Logo, company name

Description automatically generated

[**COMP8060\_27330 Scientific**](https://cit.instructure.com/courses/67674) **Programming in Python Project Report**

**Submitted By:**

**R00195427**

**Job Thomas Thekkekara**

1. **DATA CLEANING**

The data set I obtained was an IMDB movie dataset consisting of 5043 records and 28 fields. The data was read into a Pandas data frame first. Prescence of duplicate records were checked using duplicated() in Pandas and the duplicate records were dropped using drop\_duplicates() by keeping only the first occurrence. All the data types were checked using info(). Then the count of null values were checked using isnull() and sum(). It was observed that there were several null values present in the data.

Most of the null values in categorical columns that were not used in the analysis were replaced with the most frequently occurring value in that column. That is, Mode. This was found out by using value\_counts () which gives the number of occurrences of a particular value in a column. idxmax() was used along with this to get the most frequently occurring value in that particular field. Missing values were replaced with mode using replace()

Null values in numerical columns that were not used in main analysis was replaced with average value by using mean() and replace(). This is because average provides a good representation of the overall distribution, and it is a standard way of imputation

Null values contained in some important fields for analysis such as Director Name, Gross etc were dropped using dropna(). Dropping them seemed appropriate as taking mean or mode will cause inaccuracies. Last character in the movie title was stripped because it contained an invalid value.

All the data cleaning operations were combined into one single function data\_cleaning() for easier execution. The function gets an uncleaned data frame, and it returns a fully cleaned data frame. During this process I learned what all are the general steps I need to check when I get a dataset such as checking data types, shape of the data, invalid values in the data(movie title) etc and how to check for duplicate records and remove them. I also learned how to check for null values and various techniques to impute or drop null values. I learned that I have to give inplace=True in replace() and dropna() so that my changes get reflected to original data frame and does not return a copy of the object. Also reset\_index() has to be called after dropping records to maintain correct index values in the data frame. The steps I followed can be used as a reference for cleaning up any dataset in the future.

1. **Most Successful Directors or Actors**

I have defined a function best\_director\_actor() that takes in the cleaned data frame and repeatedly asks a user to choose between top directors and actors by using a while loop construct. I have given a clear\_output() on all while loops in this project so that the previous outputs does not hinder the user experience. The user input choices are given in a try block so that the program can check for errors. I have given a ValueError exception so that the program does not throw any error if the user gives an invalid choice. If the user presses 1, it goes to top\_directors() else if user presses 2, it goes to top\_actors().

top\_directors() function repeatedly asks the user for the top number of directors the user wants to see. This is done using while loop, try and except blocks so that user cannot enter an invalid value. If the user enters a value n between 1 and maximum number of directors in the data, it displays the result using a visualization.

Chart, bar chart

Description automatically generated

The above figure is displayed if the user enters for top 5 directors. Sum of gross earnings from movies is used to calculate the ranking of directors. It can be seen that Steven Spielberg is the top director with the maximum gross earnings (total amount of income earned over a period of time) 4\*e^9. This is followed by Michael Bay with 2.25\*e^9. 5th position is occupied by Christopher Nolan with a gross earning of 1.75\*e^9. It is observed that gross of Steven Spielberg movies is way above others and the movies he directed has earned double that of 2nd position.

top\_actors() is implemented in the same way as directors. It displays top actors specified by the user with the following visualization.

Chart, bar chart

Description automatically generated

The above figure is displayed if the user enters for top 7actors. Actor\_1\_name field is used for this as he is the lead actor of a movie. Sum of gross earnings from all movies the actor acted is used to calculate the ranking of actors. It is seen that Harrison Ford ranks 1st with a gross earning of around 3.5\*e^9. This is followed closely by Tom Hanks & Johny Depp with an approximate gross earnings 3.25\*e^9. 7th position is occupied by Robert Downey Jr with a gross earning of nearly 2,5\*e^9. There is no clear front runner in actors like the directors as the gross earnings does not differ a lot between actors.

It is also observed that all the top directors and actors are from Hollywood. This might be due to the fact that most big budget films are from Hollywood. Also, it could be because this IMDB dataset may mostly contain information about Hollywood movies. Also, it is important to note that here gross earnings are summed up and not taken an average for each director or actor. Hence a single movie with a high gross can influence the order of top directors and actors.

While implementing this I learned how to use nunique() to get unique values in a dataset, groupby() on pandas data frame to group categorical variables based on values and nlargest() to get the top n records from a data frame based on a field. I also learned to plot horizontal barplot using Seaborn package and to give figure size, title, and axes labels. The information I learned here will become extremely useful to group data, to find largest or smallest from a dataset and to plot them visually in future. I also learned to use try and except blocks of code in python for basic error checking and to print out exception error messages. This will become really useful for implementing and debugging complex programs in later stages.

1. **Film Comparison**

A film\_comparison() was implemented which takes in the data frame and repeatedly asks the user for to enter two film names. This is again achieved by using while loop, try and except blocks of code. If the user successfully enters two movies that are in the dataset, they are given a choice of 1. IMDB Scores, 2. Gross Earnings and 3. Movie Facebook like to compare the movies they entered. IF else control structure is used to display corresponding visualizations. If the user enters 2 movie names successfully and chooses option 1. IMDB scores, following chart is displayed.

Chart, bar chart

Description automatically generated

When comparing movies Avatar and Spectre on IMDB ratings, Avatar tops with a score of 7.9 while Spectre gets a score of 6.8. Hence with IMDB Score, Avatar is a better movie.

If the user chooses option 2 to compare films based on gross earnings, following chart is displayed.

Chart, bar chart

Description automatically generated

On comparing, Avatar’s gross earnings (7.6\*e^8) is more than 3 times of that of Spectre. This clearly shows that Avatar collected more earnings through global theatre releases advertisements, satellite rights etc

If the user selects option 3 to compare two movies based on movie Facebook likes, following chart is shown.

Chart, bar chart

Description automatically generated

Surprisingly and quite contrary to other options, here Spectre tops with 8000 Facebook likes while avatar gets around 3000 likes. Hence it can be concluded that Avatar has got more IMDB Score and gross earnings when compared to Spectre, but Spectre has got wider audience appreciation with Facebook likes. As the difference between Facebook likes is quite high and contrary to other options, the underlying numbers have to be rechecked for further confirmation.

On implementing this, I learned how to compare same information on different contexts to derive an informed conclusion. I also learned to use IF else control statement in combination with user inputs, while loop, try and except blocks of code.

1. **Distribution of Gross Earnings**

gross\_distribution() function is implemented to analyse the minimum, maximum and average gross of movies over the user specified years. It repeatedly asks the user to enter a starting year and an ending year. It checks for the year entered in the list of years present in the dataset. If both the years are present, it shows the following time series graph.

Chart, line chart

Description automatically generatedThe above figure is displayed by the program if the user inputs 1998 and 2014 as start year and end year. It is observed that the maximum gross fluctuates a lot over the years. It started at 2.2\*e^8 gross in 1998 and peaked in 2009 with a gross of 7.5\*e^8. But it then came down to 3.5\*e^8 in 2014. But generally, an upward trend is observed over the years. This could be due to the rise of technologies and more advertisements on movies enabling people all around the world to watch more movies. Such fluctuations in maximum gross of movies are possible as one superhit movie can single handily change the maximum gross in a year.

It is interesting to note that the average gross of movies has remained constant over the years (0.5\*e^8) with a slight uptrend. The minimum gross of movies has not shifted at all. This is possible as there would be some movie in each year that earns very small.

While implementing this, I learned how to do identify a trend and what could have been the reason behind it for a particular trend over the years. I learned how to analyse a sequence of data points collected over a period of time and what it says about the data according to different categories. I also learned that maximum and minimum values alone do not give the true picture as it can be highly influenced by outliers.

Time series analysis and pattern recognition will help me a lot in future. Understanding past trends to predict future is one of the major applications in data science.

Also, technically I learnt how to combine data frames, to create time series plots and to print legends and gridlines on plots.

1. **Genre Analysis**

Imdb\_genre() displays all the available genres in the dataset and repeatedly asks the user to input one genre from the shown options. Upon user input the program will show the mean imdb score for that respective genre.

A picture containing application

Description automatically generated

The figure above shows that the user gave action genre as input, and it calculated the mean imdb score for all movies in that genre as 6.28. This means that out of 10, majority of action movies had only average rating (imdb score).

Each movie can have multiple genres. The data for field ‘genre’ was obtained as a pipe separated one, containing multiple genres. Hence it had to be pre-processed into lists using split() and for loops. Contains () was used to search for a user input genre in data frame and mean() was used for mean calculation.

I learned how to pre-process multiple values in a single cell and to search for a value in a data frame to do operations based on that value. This will become extremely useful when dealing with semi structured and unstructured data in future.

1. **Relation with IMDB Score**

imdb\_score() is implemented to generate graphs and figures that helps to establish relationship between imdb score and other numerical variables. Establishing the strength of relationship between variables is necessary to build models with good prediction accuracy. Proper visualizations help a lot in achieving this.

Correlation coefficient(r) is used to measure the strength of potential linear relationship between two variables. It is covariance divided by the standard deviation of x and y.

Correlation Coefficient varies only between -1 and 1. Since this range remains same for any relationship, interpretation of strength between variables become easy and standardised.

If r=1🡪 Perfect positive correlation between two variable(x&y)

If r=-1🡪 Perfect negative correlation between two variables (x&y)

If r=0🡪 No correlation between two variables(x&y)

The function gives a correlation graph and a heatmap of imdb\_score vs rest of the numerical variables.

Chart

Description automatically generated

Chart, bar chart

Description automatically generated

On observing the above correlation plots, it is seen that most of the other numerical variables have weak to no correlation with IMDB\_Score. But out of all the numerical fields, num\_voted\_users(0.41),num\_critic\_for\_reviews&duration(0.3),num\_user\_for\_reviews (0.29), movie\_facebook\_likes(0.24) and gross(0.19) have moderate to weak positive correlation with IMDB\_Score. That is, as the number of voted users or number of critics for reviews increases imdb\_score also increases due to this positive correlation. Hence these variables can be considered for creating a model that predicts imdb\_score with reasonable level of accuracy.

An lmplot() using seaborn is used below to show an example.

Chart, scatter chart

Description automatically generated

The regression line shows the positive correlation between number of critic for reviews and imdb\_score. As the number increases, imdb\_score also increases. But it is not strongly correlated as the points are not tightly packed around the line, but rather it is scattered.

Graphical user interface, application, Word

Description automatically generated

The above shown scatter plots were generated by the function imdb\_scatter\_plots (). Each numerical variable is plotted against imdb\_score. It can be seen that some of them shows a pattern indicating linear relationship while some other variables such as Actor1 facebook like, actor 2 facebook like, aspect ratio, cast total facebook like etc shows no pattern or trend as they have no relationship with imdb\_score.

On implementing this, I learned about correlation coefficient, how to interpret the values and to measure the strength of relationship between variables. I also learned to plot correlation matrix, lmplot and to plot multiple scatterplots in a single figure by specifying number of subplots.

Plotting correlation matrix and scatterplots will enable me to understand relationship between variables in order to effectively select best variables for building machine learning models with reasonable level of accuracy.

**7. Conclusion**

First IMDB movie data set was cleaned for processing. Then different functions were implemented to check top directors and actors based on gross, to compare 2 different movies based on 3 criterions, to do trend analysis on minimum, maximum and average gross over the years, to do genre analysis and finally to show relationship between imdb score and rest of the numerical variables.