




 **omondiimmanuel / dsc-phase-1-project-v2-4** Public


forked from [learn-co-curriculum/dsc-phase-1-project-v2-4](#)


 View license


 0 stars  439 forks


 Star


 Watch


 Code


 Pull requests


 Actions


 Projects


 Wiki

 Security



 Insights



 Settings

 master ▾




This branch is [6 commits ahead](#) of learn-co-curriculum:master.

 Contribute ▾  Sync fork ▾

 **omondiimmanuel** ... 1 minute ago 

[View code](#)

 README.md 

FINAL PRJECT SUBMISSION DETAILS

Student name: OMONDI IMMANUEL OCHIENG Student pace: part time Scheduled project
review date/time: 16/04/2023 Instructor name: Noah kandie

TITLE: ANALYSIS TO FIND OUT POTENTIAL MOVIES FOR MICROSOFT NEWS STUDIO.

#BUSINESS UNDERSTANDING Microsoft had been observing how major companies had been creating movie content and decided to come up with their own movie studio. To do this however, they required an in-depth analysis of which movies had been performing well in the box office. This notebook will bring all the data sources together on movies for further analysis.

DATA UNDERSTANDING

In order for there to be accurate, easier and proper data analysis, for this project i will make use of database files and some csv files stored on my local computer. My reason for using csv and database file is that they are easier to read and open using inbuilt python libraries. They are also not as cumbersome as json files.

DATA SOURCES:

bom.movie_gross.csv.gz rt.movie_info.tsv.gz rt.reviews.tsv.gz tmdb.movies.csv.gz
tn.movie_budgets.csv.gz

DATA PREPARATION

The steps i would follow during data gathering and preparation are as follows:

1. importing of necessary libraries in order to read the data files. the libraries i will be using are pandas and matplotlib
2. python import pandas as pd importing of pandas will enable me read directly from the csv files using pandas read csv files function.
3. import matplotlib.pyplot as plt
4. Set %matplotlib inline i also include matplotlib inline to enable inline plotting on the notebook
5. import style import warnings so as to get rid of any warnings
6. importing of matplotlib will aid in illustrating of my data findings. Matplotlib comes with an attribute called pyplot for easier plotting.
7. The next step of my data preparation process would be to open and read the necessary files and assign them to variables.

The first file of interest is (imdb.movies.csv.gz) We open it using the code: (tmdb=
pd.read_csv('tmdb.movies.csv.gz') tmdb.head(18)

This code works by reading files using the pandas attribute read_csv to access the information

tmdb.head(8) function then displays the first 8 rows of the dataset.

Data analysis

Some of the movies encountered in the first data set:

1. Harry Potter and the Deathly Hallows, Date: 2010-11-19
2. How to Train Your Dragon, date:2010-03-26
3. Iron Man , date: 2010-05-07
4. Toy Story , date: 1995-11-22
5. [28, 878, 12] 27205 en Inception 27.920 2010-07-16 Inception 8.3
6. Percy Jackson & the Olympians: The Lightning T

tmdb.info()

the code gives us a general overview of the structure of the dataset.

It is a pandasdataframe with 9 columns by 26517 rows

tmdb.columns getting all columns of the dataset

tmdb.tail() This command gives back the last 5 rows of the dataset From there we can now see the dataset has 26516 not 26517 rows

data cleaning

The next step is to perform some data cleaning. The data_set has some null values in some columns and some rows are displaying unreal values

tmdb.isnull() This code identifies null values in the dataset tmdb.shape checks the shape of the dataset, it has 26517 columns by 9 rows.

tmdb.dropna(how ='all').shape This code checks for null values in all rows, drops them if null, since the shape remains the same, it proves that the dataset has no rows reading null

tmdb.isnull().sum() getting count of all null values to confirm they have been dropped

tmdb.shape this code shows there has been a reduction in number of rows showing they have been dropped. (26381, 10)

```
tmdb= tmdb.loc[tmdb['vote_count'] > 10000]
```

in this cell we selected the rows whose (VOTE_COUNT) was greater than 10,000

tmdb.shape judging by this output, we can see definitely some rows were dropped. (72, 10)

tmdb = tmdb['vote_count'].sort_values(ascending= True) This code sorts all the values in the VOTE_COUNT column in ascending order

tmdb.describe() This code gives a return of mathematical operations performed in the dataset

tmdb = tmdb.loc[tmdb['vote_average'] > 7.7] In this code we are selecting only the rows with a vote_average greater than 7.7

tmdb.drop(tmdb.columns[[0]], axis= 1) lets now drop the unwanted column

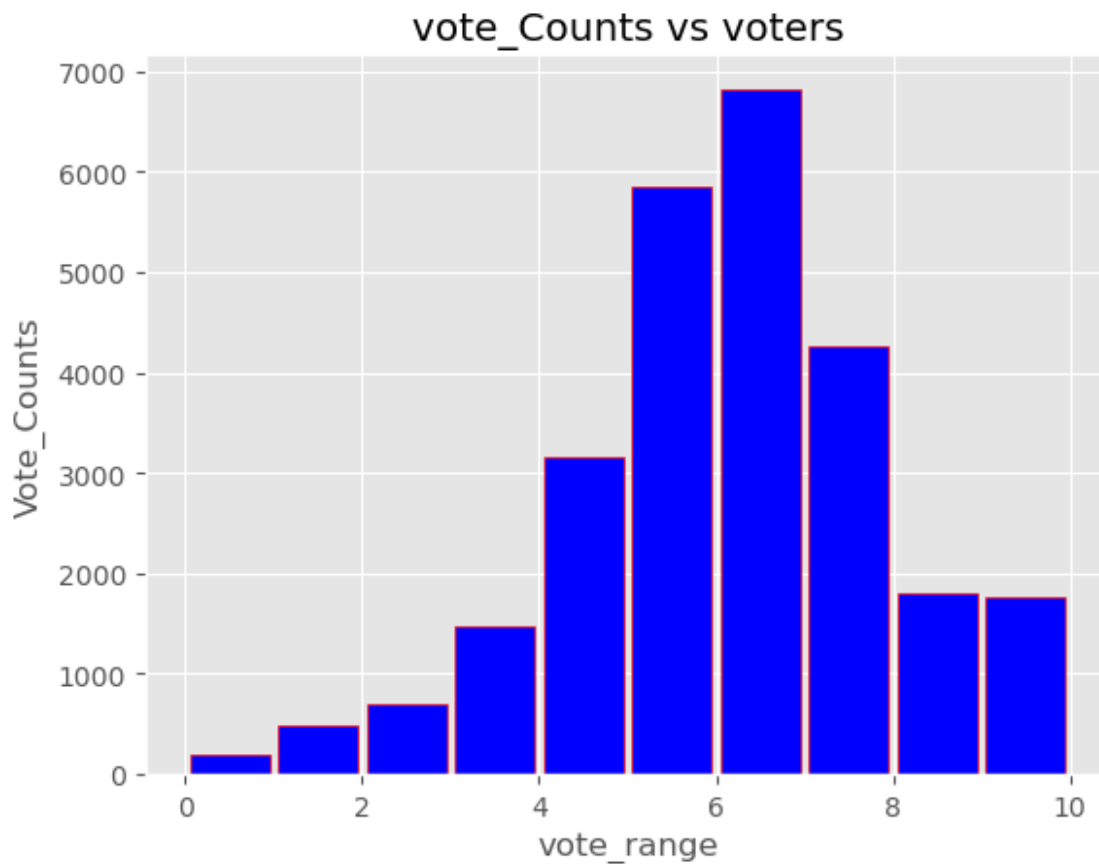
lets now select the rows containing vote_count > 500

```
tmdb= tmdb.loc[tmdb['vote_count'] > 500 ] tmdb.head(8)
```

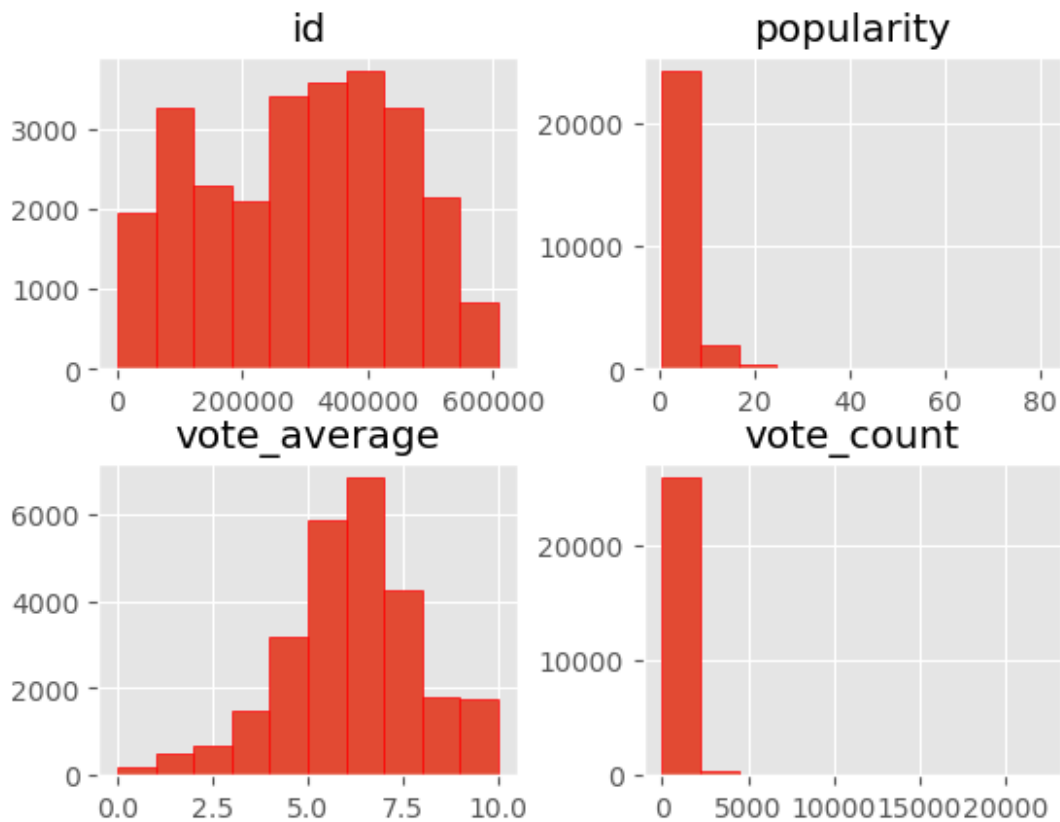
Data analysis

Next i will plot a histogram using vote_count columns and popularity columns to compare which movies are most popular. The function for plotting histogram is illustrated as follows:

```
plt.style.use('ggplot') plt.hist(votes, bins= 10, edgecolor= 'red', color= 'blue', rwidth= 0.9)
plt.title('vote_Counts vs voters') plt.xlabel('vote_range') plt.ylabel('Vote_Counts')
```



i also plotted a third histogram `tmdb.hist(bins= 10, edgecolor= 'red')`



Next i plotted a scatterplot of the dataset `tmdb.plot.scatter(x= 'vote_average', y= 'vote_count', s= 0.5)` [image](#)

Lets now try comparing movie data of a second dataset

The dataset is tn.movie_budgets.csv.gz lets proceed with opening it `rt= pd.read_csv('tn.movie_budgets.csv.gz', index_col= 0)`

We open it using the `pd.read` function.

Lets now check the dataframe for general info of the dataset `#rt.info()`

`#rt.tail()` getting last few rows of the dataset

checking for null values

`rt.isnull().head()`

lets now get a mathematical count of the dataset `#rt.describe()`

#Data cleaning lets start by converting the columns with dollar signs to integers

```
#rt['production_budget']= rt['production_budget'].str.replace('$','').str.replace(',','')
rt['production_budget']= pd.to_numeric(rt['production_budget'])
```

lets do the same for the remainig columns

```
#rt['domestic_gross']= rt['domestic_gross'].str.replace('$','').str.replace(',','') rt['domestic_gross']=
pd.to_numeric(rt['domestic_gross'])
```

```
#rt['worldwide_gross']= rt['worldwide_gross'].str.replace('$','').str.replace(',','')
rt['worldwide_gross']= pd.to_numeric(rt['worldwide_gross'])
```

Lets get ing rid of rows where production budget is not equal to 0

`rt= rt[rt['production_budget'] != 0]`

getting rid of rows where domestic_gross is not eaqual to 0

`rt= rt[rt['domestic_gross']!= 0]`

getting rid of rows where worldwide gross is not equal to 0

`rt= rt[rt['worldwide_gross'] != 0]`

checking for null values `#rt.isnull().value_counts`

Lets now select only rows where production_budget is greater than/equal to 500000

```
rt= rt[rt['production_budget'] >= 500000 ]
```

We will do the same for domestic_gross and worldwide_gross columns.

```
#rt= rt[rt['domestic_gross'] >= 500000] #rt= rt[rt['worldwide_gross'] >= 500000] rt.head(8)
```

data visualization

In these section we are going to visualize the data and performe various comparisons

Lets start by gettin an overall plot #rt.plot() [image](#)

Lets now generate subplots #rt.plot(title= 'comparison', subplots= True, figsize= (10,15)) [image](#)

Lets now work with the last dataset

opening the file #bm= pd.read_csv('bom.movie_gross.csv.gz', index_col= 0) bm.head()

Getting general info on the dataset #bm.info

Checking the dataset columns #bm.columns

getting last few rows of data #bm.tail()

Getting mathematical counts of the dataset bm.describe()

data cleaning

lets start by checking null values in the dataset

```
#bm.isnull()
```

lets count the number of null values #bm.isnull().sum()

The output shows we have null values in 3 columns

lets proceed to drop them #bm= bm.dropna()

```
#bm.isnull().sum() Counting to check for null values
```

lets now sort the dataframe by the column(years) #bm = bm.sort_values('year')

data visualization

Lets start by a basic plot

```
#bm.plot() plt.title('comparison')
```

[image](#)

Lets proceed plotting a bar graph

```
#plt.bar(years, domestic) plt.ylabel('domestic_gross') plt.xlabel('years') plt.title('domestic gross over the years')
```

[image](#)

Lets proceed with plotting another bar graph.

```
#plot.bar(title= 'Bm_Movie_Production', subplots= True, figsize= (21,20))
```

[image](#)

Final bar graph plot In this plot we selected the first 10 values of the dataset

```
#bm.head(10).plot( 'studio', 'domestic_gross', kind='bar', color= 'g', figsize= (15,6)) plt.title('movie by domestic_gross comparison')
```

[image](#)

I also plotted a scatter plot

```
#bm.head(10).plot( 'domestic_gross', 'foreign_gross', kind='scatter', color= 'r', figsize= (10,5)) plt.title(' domestic_gross by foreign_gross comparison')
```

[image](#)

Releases

No releases published

[Create a new release](#)

Packages

No packages published

[Publish your first package](#)

Languages

● Jupyter Notebook 100.0%