DIC Project Report

CSE 587B: Data Intensive Computing | Instructor: Dr. Shamsad Parvin

Name	**UBIT**	**Person Number**
Vivian Vincent Dmello	vivianvi	50540941
Chaitanya Deepak Yeole	cyeole	50535530
Vangala Eshwar Sai Prithvi Kiran	eshwarsa	50469012

Dataset: https://www.kaggle.com/datasets/victorsoeiro/netflix-tv-shows-and-movies

PHASE 1

1. Problem Statement:

Given a streaming platform such as Netflix, we want to identify which kinds of films go on to have great ratings. Based on things such as Run time, Genre, Actors, IMDB score and number of Votes, we want to see what we can use to predict movies with a good score.

Questions we are trying to answer:

- What determines this score? The Genre, Actors, Number of people who Voted/Watched it, the Run Time, etc?
- Which of these has the most value when it comes to determining this rating?
- What kind of impact does having different combinations of Genres have?

A. Why is this a significant problem?

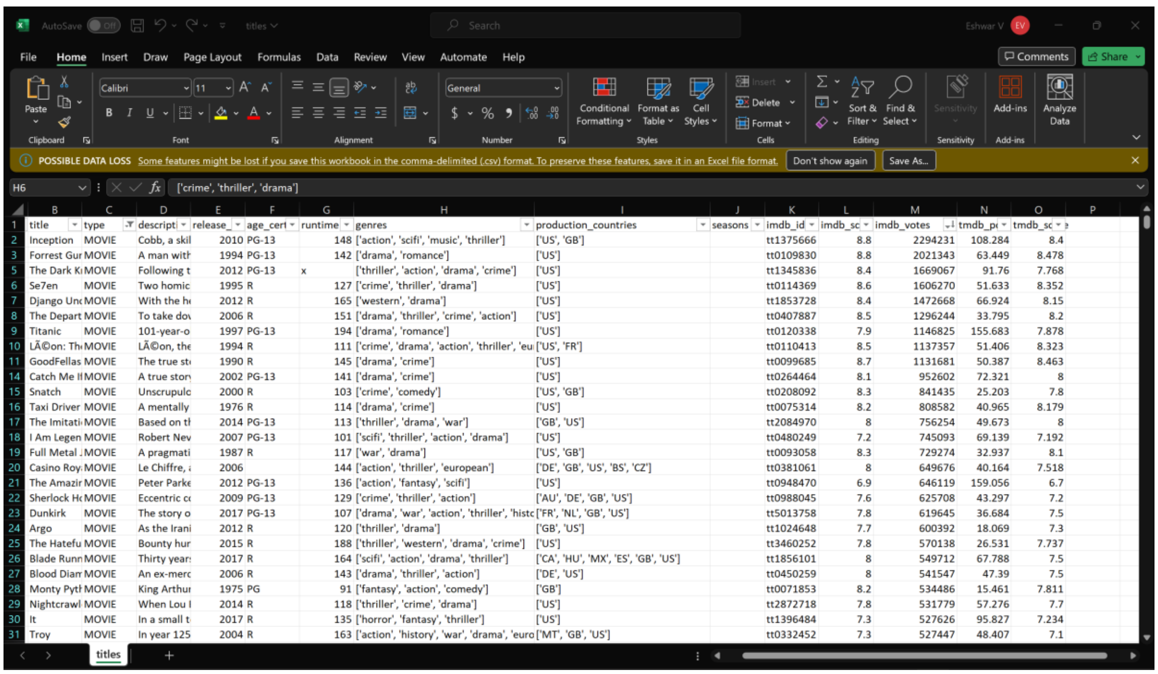
There is an ocean of content out in the world when it comes to watchable material. Movies and TV Shows are just a few among the many ways people consume media. These attention capturing moving pictures aren't just there to entertain us but also teach us many things. However, the real world is one in which only the popular Shows and Movies come out on top. We cannot have fantastic media to watch without investors paying the bills and investors are drawn to profit. Also, the base desire of any consumer is to have not just one amazing show to consume, but to have multiple such shows readily available. Just from these points we can see that there is an intense need to be able to filter out the best possible stories and provide those to consumers as they are the most likely to be watched. After all many people pay attention to a Movies 'score' or 'rating' before they try to watch it!

B. What is the potential of your project to contribute to the problem domain and why is it crucial?

Our project can help contribute to the problem of ‘selection’ and ‘prediction’. Our idea is to be able to use the readily available data to be able to choose from among sample movies the ones which would be the best rated and therefore most valued. Being able to choose this would involve looking at statistics of what makes a Movie good from the dataset we have chosen and using ML to be able to train a model to be able to pick out similar such shows. This would give a clearer idea on which films can be chosen and pushed to consumers.

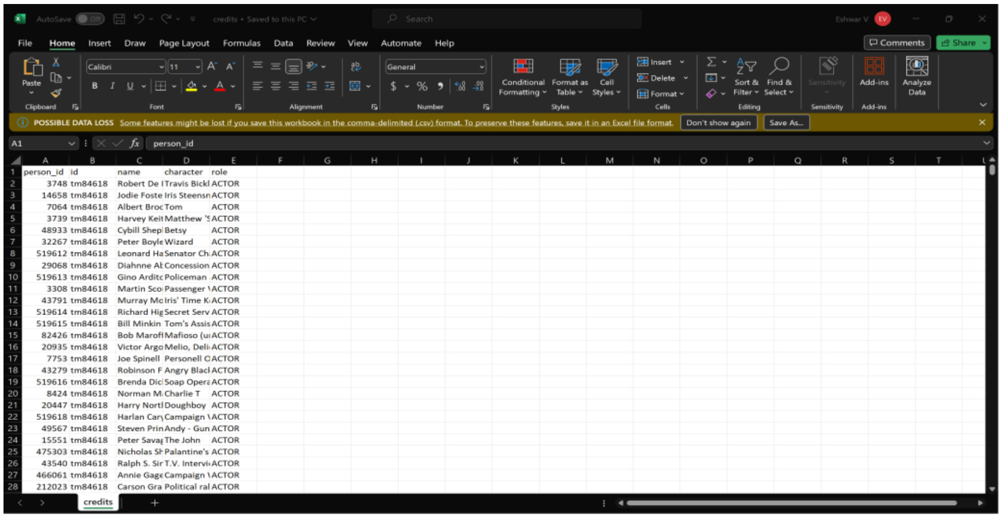
2. Data Sources:

Our dataset is taken from Kaggle. This dataset is one which originally has been split into 2 different datasets – Titles and Credits Datasets. Titles Dataset has around 5k+ records with 15 features.



	title	type	descripti	release_	age_cert	runtime	genres	production_countries	seasons	imdb_id	imdb_sc	imdb_votes	tmdb_p	tmdb_sc
2	Inception	MOVIE	Cobb, a skil	2010	PG-13	148	['action', 'scifi', 'music', 'thriller']	['US', 'GB']		tt1375666	8.8	2294231	108.284	8.4
3	Forrest Gur	MOVIE	A man with	1994	PG-13	142	['drama', 'romance']	['US']		tt0109830	8.8	2021343	63.449	8.478
5	The Dark K	MOVIE	Following t	2012	PG-13	x	['thriller', 'action', 'drama', 'crime']	['US']		tt1345836	8.4	1669067	91.76	7.768
6	Se7en	MOVIE	Two homic	1995	R	127	['crime', 'thriller', 'drama']	['US']		tt0114369	8.6	1606270	51.633	8.352
7	Django Unc	MOVIE	With the h	2012	R	165	['western', 'drama']	['US']		tt1853728	8.4	1472668	66.924	8.15
8	The Depart	MOVIE	To take do	2006	R	151	['drama', 'thriller', 'crime', 'action']	['US']		tt0407887	8.5	1296244	33.795	8.2
9	Titanic	MOVIE	101-year-o	1997	PG-13	194	['drama', 'romance']	['US']		tt0120338	7.9	1146825	155.683	7.878
10	LÃ©on: Th	MOVIE	LÃ©on, the	1994	R	111	['crime', 'drama', 'action', 'thriller', 'eu	['US', 'FR']		tt0110413	8.5	1137357	51.406	8.323
11	GoodFellas	MOVIE	The true st	1990	R	145	['drama', 'crime']	['US']		tt0099685	8.7	1131681	50.387	8.463
14	Catch Me If	MOVIE	A true story	2002	PG-13	141	['drama', 'crime']	['US']		tt0264464	8.1	952602	72.321	8
15	Snatch	MOVIE	Unscrupulo	2000	R	103	['crime', 'comedy']	['US', 'GB']		tt0208092	8.3	841435	25.203	7.8
16	Taxi Driver	MOVIE	A mentally	1976	R	114	['drama', 'crime']	['US']		tt0075314	8.2	808582	40.965	8.179
17	The Imitati	MOVIE	Based on th	2014	PG-13	113	['thriller', 'drama', 'war']	['GB', 'US']		tt2084970	8	756254	49.673	8
18	I Am Legen	MOVIE	Robert Nev	2007	PG-13	101	['scifi', 'thriller', 'action', 'drama']	['US']		tt0480249	7.2	745093	69.139	7.192
19	Full Metal J	MOVIE	A pragmati	1987	R	117	['war', 'drama']	['US', 'GB']		tt0093058	8.3	729274	32.937	8.1
20	Casino Roy	MOVIE	Le Chiffre, a	2006		144	['action', 'thriller', 'european']	['DE', 'GB', 'US', 'BS', 'CZ']		tt0381061	8	649676	40.164	7.518
21	The Amazir	MOVIE	Peter Parke	2012	PG-13	136	['action', 'fantasy', 'scifi']	['US']		tt0948470	6.9	646119	159.056	6.7
22	Sherlock Hc	MOVIE	Eccentric cc	2009	PG-13	129	['crime', 'thriller', 'action']	['AU', 'DE', 'GB', 'US']		tt0988045	7.6	625708	43.297	7.2
23	Dunkirk	MOVIE	The story o	2017	PG-13	107	['drama', 'war', 'action', 'thriller', 'histc	['FR', 'NL', 'GB', 'US']		tt5013758	7.8	619645	36.684	7.5
24	Argo	MOVIE	As the Irani	2012	R	120	['thriller', 'drama']	['GB', 'US']		tt1024648	7.7	600392	18.069	7.3
25	The Hatefu	MOVIE	Bounty hur	2015	R	188	['thriller', 'western', 'drama', 'crime']	['US']		tt3460252	7.8	570138	26.531	7.737
26	Blade Runn	MOVIE	Thirty years	2017	R	164	['scifi', 'action', 'drama', 'thriller']	['CA', 'HU', 'MX', 'ES', 'GB', 'US']		tt1856101	8	549712	67.788	7.5
27	Blood Diam	MOVIE	An ex-merc	2006	R	143	['drama', 'thriller', 'action']	['DE', 'US']		tt0450259	8	541547	47.39	7.5
28	Monty Pyth	MOVIE	King Arthur	1975	PG	91	['fantasy', 'action', 'comedy']	['GB']		tt0071853	8.2	534486	15.461	7.811
29	Nightcrawl	MOVIE	When Lou I	2014	R	118	['thriller', 'crime', 'drama']	['US']		tt2872718	7.8	531779	57.276	7.7
30	It	MOVIE	In a small t	2017	R	135	['horror', 'fantasy', 'thriller']	['US']		tt1396484	7.3	527626	95.827	7.234
31	Troy	MOVIE	In year 125	2004	R	163	['action', 'history', 'war', 'drama', 'euro	['MT', 'GB', 'US']		tt0332452	7.3	527447	48.407	7.1

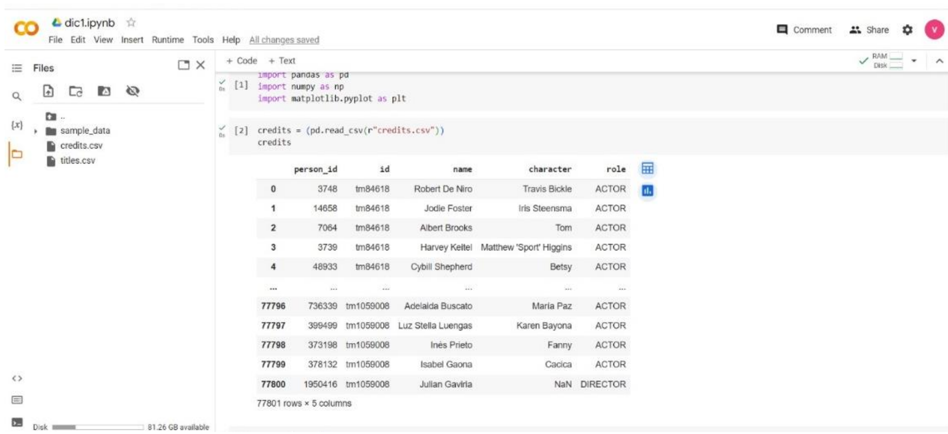
Credits data set on the other hand has over 70k+ records with 5 features.

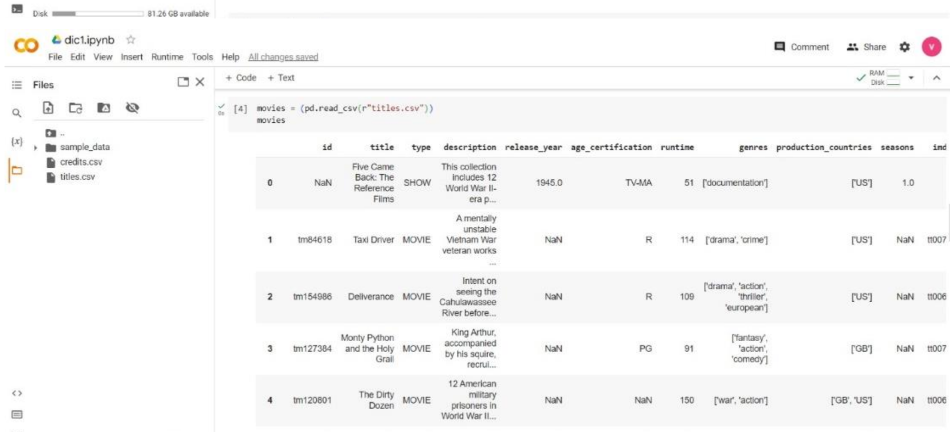


Link for the datasets - [https://www.kaggle.com/datasets/victorsoeiro/netflix-tv shows-and-movies](https://www.kaggle.com/datasets/victorsoeiro/netflix-tv-shows-and-movies)

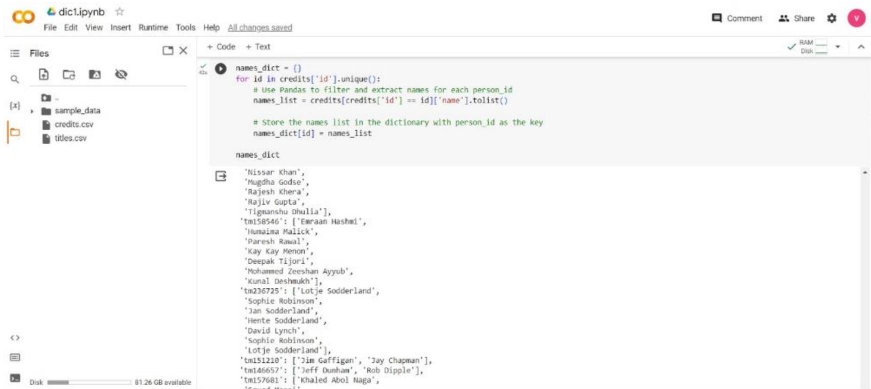
3. Data Cleaning/Processing:

1. Merged the titles and credits dataset:
   In the initial step, we have merged the two datasets we have taken on the basis of the ID field found in both datasets. First we have read the datasets and stored into dataframes as below:

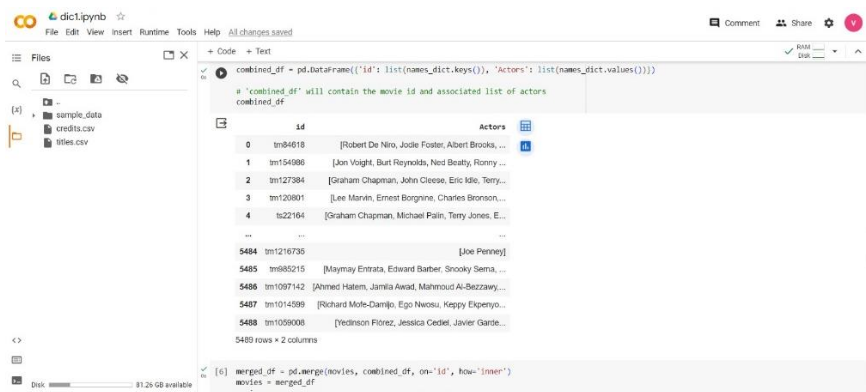




Next we have stored the Actor names as lists within a dictionary keeping the ID for all those actors as the key value:

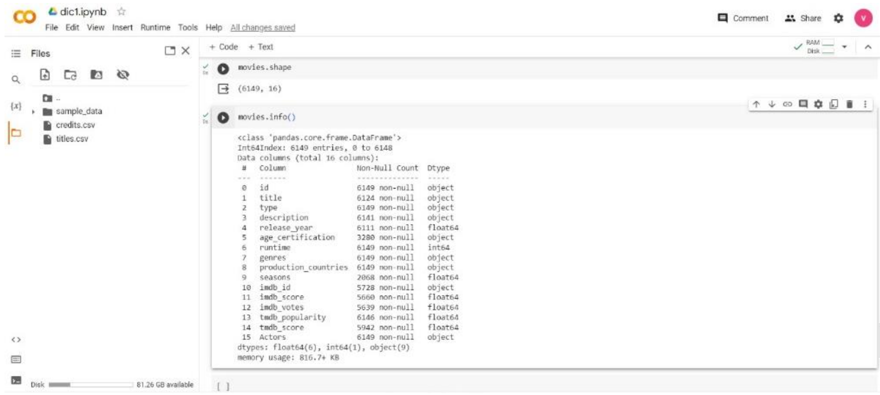


Now we have created a Dataframe containing the ID and list of Actor names:

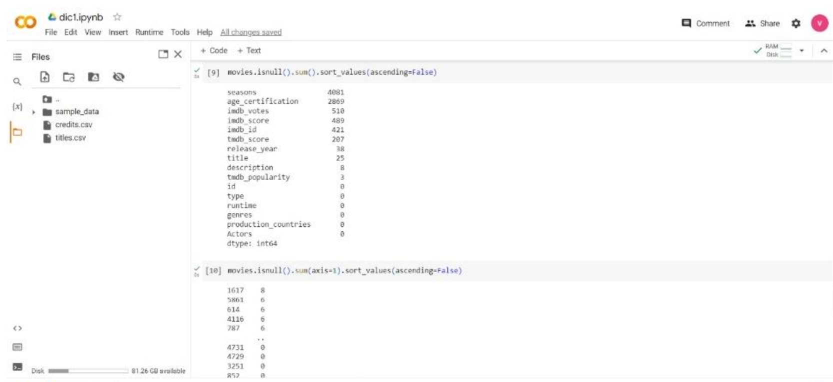


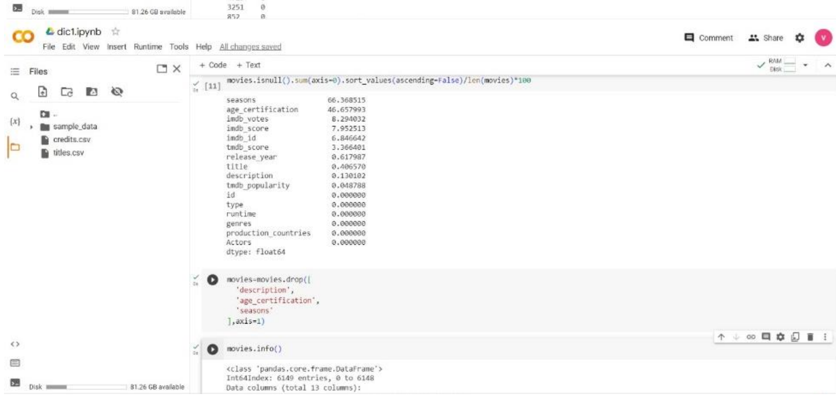
Finally we have merged both datasets on the basis ID field as below:

Below is the information of the final combined Dataframe:

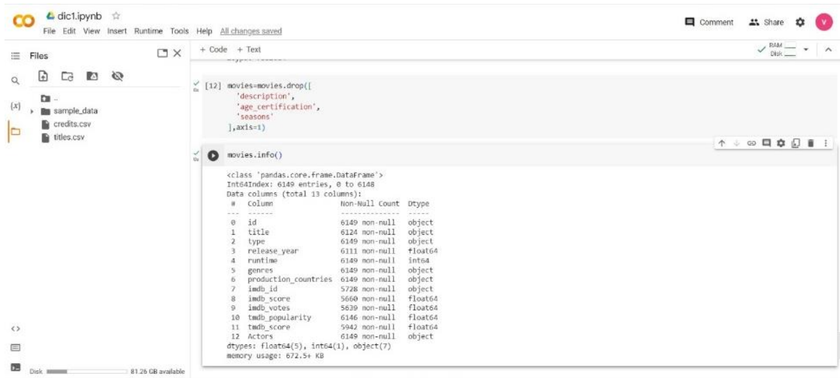


2. Removed Description, age_certification and season columns:
   For this step we have checked which columns are having the most count of NULL values as below:

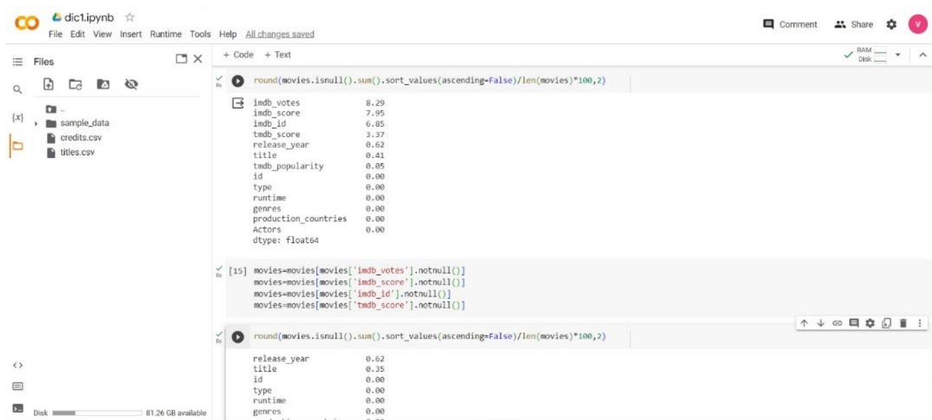




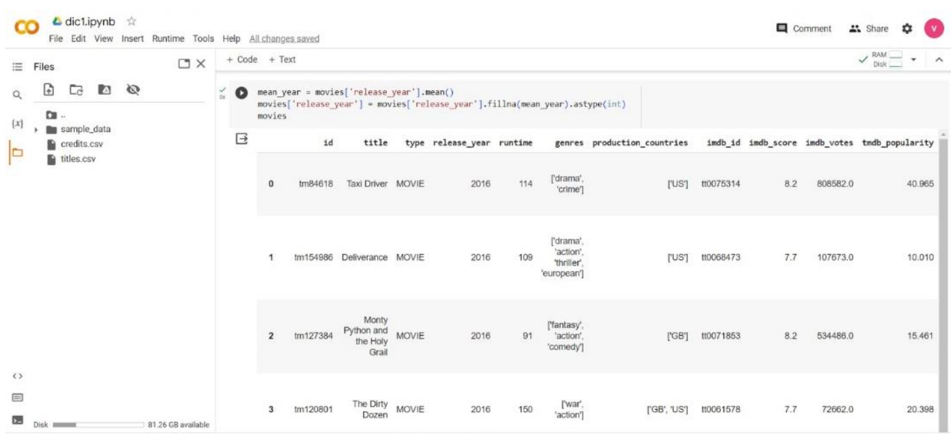
Based on the above statistics, we have removed the Description, Age_certification and Season columns as they are having high amount of NULL values and will also not be of much use while training the ML models.



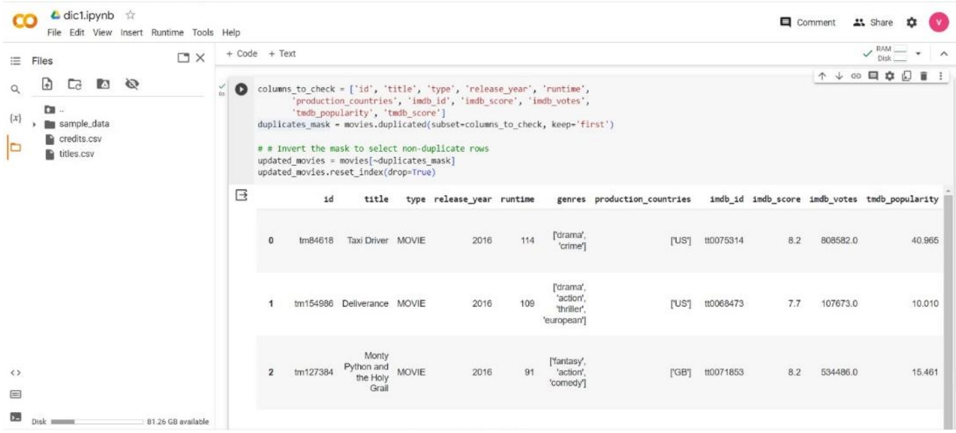
3. Calculated percentage of empty rows, then removed empty rows (blank spaces or Nulls):
   Here we are checking how many of the remaining columns have NULL values and using those columns, we have removed all the NULL rows as we require these columns later on.



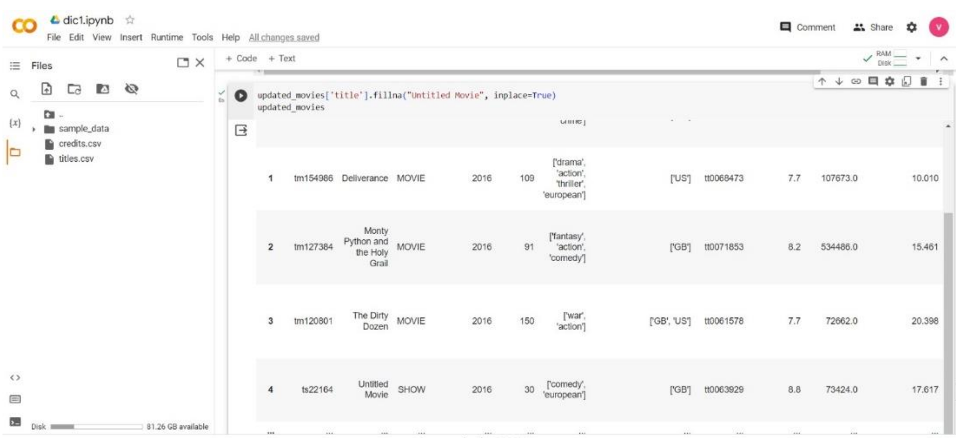
4. Filled in Missing values of Year column:
   In this step we have checked Year column and replaced all its NULL values using the mean value for the Year column as we cannot have this field be empty:



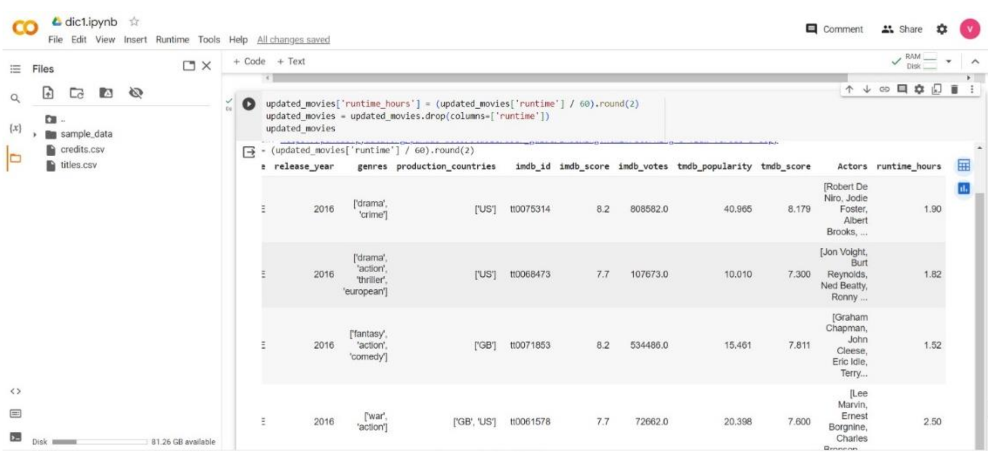
5. Removed duplicate rows:



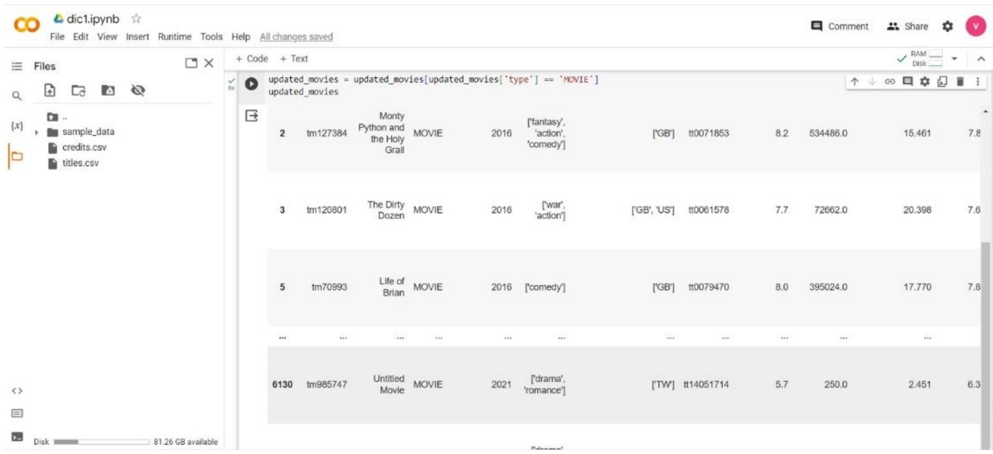
6. Movies which have Title field as empty, we have replaced as 'Untitled Movie':
   For this step we have replaced the Movies with empty titles to have a common 'Untitled Movie' name. This we will keep as it is as we are more concerned about predicting the final rating. The movie name only serves as a way to differentiate different proposals as per the problem statement we have given.



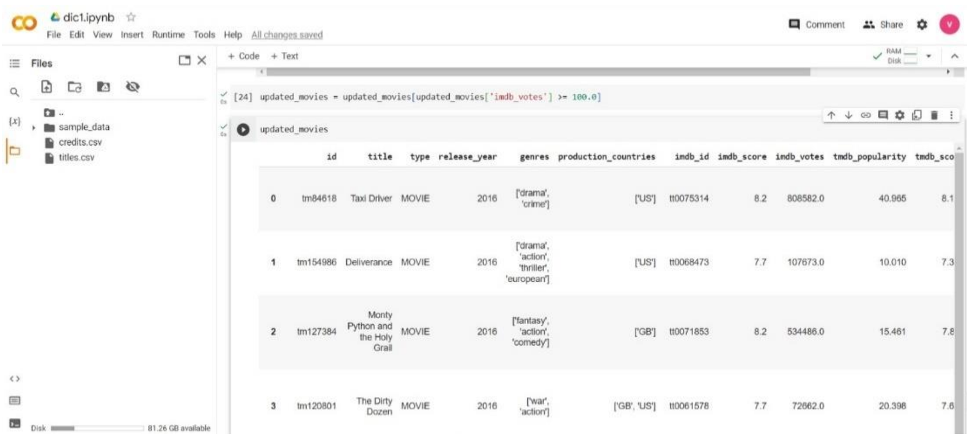
7. Converting run time from minutes to hours:
   We have converted runtime from minutes to hours to allow for better feature scaling:



8. We are removing all the data regarding shows and keeping only movies:
   For this step we are taking into consideration only Movies and rejecting TV shows to allow for better training in one specific field. Conflating both would lead to incorrect results as runtimes and actors would vary by a lot.



9. Removing outliers for Votes field:
   Here we have removed outliers lesser than 100 votes from the IMDB votes field.



10. Splitting genres through one-hot encoding and keeping only most relevant entries, ie with most frequency:
    Here we have performed one-hot encoding on the Genre column to create a numerical value for each class of this categorical variable to allow it to be included for processing within the ML models:

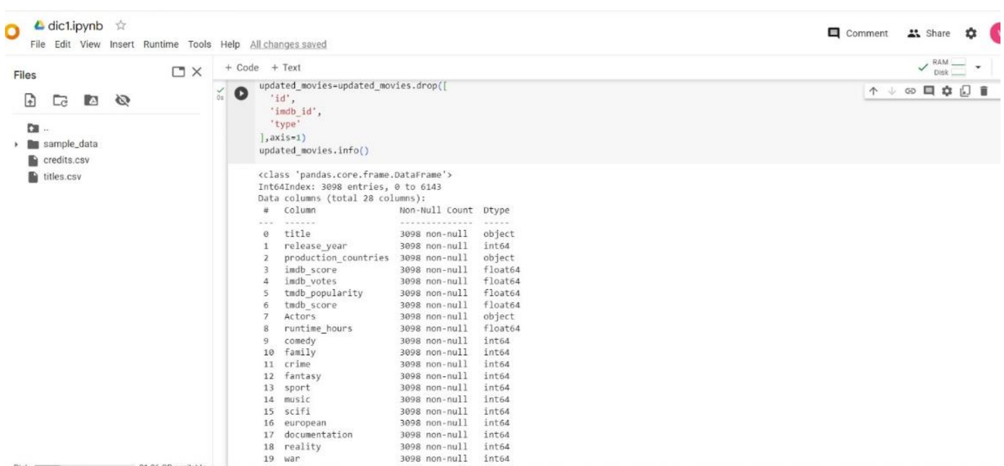
```
import ast

unique_genres = set([genre for genres_list in updated_movies['genres'] for genre in ast.literal_eval(genres_list)])

cleaned_genres = [genre.replace(' ', '_').replace(',', '').replace('[', '').replace(']', '') for genre in unique_genres]

one_hot_encoded_genres = updated_movies['genres'].apply(lambda x: pd.Series([1 if genre in x else 0 for genre in unique_genres], index=cleane

updated_movies = pd.concat([updated_movies, one_hot_encoded_genres], axis=1)

updated_movies.drop('genres', axis=1, inplace=True)
```

```
[28] updated_movies
```

```
updated_movies.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 3098 entries, 0 to 6143
Data columns (total 31 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   id                    3098 non-null   object
 1   title                 3098 non-null   object
 2   type                  3098 non-null   object
 3   release_year          3098 non-null   int64
 4   production_countries  3098 non-null   object
 5   imdb_id               3098 non-null   object
 6   imdb_score            3098 non-null   float64
 7   imdb_votes            3098 non-null   float64
 8   tmdb_popularity       3098 non-null   float64
```

As the final step, we have also removed the columns which we deemed unnecessary such as ID, IMDB_ID and TYPE:



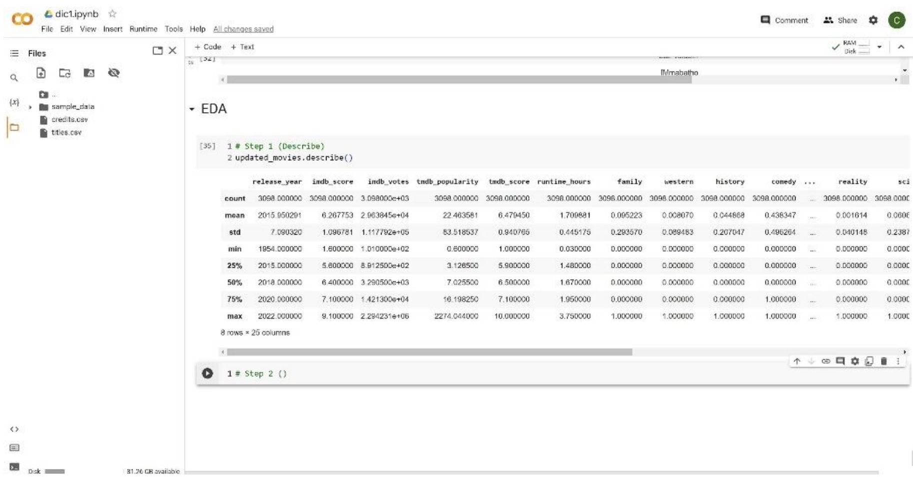
```
updated_movies=updated_movies.drop([
  'id',
  'imdb_id',
  'type'
],axis=1)
updated_movies.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 3098 entries, 0 to 6143
Data columns (total 28 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   title                 3098 non-null   object
 1   release_year          3098 non-null   int64
 2   production_countries  3098 non-null   object
 3   imdb_score            3098 non-null   float64
 4   imdb_votes            3098 non-null   float64
 5   tmdb_popularity       3098 non-null   float64
 6   tmdb_score            3098 non-null   float64
 7   Actors                3098 non-null   object
 8   runtime_hours         3098 non-null   float64
 9   comedy                3098 non-null   int64
 10  family                3098 non-null   int64
 11  crime                 3098 non-null   int64
 12  fantasy               3098 non-null   int64
 13  sport                 3098 non-null   int64
 14  music                 3098 non-null   int64
 15  scifi                 3098 non-null   int64
 16  european              3098 non-null   int64
 17  documentation         3098 non-null   int64
 18  reality               3098 non-null   int64
 19  war                   3098 non-null   int64
```

4. Exploratory Data Analysis:

As part of the EDA, we are performing the below steps:

1. Checking measures of Central Tendency and Variance:
   Using Describe() we are checking the Mean, Median, Standard Deviation, Maximum and Minimum value over the dataset to better understand its distribution:



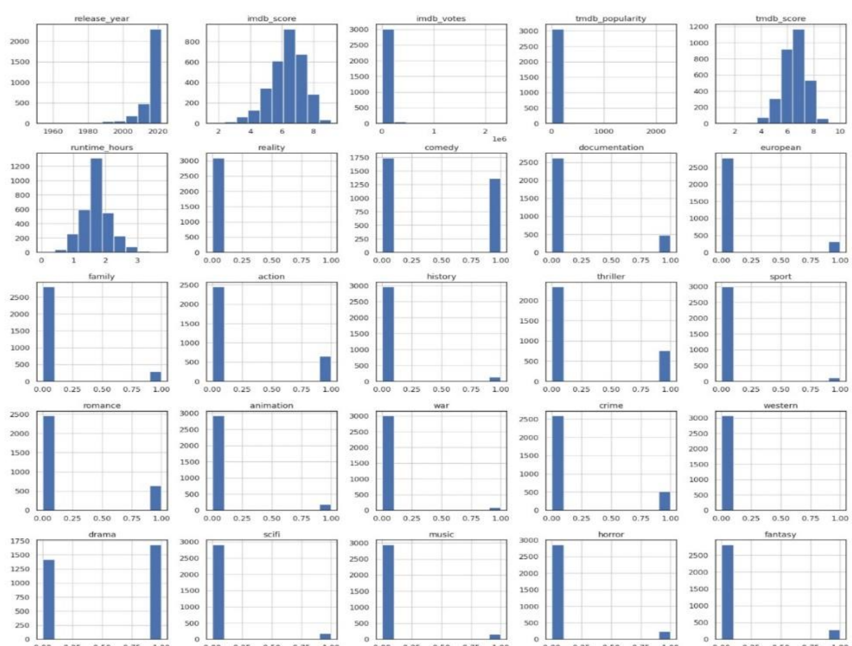
```
1 # Step 1 (Describe)
2 updated_movies.describe()
```

	release_year	imdb_score	imdb_votes	tmdb_popularity	tmdb_score	runtime_hours	family	western	history	comedy	...	reality	sci
count	3098.000000	3098.000000	3.098000e+03	3098.000000	3098.000000	3098.000000	3098.000000	3098.000000	3098.000000	3098.000000	...	3098.000000	3098.000
mean	2015.950291	6.267753	2.963845e+04	22.463581	6.479450	1.709881	0.095223	0.008070	0.044868	0.436347	...	0.001614	0.060
std	7.090320	1.096781	1.117792e+05	83.518537	0.940765	0.445175	0.293570	0.089483	0.207047	0.496264	...	0.040148	0.2387
min	1954.000000	1.600000	1.010000e+02	0.600000	1.000000	0.030000	0.000000	0.000000	0.000000	0.000000	...	0.000000	0.000
25%	2015.000000	5.600000	8.912500e+02	3.126500	5.900000	1.480000	0.000000	0.000000	0.000000	0.000000	...	0.000000	0.000
50%	2018.000000	6.400000	3.290500e+03	7.025500	6.500000	1.670000	0.000000	0.000000	0.000000	0.000000	...	0.000000	0.000
75%	2020.000000	7.100000	1.421300e+04	16.198250	7.100000	1.950000	0.000000	0.000000	0.000000	1.000000	...	0.000000	0.000
max	2022.000000	9.100000	2.294231e+06	2274.044000	10.000000	3.750000	1.000000	1.000000	1.000000	1.000000	...	1.000000	1.000

8 rows × 25 columns

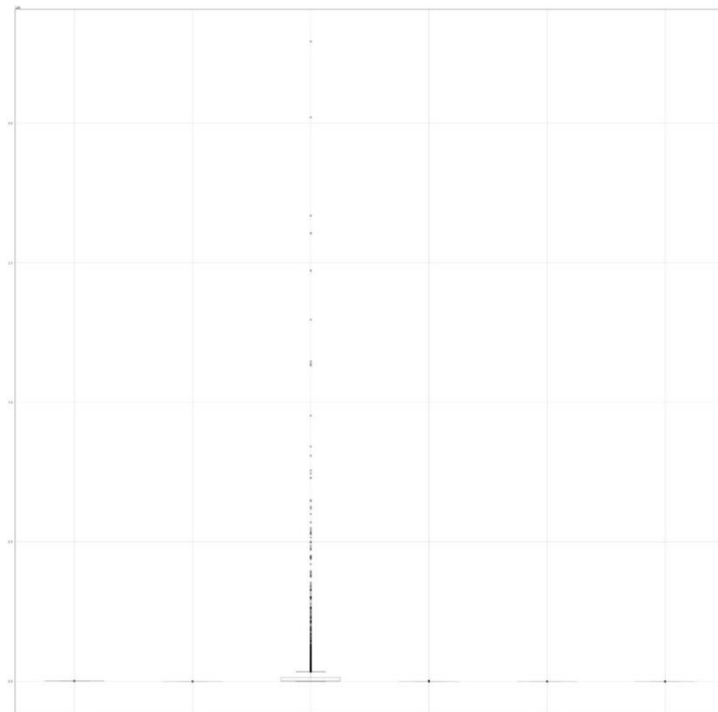
```
1 # Step 2 ()
```

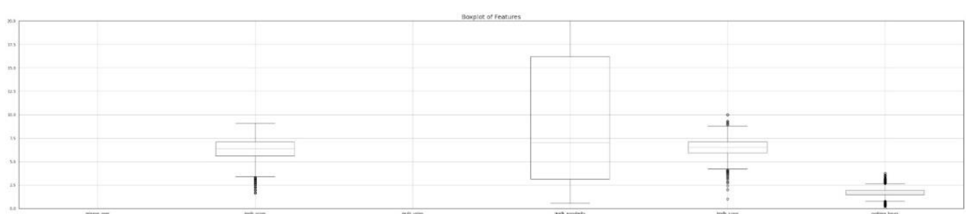
2. Plotting Histogram:
   Here we have plotted histograms to understand the distribution of all the fields. We can see that both score fields have normal distributions along with run-time hour fields. The genre fields are all one hot encoded to have only 1 value so for models which require prediction based on normalized values such as logistic regression, we will assign weighted value to each genre based on total frequency and combine all the separate fields into one to allow for better prediction for such type of models. For models requiring classification, we can use the remaining data and fields with good correlation to predict the genre of the particular movie.



3. Plotting Boxplots:

   We have used boxplots to identify outliers in all the fields and as expected, there will not be anything in the genre fields however it is showing we have outliers in the votes field. We can remove these particular outliers going forward for a few specific models that we will utilize depending on the requirement however for regression analysis, it might be better to keep the values since the number of votes being greater signifies a greater weightage toward the film score being more solid.

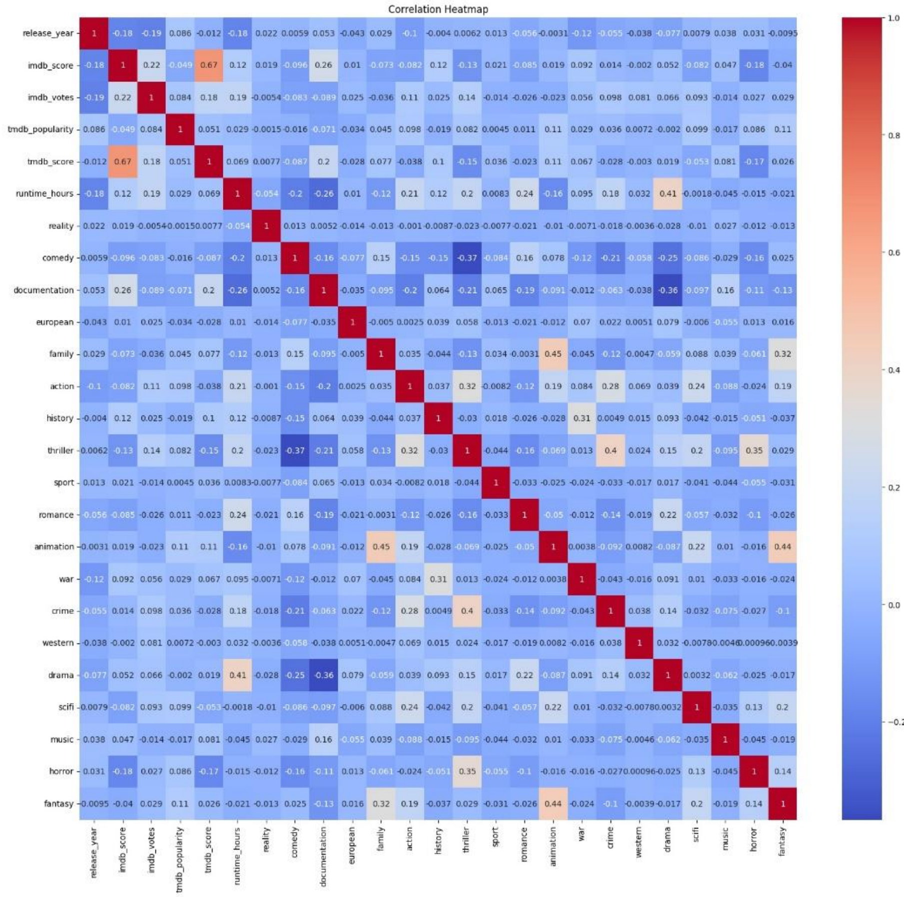




   From the above, we can see that the scores also have a few outliers although they should not pose much of a problem as there are not too many.

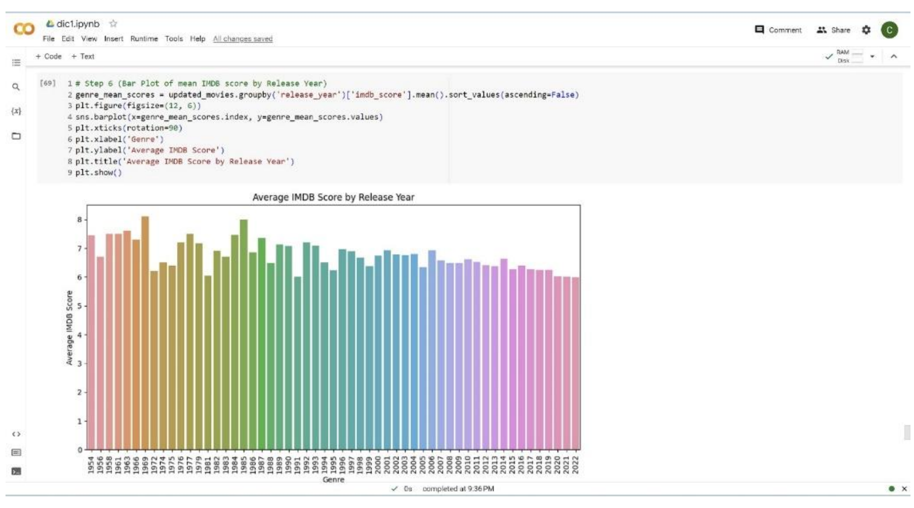
4. Plotting Correlation Matrix:

   We have plotted a correlation matrix to check correlation between features. We can see that the IMDB score has good correlation with the TMDB score and a pretty good correlation with the IMDB votes. The other genre fields which were one hot encoded have differing levels of correlation with all the fields although there are particular genres which have good correlation with other genres.



Based on the relation between IMDB score and TMDB score we can see that we can use one system to more or less predict the score in the other system, if model is trained accurately.

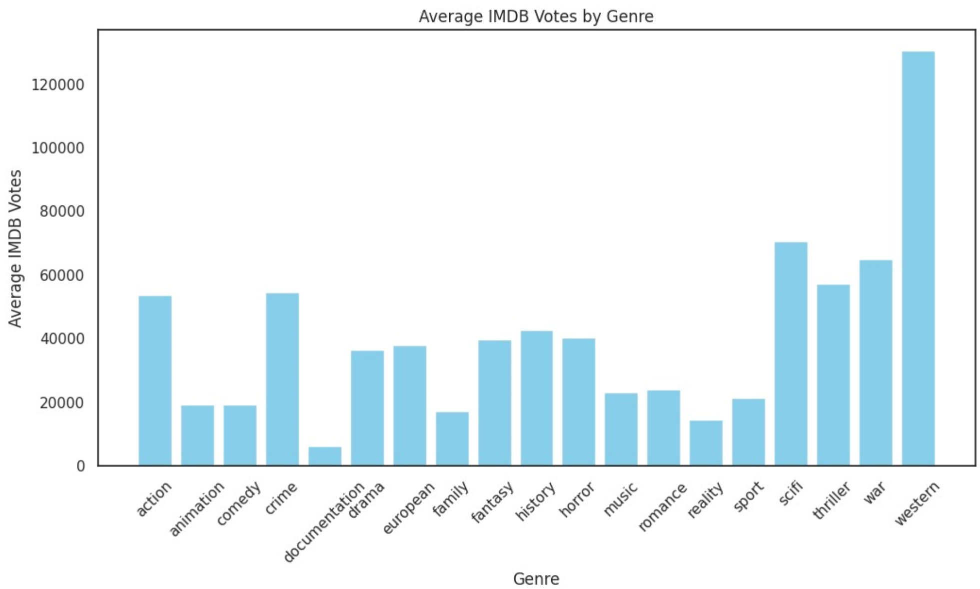
5. Plotting bar plot:
   Here we have plotted a bar plot between IMDB score and year. We can see that the variation is quite little majority of scores being between 6 and 8 throughout the years.



6. Frequency Bar plot between each Genre and Votes:

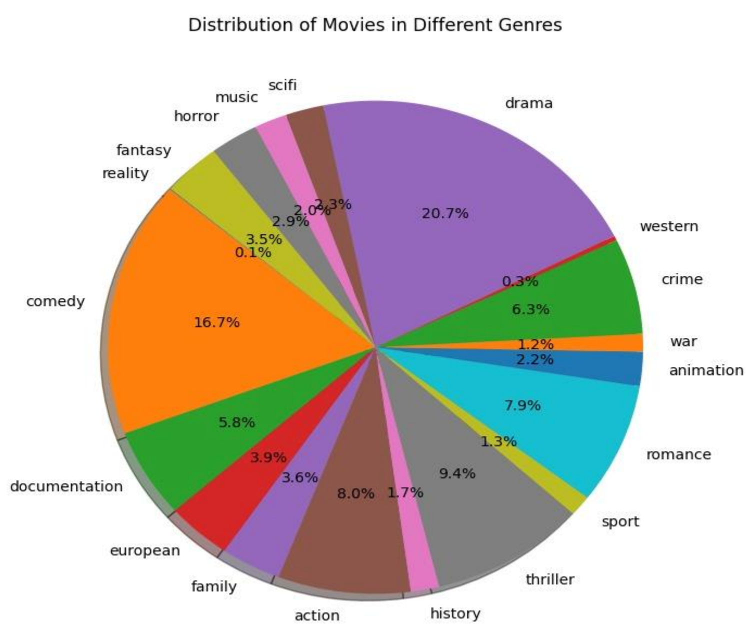
   Here we are checking the total number votes per genre to understand which genres are the most common and therefore most popular. From below we can see that westerns are extremely popular followed by Scifi and War themed movies.

   Based on this we should be able to identify the more popular genres through the number of votes.



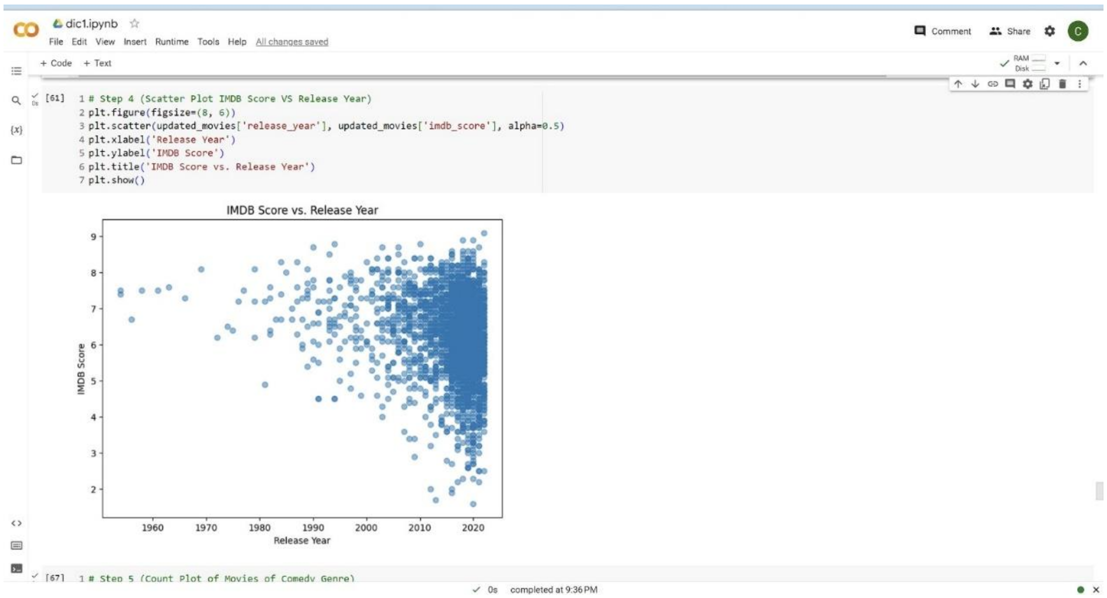
7. Pie chart to show percentage of all genres:

   We can see the percentage of Genres for all of the films as below. From below we are able to see that drama and comedy are the most common followed by romance and action genres. Through this we can see that these are important Genres although further analysis is required.



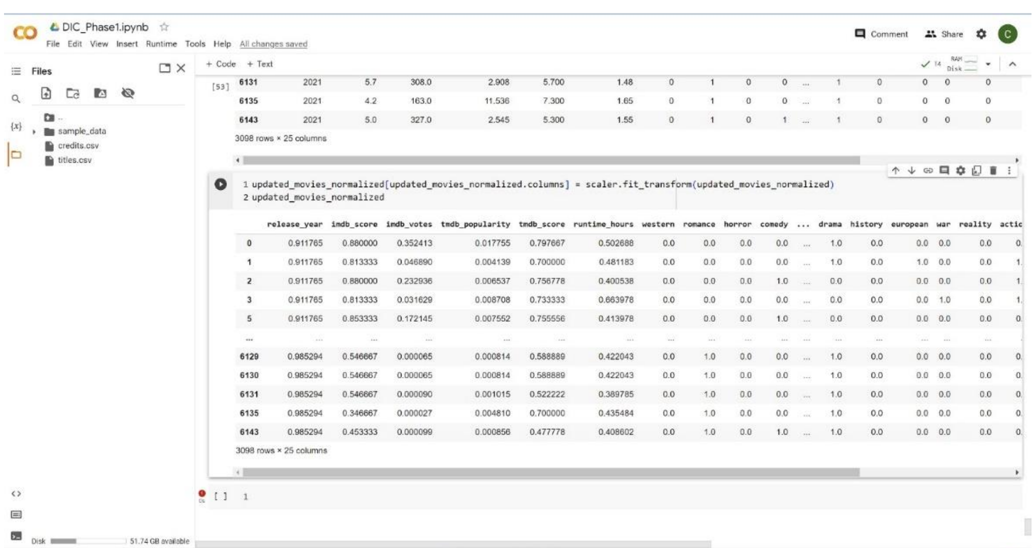
8. Scatter plot:

   Here we are plotting a scatter plot between score and release year to check score throughout the years. We see that a huge amount of content is generated progressively throughout the years and the vast majority of content is concentrated in the 2000 to 2020 years. Based on this we will aim to remove the films which are lesser than 1980 to remove the outliers from that time and keep the rest.



9. Normalize the data:

   Here we have normalized the data to allow for features such as Year and Population to be scaled properly and not cause overfitting due to their extreme values. Also, through this normalization, it will be easier to fit the training model for a few ML algorithms.



6131	2021	5.7	308.0	2.908	5.700	1.48	0	1	0	0	...	1	0	0	0	0
6135	2021	4.2	163.0	11.536	7.300	1.65	0	1	0	0	...	1	0	0	0	0
6143	2021	5.0	327.0	2.545	5.300	1.55	0	1	0	1	...	1	0	0	0	0

	release_year	imdb_score	imdb_votes	tmdb_popularity	tmdb_score	runtime_hours	western	romance	horror	comedy	...	drama	history	european	war	reality	actic
0	0.911765	0.880000	0.352413	0.017755	0.797667	0.502688	0.0	0.0	0.0	0.0	...	1.0	0.0	0.0	0.0	0.0	0.
1	0.911765	0.813333	0.046890	0.004139	0.700000	0.481183	0.0	0.0	0.0	0.0	...	1.0	0.0	1.0	0.0	0.0	1.
2	0.911765	0.880000	0.232936	0.006537	0.756778	0.400538	0.0	0.0	0.0	1.0	...	0.0	0.0	0.0	0.0	0.0	1.
3	0.911765	0.813333	0.031629	0.008708	0.733333	0.663978	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	1.0	0.0	1.
5	0.911765	0.853333	0.172145	0.007552	0.755556	0.413978	0.0	0.0	0.0	1.0	...	0.0	0.0	0.0	0.0	0.0	0.
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
6129	0.985294	0.546667	0.000065	0.000814	0.588889	0.422043	0.0	1.0	0.0	0.0	...	1.0	0.0	0.0	0.0	0.0	0.
6130	0.985294	0.546667	0.000065	0.000814	0.588889	0.422043	0.0	1.0	0.0	0.0	...	1.0	0.0	0.0	0.0	0.0	0.
6131	0.985294	0.546667	0.000090	0.001015	0.522222	0.389785	0.0	1.0	0.0	0.0	...	1.0	0.0	0.0	0.0	0.0	0.
6135	0.985294	0.346667	0.000027	0.004810	0.700000	0.435484	0.0	1.0	0.0	0.0	...	1.0	0.0	0.0	0.0	0.0	0.
6143	0.985294	0.453333	0.000099	0.000856	0.477778	0.408602	0.0	1.0	0.0	1.0	...	1.0	0.0	0.0	0.0	0.0	0.

For this step we will create a separate column representing Genre as 1 value numerically. Here we wanted to create a column that is the sum of the genres with each genre being multiplied with a proportional weighted sum. It is our belief that this would be useful for at least one ML model in phase 2.

PHASE 2

Decision Tree Classifier:

Justification:

- Because of its simplicity of interpretation and capacity to manage intricate, non-linear relationships within the data, the Decision Tree method was selected.
- Movies can be good for a lot of varied reasons (genres, directors, budgets, etc.), and Decision Trees' non-parametric structure makes it possible to model these interactions well.

Work Done:

- A grid search was performed across multiple hyperparameters, such as tree depth and the minimal number of samples needed to split an internal node, to improve the Decision Tree.
- This was essential to maintaining the model's generalizability and avoiding overfitting.

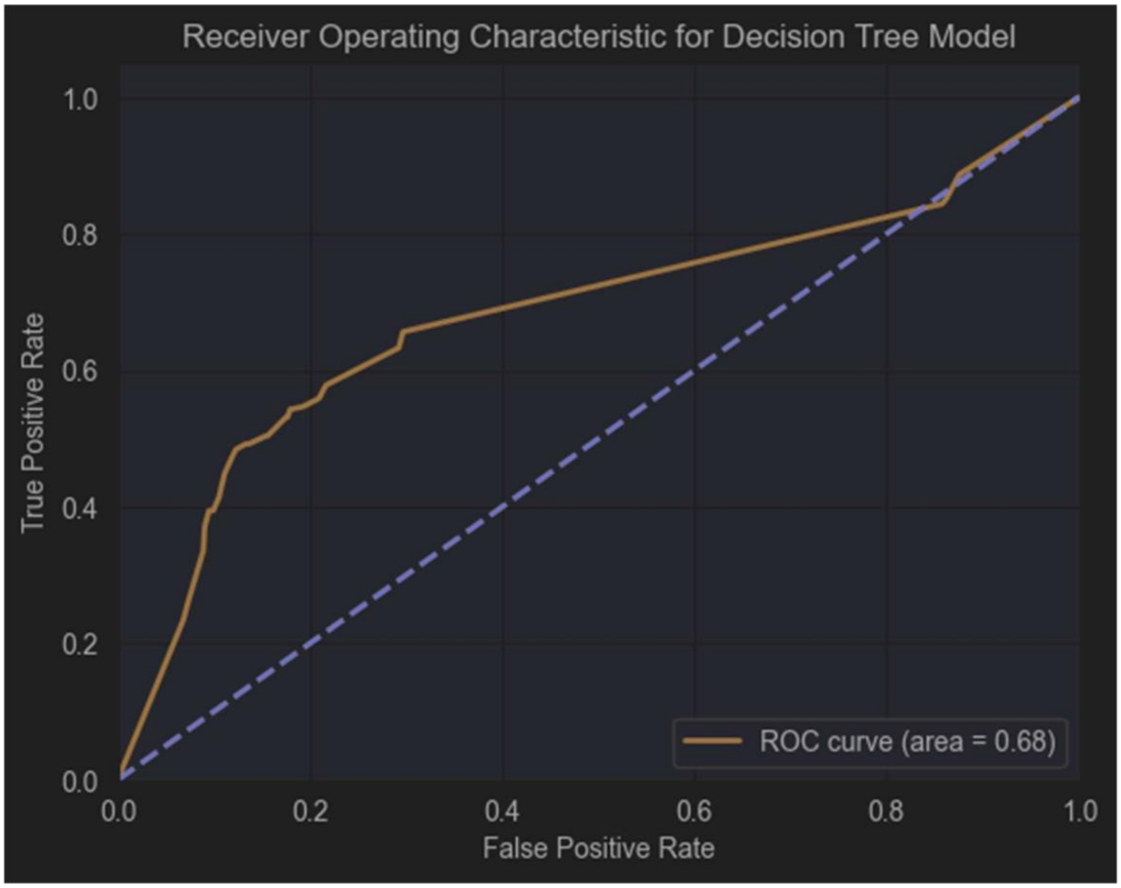
Model Effectiveness:

- After the model was evaluated, the Decision Tree obtained an AUC value of 0.6797 and an accuracy of roughly 76.24%.

```
Decision Tree Model Accuracy: 0.7623655913978494
AUC Score: 0.6797077846810089
```

ROC Curve Analysis:

- With an AUC of 0.68, the Decision Tree model's ROC curve indicates a decent ability to distinguish between the classes. Given that the curve is closer to the diagonal line of no discrimination than the ideal top-left corner, this shows that there is some opportunity for improvement.



Logistic Regression:

Justification:

- Because of its effectiveness in predicting probabilities—a critical skill when dealing with binary outcomes like good vs. bad movies—Logistic Regression was chosen. It can be used as a benchmark for more complicated models, providing a linear decision boundary as a baseline.

Work Done:

- To prevent both overfitting and underfitting in Logistic Regression, the regularization strength (C) was adjusted to strike the ideal balance between training data fit and model complexity.

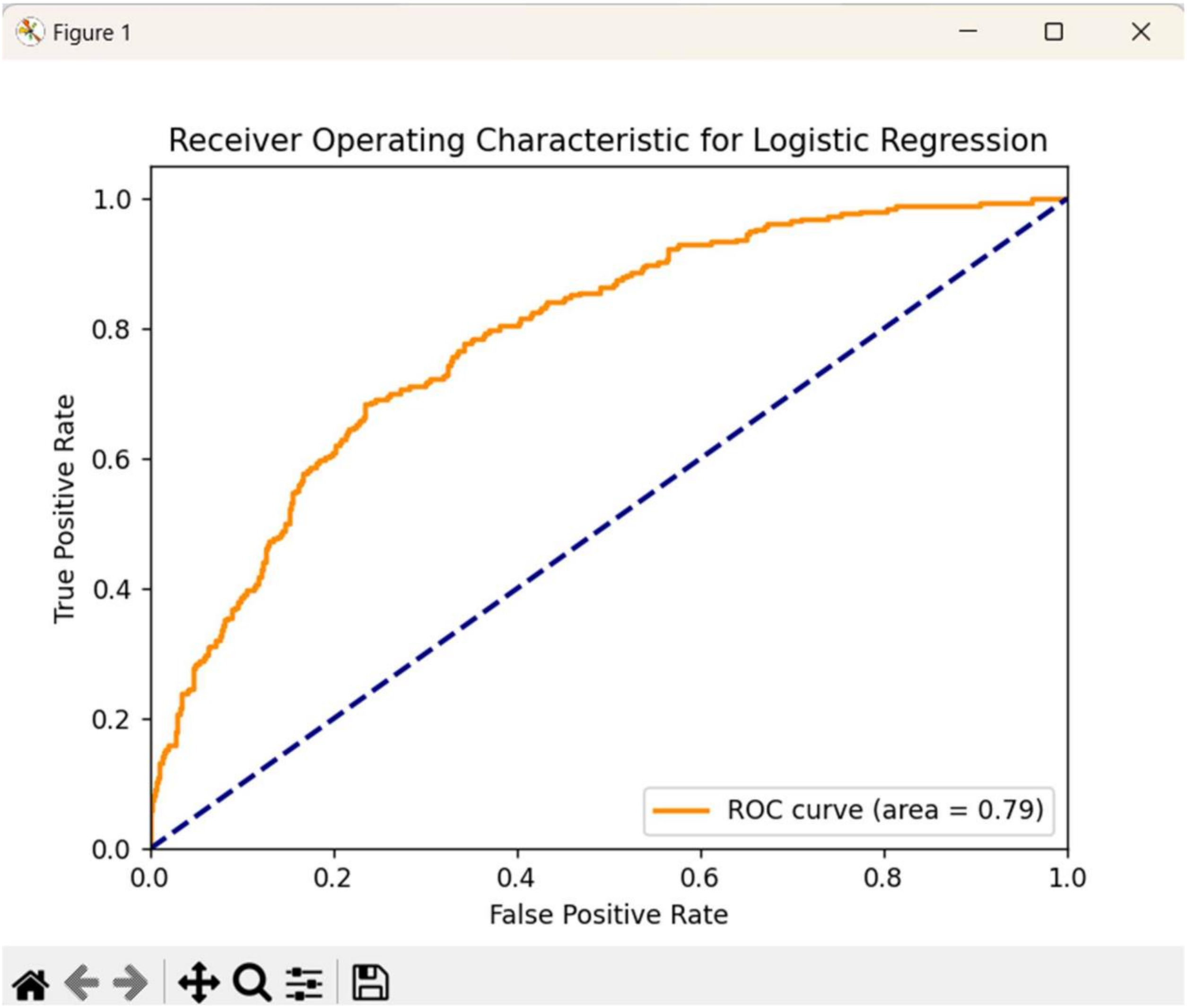
```
Logistic Regression Model Accuracy: 0.7602150537634409
AUC Score: 0.78573581231454
```

Model Effectiveness:

- In terms of accuracy, the Logistic Regression model performed better than the Decision Tree, achieving an AUC score of 0.7857 and an approximate 76.02% superiority.

ROC Curve Analysis:

- With an AUC of 0.79, the ROC curve for the Logistic Regression model shows improved discriminative capacity. In comparison to the Decision Tree, the curve is closer to the top-left corner, indicating a greater true positive rate for most thresholds and a smaller false positive rate.



Insights and Conclusions:

- At the specified threshold of IMDb scores, the higher AUC score of the Logistic Regression model suggests that it is more successful in differentiating between 'good' and 'not good' movies. Despite the possible non-linearity in the data relationships, the enhanced discriminative performance of Logistic Regression indicates that the linear decision boundary is suitable for this dataset.
- Although the two models' accuracy scores are similar, the greater AUC of the logistic regression suggests that accuracy alone may not be the most useful statistic for assessing the model, particularly in cases where the data is unbalanced.

Linear Regression:

Justification:

- We chose linear regression, as it predicts the continuous output variables based on the independent input variables.
- In our scenario, the target variable is *imdb_score* as it ranges from 0 – 1, and we have independent variables like *release_year, imdb_votes*, *runtime_hours*, and different *genres*.
- We wanted to check the linear relationship between our target variable and all the independent variables to determine the best model for our dataset.

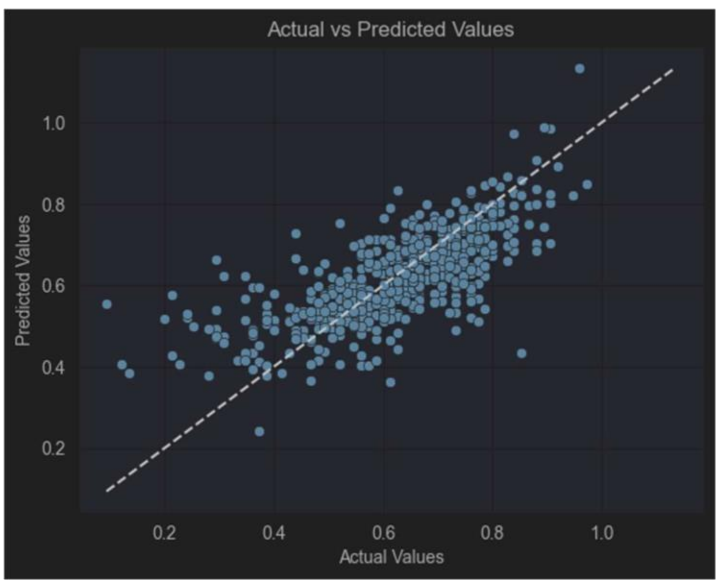
Work Done:

- Splitting the dataset into training and testing set in 80-20% ratio.
- We selected the normalized version of our dataset by removing *title*, *production_countries*, and *actor's* columns.
- Linear regression does not require any external hyperparameters to train.

Effectiveness:

- We have calculated two performance metrics, Mean Squared Error (MSE) and R-squared (R^2):
    - Mean Squared Error (MSE): It is a function that measures how close the data points are to the regression line. We obtained 0.009243, which is quite low, and it represents that our predictions are close to our actual values.
    - R-squared (R^2): It is a function that calculates how well the data fits with the regression model. We obtained 0.5405 which means it is 54.05% variance in the *imdb_score* can be explained by the model. As less than half of the model is unexplained, this may be due to non- linear relationships between some features and *imdb_score*.

- This is the heatmap of all the features correlated to each other, representing which features are strongly associated with each other.
- Higher values show they are more related to each other and can be used further for data processing. Lower values represent those features and not that co-related.



- As we can see that many data points are concentrated in the middle and for some data points, the error is too high, meaning that those points are not linearly related.
- This model effectively analyzes and provides how features are linearly related in providing higher IMDB scores, however it does not capture non-linear relationships.

Random Forest Regression:

Justification:

- Random Forest regression is an ensemble learning method which creates multiple decision trees during training and provides an output of the average of all the predictions of all trees.
- As we saw above, there are some non-linear relationships between the target column and feature columns.
- Random Forest can check non-linear relationships between the features and the target variable. This may provide a better understanding or fit for our dataset.
- Random forest reduces overfitting, as it averages multiple decision trees.
- It works for both categorical and continuous variables.

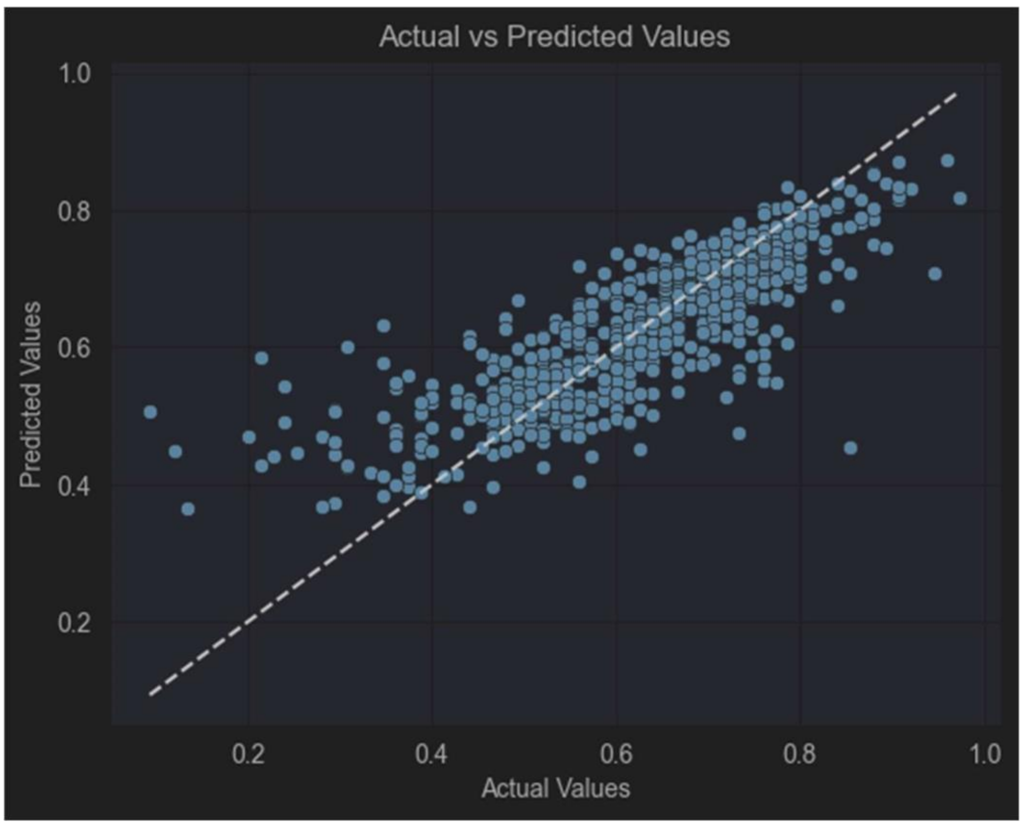
Work Done:

- Splitting the dataset into training and testing set in 80-20% ratio.
- We selected the normalized version of our dataset by removing *title*, *production_countries*, and *actor's* columns.
- Random Forest has various hyperparameters:
    - *n_estimators (Number of Trees)*
    - *max_depth (Maximum Depth of Each Tree)*
    - *min_samples_split (Minimum Number of Samples required to Split an Internal Node)*
    - *min_samples_leaf (Minimum Number of Samples required at a leaf node)*
    - *max_features (Number of features to consider when splitting)*
- To tune the hyperparameters, we used trial and error method by manually increasing and decreasing each hyperparameter, and finalizing on the hyperparameters which gave us the least Mean Square Error and High R-squared. Some of the observations are mentioned below:

n_estimators	***max_depth***	***min_samples_split***	***min_samples_leaf***	***max_features***	***MSE***	R^2
100	None	2	1	1	0.008678	56.86%
500	None	2	1	1	0.008528	57.62%
1000	None	2	1	1	0.008472	57.89%
1500	None	2	1	1	0.008451	57.99%
2500	None	2	1	1	0.008447	58.01%
1000	None	2	1	5	0.006842	65.99%
2000	17	2	1	6	0.006822	66.09%

Effectiveness:

- We have calculated two performance metrics, Mean Squared Error (MSE) and R-squared (R^2):
  - Mean Squared Error (MSE): It is a function that measures how close the data points are to the regression line. We 0.006822, which is lower than our Linear Regression model, means that it predicts the value more close than Linear Regression model.
  - R-squared (R^2): It is a function that calculates how well the data fits with the regression model. We obtained 0.66087 which means it is 66.087% variance in the *imdb_score* can be explained by the model. This higher R^2 value indicates a better fit.



- From the above graph we can see that the data points are less dispersed and are more concentrated towards this line.
- We can figure out that data points with actual values of below 0.4 are not predicted accurately. This may be due to the smaller number movies with imdb scores of 0.4 or 4 provided to our model. If we increase our dataset for such values then, this model may fit in a better way for our dataset.
- Because of its capability of handling non-linear relationships and providing average predictions of 2000 decision trees, this model is less prone to overfitting.

KMeans:

Justification:

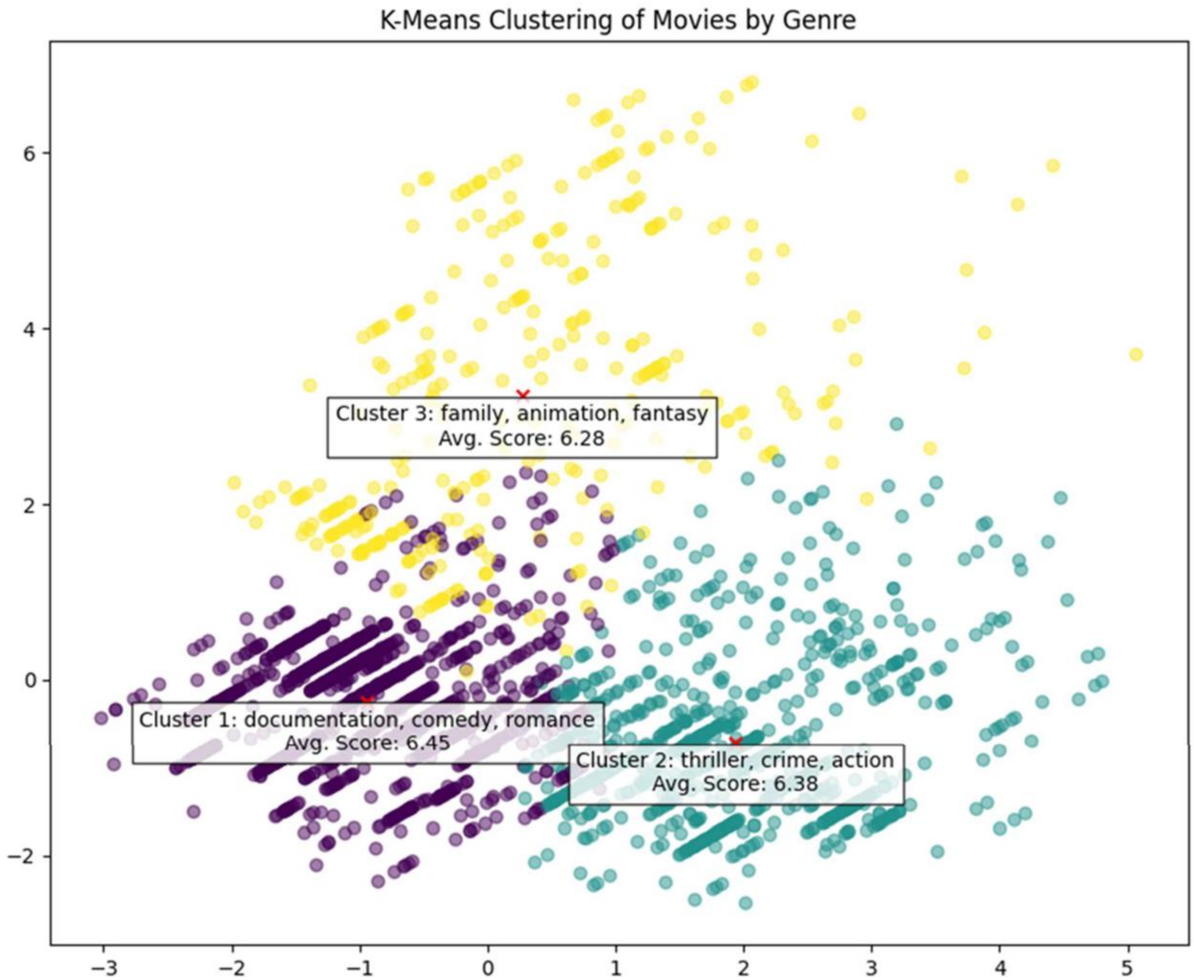
- We chose KMeans to be able to find out more information regarding the genres of the film. To be more specific, we wanted to see if we could glean any intelligence about the combination of genres through this method.
- We believe KMeans is an effective model to find out this info as it is applied on unstructured data to gain insight into possible patterns within the data.

Work Done:

- As this Algorithm does not require data to be trained, we have only separated the genres from the processed dataset, standardized them, and then applied the KMeans algorithm.
- The three least common genres ('reality', 'war', 'western') have been removed.
- We then plotted the graph after reducing dimensionality to 2 through PCA. Within the clusters, we tracked the 3 most common genres and displayed those.
- Also, we have calculated the overall score of the movies in each cluster to give an idea of the relation of the score to the clustering of the genres.

Effectiveness:

- Our KMeans output visualized below reveals clusters of movies based on their genres and their average IMDb scores. We have taken k=3 as the best value from the elbow graph. Within the obtained 3 clusters, we see that the most common movie genres among each cluster are as: Cluster 1 – Documentation, Comedy, Romance; Cluster 2 – Thriller, Crime, Action; Cluster 3 – Family, Animation, Fantasy. Based on this info, we can see that these 3 movies are grouped together suggesting that there is prevalence of combinations of the genres in each cluster. For example, movies with a combination of genres such as Action, Crime and Action, Thriller and Thriller, Crime are observed in cluster 3. The same applies to the other 2 clusters. We also observe that average score is highest in cluster 1 which suggests that movies with genre combinations in cluster 1 such as Comedy, Romance and Documentation are the ones which are highly received hence, the higher score. Cluster 2 has the second highest score suggesting that movies with genres such as thriller, crime and action are also quite popular. From this output we can observe that the highest genres obtained per cluster are an excellent choice of combination. Choosing such combinations would be a clever idea to obtain the highest rating.



KNN:

Justification:

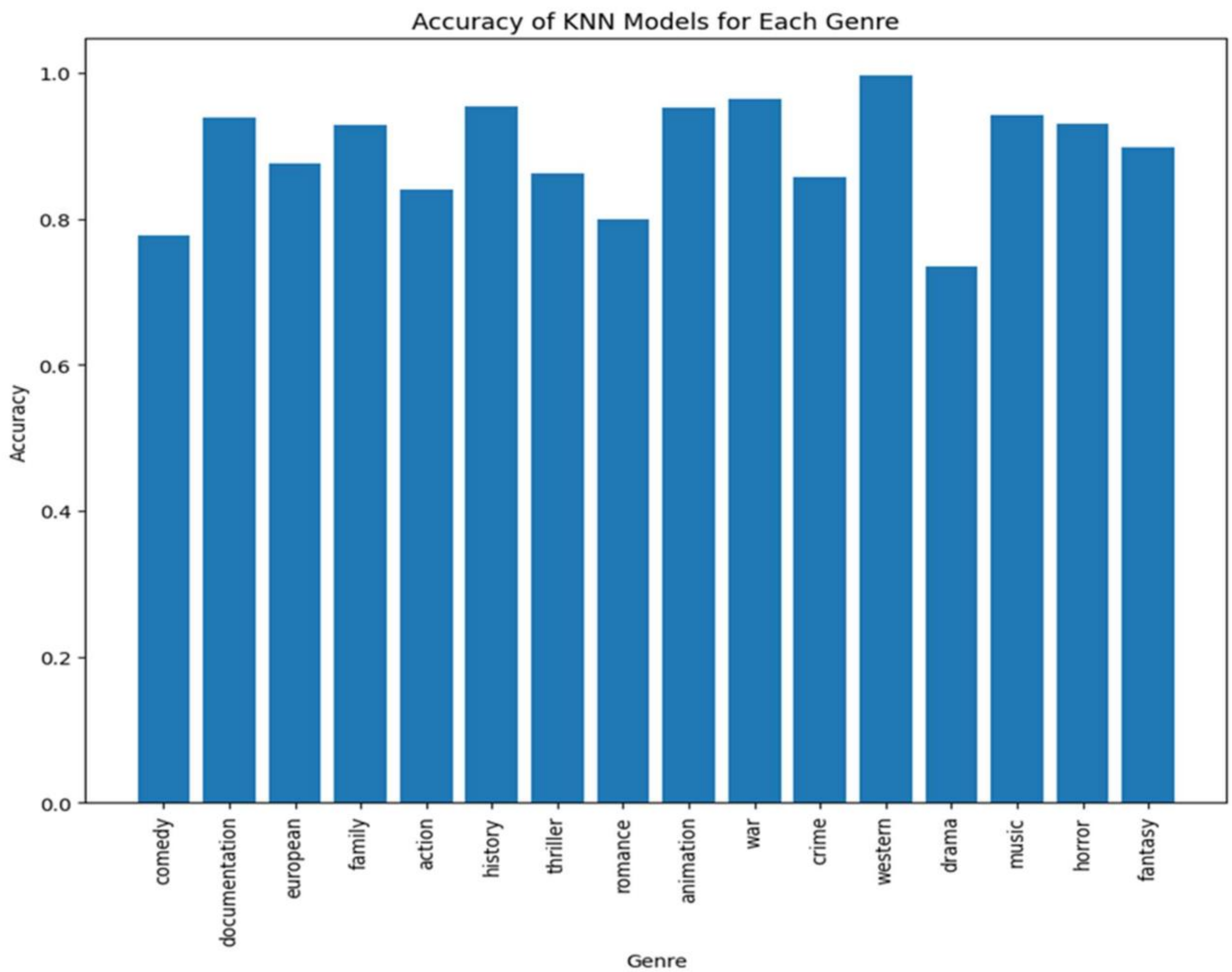
- We chose KNN here as it is a simple model which can help us to create a good identification system using its classification ability.
- Our aim here is to be able to identify the genre which in turn can be used for recommendation systems for users. Or this can also be used to correctly identify genres by production enterprises such as Netflix so that they can push forward those with greater chances of success.

Work Done:

- For this model we have taken the input to be the dataset and the target identification genre of our choice. On this basis, we have split the dataset into the target feature (genre to be identified) and the rest of the input features. After splitting the data into training and testing datasets, we trained the model using the training set and then validated the model on the testing set.
- One more thing we have done here is to try and predict each genre separately. To do this, we have provided a list of genres excluding the ones with lesser count of movies or those returning incorrect values and then iterated the model over each genre to see how well it is predicting each one.
- We have created a classification report for each genre and its separate confusion matrix as well. After plotting all of these, we have also plotted a final graph to store the accuracy obtained by the model on predicting each genre.

Effectiveness:

- We can see that our model returns predictions with quite good accuracy overall. The accuracies obtained are as below:



- Based on the above, we can say with certainty that the genre of any given movie can be predicted properly by utilizing KNN. This is immensely helpful in being able to create a recommendation system for users. It is not only in creating movies with the highest score but also in providing for the individual tastes of different users. By catering to these needs, we can properly channel the content based on the user and maximize what kinds of films can get pushed to production based on the overall popularity of each genre. Thus, this KNN application may not have been used to predict the score but by being able to successfully predict the genre, we have opened another gateway into the success of films which lies in distribution of films to users based on content.

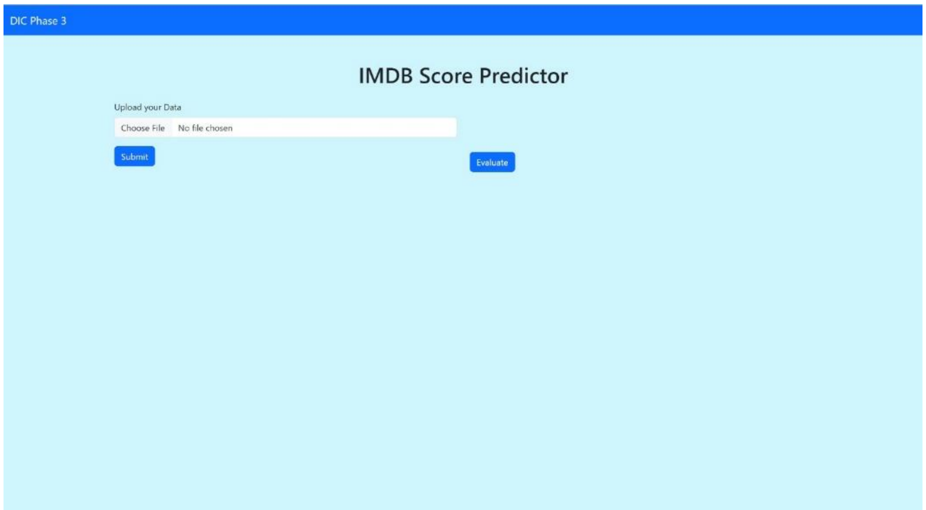
PHASE 3

Working instructions:

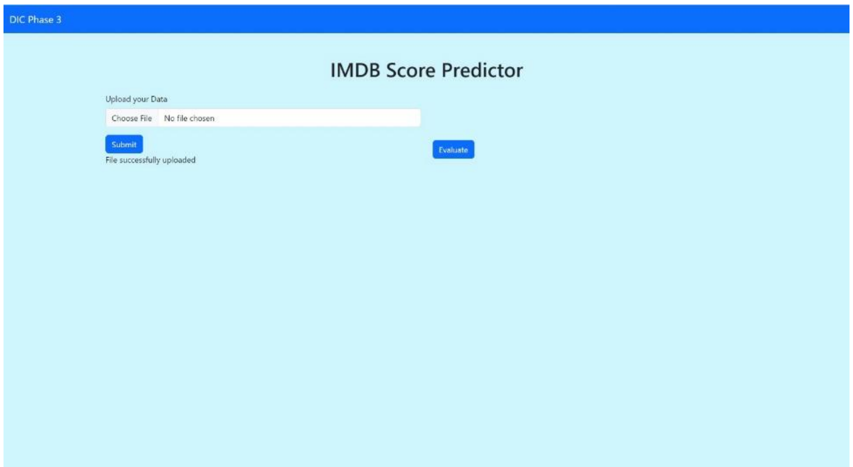
1. First open code for phase-3 (recommended to open through Jupyter Notebook.)

2. Run the code.

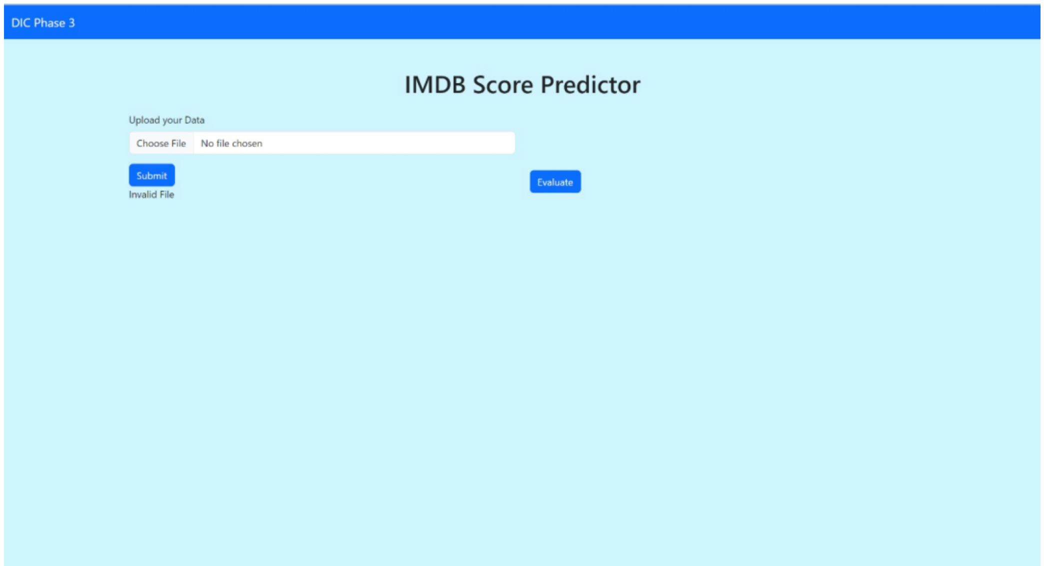
3. Open 127.0.0.1:5000 to view the generated website.



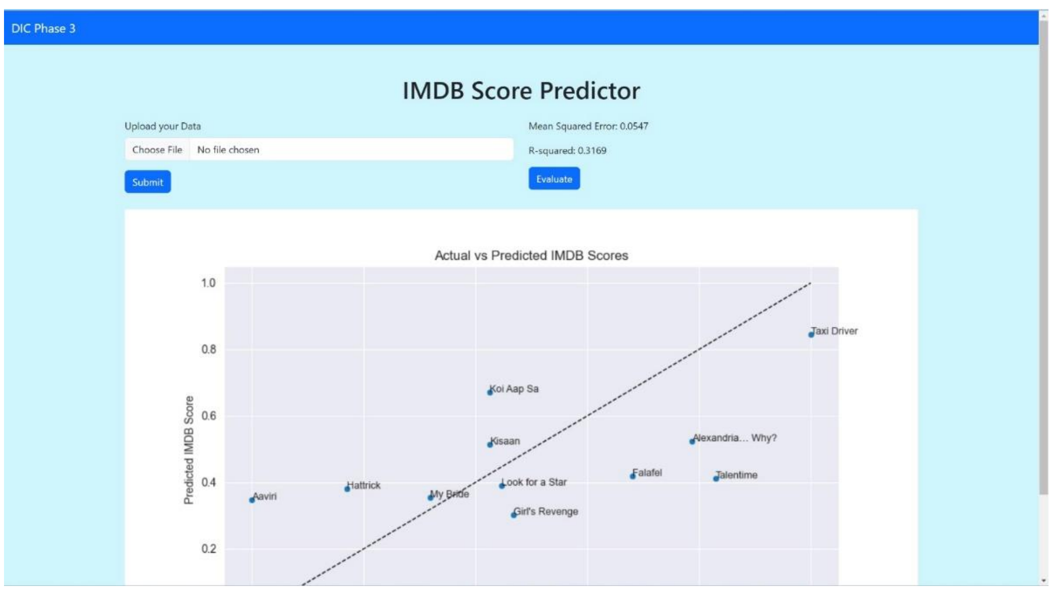
4. Click on <Choose File> and select the dataset provided. Next click submit in order to view if file is uploaded successfully. If successful, message should appear as below:



5. In case of ‘invalid file’ message as shown below, please upload the corrected file in format as the provided file:



6. Click on evaluate and the visualization will be provided. The output showcasing the predicted evaluation metrics and visualization should appear as below:



Relevant Notes:

- In our model we have made use of the Random Forest model as that is the model which has returned the highest accuracy when predicting the IMDB score.
- A few other reasons as to why we have chosen Random Forest model is as follows:
  1. Random Forest reduces overfitting, as it averages multiple decision trees.
  2. We observed some non-linear relationships between target column and feature columns. Random forest can check non-linear relationships between the target and features so it is a well rounded choice.
  3. This model can be utilized for both categorical and continuous variables so it can work in order to predict the genre as well as the score.
- Random Forest has various hyperparameters:
  1. n_estimators (Number of Trees)
  2. max_depth (Maximum Depth of Each Tree)
  3. min_samples_split (Minimum Number of Samples required to Split an Internal Node)
  4. min_samples_leaf (Minimum Number of Samples required at a leaf node)
  5. max_features (Number of features to consider when splitting)
- To tune the hyperparameters, we used trial and error method by manually increasing and decreasing each hyperparameter, and finalizing on the hyperparameters which gave us the least Mean Square Error and High R-squared. Some of the observations are mentioned below:

n_estimators	***max_depth***	***min_samples_split***	***min_samples_leaf***	***max_features***	***MSE***	***R^2***
100	None	2	1	1	0.008678	56.86%
500	None	2	1	1	0.008528	57.62%
1000	None	2	1	1	0.008472	57.89%
1500	None	2	1	1	0.008451	57.99%
2500	None	2	1	1	0.008447	58.01%
1000	None	2	1	5	0.006842	65.99%
2000	17	2	1	6	0.006822	66.09%

- We have returned the Mean-square and R^2 error along with a visualization of the predicted value against the actual value in order to estimate the working of the model.

Recommendations related to the problem statement:

Problem Statement:

Given a streaming platform such as Netflix, we want to identify which kinds of films go on to have great ratings. Based on things such as Run time, Genre, Actors, IMDB score and number of Votes, we want to see what we can use to predict movies with a good score.

Questions we are trying to answer:

- What determines this score? The Genre, Actors, Number of people who Voted/Watched it, the Run Time, etc?
- Which of these has the most value when it comes to determining this rating?
- What kind of impact does having different combinations of Genres have?

A. Why is this a significant problem?
There is an ocean of content out in the world when it comes to watchable material. Movies and TV Shows are just a few among the many ways people consume media. These attention capturing moving pictures aren't just there to entertain us but also teach us many things. However, the real world is one in which only the popular Shows and Movies come out on top. We cannot have fantastic media to watch without investors paying the bills and investors are drawn to profit. Also, the base desire of any consumer is to have not just one amazing show to consume, but to have multiple such shows readily available. Just from these points we can see that there is an intense need to be able to filter out the best possible stories and provide those to consumers as they are the most likely to be watched. After all many people pay attention to a Movies 'score' or 'rating' before they try to watch it!

B. What is the potential of your project to contribute to the problem domain and why is it crucial?
Our project can help contribute to the problem of 'selection' and 'prediction'. Our idea is to be able to use the readily available data to be able to choose from among sample movies the ones which would be the best rated and therefore most valued. Being able to choose this would involve looking at statistics of what makes a Movie good from the dataset we have chosen and using ML to be able to train a model to be able to pick out similar such shows. This would give a clearer idea on which films can be chosen and pushed to consumers.

Benefit to Users:

What users can learn from this is of course, the output predictions of the IMDB score. They can use this completed product to be able to gauge what the IMDB score a particular movie or multiple movies may be based on the input data provided.
Users can use this functionality to find out the score of new movies which may not have a score value within IMDB although the likelihood of that is quite low.
An interesting use though, is for Users to be able to input the idea for a movie (with its relevant details filled out in the input excel sheet) and then see how this movie will do when compared to all the other movies the model was trained on. In other words, this product can help users to estimate a score for new movies or works still in their creation phase. Users can also use this model as a different source of score prediction instead of the many and often unreliable sources out there. This may help benefit users when it comes to choosing between various movies they want to watch or when they want to be able gauge the score of a movie.
A few examples of the problems that can be solved are summarized as follows:

1. User needs to choose a particular movie to watch among multiple similar movies. He can use the model to gauge the IMDB score and pick out the best.
2. User needs to filter out what new Ideas can be pushed into production. Here, user can input each idea in format of movie to the model and estimate the score.
3. User wants to pick out newly released movies in the market and choose one among them to watch.
4. User wants to re-estimate a score of very old movies using the model and see how it would compare to its previous score.

How to Extend this project:

What users can learn from this is of course, the output predictions of the IMDB score. They can use this completed product to be able to gauge what the IMDB score a particular movie or multiple movies
This project was implemented by training the model on a couple thousand movies from a dataset obtained in Kaggle. This dataset was taken from the Database of Netflix. Multiple models were then implemented with the focus of being able to predict the IMDB score of the input movies and see how well it would do. Among the models used, we even implemented a few cases wherein we tried to predict the Genre as well. Here are some ways in which we can extend this project:

- Train the model using datasets from other major streaming services, such as HBOMax, Amazon Prime, Disney+, etc along with box office statistics in order to account for more variability and improve model performance.
- To try to predict other attributes of the movie or even predict not just one but 2 or more genres the movie might be having.
- Try and see what additional genre could be mixed into an idea to improve the likelihood of the IMDB score being higher.

- Take other kinds of input such as summarized paragraphs of the movie plot and use NLP or sentiment analysis to further hone predictions.
- Implement this kind of model not only on English movies but on movies from different languages and cultures.
- Extend this model to include other forms of media such as TV Series, Anime, Books, etc.

Other Avenues that can be explored:
Although the primary aim revolved around IMDB score prediction, other forms of prediction could also be useful. A few are as follows:

- Model to generate personalized picks of movies for Users based on what they watch.
- Model which would compare popular movies across different times and see how old movies would fare in todays day and age.
- A model which could find weakpoints in a movie pitch to allow for further improvement.

REFERENCES:

1. https://github.com/AlexTheAnalyst/PandasYouTubeSeries/blob/main/Pandas%20101%20-%20Data%20Cleaning%20in%20Pandas.ipynb
2. https://medium.com/analytics-vidhya/identifying-cleaning-and-replacing-outliers-titanic-dataset-20182a062893
3. https://www.geeksforgeeks.org/detect-and-remove-the-outliers-using-python/
4. https://www.adamsmith.haus/python/answers/how-to-remove-outliers-from-a-pandas-dataframe-in-python
5. https://www.itl.nist.gov/div898/handbook/eda/section1/eda11.html
6. https://www.analyticsvidhya.com/blog/2020/04/feature-scaling-machine-learning-normalization-standardization/
7. https://www.digitalocean.com/community/tutorials/normalize-data-in-python
8. https://www.kaggle.com/datasets/victorsoeiro/netflix-tv-shows-and-movies
9. https://scikit-learn.org/stable/modules/classes.html#regression-metrics
10. https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html
11. https://towardsdatascience.com/random-forest-regression-5f605132d19d
12. https://www.investopedia.com/terms/r/r-squared.asp
13. https://statisticsbyjim.com/regression/interpret-r-squared-regression/
14. https://scikit-learn.org/stable/modules/classes.html#module-sklearn.metrics
15. https://scikit-learn.org/stable/model_selection.html
16. https://scikit-learn.org/stable/modules/preprocessing.html
17. https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html
18. https://scikit- learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html
19. https://scikit-learn.org/stable/modules/tree.html
20. https://www.geeksforgeeks.org/decision-tree-implementation-python/#
21. https://towardsdatascience.com/implementing-a-decision-tree-from-scratch- f5358ff9c4bb
22. https://statisticsbyjim.com/regression/interpret-r-squared-regression/
23. https://www.ibm.com/topics/linear-regression
24. https://scikit- learn.org/stable/modules/generated/sklearn.metrics.confusion_matrix.html
25. https://www.spiceworks.com/tech/artificial-intelligence/articles/what-is-linear- regression/

Peer Evaluation Form for Final Group Work

CSE 587B

Group member 1: Vivian Vincent Dmello

Group member 2: Chaitanya Deepak Yeole

Group member 3: Vangala Eshwar Sai Prithvi Kiran

Evaluation Criteria	Group member 1	Group member 2	Group member 3
How efectively did your group mate work with you?	5	5	5
Contribution in writing the report	5	5	5
Demonstrates a cooperative and supportive attitude.	5	5	5
Contributes significantly to the success of the project.	5	5	5
TOTAL	20	20	20

Also please state the overall contribution of your teammate in percentage below, with total of all the three members accounting for 100% (33.33+33.33+33.33 ~ 100%):

Group member 1: 33.33%

Group member 2: 33.33%

Group member 3: 33.33%