

Submitted in part fulfilment for the degree of BEng(hons)

**Prediction of IMDb ratings using supervised machine learning**

Jack Forryan

16th June 2021

Supervisor: James Woodcock

**Contents**

[**Abstract**](#_gjdgxs) **4**

[**Executive Summary**](#_30j0zll) **5**

**1** [**Introduction**](#_1fob9te) **1**

**2** [**Literature Review**](#_3znysh7) **2**

[2.1 Data collection](#_tyjcwt) 2

[2.2 Linear regression](#_1t3h5sf) 3

[2.3 Ridge, Lasso and Ordinary Least Squares as Regression methods](#_2s8eyo1) 4

[2.4 K-fold cross validation](#_17dp8vu) 5

[2.5 Related works](#_3rdcrjn) 6

[**3 Motivation, Essential Knowledge and Problem Analysis**](#_26in1rg) **9**

[3.1 Motivation](#_35nkun2) 9

[3.2 Essential Knowledge](#_1ksv4uv) 10

[3.2.1 Introduction to Computer Science and Programming using Python - Kaggle course](#_44sinio) 10

[3.2.2 Aurélien Geron’s: Hands-On Machine Learning with Scikit-Learn and TensorFlow](#_2jxsxqh) 10

[3.2.3 Introduction to Machine Learning - Kaggle course](#_z337ya) 10

[3.3 Problem Analysis](#_3j2qqm3) 10

[3.4 Hypothesis](#_1y810tw) 11

[**4 Method**](#_4i7ojhp) **11**

[4.1 Overview](#_2xcytpi) 11

[4.2.1 Method Diagram:](#_1ci93xb) 13

[4.2.2 Data information](#_2bn6wsx) 14

[4.3 In Depth Method](#_3as4poj) 14

[4.3.1 Installing the prerequisite programs](#_1pxezwc) 14

[4.3.2 Necessary libraries](#_2p2csry) 15

[4.3.3 Collection of relevant data from IMDb](#_23ckvvd) 15

[4.3.4 Simple regression testing, model creation and train\_test\_split](#_41mghml) 18

4.3.5 [Data visualisation and results](#_2grqrue) 19

4.4.1 [IMDb overhaul and Neural Networking implementation](#_vx1227) 19

4.4.2 [Kaggle implementation](#_3fwokq0) 19

[**5 Results**](#_1v1yuxt) **20**

[5.1 Data collection](#_4f1mdlm) 20

[5.1.2 Train and test split, linear regression comparisons, metric selection](#_28h4qwu) 21

[5.1.3 Kaggle dataset implementation for better R2 values](#_46r0co2) 23

[5.1.4 Addition of a multilayer perceptron Neural network to achieve a better MAE](#_3l18frh) 24

[**6 Evaluation**](#_2zbgiuw) **25**

[6.1 Data collection and preprocessing:](#_1egqt2p) 25

[6.2 Metric selection and model evaluation](#_3ygebqi) 25

[**7 Conclusion and Further Work**](#_2dlolyb) **26**

[**Bibliography**](#_sqyw64) **28**

[**List of figures**](#_31psuiudzjem) **31**

# Acknowledgements

I would like to thank my supervisor Dr. James Woodcock who has helped guide me through my time working on this project, also serving as an important point of communication for when I encountered problems.

# Abstract

This study will test whether you can successfully predict a cinematographic film’s rating based on the information available prior to its release. Once a successful model is created it could be trained upon a set of information ‘tags’ relating to the relevant film, resulting in an accurate way to predict whether the film is worth watching, before it has been formally rated.

The study uses data collected from IMDb (Internet Movie Database), a website that aggregates and provides information about film and television programs as well as other information, including budget, cast, crew and directors. For the purposes of this study, the information was collected using BeautifulSoup, a Python library for extracting data from HTML, XML, and other markup languages. It allows the user to extract specific data from a webpage, remove the HTML markup, and then save that data. Once the data is extracted, linear regression is used in order to build a model that can predict the relevant film’s review score (which ranges from 0.0 to 10.0).

The goal of this study is to (a) identify which specific variables have the most significant impact on a film’s rating, and (b) to create a model that can predict a film’s review score with a significant degree of accuracy. By expanding the amount of data used and by setting more restrictions on the data, it would likely be possible to make this model produced in this study even more accurate.

# Executive Summary

This report will investigate whether or not we can predict a film’s rating based on the information available prior to the relevant film’s release. The primary goal is to isolate which feature(s), such as genre, director and/or runtime, contribute the most to a film’s review score. The report is based on data relating to 100 films released between 2015 and 2020.

People make many assumptions about what makes a film ‘successful’. For example, the most popular film genres in recent times are ‘action’ and ‘adventure’, a fact evidenced by the incredible success of mainstream superhero and fantasy films. Consider, for example, the top 3 highest grossing movies over the past 10 years:

1. Avengers: Endgame (2019) – £2.1 billion
2. Star Wars: The Force Awakens (2015) £1.6 billion
3. Avengers: Infinity War (2018) – £1.6 billion

[17]

On the other hand, horror films are less well-regarded (perhaps because fewer people actively enjoy being frightened). It will be interesting to consider, therefore, whether the genre of a movie in fact has a significant effect on the film’s predicted rating (beyond its obvious relationship with a film’s box office success). Similarly, one might consider a film’s runtime, another factor that is commonly seen as indicative of a film’s success. Maybe people can lose interest when a film runs too long, so one might assume a longer runtime would negatively impact that film’s rating. Again, it will be interesting to see if this is borne out in the results of this study.

In order to determine if these high-level assumptions hold true in the data, an algorithmic system can be developed to make predictions based on a wide array of historic data. This is where the concepts and frameworks of supervised machine learning can be utilised. Supervised machine learning is a common technique in the field of artificial intelligence, and allows the user to ‘construct algorithms that are able to produce general patterns and hypotheses by using externally supplied instances to predict the fate of future instances’. [2] Using a supervised machine learning method (specifically, linear regression) this report will attempt to identify if a consistent link can be made between the features associated with a film and its rating upon release. We hypothesise that if we can observe a link between film features and film rating over past data, the same link can be used to predict a film’s rating in the future, as we should be able to model the relationship between our selected features and our continuous target variable, (i.e. the film’s rating).

To test this hypothesis, data on 100 films from the period from 01/01/2015 to 01/01/2020 was collected from IMDb. Once the features were isolated and stored in a dictionary, they were saved to a database. Instead of using a customary train and test split, a K-fold cross validation was implemented to train the system. K-fold cross validation is a method of splitting up data into training sets and test sets. A K-fold cross validation was used due to the relatively small size of the data set. Using the predictions outputted by our trained model, we were then able to evaluate the results and measure the model’s accuracy. This study ultimately did not result in a significantly accurate predictive model, however a clear link was established in that a positive R-squared value and a sufficiently low MAE value were obtained, which suggests that with modifications an even more successful model could be built. It would likely be possible to improve the accuracy of the model with more data and time, and this study therefore outlines a number of ways that one could continue to build and improve the model.

A material technical issue occurred during the final week of this project, when IMDb updated their entire website, causing all of the data to have different html headers and tags, rendering the web scraper unusable for the time being. Fortunately, a usable dataset had already been retrieved from IMDb prior to this update, so the linear regression model can still be shown to work.

Finally, this project also touches on a number of potential legal, ethical, or even professional issues. One might question whether the general public should know prior to a film’s release how it will likely perform, as this might greatly affect how much a movie will gross, and could have a number of negative effects on competition within the film industry. One can imagine how it might benefit large, wealthy film studios who have the resources to fund this sort of artificial intelligence research during film development and marketing, to the detriment of small independent film producers who might then struggle to gain a foothold in the market.

# Introduction

This project draws on multiple concepts that are at the forefront of modern Computer Science and amalgamates them together. It touches on data analysis, data visualisation, and machine learning (specifically multivariate linear regression). Machine learning is becoming increasingly important in the modern world, due to the unprecedented amount of data now available to everybody on the internet, which provides various opportunities for large-scale data analysis. Machine learning is now integral to almost all companies, since it allows them to use customer data to increase profits and compete in the general market. There are many examples of Machine learning, one of the most notable being Google Translate, which uses a neural networking method for machine translation, allowing the user to translate between almost any two languages.

Our aim in this report is to test the hypothesis that information stored in IMDb could be used to predict film ratings, and to isolate the most important features for predicting such ratings. If this is successful, the model could help a user to decide whether a film is worth watching before it is released, and could help filmmakers and producers understand why some movies perform better or worse than others.

However, training such a model is not a simple case of lifting all of the data from a film’s IMDb page and running an algorithm. Before the model can be properly trained, the data must be processed into a format that is easily digestible by a machine learning model. This project makes use of a python library called ‘BeautifulSoup’ created specifically for this purpose, which allows the user to pick out a small number of relevant features from any specific IMDb hyperlink, which vastly reduces the amount of data that has to be parsed through the algorithm. This has the effect of reducing the runtime of the algorithm and allows us to train the model using a standard computer with limited processing power.

Once the data has been processed and saved to a file, we can then test our hypothesis that there ought to be some link between a specific film’s features (i.e. those features that were selected during data pre-processing) and the rating it eventually received. In this study, then, we will implement a method for identifying this link. We will then consider whether the predictions made by the model are accurate. The study will be a success if we can consistently generate a predicted movie rating value that is close in number to the actual rating the movie received upon release.

# Literature Review

This chapter will initially focus on two of the most important concepts in the creation of the model:

**Data collection:** as a way of scraping important data directly from IMDb through the use of the Python library BeautifulSoup. BeautifulSoup allows you to sort content based upon tags, thus giving us a way to isolate important data contained on an IMDb page and store it in a database to be used by our regression model.

**Supervised machine learning:** specifically linear regression, as a way to predict a film’s rating, using the film’s features (such as genre, director, stars, runtime), as our independent variables, and the film’s rating (on a scale of 0.0 - 10.0) as our dependent variable.

## 

## 2.1 Data collection

BeautifulSoup is a method for collecting data from a specific URL. In this study, the base URL we used is ‘[https://www.IMDb.com](https://www.imdb.com)’. IMDb displays specific movie links in batches of 100, meaning on each individual IMDb page that is parsed through, there are 100 links containing movie specific data that must be collected. First, a ‘start page’ is fed into the algorithm, and then a method to determine which IMDb pages we will collect movie data on, each page containing 100 movies. Following retrieval of the relevant data, a dictionary of films and their specific information is created, which can then be sorted into a list of data of each film’s relevant headings, and the information related to that heading. For example, the movie rating can be saved as a float value between 0 and 10, and the name of the director can be saved as an object. Once this dictionary has been created, the Python module ‘pickle’ can be used to save the database into a file that can be easily accessed.

## 

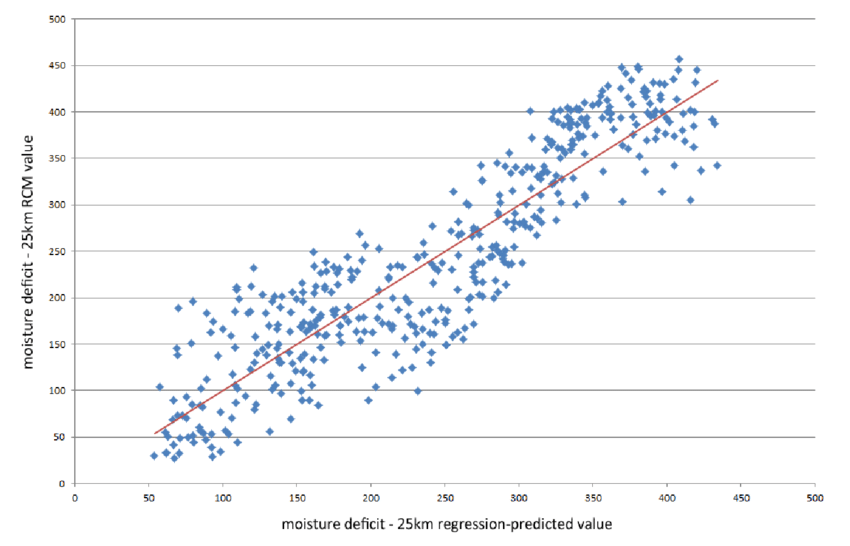
## 2.2 Linear regression

Linear regression is, at its core, a machine learning algorithm. The key idea of machine learning being that a machine can be taught to make predictions about future data having been trained upon past data. Linear regression models are subsequently used to predict the relationship between variables. In this case the film’s rating is the dependent variable, and the factors that are being used to predict the film’s rating are the independent variables. As there are multiple independent variables being used to predict a single dependent variable, we are technically using multiple linear regression, or multivariate linear regression. The formula for multiple regression is as follows:

Where y is the predicted value of the dependant variable (movie rating), β0 is the y intercept, is the regression coefficient () of the first independent variable (), is the regression coefficient for the last independent variable and is the model error or variation in y. Simple linear regression can often result in the problem of overfitting (where the model learns to fit the past data too accurately, which can negatively impact its ability to predict future data), so ridge regression has been used in this study as a way to reduce model complexity and prevent overfitting. Once we have added all of the features to the model (the independent variables) we will be able to create a scatterplot of the movie rating predictions against the actual movie ratings that are observed, creating a scatterplot and line of best fit to easily visualise the results. We can then further isolate the most important features, specifically the features that most positively or negatively impact a movie’s rating, so we have a deeper understanding of why a movie may not receive the exact rating the model has predicted.

At its core, regression is the study of dependence, and is a means to summarize observed data in a way that can be easily deciphered. [4] Weisberg’s study from 2005 on applied linear regression suggests that the observation of data is necessary in order to decide how to generate the model. A key part of observing data in this case is to display graphs of the data, so as to see how the model is performing. To show how well the linear regression algorithm that will be implemented in this project is performing, scatterplots will be used. Specifically, as there are multiple independent values being used in order to predict the movie rating, scatterplot matrices will be used so that we can organise all of the individual scatterplots into an easily readable format. In the next section we will briefly discuss the different regression algorithms that were applicable to this study, and why one was chosen over the others.

A linear regression plot will resemble the following image:



*Figure 1: Example linear regression plot*

Where the x axis will be the predicted IMDb ratings, and the y axis will be the actual IMDb ratings, with a line of best fit drawn through the data.

## 2.3 Ridge, Lasso and Ordinary Least Squares as Regression methods

The three main regression methods considered in this report were Ridge Regression, Lasso Regression and Ordinary Least Squares. Each method could be used as the primary regression algorithm in this project, however there are benefits and detriments to each, which must be considered. Ordinary Least Squares (“**OLS**”) is the simplest regression method available. It is an ‘unbiased’ method, which means that no ‘weights’ are given to the predictor variables, so that they are all considered as important to the output as one another. This is desirable when used on simple training data. However, when some points in the data have excessively large or small values compared to the rest of the training data, it can detrimentally affect the model’s performance. To understand why this is the case, we need to understand how OLS operates. OLS attempts to minimise the sum of the squared errors between the values of the dependent variables in our training set, and the OLS model’s predictions for those values. This essentially means that by minimising the sum of the squared error between these values, you are making the predicted values and actual values as close to each other as possible, thus increasing the accuracy of the prediction. However, due to the fact that the values are squared when some values are much larger than others (potentially due to anomalous data) they will have a larger effect on the predictions of the model, likely resulting in overfitting. One way to combat these shortcomings of OLS was to use Lasso or Ridge regression. Lasso regression, however, can sometimes result in eliminating features from the model, which we do not want, because we want to account for every independent variable we are retrieving from IMDb. On this basis, the decision to use Ridge regression was made. Ridge regression, like the OLS method, works to minimise a loss function that includes the sum of squared regression residuals. The difference between Ridge regression and OLS however, is that Ridge regression includes a penalty parameter in its loss function: λ times the model complexity, measured by the sum of squared regression weights. By using this penalty, overfitting can be avoided by shrinking the weights towards zero. This is therefore a biased form of regression (as opposed to the unbiased approach of OLS). Ridge regression was first theorized in 1970 by Hoerl and Kennard and was observed to provide estimates of regression coefficients with a smaller total mean square error than OLS. While there are situations where OLS may be more appropriate to use, they are very limited. For any relatively complex data sets (such as the one relevant for this report), Ridge regression and Lasso regression have been observed to outperform OLS in almost every simulation. [11]

## 2.4 K-fold cross validation

K-fold cross validation is an alternative to the ‘Train Test Split’ approach used by many regression algorithms. A ‘Train Test Split’ approach works by splitting the data that has been retrieved into training and test sets. The model will train itself on the training set and then use the test set to check whether the model is working as intended. This is usually done at around an 80:20 train:test split. This can, however, lead to a high bias for small datasets, because important information may still be contained in the test set comprising the remaining 20% of the data. Given the relatively small dataset for this project, it was preferable to use the K-fold cross validation technique, instead of a traditional ‘Train Test Split’. K-fold cross validation avoids the problems of the basic train test split by ensuring that every individual datum has the chance to appear in both a training set and a test set. This is done by dividing the data into batches (10 is a commonly used batch-size). If 10 sets of data are created, then training would occur over 9 (i.e. 10 minus 1) sets and the model would be validated using the remaining set. This process is then repeated until every single set has been used as the test set and an average is recorded, which will serve as the model's performance metric. This makes sure that all data is considered equally and that no important information is left out and not trained upon.

## 2.5 Related works

A 2017 study [5] used the same principles of regression analysis, but instead applied it to a separate database of video game reviews, called STEAM. However, instead of directly considering genre and the budget that went into creating the game, sentiment analysis was used, which is a method of reviewing the actual text related to the game review and taking note of positive or negative words that may affect its rating. To do this, a sentiment dictionary must be made, composed of positive and negative words. In this study, however, we only needed to create a dictionary of the various features related to a specific film, dramatically reducing the amount of data analysis that was required. The authors of the STEAM study found that this sentiment analysis method only had a small effect on the usability of the game review.

A similar study using sentiment analysis in relation to IMDb was carried out in 2016 [6] however they used a class based prediction method based on four classes, ranging from terrible to excellent, to sort their results. Films with a rating of 0-2.4 were classed as ‘terrible’, and those with a rating of 7.5 to 10 were classed as ‘excellent’. This class-based method would likely increase the stated accuracy of the model (as it only needs to make correct predictions between those bounds), however it will never give an exact predicted value. As such, while it may be considered a successful model if it can correctly predict the class, there may still be a large difference between the predicted movie rating and the actual movie rating within those bounds. This 2016 study used logistic regression instead of linear regression, as it involves a classification problem where predicted movie ratings are linked to their respective four classes. In this current project, however, we are trying to predict the exact value of the movie rating (i.e. a continuous variable), and therefore linear regression was the chosen method.

While in this study we take any review and give it the same weighting, another study involving IMDb data [7] instead focused on figuring out whether reviews are actually helpful, by creating a model that can predict the helpfulness of a review. There is potential in this method to be applied in tandem with the study being conducted in this project. A study conducted in 2016 [8] considered the common ‘troll mentality’, wherein reviewers would submit a number of unhelpful and/or irrelevant reviews (e.g. ‘copy and pastes’ of old speeches, such as Martin Luther Kings ‘I have a Dream’ speech, or just one word repeated over and over). These reviews are often made multiple times by the same user in an attempt to influence the film’s rating. The ability to completely remove these uninformative and unhelpful reviews from the dataset would remove the amount of data needed to train the model and likely remove inconsistencies. This would require the need to utilise sentiment analysis over the entire body text of each review however, instead of simplifying the reviews to just the specific variables scraped from IMDb.

A 2017 study utilised a regression based prediction algorithm for TV ratings [9] using Ridge regression. As previously discussed, Ridge regression is the specific regression method used in this project to create a predictive model. Ridge regression is a regularisation method which can treat each predictor variable differently depending on how much it affects the outcome. This 2017 study also used K-fold cross-validation in its train test split method, which (as discussed) allows the user to divide the training set into a number of batches, and then to ensure that every piece of data gets trained upon and that there is none left out, thus potentially increasing the accuracy of the model. The 2017 study [9] concluded that the use of K-fold cross validation and ridge regression does in fact optimise model reliability, and that it is an applicable and useful method to be used in the artificial intelligence field.

Another machine learning technique that is commonly used in this area of study is ‘Deep Neural Networks’, specifically DeepFM. A ‘neural network’ is simply a series of algorithms that tries to analyse data sets in a way that resembles the way the human brain operates, through a network of virtual ‘neurons’. DeepFM is a newer model that combines both Factorization Machines and Deep Neural Networks to create a predictive model. One study [10] using DeepFM among other Deep neural networking methods suggests that these neural networking methods perform better than any other machine learning techniques in all metrics tested in the study. However, perhaps surprisingly, sentiment analysis detrimentally affected the results in this case, rather than increasing the accuracy. Whilst neural networks may perform better than regression techniques, they are also far more time consuming and require a lot more processing power than normal regression techniques.

Studies involving ‘data mining’ were also taken into account. Data analysis and visualisation are becoming increasingly important disciplines as the size of databases continues to grow. There is no longer any reason for traditional, manual data analysis techniques to be used over the more modern techniques now being implemented. While traditional methods allowed a user to make queries through a prebuilt application, these queries could not be used to understand the data or to go on to then make predictions as to how future data may look. Thus data mining has become increasingly important, and has allowed for the possibility of ‘computer-driven data exploration’ [12]. This has allegedly opened up various options as to how databases can be interacted with, and has made it possible to explore data that humans would not be able to explore with the methods previously available to them. Data mining is described as a ‘necessity’ now that manual analysis techniques cannot keep up with the rate of growth of modern data sets. There are many ‘Data mining methods’ but the one being focused on in this study is Predictive Modelling. Described in [12] to be a method to ‘predict some field(s) in a database based on other fields’.

One final topic was researched in relation to this study, namely the difference between neural networking techniques and regression techniques, and which performs better and would be more suitable for use in this project. Neural networking algorithms essentially train themselves, and are able to determine which input variables are the most important in regards to a particular output. In contrast, when using a logistic regression model such as the one used in this study, the developer of the model needs to dictate and understand which variables are contributing the most to a particular output, which gives the developer the opportunity to decide for themselves on which regression method is best suited for the task. Neural network models are also (on average) more difficult to interpret than logistic regression models, on the basis that logistic regression models allow the user to easily predict and observe the probability of specific outcomes.

Finally, one very important negative to neural network modelling is the much greater computational resources needed to create the model. A well regarded 1996 study [13] states that for a neural network to converge to a low error ‘optimum learning state’ it could take ‘hours to days’, whereas logistic regression models with large datasets can be tested and evaluated almost instantly. While this study is slightly outdated, it still concisely lays out various issues that are still prevalent in regards to neural networking, and while the time convergence of neural networking methods has improved since 1996 (due to technological developments and increased processing power), it still remains true that neural networking is slower and more computationally expensive than logistic regression.

# 3 Motivation, Essential Knowledge and Problem Analysis

## 

## 3.1 Motivation

The success of this project relies on the ability to establish a link between the features associated with a movie and its rating using data mining techniques and regression. A successful algorithm will demonstrate this link by showing the ability to predict a film’s rating with a significant degree of accuracy. The potential of this system has been explored through many different fields of computer science, mainly through the use of artificial intelligence and machine learning. While studies are becoming progressively more successful, and prediction methods using artificial intelligence techniques are becoming more widely used, there have not yet been any specific applications published that can reliably make these predictions with consistent accuracy. While my own personal interest in the film industry played a significant role in why this study was conducted, I believe that the work done in this study will help to demonstrate the various different methods that can be used in predictive algorithms, and will act as a starting point for anyone interested in the field, but who has not yet researched the topic. Referenced below are the most important sources of information critical to the creation of this project.

## 3.2 Essential Knowledge

## 3.2.1 Introduction to Computer Science and Programming using Python - Kaggle course

While this Kaggle course is aimed at beginners in the field of computer science, it provides a very concise and useful introduction to data structures, which play an integral part in the data analysis and visualisation in this study. It is a very good starting point for anyone looking to begin working with big data. [14]

## 3.2.2 Aurélien Geron’s: Hands-On Machine Learning with Scikit-Learn and TensorFlow

This was the most important reference point involved in the development of this project. Aurélien Geron has been involved in many different Computer Science fields, and decided that instead of selling his works on Machine Learning, he would create a free-to-access github repository, containing Jupyter a number of notebooks. These notebooks were integral to my understanding of Machine Learning, and served as a very useful guide for how to use Scikit-learn in Python. This github repository is informative, lucid and provides explanations for every code snippet, helping to provide the reader with a better understanding as to why each process is necessary in the construction of regression prediction models. [15]

## 3.2.3 Introduction to Machine Learning - Kaggle course

Another course run by Kaggle, ‘Intro to Machine Learning’ also helped lay out a roadmap of the various processes that must be completed in order to build a machine learning model. While this course was similarly aimed at relative beginners to the field, it provides a very clear outline of the steps undertaken in this study, and how each should be broken down in order to make the implementation more manageable. This course also offers the ability to enter into Machine Learning Competitions to gauge user progress, and use what has been taught in a competitive environment. [16]

## 3.3 Problem Analysis

Before building the ridge regression model, a problem analysis was carried out in order to understand exactly how the model would function. First, data preprocessing must be implemented so that feature selection has been completed. Then, the ridge regression must be trained using cross validation with all of the selected features. Then the difference between the actual movie rating and predicted movie rating can be evaluated from the results. At which point an estimation for the model’s performance can be made.

One main problem encountered in data preprocessing was the issue of how the model would interpret features such as ‘director’ and ‘starring actors’. Normalization of the data was required to ensure that these features could be interpreted by the regression model. When a specific starring actor or director would appear on a film, the value 1 would be substituted in to indicate their involvement, and a 0 to indicate when they do not appear on a film.

## 3.4 Hypothesis

Based on the knowledge that you can predict the relationship between variables by using data recorded from the past to predict data in the future, we hypothesise that the features associated with a specific film will have some relation to that film’s rating on release. Thus, we expect to observe this relation through the construction of a working linear regression model involving IMDb variables.

# 4 Method

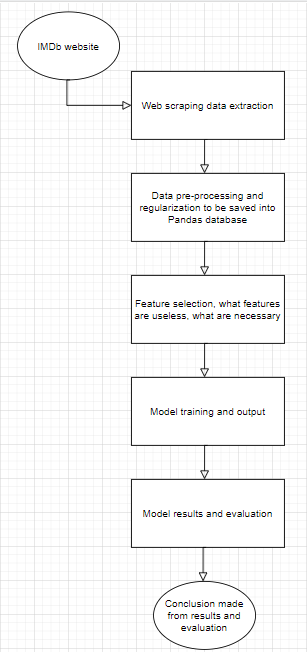
## 4.1 Overview

To test the hypothesis, the following steps were carried out:

1. Data was scraped from the Internet Movie Database (IMDb) using BeautifulSoup.
2. A dictionary was created with the information that was retrieved from each link, specifically the features that had been collected in relation to each film.
3. Using Pandas (a Python library for data manipulation and analysis) the data was passed into a dataframe.
4. Further data preprocessing was carried out, where null values in the data gathered were removed, so as to decrease the useless values in the data that were of no use to the regression algorithm, such as where there was no runtime specified. This helped to improve performance during runtime however the negative is that less data can lead to overfitting.
5. A Train Test algorithm was implemented, specifically the K-fold cross validation method explained in section 2.4.
6. Multiple regression models were tested to compare performance. For the reasons stated above, the model ultimately used in this study was the Ridge regression model.
7. The results were evaluated by calculating the R-squared value and the MAE (mean absolute error) of the model, both of which are very helpful when it comes to testing the model’s performance. As they indicate whether the model is fitting the data well, and how far the predicted values are from the actual values.
8. Having received all of this data, graph plots were made to easily display how the model had performed.
9. Further comparative analysis was carried out, including the creation of a neural network to compare the performance of linear regression methods and a Multilayer Perceptron for this specific prediction problem.
10. Another much larger dataset was tested to show how to negate the issue of overfitting due to a too-small dataset.
11. A simple gradient boosting algorithm was also implemented (xgboost), to see whether it would improve execution speed and model performance.

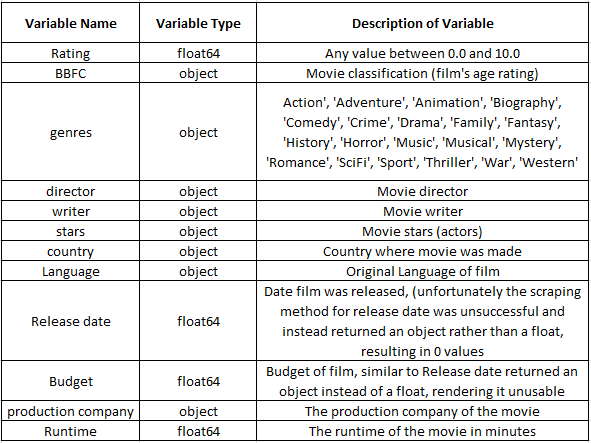
We have chosen to lay the method out as a sequence of individual steps, which an interested reader could follow in order to make it possible for them to recreate and test a similar model of their own, for comparisons and further work. The 11 steps set out above outline some of the most important functions and techniques required to build a web scraper and to test different Machine Learning algorithms, and could serve as a base for anybody looking to create a similar study of their own.

## 4.2.1 Method Diagram:



*Figure 2: Method diagram*

## 4.2.2 Data information



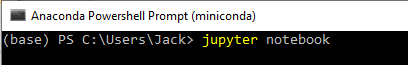
*Figure 3: Data information*

## 4.3 In Depth Method

## 4.3.1 Installing the prerequisite programs

All of the code needed for this study was produced in Jupyter Notebooks. However, due to the size of the project it was also necessary to use content from a local machine, so that it could be locally saved. To do this, Anaconda was installed, which is a Python distribution for data science, and which comes with pre-loaded libraries essential to this project. One of these pre-loaded libraries is NumPy, which is fundamental for scientific computing within Python. Other libraries used include Matplotlib (for all graph work) and pandas (which is useful for data manipulation).

After installing Anaconda and running the relevant executable, an Anaconda powershell prompt will open. To get to the Notebook dashboard, you simply need to enter the command Jupyter Notebook, as shown below:

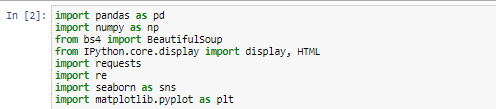


*Figure 4: How to create jupyter notebook through Anaconda*

From the Notebook dashboard, a notebook can be created and implementation can begin.

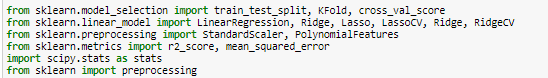
## 4.3.2 Necessary libraries

Data scraping and data visualization libraries:



*Figure 5: Libraries for data manipulation*

Model testing and evaluation libraries:



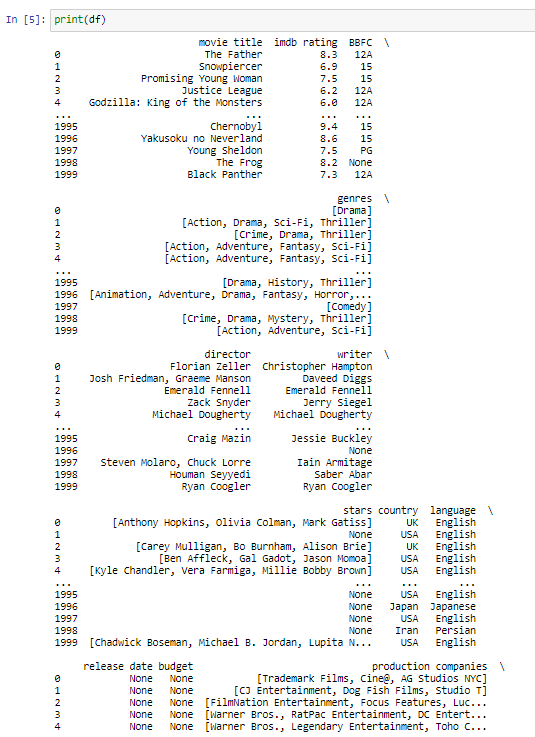
*Figure 6: Libraries for testing and evaluation*

## 4.3.3 Collection of relevant data from IMDb

Using the advanced search query function from IMDb, it was possible to create a link which contained only a specific subset of films for analysis. The link used for this project contained only movies between 2015 and 2020. For other tests, search functions were utilised to filter the films by their country of origin, their language, and various other release-date periods. In theory, this would allow the user to create a number of different datasets for testing. The function ‘get\_movie\_links’ creates a list of links to be scraped for film data. Within this function you can choose how many pages to scrape, (with each page containing 100 movies). This returns a list containing links to each film’s individual page.

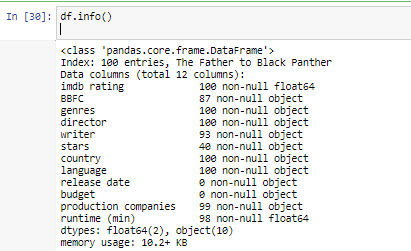
After choosing how many pages of movies to scrape, the data was scraped from IMDb and turned into a dictionary representing the specific film and its related information. This function was called ‘get\_movie\_data’. The link retrieved in the function ‘get\_movie\_links’ was appended with IMDb’s base URL "[https://www.IMDb.com](https://www.imdb.com)" which created a specific movie link, and from that link it was possible to retrieve specific information regarding the movie. Having created this movie dictionary, the data was saved locally to a pickle file, so it could be easily accessed and transformed during testing and evaluation. The data was then converted into a dataframe using pandas. Examples of these dataframes are set out below.

By requesting to print the dataframe, it returned the following:



*Figure 7: Film and feature dataframe*

This dataframe was a useful indicator that the data collection functions were working as intended. The following figure sets out further information from the dataframe:

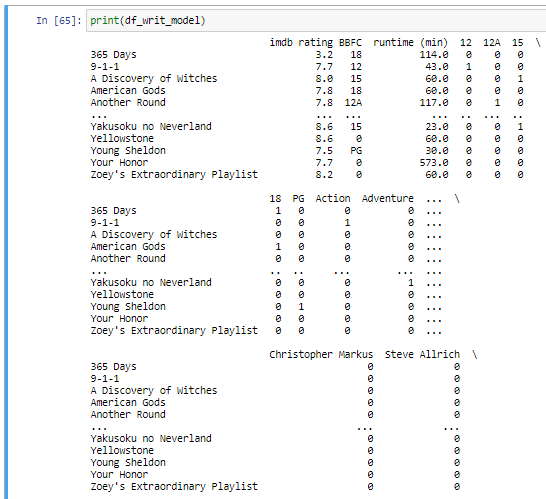


*Figure 8: Dataframe information dropdown*

It was then necessary to carry out further data preprocessing, as the linear regression algorithms used in this study cannot simply train using object variables, such as the name of a production company, or a director or starring actor. As such, hot encoding was implemented, where instead of the dataframe having these values represented as strings, they were instead represented as integers. For example, if Quentin Tarantino was to direct a specific film, the integer would be 1, and it would be 0 if he did not direct the film. To achieve this, dummy variables were created for each specific string, using the following code:

BBFC\_dataframe = pd.get\_dummies(IMDb\_drop['BBFC'])

By encoding the variables in this way, it was possible to perform linear regression on the data that was retrieved. Figure 9 below shows how these values were encoded as integer values representing where there is an instance of a director or starring actor featuring on a specific film:



*Figure 9: Encoded dataframe ready for testing*

## 4.3.4 Simple regression testing, model creation and train\_test\_split

In order to test specific models, such as a simple Linear Regression model against a Ridge or Lasso model, K-fold cross validation was used, which resulted in R-squared values that could be compared to observe model performance. Having observed which model performed better, we were then able to perform actual data visualisation, and to plot the predicted IMDb rating against the actual IMDb rating using Matplotlib. This provided a good representation of the results achieved by the model. At this point, having used a K-fold cross validation train and test split to train the model, we were able to observe and visualise the results. A separate metric tested after the model had been trained was the MAE (mean absolute error). For this project, an MAE value of 1 would suggest that, on average, the prediction of the model is within the range +- 1 of the actual IMDb rating (i.e. if the actual rating is 7.5, the model will predict within the bounds 6.5 and 8.5). As discussed above, we used 10 splits for the K-fold cross validation function in this project, as this was observed to give the best performance.

## 4.3.5 Data visualisation and results

The key value that we measured in relation to this model was the mean cross fold validation R-squared value, for both simple linear regression and for Ridge regression (so as to compare the performance of both methods). In addition, the MAE (mean absolute error) was computed as a secondary performance metric. The reasoning for choosing these options for results and evaluation is set out in section 5 (*Results*) below. Finally various plots were produced to better demonstrate the model performance, such as the model predictions vs the Target predictions.

## 4.4.1 IMDb overhaul and Neural Networking implementation

Due to the IMDb website overhaul that occurred in the final week of May, the web scraper produced for this project ceased to function, resulting in a smaller dataset than intended. For this reason, an additional neural networking model was created and trained on the same dataset, so as to be compared against the linear regression ridge model. I opted to create an MLP, (Multilayer perceptron) neural network. After creating the neural network, various different configurations were tested to optimise the model. This included changing the amount of epochs (i.e. the number of times the model trains upon the data). Multiple runs of the neural network model were carried out with varying numbers of epochs, until the optimal epoch numbers were found. These numbers were: 50 for the smaller data set produced by our scraper, and 5 for the larger Kaggle dataset (see below).

## 4.4.2 Kaggle implementation

Due to the same IMDb overhaul, which resulted in the smaller than intended dataset, it was also decided to use a premade Kaggle dataset to demonstrate how a much larger dataset can significantly reduce the issue of overfitting data. The dataset retrieved from Kaggle was a csv file containing 85,855 movies with 49 attributes. [18] The same data preprocessing techniques were used on this dataset. However, the only features added were genre and runtime, as the entire code block had to be changed to account for different variable identifiers and tags. The same linear regression algorithms were run on this dataset, and a neural network was also implemented. These steps were integral in showing the advantages and disadvantages of having small and large datasets.

# 5 Results

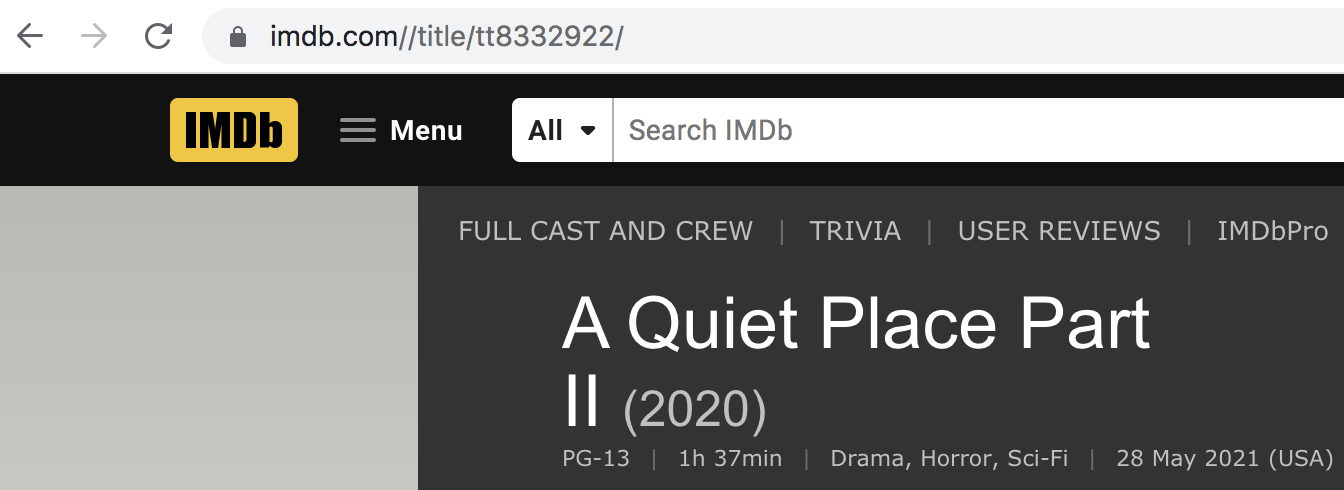
## 5.1 Data collection

The web scraper’s first job was to collect title links to be appended to the base url of IMDb ‘[https://www.IMDb.com](https://www.imdb.com)’, these title links appear in the form shown below:



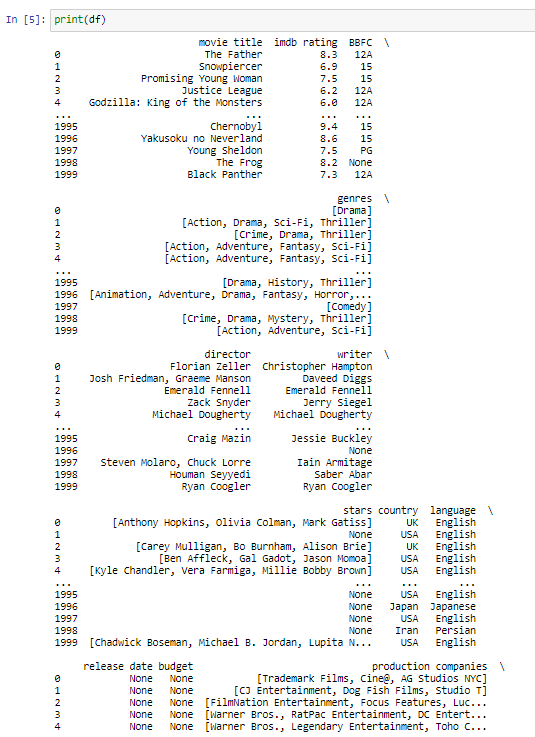
*Figure 10: IMDb links retrieved by scraper*

If we append ‘/title/tt8332922’ to the base IMDb url, it takes us to:



*Figure 11: Full url and link concatenation test*

It was possible to conclude from this that the links were being collected as intended. From these links the function ‘parse\_IMDb\_features’ was then called which created a dictionary of all the necessary features concerning the specific film. These dictionaries were then turned into a dataframe containing all of the films on the IMDb search page that the web scraper was used upon. Using the print function to inspect that dataframe returned the following result:



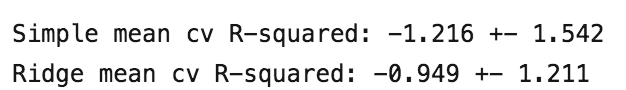
*Figure 12: Dataframe inspection*

The dataframe was then converted into a pickle file labelled ‘IMDb\_data\_test’ so that the web scraper did not need to be repeatedly run (as it took a long time to execute). This had the fortunate, but unintended, effect of ensuring that we still had a saved dataset after the IMDb overhaul, which caused the scraper to cease functioning. Various plots of the IMDb rating against runtime were then created using seaborn, a library for making statistical graphics in Python.

Finally, the data was appropriately encoded for the linear regression algorithm and a neural network. This was done during feature selection where dummy variables were created for object variables such as ‘director’, ‘genre’ and ‘writer’, which variables were sequentially added to a new dataframe until the chosen features were all contained and the linear regression algorithms could be trained upon the appropriate data.

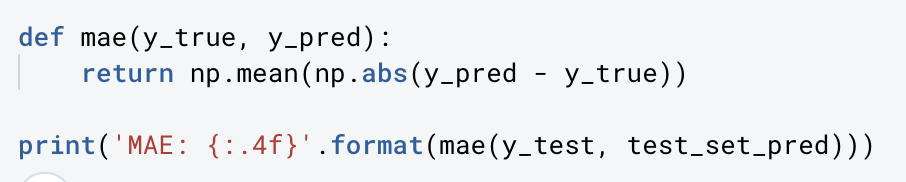
## 5.1.2 Train and test split, linear regression comparisons, metric selection

Subsequently, the simple linear and ridge regression models were trained and tested using K-fold cross validation. Both simple linear regression and ridge regression were used so that the methods could be compared. Here we opted to compare the R2 values, which is a statistical measure of how close the data is to the fitted regression line. Essentially, it is a measure of how well your model fits your data. This resulted in values of:



*Figure 13: R-squared values*

An ideal R2 value would be between 0 and 1. These negative values are potentially a result of the model overfitting of the data due to the small dataset. What is clear, though, is that Ridge regression is clearly outperforming simple linear regression (as the R2 value is closer to 0).

A MAE (mean absolute error) of 1.0257 was also calculated when running this algorithm, using the code:   
  


*Figure 14: Code to find Mean Absolute Error*

This result suggests that there was an average of +-1.0257 inaccuracy in predicting the rating of the film. So if the film was rated 8, the prediction would be somewhere in the range 8 +-1.0257. Given that a completely random predictive model should result in an MAE of +-5.0, this showed that the model was clearly performing with some degree of accuracy.

Having trained the regression algorithm we were then able to plot a graph of the actual IMDb ratings vs the Model IMDb predictions, resulting in:



*Figure 15: Web scraped model predictions vs actual ratings*

Unfortunately, it appears from this graph that the line of best fit isn’t actually through what the best fit for the data would be, and instead is just a y=x line, suggesting that the model has not fit the data as intended.

## 5.1.3 Kaggle dataset implementation for better R2 values

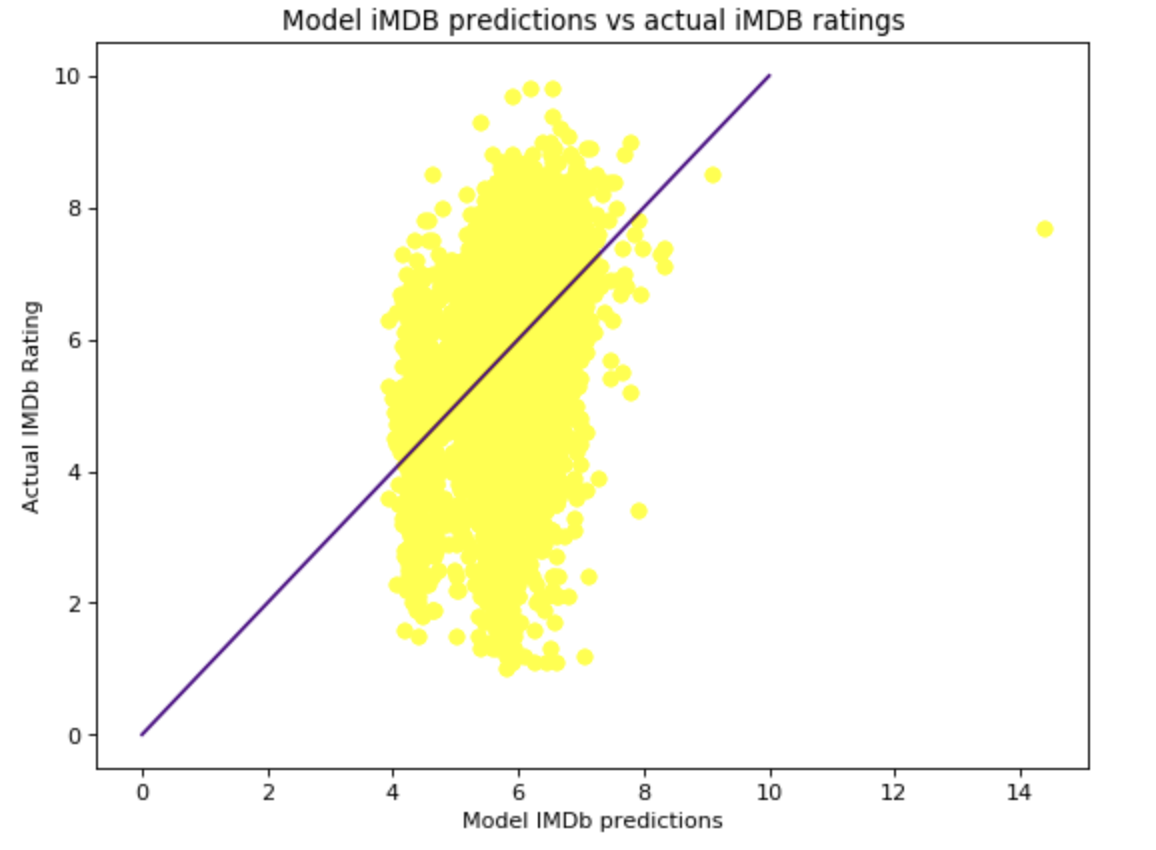
Given the R2 value suggested that the chosen model was poorly fitting the data, likely due to overfitting as the dataset was too small, a decision was made to use the same linear regression algorithms on a much larger Kaggle dataset [18], and to again compute the R2 value and MAE. However the only features added in this model were the genre and runtime, as opposed to the much larger number of features used in the dataset obtained from the web scraper.

The use of the same train and test split and regression methods on a larger dataset resulted in R2 values of:



*Figure 16: Kaggle dataset R-squared values*

While marginal, this shows that Ridge regression is still performing better than simple linear regression, and the respective plot showed a higher amount of clustering around the line of best fit, suggesting that the model fit the data better with a larger dataset:



*Figure 17: Kaggle model predictions vs actual ratings*

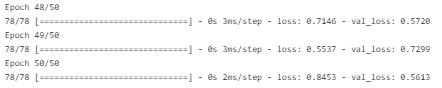
Here, again there seems to simply be a y=x line of best fit, and not a line that actually shows the best fit for the model, there is also an anomalous value, where the model predicts a film’s rating to be over 14 which is impossible. In a future model, an upper bound of 10 should be specified.

## 5.1.4 Addition of a multilayer perceptron Neural network to achieve a better MAE

In a study published in 1997 [19] it was suggested that a Neural Network ‘may be thought of as a simplified model of the networks of neurons that occur naturally in the animal brain’. Neural networks are essentially optimisation problems. In the MLP we used, a loss value was calculated (which is equivalent to the MAE) by using backpropagation over a number of epochs. By repeatedly minimising the loss function over each iteration, you can eventually converge on a low value. An epoch is defined as a ‘hyperparameter of gradient descent that controls the number of complete passes through the training dataset.’ More simply, it is the number of iterations that the neural network will complete before outputting a final value.

Shown below are the results from using the Neural network on both the Kaggle dataset and web scraped dataset:

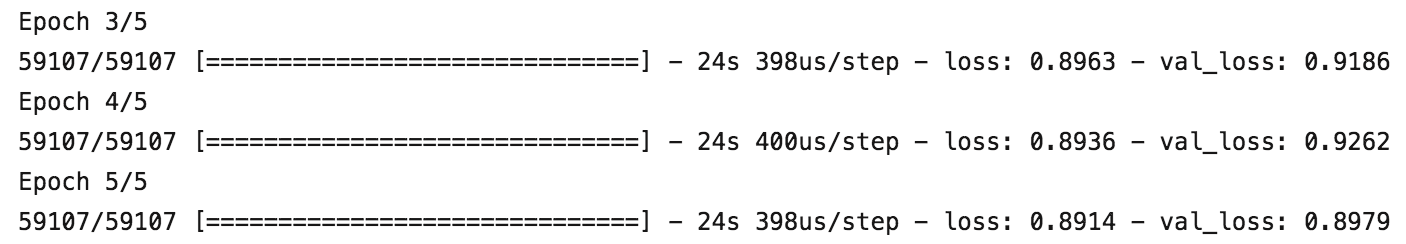
Web scraped dataset with 50 epochs due to lower amount of data:



*Figure 18: Web scraped neural network MAE results*

Here the MAE (val\_loss) is 0.5613 by the final iteration, showing a large improvement over the linear regression MAE.

Kaggle dataset with only 5 epochs due to high amount of data:



*Figure 19: Kaggle neural network MAE results*

Here we can see that even though the Kaggle dataset is performing better when calculating an R2 value, it is outputting a worse MAE than the web scraped dataset. However, the difference between the loss and the val\_loss is more consistent, suggesting that there is a lack of overfitting. This is in line with the way that the larger dataset was performing throughout our testing.

All of the tests for this study were run through deepnote, and the results will be maintained and can be accessed through their respective links. [20]

# 6. Evaluation

## 6.1 Data collection and preprocessing:

The initial web scraping algorithm performed as intended and collected the necessary links and features from IMDb. With more time, it would have been preferable to create a data scraper that could iterate through search pages on IMDb, so as to collect more films and create a larger dataset. The data preprocessing all functioned correctly and made it possible for the IMDb data to be saved within a dataframe in such a way that linear regression could be performed on the relevant features. The data preprocessing was time consuming, as each feature needed to be added individually so that the data could be properly encoded.

## 6.2 Metric selection and model evaluation

The negative R2 value from the web scraped dataset indicated that a model containing only 100 movies was too small, and thus the model poorly fit the data, resulting in overfitting. To combat this, it would be best to reuse the web scraper to scrape a much larger dataset of around 2000+ movies to perform the regression and Neural Networking techniques upon. However, due to the unfortunate IMDb website overhaul, the web scraper would need to be fully reformatted to account for all of the HTML changes linked to the website update. Given the time constraints of this project, this was unfortunately not possible. That being said, after analysing the much larger dataset from Kaggle, it is clear that a larger dataset resulted in a much better R2 value. This could potentially have given an even better result if the dataset had included the same number of features as the web scraped model, but further testing would have to be done to investigate this hypothesis. With a more complete dataset, we could further investigate this claim by testing the model with various numbers of features added, and to then evaluate if the model actually performs better with a larger dataset containing more features.

The MAE (mean absolute error) of 1.0257 when using the web scraped dataset suggests that the model was making predictions of the film's rating within a consistent range around the actual rating of the film. Theoretically, an MAE value of 5.0 would suggest that the model was randomly predicting values for the films rating, with no correlation to the actual film rating. While 1.0257 is a relatively good initial MAE, it could likely be improved again with the use of a larger dataset. The Kaggle dataset suggests that this hypothesis is true, since it returned a better MAE than the web scraped dataset. The use of a neural network also further improved the MAE value, suggesting that in future implementations it may be optimal to only use a neural network rather than linear regression algorithms.

Although the initial models on the web scraped dataset did not perform exactly as intended, the subsequent tests using larger datasets and different Machine learning methods showed immediate improvements, and suggested that using these techniques could help us to predict the ratings of films in the future, especially with further testing.

One large issue with web scraping on a whole has been demonstrated within this project, namely that a web scraper is only functional as long as a web page does not change significantly. An implementation of this sort would constantly have to be monitored, so that it could be updated whenever there is an update to the website it is being used upon. This would be very time consuming, due to the constant updates needed to be made to the web scraping code.

# 7 Conclusion and Further Work

Over the course of this project a hypothesis was developed, and a testing method for that hypothesis. From the results, we are able to suggest that there is a clear link between the features associated with a film and the rating it receives upon release. This conclusion was seen in all of the methods we used (to different extents). In the regression testing of the Kaggle dataset, the low R2 value suggests that the model was beginning to closely fit the data, and an obvious correlation was seen between features and rating. There were also examples in the web scraped dataset of convergence. Similarly, the implementation of a neural network showed that values such as the MAE could be improved upon using different coding techniques.

Whilst we are unable to show a consistent link between all methods, this project could still provide value to the interested reader, as all of the methods tested could be built upon further to identify this link easier in the future. To be specific, further work in this area should aim to build a web scraper capable of retrieving more data, so as to avoid overfitting due to the smaller dataset. By utilising a much larger dataset, the model would have access to more data to train and test upon, hopefully improving its accuracy of predictions. Another potential improvement to this project could be to increase the amount of features used to make the predictions, such as adding a film’s budget or release date, and it could be taken even further by implementing sentiment analysis, so that key phrases involved in explaining the film’s plot or premise could be analysed and used by the model when it makes predictions.

Considering the vast amount of data contained on IMDb, it is relatively time-consuming to continuously update a web scraping algorithm to function on a constantly updating platform. This should be taken into account by anybody aiming to carry out a similar project in the future, with the aim of collecting similar datasets.

If in the future a consistent model were to be trained upon the same constraints used in this project, it would likely open up the avenue to further research upon the topic, where this model could be used to create a platform that could predict the rating of any film to be released in the near future with a high degree of accuracy. This could help the movie industry in multiple ways, such as allowing production companies to know which features help a film to perform as well as possible, or allowing the general public to make decisions upon whether it is worth going to see certain films.

# Bibliography

[1] Zheng, C., He, G. and Peng, Z., 2015. A Study of Web Information Extraction Technology Based on Beautiful Soup. *JCP*, *10*(6), pp.381-387.

[2] Singh, A., Thakur, N. and Sharma, A., 2016, March. A review of supervised machine learning algorithms. In *2016 3rd International Conference on Computing for Sustainable Global Development (INDIACom)* (pp. 1310-1315). Ieee.

[3](2021) IMDb, [Online]. Available: [https://www.IMDb.com](https://www.imdb.com) (visited on 27/05/2021).

[4] Weisberg, S., 2005. *Applied linear regression* (Vol. 528). John Wiley & Sons.

[5] Lee, L.S., Lee, S.H. and Hwang, H.S., 2017. A study of evaluating of game review data using multiple regression. *International Journal of Innovation, Management and Technology*, *8*(2), p.95.

[6] Latif, M.H. and Afzal, H., 2016. Prediction of movies popularity using machine learning techniques. *International Journal of Computer Science and Network Security (IJCSNS)*, *16*(8), p.127.

[7] Liu, Y., Huang, X., An, A. and Yu, X., 2008, December. Modeling and predicting the helpfulness of online reviews. In *2008 Eighth IEEE international conference on data mining* (pp. 443-452). IEEE.

[8] Sobkowicz, A. and Stokowiec, W., 2016, May. Steam Review Dataset-new, large scale sentiment dataset. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016) Workshop Emotion and Sentiment Analysis* (pp. 55-58).

[9] Ma, N., Zhao, S., Sun, Z., Wu, X. and Zhai, Y., 2019. An improved ridge regression algorithm and its application in predicting TV ratings. *Multimedia Tools and Applications*, *78*(1), pp.525-536.

[10] Cheuque, G., Guzmán, J. and Parra, D., 2019, May. Recommender systems for Online video game platforms: The case of STEAM. In *Companion Proceedings of The 2019 World Wide Web Conference* (pp. 763-771).

[11] Hoerl, A.E., Kannard, R.W. and Baldwin, K.F., 1975. Ridge regression: some simulations. *Communications in Statistics-Theory and Methods*, *4*(2), pp.105-123.

[12] Fayyad, U. and Stolorz, P., 1997. Data mining and KDD: Promise and challenges. *Future generation computer systems*, *13*(2-3), pp.99-115.

[13] Tu, J.V., 1996. Advantages and disadvantages of using artificial neural networks versus logistic regression for predicting medical outcomes. *Journal of clinical epidemiology*, *49*(11), pp.1225-1231.

[14] (2021). edx Course, introduction to Computer Science and Programming, [Online]. Available: <https://www.edx.org/course/introduction-to-computer-science-and-programming-7>

[15] (2021). github repository created by Aurélien Geron, the fundamentals of Machine Learning and Deep Learning in Python using Scikit-Learn, Keras and TensorFlow 2.[Online].   
Available: https://github.com/ageron/handson-ml2

[16] (2021). Kaggle course, Intro to Machine Learning. [Online]  
Available: https://www.kaggle.com/learn/intro-to-machine-learning

[17] (2021). 50 highest grossing movies between 2010 and 2020. [Online]  
Available: https://inews.co.uk/culture/film/highest-grossing-movies-films-box-office-success-2010-2019-last-decade-381194

[18] (2021). Kaggle movies dataset. [online]   
Available: [https://www.kaggle.com/stefanoleone992/IMDb-extensive-dataset?select=IMDb+ratings.csv](https://www.kaggle.com/stefanoleone992/imdb-extensive-dataset?select=IMDb+ratings.csv)

[19] Gurney, K., 1997. *An introduction to neural networks*. CRC press.

[20] (2021) deepnote code links. [online]

Available: (Kaggle Dataset)   
<https://deepnote.com/project/Final-working-Kaggle-and-webscraper-lin-reg-and-Neural-Networking-project-work-gH_eSLdIT3afLmq728aCEA/%2FKaggle_dataset_working.ipynb>

Available: (Web scraped Dataset)  
<https://deepnote.com/project/Final-working-Kaggle-and-webscraper-lin-reg-and-Neural-Networking-project-work-gH_eSLdIT3afLmq728aCEA/%2Fweb%20scraped%20dataset.ipynb>

# List of figures

[Figure 1: Example linear regression plot 4](#_4d34og8)

[Figure 2: Method diagram 13](#_3whwml4)

[Figure 3: Data information 14](#_qsh70q)

[Figure 4: How to create jupyter notebook through Anaconda 14](#_49x2ik5)

[Figure 5: Libraries for data manipulation 15](#_147n2zr)

[Figure 6: Libraries for testing and evaluation 15](#_3o7alnk)

[Figure 7: Film and feature dataframe 16](#_ihv636)

[Figure 8: Dataframe information dropdown 17](#_32hioqz)

[Figure 9: Encoded dataframe ready for testing 18](#_1hmsyys)

[Figure 10: IMDb links retrieved by scraper 20](#_2u6wntf)

[Figure 11: Full url and link concatenation test 20](#_19c6y18)

[Figure 12: Dataframe inspection 20](#_3tbugp1)

[Figure 13: R-squared value comparison 21](#_nmf14n)

[Figure 14: Code to find Mean Absolute Error 22](#_37m2jsg)

[Figure 15: Web scraped model predictions vs actual ratings 22](#_1mrcu09)

[Figure 16: Kaggle dataset R-squared values 23](#_2lwamvv)

[Figure 17: Kaggle model predictions vs actual ratings 23](#_111kx3o)

[Figure 18: Web scraped neural network MAE results 24](#_206ipza)

[Figure 19: Kaggle neural network MAE results 24](#_4k668n3)