Movie Box Office Gross Prediction Introduction using IBM Watson Machine Learning

Jeya Santhosh Kumar D 19BCE0762

Nowadays, hundreds of movies are produced and released every year. Among them, there exist both great movies and crappy ones. Therefore, how do we know their qualities before we do not see the movie ourselves? Or how can we choose a great movie to enjoy and relax on our weekends? Most of the time, we will turn to the movie score or have a look at its review to decide. IMDb website is just a good choice to refer at this time. Due to its popularity, IMDb website contains a great deal of information about movies and the comments from audiences. The scores which IMDb gives are highly recognized by the public, representing the quality of content as well as audience's favour to some extent. Therefore, in this research, we will try to unveil the important factors influencing the score on IMDb website and propose an efficient approach to predict it. The data we use in our paper comes from IMDb 5000 Movie Dataset on Kaggle. It contains 28 variables for 5042 movies and 4906 posters, spanning across 100 years in 66 countries. There are 2399 unique director names and thousands of actors/actresses. The worldwide Box office revenue 2016 was 38.3 billion USD with hundreds of new movies made each year. If one can use a computer to predict how successful a movie will be even before it is released, this would be a powerful tool to use. An aspiring Hollywood director or a movie studio with some technical skills could predict whether their movie idea is going to be a safe investment. With the vast amount of data published on the Internet and the increasing power of the modern computer, is it possible to take advantage of these resources to make predictions. The study can be used as a proof of concept for applications in other areas, and should highlight some of the challenges one needs to overcome to successfully create a prediction model. This idea could in theory be extended to predict credit ratings, the stock market or housing market. The only requirement being a vast and reliable data source. When combining the questions mentioned above to form a problem statement, formulating good as a measurement of a movies rating and sales, the following problem statement was produced. Movie ratings in recent years are influenced by many factors that makes the accurate prediction of ratings for the new movies being released a difficult task. There also have been various semantic analysis techniques to analyze user reviews which were applied to analyze the IMDb movie ratings. None of the studies has succeeded in suggesting a model good enough to be used in the industry. In this project, we attempt to use the IMDb dataset to predict the Cinema has a profound impact on our society. Cinema is one of the most powerful media for mass communication in the world. Cinema has the capacity to influence society both locally and globally. Many different kinds of movies are made every year. Some movies portray historical events, some create a culture, while some provide fantasy, and some do many more. We perform an exploratory analysis of the data and observe some interesting phenomenon, which also helps us improve our prediction strategy as well as we will get to know about the features which affect the movie IMDb score. Our results finally show that we achieve a good prediction accuracy of IMDb score on this dataset

Literature Survey

Success of a movie primarily depends on the perspectives that how the movie has been justified. In early days, a number of people prioritized gross box office revenue, initially. Few previous work portend gross of a movie depending on stochastic and regression models by using IMDb data. Some of them categorized either success or flop based on their revenues and apply binary classifications for forecast. The measurement of success of a movie does not solely depend on revenue. Success of movies rely on a numerous issue like actors/actresses, director, time of release, background story etc. Further few people had made a prediction model with some pre-released data which were used as their features . In most of the case, people considered a very few features. As a result, their models work poorly. However, they ignored participation of audiences on whom success of a movie mostly depends. Although few people adopt many applications of NLP for sentiment analysis) and gathered movie reviews for their test domain. But the accuracy of prediction lies on how big the test domain is. A small domain is not a good idea for measurement. Again most of them did not take critics reviews in account. Besides, users' reviews can be biased as a fan of actor/actress may fail to give unbiased opinion. M. T. Lash and K. Zhao's main contribution was, firstly they developed a decision support system using machine learning, text mining and social network analysis to predict movie profitability not revenue. Their research features several features such as dynamic network features, plot topic distributions means the match between "what" and "who" and the match between "what" and "when" and the use of profit based star power measures. They analyzed movie success in three categories, audience based, released based and movie based. Their hypothesis based on the more optimistic, positive, or excited the audiences are about a movie, the more likely it is to have a higher revenue. Similarly, a movie with more pessimistic and negative receptions from the public may attract fewer people to fill seats. They retrieve data from different types of media. Such as Twitter, comments from YouTube, blogs, new 5 articles and movie reviews, star rating from reviews, the sentiment of reviews or comments have been used as a means for assessing audience's excitement towards a movie. Their original dataset collected from both Box-office Mojo and IMDb. They focused on the movies released in USA and excluded all foreign movies from their experiment. In A neural network had been used in the prediction of financial success of a box office movie before releasing the movie in theatres. This forecasting had been converted into a classification problem categorized in 9 classes. The model was represented with very few features. In A. Sivasantoshreddy, P. Kasat, and A. Jain tried to predict a movie box-office opening prediction using hype analysis. A neural network had been used in the prediction of financial success of a box office movie before releasing the movie in theatres. This forecasting had been converted into a classification problem categorized in 9 classes. The model was represented with very few features. In, it was tried to improve movie gross prediction through News analysis where quantitative news data generated by Lydia (high-speed text processing system for collecting and analyzing news data). It contained two different models (regression and k-nearest neighbour models). But they considered only high budget movies. The model failed if common word used as name and it could not predict if there were no news about a movie. M.H Latif, H. Afzal who used IMDB database only as their main source and their data was not clean. Again their data was inconsistent and very noisy as mentioned by them. So they used Central Tendency as a standard for filling missing values for different attributes. K. Jonas, N. Stefan, S. Daniel, F. Kai use sentiment and social network analysis for prediction their hypothesis was based on intensity and positivity analysis of IMDb sub forum Oscar Buzz. They had considered movie critics as the influencer and their predictive perspective. They used bag of word which gave wrong result when some words were used for negative means. There was no category award and only concerned with the award for best movie, director, actors/actress and supporting actors/actress. In some cases, success prediction of a movie was made through neural network analysis. Some researchers made prediction based on social media, social network and hype analysis where they calculated positivity and number of comments related to a particular movie. Moreover, few people had predicted Box Office movies' success based on Twitter tweets and YouTube comments.

Theoretical Analysis

Hardware Requirements

i5 Processor

GTX 1060 GPU

8GB RAM

Software Requirements

Sci Kit Learn

Python3

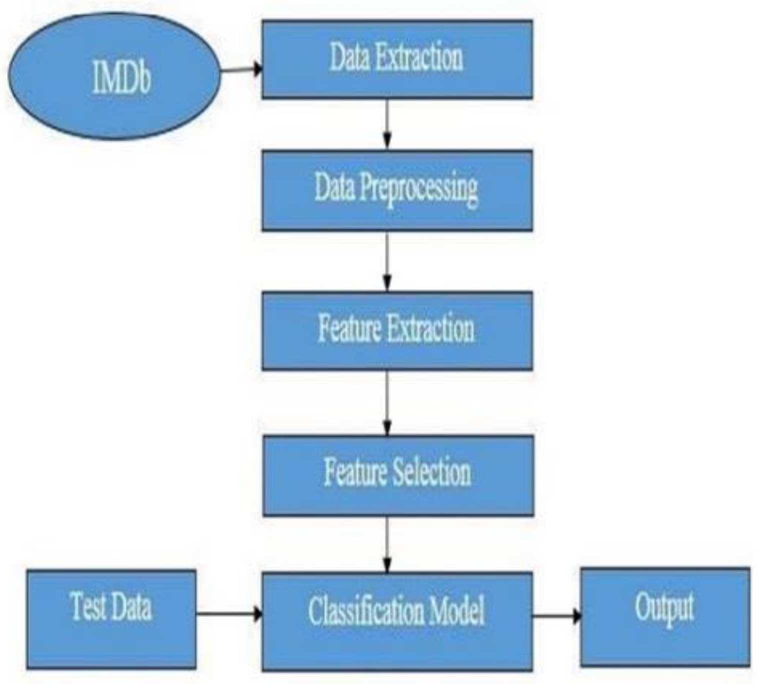
Flask

Numpy

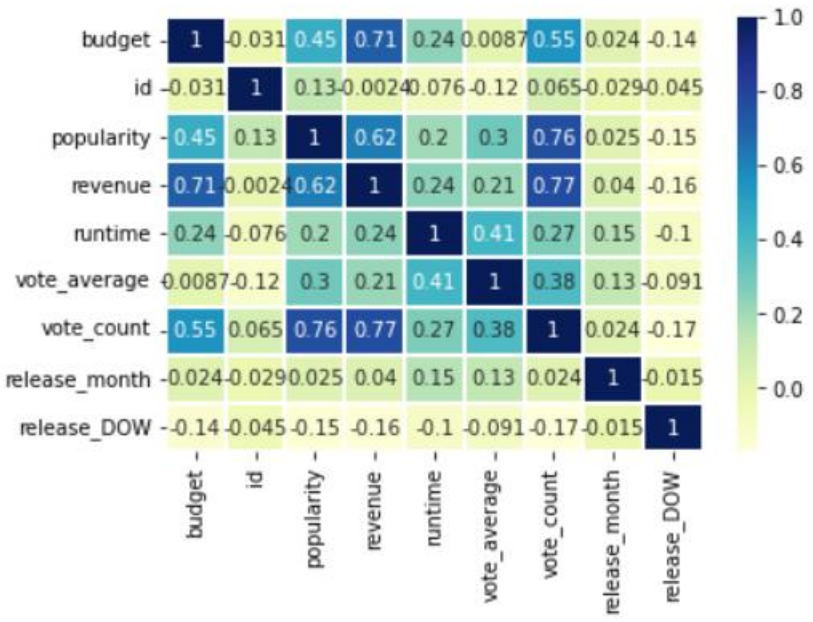
Pandas

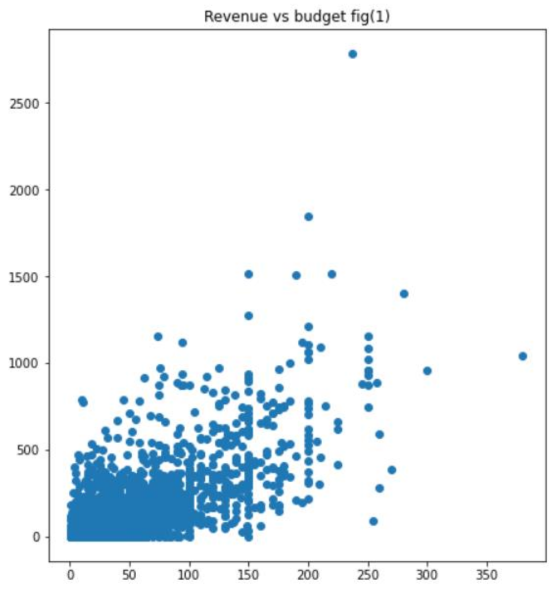
PyCharm IDE

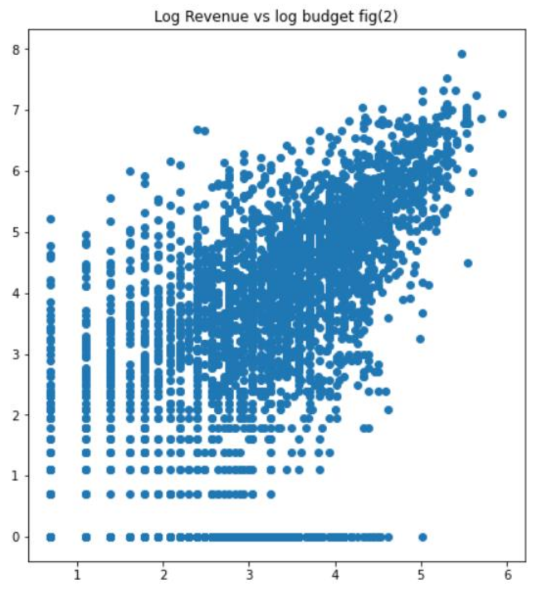
Flowchart

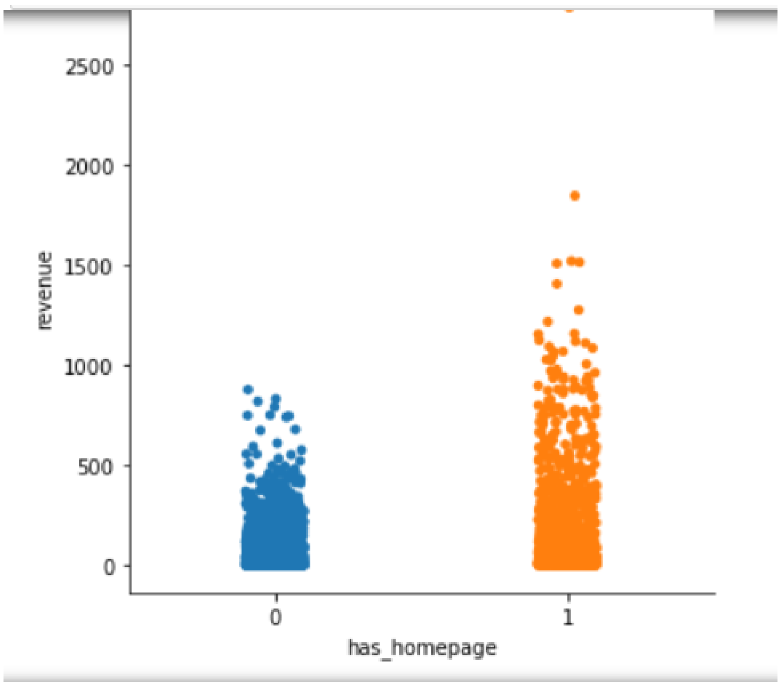


Experimental Investigations

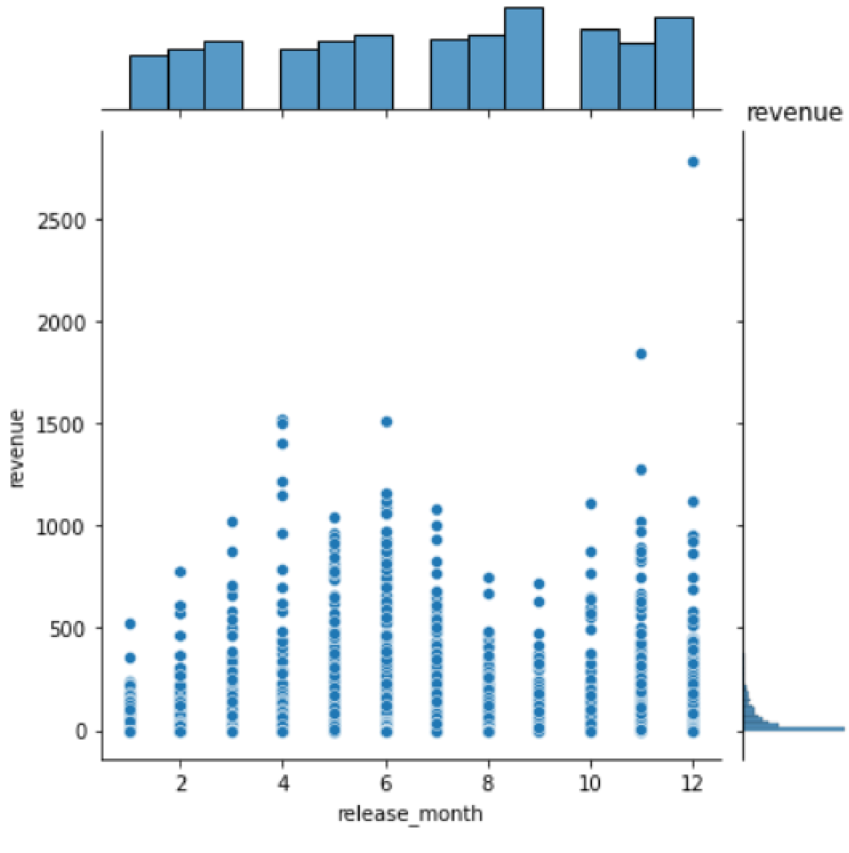








revenue
2500
2000
1500
1000
500
0
0
1
has_homepage



revenue
2500
2000
1500
1000
500
0
2
4
6
8
10
12
release_month
revenue

Advantages and Disadvantages of Proposed Solution

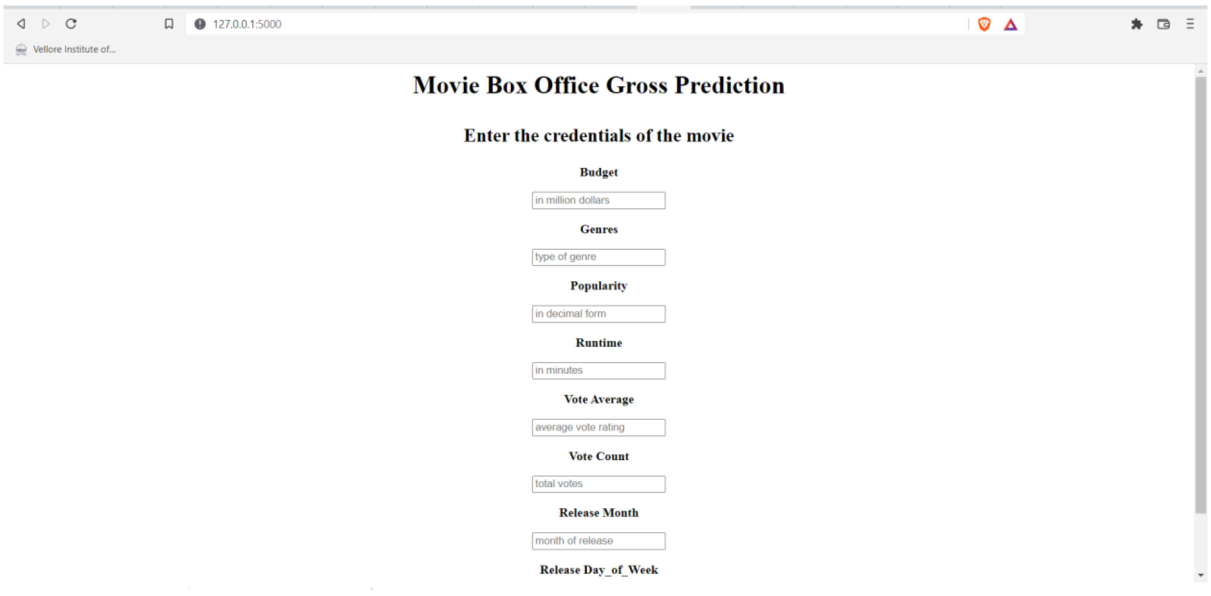
Advantages

Linear Regression is simple to implement and easier to interpret the output coefficients. Linear Regression is susceptible to over-fitting but it can be avoided using some dimensionality reduction techniques, regularization (L1 and L2) techniques and cross-validation.

Disadvantages

On the other hand in linear regression technique outliers can have huge effects on the regression and boundaries are linear in this technique. But then linear regression also looks at a relationship between the mean of the dependent variables and the independent variables. Just as the mean is not a complete description of a single variable, linear regression is not a complete description of relationships among variables. Diversely, linear regression assumes a linear relationship between dependent and independent variables. That means it assumes that there is a straight-line relationship between them. It assumes independence between attributes.

Result



Movie Box Office Gross Prediction

Enter the credentials of the movie

Budget

in million dollars

Genres

type of genre

Popularity

in decimal form

Runtime

in minutes

Vote Average

average vote rating

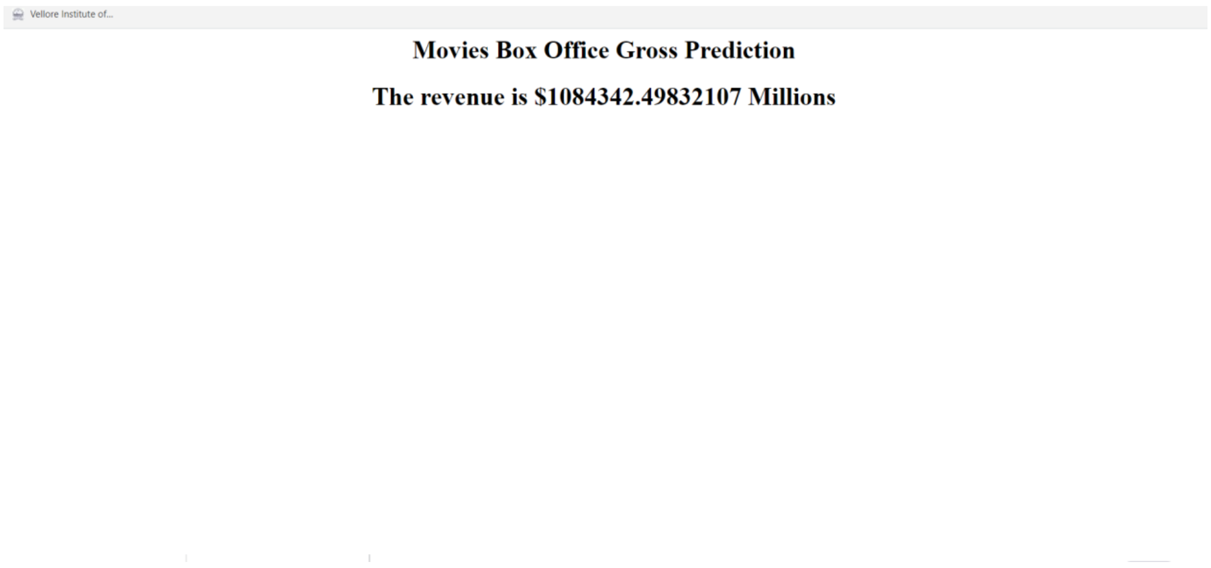
Vote Count

total votes

Release Month

month of release

Release Day_of_Week



Movies Box Office Gross Prediction

The revenue is $1084342.49832107 Millions

Advantages and Disadvantages

Advantages

Linear Regression is simple to implement and easier to interpret the output coefficients. We know the relationship between the independent and dependent variable have a linear relationship, this algorithm is the best to use because of it’s less complexity to compared to other algorithms. Linear Regression is susceptible to over-fitting but it can be avoided using some dimensionality reduction techniques, regularization (L1 and L2) techniques and cross-validation.

Disadvantages

Outliers can have huge effects on the regression and boundaries are linear in this technique. Linear regression assumes a linear relationship between dependent and independent variables. That means it assumes that there is a straight-line relationship between them. It assumes independence between attributes. Linear regression also looks at a relationship between the mean of the dependent variables and the independent variables. Just as the mean is not a complete description of a single variable, linear regression is not a complete description of relationships among variables.

Applications

This Gross prediction software can be used by Film Production Companies to predict their Gross.

Conclusion and Future Work

The IMDb dataset is an interesting dataset to analyze. After building the five models we found out that the Random Forest represents the movie features more accurately. The success percentage for all models are better in comparison to the previous studies. The results obtained are better than that of some standard libraries and similar studies. A movie success does not only depend on features related to movies. Number of audience plays very important role for a movie to become successful. Because the whole point is about audiences, the whole industry will make no sense if there is no audience to watch a movie. Number of ticket sold during a specific year can indicate the number of audiences of that year. In the future, we would like to increase both the number of movies and features in the dataset. We would also like to include other social media sources of movie data collection such as Twitter and YouTube. Other learning models that we want to apply to the movie data are the following supervised learning models: MLP and Bagging. We are interested in comparing results from these models with those expressed herein.

Bibliography

[1] "Global box office revenue 2016 | Statistic." [Online]. Available: https://www.statista.com/statistics/259987/global-box-officerevenue/. [Accessed: 03-Jun-2018].

[2] S. Gopinath, P. K. Chintagunta, and S. Venkataraman, "Blogs, Advertising, and Local-Market Movie Box Office Performance," Management Science, vol. 59, no. 12, pp. 2635–2654, 2013.

[3] M. C. A. Mestyán, T. Yasseri, and J. Kertész, "Early Prediction of Movie Box Office Success Based on Wikipedia Activity Big Data," PLoS ONE, vol. 8, no. 8, 2013.

[4] J. S. Simonoff and I. R. Sparrow, "Predicting Movie Grosses: Winners and Losers, Blockbusters and Sleepers," Chance, vol. 13, no. 3, pp. 15–24, 2000.

[5] A. Chen, "Forecasting gross revenues at the movie box office," Working paper, University of Washington, Seattle, WA, June, 2002

[6] M. S. Sawhney and J. Eliashberg, "A Parsimonious Model for Forecasting Gross Box-Office Revenues of Motion Pictures," Marketing Science, vol. 15, no. 2, pp. 113–131, 1996

Appendix

Source Code:

```
import numpy as np
#import the necessary libraries
import pandas as pd #data manipulation
import json #for reading json object
import pickle # For saving the model file
from ast import literal_eval#to evaluate the string as pyhton expression
credits=pd.read_csv(r"C:\Users\Jeya Samthosh
Kumar\Desktop\Data\tmdb_5000_credits.csv")
movies_df=pd.read_csv(r"C:\Users\Jeya Samthosh
Kumar\Desktop\Data\tmdb_5000_movies.csv")
credits_column_renamed=credits.rename(index=str,columns={"movie_id":"id"})
movies=movies_df.merge(credits_column_renamed,on="id")
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)
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 []
features = ['keywords', 'genres']
for feature in features:
    movies[feature] = movies[feature].apply(get_list)
movies['genres']  = movies['genres'] .str.join(', ')
movies = movies.dropna(subset = ['director','runtime'])
movies["revenue"]=movies["revenue"].floordiv(1000000)
movies["budget"]=movies["budget"].floordiv(1000000)
movies = movies[movies['budget'] != 0]
movies['release_date'] =
pd.DataFrame(pd.to_datetime(movies['release_date'],dayfirst=True))
movies['release_month'] = movies['release_date'].dt.month
```

```python
movies['release_DOW'] = movies['release_date'].dt.dayofweek
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'])
movies['has_homepage'] = 0
movies.loc[movies['homepage'].isnull() == False, 'has_homepage'] = 1
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','ha
s_homepage'],axis = 1)
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
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
x=movies_box.iloc[:,[0,1,2,4,5,6,8,9]]
x=pd.DataFrame(x,columns=['budget','genres','popularity','runtime','vote_avera
ge','vote_count','release_month','release_DOW'])
y=movies_box.iloc[:,3]
y=pd.DataFrame(y,columns=['revenue'])
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
x=sc.fit_transform(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()
```

For Flask:

```
from flask import Flask, render_template, request
import pickle
app = Flask(__name__)

model = pickle.load(open("model_movies.pkl","rb"))
scalar = pickle.load(open("scalar_movies.pkl","rb"))

@app.route('/')
def hello():
    return render_template("Demo2.html")

@app.route('/resultnew', methods = ['POST'])
def User():
    b = request.form["bg"]
    c = request.form["ge"]
    d = request.form["pr"]
    e = request.form["rt"]
    f = request.form["va"]
    g = request.form["vc"]
    i = request.form["rm"]
    j = request.form["rd"]
    t =
[[float(b),float(c),float(d),float(e),float(f),float(g),float(i),float(j)]]
    print(t)
    y = scalar.transform(t)
    pred = model.predict(y)
    return render_template("resultnew.html",out="The revenue is
$"+str(pred[0][0])+" million")

if __name__ == '__main__':
    app.run(debug = True)
```

For IBM Flask

```
from flask import Flask, render_template, request
import pickle
import requests

# NOTE: you must manually set API_KEY below using information retrieved from
your IBM Cloud account.
API_KEY = "hZvyweXuya3oowdRQWKJyWom-ajjelgKNi6d6RuZAoDW"
token_response = requests.post('https://iam.cloud.ibm.com/identity/token',
data={"apikey":
 API_KEY, "grant_type": 'urn:ibm:params:oauth:grant-type:apikey'})
mltoken = token_response.json()["access_token"]
```

```
header = {'Content-Type': 'application/json', 'Authorization': 'Bearer ' +
mltoken}
app = Flask(__name__)

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

@app.route('/')
def hello():
    return render_template("Demo2.html")

@app.route('/resultnew', methods = ['POST'])
def User():
    b = request.form["bg"]
    c = request.form["ge"]
    d = request.form["pr"]
    e = request.form["rt"]
    f = request.form["va"]
    g = request.form["vc"]
    i = request.form["rm"]
    j = request.form["rd"]
    t =
[[float(b),float(c),float(d),float(e),float(f),float(g),float(i),float(j)]]
    y = scalar.transform(t)
    payload_scoring = {"input_data": [{"fields":
['f0','f1','f2','f3','f4','f5','f6','f7'], "values": y}]}

    response_scoring = requests.post('https://us-
south.ml.cloud.ibm.com/ml/v4/deployments/ae093fd9-d762-4dd4-99f1-
da88a5605bb8/predictions?version=2022-06-06', json=payload_scoring,
     headers={'Authorization': 'Bearer ' + mltoken})
    print("Scoring response")

    pred=response_scoring.json()
    output=pred['predictions'][0]['values'][0][0][0]
    return render_template("resultnew.html",out="The revenue is
$"+str(output)+" million")

if __name__ == '__main__':
    app.run(debug = False)
```