**SQL/NoSQL Final Project**

**Data Analysis TMBD 5000 Movie**

**Siddhesh Mirjankar Jack Murakami Shanmukh Pinna**

**Table of Contents:**

[**TMDB Movie data set** 1](#_Toc14812859)

[About the data set: 1](#_Toc14812860)

[Mongo DB 2](#_Toc14812861)

[Creation of Database in Mongo DB: 2](#_Toc14812862)

[Data Import: 3](#_Toc14812863)

[Verifying the records after import 3](#_Toc14812864)

[Connecting to Mongo DB from Python: 3](#_Toc14812865)

[Exploratory Data Analysis 4](#_Toc14812866)

[Data Cleansing: 4](#_Toc14812867)

[Verification of Null values: 4](#_Toc14812868)

[Verification of data types 5](#_Toc14812869)

[Trend analysis: 5](#_Toc14812870)

[Box Plots 7](#_Toc14812871)

[**Analysis on Budget, Revenue and Popularity with Genre** 7](#_Toc14812872)

[Scatter plots: 8](#_Toc14812873)

[Correlation heatmap 10](#_Toc14812874)

[**Visualizations through Tableau** 11](#_Toc14812875)

[**Linear Regression:** 12](#_Toc14812876)

[Multiple linear regression equation 12](#_Toc14812877)

[Linear regression Analysis: 13](#_Toc14812878)

[Linear regression conclusion: 13](#_Toc14812879)

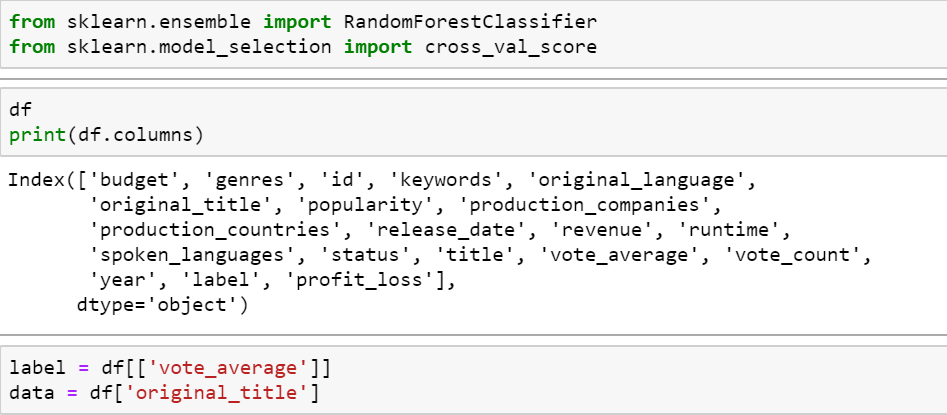
[**Naïve Bayes** 13](#_Toc14812880)

[Naive Bayes implementation 14](#_Toc14812881)

[Naive Bayes conclusion: 14](#_Toc14812882)

[**Random Forest Algorithm:** 14](#_Toc14812883)

[**Random Forest Implementation:** 14](#_Toc14812884)

[**** 15](#_Toc14812885)

[**KMeans Algorithm:** 16](#_Toc14812886)

[Elbow Point method: 16](#_Toc14812887)

[KMeans Implementation 16](#_Toc14812888)

[**Agglomerative Hierarchical Clustering Algorithm:** 17](#_Toc14812889)

[Agglomerative Hierarchical Clustering Implementation: 18](#_Toc14812890)

[**Conclusion** 19](#_Toc14812891)

[**Contributions** 19](#_Toc14812892)

# 

# **TMDB Movie data set**

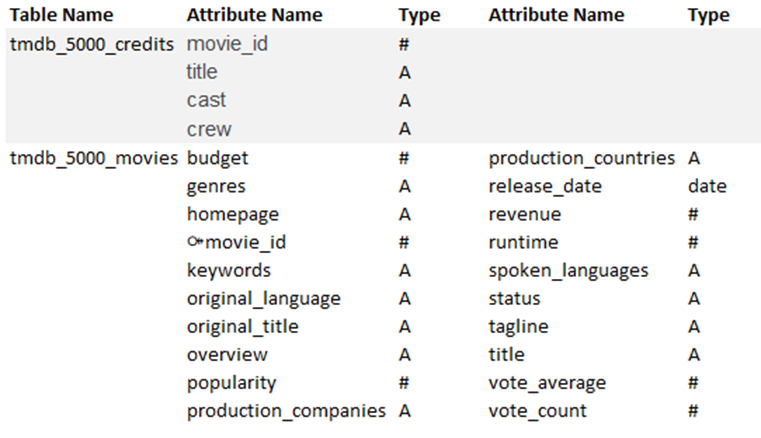
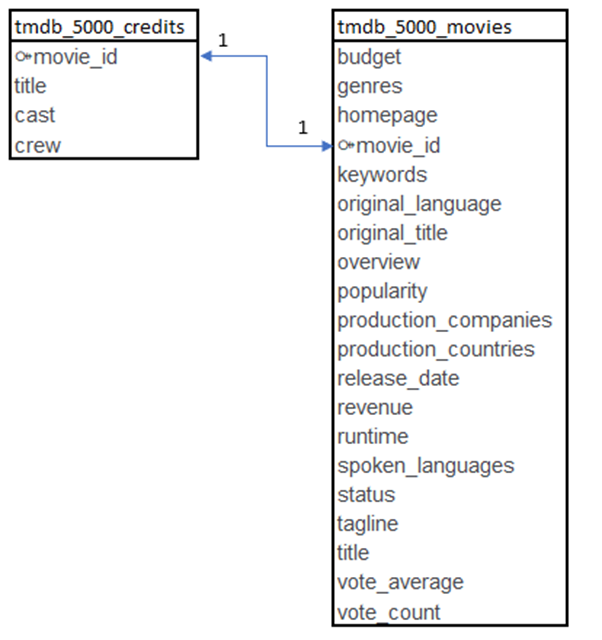
Data set has been taken from kaggle

<https://www.kaggle.com/tmdb/tmdb-movie-metadata#tmdb_5000_movies.csv>

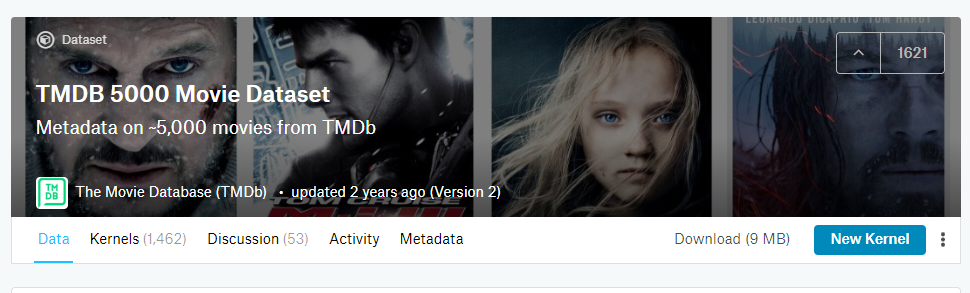
# About the data set:

The Movie database is a metadata of movies released from year 1916 to 2016

The data consists of 2 main tables that are joined by the primary key called “movie\_id.” The credits table (tmdb\_5000\_credits) is descriptive in nature with several fields consisting of lengthy json. For the project, the focus is primarily on the data delivered through the movie table (tmdb\_5000\_movies) which consists of several categorical and numerical data. We have commenced with an exploration of the data, we then ran analysis on the data in order to give movie executives useful data for predicting the profitability of the movie.



The TMDB 5000 database on Kaggle.com is the basis for our data. We used MongoDB to create our database. The compatibility of mongo with python, with the data analysis required are the reasons for the selection.



# Mongo DB

## Creation of Database in Mongo DB:

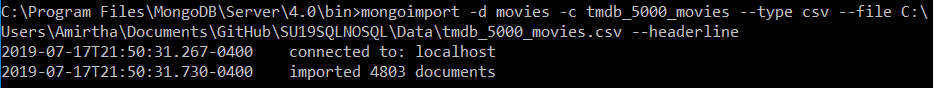






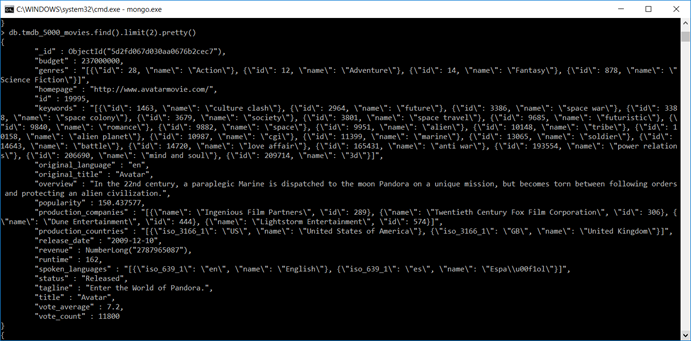
## Data Import:

**Script**: C:\Program Files\MongoDB\Server\4.0\bin>mongoimport -d movies -c tmdb\_5000\_movies --type csv --file C:\Users\Amirtha\Documents\GitHub\SU19SQLNOSQL\Data\tmdb\_5000\_movies.csv –headerline



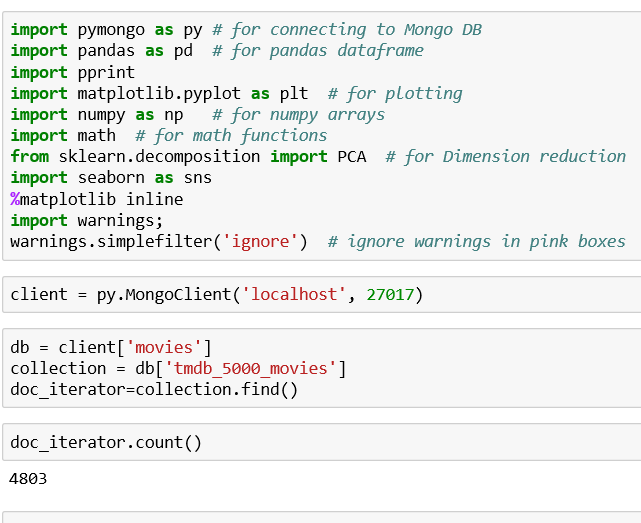
## Verifying the records after import

**Script:**db.tmdb\_5000\_movies.find().limit(2).pretty()



## Connecting to Mongo DB from Python:

We have used PyMongo module from python to connect to Mongo DB



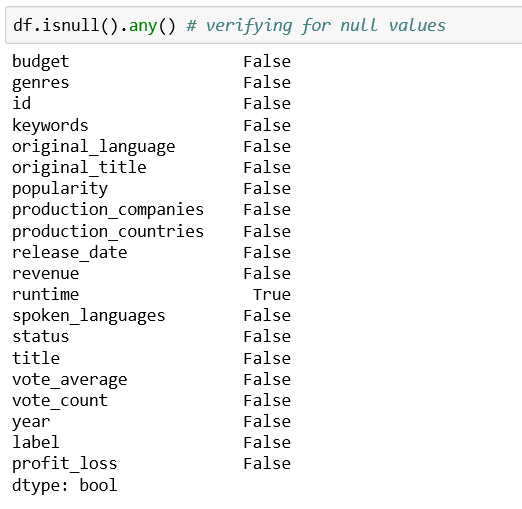
# Exploratory Data Analysis

## Data Cleansing:

* Removed 1000 records with missing budget details
* Data type of below elements are changed
  + release date changed to date to represent correct format
  + Run time changed to integer from string
* Year column derived from date for analysis
* First genre is derived from genre column

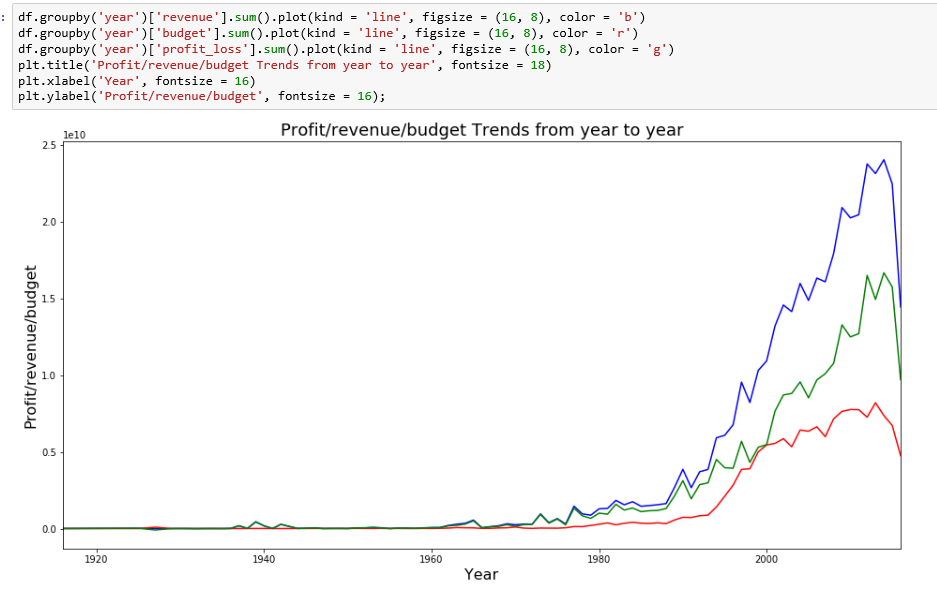
## Verification of Null values:

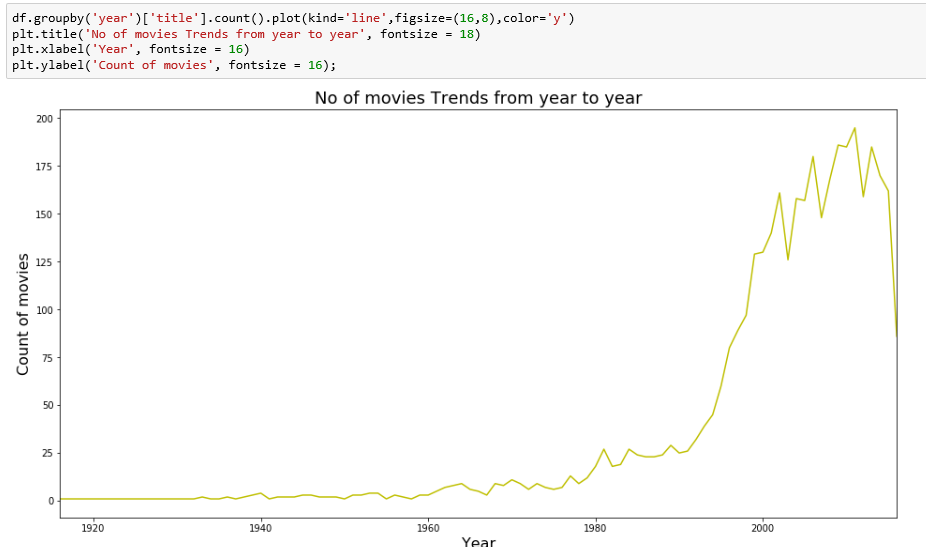
Data set does not have any null values



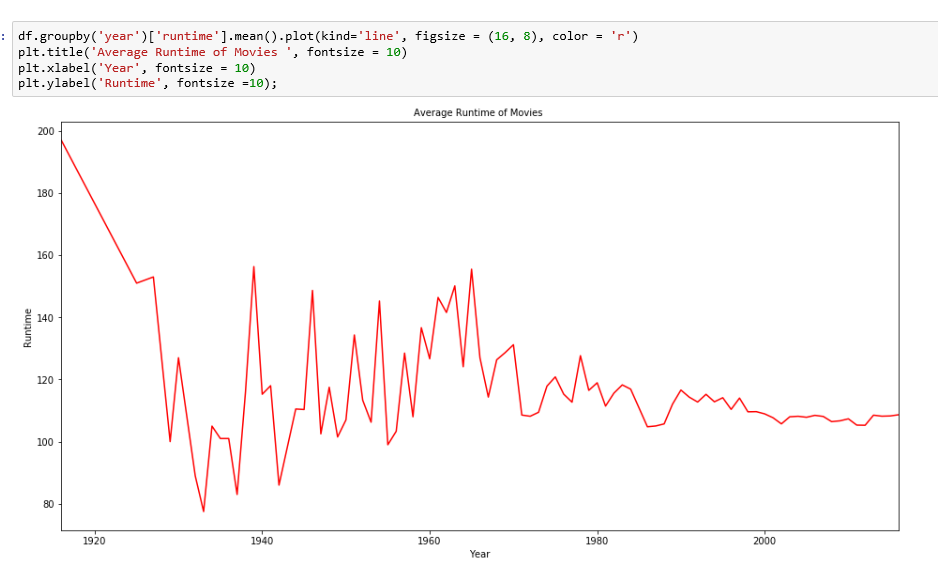
## Verification of data types

## Trend analysis:





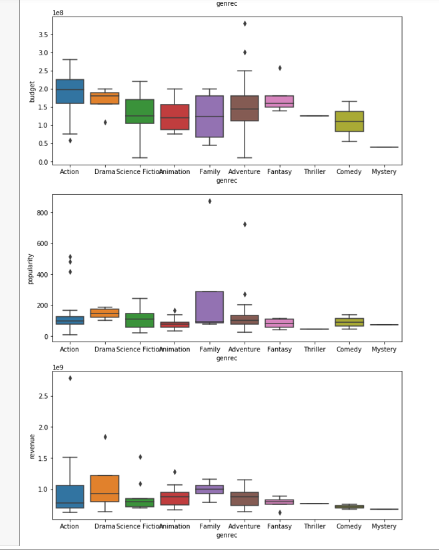
* Number of movies per year, budget, revenue and profitability has remained flat from 1916 to 1975 however these have increased considerably over the last 40 years
* Revenue generated from movies is much higher than budget across the periods
* Slight reduction in no. of movies in the year 2016, can be due to missing collection of movies in the last year of data collection



* Average Run time per movie per year was fluctuating drastically during 1916 to 1980
* From 1980 average run time is maintained between 100 and 120 minutes

## Box Plots

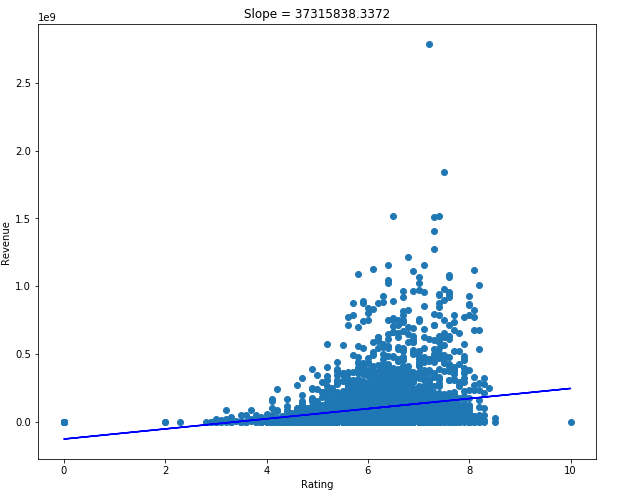
### **Analysis on Budget, Revenue and Popularity with Genre**



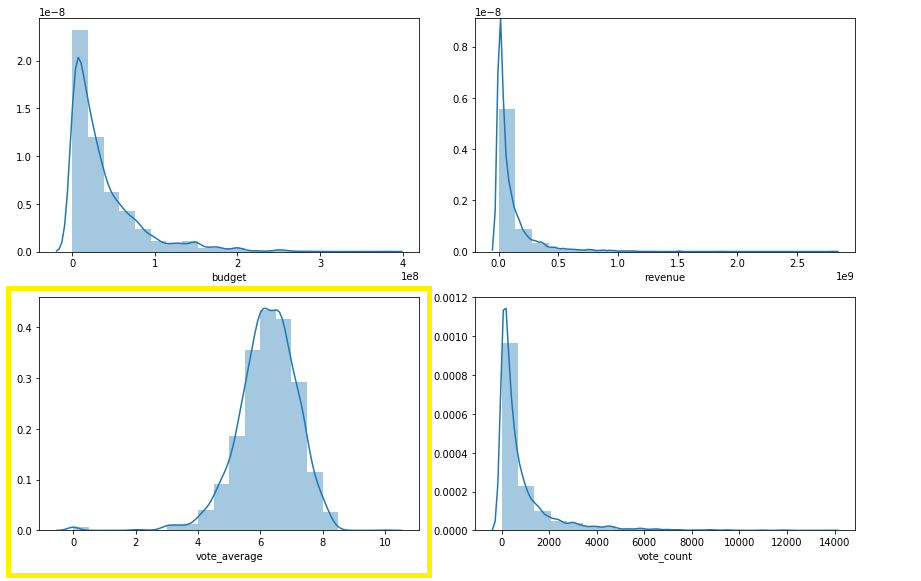
* More Budget is spent on Action and drama as compared to other genre
* Budget for family movies has wide range
* Revenue is more from Drama and family
* Popularity is high for family genre movies compared to others genre

## Scatter plots:

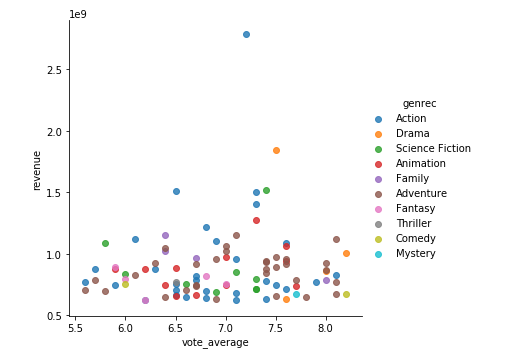
We started with a density plot of the movie user rating against revenue and saw a good clustering of the data, but the linearity wasn’t there. We noticed that the slope of the correlation is quite flat, and the data was not as linear as you would like to see if trying to make a reliable prediction using the data

.

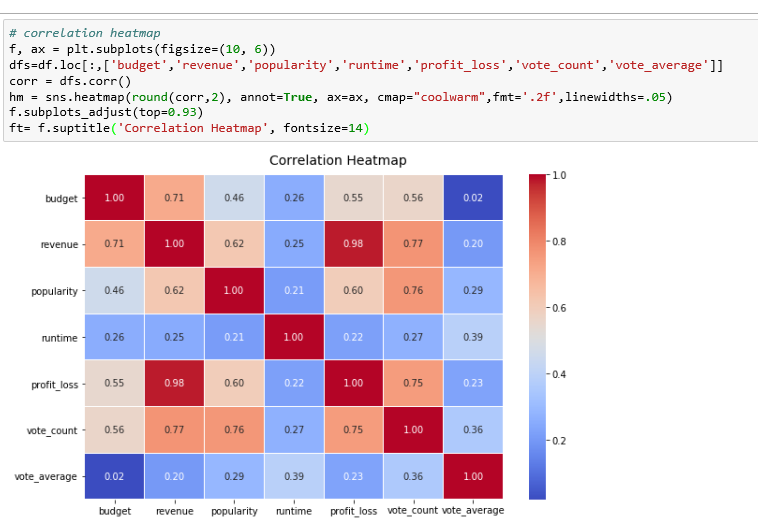
Looking at a histogram of the 4 major numerical fields, we saw 3 left heavy distributions and one (for vote average) that looked quite a bit more normal than the rest. This helped us see that vote average will probably be useful later in our analysis.



A breakdown of the top 100 movies showed that visually, one genre isn’t obviously better at generating revenue than the others. However, the action genre has many of the top generators so a loose analysis suggested that if a producer wanted to hit a box-office home run, an action movie would want to be a part of the equation.

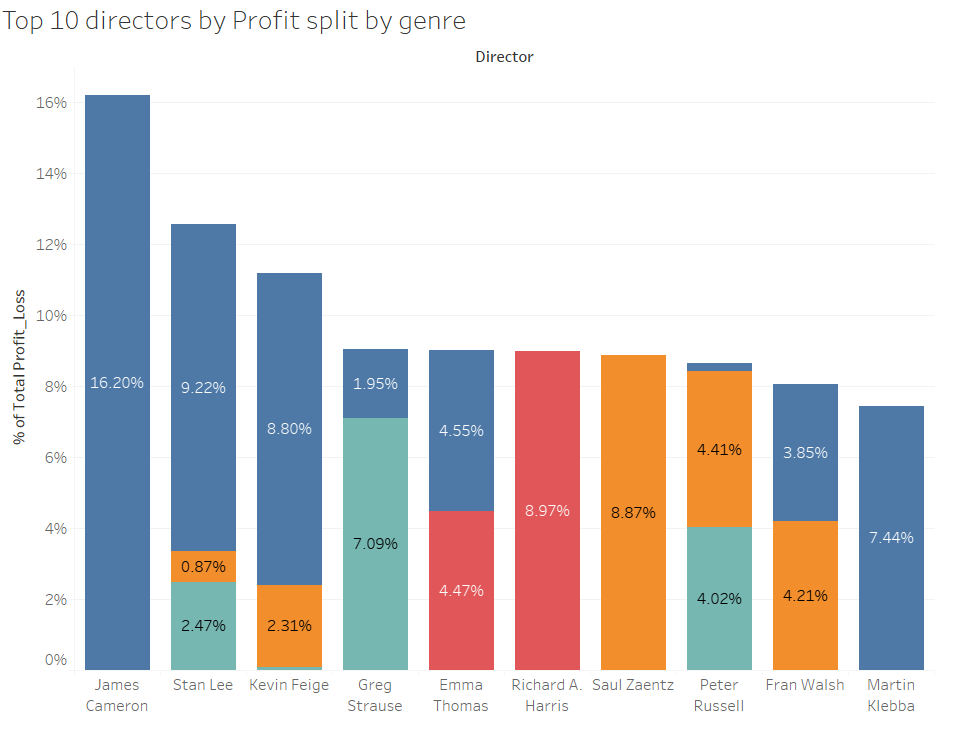


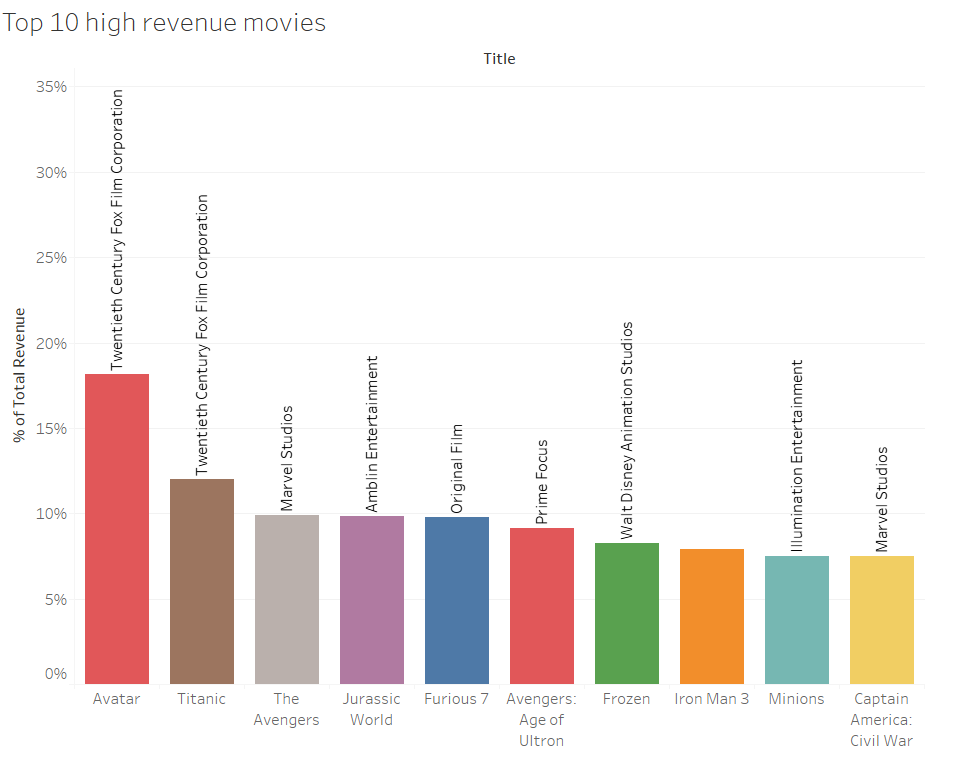
## Correlation heatmap

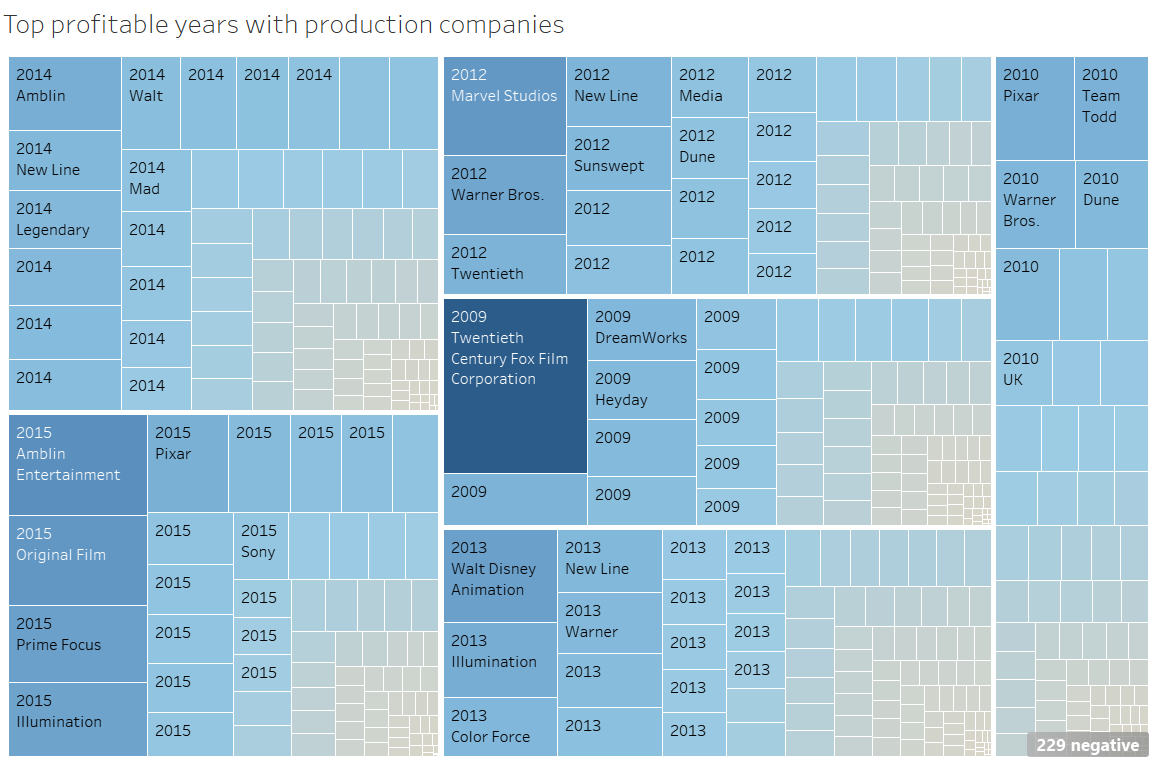


* Revenue and profit were having high correlation with almost 1
* Revenue and Budget had reasonably high correlation with 0.72
* Vote count was having reasonably high correlation with revenue, popularity and profit and loss
* Vote average ,Run time and other features were either not correlated or very low correlated since all values are below 0.4
* Popularity and revenue, Popularity and vote count were mildly correlated

# **Visualizations through Tableau**







* 2014 is more profitable followed by 2015 & 2012 compared to other years.

# **Linear Regression:**

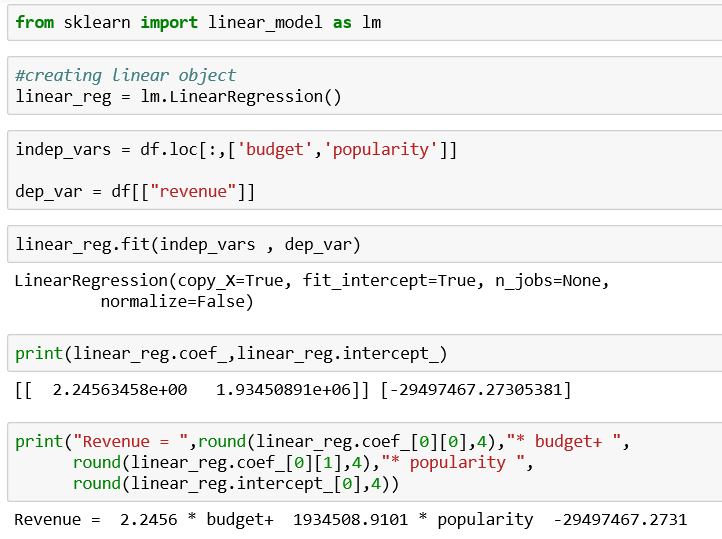
* We used a Linear model from Scikit learn package for deriving the linear regression equation Y=mx+b
* Revenue columns were considered as dependent variable
* Budget, popularity was considered as independent variables

## Multiple linear regression equation

Revenue = b1 Budget*+ b2* Popularity + b0

**Steps**

* Created Linear regression model with fit intercept as true and normalize as false
* Trained linear regression model with dependent variables and independent variables
* Derived the model coefficients and model intercept to fill values in the above linear regression equation



Revenue = 2.2456 \* budget+ 1934508.9101 \* popularity -29497467.2731

## Linear regression Analysis:

* Intercept -29497467.2731 indicates y- intercept value when all other variables are zero
* Coefficient estimate budget 2.2456 indicates, while keeping all other variables constant a unit increase in budget will increase Revenue by 2.2456
* Coefficient estimate popularity 1934508.9101 indictates, while keeping all other variables constant a unit increase in popularity will increase revenue by 1934508.9101

## Linear regression conclusion:

* Popularity is the major factor for increase in revenue
* Budget of the movie is very less factor for increase in revenue

# **Naïve Bayes**

* We used Gaussian NB of Naïve Bayes from SciKit learn package for classifying the data
* We then split the Data set into training model and test model
* We used Cross validation with n fold method
* We have used Cross validation through SciKit learn package to compare n fold method

## Naive Bayes implementation

* We created a Gaussian NB instance.
* We trained the model using training data set features and label.
* We predicted the labels on the test data set based on the above trained model and achieved an accuracy rate of 37.08%



## Naive Bayes conclusion:

Naive Bayes algorithm used probability and classified the labels based on the features since the data set had 4 labels, Naïve Bayes accuracy rate was very low hence we needed to use other classification algorithms

# **Random Forest Algorithm:**

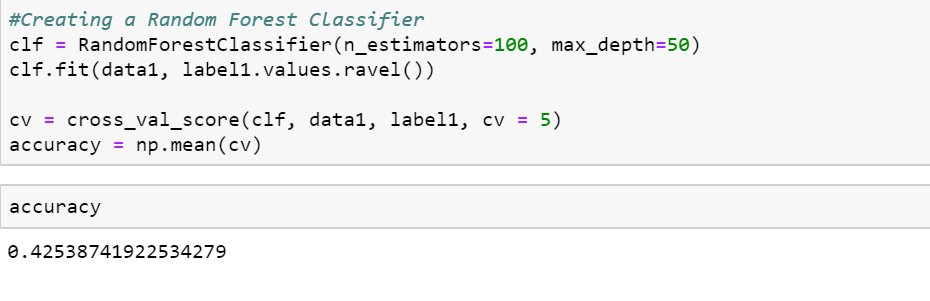
* We used RandomForestClassifier from SciKit learn package for classifying the data
* We converted the data set into dummy variables
* We used cross validation with n fold method
* We used cross validation through SciKit learn package to compare n fold method

# **Random Forest Implementation:**

* A RandomForestClassifier is created.
* Converted the data into dummy variables using get\_dummies function of the pandas library in order to classify categorical data
* Trained the model using trained data set features and label.
* Predicted the labels on the test data set based on the above trained model.
* Achieved an accuracy rate of 42.5%

# 





# **KMeans Algorithm:**

* We used KMeans algorithm for clustering the data set.
* We used KMeans package from SciKit learn package.
* We derived the number of clusters for applying KMeans algorithm using elbow point method.

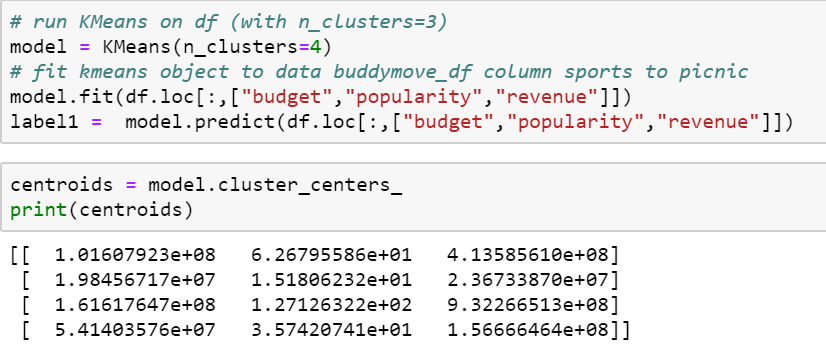
## Elbow Point method:

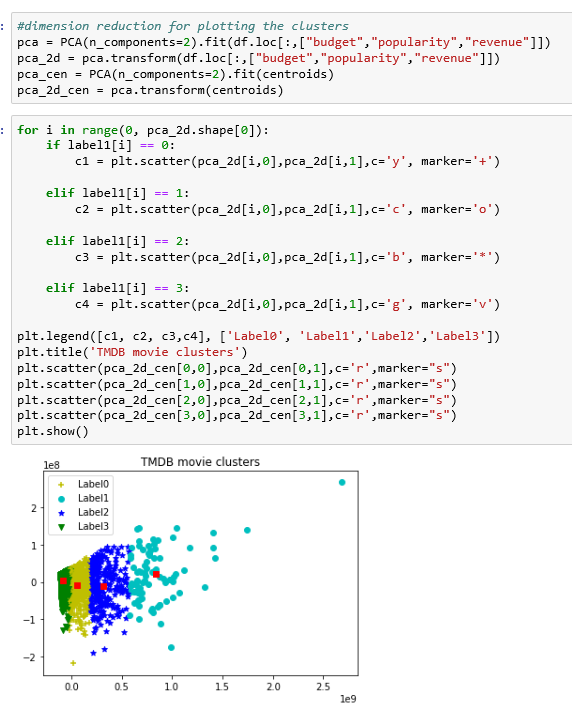
As per elbow point method 2 is the optimum number of clusters

Move data set does not look ideal for clustering through KMeans algorithm

## KMeans Implementation

* Created a KMeans object with 4 clusters.



* Trained the model with heart data set without labels.

# **Agglomerative Hierarchical Clustering Algorithm:**

* We used Hierarchical Clustering algorithm for clustering the data set.
* We used AgglomerativeClustering package from SciKit learn package.
* Linkage and Dendrogram functions from Scipy package used.
* Applied AgglomerativeClustering on the same number of clusters as KMeans in order to provide a comparison.

## Agglomerative Hierarchical Clustering Implementation:

* Used SciPy’s linkage method to compute distances between clusters.
* Used SciPy’s dendrogram method to visualize the dendrogram formed.
* Used AgglomerativeClustering method in order to cluster the data set.
* Predicted the labels and compared them with KMeans





# **Conclusion**

* Popularity is the decent predictor of how a movie will perform at the box office (revenue).
* Despite a relatively high correlation, the budget of the movie is much less a factor in predicting its performance.

# **Contributions**

**Shanmukh Pinna:**

* Loaded data into Mongo and connected through Jupityer notebook for analysis
* Performed exploratory data analysis
* Performed analysis on Linear Regression & Kmeans
* Performed trend analysis, correlation heatmap
* Performed Visualizations through Tableau
* Contributed to the final report
* Contributed text for final recording

**Siddhesh Mirjankar**

* Contributed to identifying the dataset
* Loaded data into Mongo and connected through Jupityer notebook for analysis
* Performed exploratory data analysis
* Performed analysis on Random Forest Classification & Hierarchical Clustering
* Contributed to the final report
* Contributed text for final recording

**Jack Murakami**

* Contributed to identifying the dataset
* Loaded data into Mongo and connected through Jupityer notebook for analysis
* Performed Naive bayes classification.
* Performed analysis through scatter plots, box plots
* Contributed to the final report
* Recorded video