for each.

KNN Feature Correlation Analysis: KNN regressor utilizes correlation between features and the target variable, the analysis was done to determine the variables' importance. It was visualized using bar plot, which reveals the relative importance of features, where body_type_SUV, make_Land Rover, and make_Mercedes-Benz showed positive correlations with price. Conversely, age and mileage exhibited strong negative correlations, indicating that older and more used vehicles tend to have lower prices. These results align with expectations in the vehicles market. Thus, this analysis affirms the utility of these features in constructing the models hence the need to incorporate them when estimating prices.

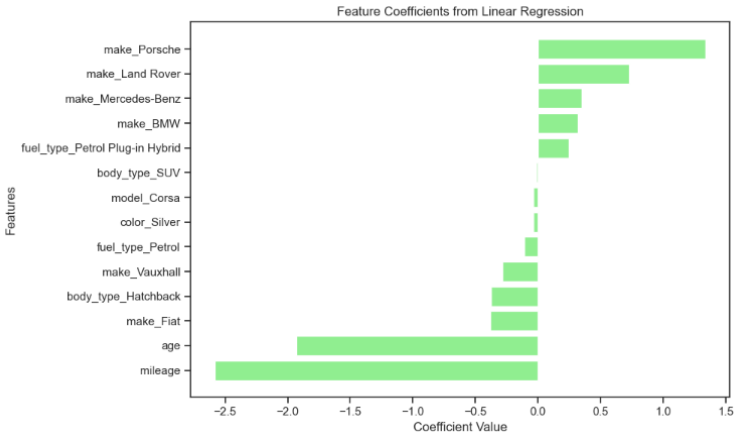


Feature Importance for Decision Tree Regressor:



The Decision Tree Regressor has an internal split rules which decides the importance of the features. The results of the box plot showcase that age is a key feature of car prices, body_type_Hatchback and mileage as the second and third considerations respectively. However, to a lesser degree the other features such as make_Land Rover and make Porsche also confounded this relationship. This also highlights the tree model's ability to capture non-linear relationships and prioritize features that impact price heavily.

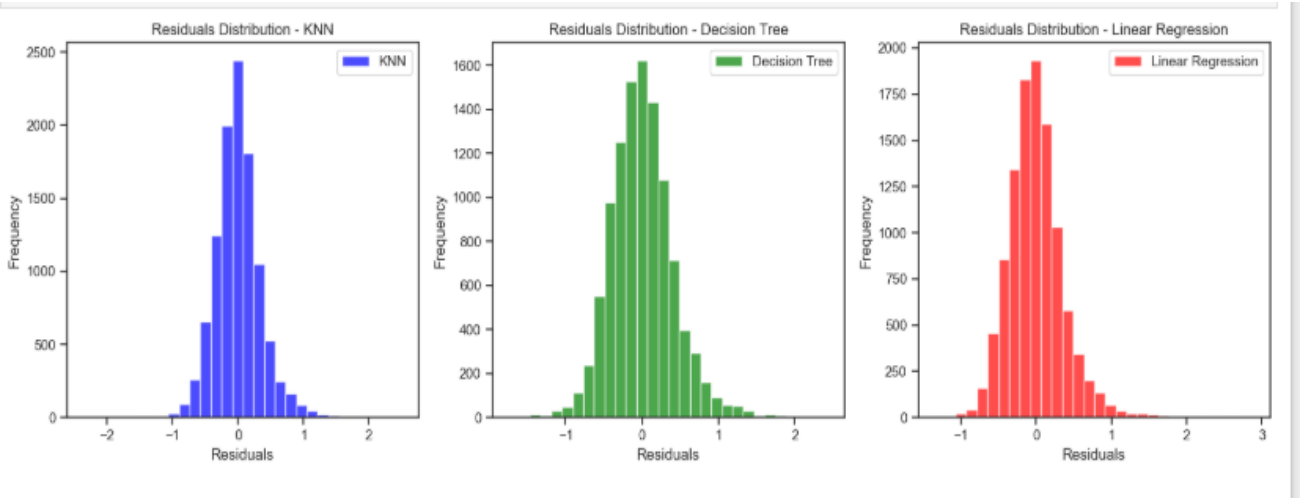
Feature Importance from Linear Regressor: The Linear Regression model established a quantitative significance measure of features influence on the price through the coefficients of determination. Out of all available features, both make_Porsche and make_Land Rover had the highest positively-coefficients, pointing at their price premium effects. However, there was a high degree of emphasis placed on age and mileage with negative co-efficient that supported the deduction of value from the car. The coefficients of features like fuel_type_Petrol Plug-in Hybrid and make_BMW also indicated a notable influence. In all models, it was found that age and mileage were the most sensitive predictors of low car prices. Indeed, many brand-related features including make_Porsche and make_Land Rover demonstrated positive influence. The difference in feature importance between the models show that the methods applied in each model are distinct and the analysis gives a comprehensive understanding on feature relevance for predicting car prices.



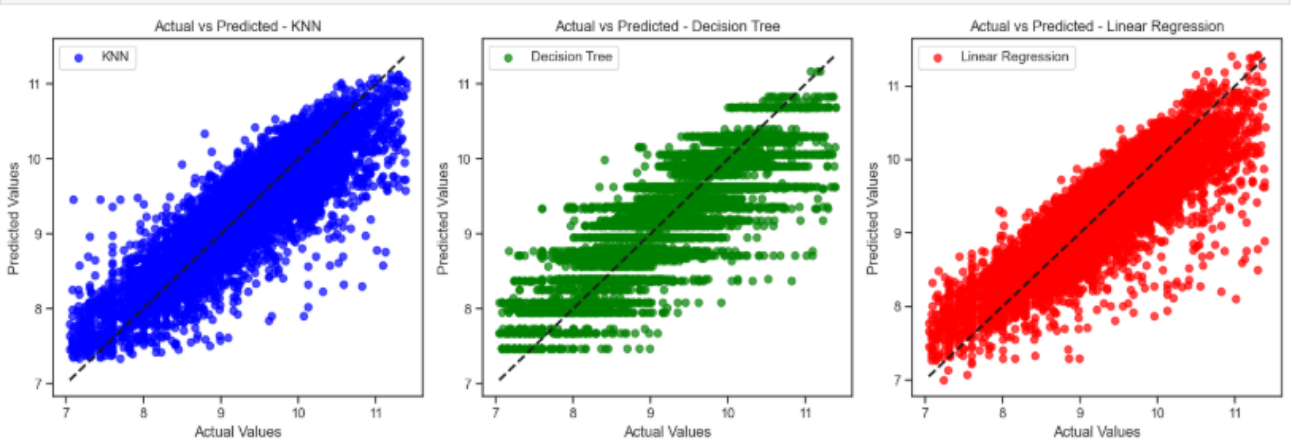
4.3. FINE-GRAINED EVALUATION:

In the fine-grained evaluation, a detailed analysis of the model was carried out at the residual-level and the instance-level to analyse the model performance concerning the specific errors and the general accuracy. The process included three key steps: Residual analysis, instance level analysis as well as performance visualisation and finally the overall performance of up to RMSE and MAPE. These analyses not only quantified the models' accuracy but also highlighted their specific strengths and weaknesses in capturing the patterns within the dataset.

Residual Analysis: A thorough residual analysis was performed to understand the error distribution for each of the three models: Knn, decision and linear regressors. Histogram plots were made from the residual values which are calculated by subtracting the actual values with the predicted ones. These histograms showed that residuals are cantered near zero for all three models, indicating a satisfactory level of fit. Out of the three models, KNN had a closer spread of residuals to the centre showing that this model was best suited to reduce the errors of prediction.



An instance-level analysis was performed to look at the actual and predicted values of a few instances to have a **granular perspective** at how the models performed. In all the three models the actual values as were plotted against the predicted values using scatter plots to check their fit. In general, the observed performance of the KNN and Linear Regression models appeared to be more aligned to the diagonal (ideal prediction line), which in turn confirmed their ability to capture the key trends in the data sets**.**



Error Metrics Evaluation: To assess the model performance, error metrics were computed. All three models were validated using Root Mean Square Error (RMSE) and Mean Absolute Percentage Error (MAPE). The prediction results revealed least RMSE of 0.3585 and least MAPE of 2.84% in the case of KNN, which

indicates the superior predictive accuracy of the model. The Decision Tree as powerful modelling tool that captured some of the non-linear relationships showed higher errors represented by RMSE (0.4131) and MAPE (3.39%) comparing with previous models. Linear Regression also performed well with the RMSE (0.3740) and MAPE (2.97%).

```
Error Metrics:
KNN - RMSE: 0.3360, MAPE: 2.64%
Decision Tree - RMSE: 0.4028, MAPE: 3.29%
Linear Regression - RMSE: 0.3528, MAPE: 2.76%
```

The detailed analysis given helped in understanding the specifics of the given models and identified that among them all KNN is the best fit for the given dataset. Its capacity to generate residuals that are closely packed, more accurate fit between estimated values and observed values and the least error statistics all pointed to higher levels of actual prediction accuracy. Linear Regression offered a close alternative, but the Decision Tree model although capable to detect the non-linear relationship had comparatively higher error rates for this task. This detailed analysis ensures a well-informed decision regarding the model's applicability and robustness for future predictions.

CONCLUSION:

This report presents an overview of the machine learning workflow, staring from the data pre-processing stage and through model development and model assessment stages. Three models—KNN, Decision Tree, and Linear Regression-were utilized, with each being pre-processed, tuned, fine-tuned, and trained. Evaluations were conducted based on error metrics and predictive accuracy. This concludes that the KNN regressor was proven to achieve high accuracy in this diverse car dataset in order to predict an approximate price of the car for sale. The R^2 scores and evaluation metrics across all three models were strong, highlighting the robustness of the approach. The reliability and consistency of predictive results highlight the importance of iterative approaches, as demonstrated by this work.