Loughborough University London

Cover Sheet for Coursework

Participants must complete this cover sheet to accompany each piece of coursework submitted. **No work will be marked without completion of this sheet.**

Student Name:		Student number:	
Submission Date:	16th December 2020		
Programme Title:	Cyber Security and Big Data		
Module Title:	Principles of Data Science		
Assignment Title:	Automobile Price Prediction		

Declaration

By making this submission I confirm that the attached coursework is my own work and that anything taken from or based upon the work of others – or previous work of mine – has its source clearly and explicitly cited; I understand that failure to do so may constitute Academic Misconduct.

Contents

1 Introduction

1.1 Aims and Objectives

This project analyses an automobile dataset with the aim of obtaining a model which can be used to predict prices of automobiles using their attributes and in order to do this, the following questions were factored:

- What useful insights can be gotten from the data set ?
- What attributes play a key role in factoring the price of automobiles ?
- How does the amount of attributes used in model fitting affect the accuracy of proposed models in predicting prices of automobiles ?
- How does the amount data points available for training models affect their accuracy in predicting car prices ?

The following objectives have been set based on the goal of this project:

- Perform data cleaning in to ensure that all missing values are filled, data in all columns are in the appropriate formats, outliers are removed etc.
- Visualise the data set to uncover minute or useful pieces of information which are not immediately evident and improve understanding of the dataset.
- Split data set into training and validation sets. Using training set, Identify and select features which are most useful for model fitting.
- Fit models using training set, check accuracy using validation sets.
- Evaluate model performance by examining curse of dimensionality, using model error measurements, cross validation, R2 values etc.

1.2 Dataset Description

Name	**Description**
Title	Automobile Data Set
Data-set Size	Rows: 205 Columns: 26
Data-set Characteristics	Multivariate
Data-set Description	Autos are split into 3 different entity types: I. Auto characteristics II. Insurance risk rating (Symboling) III. Normalized losses in use compared to other cars. A symbolling value of +3 indicates the auto is risky while -3 tells us it is relatively safe. Normalized losses represent the average loss per car each year. Each value is normalized for all autos with a particular size class (e.g. station wagons, sport, 2-door etc.)
Source	https://archive.ics.uci.edu/ml/datasets/Automobile
Date Donated	19/05/1987
Associated Task	Regression
Missing values	59
Attributes	26
Attribute Characteristics	Categorical, Integer, Real
Attribute information	
Attribute	**Range**
symbolizing	-3, -2, -1, 0, 1, 2, 3
normalized-losses	Continuous: 65-256
fuel-type	diesel, gas
aspiration	std, turbo
num-of-doors	four, two
body-style	hardtop, wagon, sedan, hatchback, convertible
Drive-wheels	4wd, fwd, rwd
Engine-location	front, rear
Wheel-base	Continuous: 86.6-120.9
Length	Continuous: 141.1-208.1
Width	Continuous: 60.3-72.3
Height	Continuous: 47.8-59.8
Curb-weight	Continuous: 1488-4066
make	Alfa-Romero, Audi, BMW, Chevrolet, Dodge, Honda, Isuzu, jaguar, Mazda, Mercedes-benz, mercury, Mitsubishi, Nissan, Peugeot, Plymouth, Porsche, Renault, Saab, Subaru, Toyota, Volkswagen, Volvo
engine-type	dohc, dohcv, l, ohc, ohcf, ohcv, rotor
num-of-cylinders	eight, five, four, six, three, twelve, two
engine-size	Continuous: 61-326
fuel-system	1bbl, 2bbl, 4bbl, idi, mfi, mpfi, spdi, spfi
bore	Continuous: 2.54-3.94
stroke	Continuous: 2.07-4.17
compression-ratio	Continuous: 7-23
horsepower	Continuous: 48-288
peak-rpm	Continuous: 4150-6600
city-mpg	Continuous: 13-49
highway-mpg	Continuous: 16-54
price	Continuous: 5118 -45400

Table 1: Dataset Overview

2 Methodology & Model Overview

A brief summary of the approach taken in order to achieve the projects aim and objectives is shown in figure 1.

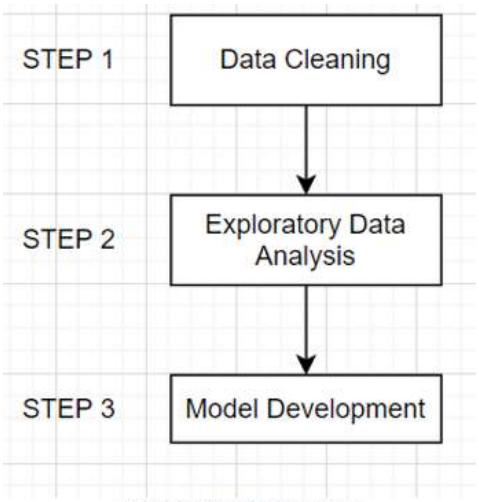

Figure 1: Model Overview

2.1 Data Cleaning

During data collection and storage, there tends to be quality issues due to errors like misspellings, missing information, invalid data or wrong data formats [1]. Data cleaning is the process by which these errors or inconsistencies are removed from raw data in order to convert it to a form that is suitable for further analysis. In cleaning the selected dataset, the following steps were followed:

- **Data formatting**: this ensured that data was stored properly to prevent inaccuracies during modelling. Some formatting procedures used in this project were adding headers in each column, ensuring use of correct data-types in storing variables and removing outliers.
- **Identifying and handling missing/duplicate values:** once missing values were spotted, they were either dropped or replaced. It is often better to replace missing values or even leave them rather than drop them. Missing numerical variables were replaced using an average value (mean) while missing categorical variables were replaced using the highest occurring values (mode).
- **Encoding:** regression and classification analysis often makes use of categorical variables but only numerical values are used to develop machine learning models [2]. Encoding helps solve this issue by converting nominal/categorical variables to numerical form. In this project label encoding was used to encode categorical variables.

2.2 Exploratory Data Analysis [EDA]

EDA helps an analyst address the question “what is going on here?” when a new dataset is being examined [3]. It is a way of telling a story about a dataset using statistical tools. EDA is very important as it allows a data analyst to:

- Extract important variables through hypothesis testing.
- Gain a better understanding of a dataset.
- Uncover relationships between variables.

In this project visualization tools (e.g. bar graphs, heatmaps, boxplots etc.) were used to describe the dataset alongside descriptive statistical tools and probability distribution concepts. This can discussed in section 3.2.

2.3 Model Development

Regression analysis is a popular technique used to determine the relationship between a dependent variable (in this case the target variable - price) and 'n' number of independent variables (in this case, car attributes) [4]. "A model of the relationship is hypothesized, and estimates of the parameter values are used to develop an estimated regression equation." [5].

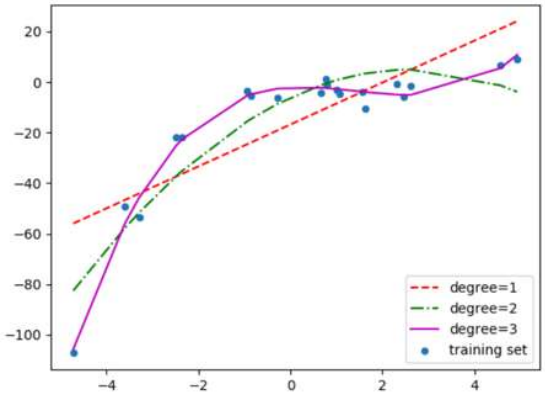

Figure 2: Fitting different models unto a dataset [6]

A linear model does not always give the best fit when modelling a set of data points and as a result polynomial regression is used to try and fit more complex models. This is shown in figure 2 [6]where different models are used on a dataset, it is immediately clear that a linear model (degree = 1) would perform poorly if used in predicting the datapoints.
A generic polynomial regression equation is shown in equation 1 [7].

$$y(x, \mathbf{w}) = w_0 + w_1 x + w_2 x^2 + \ldots + w_M x^M = \sum_{j=0}^{M} w_j x^j \quad (1)$$

In this project, polynomial curves were used to find an equation that models the datapoints as best as possible in a technique referred to as multiple/multivariate polynomial regression [8].

2.3.1 Model Fitting and Evaluation

During model/curve fitting, different tests were done to try and determine the best performing model. These include:

- Exploring how different ratios of training and validation datasets affect the performance of the model.
- Exploring how the number of variables used in model fitting affected the performance of the model.

A further discussion and analysis of the test results can be found in section 3.3.
In order to ensure that the model selected did not overfit or underfit datapoints, different methods or metrics were used to measure model performance or 'goodness of fit':

- **Root Mean Squared Error (RMSE):** RMSE is measure of how close the observed data points are to the values predicted by a model. It is a good measure of how accurate

the proposed model is and is very important in predictive modelling [9]. Low RMSE values are an indication of a good model. Mathematically [7],

$$\text{RMSE} = \sqrt{MSE} = \sqrt{\Sigma_{i=1}^{N}(y_{i,observed} - y_{i,predicted})^2 / N} \quad (2)$$

Where:
MSE = mean squared error
N = sample size
$y_{i,observed}$ = observed values
$y_{i,predicted}$ = predicted values

- **R^2 values:** R^2 values are also a measure of how close data is to the fitted model. R^2 is the percentage of variation explained by a linear model [10]. R^2 values tend to be between 0 and 1 with 1 representing a good fit and 0 a poor fit however they can be negative. Negative R^2 values tend to arise when the model is significantly worse or there is overfitting such that the model is only good in predicting the dataset used in training it.

- **Bias-variance trade-off:** underfitting occurs when a model is not flexible enough and is poor at predicting training and validation data. It often occurs when the dataset is not large enough. Overfitting occurs when a model works well for training data but fails for validation data. The model is said to be too flexible [7]**.**
  In order to prevent overfitting or underfitting, an understanding of bias-variance error trade-off is important. The total error of a model is a sum of bias^2, the variance and irreducible error and as such it is important to minimize bias and variance errors in order to get a good fit [11]. Figure 3 shows how bias and variance affects the performance of a model on a dataset where the centre of the target would be a perfect prediction [11].

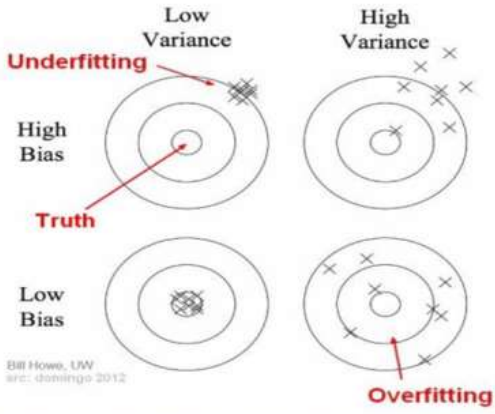

Figure 3: Explaining bias and variance using a bulls-eye diagram [11]

Table 2 summarizes the effect of bias and variance on a models performance.

	Training error	Validation error
High Bias (underfitting)	High	High
High variance (Overfitting)	Low	High

Table 2: Bias-Variance and model performance

3 Results & Discussion

3.1 Data Cleaning

The initial state of the raw data is shown in figure 4.

```
# Import dataset and print first 10 rows
dataset = pd.read_csv('imports-85.data',header = None,  sep = ',')
dataset.head(10)
```

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
0	3	?	alfa-romero	gas	std	two	convertible	rwd	front	88.6	168.8	64.1	48.8	2548	dohc	four	130	mpfi	3.47	2.68	9.0	111	5000	21	27	13495
1	3	?	alfa-romero	gas	std	two	convertible	rwd	front	88.6	168.8	64.1	48.8	2548	dohc	four	130	mpfi	3.47	2.68	9.0	111	5000	21	27	16500
2	1	?	alfa-romero	gas	std	two	hatchback	rwd	front	94.5	171.2	65.5	52.4	2823	ohcv	six	152	mpfi	2.68	3.47	9.0	154	5000	19	26	16500
3	2	164	audi	gas	std	four	sedan	fwd	front	99.8	176.6	66.2	54.3	2337	ohc	four	109	mpfi	3.19	3.40	10.0	102	5500	24	30	13950
4	2	164	audi	gas	std	four	sedan	4wd	front	99.4	176.6	66.4	54.3	2824	ohc	five	136	mpfi	3.19	3.40	8.0	115	5500	18	22	17450
5	2	?	audi	gas	std	two	sedan	fwd	front	99.8	177.3	66.3	53.1	2507	ohc	five	136	mpfi	3.19	3.40	8.5	110	5500	19	25	15250
6	1	158	audi	gas	std	four	sedan	fwd	front	105.8	192.7	71.4	55.7	2844	ohc	five	136	mpfi	3.19	3.40	8.5	110	5500	19	25	17710
7	1	?	audi	gas	std	four	wagon	fwd	front	105.8	192.7	71.4	55.7	2954	ohc	five	136	mpfi	3.19	3.40	8.5	110	5500	19	25	18920
8	1	158	audi	gas	turbo	four	sedan	fwd	front	105.8	192.7	71.4	55.9	3086	ohc	five	131	mpfi	3.13	3.40	8.3	140	5500	17	20	23875
9	0	?	audi	gas	turbo	two	hatchback	4wd	front	99.5	178.2	67.9	52.0	3053	ohc	five	131	mpfi	3.13	3.40	7.0	160	5500	16	22	?

Figure 4: First 10 rows of raw data

3.1.1 Data formatting

The first step was inserting column names as shown in figure 5.

```
# Add column names as they are missing from raw data

columns = ['symboling', 'normalized-losses', 'make', 'fuel-type', 'aspiration', 'num-of-doors', 'body-style',
           'drive-wheels', 'engine-location', 'wheel-base','length', 'width', 'height',
           'curb-weight', 'engine-type', 'num-of-cylinders', 'engine-size','fuel-system',
           'bore', 'stroke', 'compression-ratio', 'horsepower', 'peak-rpm', 'city-mpg',
           'highway-mpg','price']

print('Total number of columns in Automobile dataset =' , len(columns))

dataset.columns = columns
dataset.head()
```

```
Total number of columns in Automobile dataset = 26
```

	symboling	normalized-losses	make	fuel-type	aspiration	num-of-doors	body-style	drive-wheels	engine-location	wheel-base	length	width	height	curb-weight	engine-type	num-of-cylinders
0	3	?	alfa-romero	gas	std	two	convertible	rwd	front	88.6	168.8	64.1	48.8	2548	dohc	four
1	3	?	alfa-romero	gas	std	two	convertible	rwd	front	88.6	168.8	64.1	48.8	2548	dohc	four
2	1	?	alfa-romero	gas	std	two	hatchback	rwd	front	94.5	171.2	65.5	52.4	2823	ohcv	six

Figure 5: Column headers added to data frame

After inserting column names the data type of each column was cross-checked against the data description and corrected where required. This is shown in figure 6.

```
# Ensure proper datatypes
df[['bore','stroke', 'horsepower','peak-rpm']] = df[['bore','stroke','horsepower','peak-rpm']].astype('float')
df['normalized-losses'] = df['normalized-losses'].astype('float')
df['price'] = df['price'].astype('int')
print(df.dtypes)
```

```
symboling              int64
normalized-losses    float64
make                  object
fuel-type             object
aspiration            object
num-of-doors          object
body-style            object
drive-wheels          object
engine-location       object
wheel-base           float64
length               float64
width                float64
height               float64
curb-weight            int64
engine-type           object
num-of-cylinders      object
engine-size            int64
fuel-system           object
bore                 float64
stroke               float64
compression-ratio    float64
horsepower           float64
peak-rpm             float64
city-mpg               int64
highway-mpg            int64
price                  int64
dtype: object
```

Figure 6: Ensuring correct data types

Finally outliers were removed from the dataset as they could represent unique cases or errors which could affect the accuracy of proposed models. A Z-score was used to identify outliers as it gives a measure of how far a datapoint is from the dataset based on how far it is from the mean/average of the dataset [12]. A threshold Z-score of 3 was used [13]. This is illustrated in figure 7.

```
# Check for outliers
from scipy import stats
zscore = np.abs(stats.zscore(df_Enc))
zscore

# remove all data with outliers
df2 = df_Enc
df2 = df2[(zscore <3).all(axis = 1)]
print("Shape of the original dataframe is :", df_Enc.shape)
print("Shape of the clean dataframe is :", df.shape)

del_entries = df_Enc.shape[0] - df2.shape[0]
print("Entries deleted in the dataframe are :",  del_entries)
```

```
Shape of the original dataframe is : (201, 26)
Shape of the clean dataframe is : (201, 26)
Entries deleted in the dataframe are : 47
```

Figure 7: Removing outliers

3.1.2 Dealing with missing values

As stated previously, in handling missing values, data in numerical columns were replaced with the mean while those in categorical columns were replaced with the highest occurring values. This is shown in figure 8.

```
# search through and replace missing values with 'NaN'
dataset.replace('?',np.NAN, inplace= True)

# check for and print total number of missing values
missing = dataset.isnull().sum().sum()
print("Total no of missing values add up to " ,missing)
```

```
Total no of missing values add up to  59
```

```
# Handling missing values in the numerical columns
for col in numerical_col:
    df[col].fillna(df[col].mean(), inplace = True)

# Handling missing values in the categorical columns
for col in categorical_col:
    df[col].fillna(df[col].mode()[0], inplace = True)

# Check that there are no more missing values
print("Total number of  missing vaues in updated dataset is :" , df.isnull().sum().sum())
```

```
Total number of  missing vaues in updated dataset is : 0
```

Figure 8: Handling missing values

3.1.3 Data Encoding

The final data cleaning step was encoding categorical variables using label encoding. This is shown in figure 9.

```
# Encode categorical data using label Encoder
df_Enc = df.copy()
from sklearn.preprocessing import LabelEncoder
Enc = LabelEncoder()
df_Enc[categorical_col] = df_Enc[categorical_col].apply(Enc.fit_transform)

df_Enc.to_csv('LabelEncClean.csv')
df_Enc.head()
```

	symboling	normalized-losses	make	fuel-type	aspiration	num-of-doors	body-style	drive-wheels	engine-location	wheel-base	length	width	height	curb-weight	engine-type	num-of-cylinders	engine-size
0	3	122.0	0	1	0	1	0	2	0	88.6	168.8	64.1	48.8	2548	0	2	130
1	3	122.0	0	1	0	1	0	2	0	88.6	168.8	64.1	48.8	2548	0	2	130
2	1	122.0	0	1	0	1	2	2	0	94.5	171.2	65.5	52.4	2823	4	3	152
3	2	164.0	1	1	0	0	3	1	0	99.8	176.6	66.2	54.3	2337	2	2	109
4	2	164.0	1	1	0	0	3	0	0	99.4	176.6	66.4	54.3	2824	2	1	136

Figure 9: Encoding categorical variables

3.2 Exploratory Data Analysis

3.2.1 Descriptive statistics

Figure 10 gives a brief statistical summary of some columns in the dataset using measures of spread and central tendency such as the mean, standard deviation(std), median(50%), lower quartile(25%), upper quartile(75%), and range which is the difference between the maximum value(max) and minimum value(min).

```
# Summarize basic statistics of the dataframe
df.describe()
```

	symboling	normalized-losses	wheel-base	length	width	height	curb-weight	engine-size	bore	stroke	compression-ratio	horsepower
count	201.000000	201.00000	201.000000	201.000000	201.000000	201.000000	201.000000	201.000000	201.000000	201.000000	201.000000	201.000000
mean	0.840796	122.00000	98.797015	174.200995	65.889055	53.766667	2555.666667	126.875622	3.330711	3.256904	10.164279	103.396985
std	1.254802	31.99625	6.066366	12.322175	2.101471	2.447822	517.296727	41.546834	0.268072	0.316048	4.004965	37.365602
min	-2.000000	65.00000	86.600000	141.100000	60.300000	47.800000	1488.000000	61.000000	2.540000	2.070000	7.000000	48.000000
25%	0.000000	101.00000	94.500000	166.800000	64.100000	52.000000	2169.000000	98.000000	3.150000	3.110000	8.600000	70.000000
50%	1.000000	122.00000	97.000000	173.200000	65.500000	54.100000	2414.000000	120.000000	3.310000	3.290000	9.000000	95.000000
75%	2.000000	137.00000	102.400000	183.500000	66.600000	55.500000	2926.000000	141.000000	3.580000	3.410000	9.400000	116.000000
max	3.000000	256.00000	120.900000	208.100000	72.000000	59.800000	4066.000000	326.000000	3.940000	4.170000	23.000000	262.000000

Figure 10: Statistical summary of the first 12 columns of our data frame

A frequency distribution of some categorical variables using bar graphs is presented in figure 11. It can be inferred that the dataset was dominated with vehicles which ran on gas with around 90% of automobile running on gas. Another interesting observation is that automobiles with rear positioned engines were scarcely represented in this data set. Frequency distributions of automobiles based on their make and body-style is also shown.

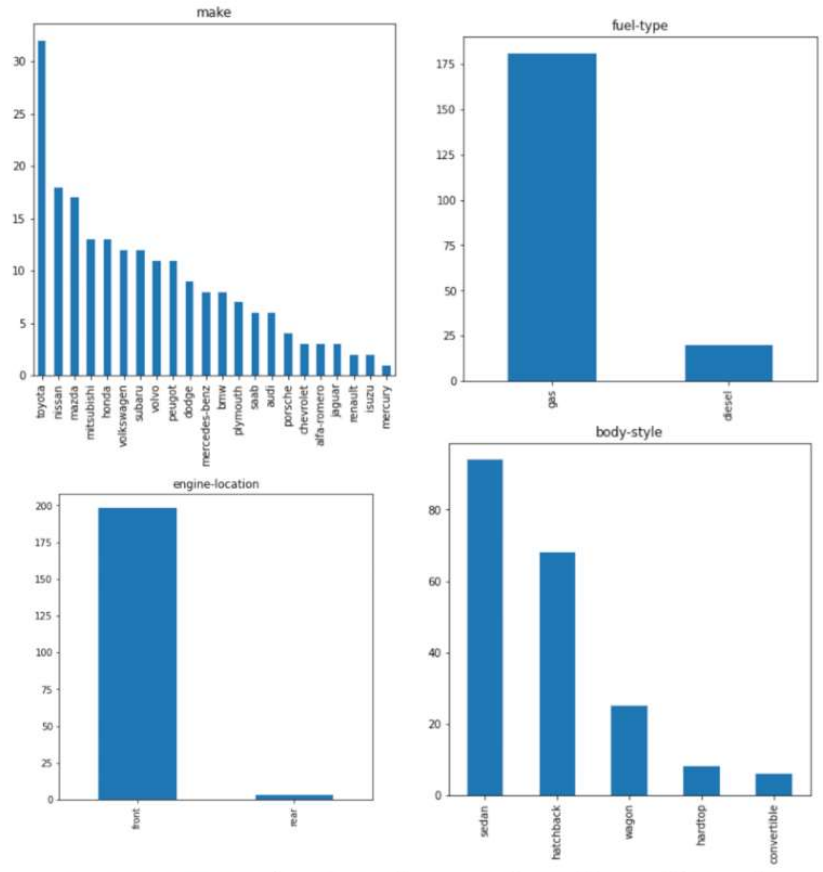

Figure 11: Frequency distribution of categorical variables using bar graphs

By observing the histograms presented in figure 12 it is seen that the distributions of engine-size and horsepower are positively skewed as data is more spread out to the right of the mean. A positive kurtosis can also be inferred seeing as the distributions are heavily tailed.

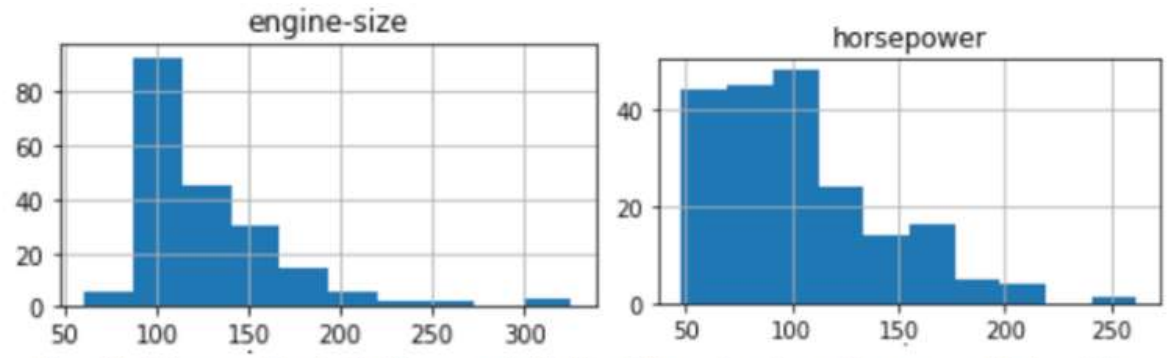

Figure 12: Histograms showing the frequency distribution of the engine-size and horsepower of Automobiles

Figure 13 shows a box plot based on the make of the automobiles. It is immediately seen that the most expensive automobile in this dataset is made by Mercedes-benz while the cheapest vehicle is a Chevrolet. It is also found that BMW vehicles had the greatest price range while Mercury has the lowest price range. The greatest median price is seen in the Jaguar brand. Price outliers are found in the Audi, Dodge, Honda, Mitsubishi, Plymouth, Porsche and Toyota brands. Finally, it is seen that the make of the automobiles have an impact on their price.

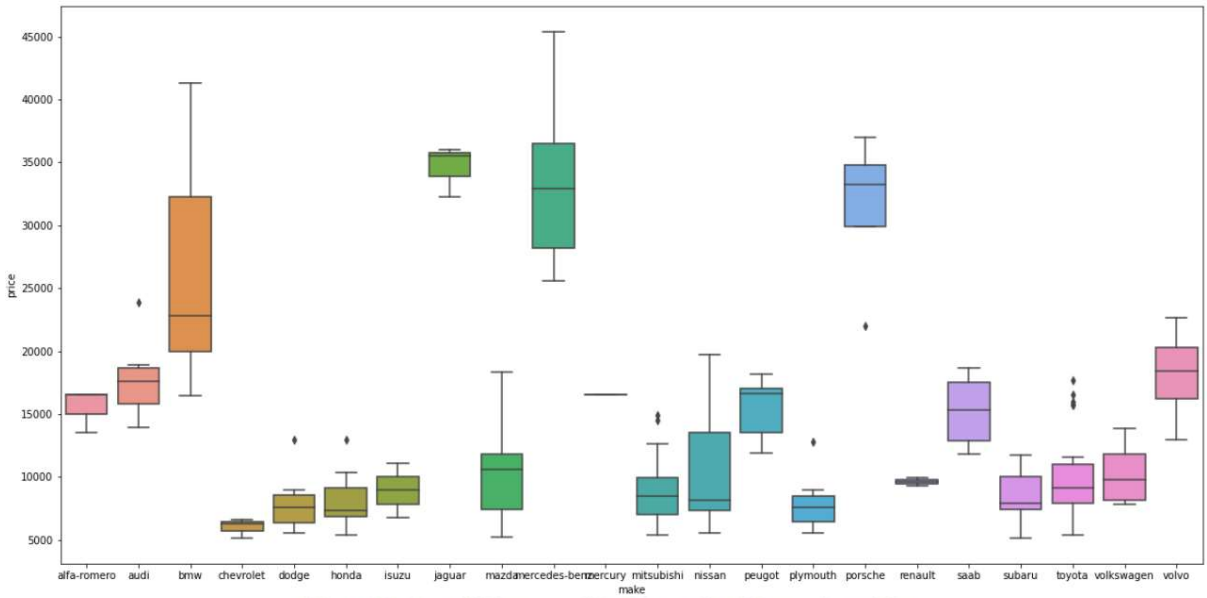

Figure 13: Box plots grouped by the make of the automobiles

The correlation heatmap shown in figure 14 shows the relationship between the numerical variables. The values represented by the heatmap are Spearman correlation coefficients which show the relationship between paired variables [14]. A breakdown of what these values represent is as follows:

- Close to +1: Large positive relationship
- Close to -1: Large negative relationship
- Close to 0: no relationship

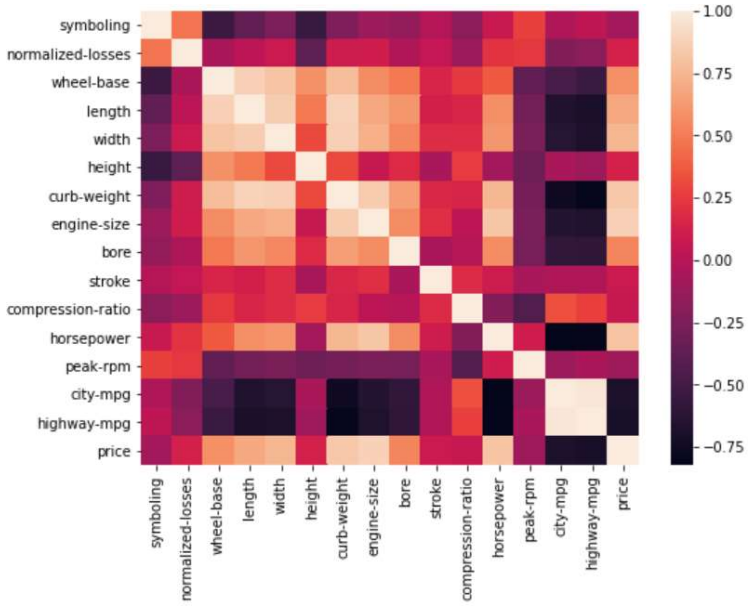

Figure 14: heat map showing correlation between numerical variables

Figure 15 shows the relationship between different variables and price. It can be used to further demonstrate what the values in figure 14 represent. These relationships were used in feature selection.

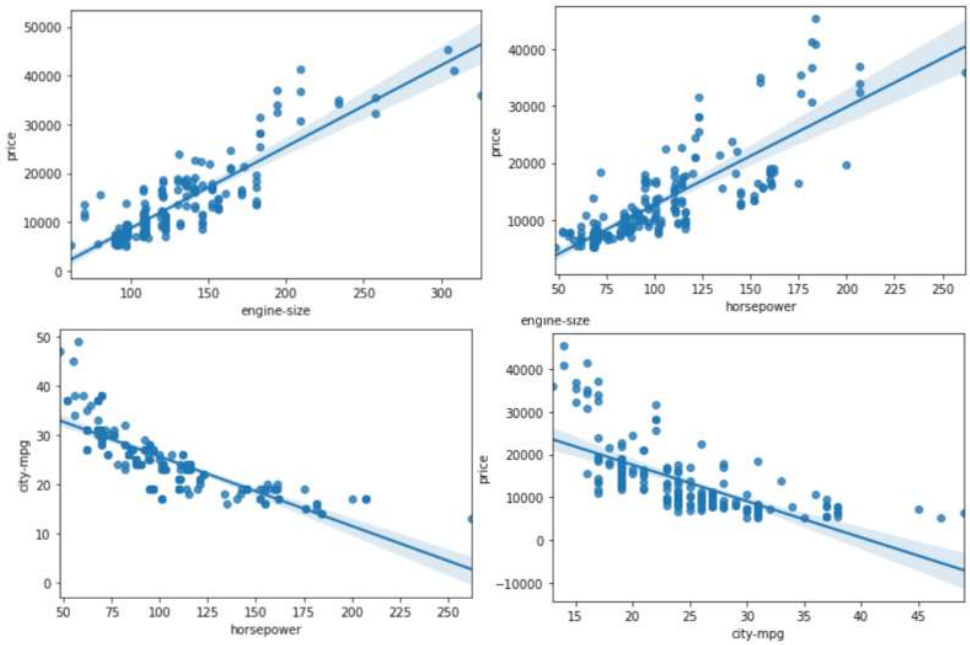

Figure 15: Relationship between different variables

3.2.2 Probability distribution

The principle maximum likelihood estimation of maximum likelihood estimation was used to find the values of mu and sigma that best describe engine size values. The maximum likelihood estimate calculated mu to be 11.14 and sigma as 1.675 . The procedure for this is shown in figure16.

```
# use MLE to find the parameters (mu and sigma) of the normal distribution that would best describe transformed data
from scipy.stats import norm

mu, sigma = norm.fit(sqrt_engSize)
print(mu, sigma)
```

```
11.138608712150505 1.6754157239778287
```

```
# get the probability density function parametrised by our estimated parameters
sqrt_engSize_normal = norm.pdf(sqrt_engSize, loc=mu, scale=sigma)

sqrt_engSize.hist(density=True, alpha=0.5)
plt.plot(sqrt_engSize, sqrt_engSize_normal, color='red')
```

```
[<matplotlib.lines.Line2D at 0x7f6575cce8d0>]
```

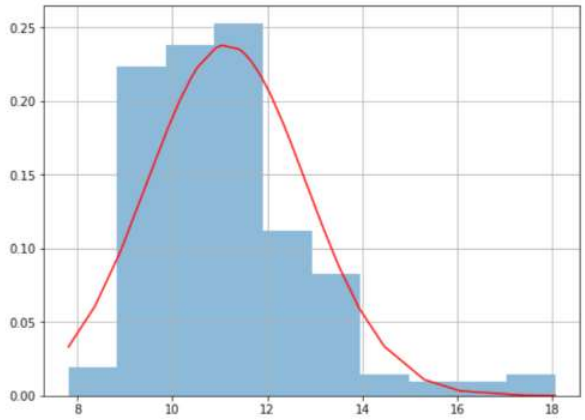

Figure 16: Maximum likelihood estimation of engine size

3.2.3 Hypothesis test

Hypothesis: the number of doors in an automobile plays a part in its price.

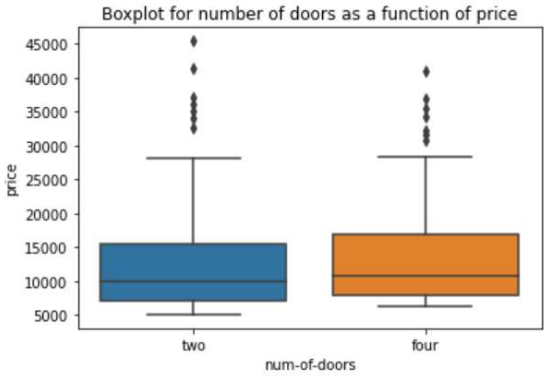

Figure 1: Box plot of showing number of doors as a function of price

To validate this hypothesis, an Anova test was done and by examining the p-value of the results a conclusion is made.

```
grouped_doors = df[['num-of-doors', 'price']]. groupby(['num-of-doors'])
grouped_doors.head()

# ANOVA Test
f_val, p_val = scipy.stats.f_oneway(grouped_doors.get_group('two')['price'], grouped_doors.get_group('four')['price'])

print("ANOVA results for num-of-doors: F =", f_val, "P = ", p_val)
```

```
ANOVA results for num-of-doors: F = 0.3589973711707057 P =  0.5497450927348373
```

Figure 18: results of Anova test

By examining the results of the Anova test in figure 18 a p-value of 0.55 was gotten meaning the null hypothesis was rejected.

3.3 Model Fitting and Selection

3.3.1 Effect of number of features

From figure 19 it is seen that as more features are used in developing a model, its accuracy in predicting new data points in reduced. This is due to a phenomena known as curse of dimensionality. Increasing the number of features used in training a model makes the regression curve used to fit data points too flexible thus resulting in overfitting. This is shown by observing the validation curves when 9 and 7 features were used. There was little or no error in any of the models used in training the dataset in contrast to the huge test errors. It is also seen that when higher order polynomial curves are used, overfitting occurs as training error reduces and test errors increase. A training and validation spilt of 80% and 20% was used in this test [15]. This is discussed further in section 3.3.2.

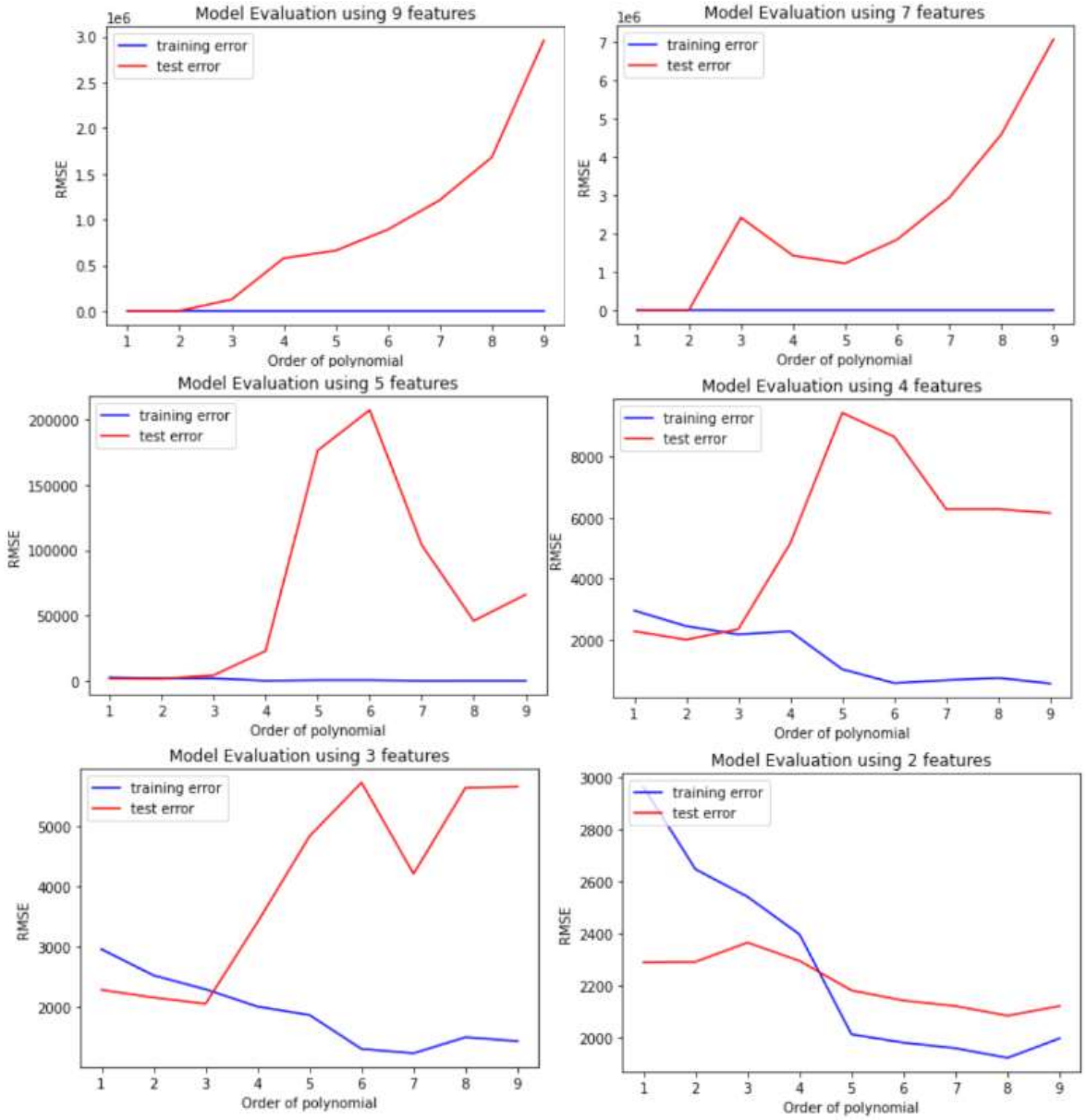

Figure 19: validation curves showing effect of using different number of features

Using bias-variance trade off principles on the results from 19 it was then decided that 5 features gave the best results.

3.3.2 Effect of data size

From table 3 it is seen that when more data is available to train the model, its accuracy increases. The anomalies in the results can be due to cases of overfitting.

		Split (Training set : Test Set)				
Split (Training set : Test Set)	Order	90:10	80:20	70:30	60:40	50:50
Test set RMSE Values	1	1745.8	2003.13	2233.3	2348.52	2352.75
	2	1559.57	1767.07	2265.24	2433.61	4155.2
	3	11556.8	4463.86	5300.61	17990.7	21698.5
	4	12200.6	23063.2	686389	179154	401197
	5	53808.7	176195	596093	351305	354998

Table 3: RMSE for different ratios of training and validation datapoints (Using 5 features)

The two top performing models are seen from the 90:10 and 80:20 splits using 2nd order polynomial curves but model from the 80:20 split was chosen [15]. An extra performance check was then done to verify the suitability of the selected model by looking at its R^2 value as shown in figure 20.

```
poly_orders = [1, 2, 3, 4, 5, 6, 7, 8, 9]


# Training the Polynomial Regression model on the whole dataset
from sklearn.preprocessing import PolynomialFeatures
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error,r2_score

for orders in poly_orders:
    print("The order of the polynomial is", orders)
    poly_reg = PolynomialFeatures(degree = orders)
    x_poly_train = poly_reg.fit_transform(x_train)
    x_poly_test = poly_reg.transform(x_valid)

    # Build MLR model
    regressor = LinearRegression()
    regressor.fit(x_poly_train, y_train)
    y_pred   = regressor.predict(x_poly_test)
    y_hat    = regressor.predict(x_poly_train)
    rmse_train= np.sqrt(mean_squared_error(y_train,y_hat))
    rmse_test = np.sqrt(mean_squared_error(y_valid,y_pred) )

    print( "using an 80:20 split, the RMSE for train set is",  rmse_train )
    print("using an 80:20 split, the R2- score for tRAIN set: %.2f" % r2_score(y_train,y_hat))
    print( "using an 80:20 split, the RMSE for test set is",  rmse_test)
    print("using an 80:20 split, the R2- score for test set: %.2f" % r2_score(y_valid,y_pred))
```

```
The order of the polynomial is 2
using an 80:20 split, the RMSE for train set is 2211.088550997677
using an 80:20 split, the R2- score for tRAIN set: 0.85
using an 80:20 split, the RMSE for test set is 1767.0698154576096
using an 80:20 split, the R2- score for test set: 0.87
```

Figure 20: R^2 score check

4 Conclusion & Future works

It is important to note that finding the best model is an iterative process and although a fairly decent model has been developed in this project, it is just the first iteration and steps can be taken to improve the model accuracy/ performance. These include but are not limited to:

- **Regularization**: Lasso or Ridge regression could have been used to help reduce overfitting and improve the accuracy of the model through regularization by penalizing coefficients in order to maximize the likelihood function [16].
- **Anova testing (categorical variables)**: in feature selection, the 'SelectKBest' function from the sckikit-learn library was used [17]. However, upon further inspection it was found that using this method disregarded the categorical variables which actually influence car price values. Performing hypothesis test on the categorical variables

(see section 3.2.3) would have been a good way to select more useful predictor variables which could have led to better performing models.

- **Normalizing predictor variables**: normalizing is all about ensuring features have equal numerical weightings during feature selection and greatly influence the performance of models [18]. In this project data wasn't normalized and exploring different normalization techniques could have led to an improved model accuracy.
- **K-fold Cross-validation**: in this project, the model is trained using a training set and evaluated based on entirely new data (validation/test set). This is known as the holdout method [19]. A better approach for model evaluation would have be to use the K-fold cross validation where "the data set is divided into k subsets, and the holdout method is repeated k times" [19].

5 References

[1] E. Rahm and H. Do, "Data cleaning: Problems and current approaches," *IEEE Data Eng. Bull.*, vol. 23, no. 4, pp. 3–13, 2000, Accessed: Dec. 16, 2020. [Online]. Available: https://www.researchgate.net/publication/220282831_Data_Cleaning_Problems_and_Current_Approaches.

[2] K. Potdar and C. Pai, "A Comparative Study of Categorical Variable Encoding Techniques for Neural Network Classifiers Blind Aid System View project A Portable Aid System for the Visually Impaired View project A Comparative Study of Categorical Variable Encoding Techniques for Neural Network Classifiers," *Artic. Int. J. Comput. Appl.*, vol. 175, no. 4, pp. 975–8887, 2017, doi: 10.5120/ijca2017915495.

[3] J. T. Behrens, "Principles and Procedures of Exploratory Data Analysis," *Psychol. Methods*, vol. 2, no. 2, pp. 131–160, 1997, doi: 10.1037/1082-989X.2.2.131.

[4] Claudia Angelini, "Regression Analysis - an overview | ScienceDirect Topics," *Encycl. Bioinforma. Comput. Biol. Sci. Direct*, vol. 1, pp. 722--730, 2019, Accessed: Dec. 16, 2020. [Online]. Available: https://www.sciencedirect.com/topics/medicine-and-dentistry/regression-analysis.

[5] E. Ostertagová, "Modelling using polynomial regression," in *Procedia Engineering*, Jan. 2012, vol. 48, pp. 500–506, doi: 10.1016/j.proeng.2012.09.545.

[6] A.Agarwal, "Polynomial Regression. This is my third blog in the Machine… | by Animesh Agarwal | Towards Data Science," 2018. https://towardsdatascience.com/polynomial-regression-bbe8b9d97491 (accessed Dec. 16, 2020).

[7] V.Silva, "Regression Analysis." Accessed: Dec. 16, 2020. [Online]. Available: https://learn.lboro.ac.uk/pluginfile.php/1690653/mod_resource/content/1/Lecture_advanced_regression.pdf.

[8] Priyanka Sinha, "Multivariate Polynomial Regression in Data Mining: Methodology, Problems and Solutions," *Int. J. Sci. Eng. Res.*, vol. 4, no. 12, pp. 962–965, 2013, Accessed: Dec. 16, 2020. [Online]. Available: http://www.ijser.org.

[9] K. Grace-Martin, "Assessing the Fit of Regression Models. The Analysis Factor," *_[http://www.theanalysisfactor.com/assessing-the-fit-of-regression-models/_ consulted on 16/10/2016]*. Accessed: Dec. 16, 2020. [Online]. Available: https://www.theanalysisfactor.com/assessing-the-fit-of-regression-models/.

[10] C.Maklin, "R Squared Interpretation | R Squared Linear Regression | by Cory Maklin |

Towards Data Science," 2019. https://towardsdatascience.com/statistics-for-machine-learning-r-squared-explained-425ddfebf667 (accessed Dec. 16, 2020).
[11] S.Singh, "Understanding the Bias-Variance Tradeoff | by Seema Singh | Towards Data Science," 2018. https://towardsdatascience.com/understanding-the-bias-variance-tradeoff-165e6942b229 (accessed Dec. 16, 2020).
[12] M.Alam, "Z-score for anomaly detection. Small-bites data science | by Mahbubul Alam | Towards Data Science," 2020. https://towardsdatascience.com/z-score-for-anomaly-detection-d98b0006f510 (accessed Dec. 16, 2020).
[13] N.Sharma, "Ways to Detect and Remove the Outliers | by Natasha Sharma | Towards Data Science," 2018. https://towardsdatascience.com/ways-to-detect-and-remove-the-outliers-404d16608dba (accessed Dec. 16, 2020).
[14] "Spearman's correlation," 2018. Accessed: Dec. 16, 2020. [Online]. Available: https://www.statstutor.ac.uk/resources/uploaded/spearmans.pdf.
[15] K. K. Dobbin and R. M. Simon, "Optimally splitting cases for training and testing high dimensional classifiers," *BMC Med. Genomics*, vol. 4, p. 31, 2011, doi: 10.1186/1755-8794-4-31.
[16] J. M. Pereira, M. Basto, and A. F. da Silva, "The Logistic Lasso and Ridge Regression in Predicting Corporate Failure," *Procedia Econ. Financ.*, vol. 39, pp. 634–641, 2016, doi: 10.1016/s2212-5671(16)30310-0.
[17] "Feature selection using SelectKBest | Kaggle," 2018. https://www.kaggle.com/jepsds/feature-selection-using-selectkbest?utm_campaign=News&utm_medium=Community&utm_source=DataCamp.com (accessed Dec. 16, 2020).
[18] D. Singh and B. Singh, "Investigating the impact of data normalization on classification performance," *Appl. Soft Comput. J.*, vol. 97, p. 105524, Dec. 2019, doi: 10.1016/j.asoc.2019.105524.
[19] "Cross Validation." https://www.cs.cmu.edu/~schneide/tut5/node42.html (accessed Dec. 16, 2020).