TAYLOR'S UNIVERSITY

Wisdom · Integrity · Excellence

August 2023 Semester

Machine Learning and Parallel Computing

(ITS66604)

Assignment 1

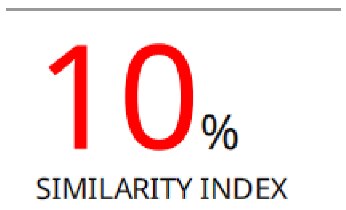
Individual Assignment (20%)

Submission Date: 11TH November 2023

STUDENT DECLARATION

1. *I confirm that I am aware of the University's Regulation Governing Cheating in a University Test and Assignment and of the guidance issued by the School of Computing and IT concerning plagiarism and proper academic practice, and that the assessed work now submitted is in accordance with this regulation and guidance.*
2. *I understand that, unless already agreed with the School of Computing and IT, assessed work may not be submitted that has previously been submitted, either in whole or in part, at this or any other institution.*
3. *I recognise that should evidence emerge that my work fails to comply with either of the above declarations, then Imay be liable to proceedings under Regulation.*

Student Name	**Student ID**	**Date**	**Signature**	**Score**
Tang Wai Kin	0346747	24/10/23	alvin	

Evidence of Originality	
Similarity Score:  10% SIMILARITY INDEX	**AI-Writing Index:**

Marking Rubrics (Lecturer's Use Only)		
Criteria	**Weight**	**Score**
Introduction, Research Goal & Objectives	10	
Related Works	20	
Methodology	15	
Implementation & Results	25	
Analysis & Recommendations	15	
Conclusion	5	
Submission Requirements	10	
Grading *Excellent 90 – 100 marks* *Good 75 – 89 marks* *Fair 40 – 74 marks* *Poor 0 – 39 marks*	Total Marks (100%)	
	Total Marks (20%)	
Remarks: Google Colab Link: https://colab.research.google.com/drive/186GF7M3tzCUEWrdxs1XA8emExV6H14OY?usp=sharing		

Acknowledgements

I would like to express my heartfelt gratitude to Ms Nicole Teah Yi Fan for guiding me throughout this assignment. Your support, guidance, and encouragement have been invaluable, and I am deeply appreciative of your efforts.

I also want to thank the countless authors, scholars, and experts whose work served as the foundation for my research.

This individual assignment would not have been possible without the combined efforts and support of all these individuals. I am truly grateful for the guidance and encouragement that I have received throughout this individual assignment.

Thank you all for being part of this incredible journey.

Sincerely,

Tang Wai Kin 0346747

Abstract

The escalating costs of new cars have prompted a surge in the used-car market, creating a need for fair and accurate pricing mechanisms. However, the growth of this market has also led to concerns about fraudulent pricing practices by dealers. To address these issues, this research proposes a machine learning-based used-car price prediction system, utilizing a dataset from Kaggle. The study evaluates three machine learning models—linear regression, polynomial regression, and decision tree—considering attributes like car brand, model, mileage, and accident history. The research aims to identify the most accurate model in predicting used-car prices while minimizing errors (MAE, MSE, RMSE) and optimizing through regularization and hyperparameter tuning. Additionally, insights from data visualization techniques aid in understanding data patterns and outliers. A review of related works underscores the significance of model fine-tuning and larger datasets in achieving higher prediction accuracy. The research's scope involves feature selection, encoding categorical variables, and model optimization. The methodology includes dataset acquisition, data preprocessing, the implementation of machine learning models, model optimization, and model evaluation using performance metrics. The overarching goal is to contribute a reliable pricing tool for the used-car market, benefiting both buyers and sellers in making informed decisions.

Table of Contents

1.0 Introduction
- 1.1 Research Goal
- 1.2 Research Objectives

2.0 Related works
- 2.1 Gap Analysis
- 2.2 Scope of research

3.0 Methodology
- 3.1 Dataset
- 3.2 Data Preparation
- 3.3 Models
- 3.4 Model Evaluation
- 3.5 Summary of methodology

4.0 Implementation and Results
- 4.1 Initial EDA
- 4.2 Descriptive EDA
- 4.3 Modelling
  - 4.3.1 Model 1
  - 4.3.1.1 Experiment 1a
  - 4.3.1.2 Experiment 1b
  - 4.3.1.3 Summary of Model 1
  - 4.3.2 Model 2
  - 4.3.2.1 Experiment 2a
  - 4.3.2.2 Experiment 2b
  - 4.3.2.3 Summary of Model 2
  - 4.3.3 Model 3
  - 4.3.3.1 Experiment 3a
  - 4.3.3.2 Experiment 3b
  - 4.3.3.3 Summary of Model 3
- 4.4 Summary of Implementation and Results

5.0 Analysis and Recommendations

6.0 Conclusions

7.0 References

1.0 Introduction

The market of used-car market has been growing rapidly in these recent years, as the price of a new car has been increasing up to 4.2% year by year in 2023 January due to increase in cost of production and government tax according to J.P Morgan Research [1]. As a result, many customers nowadays tend to consider for alternative option like used-car instead of buying a brand-new car from the manufacturer, which lead used-car sales has a significant increased over the year [2].

However, as the used-car market is growing, many fraud cases have been reported in the market as numbers of used-car dealers are taking advantage of its customer by fixing at an unrealistically undervalue and overvalue price for the used-car to increase its own profitability which led to many customers being exploited [3]. To address this major issue, it is essential to have a used-car price prediction system that is widely available for the public, to avoid any further exploitation and efficiently assess the used-car's worth based on a range of factors like "Car brand", "Model", "Milage", "Accident", and more.

Implementation of machine learning into predicting used-car price offers numerous opportunities and benefits for both customers and dealers. In terms of customer, machine learning will provide more precise and accurate price value of the used-car according to its attributes, this ensures the customer in getting a fair offer by the used-car dealer without being exploited. Moreover, customers also could use the prediction provided by the machine learning to reduce the risk of depreciation of its car after purchasing it [4]. Other than that, machine learning offers opportunities to used-car dealer too. With machine learning, used-car market will be more transparent as used-car value will be predicted and boost its customer confidence on the dealer which will lead to higher sales and market grow in the used-car market [4].

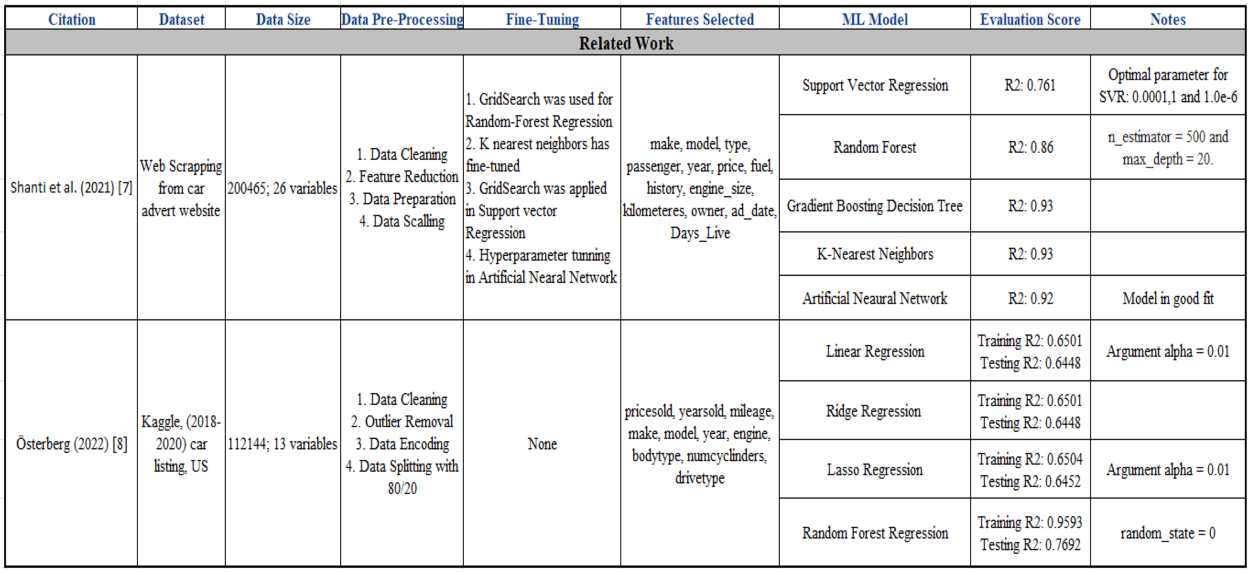
1.1 Research Goal

The research goal of developing machine learning method for prediction of used car price is to gauge the price value of used car effectively according to its attributes like "Car brand", "Model", "Milage", "Accident", and more. Therefore, the proposed of this research paper is to utilize the used car dataset retrieved from Kaggle [5] and evaluate various implementation of machine learning techniques like linear regression, polynomial regression, and decision tree, it will be used to reliably predict the price value of the used car and determine how different machine learning techniques will output different prediction of used car's price. Lastly, the prediction of used car price value from this research will be able to facilitating the buyer and seller in better decision making.

1.2 Research Objectives

The objective of this research is to utilize 3 different machine learning models (Linear regression, Polynomial regression, and decision tree) in to predicting the used car price value and evaluate which machine learning model output the highest accuracy (R2-Score) while maintaining low error (MAE, MSE, and RMSE). Other than that, optimizing the regression model by implementing regularization and hyperparameter tunning is one of the research objectives as well. This will further improve the performance of the predictive model and may output a higher accuracy while lower the error, then the performance metrics of both optimized predictive model and baseline model will analyse and compared. Lastly, the research objective also includes data visualization like scatter plot, bar chart, heatmap, pair plot, and others. As data visualization enable us to swiftly identify the data pattern, trends, and outlier easier [6].

2.0 Related Works



Citation	Dataset	Data Size	Data Pre-Processing	Fine-Tuning	Features Selected	ML Model	Evaluation Score	Notes
Related Work								
Shanti et al. (2021) [7]	Web Scrapping from car advert website	200465; 26 variables	1. Data Cleaning 2. Feature Reduction 3. Data Preparation 4. Data Scalling	1. GridSearch was used for Random-Forest Regression 2. K nearest neighbors has fine-tuned 3. GridSearch was applied in Support vector Regression 4. Hyperparameter tunning in Artificial Nearal Network	make, model, type, passenger, year, price, fuel, history, engine_size, kilometeres, owner, ad_date, Days_Live	Support Vector Regression	R2: 0.761	Optimal parameter for SVR: 0.0001,1 and 1.0e-6
						Random Forest	R2: 0.86	n_estimator = 500 and max_depth = 20.
						Gradient Boosting Decision Tree	R2: 0.93	
						K-Nearest Neighbors	R2: 0.93	
						Artificial Neaural Network	R2: 0.92	Model in good fit
Österberg (2022) [8]	Kaggle, (2018-2020) car listing, US	112144; 13 variables	1. Data Cleaning 2. Outlier Removal 3. Data Encoding 4. Data Splitting with 80/20	None	pricesold, yearsold, mileage, make, model, year, engine, bodytype, numcyclinders, drivetype	Linear Regression	Training R2: 0.6501 Testing R2: 0.6448	Argument alpha = 0.01
						Ridge Regression	Training R2: 0.6501 Testing R2: 0.6448	
						Lasso Regression	Training R2: 0.6504 Testing R2: 0.6452	Argument alpha = 0.01
						Random Forest Regression	Training R2: 0.9593 Testing R2: 0.7692	random_state = 0

Figure 1: Table of Related Works

2.1 Gap Analysis

According to the table of related works Figure 1, we able to analyse that machine learning models in the first study (Shanti et al. ,2021) [7] that have been through fine-tunning is enable of getting a higher evaluation score compared to the second study (Osterberg, 2022) [8] that has no fine-tunning implemented into its machine learning models. For example, both studies have utilized random forest regression model in predicting used car's price value but the first study (Shanti et al. ,2021) has applied Sklearn gridsearch into its random forest regression model to fine-tune the number of decision tree (n_estimators) and maximum dept of the decision tree (max_dept) while the second study (Osterberg, 2022) did not fine-tune its random forest regression model and used the default parameter argument. In result, the first study with fine-tune was able to get a significant higher R2 score of 0.86 while the second study without fine-tune achieve a lower R2 score of 0.7692. This concludes that machine learning models perform better with fine tuning, as it will obtain the best value for the parameter.

Moreover, larger data size of the dataset will improve the prediction score of the machine learning models. By comparing both studies, the first study (Shanti et al. ,2021) has utilized a larger amount of data at 200465, while the second study (Osterberg, 2022) has only utilized 112144 data which is half of the first study's dataset. In result, the first study with a larger amount of data has a higher average evaluation score between all the machine learning models than the second study. Therefore, it indicates that data size plays a significant role in achieving high evaluation score for the machine learning model.

2.1 Scope of Research

The scope of this research includes features selection where both categorical and numerical variables will be selected into implementation of the machine learning model according to its correlation and relevancy with the target variable (used car price), while low correlation and irrelevance features will be removed to avoid complexity. Additionally, categorical data will be encoded into numerical format by using one hot encoding and label encoding method which then can be used into the machine learning model for better prediction performance. Other than that, one of the machine learning models will be optimized by using fine-tune for further improvement as according to gap analysis above we analysed that utilizing fine tuning will significantly improve the prediction performance. Lastly, model evaluation will be conducted in this research between all the 3 machine learning models and the optimized model such as linear regression, polynomial regression, and decision tree model. This is to ensure which machine learning model was able to perform in the highest accuracy while maintaining low error in predicting used car's price value.

3.0 Methodology

Below is the architectural methodology of this research, as shown in Figure 1. The goal of this architectural methodology is to ensure that the development for the machine learning model of used car's price prediction meets the requirements of outputting accurate predictions, as methodology acts as a guide for the whole development process. The methodology consists of Dataset, Data Preprocessing, Machine Learning Models, Model Optimization, and Model Evaluation.

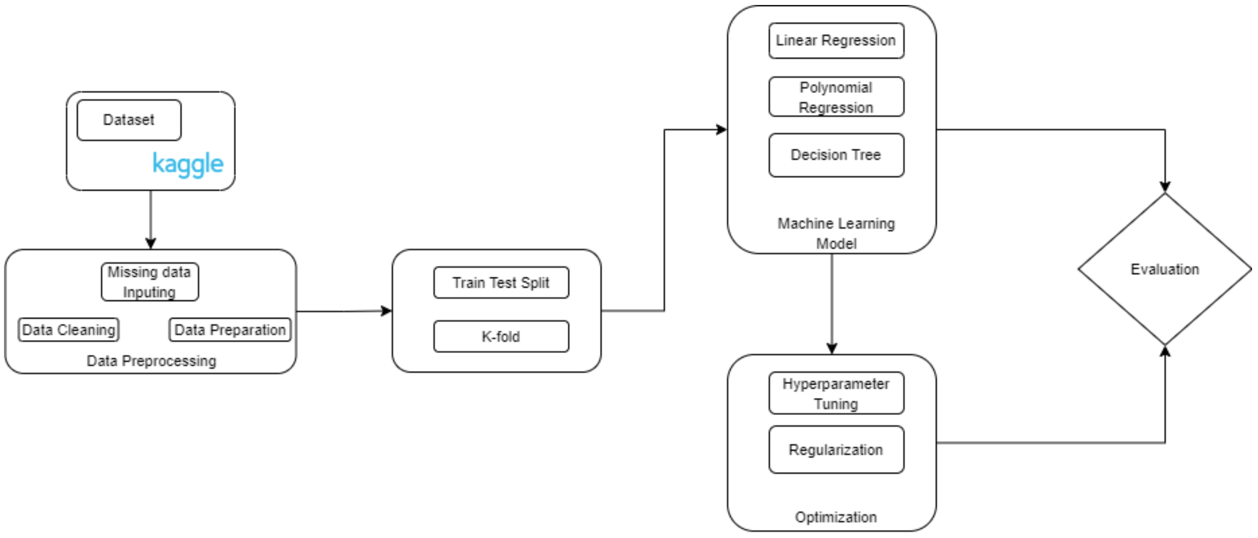
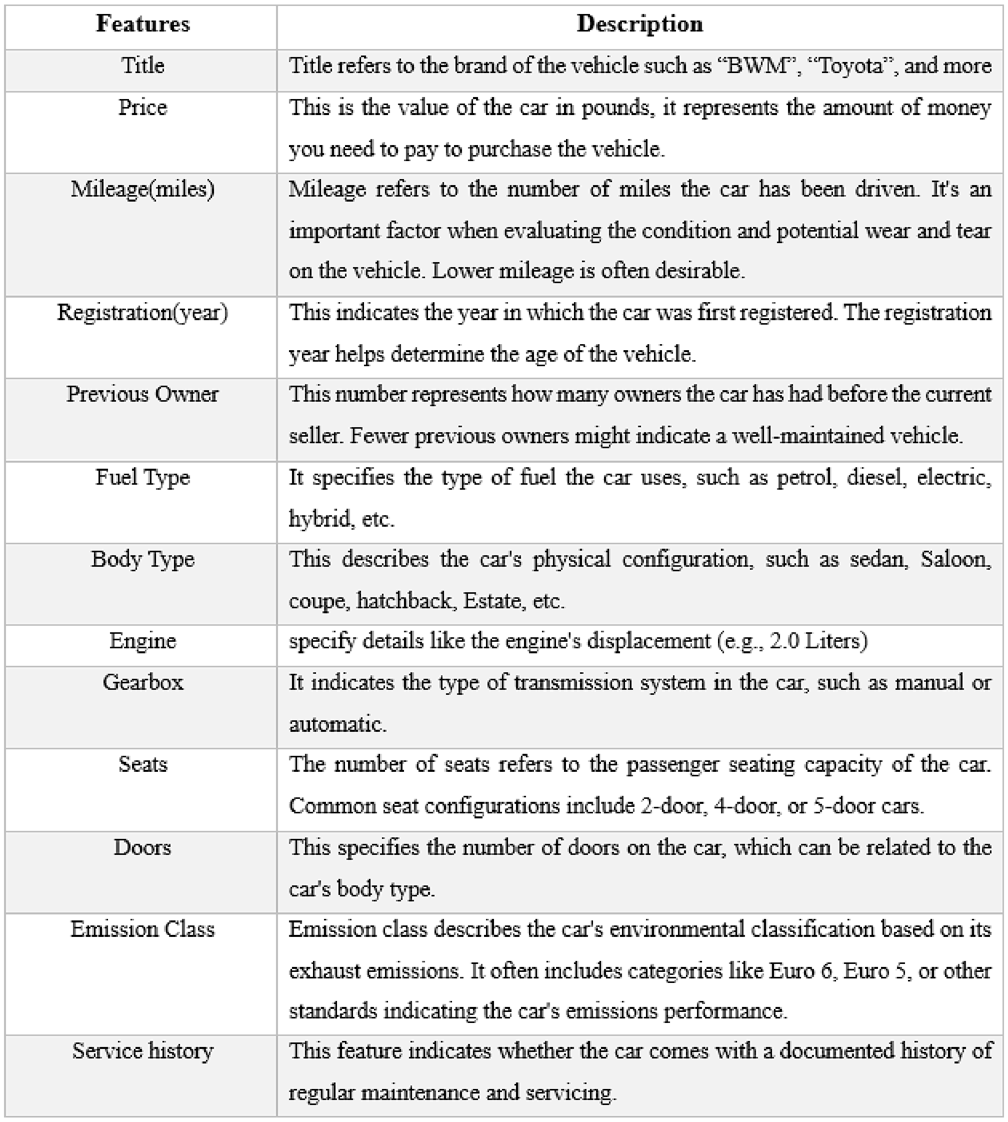


Figure 2: Methodology architecture of used car price value prediction

3.1 Dataset

The first process of this development and research is to retrieve dataset that is subject to used car price value, the dataset that is sourced then will be trained on 3 different machine learning models in the following process. There are various of widely available dataset website, but in this research, I have chosen Kaggle as my main source of dataset for the price of used car. The dataset that I have retrieve includes 9 numerical variables (ID, price, mileage, registration year, previous owner, engine, doors, seats, and emission class) and 5 categorical variables (title, fuel type, body type, gearbox, and service history) which in total has 14 variables and a total of 3685 data rows. Additionally, the main reason to source for a significantly large dataset is to maximise the accuracy of the prediction, it is also important that the dataset is in high quality in terms of accurate data, representation of real-world data and complete for the machine learning model to learn from [9].



Features	Description
Title	Title refers to the brand of the vehicle such as "BWM", "Toyota", and more
Price	This is the value of the car in pounds, it represents the amount of money you need to pay to purchase the vehicle.
Mileage(miles)	Mileage refers to the number of miles the car has been driven. It's an important factor when evaluating the condition and potential wear and tear on the vehicle. Lower mileage is often desirable.
Registration(year)	This indicates the year in which the car was first registered. The registration year helps determine the age of the vehicle.
Previous Owner	This number represents how many owners the car has had before the current seller. Fewer previous owners might indicate a well-maintained vehicle.
Fuel Type	It specifies the type of fuel the car uses, such as petrol, diesel, electric, hybrid, etc.
Body Type	This describes the car's physical configuration, such as sedan, Saloon, coupe, hatchback, Estate, etc.
Engine	specify details like the engine's displacement (e.g., 2.0 Liters)
Gearbox	It indicates the type of transmission system in the car, such as manual or automatic.
Seats	The number of seats refers to the passenger seating capacity of the car. Common seat configurations include 2-door, 4-door, or 5-door cars.
Doors	This specifies the number of doors on the car, which can be related to the car's body type.
Emission Class	Emission class describes the car's environmental classification based on its exhaust emissions. It often includes categories like Euro 6, Euro 5, or other standards indicating the car's emissions performance.
Service history	This feature indicates whether the car comes with a documented history of regular maintenance and servicing.

3.2 Data Preprocessing

After sourcing dataset of used-car price, the following process is data preprocessing, data preprocessing is a critical and important process before implementing into the machine learning model because dataset that we have often sourced from the internet may include duplicated data, missing values, and outlier values which will lead to lower accuracy or higher error rates of prediction produce by the machine learning model. The 3 main data preprocessing's components are missing data inputting, data cleaning, and data preparation.

Missing data values occur in the dataset can be commonly found and it is crucial to identify and manage the missing data values by using the isnull() function before moving on to the machine learning modelling process. There are various techniques to approach the missing data values including removing the whole data column that has missing value as this is the most time-saving and straightforward method to deal with missing data [10]. Other than that, it can utilize statistical measures like the mean, median, or mode to fill up the missing values in the dataset [10].

Data cleaning mainly focuses on fixing irregularities such as the outlier and identifying errors like duplicated data column in the used car dataset that we have sourced. Data cleaning is an important component to the data preprocessing as it is capable of maintaining high quality of the dataset and ensuring the performance of the machine learning will not be impacted due to the low quality of the dataset. Typically, the most commonly used method in data cleaning is elimination where outliers that are significantly depart from the remainder of the dataset will be removed and duplicated data are found will be remove from the dataset as well [11].

Data preparation process entails converting dataset into a format that allows the machine learning model to understand better and improve predictions performance. Some of the common methods used in data preparation includes one hot encoding and label encoding where categorical variables like car brand, model, and fuel type will be converted into numerical format which then can be used in the machine learning modelling process to increase the accuracy of the predictions [12]. Moreover, by decreasing the number of variables that will be implement into the machine learning model will be able to avoid overfitting occurs in the predictions is also known as a data preparation method [12].

3.3 Machine Learning Models

Predicting the price value of used cars are based on its variables/features and features of used cars are commonly a combination of both categorical variables and numerical variables. Therefore, to achieve the best performance it is important to utilize few different machines learning model to have a better understanding of the dataset and achieve a more robust prediction [13]. In this research, we will be implementing 3 different machine learning model including linear regression, polynomial regression, and decision tree.

Linear regression is a supervised machine learning model that is capable of identifying the best fit between a dependent variable and one or more independent variables that has a linear relationship

[14]. The equation of linear regression is $Y = b0 + b1 * X1 + b2 * X2 + ... + bn * Xn$ (nD equation). The reason of linear regression will be the implement and utilize to predict the price value of used car in this research because according to a study [15], the researchers have achieved R2 score of 0.73 (73%) by implementing linear regression into predicting used car price with a different dataset. Therefore, in this research it may achieve a similar or a improve prediction performance due to different use of dataset and features. On top of that, linear regression's representation is much simpler compared to other machine learning model.

Polynomial regression is extremely useful when the data points in the dataset is not in a straight line as the main goal of polynomial regression is to identify the curve fit to the data and make accurate predictions [16]. Polynomial regression is similar to linear regression, but it captures a non-linear relationship between the dependent variable and independent variables instead of linear relationship. The equation of polynomial regression is $y\beta\ (x) = \beta 0 + \beta 1x + \beta 2x\ 2$. The reason of implementation of polynomial regression into this research due to its flexibility as it provides more flexibility in capturing complex and non-linear relationships in the dataset. On top of that, another Kaggle project has conducted research to predict used car price with polynomial regression model and realised it was effective in predicting used car prices as it has achieved R2 score of 0.89 (89%) [17]. Therefore, experience different dataset from that study with polynomial regression may even achieve a higher accuracy performance.

Decision tree is a non-parametric supervised machine learning model that can be utilized for both regression and classification tasks. The components of decision tree consist of internal nodes, leaf nodes, branches, and root node which make up its hierarchical tree structure and decision tree works by splitting the data into 2 datasets at each node where the dept of the tree indicates how many times the data has been split by the model [18]. Lastly, the leaf nodes of the decision tree are represented as the final decision. The reasons of utilizing decision tree in this research is due to its clear visualization of the decision-making process which make the model easy to understand and interpret compared to other models [18]. Other than that, Decision tree is highly scalable as it able to handle large amounts of data and this factor is important because predicting used car's price are based on large number of features and data.

3.4 Model Evaluation

The performance metrics that will be used to evaluate the machine learning model's performance in this research includes Mean Absolute Error (MAE), Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and R Square (R2). The reason behind of the chosen metrics is because it is commonly used in regression tasks especially in predicting prices [19], therefore classification performance metric like Accuracy, Precision, Recall, and F1 score are not suitable to be evaluated in this research.

- Mean Absolute Error (MAE), it measures the average absolute difference between the predicted and actual value. It is easy to understand however it doesn't penalize large error as much as other metrics.
- Mean Squared Error (MSE), it measures the average squared difference between predicted and actual value like MAE, however it penalized larger error more heavily compared to MAE.
- Root Mean Squared Error (RMSE), it is the square root of MSE which provides an interpretation in the same units as the target variable.
- R Square (R2), it measures the proportion of the variance in the target variable that is predicted by the model, which also a measure of the goodness of the fit.

3.5 Summary of Methodology

The research methodology revolves around the development of machine learning model in used car's price prediction. It begins with dataset acquisition from Kaggle, comprising 12 variables and 4010 data columns. Data preprocessing is critical, involving handling missing data, cleaning irregularities, and preparing data for machine learning. Then three different machine learning models are deploy including linear regression for linear relationship, polynomial regression for non-linear relationship data, and decision tree for clear visualization and scalability. Lastly, the model will be evaluated by utilizing 4 performance metrics, MAE, MSE, RMSE, and R2. The main goal of this methodology is to ensure the development for the machine learning model of used car price prediction meets the requirements of outputting accurate predictions, as methodology acts as a guide for the whole development process.

4.0 Implementation and Results

Link of this implementation in Google Colab:

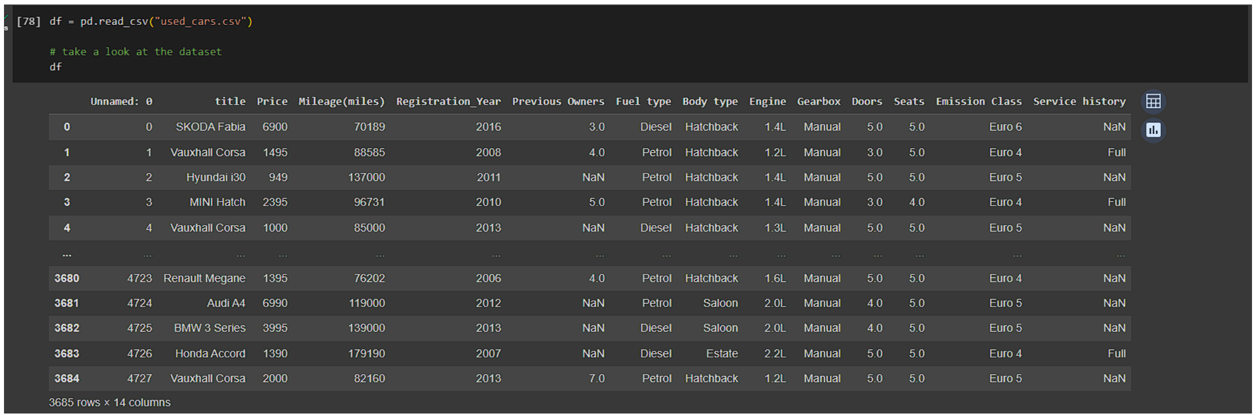
https://colab.research.google.com/drive/186GF7M3tzCUEWrdxs1XA8emExV6H14OY?usp=sharing

4.1 Initial EDA

In this following section of initial EDA, it will mainly focus on data inspection to identify any possible missing values in the dataset, basic data visualization to produce simple visualization of the data such as line plot, histogram plot, and bar chart plot. Moreover, it focuses on data summary such as identifying mean, median, standard deviation of the numerical features in the dataset. Other than that, it also includes handling missing values by filling missing data with mean value and removing unwanted label in the data.

Firstly, a (used car price) dataset was sourced from Kaggle and imported into Google Colab to have a brief view of the dataset and it contains a total of 3685 rows and 14 columns. The variable/features in this dataset have a total of 9 numerical variables (ID, price, mileage, registration year, previous owner, engine, doors, seats, and emission class) and 5 categorical variables (title, fuel type, body type, gearbox, and service history).

By analysing the dataset in figure 3, it is noticeable that some of the numerical data contained unwanted label such as "L" in the engine data and "Euro" in the data of the emission class. Removing those irrelevant characters are extremely important as machine learning model requires numeric data to operate, therefore by removing unwanted characters enables us to perform numerical operation, calculations, and statistical analyses of the dataset. To remove unwanted character such as "L" in the "Engine" data and "Euro" in the "Emission Class" data, I've utilized "str.replace()" method to replace unwanted character with an empty string efficiently, and "astype()" method to convert both "Engine" and "Emission class" variables to numerical format.



```
[78] df = pd.read_csv("used_cars.csv")

     # take a look at the dataset
     df
```

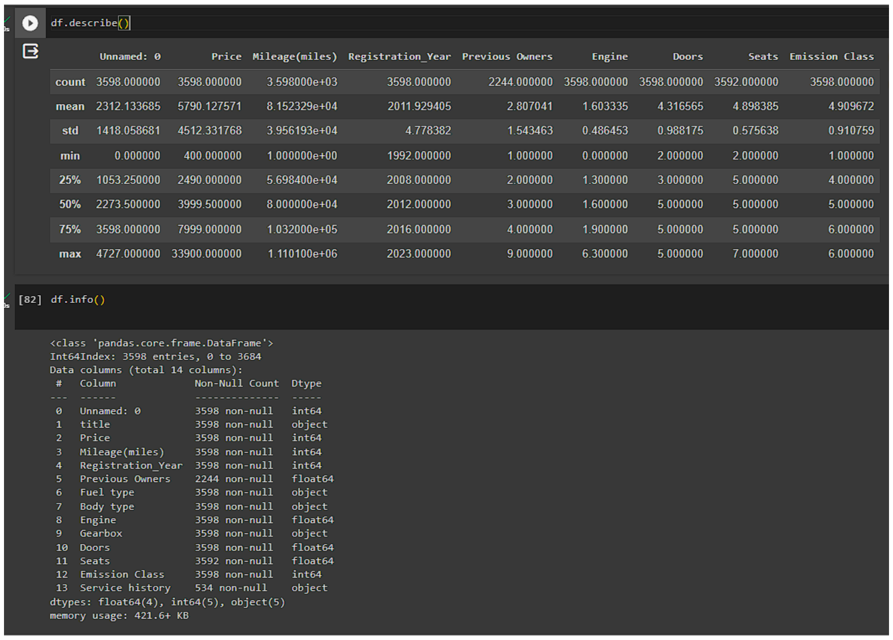
	Unnamed: 0	title	Price	Mileage(miles)	Registration_Year	Previous Owners	Fuel type	Body type	Engine	Gearbox	Doors	Seats	Emission Class	Service history
0	0	SKODA Fabia	6900	70189	2016	3.0	Diesel	Hatchback	1.4L	Manual	5.0	5.0	Euro 6	NaN
1	1	Vauxhall Corsa	1495	88585	2008	4.0	Petrol	Hatchback	1.2L	Manual	3.0	5.0	Euro 4	Full
2	2	Hyundai i30	949	137000	2011	NaN	Petrol	Hatchback	1.4L	Manual	5.0	5.0	Euro 5	NaN
3	3	MINI Hatch	2395	96731	2010	5.0	Petrol	Hatchback	1.4L	Manual	3.0	4.0	Euro 4	Full
4	4	Vauxhall Corsa	1000	85000	2013	NaN	Diesel	Hatchback	1.3L	Manual	5.0	5.0	Euro 5	NaN
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
3680	4723	Renault Megane	1395	76202	2006	4.0	Petrol	Hatchback	1.6L	Manual	5.0	5.0	Euro 4	NaN
3681	4724	Audi A4	6990	119000	2012	NaN	Petrol	Saloon	2.0L	Manual	4.0	5.0	Euro 5	NaN
3682	4725	BMW 3 Series	3995	139000	2013	NaN	Diesel	Saloon	2.0L	Manual	4.0	5.0	Euro 5	NaN
3683	4726	Honda Accord	1390	179190	2007	NaN	Diesel	Estate	2.2L	Manual	5.0	5.0	Euro 4	Full
3684	4727	Vauxhall Corsa	2000	82160	2013	7.0	Petrol	Hatchback	1.2L	Manual	5.0	5.0	Euro 5	NaN

3685 rows × 14 columns

Figure 3: Overall view of the dataset.

After removing unwanted label in the data, I've utilized df.describe() method to output a summary of the descriptive statics of the dataset such as the count, mean, standard deviation, min value, percentile, and max value of the dataset. An overview of statistics of the dataset enables us to summarize complex data into easily understandable chunk [20]. According to the output figure 4, the target variable (Price) has a mean value of 2312.13, minimum value of 0, and maximum value at 4727. While variable (Previous Owner) has a mean value of 2.8, minimum value of 1, and maximum value at 9. This information can be very useful as we are able to utilize the mean value to fill up possible missing value in the dataset.

Moreover, df.info() method is used to output a summary of the dataset's structure and its contents such as the number of non-null values, data type, and memory usage. According to figure 4, it showed that most variables such as title, price, mileage (miles), registration year, fuel type, body type, engine, gearbox, doors, seats, and emission class has a non-null value around 3598. While variable previous owner and service history has a significant low non-null value at 2244 and 534. Other than that, the dataset structure includes a total of 4 float variables, 5 integer variables, 5 object variables, and a memory usage of 421.6+ KB.



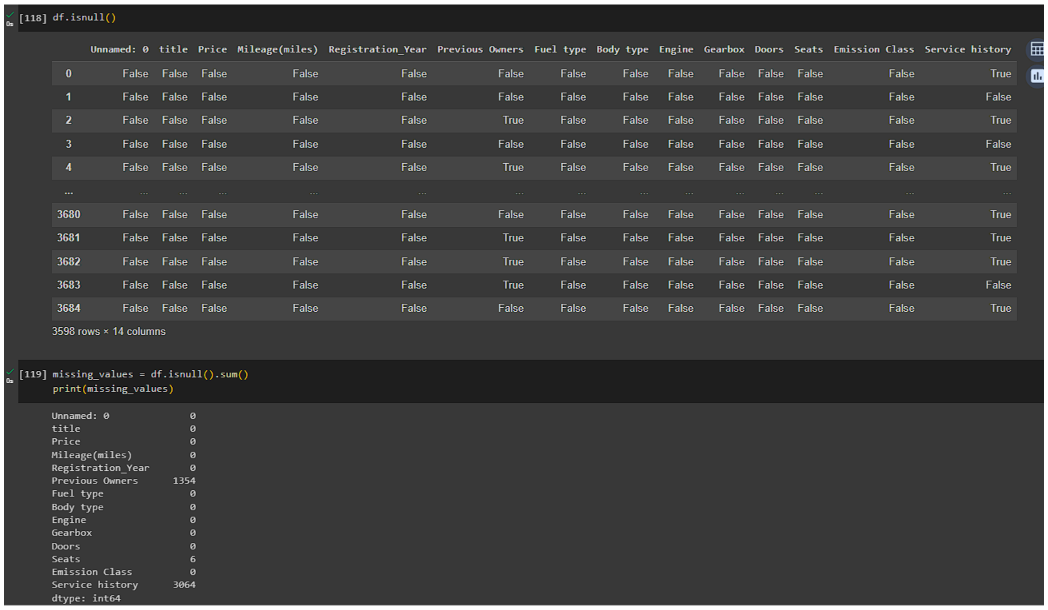
```
df.describe()
```

	Unnamed: 0	Price	Mileage(miles)	Registration_Year	Previous Owners	Engine	Doors	Seats	Emission Class
count	3598.000000	3598.000000	3.598000e+03	3598.000000	2244.000000	3598.000000	3598.000000	3592.000000	3598.000000
mean	2312.133685	5790.127571	8.152329e+04	2011.929405	2.807041	1.603335	4.316565	4.898385	4.909672
std	1418.058681	4512.331768	3.956193e+04	4.778382	1.543463	0.486453	0.988175	0.575638	0.910759
min	0.000000	400.000000	1.000000e+00	1992.000000	1.000000	0.000000	2.000000	2.000000	1.000000
25%	1053.250000	2490.000000	5.698400e+04	2008.000000	2.000000	1.300000	3.000000	5.000000	4.000000
50%	2273.500000	3999.500000	8.000000e+04	2012.000000	3.000000	1.600000	5.000000	5.000000	5.000000
75%	3598.000000	7999.000000	1.032000e+05	2016.000000	4.000000	1.900000	5.000000	5.000000	6.000000
max	4727.000000	33900.000000	1.110100e+06	2023.000000	9.000000	6.300000	5.000000	7.000000	6.000000

```
[82] df.info()

<class 'pandas.core.frame.DataFrame'>
Int64Index: 3598 entries, 0 to 3684
Data columns (total 14 columns):
 #   Column             Non-Null Count  Dtype  
---  ------             --------------  -----  
 0   Unnamed: 0         3598 non-null   int64  
 1   title              3598 non-null   object 
 2   Price              3598 non-null   int64  
 3   Mileage(miles)     3598 non-null   int64  
 4   Registration_Year  3598 non-null   int64  
 5   Previous Owners    2244 non-null   float64
 6   Fuel type          3598 non-null   object 
 7   Body type          3598 non-null   object 
 8   Engine             3598 non-null   float64
 9   Gearbox            3598 non-null   object 
 10  Doors              3598 non-null   float64
 11  Seats              3592 non-null   float64
 12  Emission Class     3598 non-null   int64  
 13  Service history    534 non-null    object 
dtypes: float64(4), int64(5), object(5)
memory usage: 421.6+ KB
```

Figure 4: Overview of the dataset's statistics and structure.

During the initial EDA, it is a key factor to identify any missing values in the dataset as missing values can significantly impact on the quality of the analysis. By utilizing the .isnull().sum() function, it can easily identify all the null value in the dataset, below figure 5 shows that there are 1354 of null values in the variable “Previous Owner”, 6 null values in the variable “Seats”, and 3064 null values in the variable “Service History”.



```
[118] df.isnull()
```

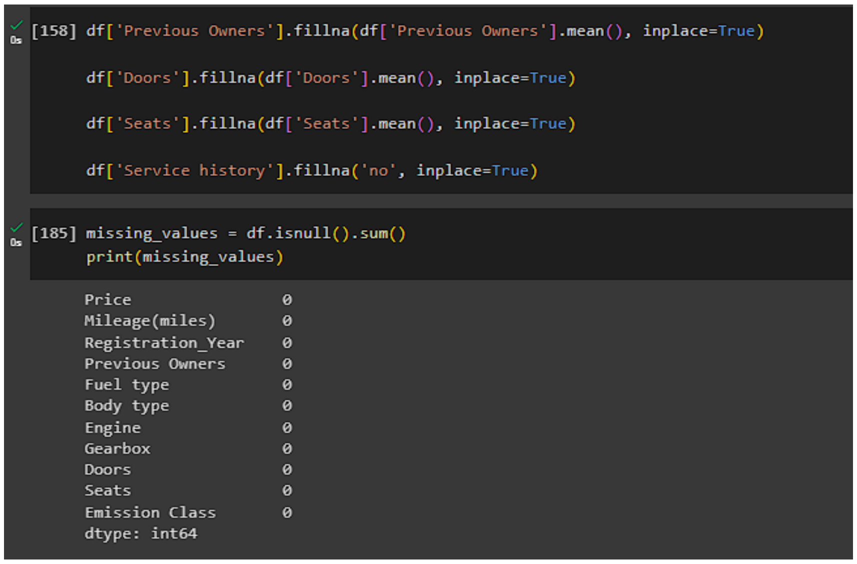
	Unnamed: 0	title	Price	Mileage(miles)	Registration_Year	Previous Owners	Fuel type	Body type	Engine	Gearbox	Doors	Seats	Emission Class	Service history
0	False	False	False	False	False	False	False	False	False	False	False	False	False	True
1	False	False	False	False	False	False	False	False	False	False	False	False	False	False
2	False	False	False	False	False	True	False	False	False	False	False	False	False	True
3	False	False	False	False	False	False	False	False	False	False	False	False	False	False
4	False	False	False	False	False	True	False	False	False	False	False	False	False	True
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
3680	False	False	False	False	False	False	False	False	False	False	False	False	False	True
3681	False	False	False	False	False	True	False	False	False	False	False	False	False	True
3682	False	False	False	False	False	True	False	False	False	False	False	False	False	True
3683	False	False	False	False	False	True	False	False	False	False	False	False	False	False
3684	False	False	False	False	False	False	False	False	False	False	False	False	False	True

3598 rows × 14 columns

```
[119] missing_values = df.isnull().sum()
      print(missing_values)

Unnamed: 0            0
title                 0
Price                 0
Mileage(miles)        0
Registration_Year     0
Previous Owners    1354
Fuel type             0
Body type             0
Engine                0
Gearbox               0
Doors                 0
Seats                 6
Emission Class        0
Service history    3064
dtype: int64
```

Figure 5: Null values in the dataset.

With such large amounts of missing value, it is important to address the issue immediately before proceeding into machine learning modelling. Data handling techniques that I’ve utilized is replace the null values in the data with the mean value of the variable’s data. As indicated, there are missing values in these variables (Previous Owner, Seats, and Service history), the mean value of “previous owner” and “seats” are filled into the null value, while null value in service history data indicates to no, therefore null value in service history will be filled up as “no”. No data were removed during this process, as it is important to keep the originality of the data to avoid loss of data, bias, and inaccurate statistical analysis. After handling missing value, it has ensured that there are no null values in the dataset anymore by using is.null().sum() function again in figure 6.



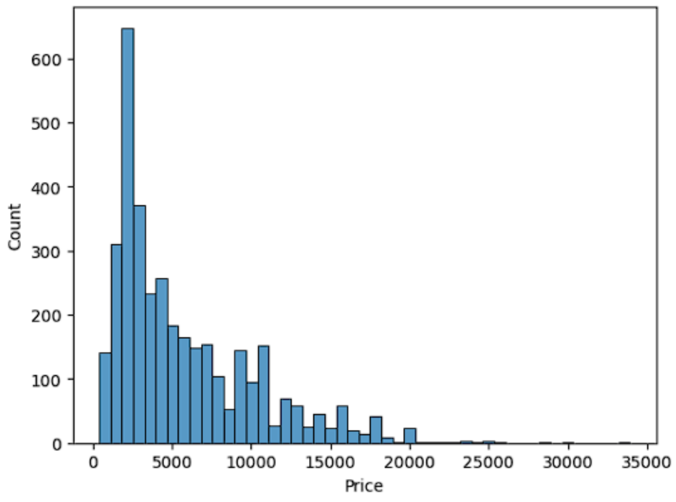
```
[158] df['Previous Owners'].fillna(df['Previous Owners'].mean(), inplace=True)

      df['Doors'].fillna(df['Doors'].mean(), inplace=True)

      df['Seats'].fillna(df['Seats'].mean(), inplace=True)

      df['Service history'].fillna('no', inplace=True)
```

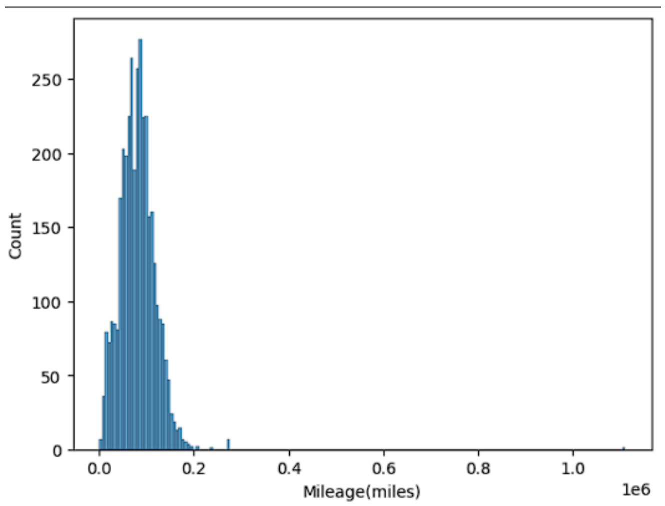
```
[185] missing_values = df.isnull().sum()
      print(missing_values)
```

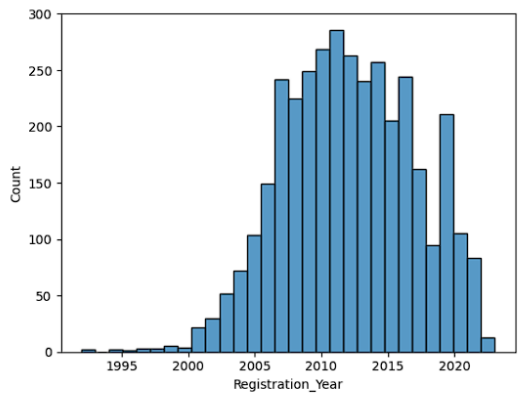
```
Price                0
Mileage(miles)       0
Registration_Year    0
Previous Owners      0
Fuel type            0
Body type            0
Engine               0
Gearbox              0
Doors                0
Seats                0
Emission Class       0
dtype: int64
```

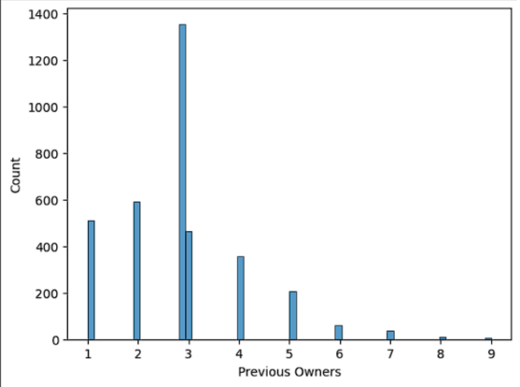
Figure 6: Data handling.

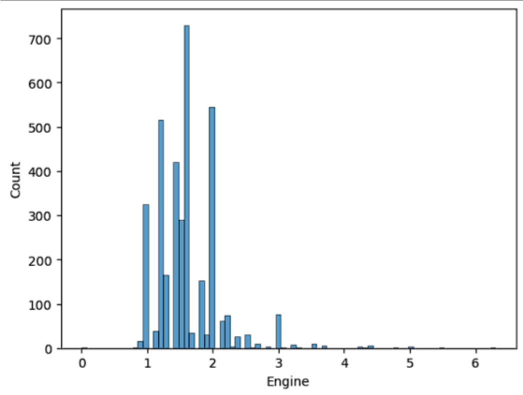
Below are so of the basic data visualization regarding used car features, the following data are visualized using "sns.histplot" function from Seaborn which enable of creating histogram plot easily. These data visualizations provide us a better understanding of the overall data.

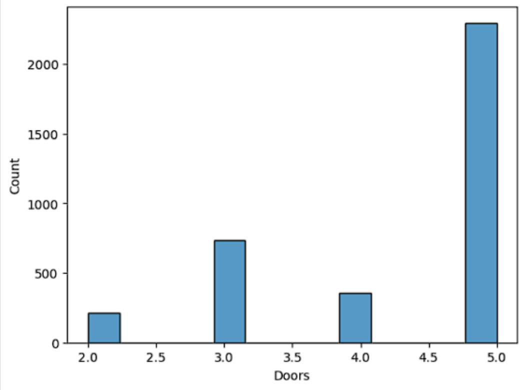


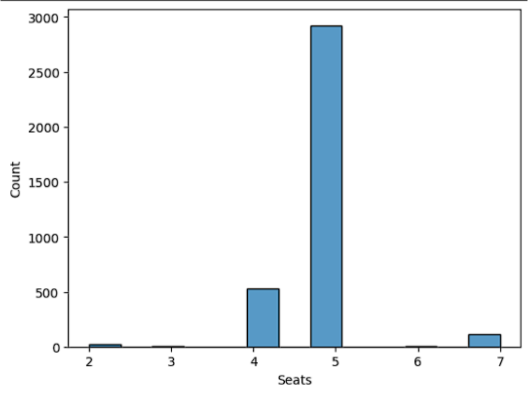


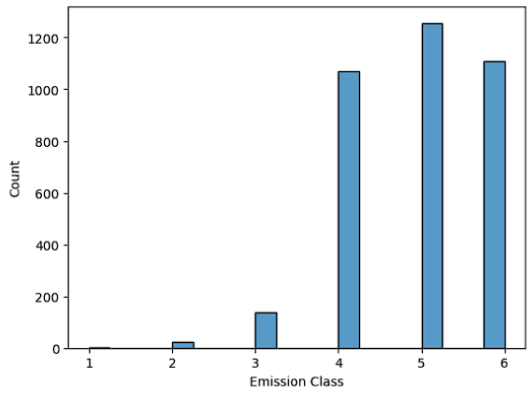

Count
Registration_Year

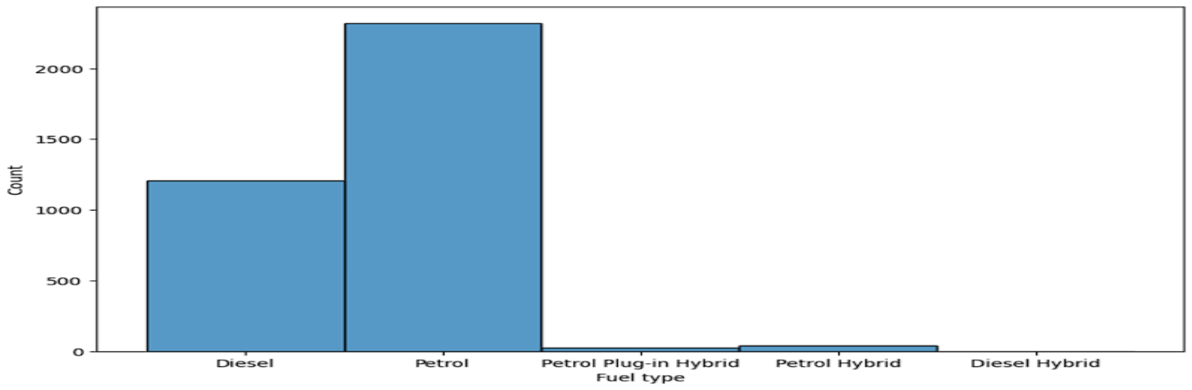

Count
Previous Owners


Count
Engine


Count
Doors


Count
Seats


Count
Emission Class

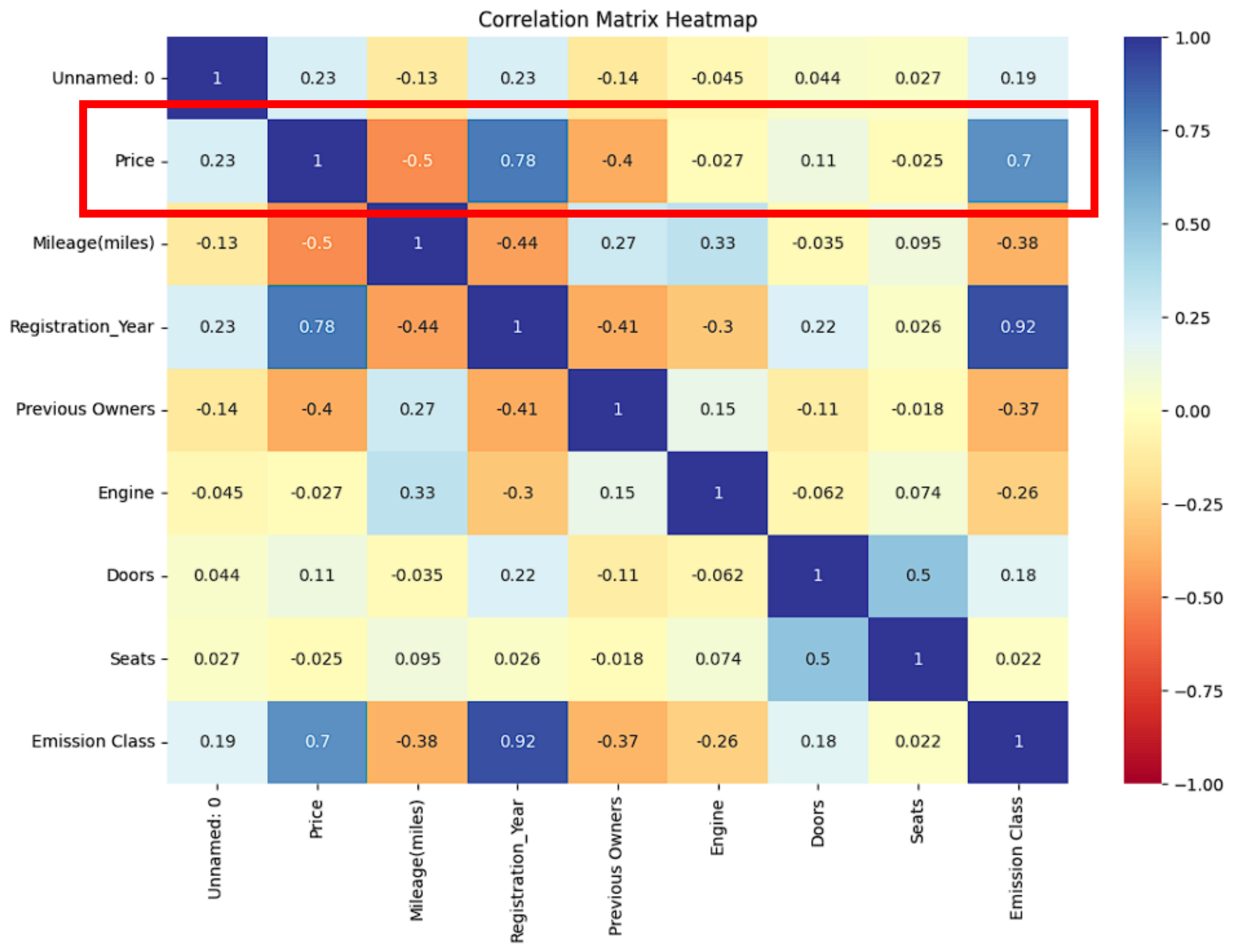

Count
Diesel
Petrol
Petrol Plug-in Hybrid
Petrol Hybrid
Diesel Hybrid
Fuel type

- The price range between 0 to 5000 has the most total count while it decreased to 20000.
- Mileage ranges between 0.0 to 0.2.
- The registration year ranges between the year 1993 to 2023 and 1996 has the lowest count at 1 while 2011 has the highest count at 286 among all the years.
- Previous owners ranges between 1 to 9, where the value of 9 has the lowest count and the value of 3 has the highest count.
- Engine ranges between 0 to7, where most of the engine is within the value of 1 to 2, and the value of 1.6L has the highest count at 730 while the value of 6.3L has the lowest count at 1 only.
- The number of doors ranging from 2 to 5, the value of 5 has the highest count at 2295 while the value of 2 has the lowest count at 211.
- The number of seats ranging from 7 to 2, the value of 5 has the highest count at 2917 while the value of 3 has the highest count at 2.
- The emission class ranges between 1 to 6, the value of 5 has the highest count at 1256 while the value of 1 has the highest count at 4.
- The fuel type has a total of 5 types including Diesel, Petrol, Petrol Plug-in Hybrid Fuel Type, Petrol Hybrid and Diesel Hybrid. Petrol has the highest count at 2322 while Diesel Hybrid has the lowest count at 1 only.

4.2 Descriptive EDA

In the following section of descriptive EDA is a continuous of the initial EDA, however it provides a deeper and more comprehensive analysis of the dataset. Such as the correlation between variables and the pattern of the data. Moreover, it includes of data encoding process by converting categorical variables into numerical format before any further machine learning modelling.

Below figure 7 shows the correlation matrix heatmap of numerical variables (ID, Price, Mileage(miles), Registration Year, Previous Owner, Engine, Doors, Seats, and Emission Class) while categorical variables are not encoded. In this research, the target variable of the dataset is price while the other variables are the features. Therefore, we will first focus the highlighted red box in the matrix heatmap to analyse which possible features has the highest correlation with the target variable (Price) while which features has a significant low correlation that may indicates irrelevant to the target variable.

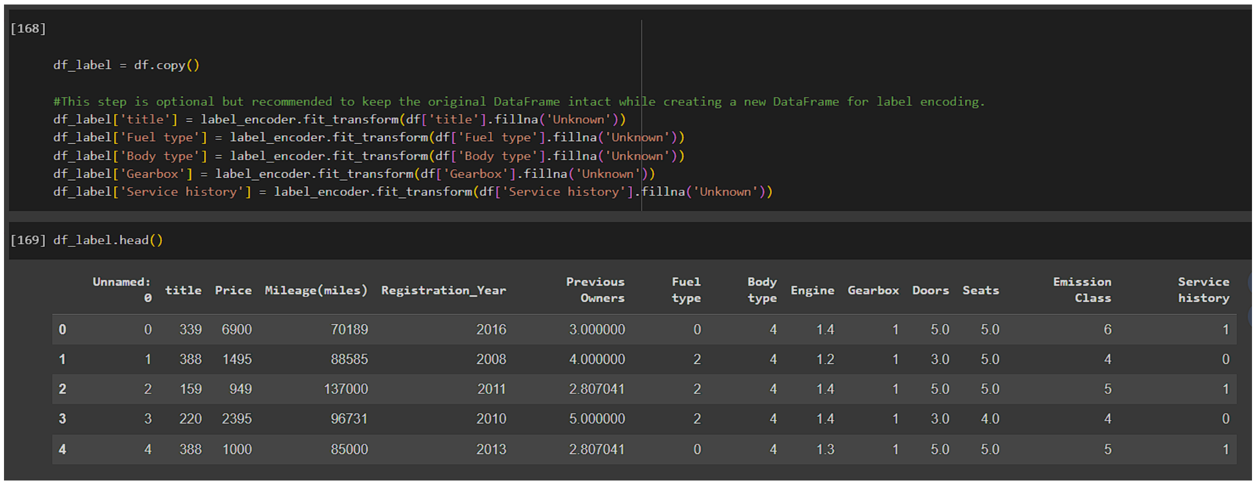


	Unnamed: 0	Price	Mileage(miles)	Registration_Year	Previous Owners	Engine	Doors	Seats	Emission Class
Unnamed: 0	1	0.23	-0.13	0.23	-0.14	-0.045	0.044	0.027	0.19
Price	0.23	1	-0.5	0.78	-0.4	-0.027	0.11	-0.025	0.7
Mileage(miles)	-0.13	-0.5	1	-0.44	0.27	0.33	-0.035	0.095	-0.38
Registration_Year	0.23	0.78	-0.44	1	-0.41	-0.3	0.22	0.026	0.92
Previous Owners	-0.14	-0.4	0.27	-0.41	1	0.15	-0.11	-0.018	-0.37
Engine	-0.045	-0.027	0.33	-0.3	0.15	1	-0.062	0.074	-0.26
Doors	0.044	0.11	-0.035	0.22	-0.11	-0.062	1	0.5	0.18
Seats	0.027	-0.025	0.095	0.026	-0.018	0.074	0.5	1	0.022
Emission Class	0.19	0.7	-0.38	0.92	-0.37	-0.26	0.18	0.022	1

Figure 7: Correlation Matrix Heatmap (Before Encoding)

From the above figure 7, it shows that variable "Registration Year" has the highest correlation with the target variable "Price" at 0.78 which indicates a positive relationship. Example of positive relationship such as where a higher registration year will have a higher price. While the second highest positive correlation with the target variable "Price" is variable "Emission Class" at 0.7. Other than that, the variable "Mileage(miles)" has a high correlation with the target variable as well at -0.5, the negative correlation value indicates a negative relationship. For example, a higher mileage (miles) will contribute to a lower price value. Additionally, variable "Previous Owners" has a correlation of -0.4 with target variable too.

As mentioned above, the correlation matrix heatmap in figure 7 only include with numerical variables before encoding the categorical variables. Therefore, in this research I've has utilized label encoding method to convert categorical variables into numerical format shown in figure 8, which then will be used to output the correlation of the target variable "Price". In figure 9, it shows the correlation matrix heatmap of all the variables in the dataset after encoding which provides a greater detail.



```
[168]

    df_label = df.copy()

    #This step is optional but recommended to keep the original DataFrame intact while creating a new DataFrame for label encoding.
    df_label['title'] = label_encoder.fit_transform(df['title'].fillna('Unknown'))
    df_label['Fuel type'] = label_encoder.fit_transform(df['Fuel type'].fillna('Unknown'))
    df_label['Body type'] = label_encoder.fit_transform(df['Body type'].fillna('Unknown'))
    df_label['Gearbox'] = label_encoder.fit_transform(df['Gearbox'].fillna('Unknown'))
    df_label['Service history'] = label_encoder.fit_transform(df['Service history'].fillna('Unknown'))
```

```
[169] df_label.head()

     Unnamed:  title  Price  Mileage(miles)  Registration_Year  Previous  Fuel  Body  Engine  Gearbox  Doors  Seats  Emission  Service
            0                                                      Owners  type  type                                  Class   history
  0         0    339   6900           70189               2016  3.000000     0     4     1.4        1    5.0    5.0         6        1
  1         1    388   1495           88585               2008  4.000000     2     4     1.2        1    3.0    5.0         4        0
  2         2    159    949          137000               2011  2.807041     2     4     1.4        1    5.0    5.0         5        1
  3         3    220   2395           96731               2010  5.000000     2     4     1.4        1    3.0    4.0         4        0
  4         4    388   1000           85000               2013  2.807041     0     4     1.3        1    5.0    5.0         5        1
```

Figure 8: Label encoded categorical variables.

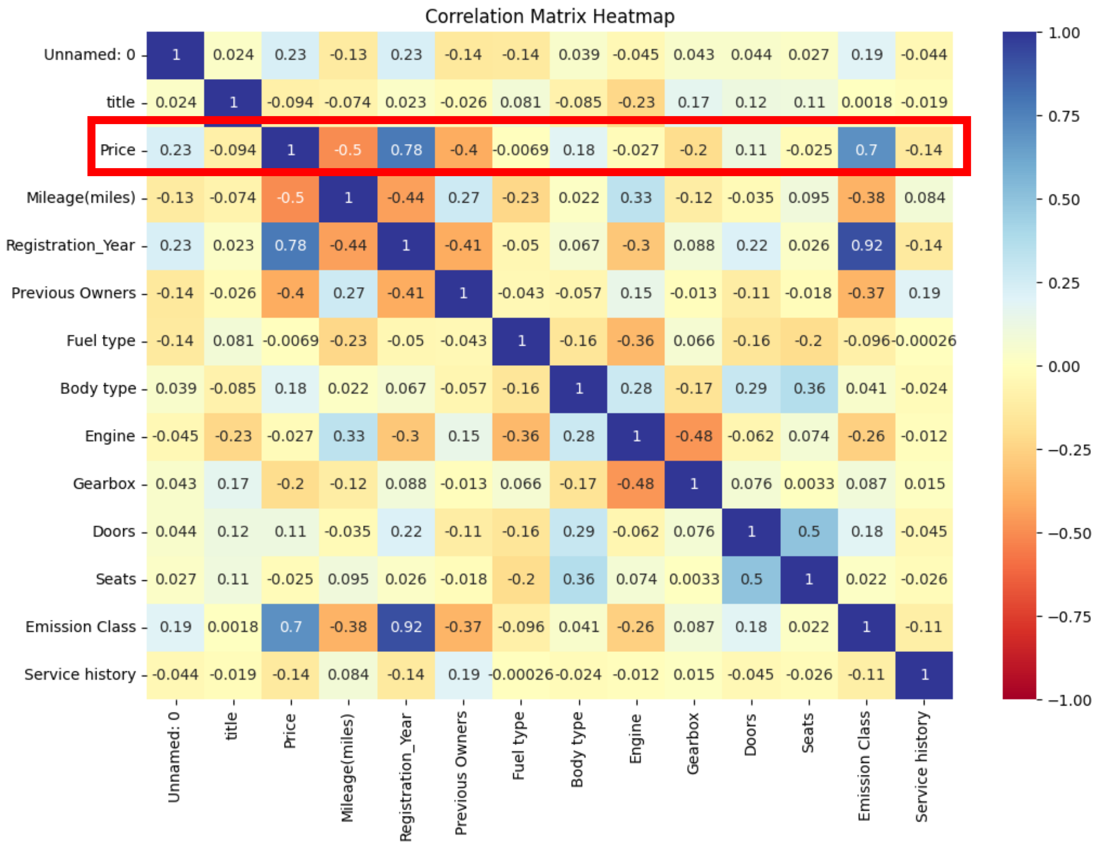


Figure 9: Correlation Matrix Heatmap (After Encoding)

As the additional of encoded categorical variables (title, Fuel type, Body type, Gearbox, and Service history) into the correlation matrix heatmap, it is noticeable that numerical variable remained the highest correlation with the target variable while encoded categorical variables has significant low correlation. Where variable "title" only has correlation at -0.094, "Fuel type" at -0.0069, "Body type" at 0.18, "Engine" at -0.2, and "Service history" at -0.14.

After analysing the correlation matrix heatmap in both figure 7 and figure 9, it can be concluded the 4 variables that have the highest correlation with the target variable "Price" are "Registration Year", "Previous Owner", "Emission Class", "Mileage(miles)". Data visualization of relationship between the 4 variables and the target variable "Price" is shown below for better understanding:

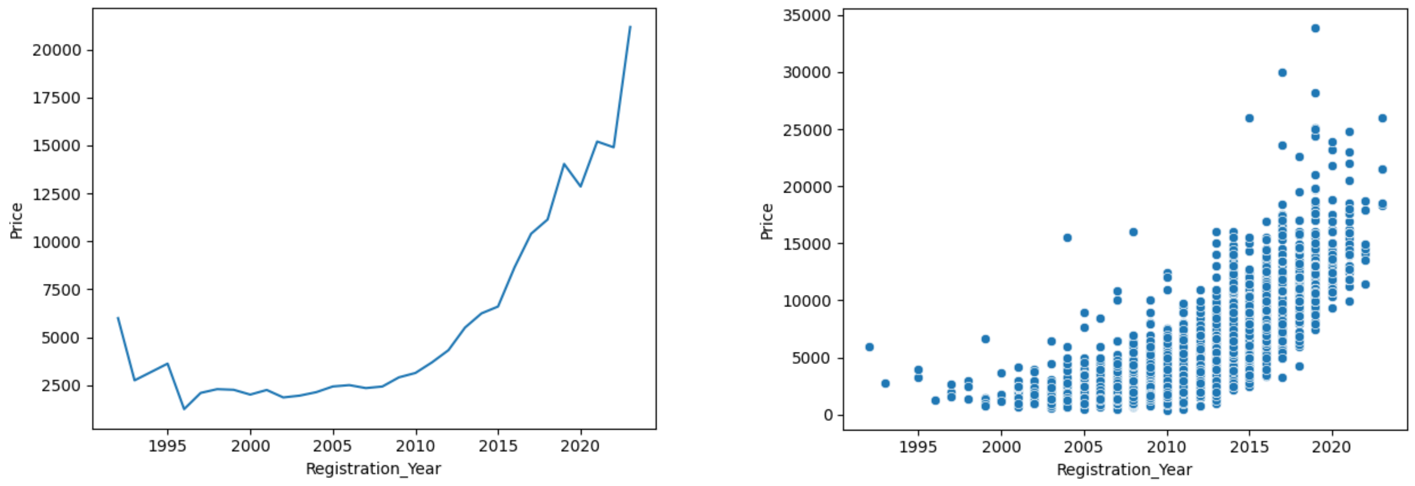


Figure 10: Line/Scatter plot relationship between Price and Registration Year.

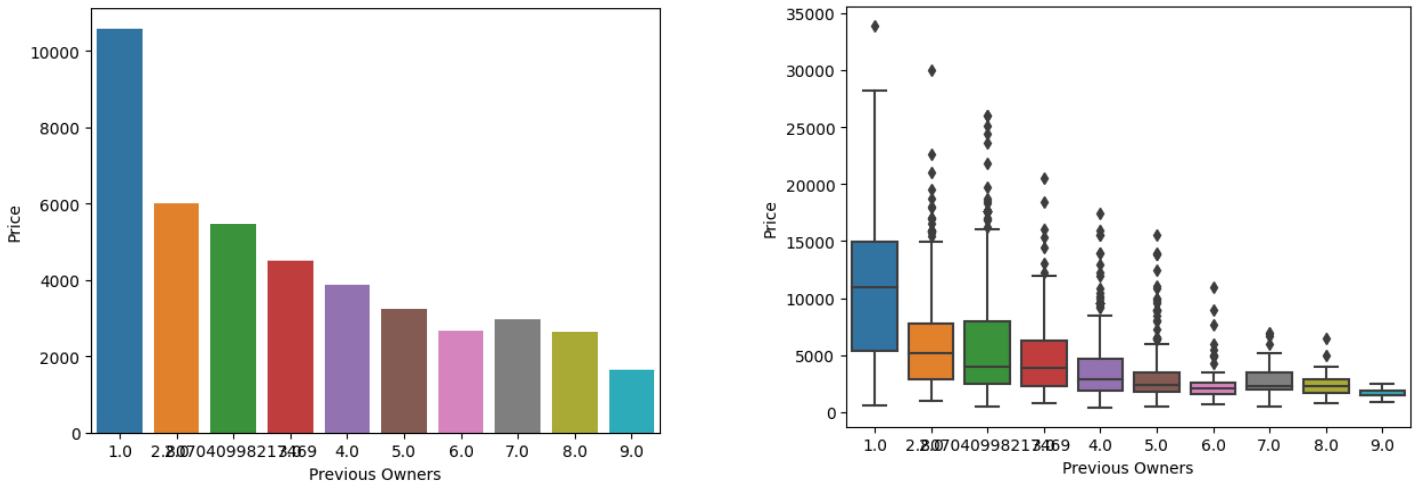


Figure 11: Bar Chart/Box plot relationship between Price and Previous Owners.

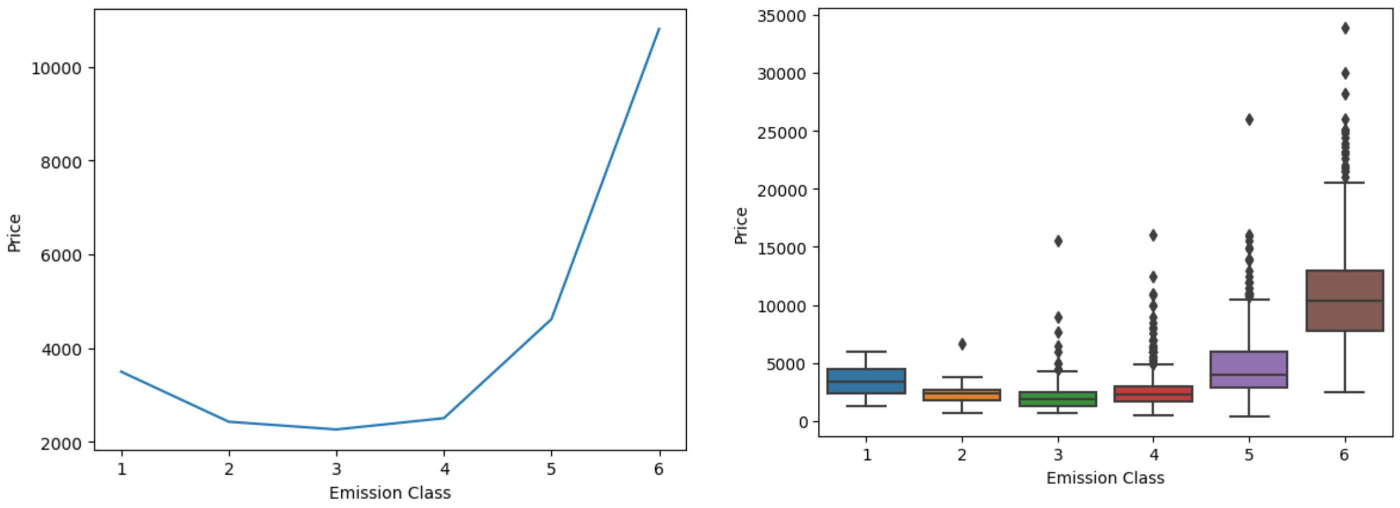


Figure 12: Line/Box plot relationship between Price and Emission Class.

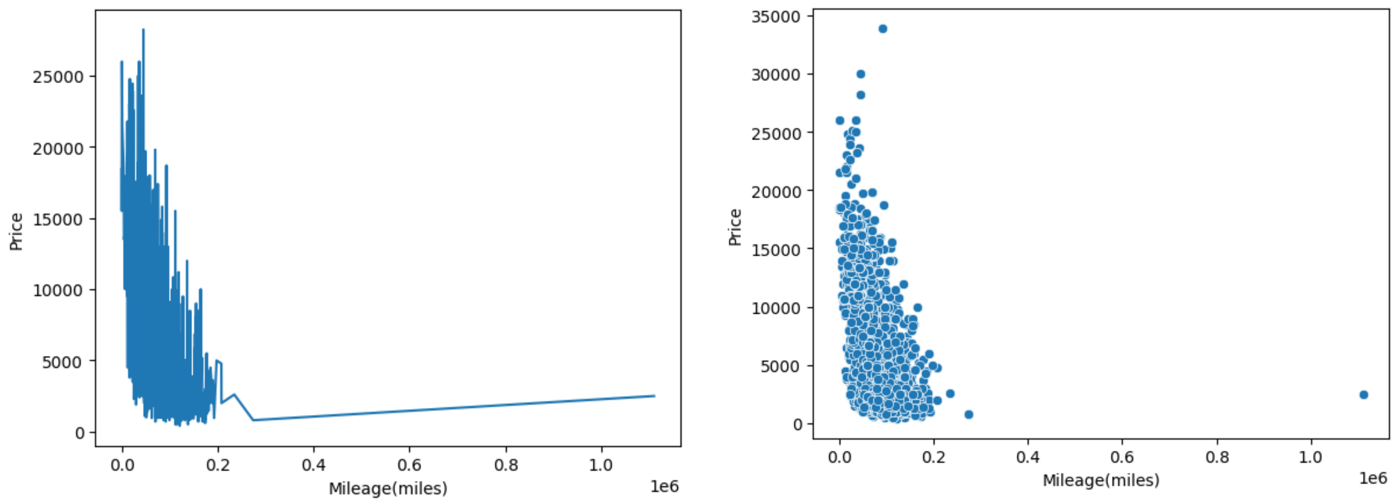


Figure 13: Line plot relationship between Price and Mileage(miles).

From figure 10 and figure 12, it shown that both independent variables “Registration Year” and “Emission Class” have a positive relationship with the target variable “Price” because the line plot shows an upward trajectory which states that a higher value of the independent variable will contributes to a higher value of used car’s price. On the other hand, figure 11 and figure 13 shown both independent variables “Previous Owner” and “Mileage(miles)” have a negative relationship with the target variable “Price” because both line plot and bar chart plot have a downward trajectory. Therefore, a lower value of the independent variable such as “Previous Owners” and “Mileage(miles)” will contributes to a higher value of used car’s price.

4.3 Modelling

4.3.1 Linear Regression Model

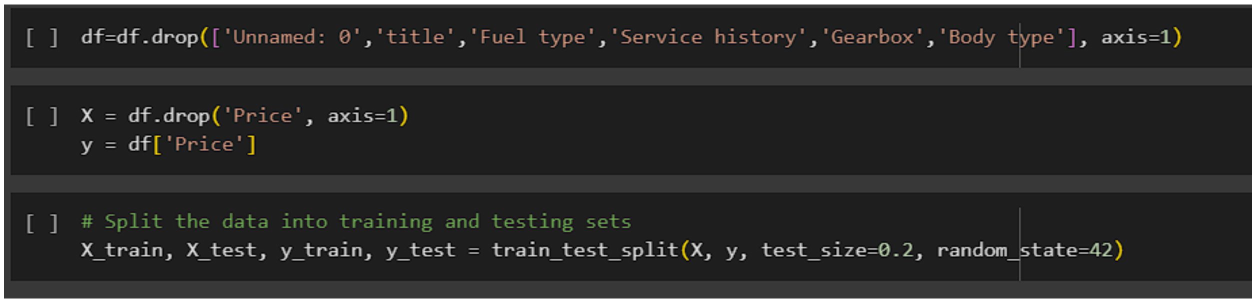
Linear regression, a supervised machine learning method, is adept at finding the best fit between a dependent variable and one or more independent variables that exhibit a linear relationship [14]. The linear regression equation takes the form of Y = b0 + b1 * X1 + b2 * X2 + ... + bn * Xn, where Y is the dependent variable, X1, X2, ... Xn are the independent variables, and b0, b1, b2, ... bn represent the coefficients in this n-dimensional equation.

4.3.1.1 Experiment 1a (LR Model)

The first experiment of linear regression model is conducted without label encoding of the categorical variables, exclusively with numerical variables only and all the categorical variables were dropped from the dataset. The reason of sidestepping label encoding during this experiment is to preserve the original data structure, therefore, while conducting experiment 1 it is required to carefully separate both numerical variables and categorical variables.

With this experiment 1a setup, it provides us a deep understanding of used car's price value predictions by utilizing the inherent numerical variables of the dataset such as "mileage(miles)", "registration year", "previous owner", etc. Moreover, this innovative approach will not only clarify the underlying links between the numerical variables and the target variable "Price", but also presents how useful linear regression can be in situation where categorical variables are not available.

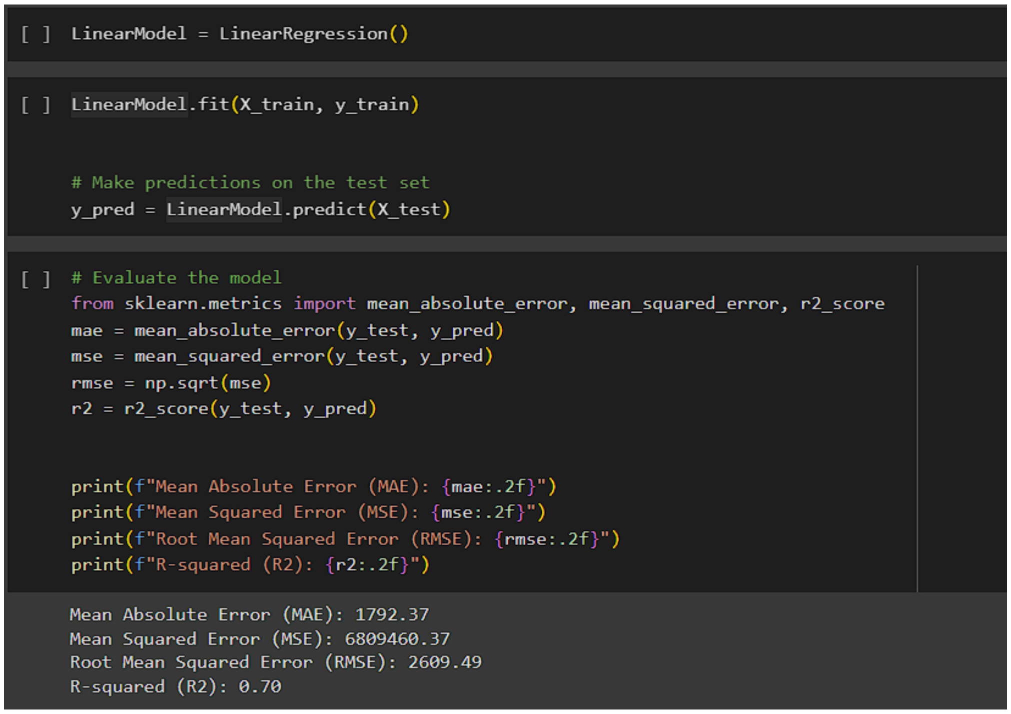
The main goal of this experiment 1 aimed to uncover useful insights regarding the relationship between the numerical variables in the dataset with the target variables "Price". Additionally, the analyse the effectiveness of linear regression model in prediction of used car's price value.



```
[ ] df=df.drop(['Unnamed: 0','title','Fuel type','Service history','Gearbox','Body type'], axis=1)

[ ] X = df.drop('Price', axis=1)
    y = df['Price']

[ ] # Split the data into training and testing sets
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

Figure 14: Experiment 1a Train Test Split.

In figure 14, the first line of code shown utilizing the .drop() function to drop all the categorical variables and irrelevant variable from the dataset such as (ID, title, Fuel type, Service history, Gearbox, and Body type) while numerical variables are kept into the dataset "df". Then independent variables and dependent variables were separated into X and y where X contained all the independent variables while y contained the dependent variable "Price". After that, the dataset was split into training and testing sets which in this experiment was set to the test size of 0.2, 80% of the data will be used for training while the remaining 20% will be used for testing and the random state was fixed at 42.



```
[ ] LinearModel = LinearRegression()
```

```
[ ] LinearModel.fit(X_train, y_train)


    # Make predictions on the test set
    y_pred = LinearModel.predict(X_test)
```

```
[ ] # Evaluate the model
    from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score
    mae = mean_absolute_error(y_test, y_pred)
    mse = mean_squared_error(y_test, y_pred)
    rmse = np.sqrt(mse)
    r2 = r2_score(y_test, y_pred)


    print(f"Mean Absolute Error (MAE): {mae:.2f}")
    print(f"Mean Squared Error (MSE): {mse:.2f}")
    print(f"Root Mean Squared Error (RMSE): {rmse:.2f}")
    print(f"R-squared (R2): {r2:.2f}")
```

```
Mean Absolute Error (MAE): 1792.37
Mean Squared Error (MSE): 6809460.37
Root Mean Squared Error (RMSE): 2609.49
R-squared (R2): 0.70
```

Figure 15: Experiment 1a Model and Performance Result.

Following onto figure 15, it has trained a linear regression model onto the training data, then the .fit() function was utilized to take the training feature data "X_train" and the corresponding target variable "y_train". During this process, the linear regression model learns the coefficients for each feature to make predictions. After the process is done, the trained linear regression model will make predictions on the test data and the predicted values of the target variable will be stored into "y_pred".

Following are the performance metrics from the experiment 1a linear regression model, this metrics can be analysed and compared with other experiments in the following section.

- Mean Absolute Error (MAE): 1792.37
- Mean Squared Error (MSE): 6809460.37
- Root Mean Squared Error (RMSE): 2609.49
- R Squared (R2): 0.70

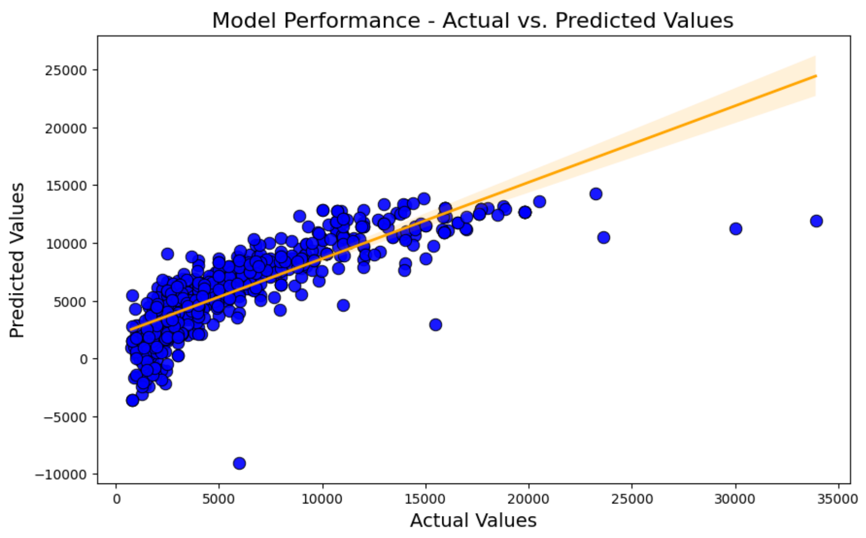


Figure 16: Scatterplot of 1a model's prediction

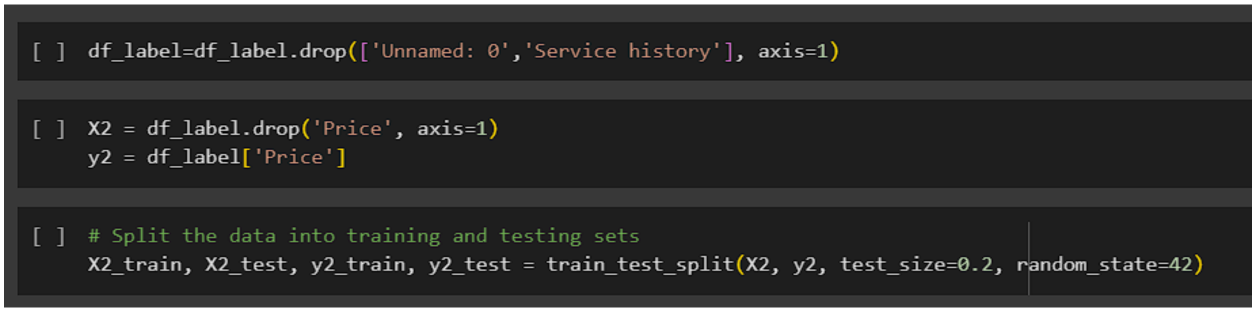
Above figure 16, shown a scatterplot and a linear regression line to visualize the performance of the linear regression model in this experiment 1a. The figure 16 provides us a better understanding of how well the linear regression model's prediction align with the actual values. As you can see all the data points were clustered around the linear regression line which indicates a good fit between the model's prediction and the actual data.

4.3.1.2 Experiment 1b (LR Model)

The second experiment of linear regression model is conducted with the process of label encoding which involved converting categorical variables into numerical format. Unlike the first experiment 1a, this experiment 1b contains both numerical and categorical variables into the machine learning predictions. The reason of label encoding the categorical variables in this experiment is to implement a wider range of features into the machine learning model which allow the model to identify the pattern of the data that was previously qualitative.

With this experiment 1b setup, it provides us the understanding of different perspectives with and without the process of label encoding. Thereby, both performance result from experiment 1a and

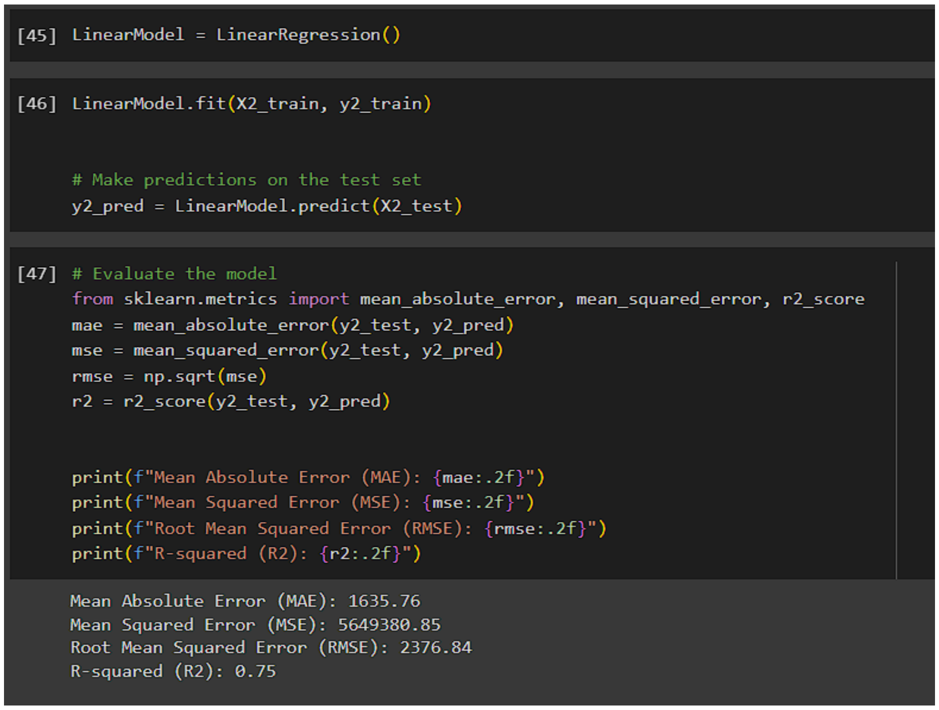
1b can be compared to analyse if the used car's price prediction has improved after introducing label encoding into the experiment. The main goal of this experiment 1b aimed to discover useful insights regarding the relationship between all the relevant variables in term of numerical and categorical with the target variable 'Price".



```
[ ] df_label=df_label.drop(['Unnamed: 0','Service history'], axis=1)

[ ] X2 = df_label.drop('Price', axis=1)
    y2 = df_label['Price']

[ ] # Split the data into training and testing sets
    X2_train, X2_test, y2_train, y2_test = train_test_split(X2, y2, test_size=0.2, random_state=42)
```

Figure 17: Experiment 1b Train test Split.

Similar with the previous experiment 1a, the data has been split into training and testing sets at the size of 80/20 and the random state of 42. However, it has utilized the label encoded dataset called "df_label" which was encoded during the data preprocessing process in figure 8 instead of using the original dataset called "df". Moreover, irrelevant variable was dropped from the label encoded dataset such as the "ID".



```
[45] LinearModel = LinearRegression()

[46] LinearModel.fit(X2_train, y2_train)


     # Make predictions on the test set
     y2_pred = LinearModel.predict(X2_test)

[47] # Evaluate the model
     from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score
     mae = mean_absolute_error(y2_test, y2_pred)
     mse = mean_squared_error(y2_test, y2_pred)
     rmse = np.sqrt(mse)
     r2 = r2_score(y2_test, y2_pred)


     print(f"Mean Absolute Error (MAE): {mae:.2f}")
     print(f"Mean Squared Error (MSE): {mse:.2f}")
     print(f"Root Mean Squared Error (RMSE): {rmse:.2f}")
     print(f"R-squared (R2): {r2:.2f}")

     Mean Absolute Error (MAE): 1635.76
     Mean Squared Error (MSE): 5649380.85
     Root Mean Squared Error (RMSE): 2376.84
     R-squared (R2): 0.75
```

Figure 18: Experiment 1b Model and Performance Result.

The same machine learning modelling process from the previous experiment 1a, however it output a different performance result compared to experiment 1a due to additional encoded categorical variables were implemented into the machine learning process. The following are the performance metrics result from this experiment 1b.

- Mean Absolute Error (MAE): 1635.76
- Mean Squared Error (MSE): 5649380.85
- Root Mean Squared Error (RMSE): 2376.84
- R Squared (R2): 0.75

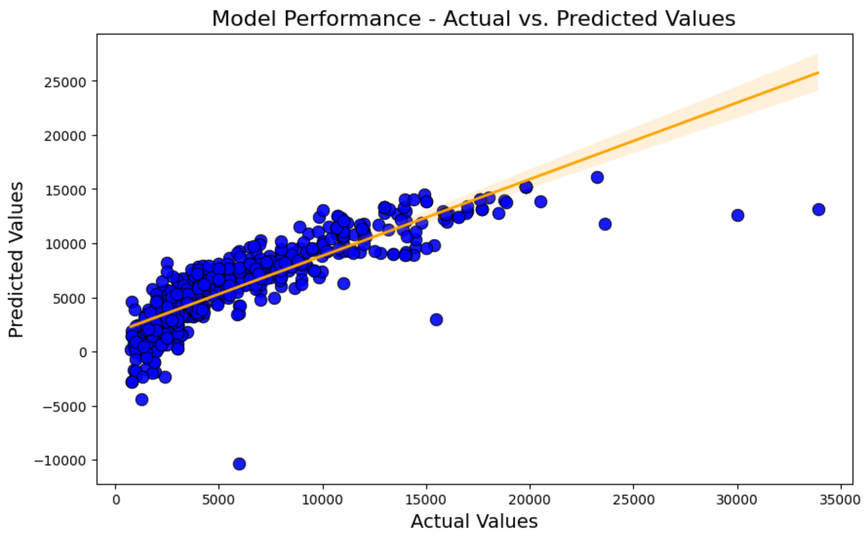


Figure 19: Scatterplot of 1b model's prediction

Referring in both figure 19 and figure 16, the regression line in figure 19 1b model's prediction is slightly closer to the data points compared to the previous regression line in figure 16 experiment 1a model's prediction. This may suggest that experiment 1b has a slight better prediction performance compared to experiment 1a linear regression model.

4.3.1.3 Summary of linear regression model

In summary of these two experiments (1a and 1b), both were conducted utilizing linear regression model to perform prediction in used car price value which both experiments were then evaluated based on their prediction performance. The main difference between the experiments lies in the treatment of encoding categorical variables, where in experiment 1a only implemented numerical variables into the model, while experiment 1b involved label encoding categorical variables to incorporate them into the model.

Experiment 1a, during the data preprocessing phase, categorical variables were dropped from the dataset before incorporating the remaining numerical variables into the linear regression model. Moreover, the dataset was split into 80% training and 20% testing sets. The performance metrics outputted from experiment 1a are:

- MAE: 1792.37
- MSE: 6809460.37
- RMSE: 2609.49
- R2: 0.70

Observation of this experiment 1a through analysing the above performance metrics shows that the model demonstrated a relatively high MAE which indicated a significant prediction error. However, the R2 score of 0.70 indicates that approximately 70% of the variation in the target variable was explained.

Experiment 1b, similar data preprocessing phase as experiment 1a however categorical variables have been label encoded and all the relevant variables have been incorporated into the linear regression model. The performance metrics outputted from experiment 1b are:

- MAE: 1635.76
- MSE: 5649380.85
- RMSE: 2376.84
- R2: 0.75

Observation of this experiment 1b through analysing the above performance metrics shows that the MAE remained relatively high however with a small percentage of decrease from 1792.37 to 1635.76. The implementation of label encoding resulted in a slightly improved R2 score at 0.75.

Other than that, the scatter plot from this experiment 1b resulted a closer fit to the data points compared to the previous experiment 1a.

In comparison of these 2 experiments, it indicated that experiment 1b with the implementation of label encoded categorical variables showed a slightly better prediction performance compared to experiment 1a. Moreover, it showed a slight lower error as well compared to experiment 1a. In conclusion, label encoding in experiment 1b enabled to improve the model capability to identify the pattern of categorical variables which leads to a better prediction performance of used car price at a R2 score of 0.75.

4.3.2 Polynomial Regression Model

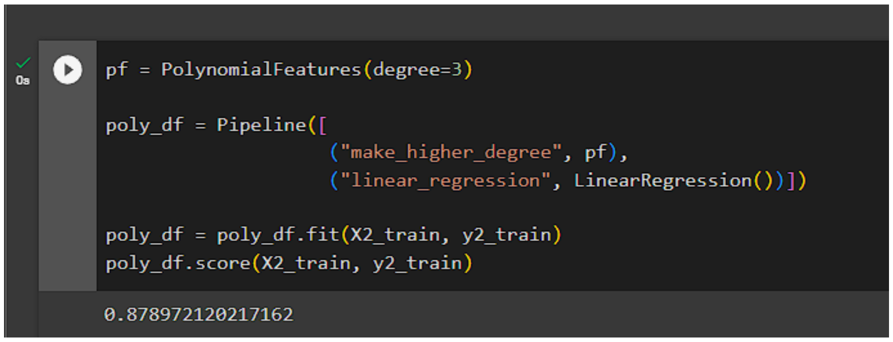
Polynomial regression is extremely useful when the data points in the dataset is not in a straight line as the main goal of polynomial regression is to identify the curve fit to the data and make accurate predictions [16]. Polynomial regression is similar to linear regression, but it captures a non-linear relationship between the dependent variable and independent variables instead of linear relationship. The equation of polynomial regression is $y\beta$ (x) = $\beta 0 + \beta 1x + \beta 2x$ 2.

4.3.2.1 Experiment 2a (PR Model)

The first experiment 2a of predicting used car's price value was conducted with the use of polynomial regression model and label encoded dataset which are X2 and y2. The reason of utilizing label encoded dataset rather than the original dataset in this first experiment was due to the findings from the previous experiment 1b as it resulted that label encoded variables able to improve the model capability to identify the patterns and the prediction performance metrics.

With this experiment 2a setup, it is capable to inherent complexities of relationships between the independent variables and the dependent variable 'Price". Unlike linear regression model from the previous experiment, implementation of polynomial regression model in this experiment 2a is capable of identifying non-linear patterns within the dataset. As polynomial regression offered a more flexible approach, the objective of this experiment is to capture the nuanced relationship, discover hidden and more complex insights, most importantly is to improve the performance of predicting used car's price value.

The code of experiment 2a shown in figure 20, it has utilized Scikit-learn to create a polynomial regression model. Firstly, it has employed a pipeline called "poly_df" that consists of 2 components including the polynomial feature class with a degree of 3 and the linear regression model to fit itself into the transformed features. Secondly, the pipeline “poly_df” was trained on the provided dataset “X2_train” and “y2_train” that was split from the previous experiment 1b. After the modelling process was completed, the R2 score was outputted at a significantly high accuracy at 0.8789.

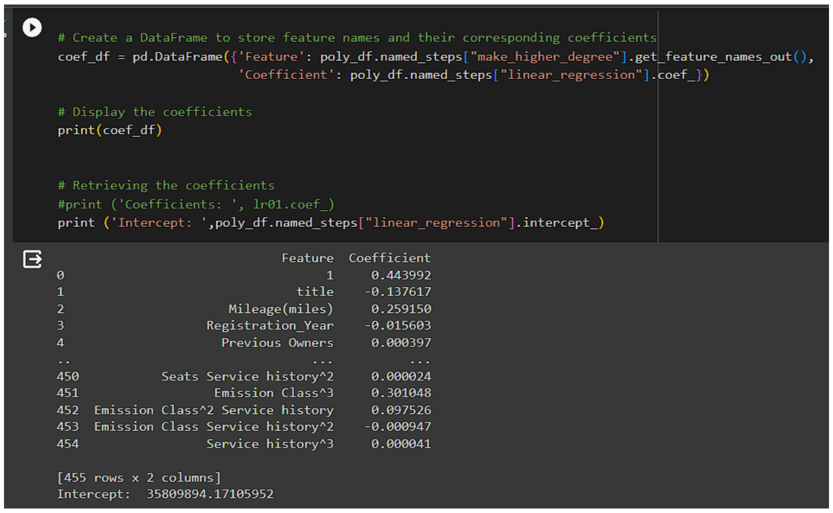


```
pf = PolynomialFeatures(degree=3)

poly_df = Pipeline([
                    ("make_higher_degree", pf),
                    ("linear_regression", LinearRegression())])

poly_df = poly_df.fit(X2_train, y2_train)
poly_df.score(X2_train, y2_train)
```

```
0.878972120217162
```

Figure 20: Experiment 2a Model and R2 Score

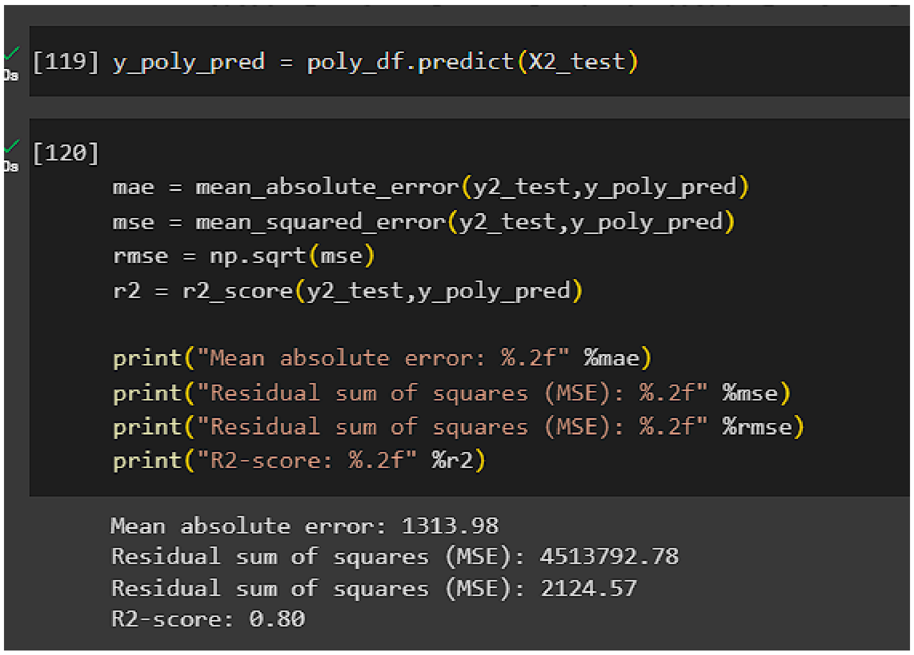
Below output from figure 21 displayed the coefficients of the features in this polynomial regression model experiment 2a, where the “feature” column represents the names of the features that were implemented into the polynomial regression model. On top of that, as the polynomial feature was set at a degree of 3, it did not include the original features from the dataset but also its interaction and a higher degree term. On the other hand, the “coefficient” column represents the corresponding coefficients for each feature, these coefficients indicate the weight of each feature on the predicted outcome. In result, the intercept is approximately 35809894.17 which is the value of the dependent variable “Price” when all other independent variables at the value of zero.



```
# Create a DataFrame to store feature names and their corresponding coefficients
coef_df = pd.DataFrame({'Feature': poly_df.named_steps["make_higher_degree"].get_feature_names_out(),
                        'Coefficient': poly_df.named_steps["linear_regression"].coef_})

# Display the coefficients
print(coef_df)


# Retrieving the coefficients
#print ('Coefficients: ', lr01.coef_)
print ('Intercept: ',poly_df.named_steps["linear_regression"].intercept_)
```

```
                          Feature  Coefficient
0                               1     0.443992
1                           title    -0.137617
2                  Mileage(miles)     0.259150
3               Registration_Year    -0.015603
4                 Previous Owners     0.000397
..                            ...          ...
450         Seats Service history^2     0.000024
451                Emission Class^3     0.301048
452  Emission Class^2 Service history     0.097526
453  Emission Class Service history^2    -0.000947
454               Service history^3     0.000041

[455 rows x 2 columns]
Intercept:  35809894.17105952
```

Figure 21: Coefficient of 2a Model



```
[119] y_poly_pred = poly_df.predict(X2_test)
```

```
[120]
      mae = mean_absolute_error(y2_test,y_poly_pred)
      mse = mean_squared_error(y2_test,y_poly_pred)
      rmse = np.sqrt(mse)
      r2 = r2_score(y2_test,y_poly_pred)

      print("Mean absolute error: %.2f" %mae)
      print("Residual sum of squares (MSE): %.2f" %mse)
      print("Residual sum of squares (MSE): %.2f" %rmse)
      print("R2-score: %.2f" %r2)
```

```
      Mean absolute error: 1313.98
      Residual sum of squares (MSE): 4513792.78
      Residual sum of squares (MSE): 2124.57
      R2-score: 0.80
```

Figure 22: 2a Polynomial Regression Model Performance Result

The performance metrics of polynomial regression model of this experiment 2a is shown in figure 22, the trained polynomial regression model was then used to predict on the testing dataset “X2_test” and “y2_test”. The following are the result of this model’s performance metrics:

- Mean Absolute Error (MAE): 1313.98
- Mean Squared Error (MSE): 4513792.78
- Root Mean Squared Error (RMSE): 2124.57
- R Squared (R2): 0.80

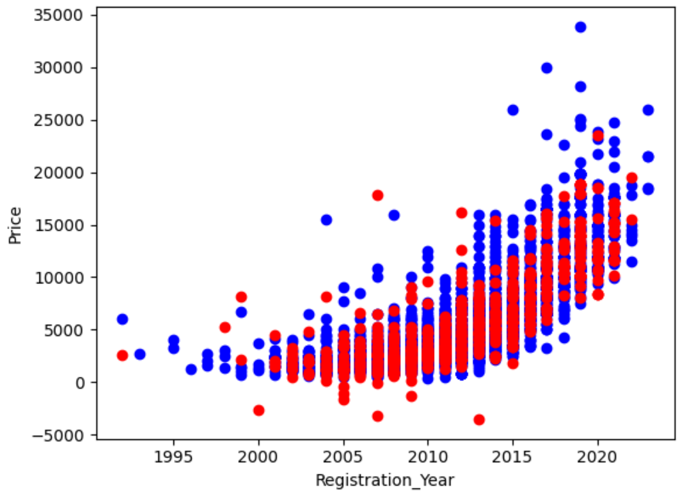


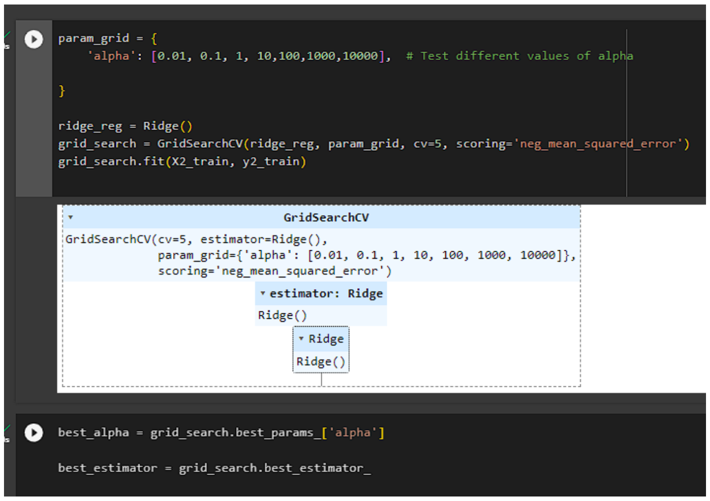
Figure 23: Scatterplot of 2a’s model prediction.

The visualization of the 2a's model prediction is shown in figure 23, as the R2 score of this model is significantly high at 0.80. Therefore, the predicted data points in red colour on the scatter plot followed a clear pattern of the testing data point in blue colour, which indicates the polynomial regression model of this experiment 2a has successfully identified the underlying relationship between the features and the target variable "Price".

4.3.2.2 Experiment 2b (PR Model)

The second experiment 2b includes fine tuning the polynomial regression model that was previously built on experiment 2a. The main reason to fine-tune the polynomial regression model is to optimize its model's parameters in order to achieve a better performance. Moreover, the objective of this experiment 2b is to minimize the mean squared error (MSE) of the polynomial regression model, as well as analyse the impact of fine-tunning key hyperparameter based on the model's performance.

Below figure 24 shows the process of fine tuning the polynomial regression model with the used of "Parameter Grid", "Ridge Regression Model", "Grid Search Cross Validation", and "Fit Grid Search". Firstly, different value such as 0.01, 0.1, 1, 10, 100, 1000, 10000 was tested to regularize the parameter "alpha" of the ridge regression model. Next, the grid search was utilized to perform the hyperparameter tunning with the scoring of minimizing the mean squared error of the model. After fine tuning process was done, it will train/fit the model with the fine-tunned hyperparameter on the training dataset "X2_train" and "y2_train".



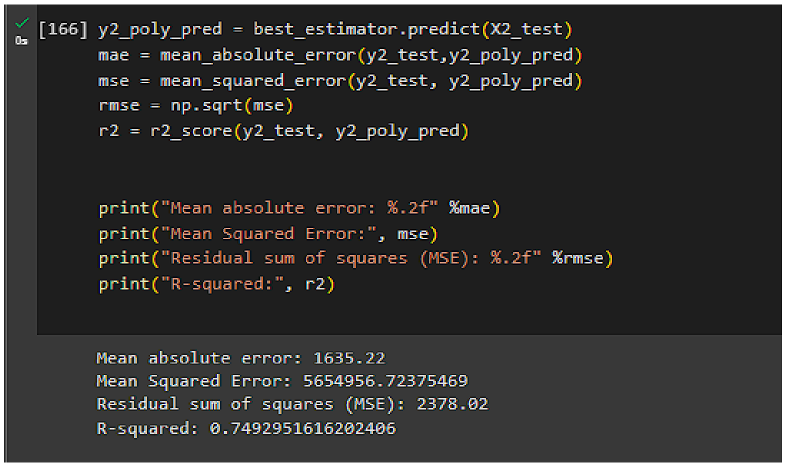
```
param_grid = {
    'alpha': [0.01, 0.1, 1, 10,100,1000,10000],  # Test different values of alpha

}

ridge_reg = Ridge()
grid_search = GridSearchCV(ridge_reg, param_grid, cv=5, scoring='neg_mean_squared_error')
grid_search.fit(X2_train, y2_train)
```

```
GridSearchCV
GridSearchCV(cv=5, estimator=Ridge(),
             param_grid={'alpha': [0.01, 0.1, 1, 10, 100, 1000, 10000]},
             scoring='neg_mean_squared_error')
estimator: Ridge
Ridge()
Ridge
Ridge()
```

```
best_alpha = grid_search.best_params_['alpha']

best_estimator = grid_search.best_estimator_
```

Figure 24: Fine-Tunning the model.

The following are the performance metrics result from the fine-tunned hyperparameter polynomial regression model:



```
[166] y2_poly_pred = best_estimator.predict(X2_test)
      mae = mean_absolute_error(y2_test,y2_poly_pred)
      mse = mean_squared_error(y2_test, y2_poly_pred)
      rmse = np.sqrt(mse)
      r2 = r2_score(y2_test, y2_poly_pred)


      print("Mean absolute error: %.2f" %mae)
      print("Mean Squared Error:", mse)
      print("Residual sum of squares (MSE): %.2f" %rmse)
      print("R-squared:", r2)
```

```
      Mean absolute error: 1635.22
      Mean Squared Error: 5654956.72375469
      Residual sum of squares (MSE): 2378.02
      R-squared: 0.7492951616202406
```

Figure 25: 2b Polynomial Regression Model Performance Result

After analysing the performance metrics from experiment 2b model, it was noticeable that the mean squared error increased compared to the previous experiment 2a model and the R2 score dropped as well from 0.80 to 0.75. Unfortunately, the experiment 2b was unable to improve the performance of the polynomial regression model successfully in terms of R2 accuracy and the errors by fine tuning the hyperparameters.

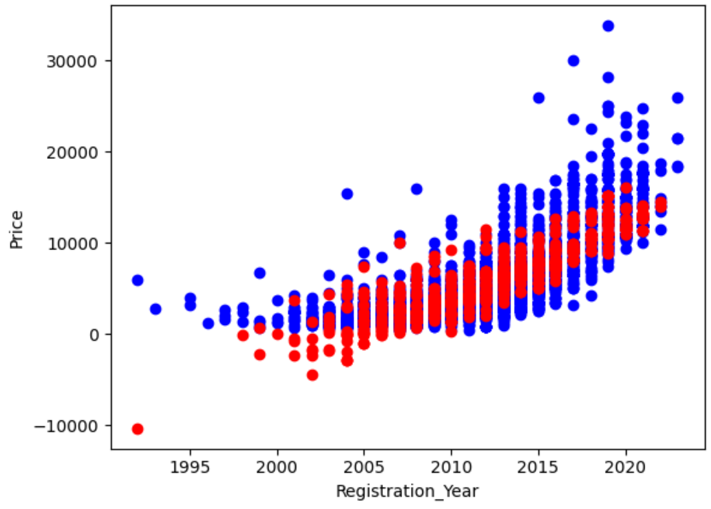


Figure 26: Scatterplot of 2b model prediction.

As comparison between both scatterplots from 2a and 2b model prediction, 2a model was able to predict the datapoint more accurately than 2b model due to better performance. Two main factors that affected 2b model prediction’s visualization was higher error and a lower R2 score compared to the previous experiment 2a model.

4.3.2.3 Summary of polynomial regression model

In summary of these two experiments (2a and 2b), both were conducted utilizing polynomial regression model to perform predictions in used car price value which both experiments were then evaluated based on their prediction performance. The main difference between the experiments lies on the use of fine-tuning, where in experiment 2a polynomial regression model was only trained on random parameter, while experiment 2b polynomial regression model’s hyperparameter was fine-tuned utilizing grid search and ridge regression for regularization. The main objective of these two experiments was to explore and identify the changes in model performance regarding the use of fine-tuning.

Experiment 2a, the polynomial regression model was trained on the training dataset “X2_train” and “y2_train” with a polynomial feature of degree at 3 to predict the used car’s price value. The following are the performance metrics from the experiment 2a:

- MAE: 1313.98
- MSE: 4513792.78
- RMSE: 2124.57
- R2: 0.80

Observation of this experiment 2a through analysing the above performance metrics shows that polynomial regression model may suggest a better machine learning model selection compared to linear regression model. As polynomial regression model can identify curve-fit of the data and non-linear relationship between the features and target variable “Price”, therefore it has a better prediction performance with a lower error rate while maintaining a higher R2 score of 0.80.

Experiment 2b, the purpose of Experiment 2b was to fine-tune the polynomial regression model developed in Experiment 2a in order to improve its performance. Optimizing the model's parameters was the main goal, with a particular emphasis on reducing the mean squared error (MSE). Ridge regression was used in the experiment as a regularization technique, and to determine the best regularization strength, a grid search was carried out over a range of alpha values (0.01, 0.1, 1, 10, 100, 1000, and 10,000). The hyperparameter space was methodically explored using grid search cross-validation, with scores determined by the minimization of mean squared error. The following are the performance metrics from the experiment 2b:

- MAE: 1635.22
- MSE: 5654956.72
- RMSE: 2378.02
- R2: 0.75

Observation of this experiment 2b shows that the experiment demonstrated a rise in mean squared error in contrast to Experiment 2a, suggesting that the process of fine-tuning did not result in an enhancement of the model's performance. Furthermore, the R2 value decreased from 0.80 to 0.75, indicating that the model may not have been well tuned for used car price prediction despite the hyperparameter modifications.

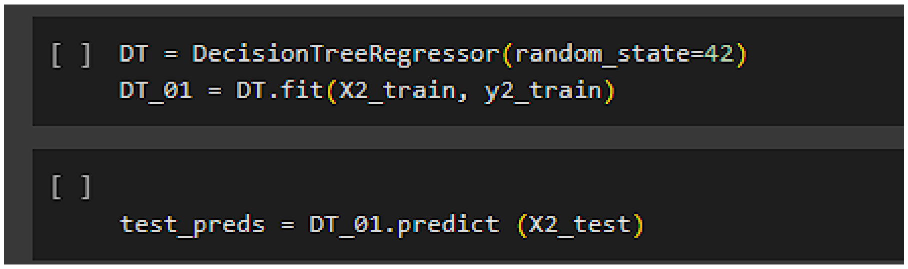
The key findings from these two experiments 2a and 2b suggested that polynomial regression model might be a better machine learning model selection compared to linear regression for predicting used car prices. As polynomial regression model capable of identifying more complex relationships within the dataset which leads to a better performance of predicting used car price.

4.3.3 Decision Tree Regression Model

Decision tree is a non-parametric supervised machine learning model that can be utilized for both regression and classification tasks. The components of decision tree consist of internal nodes, leaf nodes, branches, and root node which make up its hierarchical tree structure and decision tree works by splitting the data into 2 datasets at each node where the dept of the tree indicates how many times the data has been split by the model [18]. Lastly, the leaf nodes of the decision tree are represented as the final decision.

4.3.3.1 Experiment 3a (DT Model)

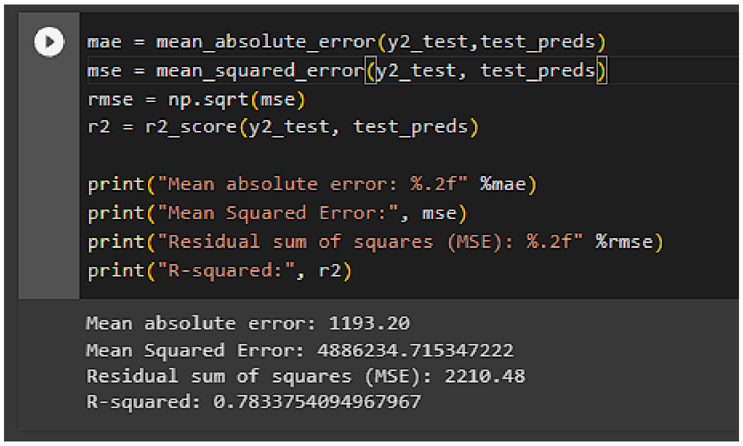
As decision tree is a popular machine learning model in predicting both regression and classification tasks. Therefore, in this experiment 3a, decision tree regression model will be employed to predict the used car price based on the label encoded dataset that was used from the previous experiments. The independent variables that were used during this experiment includes (title, Mileage, registration year, previous owner, fuel type, body type, engine, gearbox, doors, seats, emission class, and service history), the features mentioned will then be implemented and trained into decision tree model. The main objective of this experiment is to achieve high prediction accuracy in predicting the target variable “Price” by building a decision tree model.



```
[ ]  DT = DecisionTreeRegressor(random_state=42)
     DT_01 = DT.fit(X2_train, y2_train)

[ ]
     test_preds = DT_01.predict (X2_test)
```

Figure 27: Experiment 3a Decision Tree Model

The code of decision tree model implementation is shown in figure 27, firstly, the decision tree model was imported from the scikit-learn library and a random state set at 42. Next, .fit() function was used to train the decision tree model on the training dataset “X2_train” which contained all the independent variables, and “y2_train” which contained the dependent variable “Price”. After the modelling process was completed, the trained decision tree model was used to predict on the testing dataset “X2_test”.



```
mae = mean_absolute_error(y2_test,test_preds)
mse = mean_squared_error(y2_test, test_preds)
rmse = np.sqrt(mse)
r2 = r2_score(y2_test, test_preds)

print("Mean absolute error: %.2f" %mae)
print("Mean Squared Error:", mse)
print("Residual sum of squares (MSE): %.2f" %rmse)
print("R-squared:", r2)
```

```
Mean absolute error: 1193.20
Mean Squared Error: 4886234.715347222
Residual sum of squares (MSE): 2210.48
R-squared: 0.7833754094967967
```

Figure 28: 3a Decision Tree Model Performance Result

The above figure 28 shows the prediction performance of this experiment 3a decision tree model.

- Mean Absolute Error (MAE): 1193.20
- Mean Squared Error (MSE): 4886234.72
- Residual Sum of squares (RMSE): 2210.48
- R--Squared (R2): 0.78

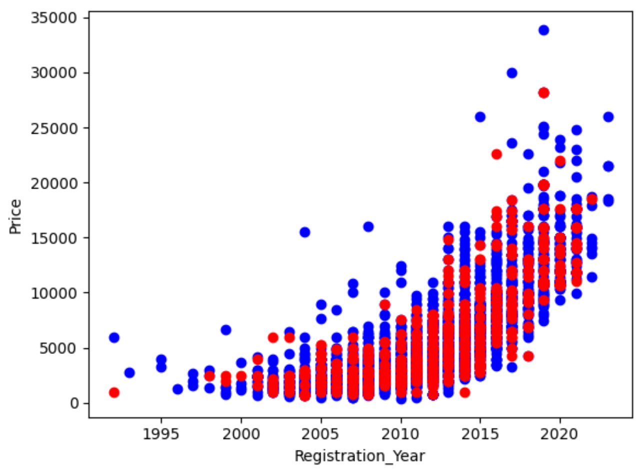


Figure 29: Scatterplot of 3a model prediction.

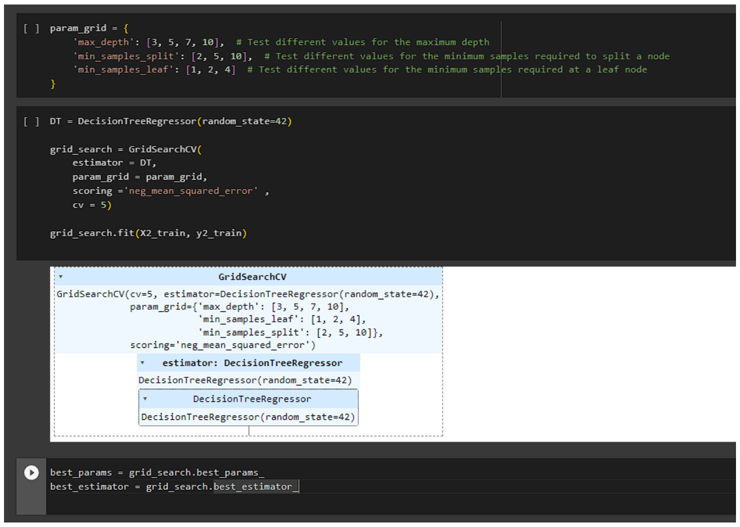
The visualization of the 3a's model prediction is shown in figure 29, as the R2 score of this model is relatively high at 0.78. Therefore, the predicted data points in red colour on the scatter plot followed a clear pattern of the testing data point in blue colour, which indicates the decision tree model of this experiment 3a has successfully identified the underlying relationship between the features and the target variable "Price".

4.3.3.2 Experiment 3b (DT Model)

The second experiment 3b was conducted to optimize the prediction performance of the decision tree model that was previously implemented in experiment 3a. By fine-tuning the decision tree model, it can search for the best parameters in terms of maximum depth, minimum samples split, and minimum samples leaf for the decision tree that will result a better performance.

The code of fine tuning the decision tree model is shown in figure 30. Firstly, the parameter grid consisted of 3 main components including maximum depth, minimum samples split, and minimum samples leaf. Each components has a combination of different hyperparameter values that will be identified the best parameter with the GridSearchCV() function. In this case, the maximum depth of the decision tree model was tested with the values ranging from (3, 5, 7, 10), the minimum samples split was tested with the values ranging from (2, 5, 10), and the minimum samples leaf was tested with the values ranging from (1, 2, 4).

Next, GridSearchCV() function was utilized to perform a cross validation grid search over the hyperparameter that was defined previously. Then the GridSearchCV will fit its object into the training dataset “X2_train” and “y2_train”, this process includes iterating over all the combinations of hyperparameters and produce the most optimized decision tree model. After fine-tunning the model, the optimized decision tree model was used to predict on the testing dataset “X2_test” and “y2_test”.



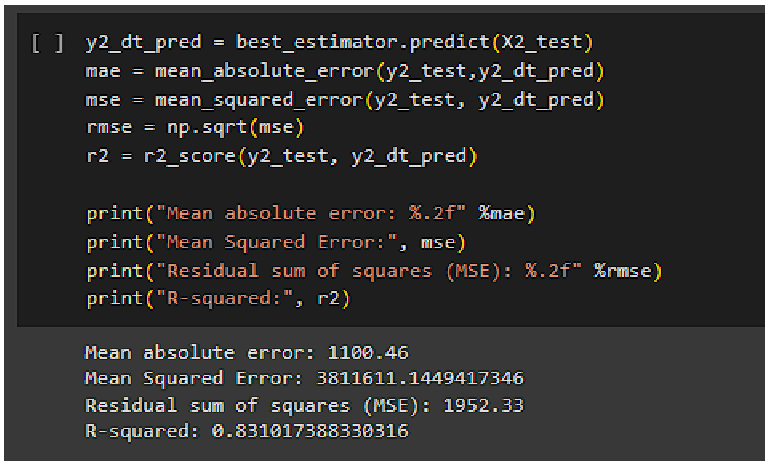
```
[ ] param_grid = {
        'max_depth': [3, 5, 7, 10],  # Test different values for the maximum depth
        'min_samples_split': [2, 5, 10],  # Test different values for the minimum samples required to split a node
        'min_samples_leaf': [1, 2, 4]  # Test different values for the minimum samples required at a leaf node
    }
```

```
[ ] DT = DecisionTreeRegressor(random_state=42)

    grid_search = GridSearchCV(
        estimator = DT,
        param_grid = param_grid,
        scoring ='neg_mean_squared_error' ,
        cv = 5)

    grid_search.fit(X2_train, y2_train)
```

```
GridSearchCV
GridSearchCV(cv=5, estimator=DecisionTreeRegressor(random_state=42),
             param_grid={'max_depth': [3, 5, 7, 10],
                         'min_samples_leaf': [1, 2, 4],
                         'min_samples_split': [2, 5, 10]},
             scoring='neg_mean_squared_error')
        estimator: DecisionTreeRegressor
        DecisionTreeRegressor(random_state=42)
              DecisionTreeRegressor
        DecisionTreeRegressor(random_state=42)
```

```
best_params = grid_search.best_params_
best_estimator = grid_search.best_estimator_
```

Figure 30: Fine tuning the decision tree model.



```
[ ] y2_dt_pred = best_estimator.predict(X2_test)
    mae = mean_absolute_error(y2_test,y2_dt_pred)
    mse = mean_squared_error(y2_test, y2_dt_pred)
    rmse = np.sqrt(mse)
    r2 = r2_score(y2_test, y2_dt_pred)

    print("Mean absolute error: %.2f" %mae)
    print("Mean Squared Error:", mse)
    print("Residual sum of squares (MSE): %.2f" %rmse)
    print("R-squared:", r2)
```

```
Mean absolute error: 1100.46
Mean Squared Error: 3811611.1449417346
Residual sum of squares (MSE): 1952.33
R-squared: 0.831017388330316
```

Figure 31: 3b Optimized Decision Tree Model Performance Result

The above figure 31 shows the prediction performance of this experiment 3b decision tree model after fine tuning. By comparing the performance metrics of before and after fine tuning the decision tree, it shows that the optimized decision tree model has greatly improved in terms of a higher accuracy while maintaining a lower error rate.

- Mean Absolute Error (MAE): 1100.46
- Mean Squared Error (MSE): 3811611.14
- Residual Sum of squares (RMSE): 1952.33
- R-Squared (R2): 0.83

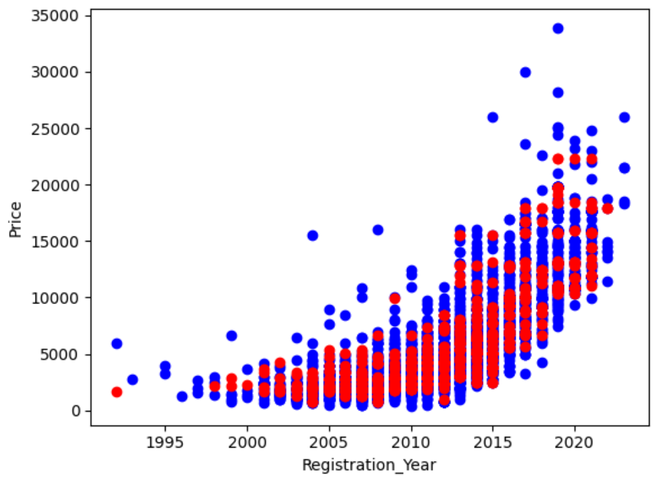


Figure 32: Scatterplot of 3b optimized model prediction.

The visualization of the 3b's model prediction is shown in figure 32, as the optimized model has a higher R2 score at 0.83 compared to the 3a model. Therefore, the predicted data points in red colour on the scatter plot able followed a clearer pattern of the testing data point in blue colour, which indicates the optimized decision tree model of this experiment 3b has been fine tunned successfully.

4.3.3.3 Summary of decision tree model

In summary of these two experiments (3a and 3b), both were conducted utilizing decision tree model to perform prediction in used car price value which both experiments were then evaluated based on their prediction performance. The main key that separates both experiments is the used of hyperparameter tuning, where experiment 3a decision tree model was trained without assigning a parameter to the model. On the other hand, 3b decision tree model was to optimize the model that was trained in the previous experiment 3a. The main objective of these two experiments was to obtain a high prediction performance while successfully optimized the based line decision tree model.

Experiment 3a, during this experiment it was mainly focused on training the base line decision tree model with no parameter such as the maximum depth, minimum samples split, and minimum samples leaf was assigned to it. Additionally, the decision tree model was trained on the training dataset “X2_train” and “y_train” that included the label encoding of categorical variables. The following are the performance metrics from the experiment 3a.

- MAE: 1193.20
- MSE: 4886234.72
- RMSE: 2210.48
- R2: 0.78

Observation of this experiment 3a through analysing the above performance metrics shows that the based line decision tree model achieved an average prediction performance. However, it may suggest that assigning no parameters to the model might achieve a lower performance. Therefore, in the next experiment 3b, the baseline model from experiment 3a will be optimized and may achieve a higher performance.

Experiment 3b, the objective of this experiment is to enhance the performance of the baseline decision tree model that was implemented in experiment 3a. The model is optimized by fine-tuning the decision tree model’s parameters such as maximum depth, minimum samples split, and minimum samples leaf. The parameters have various combinations of hyperparameter values where the maximum depth was tested with (3, 5, 7, 10), minimum samples split (2, 5, 10), and minimum samples leaf with (1, 2, 4). Then GridSearchCV() function is used to perform a cross validation grid search over the hyperparameter. Below are the performance metrics from the experiment 3b.

- MAE: 1100.46
- MSE: 3811611.14
- RMSE: 1952.33
- R2: 0.83

Observation of this experiment 3b after analysing the above performance metrics show that the base line decision tree model has successfully been optimized as the performance metrics above shown the error metrics such as MAE, MSE, and RMSE has significantly decreased where MAE has dropped from 1193.20 to 1100.46, and MSE has dropped from 4886234.72 to 3811611.14. Moreover, the R2 score has improved 6.41% from the value of 0.78 to 0.83.

4.4 Summary of implementation and result

Linear regression model

Experiment 1a, utilized only numerical variables, excluding categorical variables and aimed to understand the relationship between numerical variables and the target variable "Price".

Performance metrics result:

MAE: 1792.37

MSE: 6809460.37

RMSE: 2609.49

R2: 0.70

Experiment 1b, included label encoding of categorical variables and explored the impact of incorporating categorical variables into the model.

Performance metrics result:

MAE: 1635.76

MSE: 5649380.85

RMSE: 2376.84

R2: 0.75

Polynomial regression model

Experiment 2a, utilized polynomial regression with a degree of 3 and captured non-liner relationships between the features and the target variable.

Performance metrics result:

MAE: 1313.98

MSE: 4513792.78

RMSE: 2124.57

R2: 0.80

Experiment 2b, the polynomial regression model from experiment 2a has been fine-tuned and aimed to optimize the model parameters for better performance.

Performance metrics result:

MAE: 1635.22

MSE: 5654956.72

RMSE: 2378.02

R2: 0.75

<u>Decision tree model</u>

Experiment 3a, base line decision tree model was trained without hyperparameter running.

Performance metrics result:

MAE: 1193.20

MSE: 4886234.72

RMSE: 2210.48

R2: 0.78

Experiment 3b, the base line decision tree model was optimized by fine-tuning the hyperparameter such as maximum depth, minimum samples split, and minimum samples leaf.

Performance metric result:

MAE: 1100.46

MSE: 3811611.14

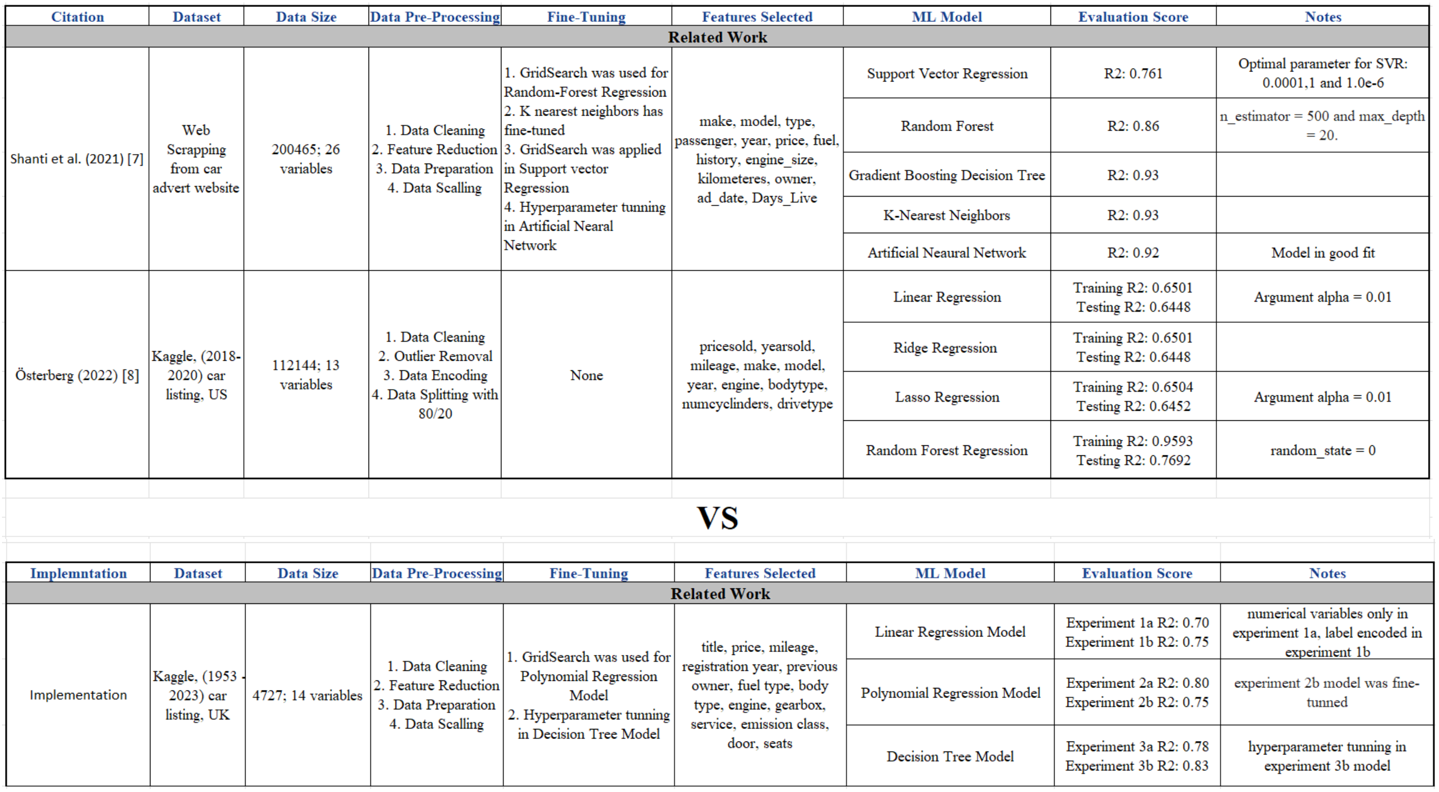
RMSE: 1952.33

R2: 0.83

5.0 Analysis and Recommendations

The problem statement of this research involves the utilization of 3 different machine learning models such as linear regression model, polynomial regression model, and decision tree model to predict the price value of used car. The main objective of this research is identifying the prediction performance of these machine learning models.

In this following section, we will interpret and analysis the comparison between benchmark and the implemented solution that have been experimented in this research. The objective of this analysis is to identify the effectiveness of different machine learning models in predicting used car's price, based on the findings, recommendations will be provided for further improvement. The table of comparison between the benchmark and the implementation of this research is shown in figure 33 for better understanding.



Citation	Dataset	Data Size	Data Pre-Processing	Fine-Tuning	Features Selected	ML Model	Evaluation Score	Notes
Related Work								
Shanti et al. (2021) [7]	Web Scrapping from car advert website	200465; 26 variables	1. Data Cleaning 2. Feature Reduction 3. Data Preparation 4. Data Scalling	1. GridSearch was used for Random-Forest Regression 2. K nearest neighbors has fine-tuned 3. GridSearch was applied in Support vector Regression 4. Hyperparameter tunning in Artificial Nearal Network	make, model, type, passenger, year, price, fuel, history, engine_size, kilometeres, owner, ad_date, Days_Live	Support Vector Regression	R2: 0.761	Optimal parameter for SVR: 0.0001,1 and 1.0e-6
						Random Forest	R2: 0.86	n_estimator = 500 and max_depth = 20.
						Gradient Boosting Decision Tree	R2: 0.93	
						K-Nearest Neighbors	R2: 0.93	
						Artificial Neaural Network	R2: 0.92	Model in good fit
Österberg (2022) [8]	Kaggle, (2018-2020) car listing, US	112144; 13 variables	1. Data Cleaning 2. Outlier Removal 3. Data Encoding 4. Data Splitting with 80/20	None	pricesold, yearsold, mileage, make, model, year, engine, bodytype, numcylinders, drivetype	Linear Regression	Training R2: 0.6501 Testing R2: 0.6448	Argument alpha = 0.01
						Ridge Regression	Training R2: 0.6501 Testing R2: 0.6448	
						Lasso Regression	Training R2: 0.6504 Testing R2: 0.6452	Argument alpha = 0.01
						Random Forest Regression	Training R2: 0.9593 Testing R2: 0.7692	random_state = 0

VS

Implemntation	Dataset	Data Size	Data Pre-Processing	Fine-Tuning	Features Selected	ML Model	Evaluation Score	Notes
Related Work								
Implementation	Kaggle, (1953 - 2023) car listing, UK	4727; 14 variables	1. Data Cleaning 2. Feature Reduction 3. Data Preparation 4. Data Scalling	1. GridSearch was used for Polynomial Regression Model 2. Hyperparameter tunning in Decision Tree Model	title, price, mileage, registration year, previous owner, fuel type, body type, engine, gearbox, service, emission class, door, seats	Linear Regression Model	Experiment 1a R2: 0.70 Experiment 1b R2: 0.75	numerical variables only in experiment 1a, label encoded in experiment 1b
						Polynomial Regression Model	Experiment 2a R2: 0.80 Experiment 2b R2: 0.75	experiment 2b model was fine-tunned
						Decision Tree Model	Experiment 3a R2: 0.78 Experiment 3b R2: 0.83	hyperparameter tunning in experiment 3b model

Figure 33: Table comparison between benchmark and implementation.

Comparing the linear regression model implemented in this research, it shown that the experiment 1b has achieved a higher R2 score at 0.75 while experiment 1a has only achieved a R2 score at 0.70. Experiment 1b has performed slight better due label encoding the categorical variables and implemented into the linear regression model unlike experiment 1a which implemented with numerical variables only, by implementing a wider range of features into the machine learning model which allow the model to identify the pattern of the data that was previously qualitative. The experiment 1a and 1b attempted has achieved the expectation as it was expected that utilizing label encoding method will further improve the baseline linear regression model's prediction performance. The recommendation of this model implementation is to utilize other encoding methods as well such as one-hot-encoding and target-encoding methods to monitor if any further improvements can be achieved.

Referring to the table above, it shown that the linear regression model implemented by the Osterberg (2020) has achieved a training R2 score of 0.6501 and testing R2 score of 0.6448 with the argument alpha value set at 0.01. On the other hand, the linear regression model implemented from this research has achieved a R2 score of 0.70 from the experiment 1a while a R2 score of 0.75 from the experiment 1b. By comparison of both most optimized performance result from the benchmark and the linear regression model implemented in this research, it shows that the linear regression model implemented in this research has a higher R2 score at 0.75 compared to the benchmark R2 score at 0.6501, which shows the percentage between 0.6501 and 0.75 is approximately 15.36%. Additionally, the linear regression model's RMSE value of the benchmark (Osterberg 2020) is 122.46% higher than the linear regression model's RMSE value that was implemented in this research.

Linear regression model implemented by the benchmark (Osterberg, 2020) achieved a lower R2 score and a higher RMSE may cause by the assigned value of argument alpha at 0.01, as the effectiveness of the linear regression model can be impacted by the argument alpha. If the value of the argument alpha is too low the linear regression model may overfit the data points while if the value of argument alpha is too high the model may underfit the data points.

Moving on to the second implemented solution, polynomial regression model. Let's compare the polynomial regression model that was implemented in experiment 2a and experiment 2b before comparing it to the benchmark for a comprehensive understanding. In experiment 2a, the baseline polynomial regression model was trained with a degree of 3, and it has achieved a R2 score of 0.80. While experiment 2b was focusing on optimizing the baseline polynomial regression model, however an unexpected lower R2 score at 0.75 was outputted, indicated a decrease of 6.25% in the R2 score. Moreover, the polynomial regression model in experiment 2b has performed significant worse in term of MAE as the MAE value has increase from 1313.98 to 2125.57 which is a 61.68% increased.

The cause of worsen the baseline polynomial regression model during the hyperparameter tunning process in experiment 2b may be caused by the inappropriate assigned hyperparameter range as it is required to assign proper value in the hyperparameter range for the grid search to lead to improvements. Other than that, it could cause by the inappropriate used of regularization method such as Ridge regression that was implemented in experiment 2b optimization due to its sensitivity to the scale of the input features. Therefore, the experiment 2b conducted did not achieved of what is expected as the expected result was improvement in prediction performance due to inappropriate assigned value in the hyperparameter range and used of regularization method. The recommendation to resolve this issue and achieve the expected result is by assigning suitable parameters range and utilize other regularization method instead such as the lasso regression.

By comparing the polynomial regression model that was implemented in this research to the benchmark in figure 33, it shows that the baseline polynomial regression model in experiment 2a has performed significantly better than the machine learning model implemented in the benchmark such as the support vector regression, linear regression, ridge regression and lasso regression. The baseline polynomial regression model implemented in this research achieved R2 score of 0.80 while the benchmark's models that ware mentioned have only achieved R2 score ranged from 0.6448 to 0.761. There are several reasons why baseline polynomial regression model implemented in this research outperformed the benchmark's model, the data characteristics of the used car price and its features might have a non-linear relationship which makes polynomial regression model a more suitable machine learning model to be implemented which led to a higher R2 score compared to the other models. As polynomial regression model is capable of identifying and capturing non-linear trends more effectively compared to linear models like support vector regression, linear regression, ridge regression and lasso regression.

Lastly, the third machine learning model implemented in this research was decision tree model. By analysing the prediction performance of experiment 3a and 3b, the decision tree's prediction performance conducted in experiment 3b has performed slightly better than decision tree in experiment 3a. It is because in experiment 3b, the baseline model decision tree was optimized by hyperparameter tunning to achieve the most suitable parameter for the model. In result, R2 score of decision tree in experiment 3b has increased from 0.78 to 0.83 which is a 6.41% increased. Moreover, the MSE of the optimized decision tree model has dramatically decreased from 4886234.72 to 3811611.14 which is a 21.99% decreased. Therefore, it indicated that the baseline decision tree model has successfully been fine-tuned.

6.0 Conclusion

In conclusion, this research resolves the growing concern of fraud in the expanding used-car market by utilizing machine learning model to predict used car price. Using dataset retrieved from Kaggle and 3 different machine learning models including linear regression, polynomial regression, and decision tree model.

Each implemented machine learning model has conducted with 2 different experiments to achieve the most optimized model and prediction performance result. By utilizing label encoding method and fine-tuning method, linear regression model and decision tree model has successfully been optimized while polynomial regression model did not achieve the expected performance result.

Throughout this study, the machine learning model that has achieved the highest prediction performance in terms of MAE, MSE, RMSE, and R2 was the optimized decision tree model implemented during experiment 3a. Which has achieved a R2 score of 0.83 while maintaining low MAE at 1100.46 and low MSE at 3811611.14.

Overall, this study provides comprehensive understanding of different machine learning models in terms of prediction of used car price, and further research may be conducted using one-hot-encoding method and other regularization methods.

References:

[1] Morgan Chase, J.P. (2022) *Inflation and the auto industry: When will car prices drop?, J.P. Morgan*. Available at: https://www.jpmorgan.com/insights/economy/economy/when-will-car-prices-drop

[2] MordorIntelligence (2023) *Malaysia used car market size & share analysis - industry research report - growth trends, Malaysia Used Car Market Size & Share Analysis - Industry Research Report - Growth Trends*. Available at: https://www.mordorintelligence.com/industry-reports/malaysia-used-car-market

[3] S, D. (2021) *Avoid getting ripped off when buying or selling a car*, *Carsome Malaysia*. Available at: https://www.carsome.my/news/item/used-car-scams

[4] YangJin, C. (2021) *Price prediction of used cars using machine learning | IEEE conference ...*, *Price Prediction of Used Cars Using Machine Learning*. Available at: https://ieeexplore.ieee.org/document/9696839

[5] Najib, T. (2023) *Used car price prediction dataset*, *Kaggle*. Available at: https://www.kaggle.com/datasets/taeefnajib/used-car-price-prediction-dataset

[6] Harkiran (2023) *Why is data visualization so important in data science?*, *GeeksforGeeks*. Available at: https://www.geeksforgeeks.org/why-is-data-visualization-so-important-in-data-science/

[7] Shanti, N. *et al.* (2021) *Machine Learning-Powered Mobile App for Predicting Used Car Prices*, *Machine learning-powered mobile app for predicting used car prices*. Available at: https://dl.acm.org/doi/fullHtml/10.1145/3502300.3502307

[8] Österberg, P. (2022a) *Price Prediction for Used Cars A Comparison of Machine Learning Regression Models*, *Price Prediction for Used Cars*. Available at: https://www.diva-portal.org/smash/get/diva2:1674070/FULLTEXT01.pdf.

[9] Javaid, S. (2023) *Guide to datasets for ML (machine learning) in 2023, AIMultiple*. Available at: https://research.aimultiple.com/datasets-for-ml/

[10] Abrahams, A. (2023) *Best practices for data cleaning and preprocessing*, *Jumping Rivers*. Available at: https://www.jumpingrivers.com/blog/best-practices-data-cleaning-r/

[11] Singh, H. (2020) *Data preprocessing*, *Medium*. Available at: https://towardsdatascience.com/data-preprocessing-e2b0bed4c7fb

[12] Buhl, N. (2023) *Mastering Data Cleaning & Data preprocessing, Data Cleaning & Data Preprocessing for Machine Learning*. Available at: https://encord.com/blog/data-cleaning-data-preprocessing/

[13] Huang, R. (2019) *When to use different machine learning algorithms: A simple guide, freeCodeCamp.org*. Available at: https://www.freecodecamp.org/news/when-to-use-different-machine-learning-algorithms-a-simple-guide-ba615b19fb3b/

[14] W3School (2022) *Machine learning - linear regression, Python Machine Learning Linear Regression*. Available at: https://www.w3schools.com/python/python_ml_linear_regression.asp

[15] MUTİ, S. and YILDIZ, K. (2023) *Using linear regression for used car price prediction - dergipark, Using Linear Regression For Used Car Price Prediction*. Available at: https://dergipark.org.tr/en/download/article-file/2242032

[16] W3School (2022b) *Machine learning - polynomial regression, Python Machine Learning Polynomial Regression*. Available at: https://www.w3schools.com/python/python_ml_polynomial_regression.asp

[17] Alifarahmandfar (2023) *car price prediction: Polynomial regression, Kaggle*. Available at: https://www.kaggle.com/code/alifarahmandfar/car-price-prediction-polynomial-regression

[18] Saini, A. (2023) *Decision tree algorithm - A complete guide, Analytics Vidhya*. Available at: https://www.analyticsvidhya.com/blog/2021/08/decision-tree-algorithm/

[19] Dorfer, T.A. (2023) *How to choose the best evaluation metric for regression problems, Medium*. Available at: https://towardsdatascience.com/how-to-choose-the-best-evaluation-metric-for-regression-problems-b9f2e60e25ef

[20] Simplilearn (2023) *What is statistical analysis? types, methods and examples: Simplilearn, Simplilearn.com*. Available at: https://www.simplilearn.com/what-is-statistical-analysis-article