RV Educational Institutions ®
RV College of Engineering ®

Autonomous
Institution Affiliated
to Visvesvaraya
Technological
University, Belagavi

Approved by AICTE,
New Delhi, Accredited
By NAAC, Bengaluru
And NBA, New Delhi

Go, change the world

DEPARTMENT OF MASTER OF COMPUTER APPLICATIONS

Machine Learning

20MCA251

Report on Movie success prediction rate

Submitted in partial fulfillment of the requirements for the award of degree of

MASTER OF COMPUTER APPLICATIONS

By

Shreyas H S
1RV20MC100

Under the Guidance Of
Dr. Andhe Dharani & Dr. Anupama Kumar
2020-2021

RV COLLEGE OF ENGINEERING®

(Autonomous Institution Affiliated to Visvesvaraya Technological University, Belagavi)

DEPARTMENT OF
MASTER OF COMPUTER APPLICATIONS

Bengaluru– 560059

CERTIFICATE

Certified that the Assignment title "*Movie Success Prediction Rate*"carried out by **Royson Lobo [1RV20MC087] and Shreyas H S [1RV20MC100]** bonafied students of **RV College of Engineering, Bengaluru** submitted in partial fulfillment for the award of **Master of Computer Applications** of **RV College of Engineering, Bengaluru affiliated to Visvesvaraya Technological University, Belagavi** during the year **2020-2021**. It is certified that all corrections/suggestions indicated for internal assessment have been incorporated in the report deposited in the departmental library. The report has been approved as it satisfies the partial academic requirement in respect of the course Machine Learning 20MCA251.

Faculty-In charge
Dr. Andhe Dharani,
Professor & Director,
Department of MCA
RVCE, Bengaluru-
560059

Table of Contents

Problem definition

***What?**

•Predict the success and failure of upcoming movie based on several attributes.

***How?**

•Classification model to find correlation between several attributes like movie rating, genre, movie duration etc.

•The system will predict success rating from the correlation between various movie criteria.

***Why?**

•This model can be used by movie watchers, producers.

•To help users to decide whether to book ticket in advance or not.

•To save money of producer.

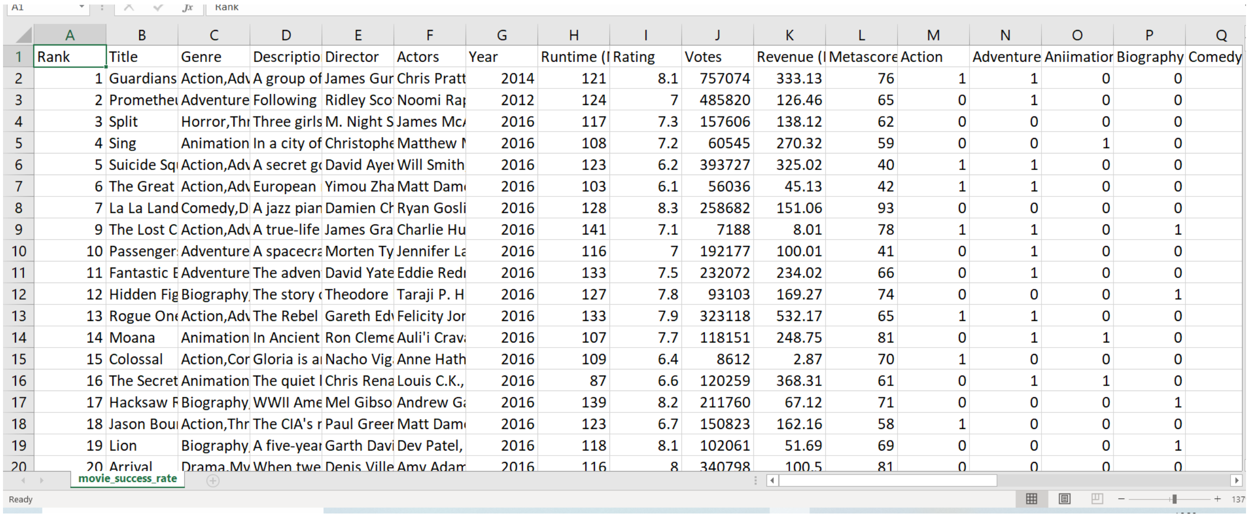
Data set and features considered

- We use movies data set to train and test our system.

- We have taken this data set from kaggle.com.

- The data set contains information about several movies such as movie title, genre, movie ratings, duration,no. of voters etc.

Sample dataset

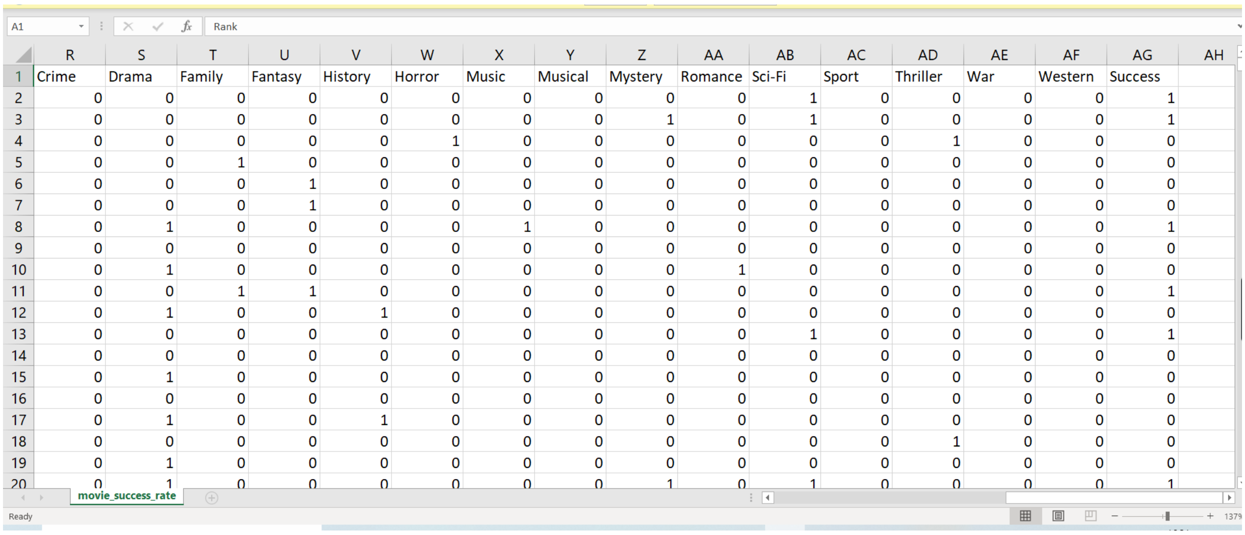


A1 fx Rank

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
1	Rank	Title	Genre	Descriptio	Director	Actors	Year	Runtime (I	Rating	Votes	Revenue (I	Metascore	Action	Adventure	Aniimatior	Biography	Comedy
2	1	Guardians	Action,Adv	A group of	James Gur	Chris Pratt	2014	121	8.1	757074	333.13	76	1	1	0	0	
3	2	Promethe	Adventure	Following	Ridley Sco	Noomi Rap	2012	124	7	485820	126.46	65	0	1	0	0	
4	3	Split	Horror,Thr	Three girls	M. Night S	James McA	2016	117	7.3	157606	138.12	62	0	0	0	0	
5	4	Sing	Animation	In a city of	Christophe	Matthew M	2016	108	7.2	60545	270.32	59	0	0	1	0	
6	5	Suicide Sq	Action,Adv	A secret go	David Ayer	Will Smith	2016	123	6.2	393727	325.02	40	1	1	0	0	
7	6	The Great	Action,Adv	European	Yimou Zha	Matt Dam	2016	103	6.1	56036	45.13	42	1	1	0	0	
8	7	La La Land	Comedy,D	A jazz pian	Damien Ch	Ryan Gosli	2016	128	8.3	258682	151.06	93	0	0	0	0	
9	9	The Lost C	Action,Adv	A true-life	James Gra	Charlie Hu	2016	141	7.1	7188	8.01	78	1	1	0	1	
10	10	Passenger	Adventure	A spacecra	Morten Ty	Jennifer La	2016	116	7	192177	100.01	41	0	1	0	0	
11	11	Fantastic B	Adventure	The adven	David Yate	Eddie Red	2016	133	7.5	232072	234.02	66	0	1	0	0	
12	12	Hidden Fig	Biography	The story o	Theodore	Taraji P. H	2016	127	7.8	93103	169.27	74	0	0	0	1	
13	13	Rogue One	Action,Adv	The Rebel	Gareth Edv	Felicity Jor	2016	133	7.9	323118	532.17	65	1	1	0	0	
14	14	Moana	Animation	In Ancient	Ron Cleme	Auli'i Crava	2016	107	7.7	118151	248.75	81	0	1	1	0	
15	15	Colossal	Action,Cor	Gloria is a	Nacho Vig	Anne Hath	2016	109	6.4	8612	2.87	70	1	0	0	0	
16	16	The Secret	Animation	The quiet l	Chris Rena	Louis C.K.,	2016	87	6.6	120259	368.31	61	0	1	1	0	
17	17	Hacksaw R	Biography	WWII Ame	Mel Gibso	Andrew Ga	2016	139	8.2	211760	67.12	71	0	0	0	1	
18	18	Jason Bou	Action,Thr	The CIA's r	Paul Greer	Matt Dam	2016	123	6.7	150823	162.16	58	1	0	0	0	
19	19	Lion	Biography	A five-year	Garth Davi	Dev Patel,	2016	118	8.1	102061	51.69	69	0	0	0	1	
20	20	Arrival	Drama,My	When twe	Denis Ville	Amy Adam	2016	116	8	340798	100.5	81	0	0	0	0	

movie_success_rate

Ready



A1 fx Rank

	R	S	T	U	V	W	X	Y	Z	AA	AB	AC	AD	AE	AF	AG	AH
1	Crime	Drama	Family	Fantasy	History	Horror	Music	Musical	Mystery	Romance	Sci-Fi	Sport	Thriller	War	Western	Success	
2	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	
3	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0	1	
4	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	
5	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	
6	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	
7	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	
8	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	1	
9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
10	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	
11	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	1	
12	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	
13	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	
14	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
15	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
17	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	
18	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	
19	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
20	0	1	0	0	0	0	0	0	1	0	1	0	0	0	0	1	

movie_success_rate

Ready

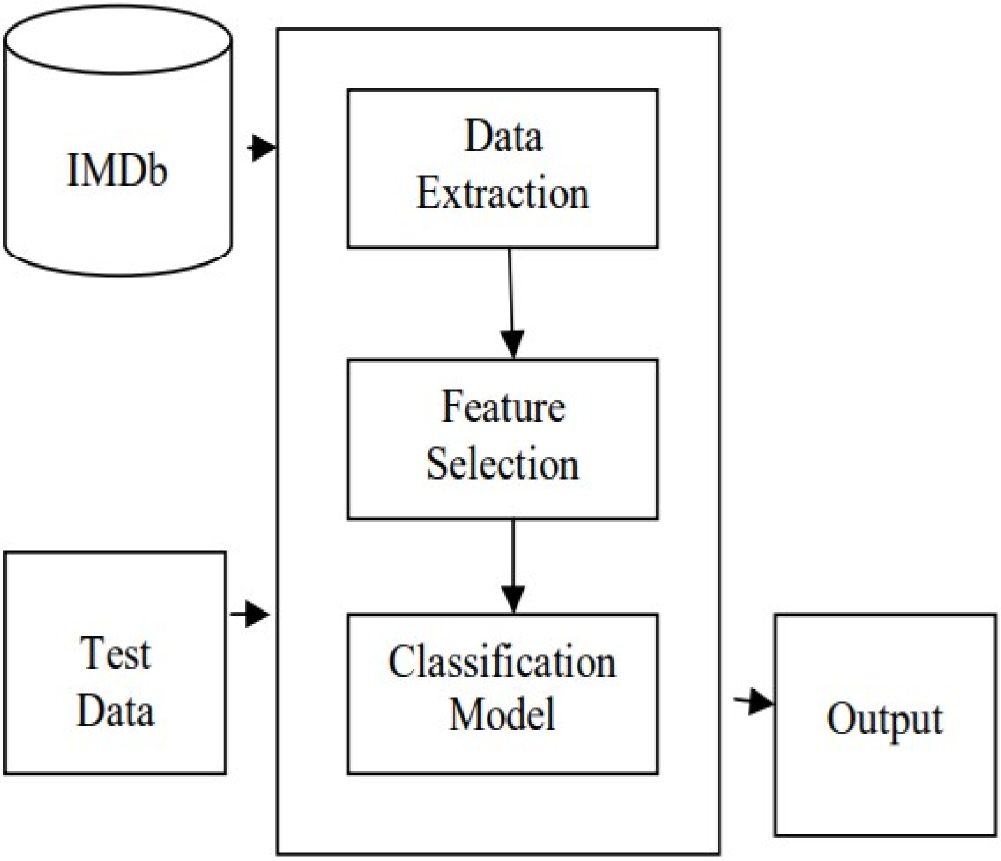
Process Model Tasks

1) Data Extraction

2) Data Pre-processing

3) Feature Extraction

4) Feature Selection.



K nearest neighbours (KNN Algorithm)

Steps followed:

1) Load the data

2) Initialize K to your chosen number of neighbors.

3) For each example in the data.

a) Calculate the distance between the query example and the current example from the data.

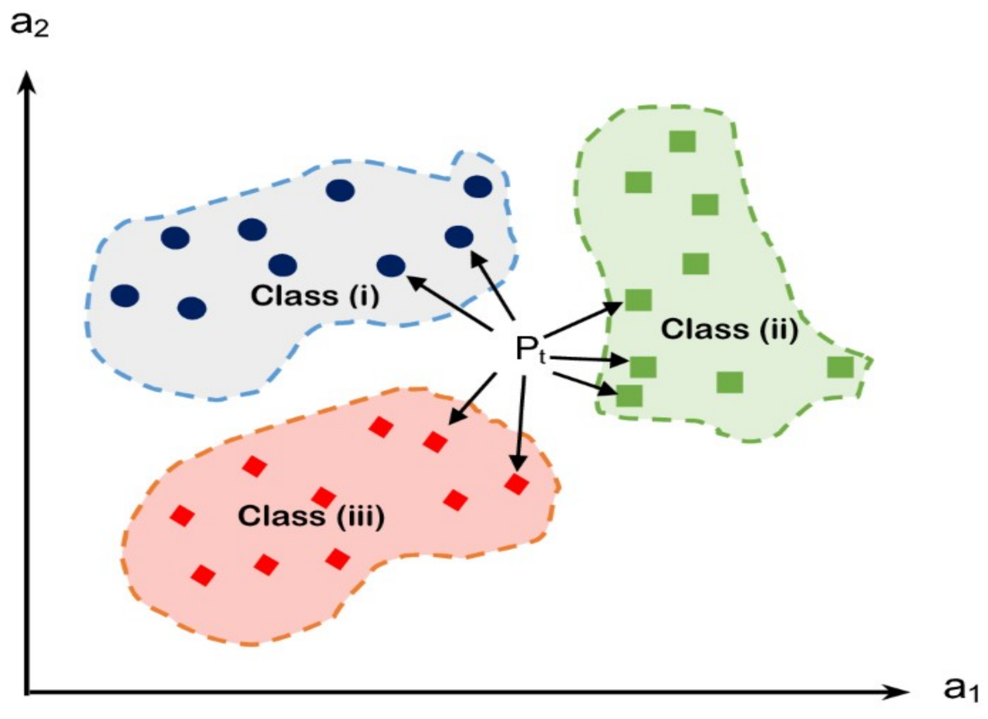
b) Add the distance and the index of the example to an ordered collection.

4) Sort the ordered collection of distances and indices from smallest to largest (in ascending order) by the distances.

5) Pick the first K entries from the sorted collection.

6) Get the labels of the selected K entries.

7) Return the mode of the K labels.



Random Forest

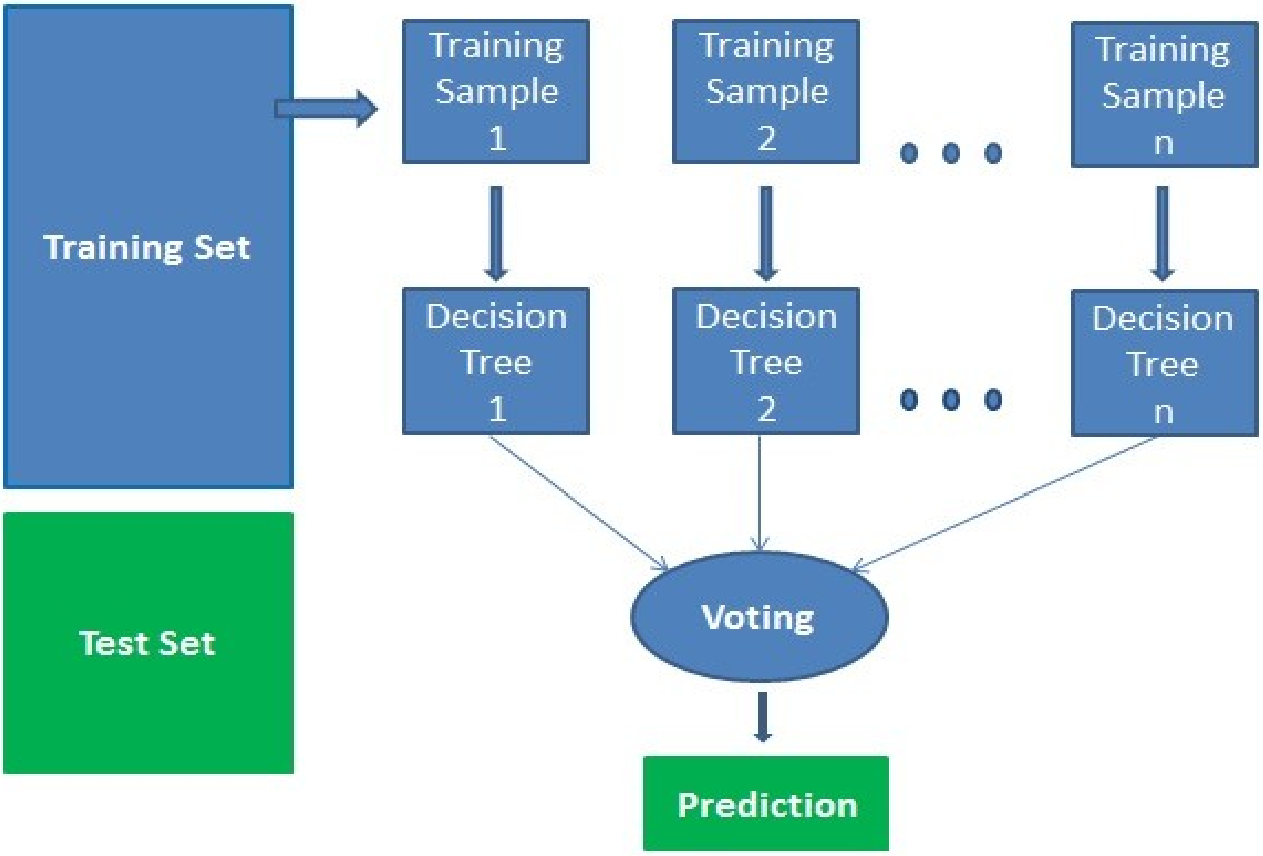
Steps followed:

1) Select random samples from a given dataset.

2) Construct a decision tree for every sample. Then it will get the prediction result from every decision tree.

3) Perform voting for every predicted result.

4) Select the most voted prediction result as the final prediction result.



Logistic Regression

- The sigmoid function is a mathematical function used to map the predicted values to probabilities.

- It maps any real value into another value within a range of 0 and 1.

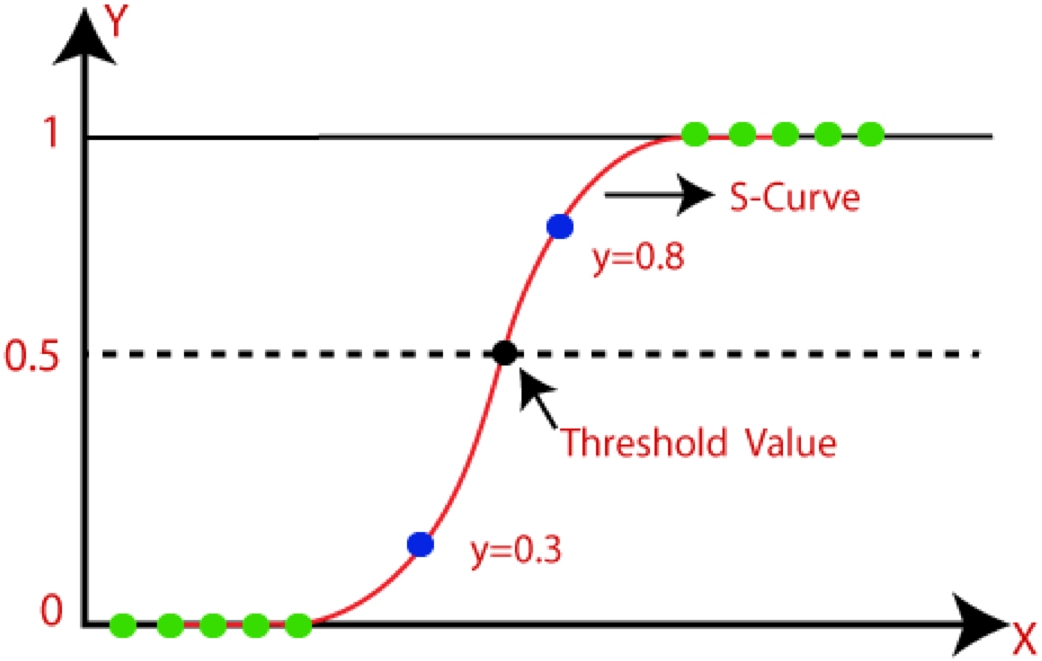
- The value of the logistic regression must be between 0 and 1, which cannot go beyond this limit, so it forms a curve like the "S" form. The S-form curve is called the Sigmoid function or the logistic function.

- In logistic regression, we use the concept of the threshold value, which defines the probability of either 0 or 1. Such as values above the threshold value tends to 1, and a value below the threshold values tends to 0.

Assumptions for Logistic Regression:

1.The dependent variable must be categorical in nature.

2.The independent variable should not have multi- collinearity.



The Logistic regression equation can be obtained from the Linear Regression equation. The mathematical steps to get Logistic Regression equations are given below:

We know the equation of the straight line can be written as:

$$y= b_0 + b_1x_1 + b_2x_2 + b_3x_3 + \cdots + b_nx_n$$

In Logistic Regression y can be between 0 and 1 only, so for this let's divide the above equation by (1-y):

$$\frac{y}{1-y}\text{; 0 for y= 0, and infinity for y=1}$$

But we need range between -[infinity] to +[infinity], then take logarithm of the equation it will become:

$$log\left[\frac{y}{1-y}\right] = b_0 + b_1x_1 + b_2x_2 + b_3x_3 + \cdots + b_nx_n$$

Steps in Logistic Regression: To implement the Logistic Regression using Python, we will use the same steps as we have done in previous topics of Regression. Below are the steps:

1.Data Pre-processing step

2.Fitting Logistic Regression to the Training set

3.Predicting the test result

4.Test accuracy of the result(Creation of Confusion matrix)

5.Visualizing the test set result.

Libraries and Functions used

Numpy

NumPy is the fundamental package for scientific computing in Python. It is a Python library that provides a multidimensional array object, various derived objects (such as masked arrays and matrices), and an assortment of routines for fast operations on arrays, including mathematical, logical, shape manipulation, sorting, selecting, I/O, discrete Fourier transforms, basic linear algebra, basic statistical operations, random simulation and much more.

Pandas

Pandas is a Python package providing fast, flexible, and expressive data structures designed to make working with “relational” or “labeled” data both easy and intuitive. It aims to be the fundamental high-level building block for doing practical, real-world data analysis in Python. Additionally, it has the broader goal of becoming the most powerful and flexible open source data analysis/manipulation tool available in any language.

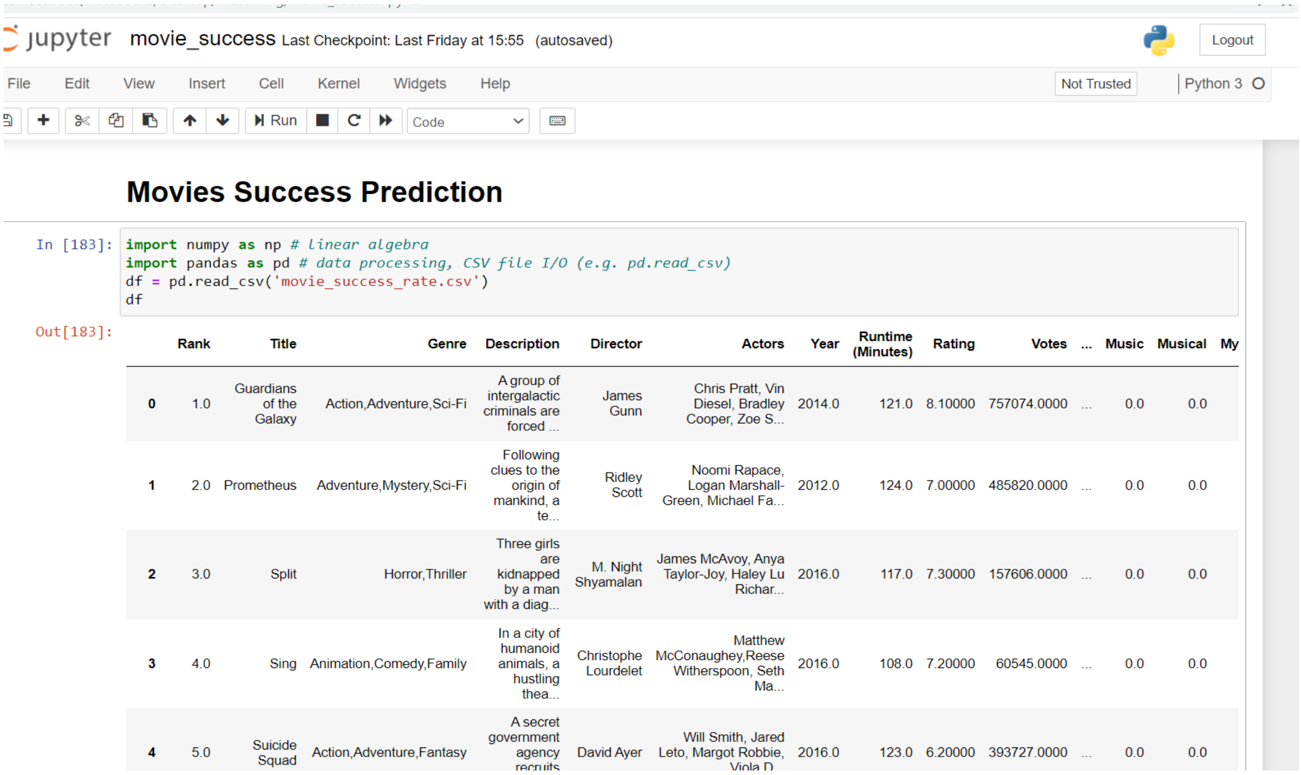
Seaborn

Seaborn is a library for making statistical graphics in Python. It builds on top of matplotlib and integrates closely with pandas data structures.
Seaborn helps you explore and understand your data. Its plotting functions operate on dataframes and arrays containing whole datasets and internally perform the necessary semantic mapping and statistical aggregation to produce informative plots. Its dataset-oriented, declarative API lets you focus on what the different elements of your plots mean, rather than on the details of how to draw them.

Scikit-learn

Scikit-learn is a library in Python that provides many unsupervised and supervised learning algorithms. It’s built upon some of the technology like NumPy, pandas, and Matplotlib.

Importing modules and displaying dataset



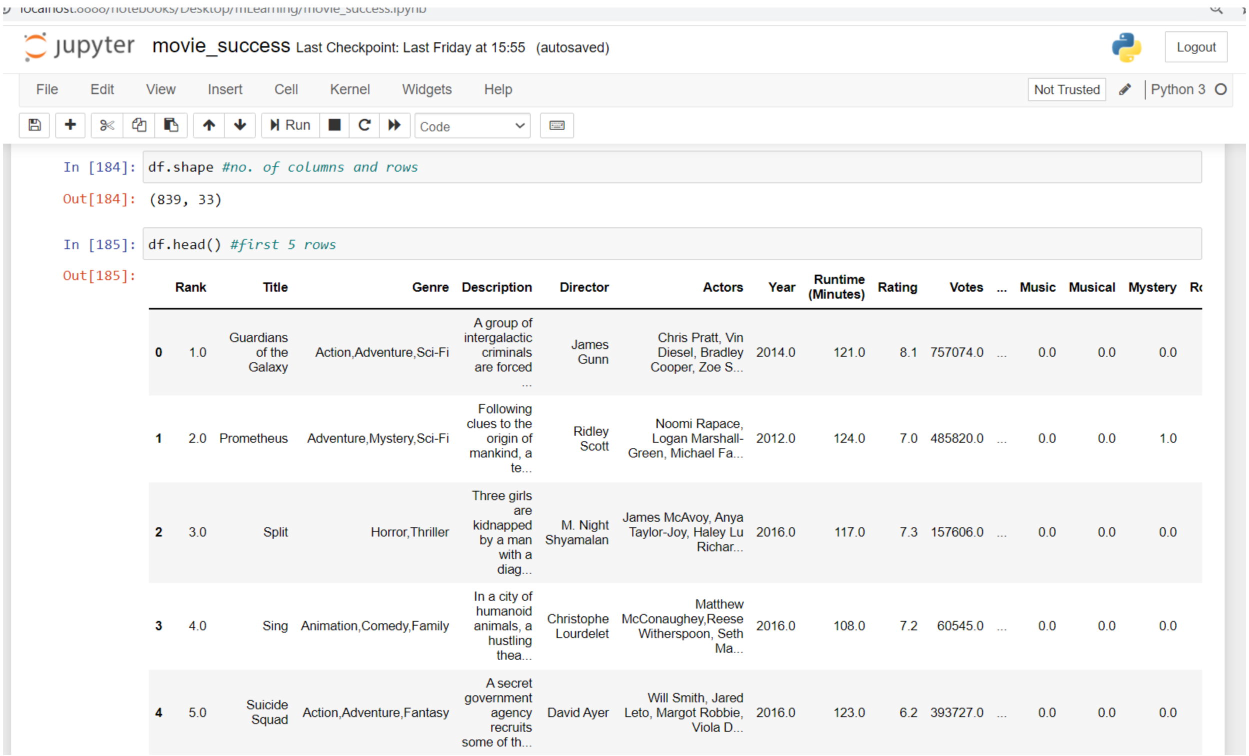
Movies Success Prediction

```
In [183]: import numpy as np # linear algebra
          import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
          df = pd.read_csv('movie_success_rate.csv')
          df
```

Out[183]:

	Rank	Title	Genre	Description	Director	Actors	Year	Runtime (Minutes)	Rating	Votes	...	Music	Musical	My
0	1.0	Guardians of the Galaxy	Action,Adventure,Sci-Fi	A group of intergalactic criminals are forced ...	James Gunn	Chris Pratt, Vin Diesel, Bradley Cooper, Zoe S...	2014.0	121.0	8.10000	757074.0000	...	0.0	0.0	
1	2.0	Prometheus	Adventure,Mystery,Sci-Fi	Following clues to the origin of mankind, a te...	Ridley Scott	Noomi Rapace, Logan Marshall-Green, Michael Fa...	2012.0	124.0	7.00000	485820.0000	...	0.0	0.0	
2	3.0	Split	Horror,Thriller	Three girls are kidnapped by a man with a diag...	M. Night Shyamalan	James McAvoy, Anya Taylor-Joy, Haley Lu Richar...	2016.0	117.0	7.30000	157606.0000	...	0.0	0.0	
3	4.0	Sing	Animation,Comedy,Family	In a city of humanoid animals, a hustling thea...	Christophe Lourdelet	Matthew McConaughey,Reese Witherspoon, Seth Ma...	2016.0	108.0	7.20000	60545.0000	...	0.0	0.0	
4	5.0	Suicide Squad	Action,Adventure,Fantasy	A secret government agency recruits	David Ayer	Will Smith, Jared Leto, Margot Robbie, Viola D	2016.0	123.0	6.20000	393727.0000	...	0.0	0.0	

No. of columns and rows

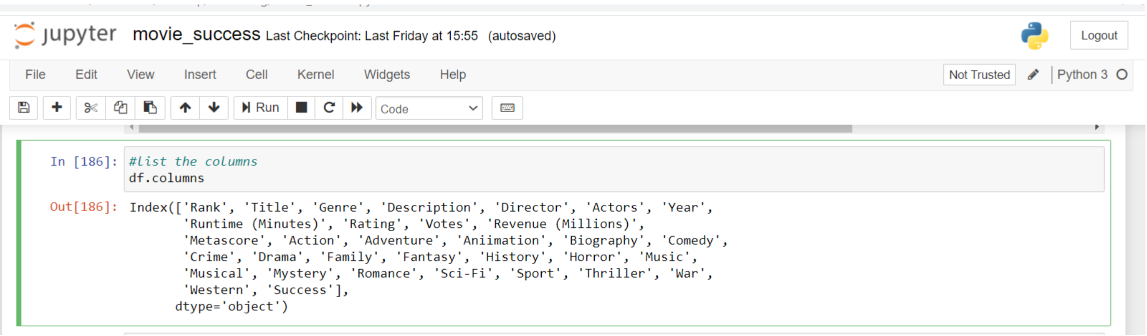

localhost:8888/notebooks/Desktop/mLearning/movie_success.ipynb
jupyter
movie_success Last Checkpoint: Last Friday at 15:55 (autosaved)
Logout
File
Edit
View
Insert
Cell
Kernel
Widgets
Help
Not Trusted
Python 3
Run
Code

```
In [184]: df.shape #no. of columns and rows

Out[184]: (839, 33)

In [185]: df.head() #first 5 rows
```

Out[185]:

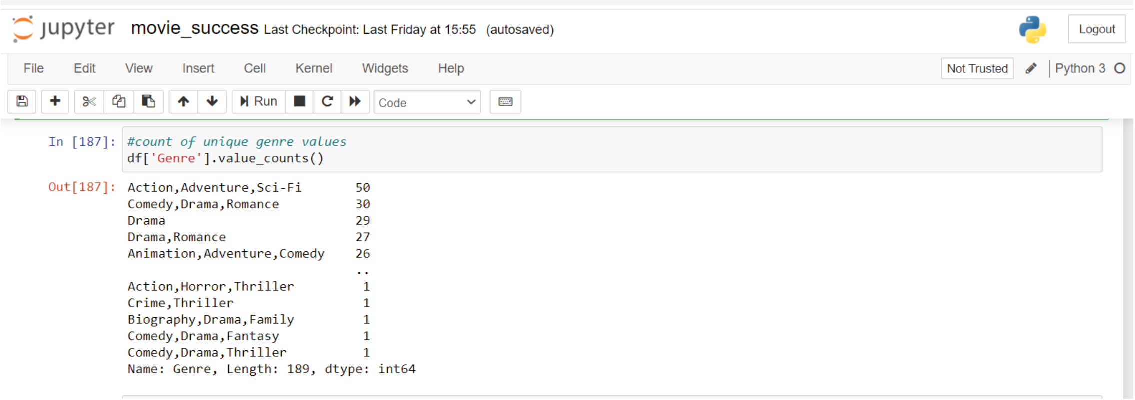
	Rank	Title	Genre	Description	Director	Actors	Year	Runtime (Minutes)	Rating	Votes	...	Music	Musical	Mystery	Ro
0	1.0	Guardians of the Galaxy	Action,Adventure,Sci-Fi	A group of intergalactic criminals are forced ...	James Gunn	Chris Pratt, Vin Diesel, Bradley Cooper, Zoe S...	2014.0	121.0	8.1	757074.0	...	0.0	0.0	0.0	
1	2.0	Prometheus	Adventure,Mystery,Sci-Fi	Following clues to the origin of mankind, a te...	Ridley Scott	Noomi Rapace, Logan Marshall-Green, Michael Fa...	2012.0	124.0	7.0	485820.0	...	0.0	0.0	1.0	
2	3.0	Split	Horror,Thriller	Three girls are kidnapped by a man with a diag...	M. Night Shyamalan	James McAvoy, Anya Taylor-Joy, Haley Lu Richar...	2016.0	117.0	7.3	157606.0	...	0.0	0.0	0.0	
3	4.0	Sing	Animation,Comedy,Family	In a city of humanoid animals, a hustling thea...	Christophe Lourdelet	Matthew McConaughey,Reese Witherspoon, Seth Ma...	2016.0	108.0	7.2	60545.0	...	0.0	0.0	0.0	
4	5.0	Suicide Squad	Action,Adventure,Fantasy	A secret government agency recruits some of th...	David Ayer	Will Smith, Jared Leto, Margot Robbie, Viola D...	2016.0	123.0	6.2	393727.0	...	0.0	0.0	0.0	

Listing the column names



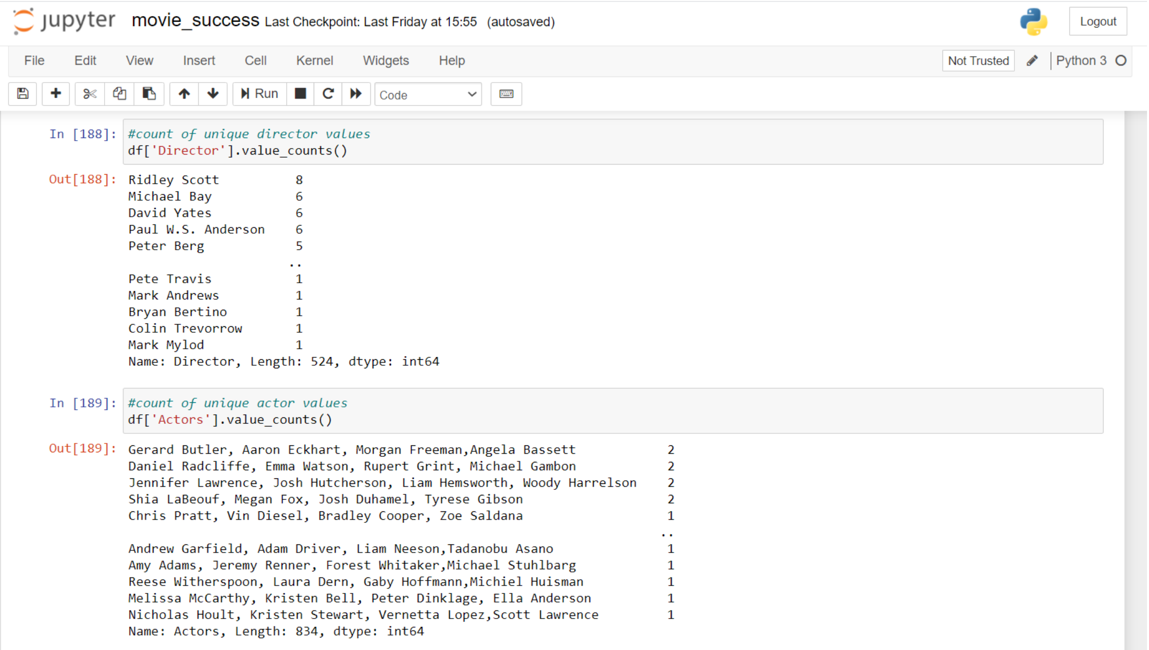
```
In [186]: #list the columns
          df.columns

Out[186]: Index(['Rank', 'Title', 'Genre', 'Description', 'Director', 'Actors', 'Year',
                 'Runtime (Minutes)', 'Rating', 'Votes', 'Revenue (Millions)',
                 'Metascore', 'Action', 'Adventure', 'Aniimation', 'Biography', 'Comedy',
                 'Crime', 'Drama', 'Family', 'Fantasy', 'History', 'Horror', 'Music',
                 'Musical', 'Mystery', 'Romance', 'Sci-Fi', 'Sport', 'Thriller', 'War',
                 'Western', 'Success'],
                dtype='object')
```

Counting unique genre values



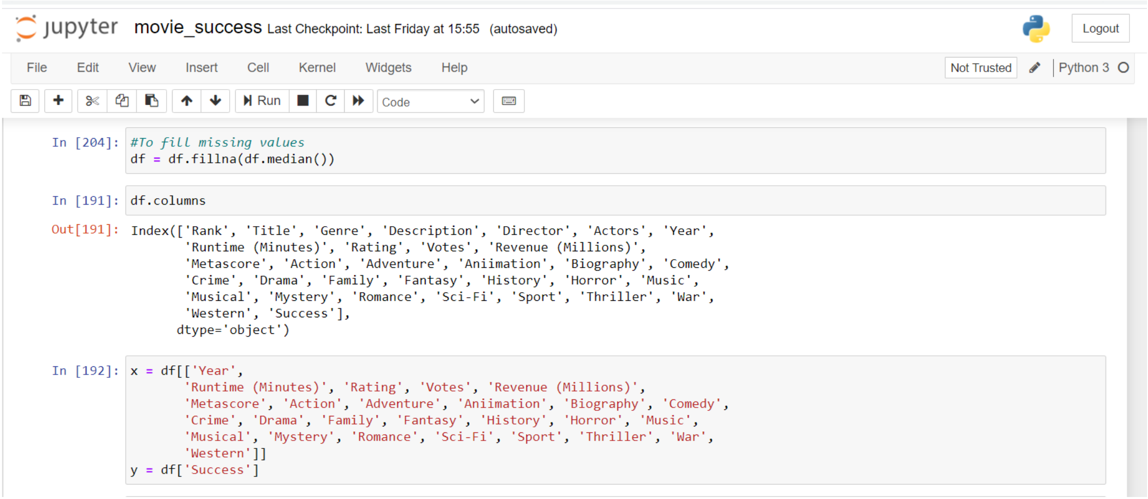
```
In [187]: #count of unique genre values
          df['Genre'].value_counts()

Out[187]: Action,Adventure,Sci-Fi       50
          Comedy,Drama,Romance          30
          Drama                         29
          Drama,Romance                 27
          Animation,Adventure,Comedy    26
                                        ..
          Action,Horror,Thriller         1
          Crime,Thriller                 1
          Biography,Drama,Family         1
          Comedy,Drama,Fantasy           1
          Comedy,Drama,Thriller          1
          Name: Genre, Length: 189, dtype: int64
```

Counting unique director and Actor values



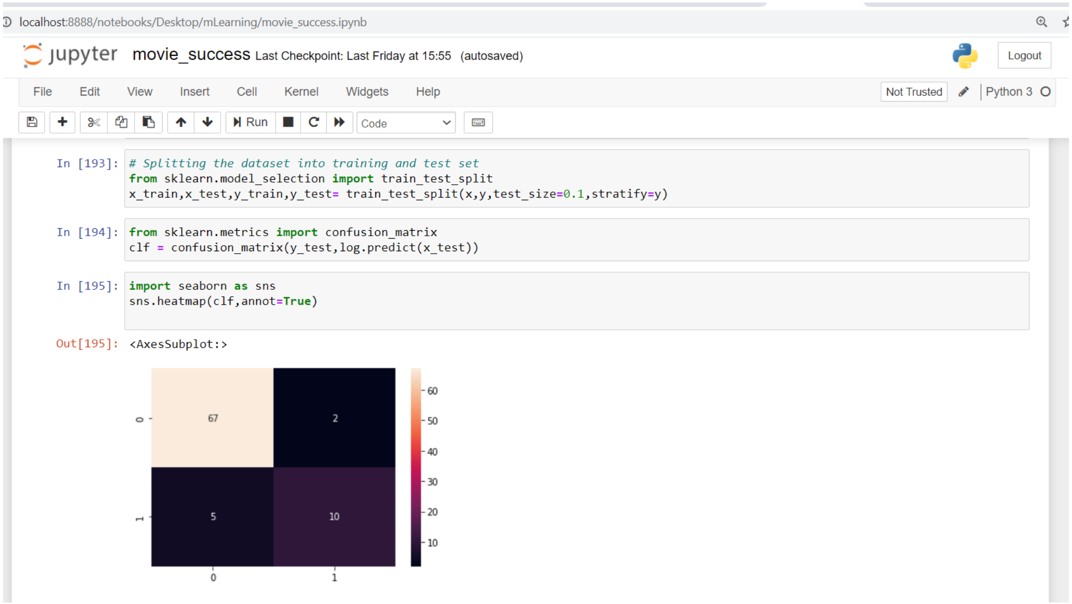
```
In [188]: #count of unique director values
          df['Director'].value_counts()

Out[188]: Ridley Scott          8
          Michael Bay           6
          David Yates           6
          Paul W.S. Anderson    6
          Peter Berg            5
                               ..
          Pete Travis           1
          Mark Andrews          1
          Bryan Bertino         1
          Colin Trevorrow       1
          Mark Mylod            1
          Name: Director, Length: 524, dtype: int64

In [189]: #count of unique actor values
          df['Actors'].value_counts()

Out[189]: Gerard Butler, Aaron Eckhart, Morgan Freeman,Angela Bassett          2
          Daniel Radcliffe, Emma Watson, Rupert Grint, Michael Gambon         2
          Jennifer Lawrence, Josh Hutcherson, Liam Hemsworth, Woody Harrelson  2
          Shia LaBeouf, Megan Fox, Josh Duhamel, Tyrese Gibson                2
          Chris Pratt, Vin Diesel, Bradley Cooper, Zoe Saldana                1
                                                                              ..
          Andrew Garfield, Adam Driver, Liam Neeson,Tadanobu Asano            1
          Amy Adams, Jeremy Renner, Forest Whitaker,Michael Stuhlbarg         1
          Reese Witherspoon, Laura Dern, Gaby Hoffmann,Michiel Huisman        1
          Melissa McCarthy, Kristen Bell, Peter Dinklage, Ella Anderson       1
          Nicholas Hoult, Kristen Stewart, Vernetta Lopez,Scott Lawrence      1
          Name: Actors, Length: 834, dtype: int64
```

Filling missing values in the dataset



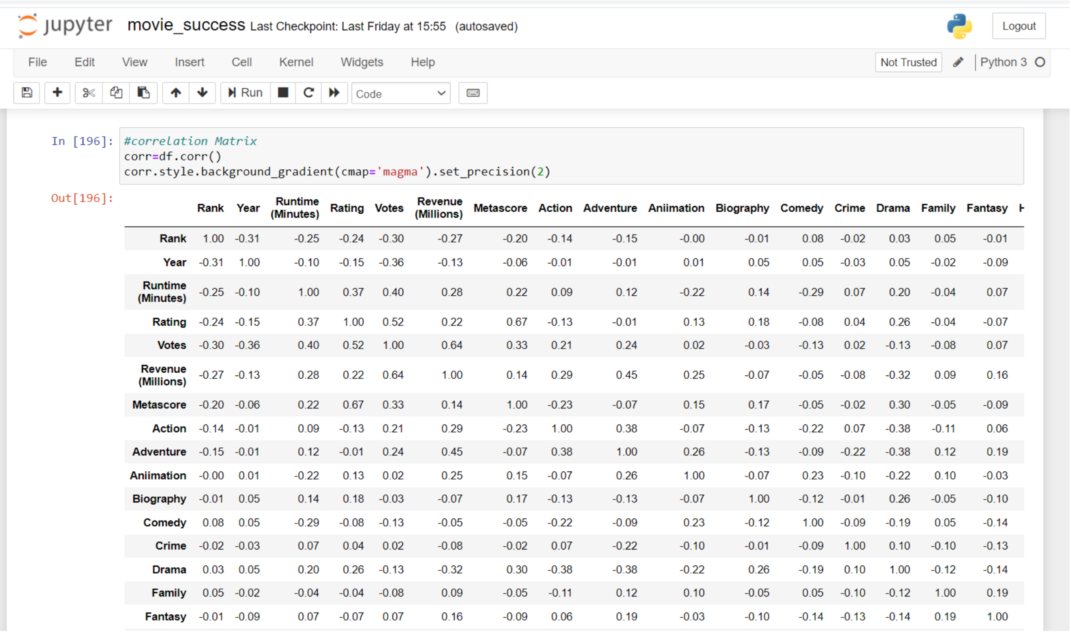
```
In [204]: #To fill missing values
          df = df.fillna(df.median())

In [191]: df.columns

Out[191]: Index(['Rank', 'Title', 'Genre', 'Description', 'Director', 'Actors', 'Year',
                 'Runtime (Minutes)', 'Rating', 'Votes', 'Revenue (Millions)',
                 'Metascore', 'Action', 'Adventure', 'Aniimation', 'Biography', 'Comedy',
                 'Crime', 'Drama', 'Family', 'Fantasy', 'History', 'Horror', 'Music',
                 'Musical', 'Mystery', 'Romance', 'Sci-Fi', 'Sport', 'Thriller', 'War',
                 'Western', 'Success'],
                dtype='object')

In [192]: x = df[['Year',
                 'Runtime (Minutes)', 'Rating', 'Votes', 'Revenue (Millions)',
                 'Metascore', 'Action', 'Adventure', 'Aniimation', 'Biography', 'Comedy',
                 'Crime', 'Drama', 'Family', 'Fantasy', 'History', 'Horror', 'Music',
                 'Musical', 'Mystery', 'Romance', 'Sci-Fi', 'Sport', 'Thriller', 'War',
                 'Western']]
          y = df['Success']
```

Splitting dataset into train and test dataset and drawing the confusion matrix for the test data


localhost:8888/notebooks/Desktop/mLearning/movie_success.ipynb
jupyter
movie_success Last Checkpoint: Last Friday at 15:55 (autosaved)
Logout
File
Edit
View
Insert
Cell
Kernel
Widgets
Help
Not Trusted
Python 3
Run
Code
67
2
5
10
0
1
0
1
60
50
40
30
20
10

```
In [193]: # Splitting the dataset into training and test set
          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,stratify=y)

In [194]: from sklearn.metrics import confusion_matrix
          clf = confusion_matrix(y_test,log.predict(x_test))

In [195]: import seaborn as sns
          sns.heatmap(clf,annot=True)

Out[195]: <AxesSubplot:>
```

Drawing the correlation matrix

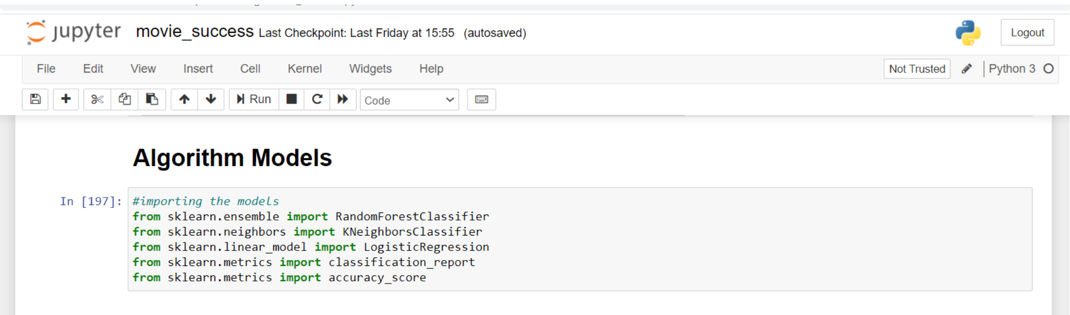

jupyter movie_success Last Checkpoint: Last Friday at 15:55 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3
Run
Code

In [196]:
```
#correlation Matrix
corr=df.corr()
corr.style.background_gradient(cmap='magma').set_precision(2)
```

Out[196]:

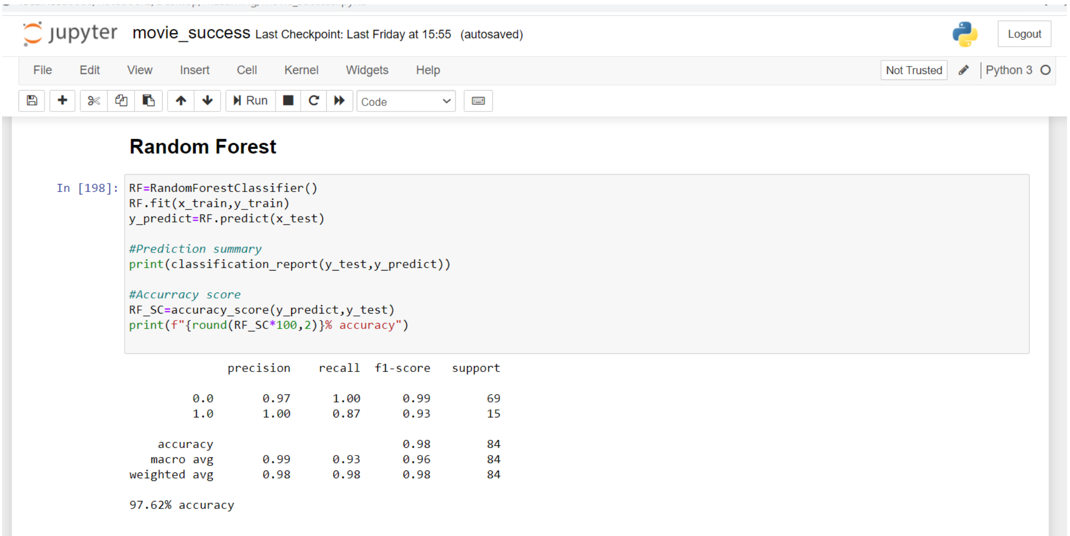
	Rank	Year	Runtime (Minutes)	Rating	Votes	Revenue (Millions)	Metascore	Action	Adventure	Aniimation	Biography	Comedy	Crime	Drama	Family	Fantasy	H
Rank	1.00	-0.31	-0.25	-0.24	-0.30	-0.27	-0.20	-0.14	-0.15	-0.00	-0.01	0.08	-0.02	0.03	0.05	-0.01	
Year	-0.31	1.00	-0.10	-0.15	-0.36	-0.13	-0.06	-0.01	-0.01	0.01	0.05	0.05	-0.03	0.05	-0.02	-0.09	
Runtime (Minutes)	-0.25	-0.10	1.00	0.37	0.40	0.28	0.22	0.09	0.12	-0.22	0.14	-0.29	0.07	0.20	-0.04	0.07	
Rating	-0.24	-0.15	0.37	1.00	0.52	0.22	0.67	-0.13	-0.01	0.13	0.18	-0.08	0.04	0.26	-0.04	-0.07	
Votes	-0.30	-0.36	0.40	0.52	1.00	0.64	0.33	0.21	0.24	0.02	-0.03	-0.13	0.02	-0.13	-0.08	0.07	
Revenue (Millions)	-0.27	-0.13	0.28	0.22	0.64	1.00	0.14	0.29	0.45	0.25	-0.07	-0.05	-0.08	-0.32	0.09	0.16	
Metascore	-0.20	-0.06	0.22	0.67	0.33	0.14	1.00	-0.23	-0.07	0.15	0.17	-0.05	-0.02	0.30	-0.05	-0.09	
Action	-0.14	-0.01	0.09	-0.13	0.21	0.29	-0.23	1.00	0.38	-0.07	-0.13	-0.22	0.07	-0.38	-0.11	0.06	
Adventure	-0.15	-0.01	0.12	-0.01	0.24	0.45	-0.07	0.38	1.00	0.26	-0.13	-0.09	-0.22	-0.38	0.12	0.19	
Aniimation	-0.00	0.01	-0.22	0.13	0.02	0.25	0.15	-0.07	0.26	1.00	-0.07	0.23	-0.10	-0.22	0.10	-0.03	
Biography	-0.01	0.05	0.14	0.18	-0.03	-0.07	0.17	-0.13	-0.13	-0.07	1.00	-0.12	-0.01	0.26	-0.05	-0.10	
Comedy	0.08	0.05	-0.29	-0.08	-0.13	-0.05	-0.05	-0.22	-0.09	0.23	-0.12	1.00	-0.09	-0.19	0.05	-0.14	
Crime	-0.02	-0.03	0.07	0.04	0.02	-0.08	-0.02	0.07	-0.22	-0.10	-0.01	-0.09	1.00	0.10	-0.10	-0.13	
Drama	0.03	0.05	0.20	0.26	-0.13	-0.32	0.30	-0.38	-0.38	-0.22	0.26	-0.19	0.10	1.00	-0.12	-0.14	
Family	0.05	-0.02	-0.04	-0.04	-0.08	0.09	-0.05	-0.11	0.12	0.10	-0.05	0.05	-0.10	-0.12	1.00	0.19	
Fantasy	-0.01	-0.09	0.07	-0.07	0.07	0.16	-0.09	0.06	0.19	-0.03	-0.10	-0.14	-0.13	-0.14	0.19	1.00	

Algorithm models


jupyter movie_success Last Checkpoint: Last Friday at 15:55 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3
Run
Code

Algorithm Models

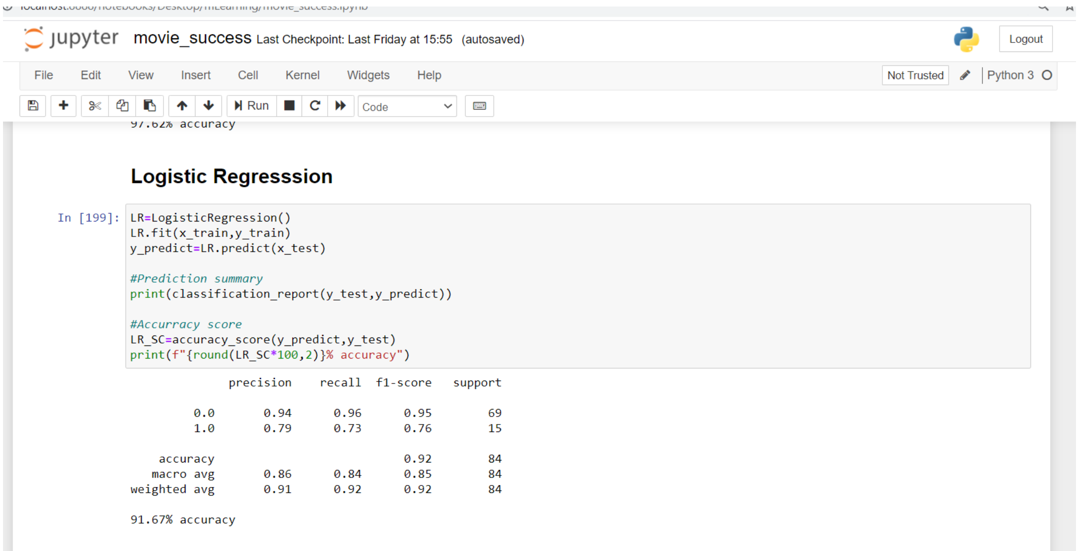
In [197]:
```
#importing the models
from sklearn.ensemble import RandomForestClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report
from sklearn.metrics import accuracy_score
```


jupyter movie_success Last Checkpoint: Last Friday at 15:55 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3
Run
Code

Random Forest

In [198]:
```
RF=RandomForestClassifier()
RF.fit(x_train,y_train)
y_predict=RF.predict(x_test)

#Prediction summary
print(classification_report(y_test,y_predict))

#Accurracy score
RF_SC=accuracy_score(y_predict,y_test)
print(f"{round(RF_SC*100,2)}% accuracy")
```

```
              precision    recall  f1-score   support

         0.0       0.97      1.00      0.99        69
         1.0       1.00      0.87      0.93        15

    accuracy                           0.98        84
   macro avg       0.99      0.93      0.96        84
weighted avg       0.98      0.98      0.98        84

97.62% accuracy
```

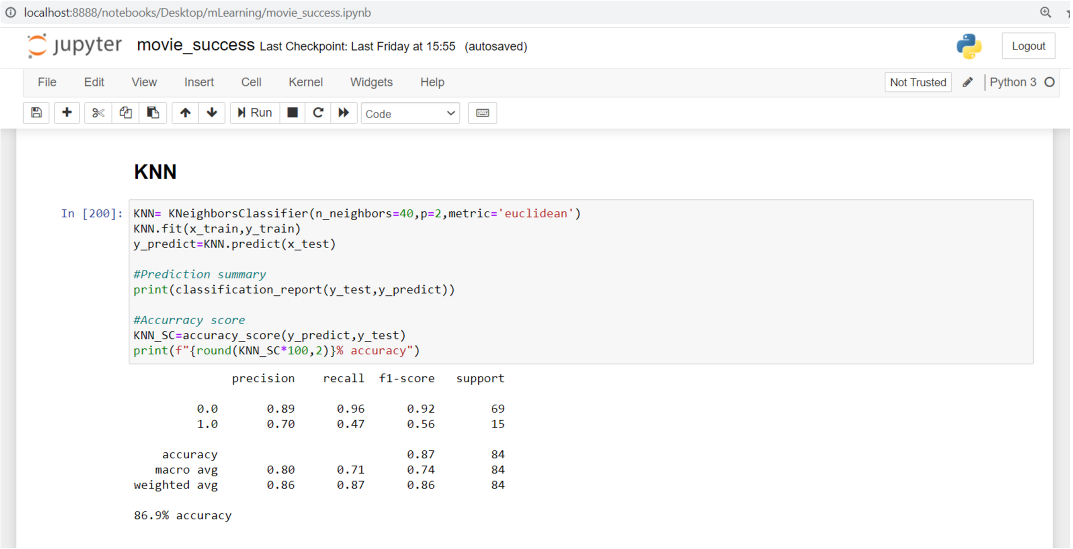


```
97.62% accuracy
```

Logistic Regresssion

```
In [199]: LR=LogisticRegression()
          LR.fit(x_train,y_train)
          y_predict=LR.predict(x_test)

          #Prediction summary
          print(classification_report(y_test,y_predict))

          #Accurracy score
          LR_SC=accuracy_score(y_predict,y_test)
          print(f"{round(LR_SC*100,2)}% accuracy")
```

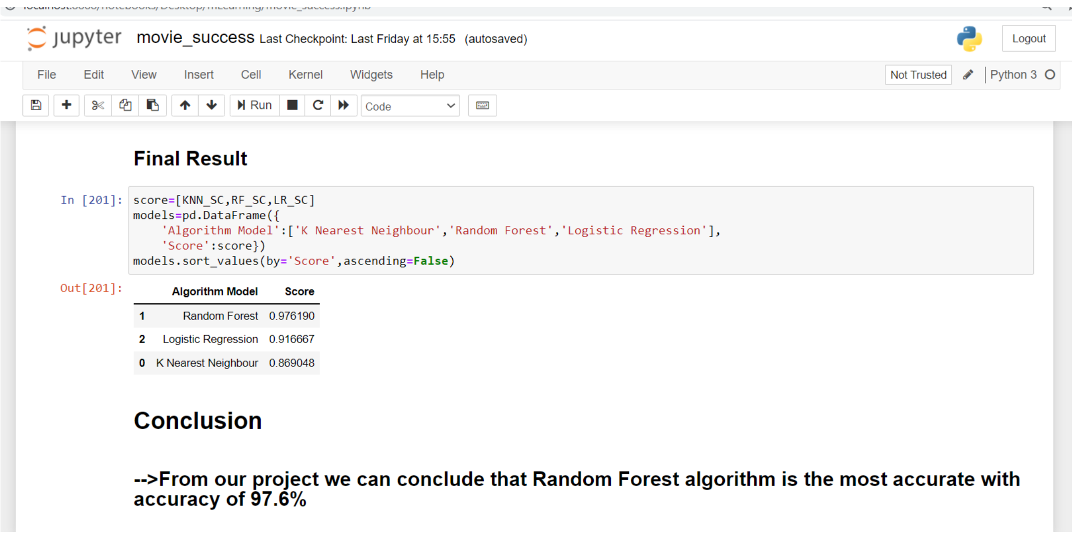
```
              precision    recall  f1-score   support

         0.0       0.94      0.96      0.95        69
         1.0       0.79      0.73      0.76        15

    accuracy                           0.92        84
   macro avg       0.86      0.84      0.85        84
weighted avg       0.91      0.92      0.92        84

91.67% accuracy
```


localhost:8888/notebooks/Desktop/mLearning/movie_success.ipynb
jupyter
movie_success Last Checkpoint: Last Friday at 15:55 (autosaved)
Logout
File
Edit
View
Insert
Cell
Kernel
Widgets
Help
Not Trusted
Python 3
Run
Code

KNN

```
In [200]: KNN= KNeighborsClassifier(n_neighbors=40,p=2,metric='euclidean')
          KNN.fit(x_train,y_train)
          y_predict=KNN.predict(x_test)

          #Prediction summary
          print(classification_report(y_test,y_predict))

          #Accurracy score
          KNN_SC=accuracy_score(y_predict,y_test)
          print(f"{round(KNN_SC*100,2)}% accuracy")
```

```
              precision    recall  f1-score   support

         0.0       0.89      0.96      0.92        69
         1.0       0.70      0.47      0.56        15

    accuracy                           0.87        84
   macro avg       0.80      0.71      0.74        84
weighted avg       0.86      0.87      0.86        84

86.9% accuracy
```



Final Result

```
In [201]: score=[KNN_SC,RF_SC,LR_SC]
          models=pd.DataFrame({
              'Algorithm Model':['K Nearest Neighbour','Random Forest','Logistic Regression'],
              'Score':score})
          models.sort_values(by='Score',ascending=False)
```

Out[201]:

	Algorithm Model	Score
1	Random Forest	0.976190
2	Logistic Regression	0.916667
0	K Nearest Neighbour	0.869048

Conclusion

-->From our project we can conclude that Random Forest algorithm is the most accurate with accuracy of 97.6%

References:

- **Data set collection:**

https://www.kaggle.com/therealsampat/predict-movie-success-rate

Websites:

- https://medium.com/analytics-vidhya/how-to-use-machine-learning-approach-to-predict-movie-box-office-revenue-success-e2e688669972
- https://www.javatpoint.com/machine-learning

Video Tutorial:

- https://youtu.be/7eh4d6sabA0