Thesis Project, 15 credits, for the degree of Bachelor of Software Development with Major in Computer Science

Spring Semester 2021

Using objective data from movies to predict other movies’ approval rating through Machine Learning

Subheading if used [Arial 22p]

Iñaki Zabaleta de Larrañaga

Author [Arial 10p Bold]

Iñaki Zabaleta de Larrañaga

Title [Arial 10p Bold]l

Using objective data from movies to predict other movies’ approval rating through Machine Learning

Supervisor [Arial 10p Bold]

Ola Johansson

Examiner [Arial 10p Bold]

Kamilla Klonowska

Abstract (maximum 250 words) [Arial 10p Bold]

Abstract text [Arial 10p]

Keywords (5–8 words) [Arial 10p Bold]

Aaaaa, Bbbbbb, Cccccc, Ddddddd, Eeeeeee, Ffffff [Arial 10p]

Table of Contents

[1. Introduction 5](#_Toc71295975)

[1.1 Related Work 5](#_Toc71295976)

[1.2 Research Questions 6](#_Toc71295977)

[1.3 Aim and Purpose 6](#_Toc71295978)

[1.4 Methods 6](#_Toc71295979)

[1.5 Thesis Structure 6](#_Toc71295980)

[2.Background 7](#_Toc71295981)

[2.1 Machine Learning Paradigms 7](#_Toc71295982)

[2.2 Machine Learning Regression Algorithms 8](#_Toc71295983)

[2.2.1 Linear Regression 8](#_Toc71295984)

[2.2.2 K-Nearest Neighbor 8](#_Toc71295985)

[2.2.3 Decision trees 8](#_Toc71295986)

[2.2.4 Neural Networks 9](#_Toc71295987)

[2.2.5 Support Vector Regressor 10](#_Toc71295988)

[3. Dataset 10](#_Toc71295989)

[3.1 Data Exploration 11](#_Toc71295990)

[3.1.1 Movies 11](#_Toc71295991)

[3.1.2 Artists 13](#_Toc71295992)

[3.2 Data Cleaning 13](#_Toc71295993)

[3.2.1 Missing Values 13](#_Toc71295994)

[3.2.2 Eliminating Redundant Data 14](#_Toc71295995)

[3.3 Data Preparation 14](#_Toc71295996)

[3.3.1 Adapting Categorical Columns 14](#_Toc71295997)

[3.3.2 Merging Movies with Artists 16](#_Toc71295998)

[3.4 Final Dataset 16](#_Toc71295999)

[4. Methodology and Experimentation 17](#_Toc71296000)

[4.1 Tools Used 17](#_Toc71296001)

[4.1.1 Hardware 17](#_Toc71296002)

[4.1.2 Software 17](#_Toc71296003)

[4.2 Algorithm Evaluation 17](#_Toc71296004)

[4.3 Algorithm Optimization 19](#_Toc71296005)

[4.4 Challenges and Limitations 20](#_Toc71296006)

[5. Results 20](#_Toc71296007)

[5.1 Testing and Optimizing Models 20](#_Toc71296008)

[5.1.1 Ridge Regression 20](#_Toc71296009)

[5.1.2 Lasso Regression 20](#_Toc71296010)

[5.1.3 K-Nearest Neighbor 20](#_Toc71296011)

[5.1.4 Decision Tree Regression 20](#_Toc71296012)

[5.1.5 Random Forest Regression 20](#_Toc71296013)

[5.1.6 Extreme Gradient Boosting 20](#_Toc71296014)

[5.1.7 Artificial Neural Network 20](#_Toc71296015)

[5.2 The Best Models 20](#_Toc71296016)

[6. Discussion and Analysis 21](#_Toc71296017)

[7. Conclusion 21](#_Toc71296018)

[8. Bibliography 21](#_Toc71296019)

# 1. Introduction

Machine Learning is a subfield of Artificial Intelligence, it addresses the question of how to build computers that improve automatically through experience [1]. There are four main paradigm types of Machine Learning: Supervised Learning, Unsupervised Learning, Semi-supervised Learning and Reinforcement Learning. The one that will be focused on is Supervised Learning can be summarized as: “where the algorithm generates a function that maps inputs to desired outputs.” [2] In this project, objective data from movies such as the runtime, director, actors, region, even the actors’ previous jobs will be used to predict subjective numbers.

The dataset for this will be provided from the biggest movie dataset in existence, IMDb directly [3] which is updated daily with more information on films and as of March 2021 has almost 8 million titles. IMDb is a website that lets their users inform themselves about movies and TV shows, but they also provide a 10-star rating system based solemnly on their users. This rating is what will be attempted to predict.

## 1.1 Related Work

Similar research has been conducted previously which was inevitable with the current growth of Machine Learning and movie industry’s constant growth.

Kyuhan Lee et al. [8] used naver.com for American movies and IMDb for foreign ones. Their aim was however to predict the movies’ performances at the box office. This means that their data is much more budget oriented as well as where the movie was being played and how many theaters had screenings. They do not explicitly describe their input data and they had a 58.5% accuracy by classifying the movies on how well they performed.

Dorfman, R et al. [9] did a research project where they did a similar experiment but the other way around. They took subjective data to create a model that predicted objectivity. For this purpose, pictures of surgical patients before and after a rhinoplasty surgery were taken and put into their created age predicting models to find the effect of the surgery to their apparent age. They found that the rhinoplasty did make on average patients look 3 years younger. This research shows that it is indeed possible to predict objective aspects based on subjective variables.

Xiaodong Ning et al. [10] used convolutional neural networks based regression for predicting a movie rating as well but they use Natural Language Processing to read the plot of the movie as well which adds many more variables to consider and could be inconsistent since they are getting their plots from IMDb which are done by users, not the actual movie producers.

Yueming Zhang [11] experimented with similar data but also datamined facebook likes of the movie, actors and directors but used information that would only be available after the movie comes out, such as number of votes and gross earnings. The algorithms used were Decision trees, K-NN and Random Forests, the results were a bit over 0.7 for all three. 0.7 means that the predictor was in average 0.7 rating points away from the real rating.

## 1.2 Research Questions

* 1.2.1 Which algorithm is most applicable for predicting movie ratings and how do they compare?
* 1.2.2 How accurately can Supervised Machine Learning techniques predict subjective values like a movie’s ratings by using movies’ objective data?

## 1.3 Aim and Purpose

Movie taste is very complicated as well as always changing at a personal level, however in a general level, it is more consistent and less variable. This opens the opportunity of studying that consistency and create predictions with the help of Machine Learning. The development of this model will explore the relationships between the many variables in movies and with that not only predict ratings better but also see which actors work better together for a higher rating.

## 1.4 Methods

https://www.analyticsvidhya.com/blog/2021/01/a-quick-overview-of-regression-algorithms-in-machine-learning/

## 1.5 Thesis Structure

# 2.Background

Machine Learning focuses on machines teaching themselves with data provided to them, however there it gets much more complicated as it is learned how that it is prepared and executed. For building a Machine Learning model, it must first be determined what is expected from the model or what it is trying to solve. Understanding the data that will be used for training the model is essential to know the objective. The five types of problems generally fall on one of these groups: [4]

1. Classification Problem: When the output needs to be classified into a limited amount of groups or a number.
2. Anomaly Detection Problem: The model monitors something learning patterns to later detect anomalies.
3. Regression Problem: The output is numeric and continuous, most of the times it is represented in trend graphs, their goal is usually avoiding diminishing returns or improve profits.
4. Clustering Problem: Similar to classification but it is a form of unsupervised learning where it looks for patterns to attempt to build clusters. New data goes into the build clusters.
5. Reinforcement Problem: When decisions need to be done based on previous experiences, generally learned on an environment. It is reliant on trial and error for knowing what are the right decisions to take being ”rewarded” for right decisions and sometimes ”punished” for the wrong ones.

## 2.1 Machine Learning Paradigms

There are many different types of data or situations that determine which machine learning paradigms to use:

1. Supervised Learning: https://somedudesays.com/2020/09/the-3-basic-paradigms-of-machine-learning/
2. Unsupervised Learning: works with unlabeled data which means there is no test data, commonly used with clusters.[5]
3. Semi-supervised Learning: https://insights.sap.com/what-is-machine-learning/
4. Reinforcement Learning

## 2.2 Machine Learning Regression Algorithms

Machine Learning relies on algorithms which have complex and advanced mathematics backing them up with each algorithm being better depending on the desired outcome and the data that is being fed to it. This project focuses on predicting the movies’ rating, which are a numeric and continuous variable, therefore the algorithms to be used will be regression algorithms. Since the data available is already labeled, the model will use supervised learning to train.

### 2.2.1 Linear Regression

Linear Regression is a very popular algorithm that works by trying to draw a line through the training data and using the line to predict for different inputs. There are however two other linear algorithms that are more convenient for more complex data like the one that will be dealt with:

**Ridge Regression**

Ridge regression stabilizes linear regressions by adding a constant to estimate the coefficients used in the model, also known as a bias. Hence, it is lowering variance and shrinkage in coefficients which also reduces the model’s complexity [19].

**LASSO Regression**

LASSO Stands for the Least Absolute Shrinkage and Selection Operator. The goal is to identify the variables and corresponding regression coefficients to minimize prediction errors. This is achieved by constraining on the model parameters, “shrinking” the regressing coefficients to zero [18].

### 2.2.2 K-Nearest Neighbor

K-Nearest Neighbor is an algorithm that stores all the training data in a n-dimensional space which means it is memory based [13]. Once an input is sent in, the model looks through the data to find the nearest k training examples and assigns the label based on those. The main advantage is it is efficient even with large test data, but its computation cost is very high.

### 2.2.3 Decision trees

**Decision Tree Regressor**

It builds the model as a large decision tree; it traverses it by breaking down the data until a termination point is reached, which assigns the value for the output. SOURCE

**Random Forest Regressor**

For standard trees, the nodes are split by using the best split among all variables but for a random forest, the best among a subset of predictors, know as gradients, are chosen randomly to be used for the split and then in a form of voting, reach the final prediction. This might sound counter-intuitive since the choosing is random but usually performs better [14].

**Extreme Gradient Booster**

Gradient boosting takes many trees to make an ensemble of them similar to random forest regressor but then uses the gradient to influence the predictions towards the correct values. Extreme Gradient Booster (XGBoost) takes it a step further by many hardware improvements and built for large datasets. It uses the data to extract potential splitting points based on feature distrbutions and assigns continious values into a bucket of values to be closer to the feature, greatly reducing the amount of splitting points. Since XGBoost was made having big datasets in mind, it is aware that most memory would not be able to handle it, so it compresses the data with a separate process to store it in the disk and decompresses it when loading back into the main memory [20]. It is one of the few algorithms capable of using null data as actual data which can store more information than imputed data.

### 2.2.4 Neural Networks

They are based on a brain’s structure using neurons and they can get very complicated. Neural Networks belong to a whole subfield of machine learning called deep learning and there are many ways of approaching them. The structure of a neural network is having several neurons on layers. There are three types of layers, the input, the hidden and the output layers. The neurons rely on weights and biases to evaluate the value of each input and propagate through the hidden layers until the output layer is reached with a hopefully correct classification [17].

### 2.2.5 Support Vector Regressor

It works by segregating the training data into different classes within a space setting a form of boundary called the hyperplane, then any input will fall into a specific class. The space between the two closest points of different classes is called the margin. [16]

Diagram, scatter chart

Description automatically generated

Figure xx: That’s a hot ass

# 3. Dataset

The whole dataset must be explored, cleaned, and prepared for a machine learning model to be able to work on it. This is a crucial part of any form of machine learning as the data is the only thing that it is provided. It is also important to figure out efficient ways to display data as adding redundant data will make the model more complex and longer to create while not necessarily providing any more accurate results.

IMDb updates all their datasets daily and they are divided into seven different sections which are:

* name.basics.tsv.gz : contains all artists that work on a movie, from actors to editors to writers and directors. What their jobs are, the titles they are known for, and age are also listed here.
* title.akas.tsv.gz : contains information on the title of the movie and region and language Size: 25 million x 8
* title.basics.tsv.gz : contains release date, genres, runtime and if the movie is adult rated. Size: 7.7 million x 9
* title.crew.tsv.gz : contains the director(s) and writer(s) of each movie (it is found in principals therefore it is not used)
* title.episode.tsv.gz : contains data on which title each episode belongs to (does not apply)
* title.principals.tsv.gz : contains which people did what
* title.ratings.tsv.gz : contains the ratings and number of votes. Size: 1.1 million x 3

## 3.1 Data Exploration

The data in total can be classified into two fields, movies and actors.

### 3.1.1 Movies

Starting off with title.basics, it contains all the information of any type of film in IMDb, that includes TV shows, movies, shorts, etc.. The first thing to do is to cut off anything that is not a movie. Once that is done, for all of the movies data it is needed to merge the new title.basics, title.akas and title.ratings which combined have a total size of 1.6 million x 18 which would normally not make sense since title.ratings has 1.1 million rows but that is because every movie that gets their title translated is listed but keeps the same titleId. The solution for this is to remove all duplicates where titleId is the same. Removing the duplicates brings the dataset to a total of 260,992 movies. However, this counts any movie that was ever done, no matter how unpopular it might be which means it could alter the predictions. Fortunately, since IMDb is a website where any user can rate a movie and the title.ratings dataset provides the number of votes, that can be used to eliminate unpopular movies that would hinder the Machine learning model. With that in mind, only movies with more than 1000 votes will be considered and the rest will be dropped from our data, bringing the data to a total of 32927 movies.

There are a few other things that must be taken into consideration but are more related to the specific aim of this project rather than to the data itself. IMDb contains data on all movies, that includes adult rated movies which are not an aim for this project. IMDb does provide a column for those movies so they can easily be removed. This is a very insignificant alteration to the data, reducing the total amount of movies by 20. This is due to that most adult rated films do not have a lot of user votes so most of them were already cut out.

Another alteration that must be made to fit this project’s goal is to only count films that could have the same people working on them to predict future movies. For this to be achieved, old movies should not be considered. The solution is to remove movies that were released before the 1970s which is also not a very significant amount since not many movies were released back then and many of these movies have missing values. This brings the final count to 28454.

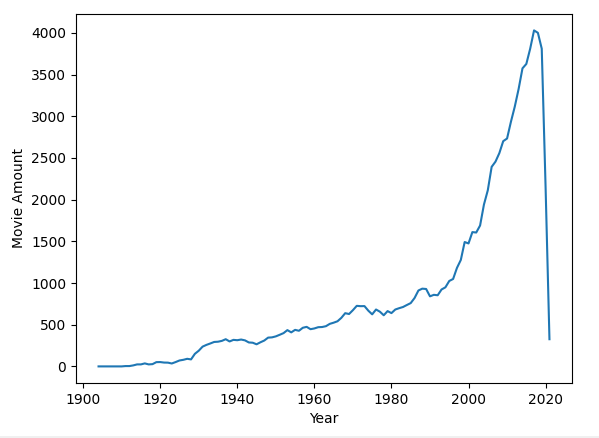


Image Text Figure 1

This graph shows the number of movies done per year which was constantly increasing until 2020 when most film releases and production got delayed due to the COVID-19 Pandemic [7]. 2021 also shows low numbers but that is because the data was taken in April 2021.

### 3.1.2 Artists

Artists have only one dataset with a full list of them and their information such as birth year and death year if any. With these two fields, it is possible to determine if the artist is alive which would be relevant for trying to predict upcoming movies. To assure this, a simple if statement is sufficient so if deathYear is null, the actor is alive. Some artists do not have a birthYear and upon inspection, those artists were either very unknown or so old, there was no data on it. With this in mind, those artists were also removed to avoid redundant data.

## 3.2 Data Cleaning

### 3.2.1 Missing Values

Missing values are a guarantee in Machine Learning when cleaning the data, there are a few approaches that can be used to handle this problem but determining which approach is best, can sometimes be difficult. The three main approaches are: [6]

* Case deletion: The rows or columns with missing data get deleted under certain circumstances.
* Single Imputation: If there is a field with missing data, all data from other rows can be used to determine what is most suitable for filling the missing field. This means it could be the mean, median or mode. Certain times it is also possible to replace the missing value with a 0 but that is more dependent on what the column means than other values in the dataset.
* Multiple Imputation: Consists of filling the data just as in single imputation but also adds a column or a new datasheet where it is marked the missing values that were filled. It is generally the best for small datasets.

Starting with the movies, after selecting the relevant values, the missing values within them were counted and it was found out that 82% of the movies were missing their language, therefore the language value was immediately dropped, this being a case deletion approach.

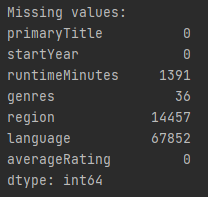


Image text Figure 2

### 3.2.2 Eliminating Redundant Data

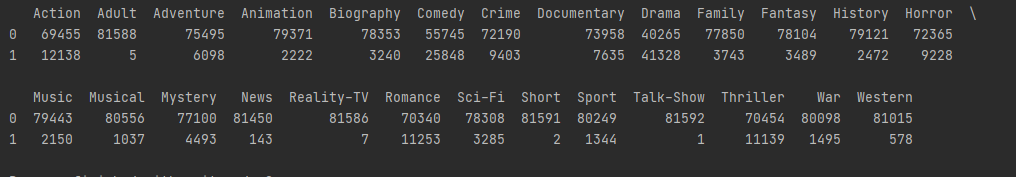
With movies data having been cut dramatically due to only keeping relatively recent movies as well as movies with certain number of votes and genres, any actors that form part of those movies that do not participate in movies that are kept, can be dropped since they would not be important later anyway. This is where the title.principals dataset is extremely important since it lists the most important artists for each movie, usually 4 actors, the director or directors and sometimes the lead producer or writer. The approach is to eliminate all the movies that were not in the new movies data and then go to the artists dataset and remove all the artists that are not in the new title.principals dataset.

## 3.3 Data Preparation

### 3.3.1 Adapting Categorical Columns

The region value as it can be seen in Figure 2, has also a quite high missing rate but a different approach to the language value was taken. The approach is called One Hot Encoding. One-Hot Encoding is a way of transforming categorical variables into vectors where all components are 0 or 1, this in turn would add n-1 columns to the table, where n is the number of classes to be used and it is minus 1 since the original column is deleted [12]. How it was used here, since there are over a hundred different regions and adding a hundred columns to the data would make it unnecessarily big, only the most frequent regions were assigned to a column and all the remaining regions were sent into a column named “uncommonRegion” All Regions with over 10 movies were chosen, bringing to a total of 69 regions plus the uncommonRegion column.

Similarly for genres, a form of One-Hot Encoding was used called Multiple Label Binarizer. Just like in One-Hot Encoding with the difference being that it is able to split the values in one same row which is necessary since many movies have more than one genre. Fortunately, there are not as many total genres as there are regions, so almost all genres were kept.



The removed genres were [Adult, Reality TV, Short and Talk-Show] since they had less than 10 movies that classified as such.

On the actors data, there is a column for what their most frequent jobs in movies. This is a similar situation to genres since a single person can have multiple jobs, however there are 41 listed jobs, while there were only 15 genres and all genres are very influential on a movie, while not all jobs are. To remove all useless data, it must first be known which jobs are the least repeated. By using Multiple Label Binarizer as well, the categorical variables were converted. Then counting the 0s and 1s of each column provides a list to see wherever there is a low number for 1s, that column is then considered useless.

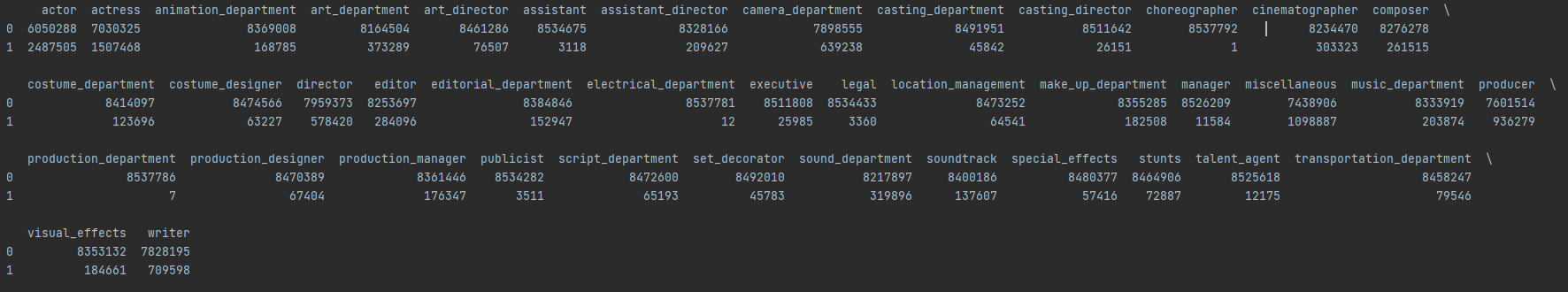


Image Text Figure 3

With this image it can be seen that [assistant, choreographer, electrical\_department, legal, manager, production\_department, publicist, talent\_agent] have a small number so those will be dropped from the data.

### 3.3.2 Merging Movies with Artists

Merging the movies data with the artists could be done in many different ways, some ideas that were considered were getting the main people that worked on the movie which is provided by title.principals and then setting their average movie rating as a value in the training dataset after splitting the validation data and training data to avoid leakage. This approach was not used since it would be using a non-objective value to predict the result and would go against the experiment’s purpose. Another idea was to train a model to predict the rating only using the movie data and use the actors to predict the deviation from the predicted rating to the real rating, the results here would be much more directed towards the positive or negative influence actors have on the previous prediction model so it was also not taken. Lastly, the approach that was taken which was to assign a column to every artist that has participated in over 10 movies, regardless of role. What made this complicated was that in title.principals, each important artist of each movie is their own row so once each artist has their column, so then the movies must be grouped and taken the max of the values they have within all the artists columns. Once that was done, it was ready to merge to the movies dataset which was also presented with some issues like what if a movie does not have any of the actors, that would mean the movie is not in principals.data and all the artists values would be null. In this situation, the single imputation method was used to fill all the null values with 0 which is logical since if there were no artists with more than 10 movies, then there will be no column that is not 0.

## 3.4 Final Dataset

The final artists\_movies dataset ended up being an immense table with a very wide width, having a final size of (28431, 3283), but upon comparing the results of a simple Random Forest Regression model with and without the artists, there was an improvement. Without having the artists, the mean absolute error was 0.724, and with it was 0.686.

The structure of the final dataset consists of the release year of the movie, followed by the runtime and then 22 different genres, 70 regions and over 3000 artists.

# 4. Methodology and Experimentation

## 4.1 Tools Used

### 4.1.1 Hardware

Device: Gigabyte Aero 15 Laptop

Processor: Intel(R) Core(TM) i7-8750H CPU @ 2.20GHz 2.21 GHz

RAM: 16.0 GB DDR4-2666

GPU: GeForce GTX 1070 Max-Q

### 4.1.2 Software

Operating System: Windows 10 64-bit

IDE: IntelliJ IDEA Community Edition

Programming Language: Python

Libraries: Sklearn, Pandas, TensorFlow, XGBoost, Matplotlib

Version Control: Github

## 4.2 Algorithm Evaluation

In order to compare the algorithms to be implemented there must be a metric to compare them on. Mean absolute error and root-mean squared error are some of the most common for regression algorithms. Mean absolute error gets all the delta between the predicted value and the actual value and adds them up and then divides it by the number of predicted values, it is a very basic and simple to understand. Root-mean squared error works similarly but squares the delta and then adds them up and divides them, it is more practical to differentiate between smaller numbers. In this project since a movie can only go from 1.0 star to 10.0 stars, using the MAE makes more sense to get a grasp of how far the average of predictions are form the real score in average.

For control measures, all algorithms will be given the same training data and tested on the same data. The dataset was also shuffled with a seed in case there was a need to rebuild it so that it keeps the same shuffle. The reason for the data to be shuffled is because the data was in order of how the movies were added to IMDb, which meant that most old movie were early in the database and newer ones later in the database. This was found out to be a problem when the algorithms were tested on cross validation. Cross validation is a form of testing algorithms where the data is split into parts and the data is trained on all but one of the parts and the remaining part is used for validation, using each different part as validation once like so:

A picture containing text, green

Description automatically generated

Figure 3. Cross validation split with 5 folds.

Before shuffling, the results for a simple Random Forest Regression cross validated in 5 folds were:

[0.90974668 0.71514131 0.70825473 0.68138801 0.71806185]

Average: 0.7465185171928843

And after shuffling the MAE scores for the same Random Forest Regression:

[0.71358281 0.71919472 0.72352911 0.69996935 0.69471987]

Average: 0.7101991702097505

As it can be seen, the results before varied a lot more between which validation set was used compared to after shuffling. Therefore, shuffling was done to avoid the need of cross validating every algorithm to get a more accurate test result since cross validating with 5 folds logically meant the training and validating of each model would take 5 times longer.

Algorithms must also be optimized to avoid overfitting and underfitting. Underfitting is the consequence of not training the model enough to be able to predict accurately, overfitting is when the model is overtrained and memorizing the training data, performing poorly when shown new data [21].

Chart, scatter chart

Description automatically generated

Figure 5. Visual example of overfitting and underfitting [21].

## 4.3 Algorithm Optimization

The algorithms presented in Chapter 2.2 were all tested and evaluated. They all were run in their simplest and quickest form at first to make sure they were appropriate for the dataset and computer resources. All of them except Support Vector Regressor were worked on further for optimizations, the reason being is Support Vector Regressor took 20 hours to completely train and validate a simple model with average results.

All algorithms will be run on their default configuration and then the model’s parameters will be tested to find the optimal configuration for lowest MAE. The results will be graphed in a MAE vs the adjusted parameter thus the graph’s lowest point will be the ideal parameter value. Once all parameters have been optimized, a final model will be tested as that algorithm’s final result, solving the first research question: “Which algorithm is most applicable for predicting movie ratings and how do they compare?”

For the second research question: “How accurately can Supervised Machine Learning techniques predict subjective values like a movie’s ratings by using movies’ objective data?” The best two performing algorithms will be adjusted further in a more dedicated manner and tested on the updated IMDB database, therefore being tested on the movie released since the obtaining of the original dataset (March 27th, 2021) and the day of test (May nth, 2021).

## 4.4 Challenges and Limitations

Some variables that could have been very useful for the prediction had many missing values for a lot of movies like language. IMDb also does not provide a movie’s budget in their database which was used on other similar studies.

Support Vector Regressor’s training time was too long for parameter optimizing. Support Vector Regressors unfortunately do not perform well for large datasets, especially if the number of features (columns) is large [22].

All models were trained and validated in a computer; no web services were used. Thus, not having the best processing power and resource availability for the models to be created. This limits how long some models take to train and limiting the optimization adjustments that could be made.

# Results

## 5.1 Testing and Optimizing Models

### 5.1.1 Ridge Regression

Ridge Regression only has one parameter that can be adjusted and that is ‘alpha’. α (alpha) controls the emphasis given to minimizing residual sum of squares vs minimizing sum of square of coefficients [23]. Basically, it is a measurement as to how fit the model should be, ideal for avoiding overfitting and underfitting. With the default value, which is either 0.1, 1.0, or 10.0 depending on performance, the MAE result is 0.66576509. The MAE for those three values of alpha were checked:

MAE RR: 0.6928529563441527 alpha=0.1

MAE RR: 0.682564569220614 alpha=1.0

MAE RR: 0.6643696529307637 alpha=10.0

This means that the value taken was close to 10 but not 10 since alphas=10 performed better than the default. Hence, alphas of values close to 10 were explored.

Chart, line chart

Description automatically generated

Figure 6. MAE vs alphas in Ridge Regression.

Deploying a for loop that increases by 0.1 starting at 9.5 for 12 instances gave the best alphas value to be 10.1 with an MAE of 0.6643688873294995.

### 5.1.2 Lasso Regression

Lasso Regression is similar to Ridge Regression in that it has a parameter alpha to adjust the model’s fitting but has the alphas set to 100 as default. Lasso Regression has another parameter ‘max\_iter’ that can control the maximum number of iterations which by default is 1000. The final parameter that can be changed is normalize which is a simple Boolean that is false by default. Normalizing means the regressors X will be normalized before regression by subtracting the mean and dividing by the l2-norm [25].

The default Lasso model gave a MAE of 0.7307341 and the same model with normalized set to true gave a MAE of 0.6673735059537524. Adjusting any of the other two parameters did not change the MAE consequently no further adjustments were done.

### K-Nearest Neighbor

K-Nearest Neighbor has many parameters. For level of optimizing in this project, only three will be adjusted. All the variables are defined and set by Sklearn [26]

* ‘n\_neighbors’: number of neighbors that will be considered to predict. By default, it is 5.
* ‘weight’: it means how much influence the neighbors have on the value to predict. there are two usual options {‘uniform’, ‘distance’} and a third one that is user-defined, will not be tested. The default is ‘uniform’, and it means that all neighbors carry the same weight. Measuring weight through their distance means that the closer the neighbor is, the larger the influence it has.
* ‘p’: stands for power parameter and it determines the distance equation to use. 1 for Manhattan Distance and 2 for Euclidean Distance. Default is Euclidean.

**Chart, line chart

Description automatically generated**

Figure 7. Comparing weights and distance formulas with a range of neighbors.

With the results listed above, it is clear that Manhattan Distance formula was better than Euclidean as well as measuring the weight is slightly worse if measured uniformly. It must be pointed out that using Manhattan Distance however, took over 5 hours for both weights while Euclidean took 10 minutes. No noticeable time difference was found between weighing the influence of neighbors by distance other than uniformly. The best MAE was 0.7449087913304548 with 27 neighbors, weight: distance and using Manhattan Distance formula to measure distance.

### Decision Tree Regression

Decision Tree Regression can use multiple criteria for when splitting the tree is best, the criteria provided by Sklearn [26] are MAE, MSE (default), Friedman\_MSE and Poisson. Friedman\_MSE is like MSE but implements the Friedman test, it works by first ranking the data and testing if different columns could come from the same “universe” [27]. Finally, Poisson that uses Poisson deviance, which uses the following formula:

Text

Description automatically generated

Where D is the deviance, are the predictions and  is the ground truth [28].

The other very important parameter for a Decision Tree Regression model is the maximum depth of the tree. Without setting a limit to it and using MSE as splitting criteria, the depth is 160 levels and 20824 leaf nodes and an MAE of 0.8801740812379109. The depth of a tree is very important to avoid overfitting, therefore the maximum depth must be adjusted. Testing the ‘max\_depth’ in the MSE model, it was found out that 160 was severely overfitted.

Chart, line chart

Description automatically generated

Figure 8. MAE vs ‘max\_depth’ in a Decision Tree Regression model using MSE for splitting criteria.

The best fit for a DTR model using MSE was having a ‘max\_depth’ of 10, resulting in a prediction MAE of 0.7041127592414346, significantly better than not setting a ‘max\_depth’. The other splitting methods are therefore going to be tested with similar ‘max\_depth’s

Chart, line chart

Description automatically generated

Figure 9. All splitting criteria graphed based on their MAE for predicting movie ratings with different tree depths.

Ironically, the MAE splitting criteria was the best at depth 11 with a prediction MAE of 0.7017232284156849 having 572 node leaves. But just like the K-Nearest Neighbor, the best performing model takes significantly longer to train than other ones for a small improvement. The Poisson splitting criteria behaved very differently from the other 3, having the best results at a depth of 92 with a MAE of 0.8424899291578087 which is still significantly worse than the other splitting methods.

Chart, line chart

Description automatically generated

Figure 10. Poisson splitting criteria on prediction performance for a wide range of depths.

### Random Forest Regression

Random Forest Regression has 3 main parameters to adjust:

* Max depth: Same principle as DTR.
* N-Estimators: Controls the amount of trees in the forest.
* Criterion: Exactly like DTR but only has the option of MSE and MAE. Using MAE however, takes around 4 hours per trained model, making it hard to optimize. For that reason, MAE will not be tested.

A default RFR model would consist of 100 estimators using MSE as splitting criteria and no depth limit with a result of 0.7245645915729089.

Deciding the max depth should be done first since it is dependent on the data while the number of estimators is dependent on the data and depth.

Chart, line chart

Description automatically generated

Figure 11. Random Forest Regression with MSE splitting. Prediction performance with 100 estimators.

Now that the best depth is known to be 27 with an MAE of 0.6745909154741764, the number of estimators can be adjusted.

Chart, line chart

Description automatically generated

Figure 11. RFR model with a max depth of 27 using MSE for splitting.

Finally, RFR reached an MAE of 0.6743714418226444 with 75 estimators and a max tree depth of 27. It must be pointed out the miniscule impact that the number of estimators had, from the worst performing with 25 estimators only being 0.3% better.

### Extreme Gradient Boosting

XGBoost is one of the most powerful algorithms here and that is due to its complexity and efficiency but also the available parameters. One of the most valuable ones is the ‘early\_stopping\_rounds’. What it does is for the model to automatically stop training when the validation score stops improving, finding the ideal number of estimators. It is set as an integer and that integer represents how many iterations of the model not improving the model must go through to stop. This completely removes the need to adjust the estimators and thus avoids overfitting in that regard. Another important parameter is ‘learning\_rate’ which also combats overfitting by multiplying the predictions of each model by the value it is set to be. Consequently, each tree that is added to the ensemble helps less, allowing for a higher number of estimators without overfitting.

A default XGBoost model is already better than most other algorithms with a result of 0.6623188399212084, this is without early stopping rounds or learning rate and faster than any other algorithm. After setting up a model with early stopping rounds of 5 and a changing learning rate, this are the results:

### Artificial Neural Network

## 5.2 The Best Models

# Discussion and Analysis

A picture containing graphical user interface

Description automatically generated

Chart

Description automatically generated

# Conclusion

# Bibliography

1. Jordan M, Mitchell T. Machine learning: Trends, perspectives, and prospects. Science [Internet]. 2015 [cited 27 March 2021];349(6245).
2. Zhang Y. New Advances in Machine Learning. Rijeka, Croatia: Books on Demand; 2010.
3. IMDb data files. 2021. Available at: < https://datasets.imdbws.com/> [Accessed 27 March 2021].
4. Alzubi J, Nayyar A, Kumar A. Machine Learning from Theory to Algorithms: An Overview. [Internet]. Delhi, India: IOP Publishing; 2018 [cited 28 March 2021]. Available from: <https://iopscience.iop.org/article/10.1088/1742-6596/1142/1/012012/pdf>
5. Flach P. Machine Learning: The Art and Science of Algorithms that Make Sense of Data. Edinburgh, UK: Cambridge University Press; 2012.
6. Scheffer, J., 2002. Dealing with Missing Data. [online] Auckland, New Zealand. Available at: <https://mro.massey.ac.nz/bitstream/handle/10179/4355/Dealing\_with\_Missing\_Data.pdf?sequence=1&isAllowed=y> [Accessed 7 April 2021].
7. Hayner, C., 2021. 68 Movies Delayed By COVID-19 (And Every New Release Date). [online] GameSpot. Available at: <https://www.gamespot.com/gallery/68-movies-delayed-by-covid-19-and-every-new-release-date/2900-3542/> [Accessed 8 April 2021].
8. Lee, K., Park, J., Kim, I. and Choi, Y., 2016. Predicting movie success with machine learning techniques: ways to improve accuracy. Information Systems Frontiers, 20(3), pp.577-588.
9. Dorfman, R., Chang, I., Saadat, S. and Roostaeian, J., 2019. Making the Subjective Objective: Machine Learning and Rhinoplasty. Aesthetic Surgery Journal, 40(5).
10. Ning, X., Yac, L., Wang, X., Benatallah, B., Dong, M. and Zhang, S., 2020. Rating prediction via generative convolutional neural networks-based regression. Pattern Recognition Letters, 132, pp.12-20.
11. Zhang, Y., 2018. Predict IMDB score with data mining algorithms. Kaggle, [online] Available at: <https://www.kaggle.com/carolzhangdc/predict-imdb-score-with-data-mining-algorithms> [Accessed 8 April 2021].
12. Rodríguez, P., Bautista, M., Gonzàlez, J. and Escalera, S., 2018. Beyond one-hot encoding: Lower dimensional target embedding. Image and Vision Computing, 75, pp.21-31.
13. Kozma L. k Nearest Neighbors algorithm (kNN). Presentation presented at; 2008; Helsinki University of Technology.
14. Liaw A, Wiener M. Classification and Regression by randomForest. [Internet]. 2002 ;2/3. Available from: <https://www.researchgate.net/profile/Andy-Liaw/publication/228451484_Classification_and_Regression_by_RandomForest/links/53fb24cc0cf20a45497047ab/Classification-and-Regression-by-RandomForest.pdf>
15. Noble, W. S., 2006. What is a support vector machine? Nature Biotechnology, 24(12), 1565–1567.
16. Waseem M. A Quick Guide To Learn Support Vector Machine In Python [Internet]. Python Programming. 2020 [cited 1 May 2021]. Available from: <https://www.edureka.co/blog/support-vector-machine-in-python/>
17. Goodfellow I, Bengio Y, Courville A, Iwasawa y, Suzuki m, Nakayama k et al. Deep Learning. MIT Press; 2016.
18. Ranstam J, Cook C. LASSO Regression. British Journal of Surgery. 2018;Volume 105(Issue 10):Page 1348.
19. Hoerl A, Kennard R. Ridge Regression: Biased Estimation for Nonorthogonal Problems. Technometrics. 1970;12(1):55-67.
20. Bhattacharyya J. Understanding XGBoost Algorithm In Detail. Analystics India. 2020;.
21. 11. Amazon Machine Learning Developer Guide [Internet]. 12th ed. AWS; 2016 [cited 4 May 2021]. Available from: <https://docs.aws.amazon.com/machine-learning/latest/dg/machinelearning-dg.pdf>
22. Raj A. Unlocking the True Power of Support Vector Regression [Internet]. Towards Data Science. 2020 [cited 5 May 2021]. Available from: <https://towardsdatascience.com/unlocking-the-true-power-of-support-vector-regression-847fd123a4a0>
23. Jain A. A Complete Tutorial on Ridge and Lasso Regression in Python [Internet]. Analytics Vidhya. 2016 [cited 5 May 2021]. Available from: <https://www.analyticsvidhya.com/blog/2016/01/ridge-lasso-regression-python-complete-tutorial/>
24. sklearn.linear\_model.LassoCV — scikit-learn 0.24.2 documentation [Internet]. Scikit-learn.org. 2020 [cited 6 May 2021]. Available from: https://scikit-learn.org/stable/modules/generated/sklearn.linear\_model.LassoCV.html
25. 1.6. Nearest Neighbors — scikit-learn 0.24.2 documentation [Internet]. Scikit-learn.org. 2021 [cited 6 May 2021]. Available from: <https://scikit-learn.org/stable/modules/neighbors.html#regression>
26. sklearn.tree.DecisionTreeRegressor — scikit-learn 0.24.2 documentation [Internet]. Scikit-learn.org. 2020 [cited 7 May 2021]. Available from: <https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeRegressor.html>
27. Friedman M. The Use of Ranks to Avoid the Assumption of Normality Implicit in the Analysis of Variance. Journal of the American Statistical Association. 1937;32(200):675-701.
28. Chen P. The Poisson Deviance for Regression [Internet]. 2021 [cited 7 May 2021]. Available from: https://peijin.medium.com/the-poisson-deviance-for-regression-d469b56959ce