CAR PRICE PEDICATION USING MACHINE LAERNING

A PROJECT REPORT

In Partial Fulfillment Of The Requirements For The Degree Of

BACHELOR OF COMPUTER APPLICATION

Under the guidance of

MAHENDRA DUTTA

By

PRERNA SHARMA,

SUMIT JAISWAL

BOKARO STEEL CITY COLLEGE

Constituent Unit Of Binod Bihari Mahato Koylanchal University, Dhanbad

In Association With

DECLARATION

We hereby declare that the project work being presented in the project proposal entitled “Car Price Predication using Machine Learning in Python” in partial fulfillment of the requirements for the award of the degree of BACHELOR OF COMPUTER APPLICATION from BOKARO STEEL CITY COLLEGE,BOKARO.

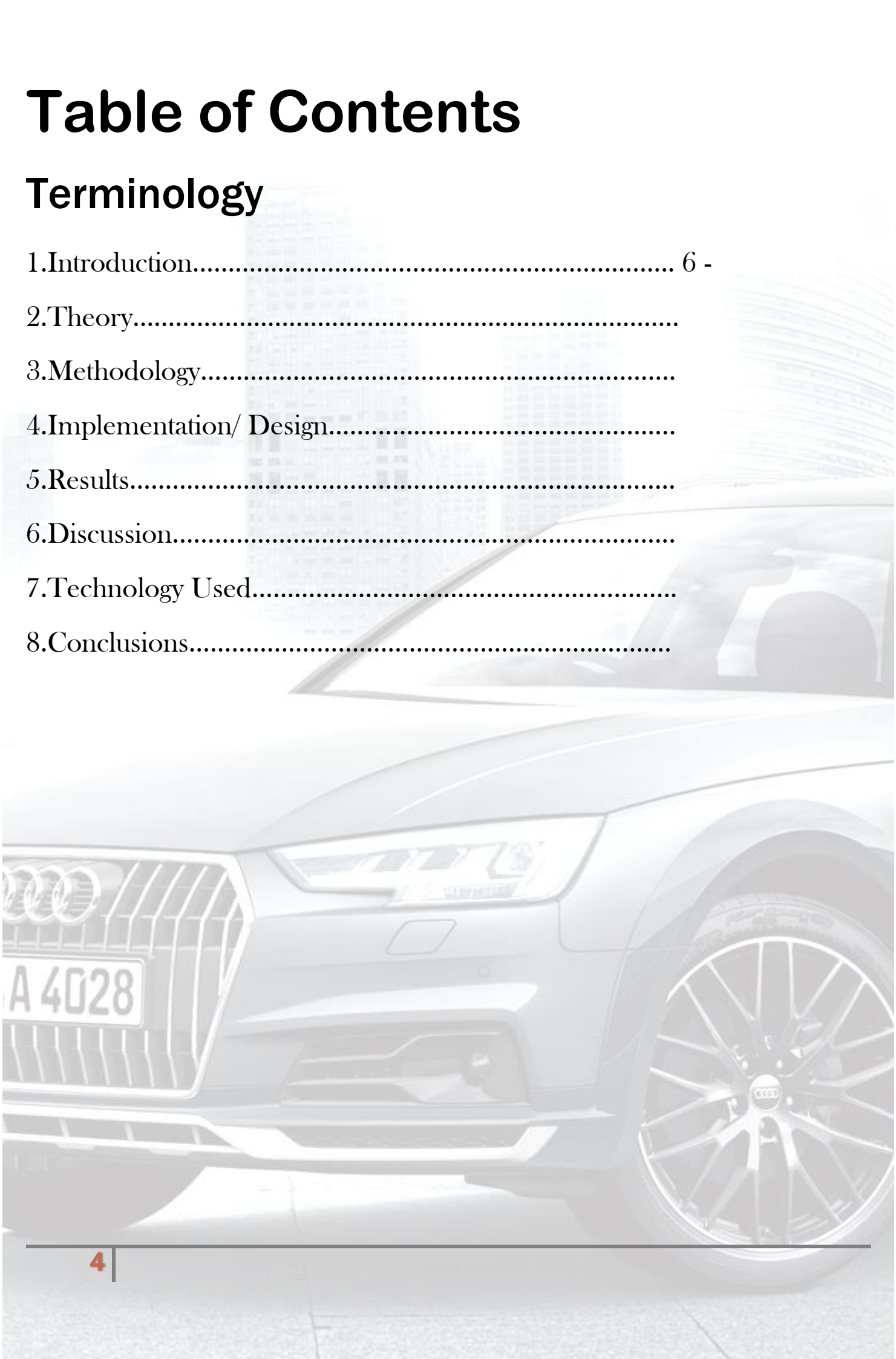
ABSTRACT

Cars of a particular make, model, year, and set of features start out with a price set by the manufacturer. As they age and are resold as used, they are subject to supply-and-demand pricing for their particular set of features, in addition to their unique history. The more this sets them apart from comparable cars, the harder they become to evaluate with traditional methods. Using Machine Learning algorithms to better utilize data on all the less common features of a car can more accurately assess the value of a vehicle. This study compares the performance of Linear Regression, OneHotEncoder ML algorithms in predicting the price of used cars. An important qualification of a price prediction tool is that depreciation can be represented to better utilize past data for current price prediction. The study has been conducted with a large public dataset of used cars. It was also able to represent average depreciation much more closely than the other algorithms, at 13.7% predicted annual geometric depreciation for the dataset independent of vehicle age.

Keywords: Machine Learning, Price Prediction, Used Cars, Regression Analysis, Depreciation.

Table of Contents

Terminology



Terminology

Depreciation	The change in net present value over time
ML	Machine Learning
Revaluation	The change in value or price of an asset that is caused by everything other than aging

INTRODUCTION

1.1 Background and problem motivation
1.2 Overall aim
1.3 Problem Statement
1.4 Research Questions
1.5 Scope
1.6 Outline

1 Introduction

Chapter 1 will serve to give the reader an understanding of the background, problem motivation, and the overall purpose and importance of the work in this report. In addition, it will outline specific scientific goals and questions which this research seeks to answer.

1.1 Background and problem motivation

New cars of a particular make, model, and year all have the same retail price, excluding optional features. This price is set by the manufacturer. Used car, however, are subject to supply-and-demand pricing. Further, used cars have additional attributes that factor into the price. These include the condition, milage, and repair history, which sets cars that may have shared a retail price apart.

The used car market is generally divided into two categories, retail and wholesale. The retail price is the higher of the two prices and is what an individual should expect when buying a car at a dealership. The wholesale price is the lower price which dealers will pay. Whether the dealer has sourced the car from a trade-in, auction, or another dealer, this price is considerably lower to ensure that the dealer will make a profit on the vehicle. Prices for peer-to-peer car sales generally lie in-between the retail and wholesale price points. Because there is no "middle-man" in peer-to-peer transactions, there is only a single price point, rather than two. A difficulty in peer-to-peer transactions is for both parties to agree on a fair price. There are many tools which provide an approximation, but do not factor in the particularities of the car into the price. Car markets are to some extent local and therefore location also affects the price. There is therefore a need for a valuation method which can make use of more of the features particular to each car, and extract information from all other previous sales of cars with shared features.
Machine learning (ML) is a subfield of Artificial Intelligence (AI) that works with algorithms and technologies to make useful inferences from

data. Machine learning algorithms are well suited to problems entailing large amounts of data which would not be possible to process without such algorithms. ML works algorithmically rather than mathematically and permit a machine to "learn" and adapt its predictions to best fit the data it has trained on.

1.2 Overall aim

The purpose of this thesis is to evaluate several different machine learning models for used car price prediction and draw conclusions about how they behave. This will deepen the knowledge of machine learning applied to car valuations and other similar price prediction problems.

1.3 Problem statement

For the purposes of car valuation, popular guides tend not to use machine learning. Instead, they source data from local sales and average the prices of many similar cars. This method works well if you have a common car with a common set of features. The condition of the car is judged very roughly, typically on a scale of one to three. Cars that are "unusual" are therefore hard to evaluate. Effectively, no inferences are drawn from similar cars but from a different make and model, whereas with machine learning, the entirety of the dataset and its features are used to train the model predictions. Using machine learning is a solution to the problem of utilization of all the data and will assist in utilizing all the features of a car to make valuations.

New cars of a particular make, model, location, and feature selection are identical in condition, function, and price. When new cars are sold for the first time they are then classified as used cars. As an asset ages, its price changes because it declines in efficiency in the current and in all future periods. Depreciation reflects the change in net present value over

time. Revaluation, on the other hand, is the change in value or price of an asset that is caused by everything other than aging. This includes price changes due to inflation, obsolescence, and any other change not associated with aging . Used cars are subject to depreciation and revaluation. Depreciation can be used as an umbrella term for both of these, and the rest of this report will follow that convention when referring to the loss of value over time. Revaluation plays a part in the depreciation of cars based on the features that they have. Power hungry cars will be less sought after when the price of gasoline is high, for example. A car with the same make, model, year, and geographic region, but this a larger engine than a different car should command a different value at different times.

In addition to the age of the car and the revaluation of its features, used cars have a unique service history that develops over time. Parts will become worn with time and miles driven (mileage). What is replaced, when it is replaced, and by whom, are all to be considered as it relates to the current working condition of the car and its desirability on the market. The particularities are difficult to account for in traditional price-setting models, as it is a major differentiator in vehicles. Generally, it is summarized in the "condition" of the car. The value of repairs or custom modifications to the car are recognized only if they noticeably improve the overall condition of the car.

Using machine learning to better utilize data on all the less common features of a car can more accurately predict the value of a vehicle. This is a clear benefit to consumers, especially those who themselves cannot ascertain the value of the vehicle that they are buying or selling and must rely on a tool. A tool that is more tailored to the non-standard features of the car can provide a more accurate price and make the market fairer for all participants.

There are several machine learning regression models that can be applied to price prediction. This work will investigate which one offers the best performance according to several criteria. The nature of machine learning is to train on past data to predict unseen data. Applied

to price prediction of cars, the data is sourced from past sales while the predictions are for the present value of cars. Therefore, a criterion for the selection of a machine learning model it remains accurate in its predictions for future years, not included in the data set.

1.4 Research Questions

The research questions that this study will answer are:

(1) Which ML model and parameters gives the best overall accuracy in making price predictions for used cars?

(2) Which ML model can most accurately assess the depreciation of a car over time?

(3) Which ML model demonstrates the best potential for development of a consumer tool for evaluating used cars or a particular subset of used cars?

These are chosen to satisfy the scientific goals. Research Question 1 will determine which of several algorithms gives the best performance in a verifiable way. Research

Question 2 will then examine and compare the behaviour of the algorithms to suggest which can best assess depreciation over time, if any. Finally, Research

Question 3 will combine the knowledge gained from the previous questions and show which of the algorithms in aggregate demonstrate the best potential for building a consumer tool for price prediction of used cars.

1.5 Scope

This work will focus on answering the research questions. They all entail a comparison of different ML algorithms for price prediction. This will be accomplished by sourcing and preparing a dataset on which all the algorithms can be trained on and compared fairly. The algorithms selected must therefore be similar enough for the same dataset to be used for all of them. This also means that no large optimization efforts on the dataset will be made to boost the performance, if these changes do not benefit the other models. Maximizing price prediction performance of any one algorithm in ways that do not offer better comparisons is outside the scope of this work

.

1.6 Outline

Chapter 2 will explain relevant theory and related work to give introductory knowledge of the concepts and related research. Chapter 3 will go over project milestones, motivations for these milestones being chosen, and how they will be accomplished. Chapter 4 will describe the implementation of the research to fulfill the project milestones. Chapter 5 will present the results of the measurements resulting from the implementation with tables and charts. Chapter 6 will discuss the results, the achievement of project milestones, and the societal and ethical implications that this work could have. Chapter 7 will present the conclusions that can be drawn from this work, definitively answer the research questions, and explore the potential for future research.

1.7 Objective

- To build a supervised machine learning model for forecasting value of a vehicle based on multiple attributes
- The system that is being built must be feature based i.e. feature wise prediction must be possible.

- Providing graphical comparisons to provide a better view.

1.8 Motivation

The automotive industry is composed of a few top global multinational players and several retailers. The multinational players are mainly manufacturers by trade whereas the retail market features players who deal in both new and used vehicles. The used car market has demonstrated a significant growth in value contributing to the larger share of the overall market. The used car market in India accounts for nearly 3.4 million vehicles per year.

1.9 Features

There will be majorly two features provided in the project note that this will be not :

- Re-sale platform: A centralized platform for car resale that will predict prices.
- Feature selection: Feature-based search and prediction.

Section I contains the introduction of our module, then objective, motivation and features of our model, Section II contains Literature Review, Section III contain the various technologies in machine learning, Section IV explains the methodology, section V describes the results and discussion, Section VI contains the conclusion and future work.

Theory

Theory

This chapter will explain relevant theory and related work. This includes concepts related to regression learning, all metrics used for the performance measurement of the models, and related research in the field of machine learning applied to price prediction.

2.1 Regression Machine Learning

Regression analysis is a fundamental concept in the field of machine learning. It is a type of supervised machine learning wherein the model is trained with both input features and output labels. It helps in establishing a relationship among the variables by estimating how they in combination arrive at an estimated output in the form of a continuous variable rather than a discrete label. The input variables are called independent variables and correspond to features in the dataset, while the output variable is called the dependent variable. The simplest of these algorithms is linear regression which assumes that the relationship of each variable is linearly proportional to the output.

2.2 Overfitting

Overfitting a model is a condition where a statistical model begins to describe the random error in the data rather than the relationships between variables. This condition can affect all supervised machine learning models. In the case of regression models, overfitting can occur when there many terms for the number of observations. This leads to the regression coefficients representing the noise rather than the actual relationships in the data. Much better prediction results on the training data is an indication of overfitting.

2.3 Linear Regression

Linear Regression is a technique to estimate the linear relationship between each of a number of independent variables and a dependent variable. Linear Regression fits a linear model with coefficients w = (w1, ..., wp) to minimize the residual sum of squares between the observed targets in the dataset, and the targets predicted by the linear approximation.

2.4 Evaluation Metrics

RMSE (root mean squared error) is a commonly used measures for evaluating the quality of predictions in regression ML. It shows how far predictions fall from measured true values using Euclidean distance. Since the error is squared in this method, a few unusually large prediction errors will skew the metric higher than more evenly distributed errors. A lower value indicates higher prediction accuracy. The equation below shows the formula for calculating RMSE, where "ŷ" is the predicted value, and "y" is the actual value.

$$RMSE = \sqrt{\sum_{i=1}^{n} \frac{(\hat{y}_i - y_i)^2}{n}}$$

R-squared is a another commonly used measure for evaluating the predictions in regression ML. It is also termed the standardized version of MSE (the squared value of RSME) because it is unaffected in magnitude by the scaling of the values in the dataset. That means that the absolute magnitude of the errors doesn't affect the R-squared measure, only the proportion of those errors to the average value. Like RMSE, a few uncommonly large values disproportionately affect the value. R-squared values are always in the range of 0-1, with one being no error

(the predictions of the ML model perfectly fit the actual data). Values closer to one means that the ML gives better predictions.

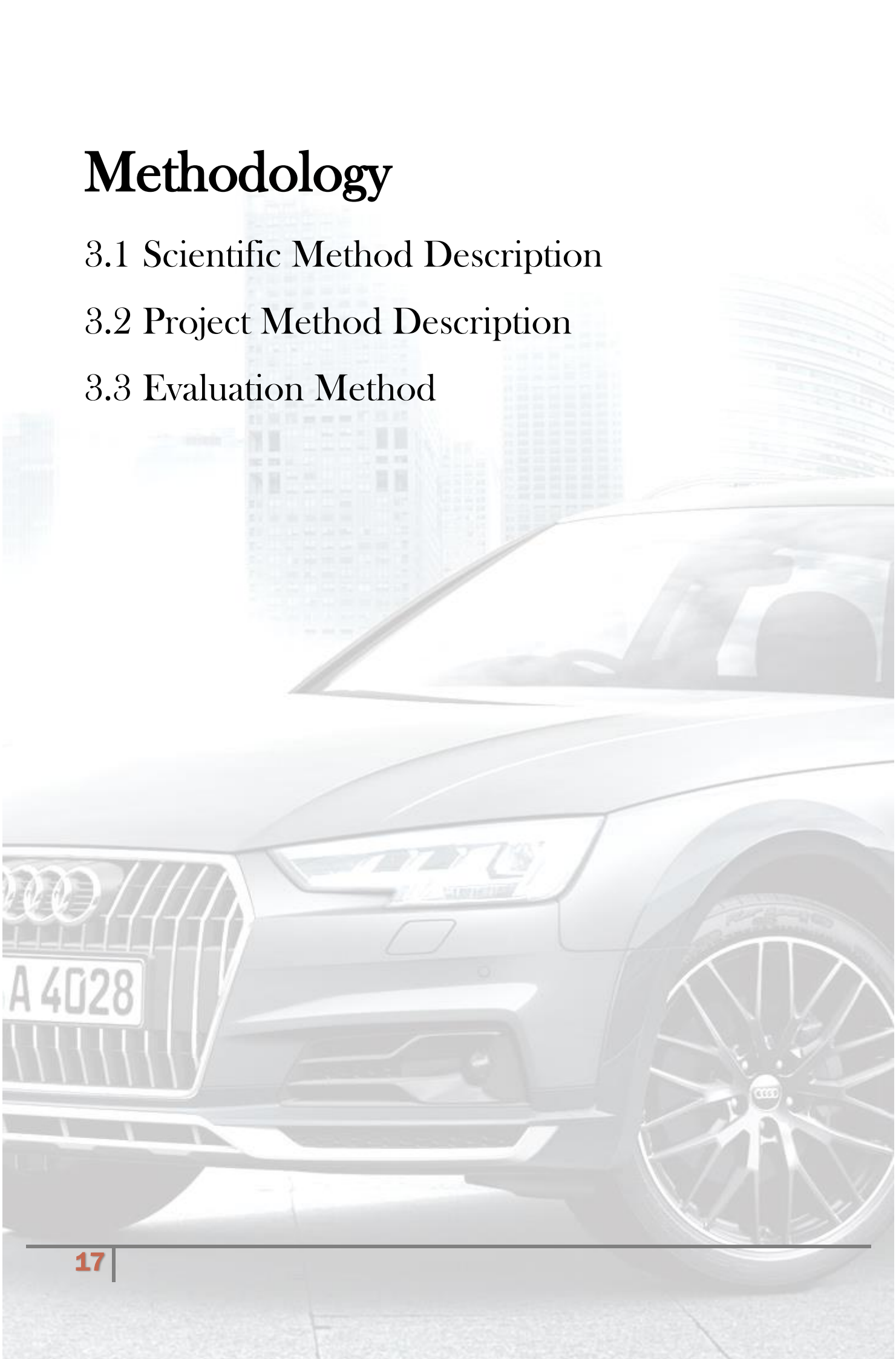
MAPE (mean absolute percentage error) is another measure for evaluating the accuracy of predictions. It is calculated by taking the absolute value of the percentage error between the actual value and the predicted value for each element. The values are then averaged to get the MAPE. This estimated error is not squared unlike for the RMSE and R-squared metrics. The individual errors before averaging depend on the proportion of the magnitude of the actual value to the magnitude of the predicted value. Perfect predictions will give a MAPE of zero, and a lower value signifies better predictions.

Methodology

3.1 Scientific Method Description

3.2 Project Method Description

3.3 Evaluation Method



Methodology

The chapter will present the method followed in performing the research.

3.1 Scientific method description

This work will use a quantitative method to achieve the scientific goals. The evaluation of models will be done by collecting and comparing various performance metrics for each of the machine learning algorithms to be tested in this work.

Machine learning models need a large amount of data to train on. The first step in performing this study is to source a sufficiently large and reliable dataset. There are several criteria for such a dataset. It must be large enough, include sufficiently many relevant features, have very few null values for those features, have reliable values, and must be distributed over several years.

To ensure the highest possible accuracy for the various models, a resultdriven iterative process including data cleaning, model training, and model testing will be used to refine the models.

3.2 Project method description

From the project statement and the scientific goals, the following project milestones were produced:

1. Study previous research into price prediction models with regression and identify the most used and most viable algorithms for the task.
2. Source an appropriate dataset of peer-to-peer car sales to use in the training of the models.
3. Remove any missing or outlier values from the dataset and make appropriate normalizations to the data.
4. Instantiate one of each of the models and make appropriate normalizations to the dataset to boost the performance of each model.
5. Measure the efficacy of the models and compare the performances.

6. Compare the model's predicted depreciation by simulating the aging of the vehicles in the dataset.

The first project milestone is to use previous research on price prediction and identify the most used and viable ML regression models. This milestone is necessary to gain an understanding of which ML models are the best candidates for developing a price prediction tool, and therefore the most relevant to study in this research.

The second project milestone is to source an appropriate dataset of peer-to-peer car sales for the model to be trained on. This milestone was chosen in order to have a sufficiently large and complete collection of car sales data for the models to provide accurate predictions and therefore meaningful comparison of them. The dataset must also span several years for the model to be able to infer prices in years future to the dataset. Keeping these criteria in mind, there are several publicly available datasets to be had from sites such as Kaggle.

The third project milestone is to remove missing and outlier values from the dataset. This milestone was chosen because many ML models are sensitive to outlier values. Datasets that are sourced by means of webscraping can often have missing, incomplete, or unreasonable values. These need to be identified and removed. A caveat to this is that removing too many infrequently occurring values can reduce the size of the dataset, which will negatively affect the prediction accuracy of the model. Removing infrequent values will also limit the potential of the model to predict similar values. For example, removing rare car makes and models means that the scope of the ML model will not include those makes and models.

The third project milestone is to remove missing and outlier values from the dataset. This milestone was chosen because many ML models are sensitive to outlier values. Datasets that are sourced by means of webscraping can often have missing, incomplete, or unreasonable values. These need to be identified and removed. A caveat to this is that removing

too many infrequently occurring values can reduce the size of the dataset, which will negatively affect the prediction accuracy of the model. Removing infrequent values will also limit the potential of the model to predict similar values. For example, removing rare car makes and models means that the scope of the ML model will not include those makes and models.

The fifth project milestone is to measure the efficacy of the models and compare the performances of each. The metrics used for this will be RSME, and R-squared. Since the models are trained and tested on the same data, these metrics can be directly compared. A consumer is likely to consider the average error in the price prediction in deciding how accurate the price prediction for their car valuation.

The sixth and final project milestone is to compare the prediction the model's predicted depreciation by simulating the aging of the vehicles in the dataset and measuring the average percentage change in the new predictions compared to the original. This milestone was chosen to add another evaluation criteria for deciding which of the models are most suited for price prediction. Being able to infer values future to the dataset helps to prevent obsolescence of the model. Used cars are a depreciating asset and the model should reflect that. Furthermore, newer vehicles in aggregate will depreciate faster than older ones. To achieve this milestone, we will simulate the aging of the vehicles in the dataset by incrementing the features year sold and Year. The feature Mileage must also be increased by the average miles that are driven in a year. According to the Federal Highway Administration, American cars are driven an average of 14,263 miles per year. Thus, for each vehicle in the dataset, these three values will be increased and fed into the model to generate predictions for the aging of the vehicles. Thereafter, the percentage change in the predicted price will be recorded for each of the models. Previous studies show that geometric depreciation is a good approximation of real vehicle depreciation in developed countries for used cars. The annual depreciation rate for this distribution was found to be in the range of 15-31% in one such study. Geometric depreciation means that the percentage decrease in value each year is constant. By measuring the predicted depreciation for cars with

different ages, it can be shown whether the models approximate geometric depreciation. The expected result assuming geometric depreciation is that the cars, regardless of their ages, approximately lose the same percentage of their value each year. Additionally, this value can be expected to be in the range of 15-31%. A caveat for this value is that the dataset is not necessarily representative of the population of cars. Cars are not worth anything are not sold, and not represented in the dataset. Additionally, some cars can increase in value and subsets of cars that are sold more often will be overrepresented.

3.3 Evaluation method

This work will be evaluated by how well the results derived from the method description are able to produce satisfactory answers to the research questions. The method should be able to produce conclusive answers to the first two research questions. It will be possible to train and test the Machine Learning models chosen, so long as they are viable for regression analysis. Through the creation of dummy variables, the categorical features in the dataset can be converted to continuous variables to be used as inputs in the regression models. This can however lead to a loss of information and reduced performance of the various ML models to different degrees. This work is contingent on the ability to fairly compare the performances of the algorithms according to several criteria, but not necessarily on achieving a very high performance for any of the algorithms, although if they do all have to achieve performance results that show that they were implemented successfully and can thus be fairly compared.

Implementation/Design

4.1 Data Sets

4.1.1 Kaggle Dataset

4.2 Data Cleaning and Normalization

4.3 Machine Learning Algorithms

4.4 Inference

4.5 Measurements

4.51 Training and Testing Accuracy Comparison

4.5.2 Inferred Price Plots

4.5.3 Measuring Despreciation

Implementation/Design

This chapter will describe the process of implementing the system. The implementation was divided into five parts titled Data Set, Data Cleaning and Normalization, Machine Learning Algorithms, Measurements, and Inference. Each of these parts are explained in their own sections as part of this chapter and are shown in the UML diagram below. The high level component of the UML diagram without a dedicated section of this chapter, Simulated Aging, is detailed in the measurement section. The entire implementation was written in Python3 in the . The librarie vs codes utilized are pandas, sklearn (sci-kit learn), NumPy, re (regular expressions), matplotlib, and seaborn. (See Appendix C for the entire source code).

4.1 Data Sets

4.1.1 Kaggle datasets

The dataset was sourced from Kaggle and includes 816 car listings from the years 2001 to 2019 from all areas in the India. It is available publicly. It includes all types of road-going consumer UML Component Diagram, Implementation Overview 22 vehicles, such as vans, pickup-trucks, and cars (See Appendix B for the published data set). This dataset shows listings of used cars and not necessarily the final sales price. The dataset does not have duplicate listings of the same car however, with the previous listings being removed as the sale was most likely unsuccessful. Therefore, the listed price may be somewhat higher than actual sales price and not reflective of the actual values of the cars. This error is consistent across the dataset (including the entries that will be used for testing) however and should not significantly affect measured model performance.

Table 1: Dataset features

FEATURE	EXPLANATION
ID	
PRICESOLD	The price at which the vehicle was listed at
YEARSOLD	The calendar year when the vehicle was sold
ZIPCODE	The zip code where the car was listed
MILEAGE	
MAKE	
MODEL	
YEAR	The production year of the vehicle
TRIM	The version/configuration of the model
ENGINE	The engine type/specification (including displacement in liters)
BODYTYPE	
NUMCYLINDERS	The number of cylinders of the engine
DRIVETYPE	The type of drivetrain (RWD, AWD, FWD, 4WD)

4.2 Data Cleaning and Normalization

The first step in cleaning the dataset provided from Kaggle was to identify variables which will not be useful for training the models. This includes features which are not correlated with price, have too many discrete values to draw inferences from, or have too many missing values. The features that were identified to be dropped from the dataset were: ID, zipcode, and Trim.

The next step is identifying and removing outliers for the ten remaining features. Keeping in mind the distribution of the data and the negative effect of removing too many values, appropriate minimum and maximum values were set for each feature to remove rows in the dataset which were extreme in any feature category. This was performed for the features pricesold, Mileage, and Year. These were chosen somewhat arbitrarily but with the purpose of removing an appropriate percentage of uncommonly occurring extreme values in the dataset. This increases the performance of the models

Table 2: Removal of Outliers

OUTLIER CATEGORY	COUNT BEFORE REMOVAL	COUNT REMOVED	MINIMUM VALUE	MAXIMUM VALUE
PRICE	122,144	11,743	1000	50,000
MILEAGE	110,401	13,950	1000	200,000
YEAR (AGE OF VEHICLE)	96,451	15,902	0	50
OTHERS	80,549	38,562	NA	NA

The remaining features were categorical variables. Since all variables provided into regression models must be continuous, strategies for making these variables continuous must be employed. If the feature is numeric in nature, then it can be made continuous. The Engine feature was inconsistently in the dataset but included the displacement which is numeric. Some entries also included the number of cylinders, while 24 omitting this from the NumCylinders feature. These could be extracted with regular expressions. The engine displacement was kept in the Engine feature rather than the engine type, which could not easily be made to be numeric.

Another strategy to convert categorical variables to numeric is creating dummy variables. If a feature assumes relatively few different values, the feature can be converted into several dummy variables where each unique categorical value becomes a different continuous variable. The values of each of these variables can only ever be zero or one. The creation of dummy variables was applied to the features Make, Model, BodyType, DriveType, and NumCylinders.

4.3 Machine Learning Algorithm

The data, after being cleaned and normalized, is split into training and test data using a randomized 80-20 split. This is to ensure that the data used for testing does not contain any of the data used for training. Thus 20% of the data is reserved for testing purposes . The training dataset was used to train the four price prediction ML models chosen: Linear Regression .All machine learning algorithms used in this report were imported from the sklearn library. Some models were provided input parameters to implement. The motivations for the choice of input parameters are explained in this section for the models that require them.

It was implemented with default parameters and random_state=0. The random state is necessary because it is a stochastic process that takes a seed value to begin. When testing different values, there was no noticeable performance increase. The random_state throughout training was therefore set consistently to 0, minimizing stochastic behavior resulting from varying the random_state.

4.4 Inference

Inference involves using the subset of the data that was reserved for testing (20%) to predict the price based on the features. This step was performed after the dataset was cleaned and normalized, and the models were optimized. The dataset was re-split, models were retrained, and inferences retaken a total of five times. This produced five separate inferences with the same parameters to be able to produce an average for the measurements. The inferences produced varies slightly each time as a result of the randomized 80-20 training-testing data split. Each model produced inferences from the same testing subset in every iteration. To judge overfitting, they were also tested on the training subset of the data.

Much better prediction results on the training data is an indication of overfitting.

For the Kaggle subset (cars sales from 2019), an inference was performed once for each model, on the entire dataset. This was done subsequent to simulating the aging of all the vehicles as described in Milestone 6 (see Project Method Description).

4.5 Measurements

The measurements taken for this study are described in this section. All of the measurements are taken from the same inference data for each model and using the formulas for the various metrics.

4.5.1 Training and Testing Accuracy Comparison

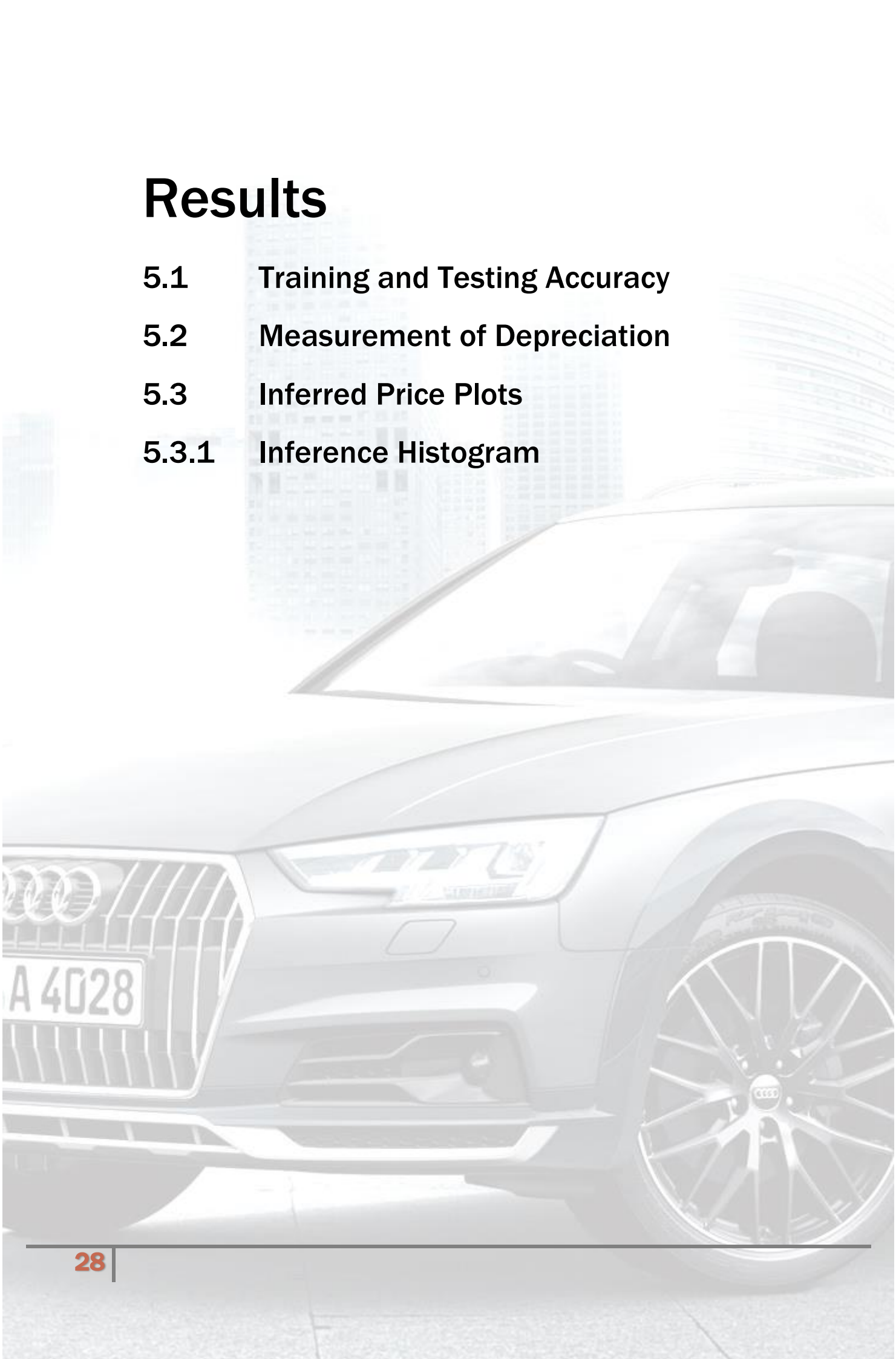
The performance metrics that were taken for machine learning algorithm is RMSE. These measurements were taken for both the training and testing inferences and averaged across all five iterations of inferences taken to produce a table of metrics.

4.5.2 Inferred Price Plots

For the first iteration of inferences (both training and testing), scatter plots were created for each algorithm where each data point is the actual price, plotted against the inferred price. A line of best was then calculated and drawn through these points as well as a line showing the actual values, y=x. If the algorithms do not demonstrate any systematic error, the line of best fit should match this line. See Appendix A for these scatter plots.

4.5.3 Measuring Depreciation

Results



Results

This chapter will present the results of all measurements performed. This includes tables demonstrating the training and testing accuracy of the models, the magnitude of coefficients for the models that utilize coefficients, depreciation measurements, an inference histogram and inference scatterplots. The results are presented with the use of tables and graphs.

5.1 Training and Testing Accuracy

Table 3 shows the prediction accuracy results for both training and testing for each of the four ML algorithms. The predictions accuracy is measured by taking the average value of five iterations.

Table 3: Performance Metrics Training and Testing Data

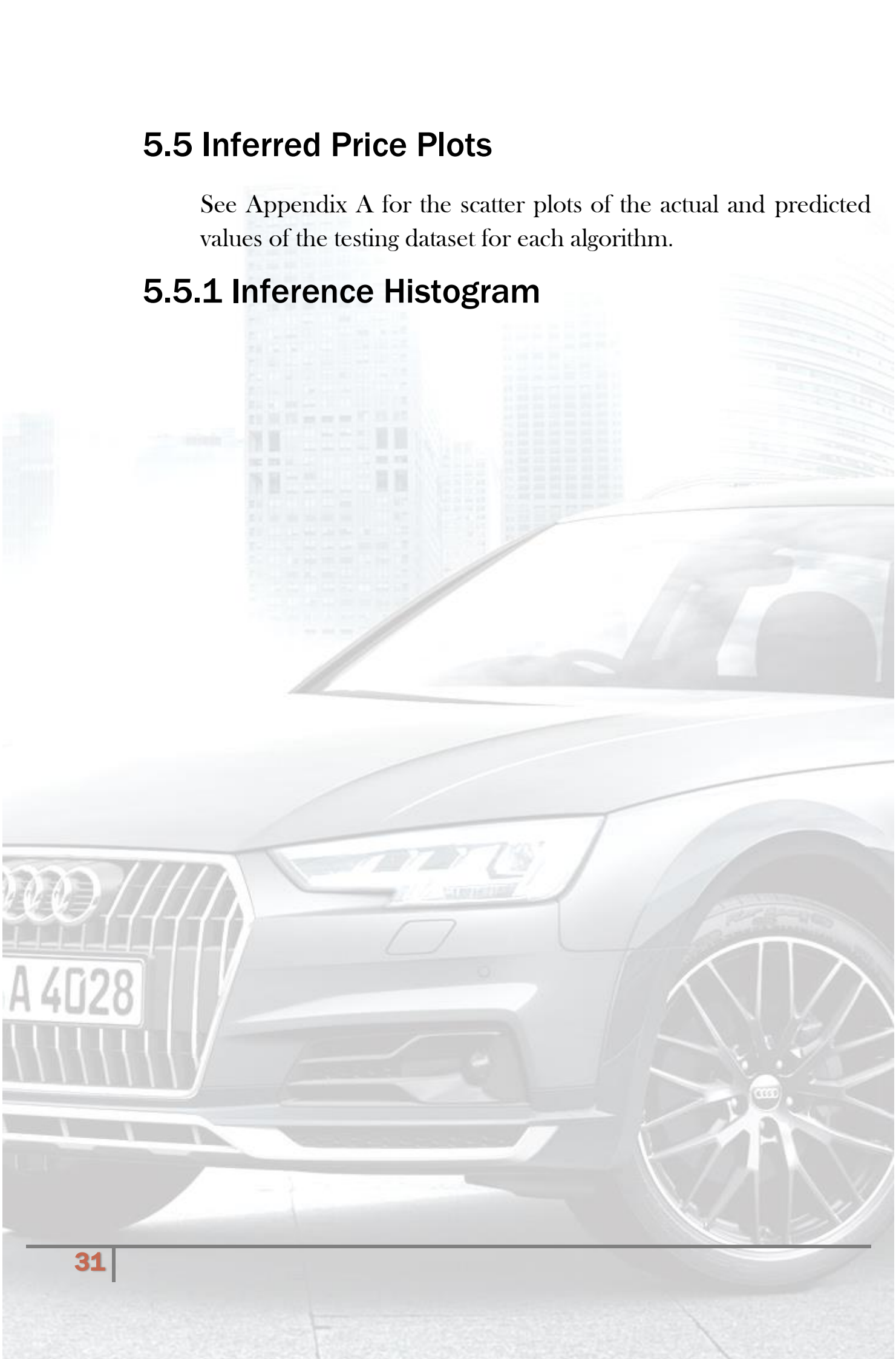
5.2 Measurement of Depreciation

The results of the measurement of depreciation are shown in the chart below. The chart below displays the percentage decrease in price when simulating the aging of the vehicles in the testing dataset, for cars of two different age categories approximately 10 years apart. This information is plotted for each algorithm to compare.

5.5 Inferred Price Plots

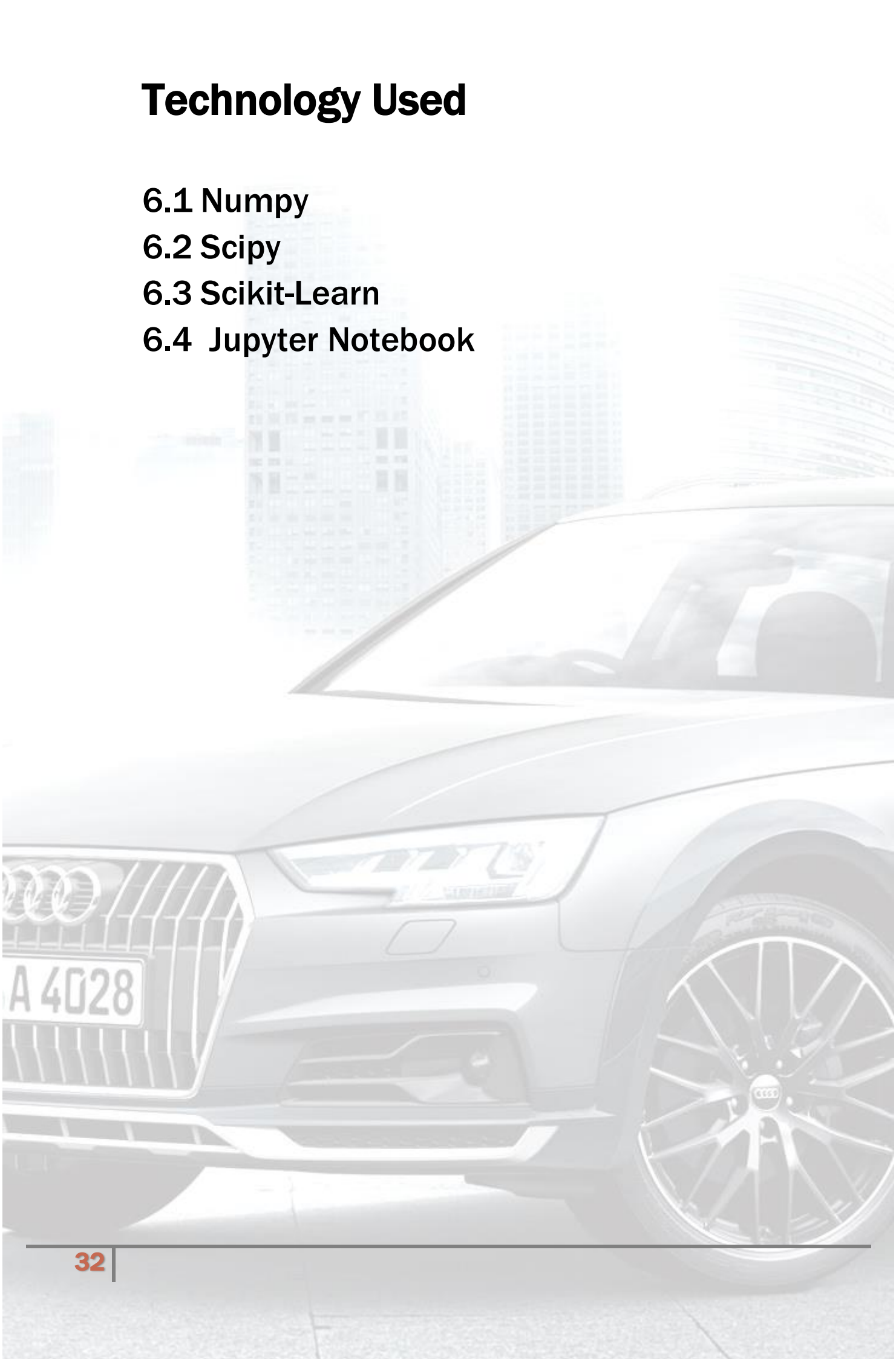
See Appendix A for the scatter plots of the actual and predicted values of the testing dataset for each algorithm.

5.5.1 Inference Histogram



Technology Used

6.1 Numpy
6.2 Scipy
6.3 Scikit-Learn
6.4 Jupyter Notebook



TECHNOLOGY USED

Python was the major technology used for the implementation of machine learning concepts the reason being that there are numerous inbuilt methods in the form of packaged libraries present in python. Following are prominent libraries/tools we used in our project.

6.1 NUMPY

NumPy is a general-purpose array-processing package[1]. it provides a high-performance multidimensional array object and tools for working with these arrays. It is the fundamental package for scientific computing with Python. Besides its obvious scientific uses, NumPy can also be used as an efficient multi-dimensional container of generic data. Arbitrary data-types can be defined using Numpy which allows NumPy to seamlessly and speedily integrate with a wide variety of databases.

6.2 SCIPY

SciPy is a free and open-source Python library used for scientific computing and technical computing. SciPy contains modules for optimization, linear algebra, integration, interpolation, special functions, FFT, signal and image processing, ODE solvers and other tasks common in science and engineering. SciPy builds on the NumPy array object and is part of the NumPy stack which includes tools like Matplotlib, pandas, and SymPy, and an expanding set of scientific computing libraries. This NumPy stack has similar users to other applications such as MATLAB, GNU Octave, and Scilab. The NumPy stack is also sometimes referred to as the SciPy stack. The SciPy library is currently distributed under the BSD license, and its development is sponsored and supported by an open community of developers. It is also

supported by NumFOCUS, a community foundation for supporting reproducible and accessible science.

6.3 SCIKIT-LEARN

Scikit-learn provides a range of supervised and unsupervised learning algorithms via a consistent interface in Python. It is licensed under a permissive simplified BSD license and is distributed under many Linux distributions, encouraging academic and commercial use. The library is built.

6.4 JUPYTER NOTEBOOK

The Jupyter Notebook is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations, and narrative text. It includes data cleaning and transformation, numerical simulation, statistical modeling, data visualization, machine learning, and much more.

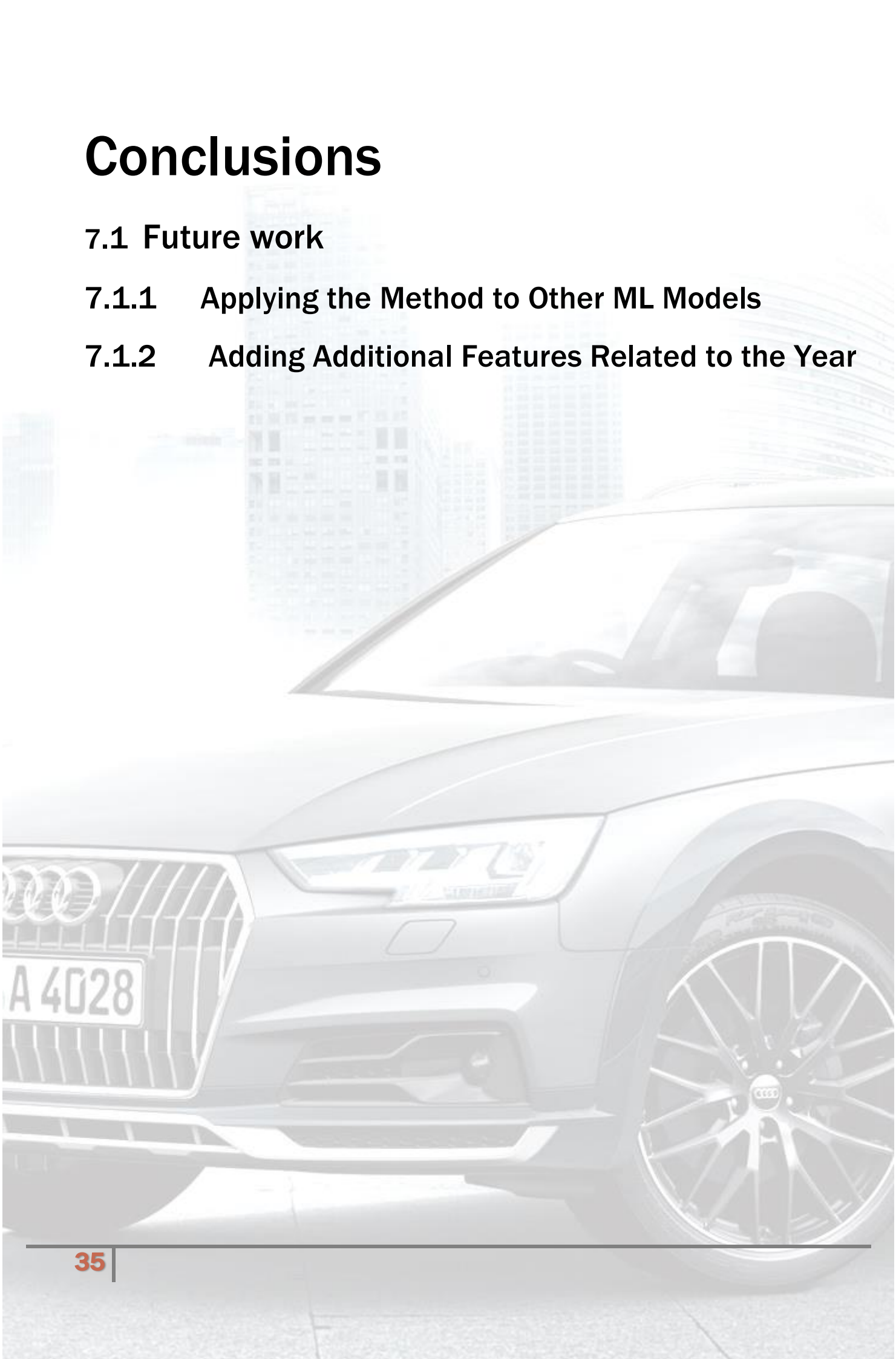
The Jupyter Notebook is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations and narrative text. It includes data cleaning and transformation, numerical simulation, statistical modeling, data visualization, machine learning, and much more.

Conclusions

7.1 Future work

7.1.1 Applying the Method to Other ML Models

7.1.2 Adding Additional Features Related to the Year



Conclusions

This chapter will summarize the work as a whole by conclusively answering the research questions proposed in section 1.4, as well as giving examples of possible future work.

The first research question was to determine which of the models and parameters gives the best overall accuracy in making price predictions for used cars. The optimal parameters were determined in the process of implementing the models, and thus each model was implemented with the parameters that yielded the best performance by trial and error. The results show that out of the four models tested, Random Forest Regression provided the highest accuracy in all of the metrics used and highest overall accuracy.

The second research question was to determine which of the models can most accurately assess the depreciation of a car over time. All of the models approximated geometric appreciation, meaning that a constant percentage of value is lost every year independent of the age of the vehicle. Random Forest Regression had a significantly higher assessed average depreciation at approximately 13.8%, compared to the others with 9.7%. This is closer to the range of 15%-31% assessed by Karl Storchmann in his analysis of international depreciation rates.

The third research question is to determine which model demonstrates the best potential for development of a consumer tool for evaluating used cars or a particular subset of used cars. The results show that Random Forest Regression performed the best on all performance metrics and for all price percentile subsets of used cars. It was also much better able to approximate the depreciation.

7.1 Future Work

This section will explain some possible future research that can expand upon the knowledge gained through this research.

7.1.1 Applying the Method to Other ML Models

This work compared the performance of four ML Regression algorithms. A way to expand this work in the future is to apply the same method for comparing these algorithms to others that are suited to regression problems. Some example algorithms is Kth Nearest Neighbor Regression (KNN). The problem of price prediction deals with continuous variables which makes it suited to regression algorithms, but by creating discrete intervals for the continuous variables such as price, other algorithms could be applied.

7.1.2 Adding Additional Features Related to the Year

A potential improvement to the predictive power of all ML models, if they are able to take advantage of the information, is to add more correlated features. There are some features which are not related to the attributes of the car, such as the price of fuel. A car that uses more fuel will be worth less when fuel costs more. Other such features could include the economic conditions, or changes in the climate.

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import matplotlib as mpl
%matplotlib inline
mpl.style.use('ggplot')
```

```
data=pd.read_csv('quikr_car.csv')
data.head()
```

	name	company	year	Price	kms_driven	fuel_type
0	Hyundai Santro Xing XO eRLX Euro III	Hyundai	2007	80,000	45,000 kms	Petrol
1	Mahindra Jeep CL550 MDI	Mahindra	2006	4,25,000	40 kms	Diesel
2	Maruti Suzuki Alto 800 Vxi	Maruti	2018	Ask For Price	22,000 kms	Petrol
3	Hyundai Grand i10 Magna 1.2 Kappa VTVT	Hyundai	2014	3,25,000	28,000 kms	Petrol
4	Ford EcoSport Titanium 1.5L TDCi	Ford	2014	5,75,000	36,000 kms	Diesel

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 892 entries, 0 to 891
Data columns (total 6 columns):
 #   Column      Non-Null Count  Dtype 
---  ------      --------------  ----- 
 0   name        892 non-null    object
 1   company     892 non-null    object
 2   year        892 non-null    object
 3   Price       892 non-null    object
 4   kms_driven  840 non-null    object
 5   fuel_type   837 non-null    object
dtypes: object(6)
memory usage: 41.9+ KB
```

Quality

- names are pretty inconsistent
- names have company names attached to it
- some names are spam like 'Maruti Ertiga showroom condition with' and 'Well mentained Tata Sumo'
- company: many of the names are not of any company like 'Used', 'URJENT', and so on.
- year has many non-year values
- year is in object. Change to integer
- Price has Ask for Price
- Price has commas in its prices and is in object
- kms_driven has object values with kms at last.
- It has nan values and two rows have 'Petrol' in them
- fuel_type has nan values

Cleaning Data

[9]:
```
backup=data.copy()
```

year has many non-year values

[10]:
```
data=data[data['year'].str.isnumeric()]
```

year is in object. Change to integer

[11]:
```
data['year']=data['year'].astype(int)
```

Price has Ask for Price

[12]:
```
data=data[data['Price']!='Ask For Price']
```

Price has commas in its prices and is in object

[13]:
```
data['Price']=data['Price'].str.replace(',','').astype(int)
```

kms_driven has object values with kms at last.

[14]:
```
data['kms_driven']=data['kms_driven'].str.split().str.get(0).str.replace(',','')
```

It has nan values and two rows have 'Petrol' in them

[15]:
```
data=data[data['kms_driven'].str.isnumeric()]
```

[16]:
```
data['kms_driven']=data['kms_driven'].astype(int)
```

fuel_type has nan values ¶

[17]:
```
data=data[~data['fuel_type'].isna()]
```

[18]:
```
data.shape
```

[18]:
```
(816, 6)
```

name and company had spammed data...but with the previous cleaning, those rows got removed.

Company does not need any cleaning now. Changing car names. Keeping only the first three words

[19]:
```
data['name']=data['name'].str.split().str.slice(start=0,stop=3).str.join(' ')
```

Resetting the index of the final cleaned data

[20]:
```
data=data.reset_index(drop=True)
```

Price has commas in its prices and is in object

[13]:
```
data['Price']=data['Price'].str.replace(',','').astype(int)
```

kms_driven has object values with kms at last.

[14]:
```
data['kms_driven']=data['kms_driven'].str.split().str.get(0).str.replace(',','')
```

It has nan values and two rows have 'Petrol' in them

[15]:
```
data=data[data['kms_driven'].str.isnumeric()]
```

[16]:
```
data['kms_driven']=data['kms_driven'].astype(int)
```

fuel_type has nan values ¶

[17]:
```
data=data[~data['fuel_type'].isna()]
```

[18]:
```
data.shape
```

[18]:
```
(816, 6)
```

name and company had spammed data...but with the previous cleaning, those rows got removed.

Company does not need any cleaning now. Changing car names. Keeping only the first three words

[19]:
```
data['name']=data['name'].str.split().str.slice(start=0,stop=3).str.join(' ')
```

Resetting the index of the final cleaned data

[20]:
```
data=data.reset_index(drop=True)
```

Cleaned Data

[21]:
```
data
```

[56]:

	year	Price	kms_driven	fuel_type_LPG	fuel_type_Petrol	company_BMW	company_Chevrolet	company_Datsun	company_Fiat	company_Force
year	1.000000	0.287193	-0.233695	0.000683	-0.191624	-0.045264	-0.012456	0.099272	-0.056051	0.036056
Price	0.287193	1.000000	-0.120854	-0.021577	-0.193596	0.133712	-0.092045	-0.030557	-0.044610	0.023763
kms_driven	-0.233695	-0.120854	1.000000	0.099385	-0.268928	-0.002245	0.011662	-0.091118	0.023598	-0.018174
fuel_type_LPG	0.000683	-0.021577	0.099385	1.000000	-0.052061	-0.004932	-0.010336	-0.004611	-0.003479	-0.003479
fuel_type_Petrol	-0.191624	-0.193596	-0.268928	-0.052061	1.000000	-0.029789	-0.059355	0.088566	0.031691	-0.073715

5 rows × 29 columns

[20]:
```
corrmat = final_dataset.corr(method='pearson')
```

[21]:
```
import seaborn as sns
corrmat = final_dataset.corr(method='pearson')
top_corr_features = corrmat.index
plt.figure(figsize=(20,15))
g=sns.heatmap(final_dataset[top_corr_features].corr(method='pearson'),annot=True,cmap="RdYlGn")
```

816 rows × 6 columns

[22]:
```
final_dataset = data[["year", "Price",  "kms_driven", "fuel_type","company"]]
final_dataset['fuel_type'].unique()
final_dataset=pd.get_dummies(final_dataset,drop_first=True)
final_dataset
```

[18]:

	year	Price	kms_driven	fuel_type_LPG	fuel_type_Petrol	company_BMW	company_Chevrolet	company_Datsun	company_Fiat	company_Force	...	company_Me
0	2007	80000	45000	False	True	False	False	False	False	False	...	
1	2006	425000	40	False	False	False	False	False	False	False	...	
2	2014	325000	28000	False	True	False	False	False	False	False	...	
3	2014	575000	36000	False	False	False	False	False	False	False	...	
4	2012	175000	41000	False	False	False	False	False	False	False	...	
...	...	...	...	...	...	...	...	...	...	...	...	
811	2011	270000	50000	False	True	False	False	False	False	False	...	
812	2009	110000	30000	False	False	False	False	False	False	False	...	
813	2009	300000	132000	False	True	False	False	False	False	False	...	
814	2018	260000	27000	False	False	False	False	False	False	False	...	
815	2013	390000	40000	False	False	False	False	False	False	False	...	

816 rows × 29 columns

[19]:
```
final_dataset.corr(method ='pearson')
```

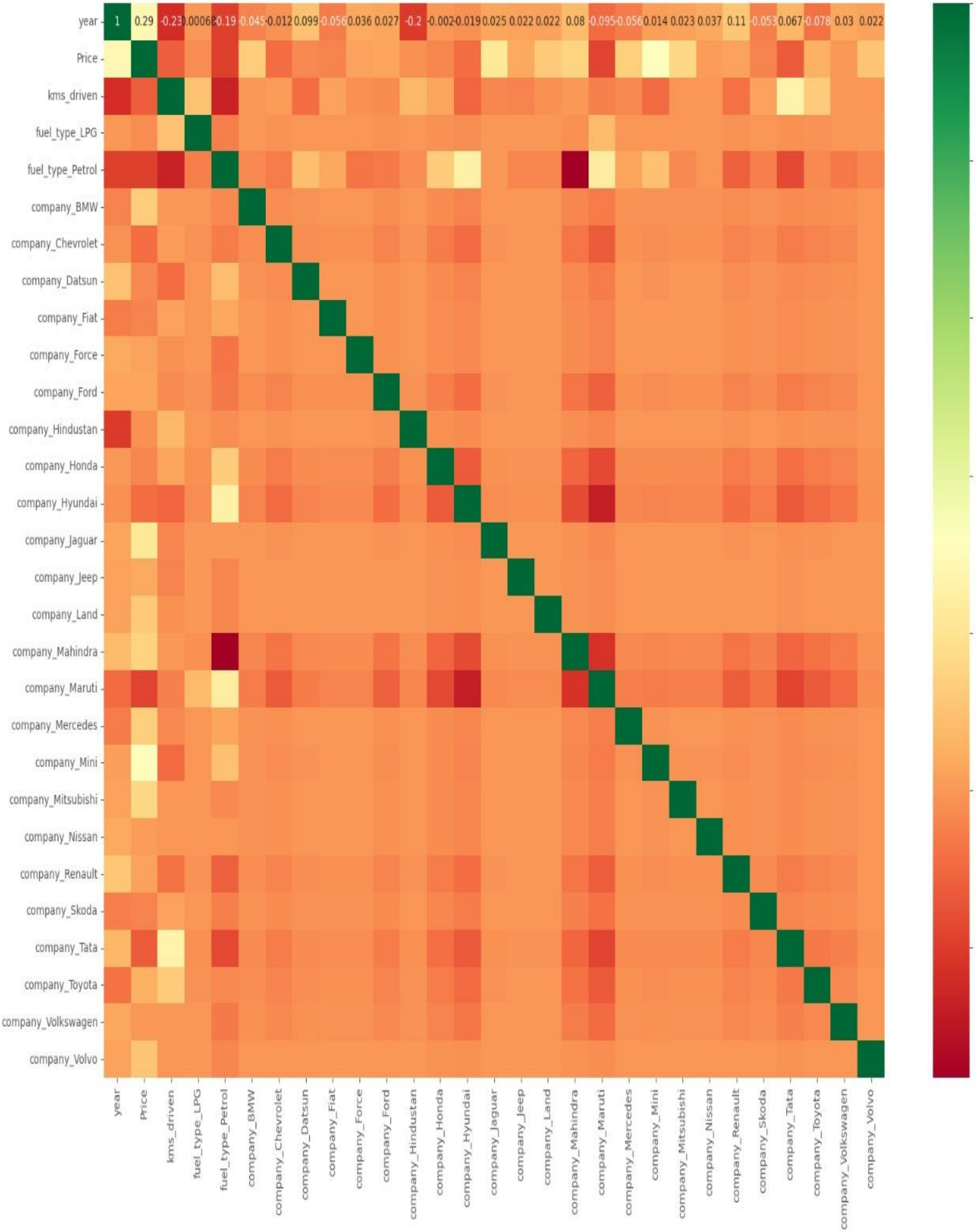
[56]:

	year	Price	kms_driven	fuel_type_LPG	fuel_type_Petrol	company_BMW	company_Chevrolet	company_Datsun	company_Fiat	company_Force
year	1.000000	0.287193	-0.233695	0.000683	-0.191624	-0.045264	-0.012456	0.099272	-0.056051	0.036056
Price	0.287193	1.000000	-0.120854	-0.021577	-0.193596	0.133712	-0.092045	-0.030557	-0.044610	0.023763
kms_driven	-0.233695	-0.120854	1.000000	0.099385	-0.268928	-0.002245	0.011662	-0.091118	0.023598	-0.018174
fuel_type_LPG	0.000683	-0.021577	0.099385	1.000000	-0.052061	-0.004932	-0.010336	-0.004611	-0.003479	-0.003479
fuel_type_Petrol	-0.191624	-0.193596	-0.268928	-0.052061	1.000000	-0.029789	-0.059355	0.088566	0.031691	-0.073715

5 rows × 29 columns

[20]:
```
corrmat = final_dataset.corr(method='pearson')
```

[21]:
```
import seaborn as sns
corrmat = final_dataset.corr(method='pearson')
top_corr_features = corrmat.index
plt.figure(figsize=(20,15))
g=sns.heatmap(final_dataset[top_corr_features].corr(method='pearson'),annot=True,cmap="RdYlGn")
```


year
Price
kms_driven
fuel_type_LPG
fuel_type_Petrol
company_BMW
company_Chevrolet
company_Datsun
company_Fiat
company_Force
company_Ford
company_Hindustan
company_Honda
company_Hyundai
company_Jaguar
company_Jeep
company_Land
company_Mahindra
company_Maruti
company_Mercedes
company_Mini
company_Mitsubishi
company_Nissan
company_Renault
company_Skoda
company_Tata
company_Toyota
company_Volkswagen
company_Volvo
1.0
0.8
0.6
0.4
0.2
0.0
−0.2

[22]:
```
data.to_csv('Car_data.csv')
```

[23]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 816 entries, 0 to 815
Data columns (total 6 columns):
 #   Column      Non-Null Count  Dtype 
---  ------      --------------  ----- 
 0   name        816 non-null    object
 1   company     816 non-null    object
 2   year        816 non-null    int32 
 3   Price       816 non-null    int32 
 4   kms_driven  816 non-null    int32 
 5   fuel_type   816 non-null    object
dtypes: int32(3), object(3)
memory usage: 28.8+ KB
```

[24]:
```
data.describe(include='all')
```

[24]:

	name	company	year	Price	kms_driven	fuel_type
count	816	816	816.000000	8.160000e+02	816.000000	816
unique	254	25	NaN	NaN	NaN	3
top	Maruti Suzuki Swift	Maruti	NaN	NaN	NaN	Petrol
freq	51	221	NaN	NaN	NaN	428
mean	NaN	NaN	2012.444853	4.117176e+05	46275.531863	NaN
std	NaN	NaN	4.002992	4.751844e+05	34297.428044	NaN
min	NaN	NaN	1995.000000	3.000000e+04	0.000000	NaN
25%	NaN	NaN	2010.000000	1.750000e+05	27000.000000	NaN
50%	NaN	NaN	2013.000000	2.999990e+05	41000.000000	NaN
75%	NaN	NaN	2015.000000	4.912500e+05	56818.500000	NaN
max	NaN	NaN	2019.000000	8.500003e+06	400000.000000	NaN

[25]:
```
data=data[data['Price']<6000000]
```

Checking relationship of Company with Price

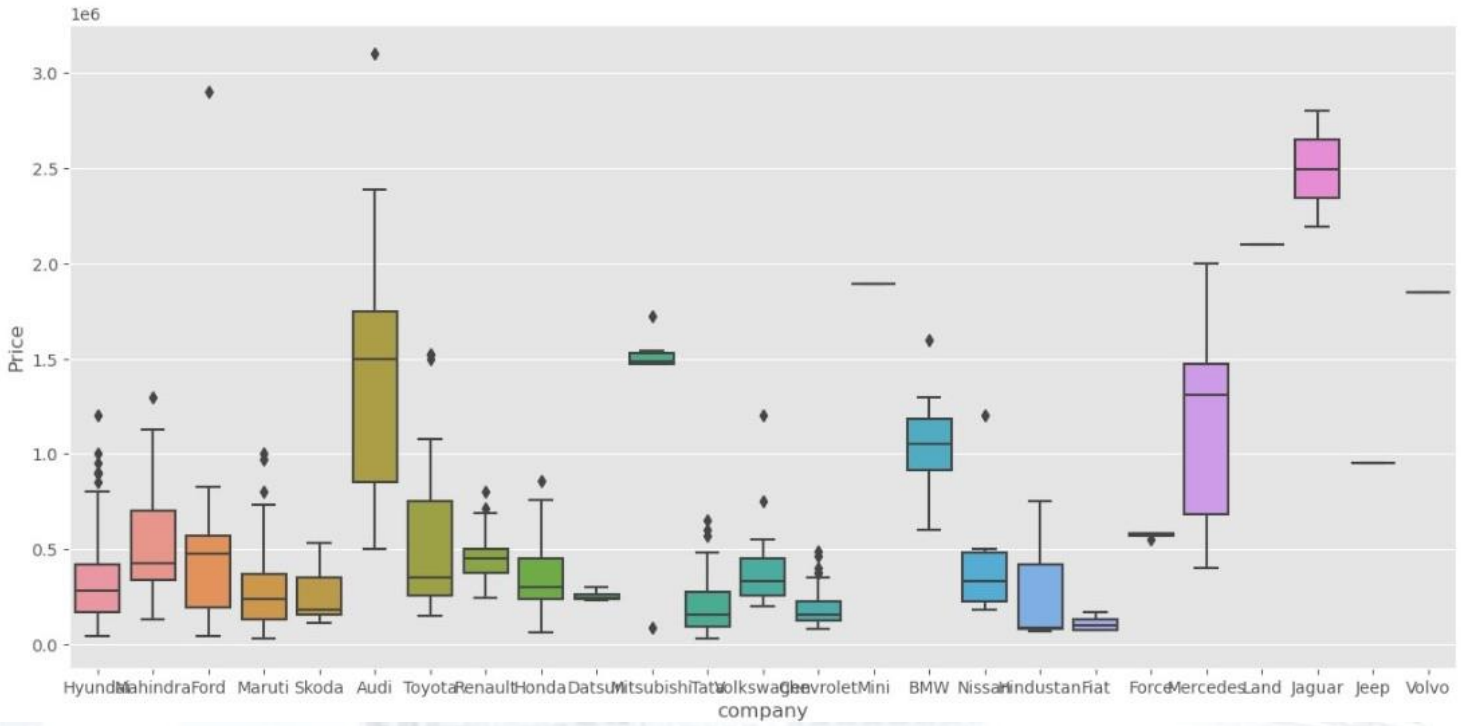
[26]:
```
data['company'].unique()
```

[26]:
```
array(['Hyundai', 'Mahindra', 'Ford', 'Maruti', 'Skoda', 'Audi', 'Toyota',
       'Renault', 'Honda', 'Datsun', 'Mitsubishi', 'Tata', 'Volkswagen',
       'Chevrolet', 'Mini', 'BMW', 'Nissan', 'Hindustan', 'Fiat', 'Force',
       'Mercedes', 'Land', 'Jaguar', 'Jeep', 'Volvo'], dtype=object)
```

[27]:
```
import seaborn as sns
```

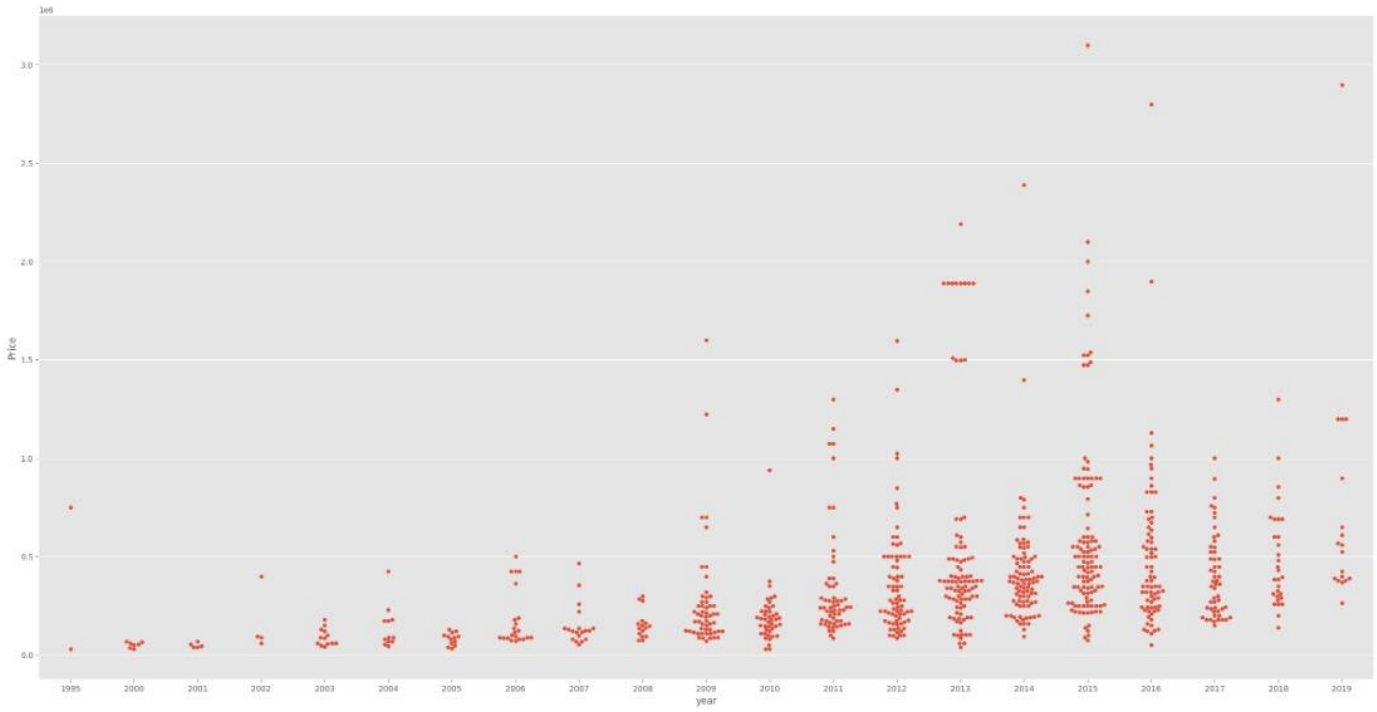
Type Markdown and LaTeX: α^2

[28]:
```
plt.subplots(figsize=(15,7))
ax=sns.boxplot(x='company',y='Price',data=data)
plt.show()
```



Checking relationship of Year with Price

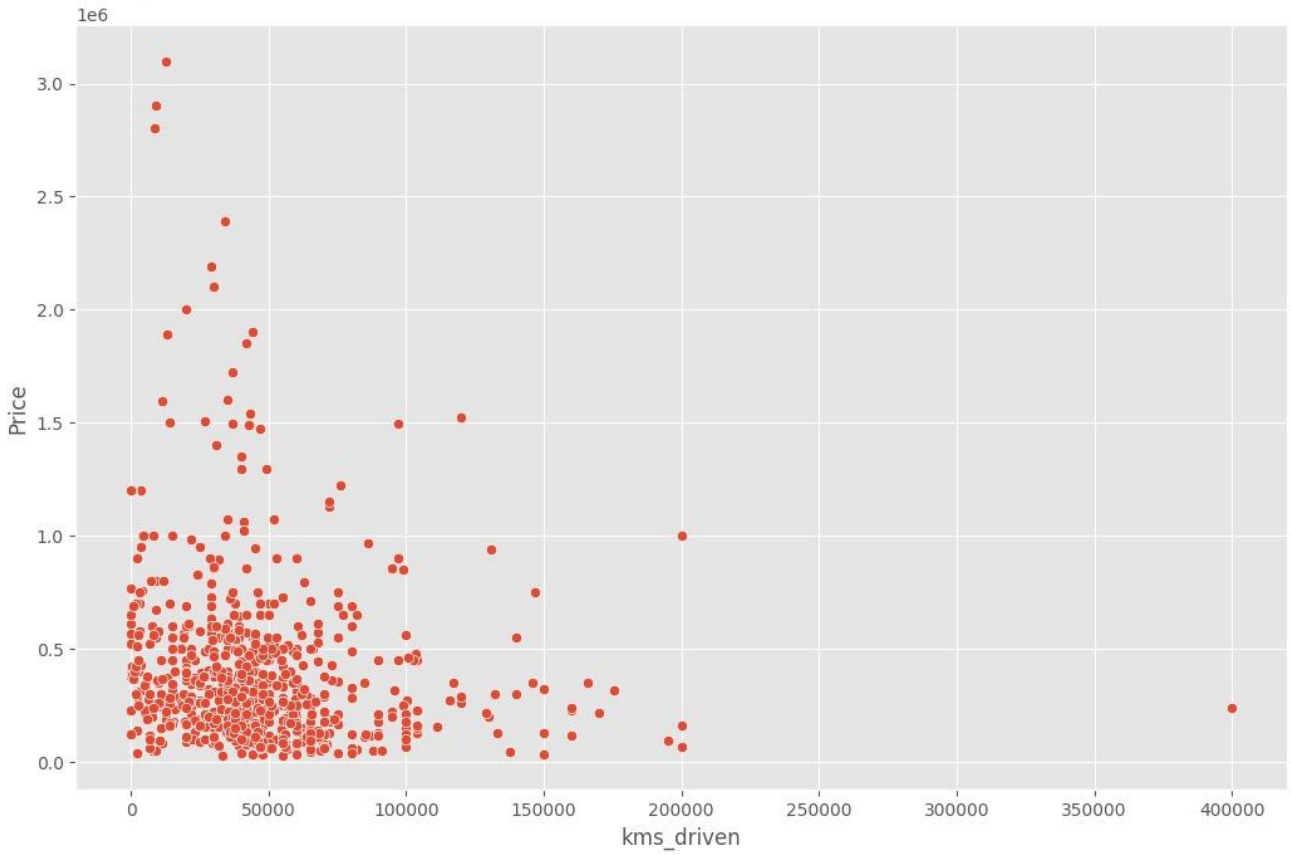
[29]:
```
plt.subplots(figsize=(30,15))
ax=sns.swarmplot(x='year',y='Price',data=data)
plt.show()
```



Checking relationship of kms_driven with Price

[30]:
```
sns.relplot(x='kms_driven',y='Price',data=data,height=7,aspect=1.5)
```

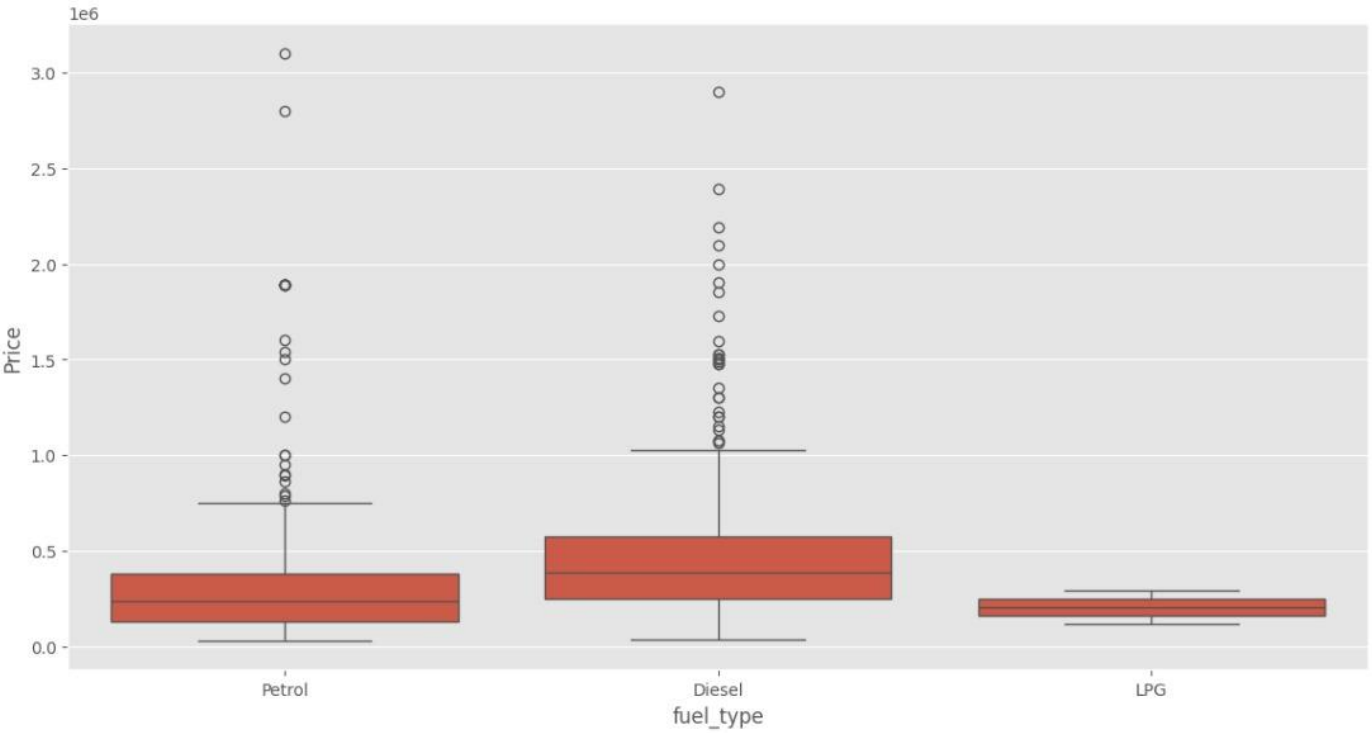
[30]:
```
<seaborn.axisgrid.FacetGrid at 0x1b3c7894e30>
```



Checking relationship of Fuel Type with Price

[31]:
```
plt.subplots(figsize=(14,7))
sns.boxplot(x='fuel_type',y='Price',data=data)
```

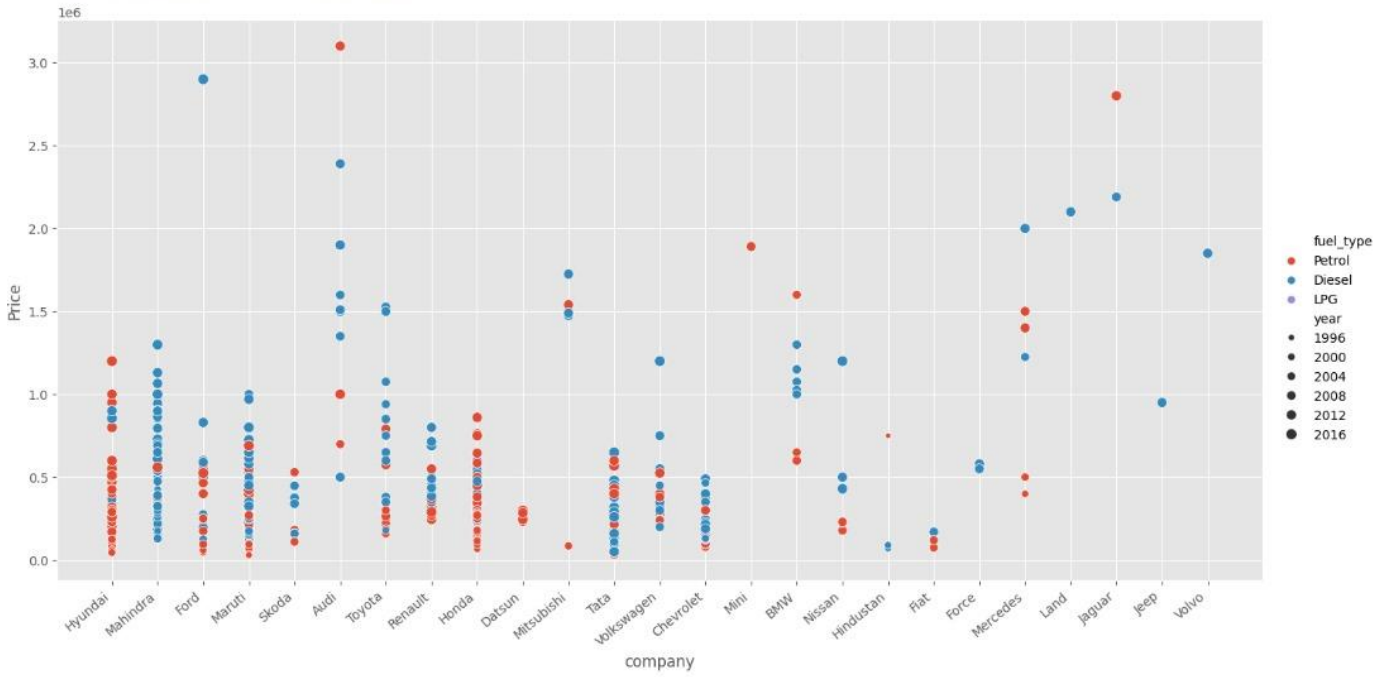
[31]:
```
<Axes: xlabel='fuel_type', ylabel='Price'>
```



Relationship of Price with FuelType, Year and Company mixed

[32]:
```
ax=sns.relplot(x='company',y='Price',data=data,hue='fuel_type',size='year',height=7,aspect=2)
ax.set_xticklabels(rotation=40,ha='right')
```

[32]:
```
<seaborn.axisgrid.FacetGrid at 0x1b3e2b94d40>
```



Extracting Training Data

[33]:
```
X=data[['name','company','year','kms_driven','fuel_type']]
y=data['Price']
```

[34]:
```
X
```

[34]:

	name	company	year	kms_driven	fuel_type
0	Hyundai Santro Xing	Hyundai	2007	45000	Petrol
1	Mahindra Jeep CL550	Mahindra	2006	40	Diesel
2	Hyundai Grand i10	Hyundai	2014	28000	Petrol
3	Ford EcoSport Titanium	Ford	2014	36000	Diesel
4	Ford Figo	Ford	2012	41000	Diesel
...	...	...	...	...	...
811	Maruti Suzuki Ritz	Maruti	2011	50000	Petrol
812	Tata Indica V2	Tata	2009	30000	Diesel
813	Toyota Corolla Altis	Toyota	2009	132000	Petrol
814	Tata Zest XM	Tata	2018	27000	Diesel
815	Mahindra Quanto C8	Mahindra	2013	40000	Diesel

815 rows × 5 columns

[35]:
```
y
```

[35]:
```
0       80000
1      425000
2      325000
3      575000
4      175000
        ...
811    270000
812    110000
813    300000
814    260000
815    390000
Name: Price, Length: 815, dtype: int32
```

Applying Train Test Split

[36]:
```
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.2)
```

[37]:
```
from sklearn.linear_model import LinearRegression
```

[38]:
```
from sklearn.preprocessing import OneHotEncoder
from sklearn.compose import make_column_transformer
from sklearn.pipeline import make_pipeline
from sklearn.metrics import r2_score
```

Creating an OneHotEncoder object to contain all the possible categories

[39]:
```
ohe=OneHotEncoder()
ohe.fit(X[['name','company','fuel_type']])
```

[39]:
```
OneHotEncoder()
```

Creating a column transformer to transform categorical columns

[40]:
```
column_trans=make_column_transformer((OneHotEncoder(categories=ohe.categories_),['name','company','fuel_type']),
                                     remainder='passthrough')
```

Linear Regression Model

[41]:
```
lr=LinearRegression()
```

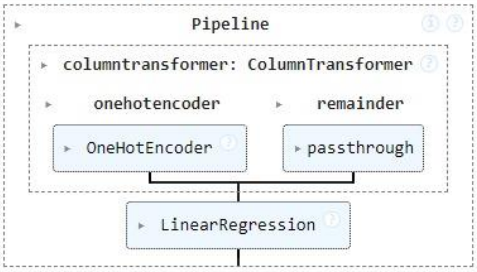
Making a pipeline

[42]:
```
pipe=make_pipeline(column_trans,lr)
```

Fitting the model

[43]:
```
pipe.fit(X_train,y_train)
```

[43]:



[44]:
```
y_pred=pipe.predict(X_test)
```

Checking R2 Score ¶

[45]:
```
r2_score(y_test,y_pred)
```

[45]:
```
0.665042450361758
```

Finding the model with a random state of TrainTestSplit where the model was found to give almost 0.92 as r2_score

[46]:
```
scores=[]
for i in range(1000):
    X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=.2,random_state=i)
    lr=LinearRegression()
    pipe=make_pipeline(column_trans,lr)
    pipe.fit(X_train,y_train)
    y_pred=pipe.predict(X_test)
    scores.append(r2_score(y_test,y_pred))
```

[47]:
```
np.argmax(scores)
```

[47]:
```
433
```

[48]:
```
scores[np.argmax(scores)]
```

[48]:
```
0.8457059012561223
```

[49]:
```
pipe.predict(pd.DataFrame(columns=X_test.columns,data=np.array(['Maruti Suzuki Swift','Maruti',2019,100,'Petrol']).reshape(1,5)))
```

[49]:
```
array([431223.99699965])
```

The best model is found at a certain random state

[50]:
```
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.2,random_state=np.argmax(scores))
lr=LinearRegression()
pipe=make_pipeline(column_trans,lr)
pipe.fit(X_train,y_train)
y_pred=pipe.predict(X_test)
r2_score(y_test,y_pred)
```

[50]:
```
0.8457059012561223
```

[51]:
```
import pickle
```

[52]:
```
pickle.dump(pipe,open('LinearRegressionModel.pkl','wb'))
```

[53]:
```
pipe.predict(pd.DataFrame(columns=['name','company','year','kms_driven','fuel_type'],data=np.array(['Maruti Suzuki Swift','Maruti',2019,100,'Petrol']).r
```

[53]:
```
array([458894.10960853])
```

[54]:
```
pipe.steps[0][1].transformers[0][1].categories[0]
```

[54]:
```
array(['Audi A3 Cabriolet', 'Audi A4 1.8', 'Audi A4 2.0', 'Audi A6 2.0',
       'Audi A8', 'Audi Q3 2.0', 'Audi Q5 2.0', 'Audi Q7', 'BMW 3 Series',
       'BMW 5 Series', 'BMW 7 Series', 'BMW X1', 'BMW X1 sDrive20d',
       'BMW X1 xDrive20d', 'Chevrolet Beat', 'Chevrolet Beat Diesel',
       'Chevrolet Beat LS', 'Chevrolet Beat LT', 'Chevrolet Beat PS',
       'Chevrolet Cruze LTZ', 'Chevrolet Enjoy', 'Chevrolet Enjoy 1.4',
       'Chevrolet Sail 1.2', 'Chevrolet Sail UVA', 'Chevrolet Spark',
       'Chevrolet Spark 1.0', 'Chevrolet Spark LS', 'Chevrolet Spark LT',
       'Chevrolet Tavera LS', 'Chevrolet Tavera Neo', 'Datsun GO T',
       'Datsun Go Plus', 'Datsun Redi GO', 'Fiat Linea Emotion',
       'Fiat Petra ELX', 'Fiat Punto Emotion', 'Force Motors Force',
       'Force Motors One', 'Ford EcoSport', 'Ford EcoSport Ambiente',
       'Ford EcoSport Titanium', 'Ford EcoSport Trend',
```

[55]:
```
y_pred=pipe.predict(X_test)
from sklearn.metrics import mean_squared_error
mse= mean_squared_error(y_test,y_pred)
rmse = np.sqrt(mse)
print("error=",rmse)
```

```
error= 162900.33069383932
```

[ ]:

THANKS YOU!