# PROFESSIONAL TRAINING REPORT

**at**

**Sathyabama Institute of Science and Technology**

**(Deemed to be University)**

Submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering Degree in Computer Science and Engineering

By

## PATIBANDLA VENKATA LOHITH KUMAR

**REG. NO. 39111135**



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**SCHOOL OF COMPUTING**

**SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY**

**JEPPIAAR NAGAR, RAJIV GANDHI SALAI,**

**CHENNAI – 600119, TAMILNADU**

**April 2022**

|  |  |  |
| --- | --- | --- |
|  | **SATHYABAMA**  **INSTITUTE OF SCIENCE AND TECHNOLOGY** (DEEMED TO BE UNIVERSITY) **Accredited with Grade “A” by NAAC**  (Established under Section 3 of UGC Act, 1956)  JEPPIAAR NAGAR, RAJIV GANDHI SALAI  CHENNAI– 600119  [**www.sathyabama.ac.in**](http://www.sathyabama.ac.in/) |  |

# DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

**BONAFIDE CERTIFICATE**

This is to certify that this Project Report is the bonafide work of **PATIBANDLA VENKATA LOHITH KUMAR (Reg. No: 39111135)** carried out the project entitled “**MOVIE BOX OFFICE GROSS PREDICTION**” under my supervision from March 2022 to April 2022.

## Internal Guide

## Mrs.Vanathi .M

**Head of the Department**

Dr. L. Lakshmanan, M.E., Ph. D

## Submitted for Viva-voce Examination held on

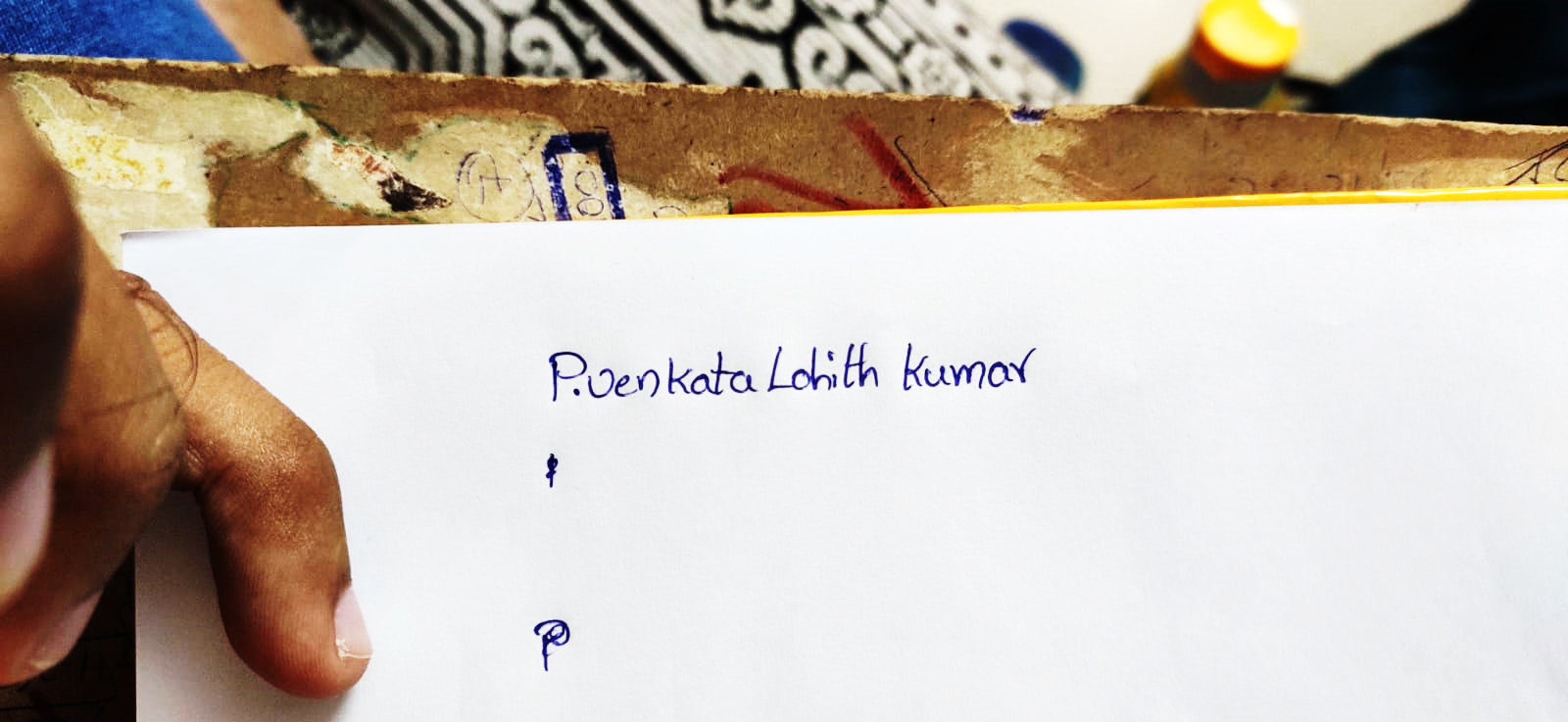
**Internal Examiner External Examiner**

# DECLARATION

I, **PATIBANDLA VENKATA LOHITH KUMAR** hereby declare that the project report entitled “**MOVIE BOX OFFICE GROSS PREDICTION”** wasdone by me under the guidance of **Mrs.Vanathi.M.** is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering Degree in Computer Science and Engineering.

## DATE:

**PLACE:**

****

**SIGNATURE OF THE CANDIDATE**

# ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to the **Board of Management of SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

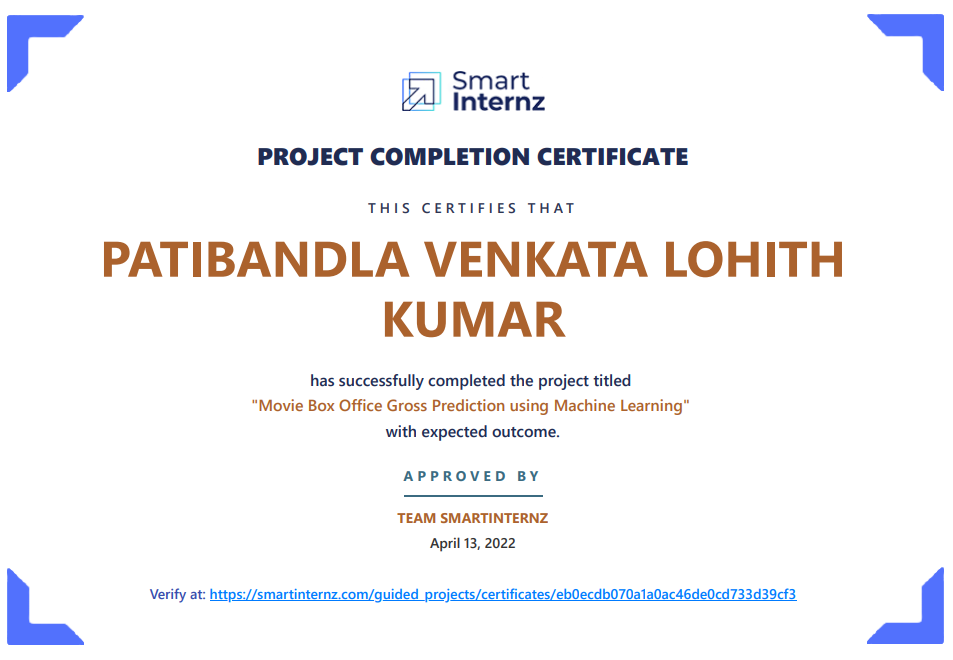
I convey my thanks to **Dr. T. Sasikala M.E., Ph. D**, Dean, School of Computing,

**Dr. S. Vigneshwari, M.E., Ph.D.** and **Dr. L. Lakshmanan, M.E., Ph.D.,** Heads of the Department of Computer Science and Engineering for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide, **Mrs.Vanathi .M.** for his valuable guidance, suggestions, and constant encouragement that paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Computer Science and Engineering** who were helpful in many ways for the completion of the project.

# TRAINING CERTIFICATE

****

# ABSTRACT

Predicting society's reaction to a new product in the sense of popularity and adoption rate has become an emerging field of data analysis, and such kind of analysis can help the movie industry to take appropriate decisions. Can film studios and its related stakeholders use a forecasting method for the prediction of revenue that a new movie can generate based on a few given input attributes like budget, runtime, released year, popularity, and so on.

This study marks as a decision support system for the movie investment sector using machine learning techniques. This project helps investors associated with this business for avoiding investment risks. The system predicts an approximate success rate of a movie based on its profitability by analyzing historical data from different sources like Online rating, Director, Budget, Pre Release business, Genre, etc.

We are using two datasets for this problem is taken from the Kaggle. The movie gross region specified in this dataset is world wide of past years. As our dataset is large its training dataset is more than the test dataset.71.90% accuracy got by using Linear regression

|  |  |  |
| --- | --- | --- |
|  | **TABLE OF CONTENTS** |  |
| **CHAPTER No.** | **TITLE** | **PAGE No** |
|  | **ABSTRACT** | i |
|  | **LIST OF ABBREVIATIONS** | iv |
|  | **LIST OF FIGURES** | v |
| **1** | **INTRODUCTION** | **1** |
|  | 1.1 Introduction to project | 1 |
|  | 1.2 Overview of project | 2 |
|  | 1.3 Development method | 2 |
| **2.** | **AIM AND SCOPE OF THE PRESENT INVESTIGATION** | **3** |
|  | 2.1 AIM | 3 |
|  | 2.2 PROJECT SCOPE | 3 |
|  | 2.3 OVERALL OBJECTIVE | 3 |
|  | 2.4 PROPOSED SOLUTION | 4 |
|  | 2.5 SYSTEM ARCHITECTURE | 5 |
| **3** | EXPERIMENTAL MATERIALS AND METHODS, ALGORITHMS USED | **6** |
|  | 3.1 EXPERIMENTAL INVESTIGATION | 6 |
|  | 3.2 MATERIALS OR REQUIREMENTS | 6 |
|  | 3.2.1 FUNCTIONAL REQUIREMENTS | 7 |
|  | 3.2.2 NON-FUNCTIONAL REQUIREMENTS | 7 |
|  | 3.2.3 ENVIRONMENTAL REQUIREMENTS | 7 |
|  | 3.3 METHODS AND ALGORITHM USED | 8 |
|  | 3.4 ALGORITHM SELECTION | 9 |
|  | 3.4.1 SUPPORT VECTOR MACHINE | 9 |
|  | 3.4.2 RANDOM FOREST CLASSIFICATION  3.4.3 K-NN CLASSIFICATION  3.5 MODEL EVALUATION | 11  12  13 |
| **4** | **RESULTS AND DISCUSSION, PERFORMANCE** | **14** |
|  | 4.1 RESULTS | 14 |
|  | 4.2 PERFORMANCE ANALYSIS | 18 |
| **5** | **SUMMARY AND CONCLUSIONS** | **19** |
|  | **REFERENCES** | 20 |
|  | **APPENDIX** | 21 |
|  | 1. SCREENSHOTS 2. SOURCE CODE | 21-23  23-30 |
|  |  |  |

|  |
| --- |
|  |
|  |
|  |

# LIST OF ABBREVIATIONS

|  |  |  |
| --- | --- | --- |
| **S.NO** | **SHORTCUT** | **ABBREVATION** |
| 1  2  3  4  5 | ML  LR  RF  K-NN  HTML | Machine Learning  Linear Regression  Random Forest  K-Nearest Neighbour  Hypertext Markup Language |

# LIST OF FIGURES

|  |  |  |
| --- | --- | --- |
| **FIGURE NO.**  2.1  3.5 | **FIGURE NAME**  *ARCHITECTURE DIAGRAM*  *CONFUSION MATRIX* | **PAGE NO.**  5  13 |
| 4.1 | *ACCURACY OF DIFFERENT MODELS* | 14 |
| 4.2 | *RUNNING THE FLASK APP* | 14 |
| 4.3 | *PASTE THE URL AND CLICK ENTER* | 15 |
| 4.4 | *HOME PAGE OF WEB APPLICATION* | 15 |
| 4.5 | *ENTER DETAILS PAGE* | 16 |
| 4.6 | *RESULT* | 17 |

# CHAPTER 1

# INTRODUCTION

* 1. Box office gross prediction is an important problem in the film industry that governs financial decisions made by producers and investors. Generally, these predictions are made using some basic statistical techniques as described in. While these approaches are common practice, they often only provide a coarse estimate of revenue prediction before a film has been released. The goal of this project is to develop a computational model for predicting box office gross based on public data for movies extracted from popular online movie databases.

This project aims to predict the box office gross using Machine Learning methods of Linear regression, Support Vector Machine, Random Forest Model, and K-NN classification model. Inclusive of these models, pre-processing techniques such as Removing Duplicate Values, Null values, dealing with categorical data using the Encoding method, and Scaling features have been used. This model can be used as a valuable tool for predicting worldwide movie box office gross.

* 1. This study marks as a decision support system for the movie investment sector using machine learning techniques. This project helps investors associated with this business for avoiding investment risks. The system predicts an approximate success rate of a movie based on its profitability by analyzing historical data from different sources like Online rating, Director, Budget, Pre Release business, Genre, etc.
  2. ML algorithms are new techniques to handle the movie box office gross data sets. This approach can help movie box office management and professionals explore better results in gross. Several statistical and machine learning approaches (e.g., simulation modeling, classification, and inference) have been used by researchers for better prediction. The gross results are more data-driven than model-dependent. We are using linear regression and some other algorithms in that we are picking the most accurate and efficient one as the best algorithm and creating a Flask app and building a website and predicting the analysis.

**CHAPTER 2**

# AIM AND SCOPE OF THE PRESENT INVESTIGATION

**2.1AIM:**

The main aim of this project is to analyze the parameters of various classification algorithms and compare their predictive accuracies to discover the best classifier for determining movie box office gross.

**2.2 PROJECT SCOPE**:

The film industry has grown immensely over the past few decades generating billions of dollars of revenue for the stakeholders . Now people can watch movies online and offline on a variety of mobile devices during leisure or travel through Netflix, Youtube and downloads . A prediction system to assess the box office success of new movies can help the movie producers and directors make informed decisions when making the movie in order to increase the chance of profitability and box office gross success. New social media tools are constantly appearing which are enabling people to gather information on films and post comments about movies. These comments can influence the initial prediction about the box office gross success of a movie which some of the existing research do not take into account. Critic reviews often come out a few days before the film is released and may, therefore, help in prediction and at the same time influence the movie box office gross.

**2.3 OVERALL OBJECTIVE**:

The project's primary goal is to train an ML model with a given Algorithm that can predict the movie box office gross. The main challenge of this project is to understand the dataset, deal with missing values, use the right performance metrics for the algorithm and train the model with good accuracy for classification. Using python and python integrated modules helps to face the challenges of a dataset and make an efficient model for predicting things. And integrate to Flask-based web application. Users can predict the disease by entering parameters in the web application.

**2.4 PROPOSED SOLUTION**:

This is a classic example of supervised learning. We have been provided with a fixed number of features for each data point, and we will aim to train a variety of Supervised Learning algorithms on this data, so that, when a new data point arises, our best performing classifier can be used to categorize the data point as a positive example or negative. Exact details of the number and types of algorithms used for training are included in the 'Algorithms and Techniques' sub-section of the 'Analysis' part.

This project focuses on the related works of various gross data sets such that algorithms were implemented using Jupyter which is a machine learning software written in Python. Various attributes that are essential in the prediction of gross were examined and the dataset of movie box office was also evaluated. This project compares various classification algorithms such as Linear regression, Random Forest, Support Vector Machine, and KNN Classification Algorithm to identify the best technique.

Based on this study, Linear regression with the highest accuracy outperformed the other algorithms and can be further utilized in the prediction of liver disease recommended to the user.

Later by using the Flask app create Html files and create a user interface to display whether the patient has a liver problem or not.

**2.5 SYSTEM ARCHITECTURE:**



*FIG – 2.1 ARCHITECTURE DIAGRAM*

**CHAPTER 3**

# EXPERIMENTAL MATERIALS AND METHODS, ALGORITHMS USED

**3.1 EXPERIMENTAL INVESTIGATIONS**

Coming to analysis or investigations three supervised learning approaches are selected for this problem. Movies is taken that all these approaches are fundamentally different from each other so that we can cover as wide an umbrella as possible in terms of possible approaches.

For each algorithm, we will try out different values of a few hyperparameters to arrive at the best possible classifier. This will be carried out with the help of the grid search cross-validation technique.

**3.2 MATERIALS OR REQUIREMENTS:**

Requirements are the basic constraints that are required to develop a system. Requirements are collected while designing the system. The following are the requirements that are to be discussed.

1. Functional requirements
2. Non-Functional requirements
3. Environmental requirements
4. Hardware requirements
5. Software requirements

* + 1. **FUNCTIONAL REQUIREMENTS:**

The software requirements specification is a technical specification of requirements for the software product. It is the first step in the requirements analysis process. It lists the requirements of a particular software system.

* + 1. **NON-FUNCTIONAL REQUIREMENTS**:

Process of functional steps,

1. Problem define

2. Preparing data

3. Evaluating algorithms

4. Improving results

5. Prediction of the result

* + 1. **ENVIRONMENTAL REQUIREMENTS:**

1. Software Requirements:

Operating system: Windows

Tool: Anaconda with Jupyter notebook, spyder

MS Excel 2013

1. Hardware Requirements:
2. Internet connection to download and activate.
3. Minimum 10GB free disk space
4. Windows 8.1 or 10 (64-bit version only) is required.
5. Minimum System Requirements To run Office Excel

2013, your computer needs to meet

the following minimum hardware requirements:

● 500-megahertz (MHz)

● 256 megabytes (MB) RAM ● 1.5 gigabytes (GB) available space ● 1024x768 or higher resolution monitor

* 1. **METHODS AND ALGORITHMS USED:**

Since this is a classification problem with the binary response, the method we attempt to try includes a support vector machine, random Forest, and k-nearest neighbors algorithms. In machine learning and statistics, classification is a supervised learning approach in which the computer program learns from the data input given to it and then uses this learning to classify new observations. This data set may simply be bi-class. (Like identifying whether the person is subscribed to a term deposit or not, whether a person is interested to subscribe) or it may be multi-class too. Some examples of classification problems are speech recognition, handwriting recognition, biometric identification, document classification, etc. In Supervised Learning, algorithms learn from labeled data. After understanding the data, the algorithm determines which label should be given to new data based on pattern and associates the patterns to the unlabeled new data.

**Used Python Packages:**

* **Numpy:**

It is an open-source numerical Python library. It contains a multidimensional array and matrix data structures and can be used to perform mathematical operations.

* **Scikit-learn:**

It is a free machine learning library for Python. It features various algorithms like support vector machine, random forests, and k-neighbours, and it also supports Python numerical and scientific libraries like NumPy and SciPy.

* **Matplotlib and Seaborn:**

Matplotlib is mainly deployed for basic plotting. Visualization using Matplotlib generally consists of bars, pies, lines, scatter plots, and so on. Seaborn: Seaborn, on the other hand, provides a variety of visualization patterns. It uses fewer syntax and has easily interesting default themes.

* **Pandas:**

It is a fast, powerful, flexible, and easy-to-use open-source data analysis and manipulation tool, built on top of the Python programming language.

* **Pickle:**

The pickle module implements serialization protocol, which provides an ability to save and later load Python objects using a special binary format.

* 1. **ALGORITHM SELECTION:**

When we have the structured dataset, and we want to estimate the continuous or categorical outcome then we use supervised machine learning methodologies like regression and classification techniques. When we have unstructured data and want to predict the clusters of items to which a particular input test sample belongs, we use unsupervised algorithms. An actual data scientist applies multiple algorithms to get a more accurate model.

* + 1. **SUPPORT VECTOR MACHINE (SVM):**

Support Vector Machine (SVM) is a supervised machine learning algorithm that can be used for both classification and regression challenges. However, it is mostly used in classification problems. Support Vectors are simply the coordinates of individual observation. The goal of a support vector machine is not only to draw hyperplanes and divide data points, but to draw the hyperplane that separates data points with the largest margin, or with the most space between the dividing line and any given data point.

* **SVM. fit ():** Model fitting is a measure of how well a machine learning model generalizes to similar data to that on which it was trained. A well-fitted model produces more accurate outcomes.
* **SVM. predict ():** The predict () function accepts only a single argument which is usually the data to be tested. It returns the labels of the data passed as an argument based upon the learned or trained data obtained from the model. Thus, the predict() function works on top of the trained model and makes use of the learned label to map and predict the labels for the data to be tested.
* **Accuracy\_score ():** When taking scientific measurements, it is important to be both accurate and precise. Accuracy represents how close a measurement comes to its true value. This is important because bad equipment, poor data processing, or human error can lead to inaccurate results that are not very close to the truth.
* **Confusion\_matrix ():** A Confusion matrix is an N x N matrix used for evaluating the performance of a classification model, where N is the number of target classes. The matrix compares the actual target values with those predicted by the machine learning model. The rows represent the predicted values of the target variable.

A support vector machine constructs a hyperplane or set of hyperplanes in a high- or infinite-dimensional space. A good separation is achieved by the hyperplane that has the largest distance to the nearest training data point of any class (so-called functional margin) since in general the larger the margin the lower the generalization error of the classifier.



Depending on the kernel type we choose the kernel parameters to have to be set. Which kernel type performs best, depends on the application and can be determined by using cross-validation.

* + 1. **RANDOM FOREST FOR CLASSIFICATION:**

Random forest can also be used for classification. It is one of the most broadly used machine learning algorithms for classification. Whether the response variable is continuous or categorical, it works in both cases. According to Friedman et al. random forests starts to become stable at around 200 trees, whereas at 1000 trees the boosting of this still keeps on improving. If trees are much smaller or there is a presence of shrinkage then the process of boosting starts to reduce. The important function of the RF is the utilization of out-of-bag (OOB) samples. For each value zi = (xi, Yi), in which term zi did not appear that makes the RF predictor by averaging only those trees which are consistent with the bootstrap samples. The OOB error estimate is then nearly identical to that which is getting by N fold cross-validation. In contrast to the other nonlinear estimators, it is possible to fit the RF in one sequence with the cross-validation being completed. The training can be finished if the OOB error stabilizes itself.

## Algorithm of RF:

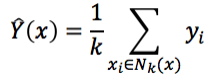
The algorithm of RF is considered as best in terms of accuracy. Even if the data is too large or includes thousands of input variables, the efficiency does not decrease and at the same time prevents to be overfitting as well and there is noneed for data pruning in it. It can be used for methods i.e. selection of best subset as well as imputation of the missing values and in both cases it performs very fine and efficient. The forest which is produced as output is also proficient for adding the data for the future.

In RF we have a learning set which is L = {(X1, Y1), ...,(Xn, Yn)} which should contain the observations of independent random vector (X, Y), where X is a vector of explanatory variables i.e. X = (X1, ..., XP ) and X< p and Y is the class label if the in the case of classification.

* + 1. **K-NN CLASSIFICATION:**

Regular linear regression makes assumptions about the structure of the data (high bias), but its predictions are stable (low variance). We need a more flexible model that makes fewer assumptions. In contrast to linear regression methods, the k-nearest neighbour methods implement non-linear boundaries to our training and test data.

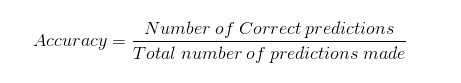
The k-NN method uses the average outcome value of its k nearest neighbors based on Euclidian distance.



* K-Fold: K-Fold splits a given data set into a K number of sections/folds where each fold is used as a testing set at some point.
* Stratified K-Fold: This is a variation of the K-Fold that returns stratified folds. The folds are made by preserving the percentage of samples for each class.
  1. **MODEL EVALUATION:**

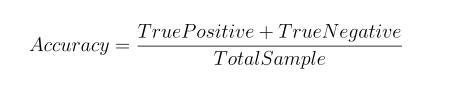
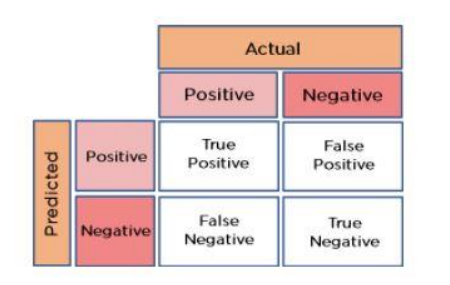
### Classification Accuracy:

Classification Accuracy is what we usually mean when we use the term accuracy. It is the ratio of number of correct predictions to the total number of input samples.



### Confusion Matrix:

Confusion Matrix as the name suggests gives us a matrix as output and describes the complete performance of the model.



*FIG 3.5 – CONFUSION MATRIX*

**CHAPTER 4**

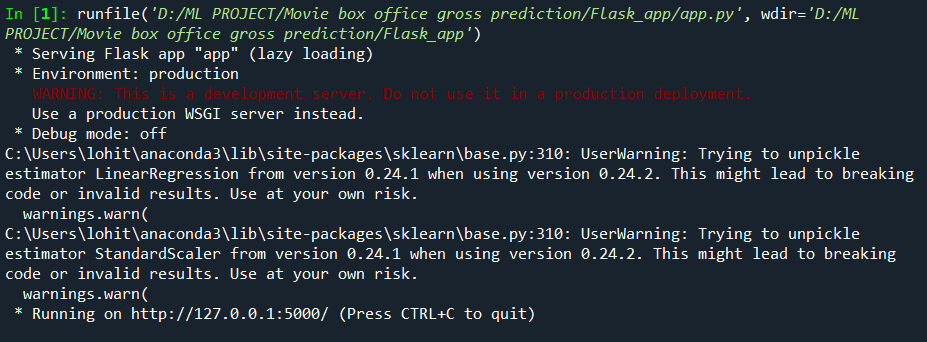
# RESULTS AND DISCUSSION, PERFORMANCE ANALYSIS

## RESULTS

## 

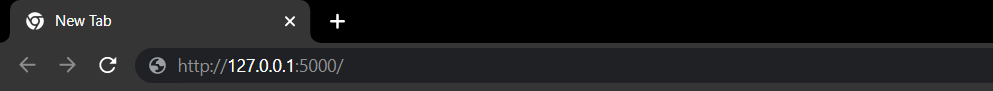
*FIG 4.1 – ACCURACY OF THE MODEL*

From the above figure we can see the accuracy of the model and the Random Forest model and linear regression model gives the same accuracy we pick this model for further process.



*FIG 4.2 – RUNNING THE FLASK APP*

After picking the Random Forest model then developing a flask app and run the code in spyder we get the following output and copy-paste the URL.

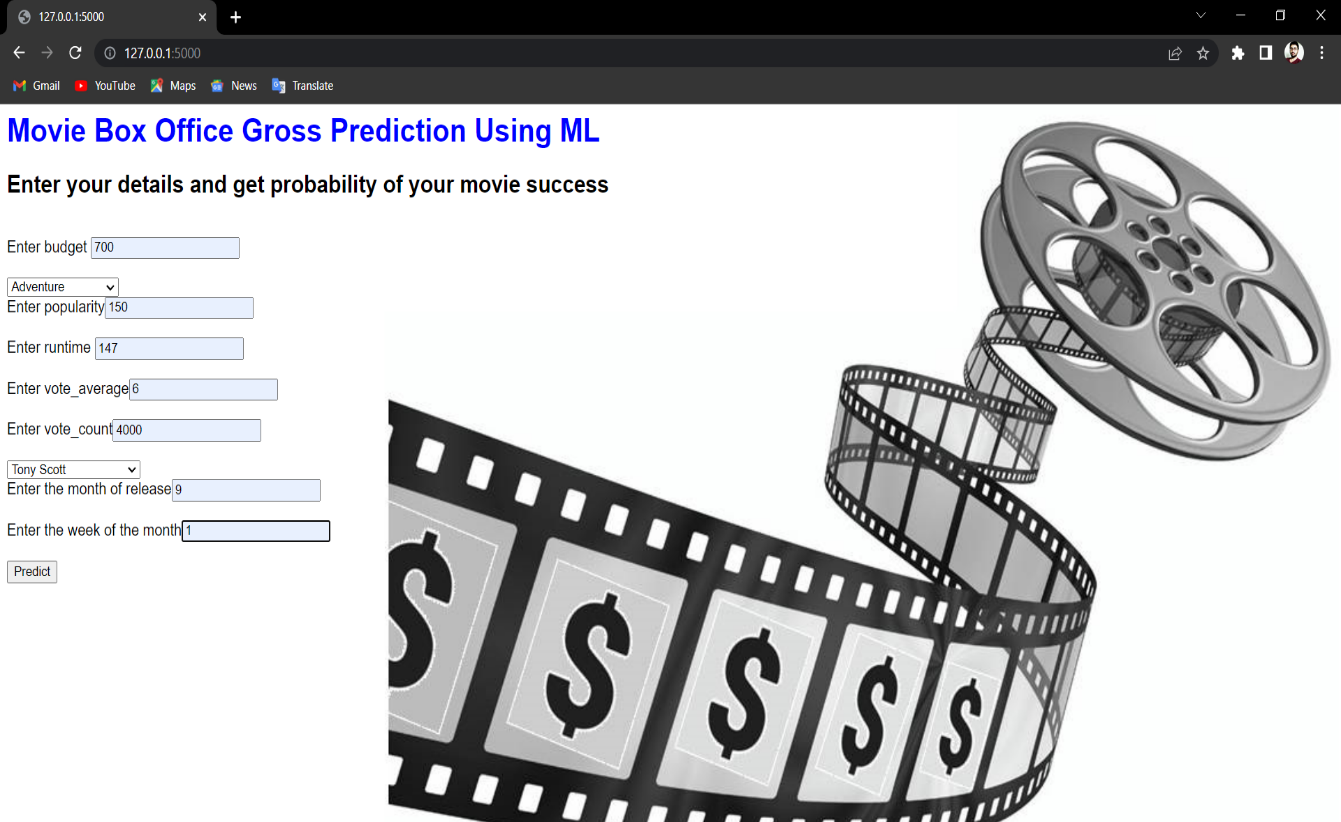


*FIG 4.3 – PASTE THE URL AND CLICK ENTER*

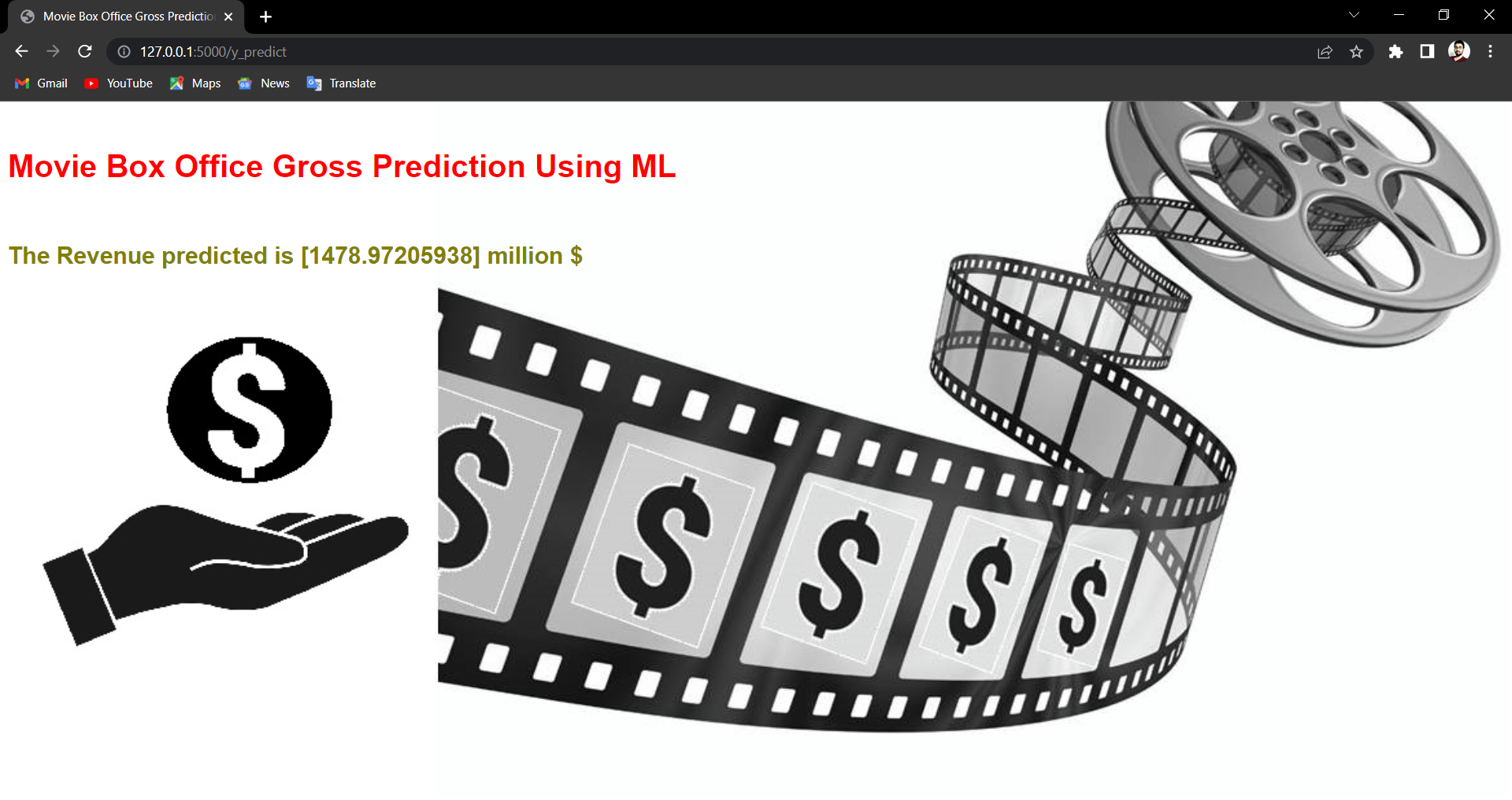
Paste the URL in the local browser and click enter’



*FIG 4.4 – HOME PAGE OF WEB APPLICATION*



*FIG 4.5 – ENTER DETAILS PAGE*



*FIG 4.6 – RESULT*

From the above figures 4.6 the results are coming correctly, by entering the values on the ENTER DETAILS page we get the result .

.

## PERFORMANCE ANALYSIS:

## One of the first desires that probably comes to mind is efficiency. When building your website, you want to be able to reach as many people as you can.

## This application can further be developed with more ideas and implementation and by using different algorithms. The accuracy score of the model can be further improved by using decision tree and also by increasing the data set, K-Nearest Neighbors algorithm is also one of the pertinent methods which can be used to predict the movie box office gross accurately. It proposes to improve the accuracy further.

**CHAPTER 5**

# SUMMARY AND CONCLUSIONS

# Prompt and timely accurate prediction of Movie box office gross plays a vital role in decreasing for producers and stakeholders. In this paper, an attempt is made to predict the presence of Movie gross using Support Linear regression , vector machine, Random Forest , K-NN classification methods of Machine Learning.

# I developed a computational model for movie box office gross prediction using a combination of features extracted from movie database metadata, budget-revenue relationship graphs, popularity-revenue relationship graphs, and movie-Revenue relationship graphs. I demonstrated that by using features extracted from these runtime-movies and revenue-movie relationship graphs, we are able to create a more accurate model than using metadata features alone.

# Among ML classification methods, LR and RF performed equal accuracy. Although, the accuracy levels for two methods performed well based on the testing data set.

# There are a variety of extensions that could be made to the existing model proposed in this paper. One alternative would be to model movie gross as a continuous quantity rather than a discrete quantity. Another extension would be to introduce a temporal model for how movie genre and actor popularity change over time, which one might suspect would lead to more accurate gross predictions. Finally, perhaps the biggest improvement that could be made would be to acquire more Movie box office gross data.

## REFERENCES

[1] 1.11. ensemble methods. scikit. (n.d.). Retrieved November 2, 2021, from [https://scikit-learn.org/stable/modules/ensemble.html#bagging-meta-estimator](https://scikit-learn.org/stable/modules/ensemble.html).

[2] 5. visualizations. scikit. (n.d.). Retrieved November 2, 2021, from <https://scikit-learn.org/stable/visualizations.html>.

[3] The absolute basics for beginners. NumPy. (n.d.). Retrieved November 2, 2021, from <https://numpy.org/doc/stable/user/absolute_beginners.html>.

[4] Matplotlib.pyplot. matplotlib.pyplot - Matplotlib 3.4.3 documentation. (n.d.). Retrieved November 2, 2021, from <https://matplotlib.org/stable/api/_as_gen/matplotlib.pyplot.html>.

[5] Narkhede, S. (2021, June 15). Understanding confusion matrix. Medium. Retrieved November 2, 2021, from <https://towardsdatascience.com/understanding-confusion-matrix-a9ad42dcfd62>.

[6] Simonoff, J. S. and Sparrow, I. R. Predicting movie grosses: Winners and losers, blockbusters and sleepers. In Chance, 2000.

[7] Joshi, M., Das, D., Gimpel, K., and Smith, N. A. Movie Reviews and Revenues: An Experiment in Text Regression. In Proceedings of the North American Chapter of the Association for Computational Linguistics Human Language Technologies Conference, 2010.

[8] Sharda, R. and Delen, D. Predicting box-office success of motion pictures with neural networks. In Expert Systems with Applications, 2006.

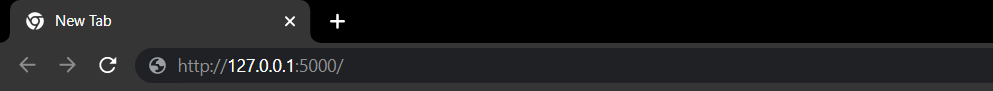
# APPENDIX:

# 

# SCREENSHOTS

The output of the flask file

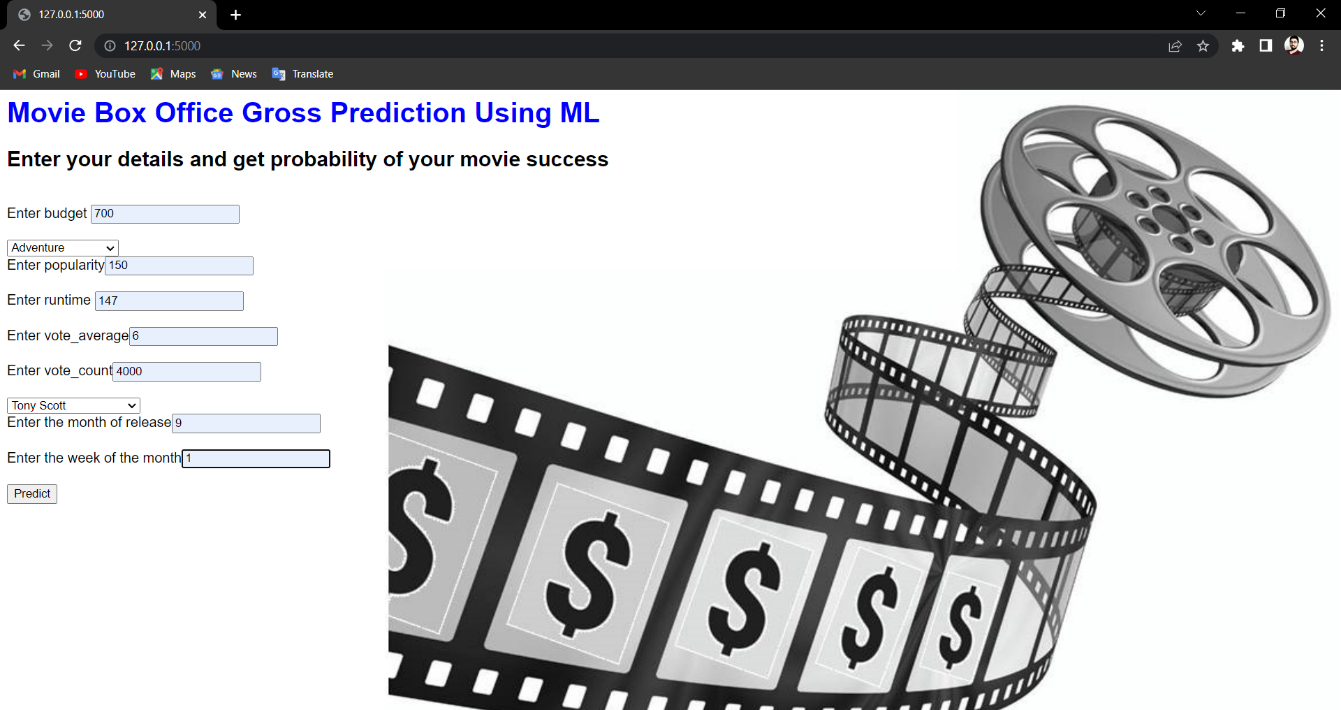
Pasting URL

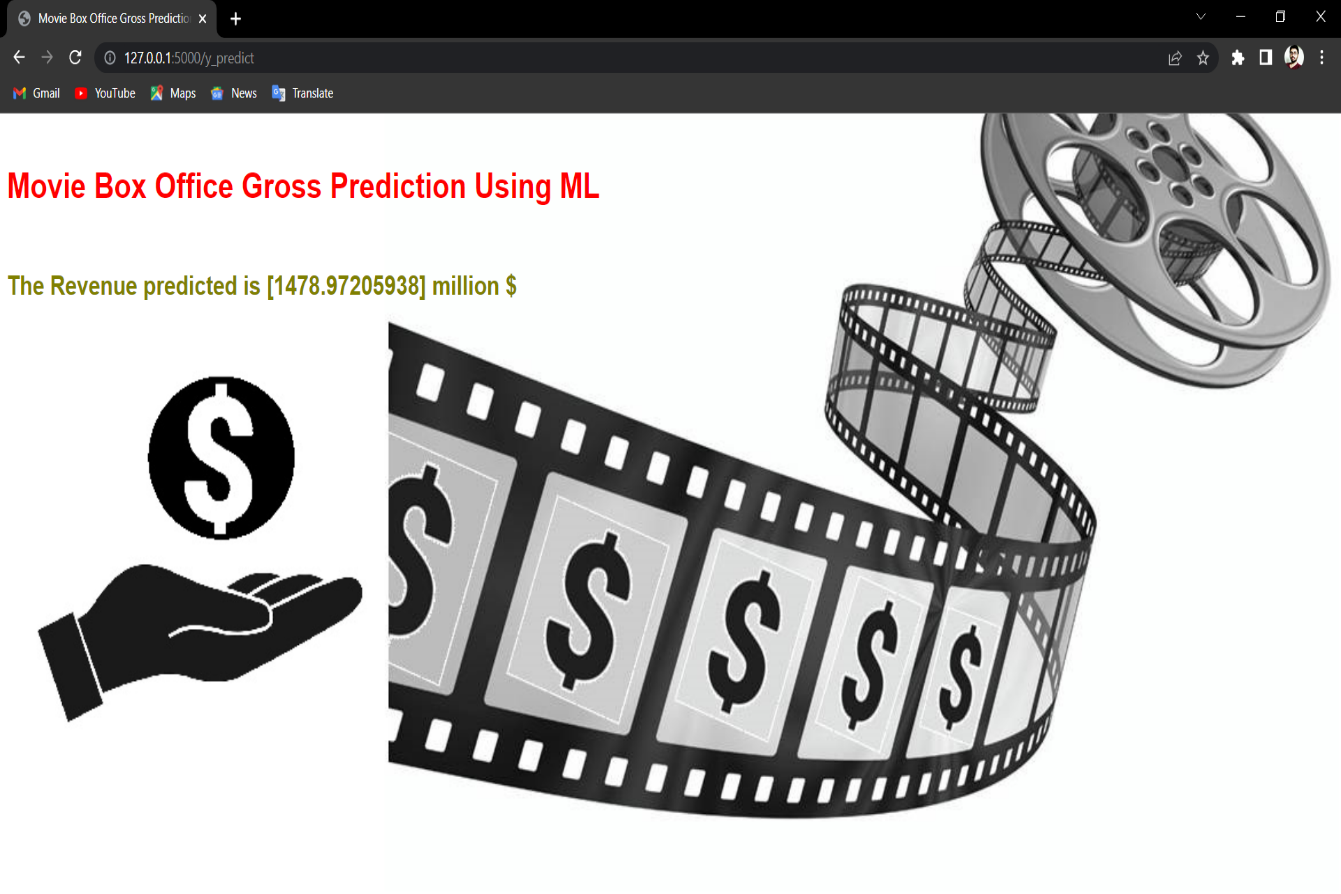


Home page of web application



Enter details page





Output for the project.

### SOURCE CODE:

import pandas as pd #data manipulation

import numpy as np #Numerical Analysis

import seaborn as sns #data visualization

import json #for reading json object

import matplotlib.pyplot as plt #data visualization

import pickle # For saving the model file

from wordcloud import WordCloud #to create word clouds

from ast import literal\_eval#to evaluate the string as pyhton expression

#Reading the dataset by using pandas read\_csv function

credits=pd.read\_csv(r"D:\ML\_training may 2020\Projects\_50\Final\Movie Box Office Gross Prediction Using ML\dataset\tmdb\_5000\_credits.csv")

movies\_df=pd.read\_csv(r"D:\ML\_training may 2020\Projects\_50\Final\Movie Box Office Gross Prediction Using ML\dataset\tmdb\_5000\_movies.csv")

#head() gives us first 5 rows of the dataset

credits.head()

credits.tail()

movies\_df.head()

#columns in the dataset

print("credits:",credits.columns)

print("movies\_df:",movies\_df.columns)

#Shape of the dataset

print("credits:",credits.shape)

print("movies\_df:",movies\_df.shape)

#Renaming the columns

credits\_column\_renamed=credits.rename(index=str,columns={"movie\_id":"id"})

movies=movies\_df.merge(credits\_column\_renamed,on="id")

movies.head()

movies.shape

#information about the datset

movies.info()

movies.describe()

# changing the crew column from json to string

movies['crew'] = movies['crew'].apply(json.loads)

def director(x):

for i in x:

if i['job'] == 'Director':

return i['name']

movies['crew'] = movies['crew'].apply(director)

movies.rename(columns={'crew':'director'},inplace=True)

from ast import literal\_eval

features = ['keywords','genres']

for feature in features:

movies[feature] = movies[feature].apply(literal\_eval)

# Returns the top 1 element or entire list; whichever is more.

def get\_list(x):

if isinstance(x, list):

names = [i['name'] for i in x]

#Check if more than 3 elements exist. If yes, return only first three. If no, return entire list.

if len(names) > 1:

names = names[:1]

return names

#Return empty list in case of missing/malformed data

return []

print (type(movies.loc[0, 'genres']))

features = ['keywords', 'genres']

for feature in features:

movies[feature] = movies[feature].apply(get\_list)

movies['genres']

movies['genres'] = movies['genres'] .str.join(', ')

movies['genres']

movies.head()

print("movies:",movies.shape)

#corr() is to find the relationship between the columns

movies.corr()

movies.isnull().any()

movies.isnull().sum()

sns.heatmap(movies.isnull(),yticklabels=False,cbar=False,cmap='viridis')

#Dropping the null values

movies = movies.dropna(subset = ['director','runtime'])

movies.isnull().sum()

movies.head(5)

#Divide the revenue and budget columns by 1000000 to convert $ to million $

movies["revenue"]=movies["revenue"].floordiv(1000000)

movies["budget"]=movies["budget"].floordiv(1000000)

movies.head(5)

#As there cannot be any movie with budget as o,let us remove the rows with budget as 0

movies = movies[movies['budget'] != 0]

movies.info()

#Let us create three new columns and extract date,month and Day of the week from the release date

movies['release\_date'] = pd.DataFrame(pd.to\_datetime(movies['release\_date'],dayfirst=True))

movies['release\_month'] = movies['release\_date'].dt.month

movies['release\_DOW'] = movies['release\_date'].dt.dayofweek

sns.boxplot(x=movies['runtime'])

plt.title('Boxplot of Runtime')

sns.boxplot(x=movies['revenue'])

plt.title('Boxplot of Revenue')

sns.boxplot(x=movies['budget'])

plt.title('Boxplot of Budget')

#removing outliers

bq\_low = movies['budget'].quantile(0.01)

bq\_hi = movies['budget'].quantile(0.99)

rq\_low = movies['runtime'].quantile(0.01)

rq\_hi = movies['runtime'].quantile(0.99)

movies = movies[(movies['budget'] < bq\_hi) & (movies['budget'] > bq\_low) & (movies['runtime'] < rq\_hi) & (movies['runtime'] > rq\_low)]

movies.shape

sns.boxplot(x=movies['runtime'])

plt.title('Boxplot of Runtime(Outliers Removed)')

sns.boxplot(x=movies['budget'])

plt.title('Boxplot of Budget(Outliers Removed)')

sns.heatmap(movies.corr(), cmap='YlGnBu', annot=True, linewidths = 0.2);

#creating log transformation for reveune

movies['log\_revenue'] = np.log1p(movies['revenue']) #we are not using log0 to avoid & and null value as there might be 0 value

movies['log\_budget'] = np.log1p(movies['budget'])

#comapring distribution of reveune and log revune side by side with histogram

fig, ax = plt.subplots(figsize = (16, 6))

plt.subplot(1, 2, 1)

plt.hist(movies['revenue']);

plt.title('Distribution of revenue');

plt.subplot(1, 2, 2)

plt.hist(movies['log\_revenue']);

plt.title('Distribution of log transformation of revenue');

#let's create scatter plot

plt.figure(figsize=(16, 8))

plt.subplot(1, 2, 1)

plt.scatter(movies['budget'], movies['revenue'])

plt.title('Revenue vs budget fig(1)');

plt.subplot(1, 2, 2)

plt.scatter(movies['log\_budget'], movies['log\_revenue'])

plt.title('Log Revenue vs log budget fig(2)');

wordcloud = WordCloud().generate(movies.original\_title.to\_string())

sns.set(rc={'figure.figsize':(12,8)})

# Display the generated image:

plt.imshow(wordcloud, interpolation='bilinear')

plt.axis("off")

plt.show()

#let's creat column called has\_homepage and pass two value 1,0 (1, indicates has home page, 0 indicates no page)

movies['has\_homepage'] = 0

movies.loc[movies['homepage'].isnull() == False, 'has\_homepage'] = 1 #1 here means it has home page

#since has\_homepage is categorical value we will be using seaborn catplot.

sns.catplot(x='has\_homepage', y='revenue', data=movies);

plt.title('Revenue for movie with and w/o homepage');

sns.jointplot(movies.budget, movies.revenue);

sns.jointplot(movies.popularity, movies.revenue);

sns.jointplot(movies.runtime, movies.revenue);

plt.show()

plt.figure(figsize=(15,8))

sns.jointplot(movies.release\_month, movies.revenue);

plt.xticks(rotation=90)

plt.xlabel('Months')

plt.title('revenue')

movies.info()

movies\_box = movies.drop(['homepage','id','keywords','original\_language','original\_title','overview','production\_companies',

'production\_countries','release\_date','spoken\_languages','status','tagline',

'title\_x','title\_y','cast','log\_revenue','log\_budget','has\_homepage'],axis = 1)

movies\_box.dtypes

from sklearn.compose import ColumnTransformer

from sklearn.preprocessing import OneHotEncoder

ct= ColumnTransformer([("on",OneHotEncoder(),[3])],remainder='passthrough')

x=ct.fit\_transform(x)

x

# Label encoding features to change categorical variables into numerical one

from sklearn.preprocessing import LabelEncoder

le = LabelEncoder()

movies\_box['director','genres']= le.fit\_transform(movies\_box['director','genres'])

movies\_box.head()

# Label encoding features to change categorical variables into numerical one

from sklearn.preprocessing import LabelEncoder

from collections import Counter as c

cat=['director','genres']

for i in movies\_box[cat]:#looping through all the categorical columns

print("LABEL ENCODING OF:",i)

LE = LabelEncoder()#creating an object of LabelEncoder

print(c(movies\_box[i])) #getting the classes values before transformation

movies\_box[i] = LE.fit\_transform(movies\_box[i]) # trannsforming our text classes to numerical values

print(c(movies\_box[i])) #getting the classes values after transformation

movies\_box.head(3)

mapping\_dict ={}

category\_col=["director","genres"]

for col in category\_col:

LE\_name\_mapping = dict(zip(LE.classes\_,

LE.transform(LE.classes\_)))

mapping\_dict[col]= LE\_name\_mapping

print(mapping\_dict)

movies\_box.head()

#Dependent Variables

x=movies\_box.iloc[:,[0,1,2,4,5,6,7,8,9]]

x=pd.DataFrame(x,columns=['budget','genres','popularity','runtime','vote\_average','vote\_count','director'

,'release\_month','release\_DOW'])

X

#Dependent Variables

y=movies\_box.iloc[:,3]

y=pd.DataFrame(y,columns=['revenue'])

y

from sklearn.preprocessing import StandardScaler

sc = StandardScaler()

x=sc.fit\_transform(x)

x

pickle.dump(sc,open("scalar\_movies.pkl","wb"))

from sklearn.model\_selection import train\_test\_split

x\_train,x\_test,y\_train,y\_test=train\_test\_split(x,y,test\_size=0.1,random\_state=0)

from sklearn.linear\_model import LinearRegression

mr=LinearRegression()

mr.fit(x\_train,y\_train)

x\_test

y\_test[0:5]

y\_pred\_mr=mr.predict(x\_test)

y\_pred\_mr[0:5]

3.76955224\*100000000

y\_test

from sklearn import metrics

print("MAE:",metrics.mean\_absolute\_error(y\_test,y\_pred\_mr))

print("RMSE:",np.sqrt(metrics.mean\_absolute\_error(y\_test,y\_pred\_mr)))

from sklearn.metrics import r2\_score

r2\_score(y\_test,y\_pred\_mr)

from sklearn.ensemble import RandomForestRegressor

rf = RandomForestRegressor(n\_jobs = -1, random\_state = 42)

rf.fit(x\_train, y\_train)

y\_pred\_mr=mr.predict(x\_test)

r2\_score(y\_test,y\_pred\_mr)

import pickle

pickle.dump(mr,open("model\_movies.pkl","wb"))

model=pickle.load(open("model\_movies.pkl","rb"))

scalar=pickle.load(open("scalar\_movies.pkl","rb"))

input=[[50,8,20.239061,88,5,366,719,7,3]]

input=scalar.transform(input)

prediction = model.predict(input)

prediction

mr.score(x\_test,y\_test)

**FLASK APP CODE:**

import numpy as np

from flask import Flask, request, jsonify, render\_template

import pickle

import pandas as pd

app = Flask(\_\_name\_\_) #initialising the flask app

filepath="model\_movies.pkl"

model=pickle.load(open(filepath,'rb'))#loading the saved model

scalar=pickle.load(open("scalar\_movies.pkl","rb"))#loading the saved scalar file

@app.route('/')

def home():

return render\_template('Demo2.html')

@app.route('/y\_predict',methods=['POST'])

def y\_predict():

'''

For rendering results on HTML

'''

input\_feature=[float(x) for x in request.form.values()]

features\_values=[np.array(input\_feature)]

feature\_name=['budget','genres','popularity','runtime','vote\_average','vote\_count',

'director','release\_month','release\_DOW']

x\_df=pd.DataFrame(features\_values,columns=feature\_name)

x=scalar.transform(x\_df)

# predictions using the loaded model file

prediction=model.predict(x)

print("Prediction is:",prediction)

return render\_template("resultnew.html",prediction\_text=prediction[0])

if \_\_name\_\_ == "\_\_main\_\_":

app.run(debug=False)