A Project Report On

Predicting the Conceptual Appeal of movies

Using Data Analytics

Submitted in partial fulﬁlment of the requirement for 8th semester

**Bachelor of Engineering**

in

Computer Science & Engineering

**VISVESVARAYA TECHNOLOGICAL UNIVERSITY,**

**BELGAUM**



*Submitted by*

**M. VAISHNAVI [1DS16CS052]**

**SOURABH S KULKARNI [1DS16CS110]**

*Under the guidance of*

**Ms Vindhya Ms Kusuma** Professor, CSE, DSCE Assistant professor, CSE, DSCE



**2019-2020**

**Department of Computer Science & Engineering**

**DAYANANDA SAGAR COLLEGE OF ENGINEERING BANGALORE-560078**

**VISVESVARAYA TECHNOLOGICAL**

**UNIVERSITY**

**DAYANANDA SAGAR COLLEGE OF ENGINEERING**

**Shavige Malleshwara Hills, Kumaraswamy Layout, Bangalore - 560078 Department of Computer Science & Engineering**



CERTIFICATE

This is to certify that the project entitled Predicting the Conceptual Quality of Movies Using Data Analytics is a bonaﬁde work carried out by M.Vaishnavi [1DS16CS052] and Sourabh S Kulkarni [1DS16CS110] in partial fulﬁlment of 8th semester, Bachelor of Engineering in Computer Science and Engineering under Visvesvaraya Technological University, Belgaum during the year 2019-20.

**Dr. Vindhya Dr. Ramesh Babu Dr. C P S Prakash**

(Internal Guide) Vice Principal & Head of Principal

Professor, Department Department, CSE, DSCE DSCE

of CSE, DSCE

Signature:....................... Signature:....................... Signature:..................

Name of the Examiners: Signature with date:

1........................................... .....................................

2.......................................... .....................................

**ABSTRACT**

With growing volumes and types of data and piquing interest in using data to produce valuable insights, it has become one of the most important areas of study in today’s era. Huge datasets are available for predictive analysis of several aspects of movies and many domains are available for making predictions. It is beneficial to all varieties of people associated with the art of movie making and watching. Stakeholders like producers can know the risks and advantages of investing in particular movies. Movie watchers can determine if the movie is up to the mark and worth their money. This paper aims to explore the different techniques used for predictive analysis. We also seek to explore what factors are necessary to predict the quality of a movie in terms of its concept and how to establish a relation between different categories.

The objective of this project is to work on the dataset available and identify various factors affecting movie ratings and thereby the quality. These parameters are further used to predict the ratings of the movie before it is released.

We used data analytics to achieve this objective. Data analytics comprises of many different ways to analyse a given dataset. For getting appropriate results from our datasets, we used multiple linear regression to train the model and then predict the results.

ACKNOWLEDGEMENT

We would like to express our gratitude **DR. C P S Prakash**, Principal of DSCE, for permitting us to utilise the resources provided by our college for the project.

We are also very grateful to our respected Vice Principal, HOD of Computer Science and Engineering Department, DSCE, **Dr. Ramesh Babu** for his encouragement and support.

We are immensely thankful to our learned and respected guide, **Ms. Kusuma H**, who constantly guided us throughout all the phases of the project and helped us with the technicalities of the project with her expertise. We deeply express our gratitude for her support.

We would also like to thank our project co-ordinator, **Dr.Vindhya** for providing us the opportunity to work on our projects and giving a wide range of topics to select from. We are very thankful for her constant support.

We would also like to thank other faculty and staff members for their kind co-operation and help.

Lastly, we would like to acknowledge our family members and classmates for providing us with moral support and encouragement.

**List of Figures**

1.1 Multiple Linear Regression . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 3

2.1 Block Diagram of the Proposed Solution . . . . . . . . . . . . . . . . . . . . . . . . . . 4

2.2 Phases in the Proposed Solution . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 5

2.3 Use Case Diagram . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 6

4.1 System Overview . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .9

4.2 System Block Diagram . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 10

4.3 Data Flow Diagram . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 11

4.4 Sequence Diagram . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .12

5.1 Directory structure . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .14

6.1 Training data testing . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 17

6.2 Test data testing . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 19

7.2 Heatmap . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 21

7.3 Failure Scenario . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 22

Contents

1 Introduction 1

1.1 Movie Prediction 1

1.2 Multiple Linear Regression 2

2 Problem Statement and Proposed Solution 4

2.1 Problem Statement 4

2.2 Proposed Solution 4

2.3 System Characteristics 6

3 Literature Survey 7

4 Architecture and Design 9

4.1 System Overview 9

4.2 Software Architecture 10

4.2.1 System Block Diagram 10

4.2.2 Data Flow Diagram 11

4.2.3 Sequence Flow Diagram 12

5 Implementation 13

5.1 Implementation Details 13

5.1.1 Hardware 13

5.1.2 Software 13

5.2 Implementation Details 14

5.2.1 Organisation of Implementation Files 14

5.2.2 Dataset Collection 15

5.2.3 Dataset Preprocessing 15

5.2.4 Weight Extraction 15

5.2.5 Training the Model 15

5.2.6 Testing the Model 16

5.2.7 Running the Flask Application 16

5.2.8 User Interface 16

6 Testing 17

6.1 Testing on Training Dataset 17

6.1.1 Action genre 17

6.1.2 Adventure genre 18

6.2 Testing on Test Dataset 19

6.2.1 Comedy genre 19

6.2.2 Romance genre 20

7 Experiments and Results 21

7.1 Training and Testing Data 21

7.2 Parameters considered 21

7.3 Experiments 22

7.4 Results 23

8 Conclusion 25

**1**

Introduction

* 1. **Movie Prediction**

Making movies involve huge investments and therefore movie prediction has an important role in movie industry. Movies are the most convenient way to entertain people. However, only few movies get high success and are ranked high. Many movies are produced by the movie industry in a year.

Movie revenue depends on various components such as cast acting in a movie, director of the movie, film critics’ review, rating for the movie, genre of the movie, etc. Due to these multiple components there is no formula that helps us to provide analysis for predicting how much revenue a particular movie will generate.

However, by analysing the IMDB score generated by previous movies, a model can be built which can help us predict the expected quality for a particular movie. As we know in today’s world, movies are one of the biggest sources of entertainment and business. To expand this business further, we need the technology through which we can predict the success rate of the movie.

Success rate of models, mechanisms and movies can be employed to predict success for a movie. It will help the viewers whether to watch the movie or not as the quality of the movie will be predicted. Stakeholders such as actors, producers, director etc. can use these predictions to make more informed decisions. They can make the decision before the movie is released.

This proposed work aims to develop a model based upon the data mining techniques that may help in predicting the success of a movie in advance thereby reducing certain level of uncertainty. This is an excellent way to find detailed information about almost every film ever made is through IMDB.

**1.2 Multiple Linear Regression**

Multiple linear regression or MLR, is a technique which statistically uses several explanatory variables for predicting the outcome of a response variable. A multiple linear regression’s goal is to model a linear relationship between the independent or explanatory variables and the dependent or response variable.

Multiple regression is the extension of OLS or ordinary least-squares regression that includes more than one explanatory variable.

The Formula for Multiple Linear Regression is

​*yi* ​= *β*0 + *β*1​*xi*1 + *β*2​*xi*2 ​+ ... + *βp*​*xip* ​+ *ϵ*

where, for *i* = *n* observations:

*yi*​ = dependent variable

*xi*​ = expanatory variables

*β*0​ = y-intercept (constant term)

*βp* = slope coefficients for each explanatory variable

*ϵ* = the  error term of the model (also called residual)​

An SLR or simple linear regression is a function which allows a statistician or an analyst to make predictions for a variable on the basis of that information which is known about another variable. This can only be used in case of two continuous variables—a dependent variable and an independent variable. The independent variable is a parameter which is used to calculate the outcome or dependent variable. An MLR model extends to various explanatory variables.

A multiple regression model is based on the following assumptions:

* There is a linear relationship between the dependent variables and the independent variables.
* The independent variables are not too highly correlated with each other.
* yi observations are selected independently and randomly from the population.
* Residuals should be normally distributed with a mean of 0 and variance σ.

The coefficient of determination or R-squared is a statistical metric which is used to measure the amount of variation in the outcome that can be explained by the independent variable’s. As more predictors are added to the MLR model, the R2 value always increases even though outcome might not be related to the predictors.

On its own, the R2 score cannot be used to identify the predictors that should be included and which ones should be excluded in the model. R2 value ranges between 0 and 1, where 0 implies that the outcome cannot be predicted by any of the independent variables and 1 implies that the outcome can be predicted without any error from the independent variables.

The output of a multiple linear regression can be displayed as an equation horizontally, or in a table form vertically.

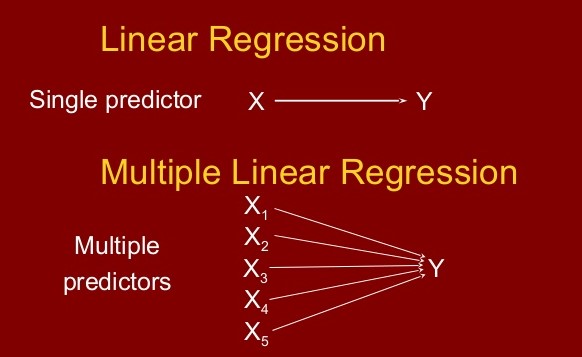


Figure 1.1: Multiple Linear Regression

**2**

**Problem Statement and Proposed Solution**

**2.1. Problem Statement**

To identify the categories that affect the quality of a movie and develop a model that takes the required parameters to predict the quality of an upcoming movie based on IMDB score.

**2.2. Proposed Solution**

The proposed solution for the problem is using data analytics. The model takes in the identified parameters which affect the quality of movies. The multiple linear regression model takes these inputs and predicts the score based on weights assigned by the model during training.

Input the identified parameters

Predict the score of the movie using the weights obtained.

Calculate the weightage for each parameter based on multiple linear regression.

Figure 2.1: Block Diagram of The Proposed Solution

The proposed solution has two phases – Training phase and Testing phase. The processing in each phase is shown as in fig. 2.2.

**Training Phase Testing Phase**

Input Parameters

Input Parameters

Estimate weights

MLR Model

Predict the score

Update the weights to correct the model.

Compare predicted output with observed output.

Figure 2.2: Phases in the Proposed Solution

We predict different measures of quality, based solely on what we know about a movie before its debut. Many attributes reveal themselves after a movie premier, but our input features include only from a dataset that is available. Our measures of movie success are diverse enough to cover a variety of perspectives, from directors’ experience to acting skills to genres.

To determine the quality of the movie, we take a complete dataset of movies with parameters such as movie name, cast, director, genre and rating. We first predict the quality of movie produced by a director in different genres. Here, we generate different values for each genre.

Similarly, we apply this to the actors/actresses present in dataset as well. This process is very important since it is illogical to determine how good a movie is just on the basis of an actor’s success in all movies. He might be talented to act in a particular genre of movie but might be quite the contrary in another genre. Analysing the ratings of different genres for an actor would help in getting a better understanding of how well an actor can act in a movie.

Now, we have predicted for many actors as well as directors based on specific genres. If a new movie is about to be released, the details of the movie are available on the internet or from the trailers. One can easily know who has directed the movie and the cast of the same.

**2.3. System Characteristics**

Figure 2.3: Use Case Diagram

The system should be able to:

1. Accept the input parameters

2. Calculate the weights of the input parameters

3. Predict the IMDB score of a new movie given its input parameters.

4. Display score.

3

**Literature Survey**

**[1] Sentiment Analysis of Movie Review Comments**

**Authors: K. Yessenov and S. Misailovic**

**Description:** This paper presents an empirical study of efﬁcacy of machine learning techniques in classifying text messages by semantic meaning. They use movie review comments from popular social network Digg as our data set and classify text by subjectivity/objectivity and negative/positive attitude. They propose different approaches in extracting text features such as bag-of-words model, using large movie reviews corpus, restricting to adjectives and adverbs, handling negations, bounding word frequencies by a threshold, and using WordNet synonyms knowledge. They evaluate their effect on accuracy of four machine learning methods-Naïve Bayes, Decision Trees, Maximum-Entropy, and K-Means clustering. They conclude our study with explanation of observed trends in accuracy rates and providing directions for future work.

**[2] Deep Learning for Sentiment Analysis of Movie Reviews**

**Authors: H. Pouransari, & S. Ghili**

**Description:** In this study, they explore various natural language processing (NLP) methods to perform sentiment analysis. We look at two different datasets, one with binary labels, and one with multi-class labels. For the binary classiﬁcation, they applied the bag of words, and skip-gram word2vec models followed by various classiﬁers, including random forest, SVM, and logistic regression. For the multi-class case, they implemented the recursive neural tensor networks (RNTN). To overcome the high computational cost of training the standard RNTN they introduce the low-rank RNTN, in which the matrices involved in the quadratic term of RNTN are substituted by symmetric low-rank matrices. They show that the low-rank RNTN leads to signiﬁcant saving in computational cost, while having a similar accuracy as that of RNTN.

**[3] Rating based Mechanism to Contrast Abnormal Posts on Movies Reviews using MapReduce Paradigm**

**Author: Piyush Gupta, Atul Sharma, Jitender Grover**

**Description:** BigData contains large amount of unstructured data in the form of movie data, facebook data, and industry data and so on. There are number of posts are posted on twitter about movies by different users. Out of these posts some of posts may be inappropriate. These posts contain negative comments as well as positive comments about movies. It is difficult to distinguish large number of positive and negative posts. To overcome this kind of problem we proposed a rating based mechanism that distinguishes abnormal posts with the help of users rating. If rating is positive then post is normal otherwise it is abnormal. To implement proposed mechanism we used hadoop platform and MapReduce paradigm.

**[4] Movie Success Prediction using Machine Learning Algorithms and their Comparison**

**Authors: Rijul Dhir, Anand Raj**

**Description:** The number of movies produced in the world is growing at an exponential rate and success rate of movie is of utmost importance since billions of dollars are invested in the making of each of these movies. In such a scenario, prior knowledge about the success or failure of a particular movie and what factor affect the movie success will benefit the production houses since these predictions will give them a fair idea of how to go about with the advertising and campaigning, which itself is an expensive affair altogether. So, the prediction of the success of a movie is very essential to the film industry. In this proposed research, we give our detailed analysis of the Internet Movie Database (IMDb) and predict the IMDb score. This database contains categorical and numerical information such as IMDb score, director, gross, budget and so on and so forth. This research proposes a way to predict how successful a movie will be prior to its arrival at the box office instead of listening to critics and others on whether a movie will be successful or not. The proposed research provides a quite efﬁcient approach to predict IMDb score on IMDb Movie Dataset. We will try to unveil the important factors inﬂuencing the score of IMDb Movie Data. We have used different algorithms in the research work for analysis but among all Random forest gave the best prediction accuracy, which is better in comparison to the previous studies.

4

Architecture and Design

4.1 System Overview

model.py

app.py

request.py

Datasets

UI

Input parameters like genre, director name, actor’s name

Predicted IMDB score

model.pkl

Train

Figure 4.1:System Overview

The overview of the system is represented in Fig.4.1. It shows the modules involved in building the system i.e,

* User interface
* Request.py
* App.py
* Model.py
* Training modules and datasets

4.2 Software Architecture

**4.2.1** **System Block Diagram**

UI

Input details Output IMDB score

to the system from the system

Linear Regression

Multiple Linear

Regression

Input Processing

Figure 4.2: System Block Diagram

The overall block diagram of the proposed system is shown in the Fig.4.2.

-User inputs the details to the system through UI.

-The overall system is represented as a Linear Regression. It consists of: Input processing and Multiple Linear Regression.

-Input details are processed and then a value is predicted from the model that is built from multiple linear regression.

-Output IMDB score will be displayed in the UI.

**4.2.2 Data Flow Diagram**

User

Figure 4.3: Data Flow Diagram

A data ﬂow diagram (DFD) is a graphical representation of the ﬂow of data through an information system. A DFD gives the preliminary overview of the system without going into great detail. Fig.4.3 represents the DFD of our proposed system. The ﬂow of the system is as follows:

1) User chooses the genre and inputs director and actor’s names.

2) The inputs will be sent to app.py.

3) Processing of the inputs and predicting output from the model which has been obtained after multiple linear regression takes place.

4) The predicted IMDB score will be displayed to the user.

**4.2.3 Sequence Diagram**

User UI Server Trained model

Choose genre(index.html) genre\_name

genre\_name.html genre\_name.html

input director and actor’s

name(genre\_name.html) button= “predict” input the details

predicted IMDB score Predicted score along Result and

(genre\_name.html) with genre\_name.html output image

Figure 4.4: Sequence Diagram

A sequence diagram is an interaction diagram that shows how objects operate with one another and in what order. Fig.4.4 represents the sequence diagram that shows the interaction between the User, interface, server and the trained model. It gives the sequence of actions from the moment of user entering the image till the resulting image displayed to the user.

5

**Implementation**

**5.1. Implementation Details**

**5.1.1 Hardware**

* **Processor**: Intel Core i7
* **Ram**: 8GB
* **GPU**: NVIDIA

**5.1.2 Software**

* **Operating System**: Windows 10 (64bit)
* **Programming Languages**: Python, HTML, CSS
* **Data Analysis Framework**: Spyder, Flask

**5.2 Implementation Details**

**5.2.1 Organization of implementation files**

Movie-master

templates

movie\_metadata.xlsx

request.py

model.py

app.py

Deployment-flask-master

action.html

adventure.html

comedy.html

index.html

romantic.html

static

style.css

css

model.pkl

Figure 5.1: Directory structure

The above diagram shows the organization of the implementation ﬁles. The detailed explanation of the ﬁgure is given in further sections.

**5.2.2 Dataset Collection**

The training dataset comprises of list of around 5000 movies with a number of attributes like director name, actor names, duration of the movie, year of release, genre, revenue collected, budget etc, along with the IMDB scores of the movies. The dataset consists of many genres and many movies which enables our learning algorithm to train with more amount of information.

The movie dataset is stored in the movie\_metadata.xlsx in the form of Excel file.

The dataset is divided in the ratio 80:20 for training and testing respectively.

The app.py, model.py and request.py contain the files for our flask application.

The template ﬁle contains all the webpages in our website.

**5.2.3 Dataset Preprocessing**

The first step in doing a data analysis project is to pre-process the data. The model.py file consists for the code to pre-process our data. The dataset consists of lots of entries of movies with a number of attributes like color, duration, year of release, director name, number of facebook likes, genre, IMDB scores, etc.

1. **Removal of unwanted attributes**- Now many of these attributes are not required while working the dataset. Hence, we remove these columns directly on the basis of initial screening. This is done using the **drop** command in Spyder.
2. **Removal of null values-** In any dataset, there will be a number of missing or null values that have to be removed to prevent any value from affecting the results produces after prediction. This is done using **dropna** command and including all the columns for which we need to check null values. For example, on giving director name in the arguments, a movie having null value for director name will be dropped from the dataset.
3. **Considering language-** We preferred to work only on movies having language as ‘English’ and hence we reduced the dataset to the movies consisting of only ‘English’ language movies.
4. **Removing outliers-** Outliers are those values which digress far from the dataset and can pose problems while calculating and predicting the IMDB score. For our dataset, we considered num\_voted\_users as a parameter for checking outlier conditions. Since IMDB scores depend greatly on the number of people who have voted for the movie, if the number of people who have voted for a movie is just a handful, then the scores wouldn’t be reliable. Hence, we find a threshold values for the num\_voted\_users below which, the corresponding movie tuple would be removed. For this we use **quantile** and give a value of 0.1 as argument. This value is stored in a variable and it is the threshold. It consists of the value which is 10% of the maximum num\_voted\_users. Then we check the dataset and remove all unwanted tuples.
5. **Creating dummies-** The attributes like director\_name, actor\_name and genres consist of names and they are categorical values. For working on a prediction model, we need numerical values. Hence, firstly we need to create dummies of the genres and then combine the results into a single column. For example, if a movie has action genre, then the dummy value of action will be 1 for that movie. If not, then a value of 0 will be assigned. Similarly for all genres, dummies will be created and we get an idea of all genres that comprise that particular movie.
6. **Reducing the dataset-** We then prefer to work on individual dataset based on particular genres. So initially we take the dataset consisting of only action movies.
7. **Finding average IMDB scores-** It’s highly possible that a director in a genre would have worked in many movies. So, we need to combine the scores for the director and produce an average score. Similarly, we do this for the actors as well.

**5.2.4 Weight Extraction**

In a linear model, the different predictors in the equation are assigned a co-efficient value which indicates the weightage of that predictor on the response variable or output. The purpose of using a multiple linear regression model for our dataset is to obtain the weightage of different parameters like directors and actors to determine which features to consider for the model.

On training the model, the weights are self-adjusted to give the best and approximately correct output value based on our training dataset. These weights are then used as the base weights to further train and reﬁne the multiple linear regression models.

**5.2.5 Training the model**

For training the model, the dataset is split into 80:20 ratios for training and testing respectively. It is done using-

**from sklearn.model\_selection import train\_test\_split  
xTrain, xTest, yTrain, yTest = train\_test\_split(x, y, test\_size = 0.2, random\_state = 100)**

A multiple linear regression model is used which is present in the in-built sklearn library of spyder. The parameters obtained from the pre-processed data are fed as input to this model along with output which is the IMDB score.

The linear regression model can be obtained by imported from the sklearn library. As we need a multiple linear regression model, we give the argument of fit\_intercept as true. It is demonstrated in the following code-

**from sklearn.linear\_model import LinearRegression**

**mlr\_model= LinearRegression(fit\_intercept=true)**

**mlr\_model.fit(x\_train,y\_train)**

The training values are fitted into the model where x\_train represents the input training data and y\_train represents the training output values.

model.py file consists of this code for model. On typing the command given below in Anaconda prompt, the training of the model takes place-

**python model.py**

**5.2.6 Testing the model**

The model can be tested using the test data available. It validates the correctness of our model by comparing observed values against actual values. Testing data consists of that part of the dataset which is unknown to the model. Our model has trained on the basis of training data and that cannot be used to validate or test the model, since it is already known to the dataset.

The following code allows us to test the model on x\_test values which we had originally obtained after splitting. The predictions are stored in a variable called predictions.

**predictions=mlr\_model.predict(x\_test)**

**5.2.7 Running the flask application**

We have made a flask application for front end part of our predictive model. The webpages for our website are stored in templates file. The first page is index.html and it consists of a dropdown for selecting the genre.

To display the user interface, we need a local host which can be started by using the following command-

**python app.py**

It provides a URL for the local host which can be opened using a web browser.

**5.2.8 User Interface**

The user interface consists of a flask application which is a website to provide user friendly interface. The source code for the web pages are present in the templates folder, where index.html is the home page.

The app.py folder consists of methods to render appropriate result and web page by clicking on the predict button provided in all the pages.

6

**Testing**

Testing is carried out with 20% of the dataset. The accuracy of the IMDB score in the results and the prediction percentage depends on the parameters considered. The following cases describe the results on testing data for different genres obtained from different parameters and iterations.

**6.1 Testing on Training Dataset**

Initially we consider those entries which were part of the training set of the data. This consists of the movies which the model has already worked upon.

**6.1.1 Action genre**

Firstly, action genre was selected from the dropdown menu. Then entries are input by the user. The information is as follows-

Director name - Adam McKay

Actor 1 name - Dwayne Johnson

Actor 2 name - Will Ferrel

The target IMDB score is for the set of details that the user has entered is 6.7 which can be seen on the left side of the image, the first entry.

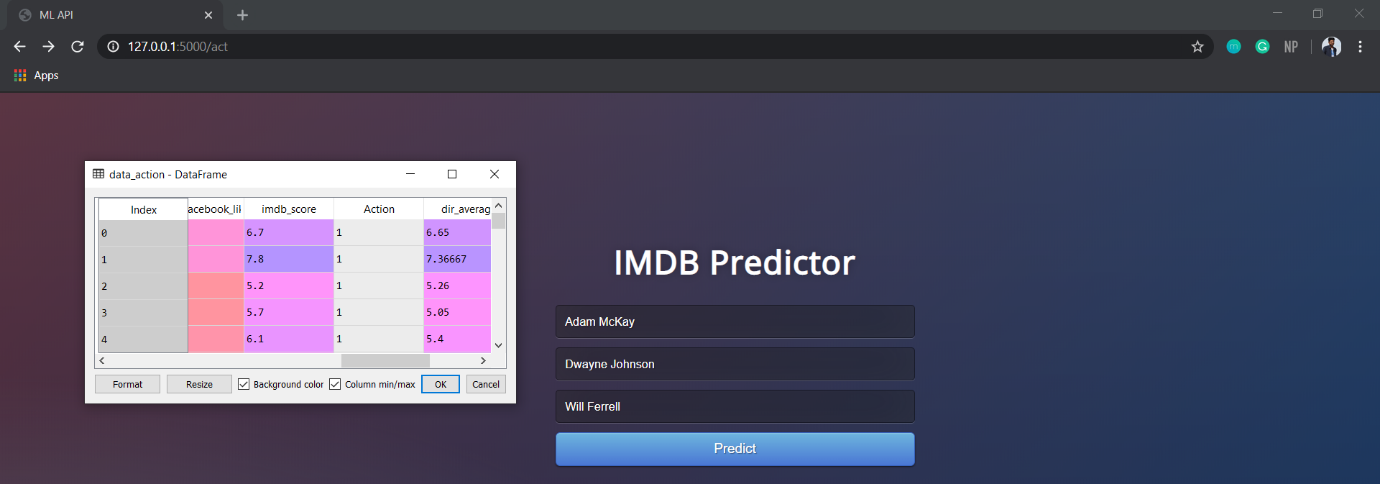


Figure 6.1: Training data

On clicking the predict button, the calculation is done and the predicted output seen is 6.98. The difference between the 2 scores is roughly 0.2 and hence acceptable.

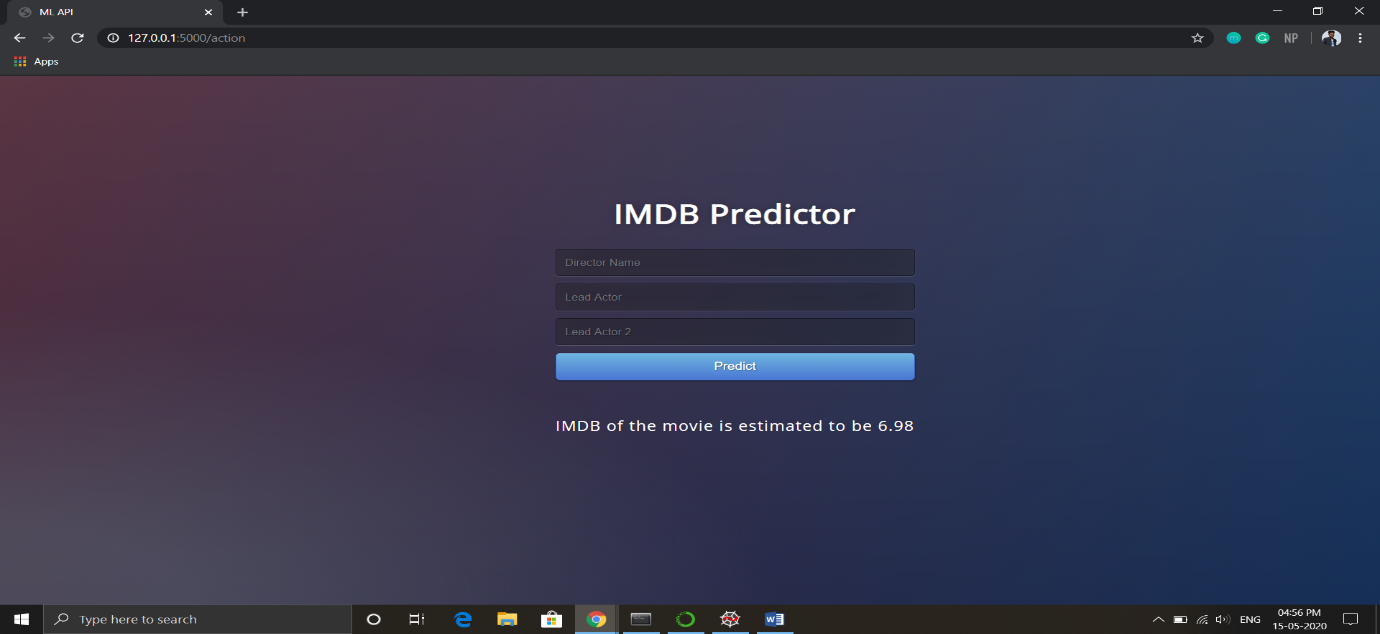


Figure 6.2: Predicted output

**6.1.2 Adventure genre**

Next, we selected adventure genre from the dropdown menu. Then entries are input by the user. The information is as follows-

Director name - Adam Shankman

Actor 1 name - Taylor Lautner

Actor 2 name - Tom Welling

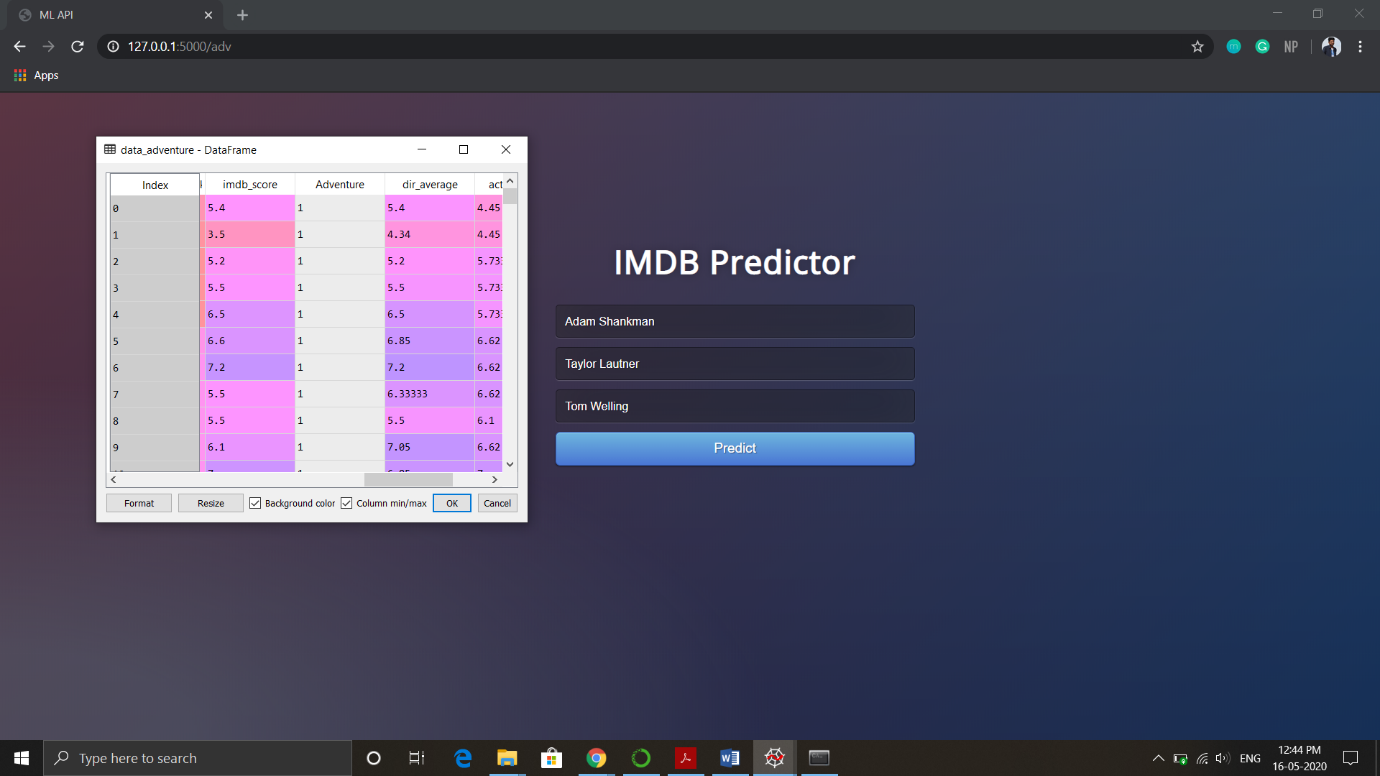


Figure 6.3: Training data

The target IMDB score is for the set of details that we have entered is 5.4. This can be seen in the dataset details attached on the left side of the image.

On clicking the predict button, the predicted output seen is 5.13. The difference between the 2 scores is roughly 0.3 and hence acceptable.

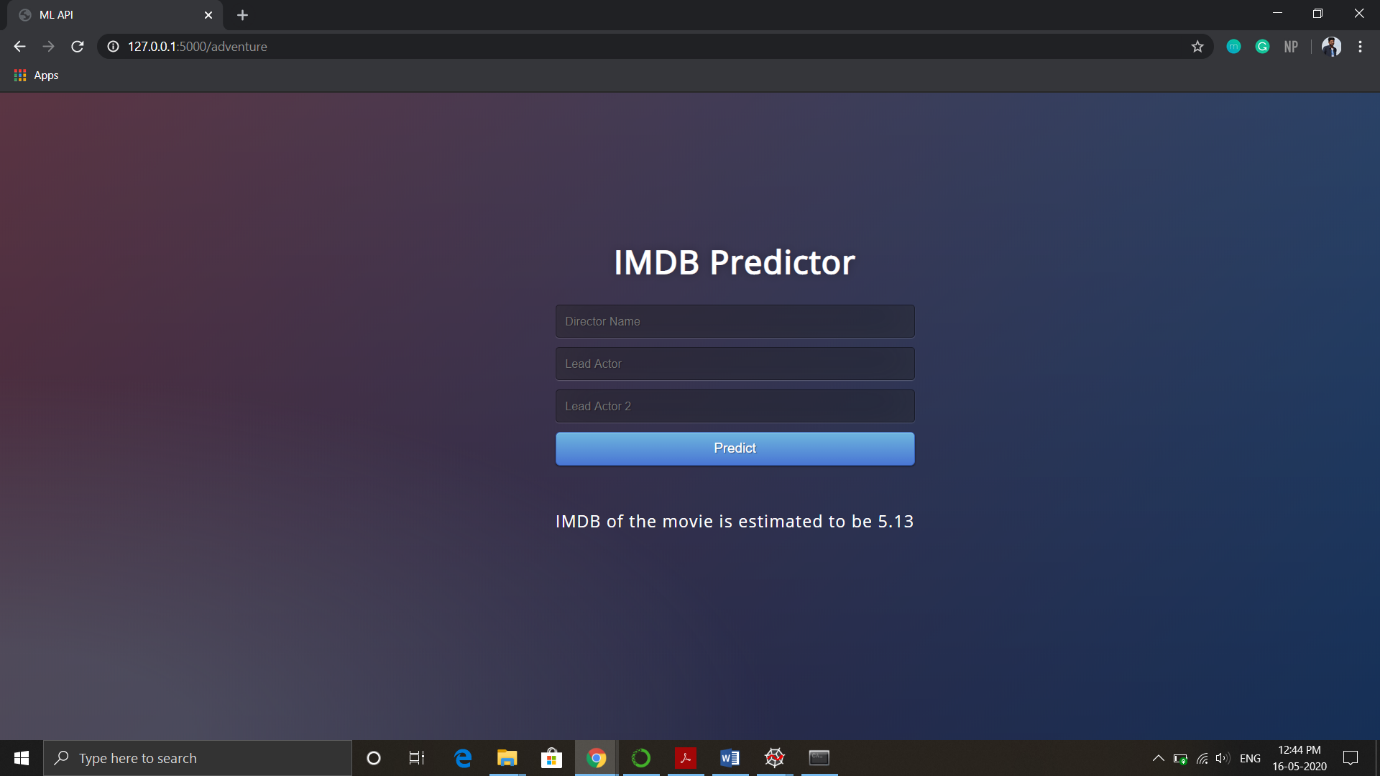


Figure 6.4: Predicted output

**6.2 Testing on Test Dataset**

We now check the results for the test dataset. It includes the part which was not considered for training.

**6.2.1 Comedy genre**

To begin, we selected the comedy genre from the dropdown menu. Then entries are input by the user. The information is as follows-

Director name - Adam Shankman

Actor 1 name - James Martin Kelly

Actor 2 name - Shane Hartline

The target IMDB score is for the set of details that we have entered is 5.9. This can be seen in the dataset details attached on the left side of the image.

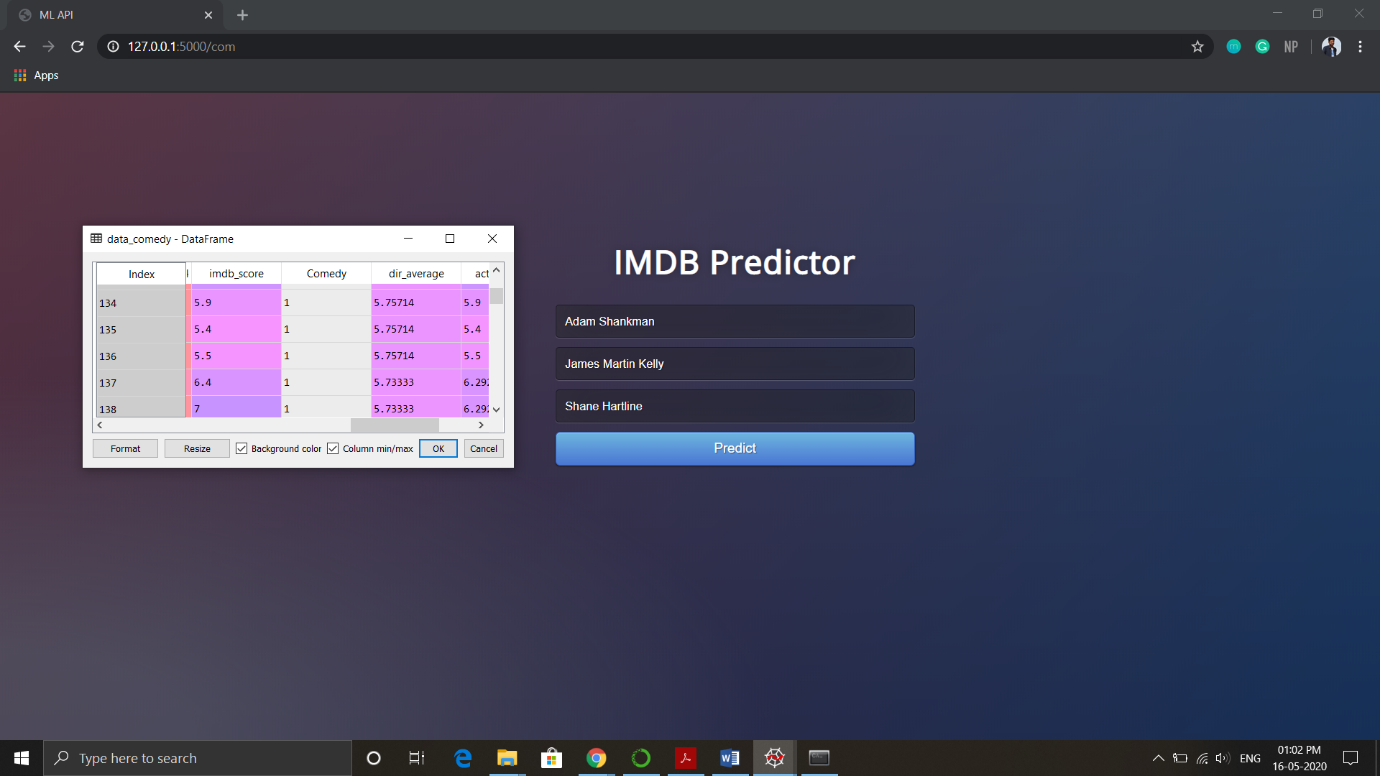


Figure 6.5: Test data

On clicking the predict button, the predicted output seen is 5.76. The difference between the 2 scores is roughly 0.2 and hence acceptable.

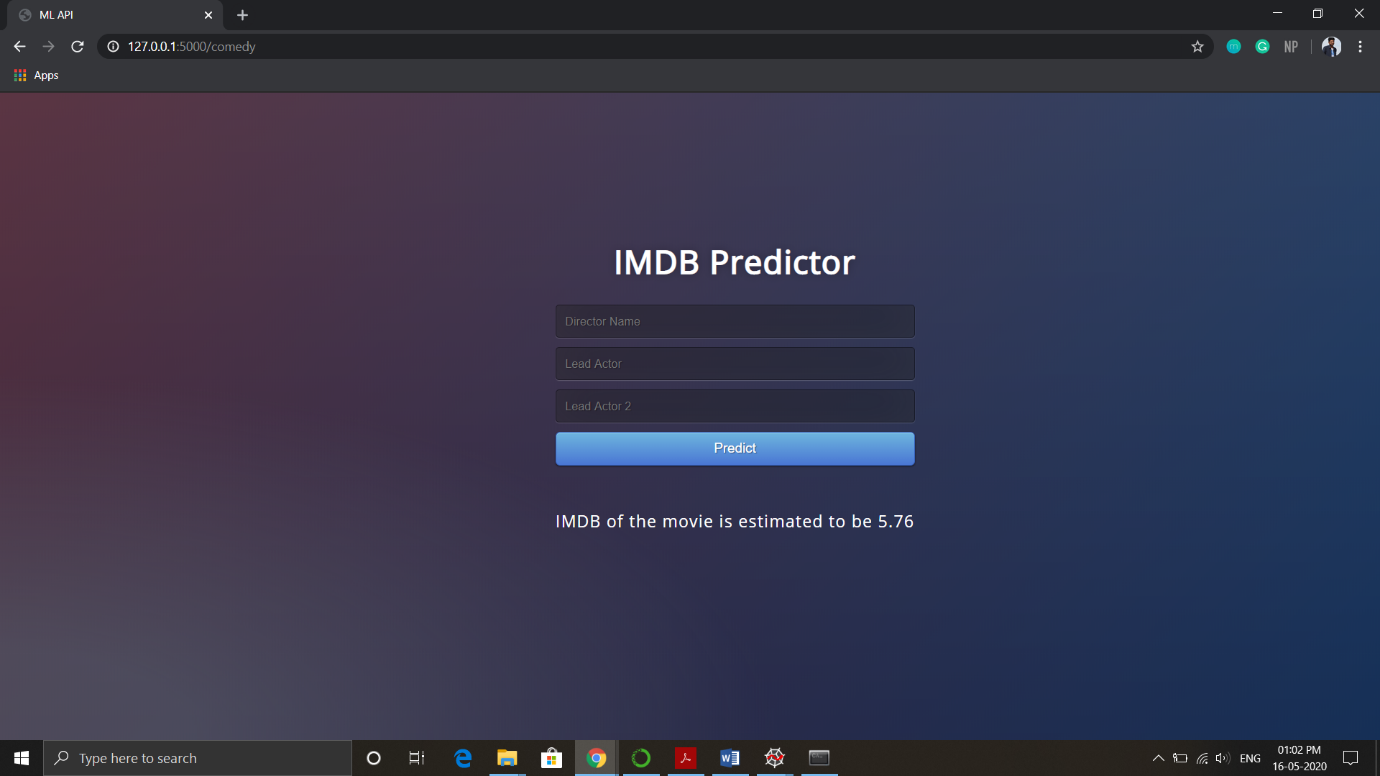


Figure 6.6: Predicted output

**6.2.2 Romance genre**

Then to check for one more entry we selected the romantic genre from the dropdown menu. Then entries are input by the user. The information is as follows-

Director name – Peter Howell

Actor 1 name - Frances Fisher

Actor 2 name - Mike Doyle

The target IMDB score is for the set of details that we have entered is 5.9. This can be seen in the dataset details attached on the left side of the image.

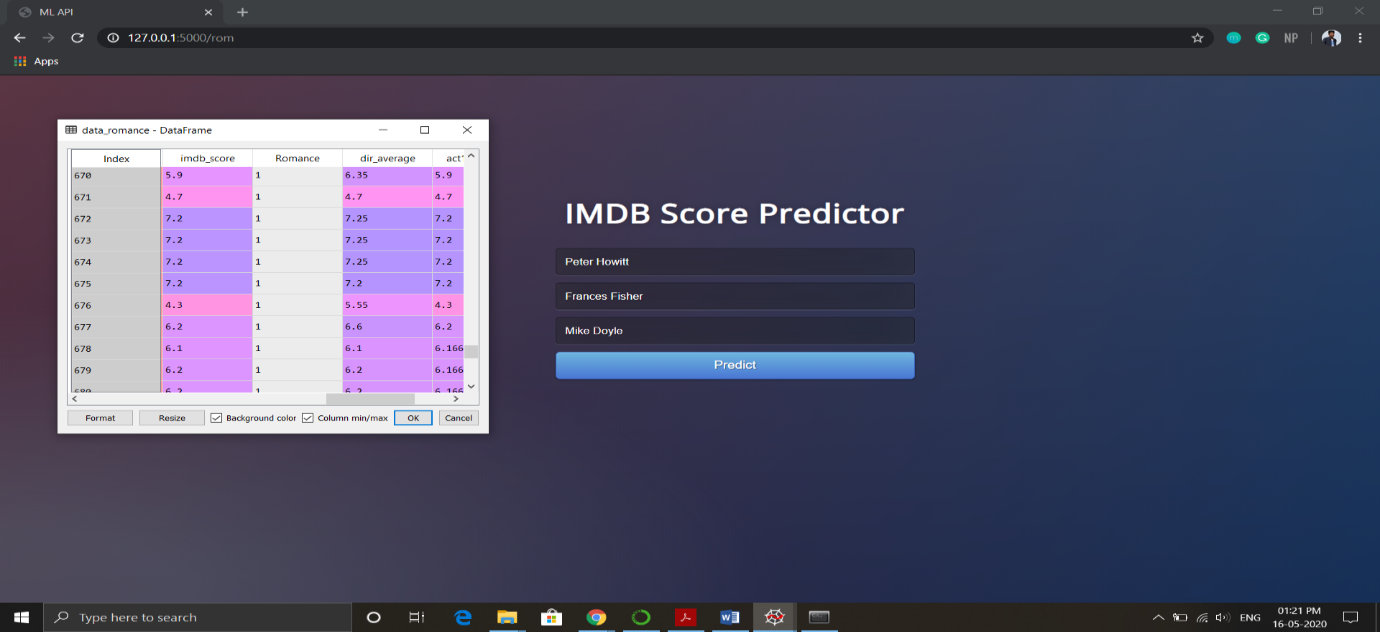


Figure 6.7: Test data

On clicking the predict button, the predicted output seen is 6.02. The difference between the 2 scores is roughly 0.2 and hence acceptable.

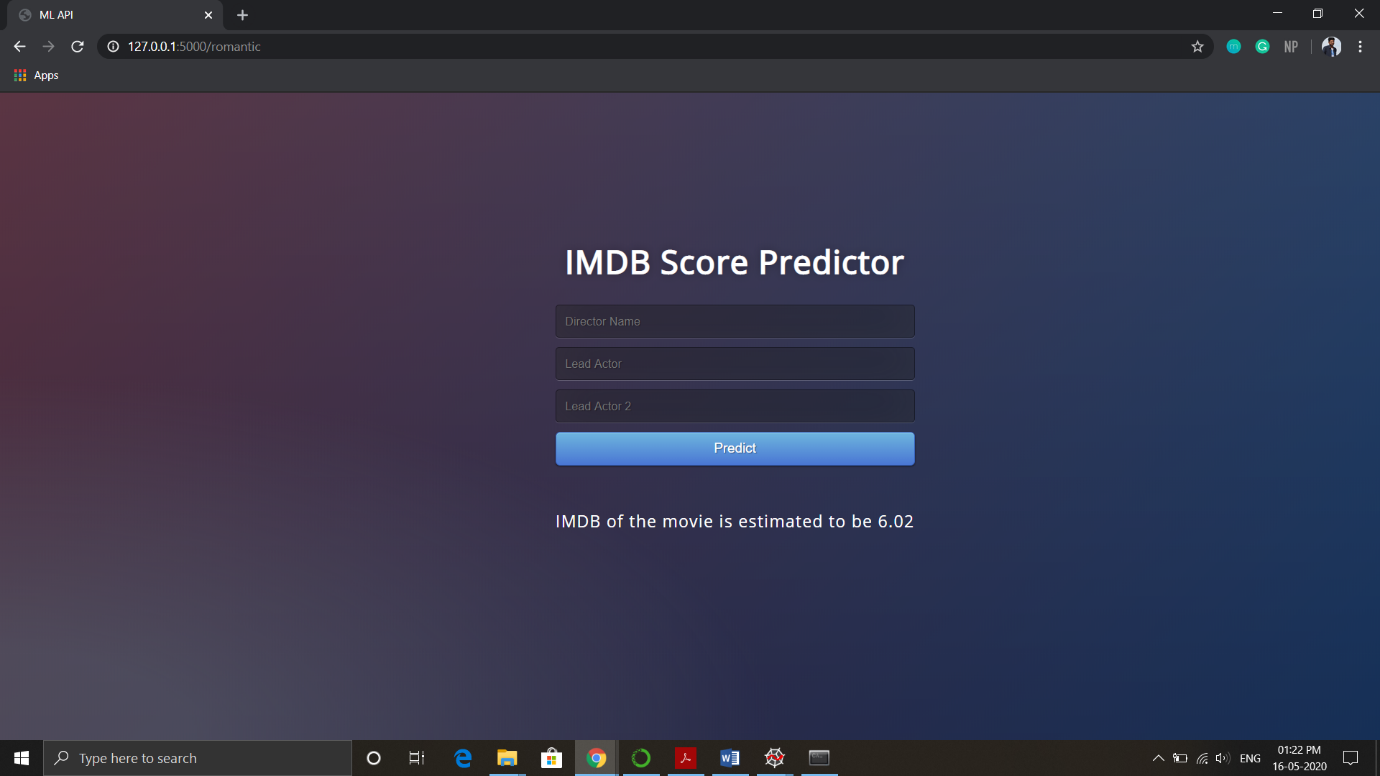


Figure 6.8: Predicted output

**7**

**Experiments and Results**

The prediction percentage and the accuracy of the bounding boxes in the results depends on the

1) Batch size i.e. the size of the training and testing data

2) Parameters considered

**7.1 Training and Testing Data**

The dataset is split into two parts using the train\_test\_split imported from scipy library. Following code is used for this purpose-

**from sklearn.model\_selection import train\_test\_split  
xTrain, xTest, yTrain, yTest = train\_test\_split(x, y, test\_size = 0.2, random\_state = 100)**

The parameters considered are-

* **test\_size** — This parameter decides the size of the data that has to be split as the test dataset. This is given as a fraction. For example, if we pass 0.5 as the value, the dataset will be split 50% as the test dataset.
* **train\_size** — We have to specify this parameter only if we’re not specifying the test\_size. This is the same as test\_size, but instead we tell the class what percent of the dataset we want to split as the training set.
* **random\_state** — Here we pass an integer, which will act as the seed for the random number generator during the split. We can also pass an instance of the RandomState class, which will become the number generator.

It splits the dataset into training data and testing data. According to our algorithm, we split it as 80:20 that is 80% training data and 20% testing data.

**7.2 Parameters considered**

The parameters that we consider in the model play an important role in how accurate the model will perform. The relationship between the input variables and the output variables can be seen using a heat map which shows the correlation between these variables.

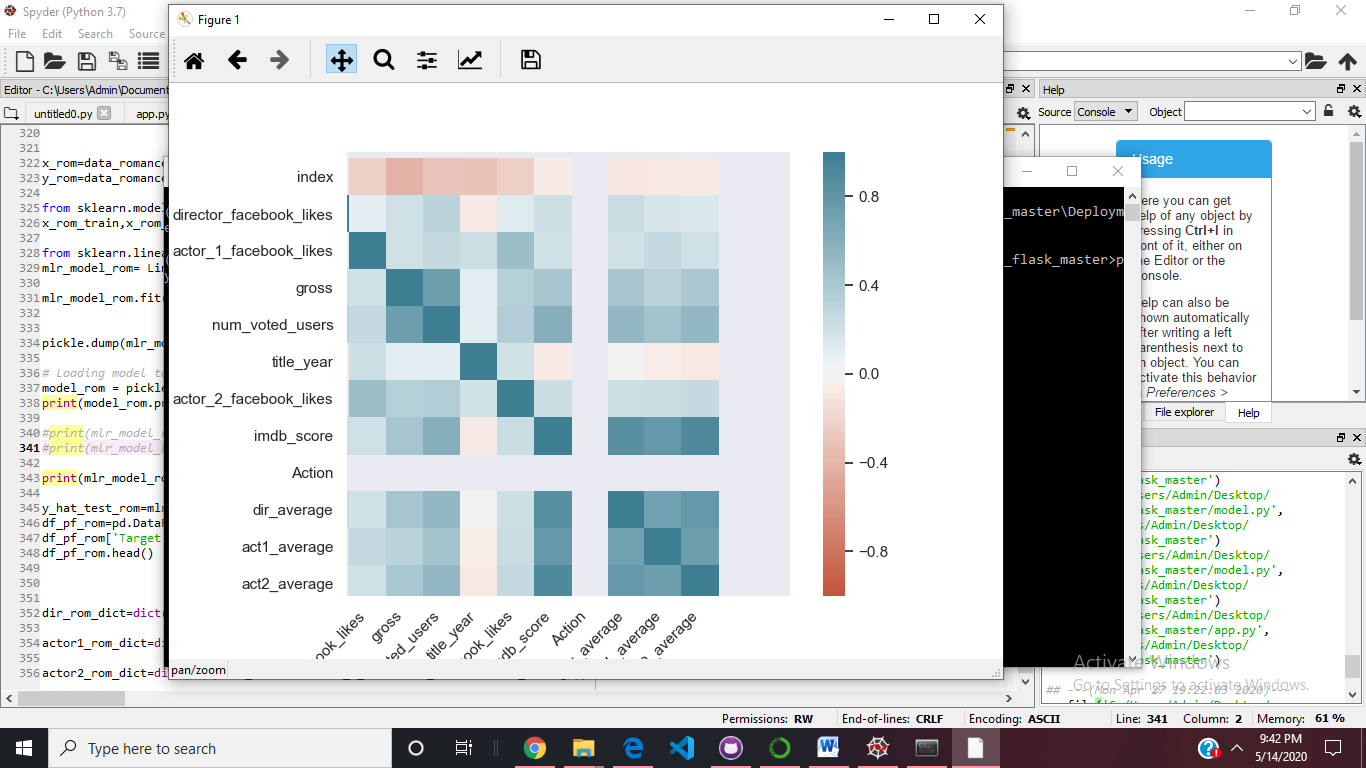
Correlation is actually a numerical value between 0 and 1 which shows the extent of relationship between two variables.

Fig 7.2 Heatmap showing the correlation between input and output variables

A dark blue gradient shows that there is a deep relationship between input and output variables and lighter shades show less dependency. Towards red color side, it shows a negative dependency.

The parameters can also be considered based on the R2 score. R2 score determines the accuracy of a model. The value lies between 0 and 1. A score between 0 to 0.5 shows that the model is not good, a score between 0.5 to 0.8 shows that it is an average model and between 0.8 to 1 shows that it is a good model.

On considering the director and actor 1 and 2 average scores as input parameters, we found that for different genres, the R2 score was always between 0.8 to 1 indicating that these were the parameters actually affecting the output.

**7.3 Experiment**

We tested the failure case scenarios and worked on displaying the error messages to the user. Since our entries are present in the form of dictionaries, they are arranged in the form of key value pairs. The keys are the names that the users enter and values are the corresponding averages.

If any name for the director, actor1 or actor2 is not present in the dataset, it won’t be a key in the dictionary. A key value error is thrown by the application. To correctly handle the error and notify the user about the error, a message is printed.

1. Typing the wrong Lead Actor 1 name

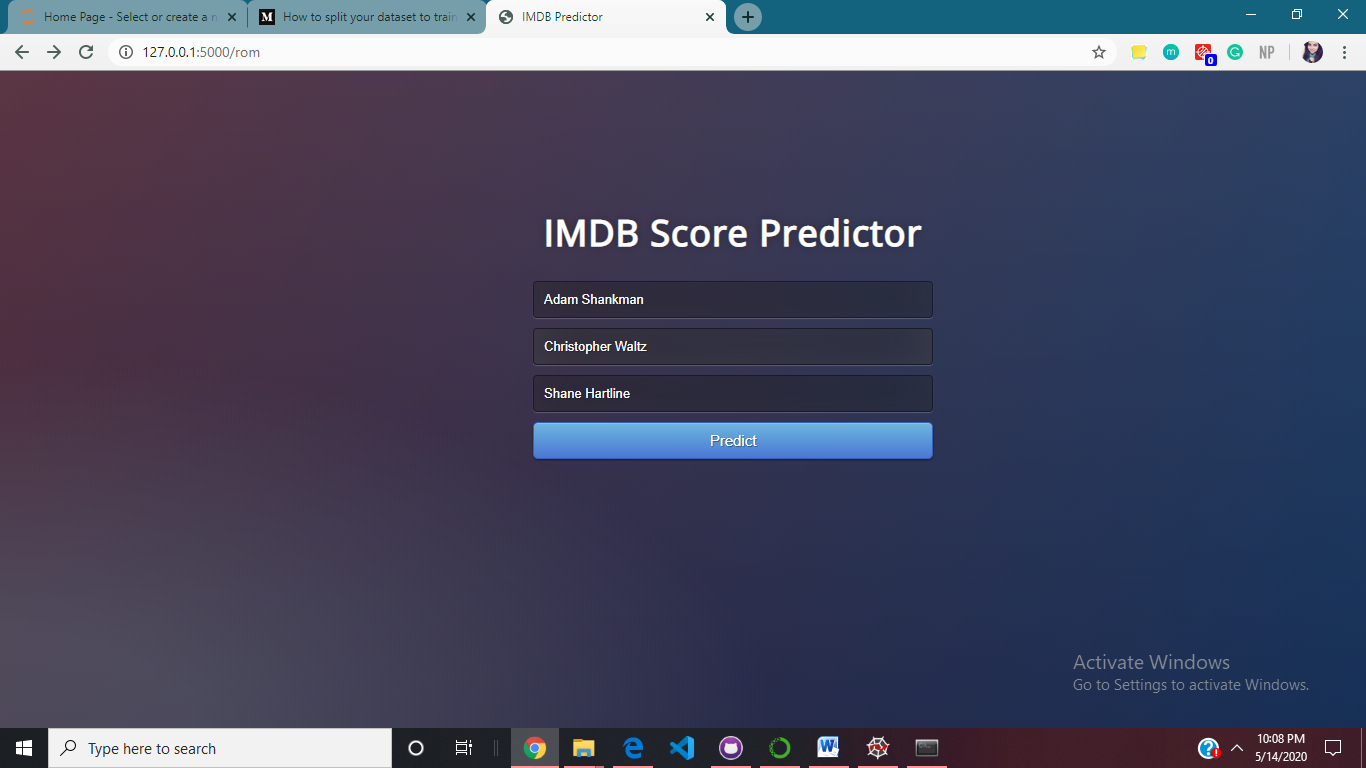


Fig 7.3.1 UI showing user inputs with wrong entry for lead actor 1

The user has entered the wrong entry for Lead Actor 1 as there is no key present in the dictionary as ‘Christopher Waltz’. Hence after clicking on the predict button, an error message is printed which indicates to the user that the lead actor 1 name is invalid.

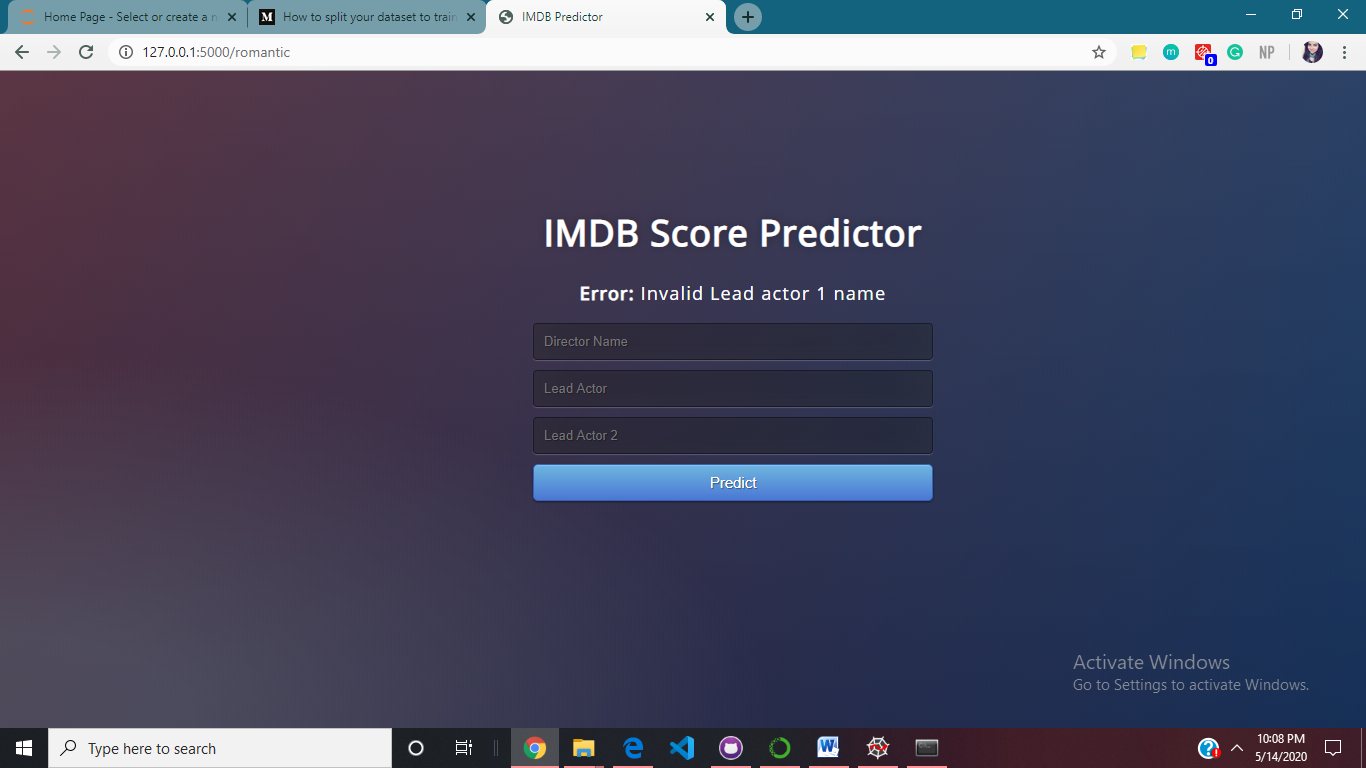


Fig 7.3.2 UI showing the appropriate error message

**7.4 Results**

* The test size of 80-20 was considered as the optimal split for training and testing data.
* Parameters that gave the best R2 score were a combination of three variables- director average score, actor 1 average score, actor2 average score. Hence these three were considered for our multiple linear regression model.
* Failure case scenarios showed error messages when user input any name that was not present in the dataset.

**8**

**Conclusion**

The proposed system is based on data analytics, to identify the parameters affecting the conceptual quality of movies and predicting the IMDB score. The objective of the proposed model is achieved using the following modules.

1) Finding and modifying the dataset for the movie score predictor consisting of around 5000 movies. The Spyder software and python libraries like pandas and seaborn are used to pre-process the data.

2) The multiple linear regression model is designed using the python library called scikit. It consists of methods to train the dataset using the MLR model. The model.py consists of the code to train the model.

3) The model was trained on the dataset by splitting it in the ratios of 80:20 as training and testing dataset respectively. The weights are assigned to the parameters according to the training and then the results are validated according to the testing dataset.

5) The results showed the prediction of IMDB score for a movie when the input parameters were input. The model was able to predict on new data as well.