Required packages

Hide

```
# This is the R chunk for the required packages
# This is the R chunk for the required packages
library(readr)
library(dplyr)
library(tidyr)
library(infotheo)
library(outliers)
library(Hmisc)
```

Executive Summary

The preprocessing steps are mentioned below :

1. The datasets are imported using the read_csv() function from the readr() package.
2. The movies.csv file has the genre information which is also present in the imdb_metadata.csv. So the genre column was dropped from the movies.csv using the subset function as displayed in the below code: new_data <- subset(new_data, select = -c(genres))
3. The movies.csv file had the title attribute which depicts the name of the movie and the release date in brackets, for Ex : Toy Story (1995). The movie_title was also present in the imdb_metadata.csv file but it has just the title Ex: Toy Story. As this attribute is a key to merge these two datasets I had to remove the bracket with release date from the movies.csv file. I did remove the brackets using the below regular expression.

reference - https://stackoverflow.com/questions/40539570/removing-parenthesis-in-r (https://stackoverflow.com/questions/40539570/removing-parenthesis-in-r)

remove the contents inside the bracket () including the brackets

movie_title <- trimws(gsub(". ∗?“,"", new_data$title))

add the vector with trimmed movie name to the new_data

new_data <- cbind(new_data, movie_title)

drop the column title as we have the movie_title column

new_data <- subset(new_data, select = -c(title))

Data

The data which is selected for analysis is related to IMDB rating of the movies and the comments or the tags given by different users to each of the movies. The first dataset is collected from the IMDB website from the source - https://www.kaggle.com/carolzhangdc/ (https://www.kaggle.com/carolzhangdc/) imdb-5000-movie-dataset. The data consists of attributes which describe the movie details in terms of budget, number of actors ,gross income, Facebook likes and IMDB rating. There are totally 37 variables for 5043 movies, spanning across years(1916 – 2016) in 65 countries. The IMDB rating is the target variable which depicts how good the movie is on a scale of 1 to 10.

Hide

```
# This is the R chunk for the Data Section
## import the imdb movie dataset
imdb_movie <- read_csv("movie_metadata.csv")
```

```
-- Column specification ----------------------------------------------------------------
cols(
  .default = col_double(),
  color = col_character(),
  director_name = col_character(),
  actor_2_name = col_character(),
  genres = col_character(),
  actor_1_name = col_character(),
  movie_title = col_character(),
  actor_3_name = col_character(),
  plot_keywords = col_character(),
  movie_imdb_link = col_character(),
  language = col_character(),
  country = col_character(),
  content_rating = col_character()
)
i Use `spec()` for the full column specifications.
```

Hide

```
head(imdb_movie)
```

color <chr>	director_name <chr>	num_critic_for_reviews <dbl>	duration <dbl>	director_facebook_likes <dbl>
Color	James Cameron	723	178	0
Color	Gore Verbinski	302	169	563
Color	Sam Mendes	602	148	0
Color	Christopher Nolan	813	164	22000
NA	Doug Walker	NA	NA	131
Color	Andrew Stanton	462	132	475

6 rows | 1-5 of 28 columns

Hide

```
NA
```

The second data was collected from the source : https://www.kaggle.com/rounakbanik/the-movies-dataset (https://www.kaggle.com/rounakbanik/the-movies-dataset). The data consists of 5 csv files. I will use specifically two files from this data. The movies.csv and tags.csv. The movies.csv contains the movieID , title and genre. The tags.csv contains the userID, movieID,tags,timestamp.

I will first merge the movies.csv and tags.csv from the second data source. I will merge these two datasets using left_join using the movieID as the key. So from this I can access the title of the movie and the tags it has received.

Hide

```
## import the dataset tags
tags <- read_csv("tags.csv")
```

```
-- Column specification --------------------------------------------------------
cols(
  userId = col_double(),
  movieId = col_double(),
  tag = col_character(),
  timestamp = col_double()
)
```

Hide

```
head(tags)
```

userId <dbl>	movieId <dbl>	tag <chr>	timestamp <dbl>
2	60756	funny	1445714994
2	60756	Highly quotable	1445714996
2	60756	will ferrell	1445714992
2	89774	Boxing story	1445715207
2	89774	MMA	1445715200
2	89774	Tom Hardy	1445715205

6 rows

Hide

```
## import the dataset movies
movies <- read_csv("movies.csv")
```

```
-- Column specification --------------------------------------------------------
cols(
  movieId = col_double(),
  title = col_character(),
  genres = col_character()
)
```

Hide

```
head(movies)
```

movieId <dbl>	title <chr>	genres <chr>
1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy
2	Jumanji (1995)	Adventure|Children|Fantasy
3	Grumpier Old Men (1995)	Comedy|Romance
4	Waiting to Exhale (1995)	Comedy|Drama|Romance
5	Father of the Bride Part II (1995)	Comedy
6	Heat (1995)	Action|Crime|Thriller

6 rows

Hide

```
## the tags dataset has the movie id and contains the tags or
## comments given by each of the user for a specific movie
## megre movies and tags dataframe to obtain the movie_title by using the left_join in mutate
new_data <- tags %>% left_join(movies, by="movieId")
```

Now I will merge the dataframe new_data obtained from the second source and the imdb_movie dataframe from the first source using the inner_join and movie_title as the key. So from this merge I will be able to obtain the movie_title , tags and the details of the movie like the number of facebook likes , cast of the the movie, IMDB rating etc.

Hide

```
## merge the two datasets
merged_data <- imdb_movie %>% inner_join(new_data, by="movie_title")
head(merged_data)
```

color <chr>	director_name <chr>	num_critic_for_reviews <dbl>	duration <dbl>	director_facebook_likes <dbl>
Color	James Cameron	723	178	0
Color	James Cameron	723	178	0
Color	James Cameron	723	178	0
Color	James Cameron	723	178	0
Color	James Cameron	723	178	0
Color	James Cameron	723	178	0

6 rows | 1-5 of 32 columns

Understand

The str displays the structure of the dataframe and displays the datatype of each attribute. The dataset does contain multiple data types , Ex: actor_names, color etc are all character variables and duration , facebook likes, title_year are numeric.

Hide

```
# This is the R chunk for the Understand Section
str(merged_data)
```

```
tibble [1,828 x 32] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
 $ color                    : chr [1:1828] "Color" "Color" "Color" "Color" ...
 $ director_name            : chr [1:1828] "James Cameron" "James Cameron" "James Cameron" "J
ames Cameron" ...
 $ num_critic_for_reviews   : num [1:1828] 723 723 723 723 723 723 723 723 723 723 ...
 $ duration                 : num [1:1828] 178 178 178 178 178 178 178 178 178 178 ...
 $ director_facebook_likes  : num [1:1828] 0 0 0 0 0 0 0 0 0 0 ...
 $ actor_3_facebook_likes   : num [1:1828] 855 855 855 855 855 855 855 855 855 855 ...
 $ actor_2_name             : chr [1:1828] "Joel David Moore" "Joel David Moore" "Joel David
Moore" "Joel David Moore" ...
 $ actor_1_facebook_likes   : num [1:1828] 1000 1000 1000 1000 1000 1000 1000 1000 1000 1000
...
 $ gross                    : num [1:1828] 7.61e+08 7.61e+08 7.61e+08 7.61e+08 7.61e+08 ...
 $ genres                   : chr [1:1828] "Action|Adventure|Fantasy|Sci-Fi" "Action|Adventur
e|Fantasy|Sci-Fi" "Action|Adventure|Fantasy|Sci-Fi" "Action|Adventure|Fantasy|Sci-Fi" ...
 $ actor_1_name             : chr [1:1828] "CCH Pounder" "CCH Pounder" "CCH Pounder" "CCH Pou
nder" ...
 $ movie_title              : chr [1:1828] "Avatar" "Avatar" "Avatar" "Avatar" ...
 $ num_voted_users          : num [1:1828] 886204 886204 886204 886204 886204 ...
 $ cast_total_facebook_likes: num [1:1828] 4834 4834 4834 4834 4834 ...
 $ actor_3_name             : chr [1:1828] "Wes Studi" "Wes Studi" "Wes Studi" "Wes Studi"
...
 $ facenumber_in_poster     : num [1:1828] 0 0 0 0 0 0 0 0 0 0 ...
 $ plot_keywords            : chr [1:1828] "avatar|future|marine|native|paraplegic" "avatar|f
uture|marine|native|paraplegic" "avatar|future|marine|native|paraplegic" "avatar|future|marin
e|native|paraplegic" ...
 $ movie_imdb_link          : chr [1:1828] "http://www.imdb.com/title/tt0499549/?ref_=fn_tt_t
t_1" "http://www.imdb.com/title/tt0499549/?ref_=fn_tt_tt_1" "http://www.imdb.com/title/tt0499
549/?ref_=fn_tt_tt_1" "http://www.imdb.com/title/tt0499549/?ref_=fn_tt_tt_1" ...
 $ num_user_for_reviews     : num [1:1828] 3054 3054 3054 3054 3054 ...
 $ language                 : chr [1:1828] "English" "English" "English" "English" ...
 $ country                  : chr [1:1828] "USA" "USA" "USA" "USA" ...
 $ content_rating           : chr [1:1828] "PG-13" "PG-13" "PG-13" "PG-13" ...
 $ budget                   : num [1:1828] 2.37e+08 2.37e+08 2.37e+08 2.37e+08 2.37e+08 2.37e
+08 2.37e+08 2.37e+08 2.37e+08 2.37e+08 ...
 $ title_year               : num [1:1828] 2009 2009 2009 2009 2009 ...
 $ actor_2_facebook_likes   : num [1:1828] 936 936 936 936 936 936 936 936 936 936 ...
 $ imdb_score               : num [1:1828] 7.9 7.9 7.9 7.9 7.9 7.9 7.9 7.9 7.9 7.9 ...
 $ aspect_ratio             : num [1:1828] 1.78 1.78 1.78 1.78 1.78 1.78 1.78 1.78 1.78 1.78
...
 $ movie_facebook_likes     : num [1:1828] 33000 33000 33000 33000 33000 33000 33000 33000 33
000 33000 ...
 $ userId                   : num [1:1828] 477 477 477 477 477 477 477 477 477 477 ...
 $ movieId                  : num [1:1828] 72998 72998 72998 72998 72998 ...
 $ tag                      : chr [1:1828] "bad science" "futuristic" "graphic design" "James
Cameron" ...
 $ timestamp                : num [1:1828] 1.26e+09 1.26e+09 1.26e+09 1.26e+09 1.26e+09 ...
 - attr(*, "spec")=
  .. cols(
  ..   color = col_character(),
  ..   director_name = col_character(),
  ..   num_critic_for_reviews = col_double(),
  ..   duration = col_double(),
  ..   director_facebook_likes = col_double(),
  ..   actor_3_facebook_likes = col_double(),
  ..   actor_2_name = col_character(),
  ..   actor_1_facebook_likes = col_double(),
  ..   gross = col_double(),
  ..   genres = col_character(),
  ..   actor_1_name = col_character(),
  ..   movie_title = col_character(),
  ..   num_voted_users = col_double(),
  ..   cast_total_facebook_likes = col_double(),
  ..   actor_3_name = col_character(),
  ..   facenumber_in_poster = col_double(),
  ..   plot_keywords = col_character(),
  ..   movie_imdb_link = col_character(),
  ..   num_user_for_reviews = col_double(),
  ..   language = col_character(),
  ..   country = col_character(),
  ..   content_rating = col_character(),
  ..   budget = col_double(),
  ..   title_year = col_double(),
  ..   actor_2_facebook_likes = col_double(),
  ..   imdb_score = col_double(),
  ..   aspect_ratio = col_double(),
  ..   movie_facebook_likes = col_double()
  .. )
```

Required Data type Conversions:

character to factor and ordered factor : The attributes content_rating and color are character attributes which must be converted to factor variables. The content Rating will be converted to ordered factor.

Hide

```
## convert the data type of content_rating from character to ordered factor
merged_data$content_rating %>% unique()
```

```
 [1] "PG-13"     "PG"        "G"         "R"         "TV-MA"     NA          "TV-14"
 [8] "Approved"  "Unrated"   "GP"        "NC-17"     "M"         "Not Rated" "X"
```

Hide

```
merged_data$content_rating <-  factor(merged_data$content_rating, levels = c("Approved" ,"G"
 ,"GP" ,"M" ,"NC-17" ,"Not Rated" ,"PG" ,"PG-13", "R", "TV-14", "TV-MA" ,"Unrated", "X"),
                                      ordered = TRUE)

## convert the data type of attribute color from character to factor
merged_data$color %>% unique()
```

```
[1] "Color"           "Black and White" NA
```

Hide

```
merged_data$color <- factor(merged_data$color, levels = c("Color","Black and White"))

## check the class of the two attributes
merged_data$color %>% class()
```

```
[1] "factor"
```

Hide

```
merged_data$content_rating %>% class()
```

```
[1] "ordered" "factor"
```

Numeric to factor: The attribute title_year displays the year when the movie was released. This attribute is numeric, which will be converted to factor.

Hide

```
## convert the title_year form numeric to factor
merged_data$title_year %>% unique()
```

```
 [1] 2009 2016 2013 2014 2005 1997 2012 2004 2010 2006 2011 2015 2008 2007 1998 2002
[17] 2003 2001 2000 1999 1991 1994 1995 1996 1992   NA 1993 1978 1980 1989 1990 1940
[33] 1986 1983 1982 1988 1979 1985 1984 1981 1960 1969 1987 1964 1962 1975 1970 1977
[49] 1968 1971 1961 1976 1939 1967 1958 1948 1959 1974 1945 1936 1937 1953 1942 1963
[65] 1949 1954 1952 1935 1957 1934 1955
```

Hide

```
year <- c(2009, 2016, 2013, 2014, 2005, 1997, 2012, 2004, 2010, 2006, 2011 ,2015, 2008, 2007,
1998, 2002, 2003, 2001,2000 ,1999, 1991, 1994, 1995, 1996, 1992, 1993, 1978, 1980, 1989, 1990
, 1940, 1986, 1983 ,1982, 1988,1979, 1985, 1984, 1981, 1960, 1969, 1987, 1964, 1962, 1975, 19
70, 1977 ,1968 ,1971, 1961 ,1976, 1939 ,1967,1958, 1948, 1959, 1974, 1945, 1936, 1937, 1953,
1942, 1963, 1949, 1954, 1952, 1935, 1957, 1934, 1955)

merged_data$title_year <- factor(merged_data$title_year, levels = year)

## check the class of the attribute title_year
merged_data$title_year %>% class()
```

```
[1] "factor"
```

Numeric to date: The data type of the attribute timestamp is displayed as Numeric. It should be converted to date format to extract the date using the as.Date.POSIXct function.

Hide

```
# convert the timestamp to date
# reference - https://discuss.analyticsvidhya.com/t/extract-date-and-time-from-unix-timestamp
-in-r/1883/2
date_col <- as.Date(as.POSIXct(merged_data$timestamp, origin="1971-01-01"))
merged_data$timestamp <- date_col

# display the head to check that the timestamp is converted to a date format
merged_data$timestamp %>% head(5)
```

```
[1] "2011-01-05" "2011-01-05" "2011-01-05" "2011-01-05" "2011-01-05"
```

Tidy & Manipulate Data I

The attribute timestamp which contains the date in the format yyyy-mm-dd. It violates the tidy principle each variable must have its own column. Here the timestamp depicts the tag or the comment, user added to a specific movie.The value year is the most important part which should be extracted from it. So we can seperate the date into year , month and date. This change will adhere to the tidy data principle.

Hide

```
# This is the R chunk for the Tidy & Manipulate Data I
## The attribute timestamp violates the tidy concept
## The timestamp can be separated to obtain the year of review or comment
merged_data <- merged_data %>% separate(timestamp, into = c("year", "month", "date"), sep =
"-")

## check that the new column with year is created
merged_data$year %>% head(5)
```

```
[1] "2011" "2011" "2011" "2011" "2011"
```

Tidy & Manipulate Data II

The popularity of a movie depends on the actor and the director in the movie. The popularity of main actor , director , cast corresponds to the number of facebook likes they get. So I can create a new column total_facebook_likes which consists of the sum of facebook likes for the main actors, director and the cast. The column total_facebook_likes will be helpful to determine the popularity of a movie.

Hide

```
## Mutate - can add all the facebook likes parameter , like
## the facebook likes of actors, directors and the whole cast into one column called total_fa
cebook_likes
merged_data <- mutate(merged_data,
       total_facebook_likes = director_facebook_likes + actor_3_facebook_likes +
         actor_1_facebook_likes + cast_total_facebook_likes)

## display the head of the mutated column total_facebook_likes
merged_data$total_facebook_likes %>% head(5)
```

```
[1] 6689 6689 6689 6689 6689
```

Scan I

The number of missing values in each of the columns is checked using the function is.na(). The missing values are present in the columns director_name, budget, total_facebook_likes, title_year. The decision to replace or omit a missing value depends on the data and the analysis. Normally the categorical values are replaced with mode while the numeric values are replaced with mean or median or constant.

Hide

```
# This is the R chunk for the Scan I
colSums(is.na(merged_data))
```

```
                color             director_name    num_critic_for_reviews 
                    2                        11                         0 
             duration   director_facebook_likes    actor_3_facebook_likes 
                    0                        11                         2 
         actor_2_name    actor_1_facebook_likes                     gross 
                    0                         0                        94 
               genres              actor_1_name               movie_title 
                    0                         0                         0 
      num_voted_users cast_total_facebook_likes              actor_3_name 
                    0                         0                         2 
 facenumber_in_poster             plot_keywords           movie_imdb_link 
                    0                         1                         0 
 num_user_for_reviews                  language                   country 
                    0                         0                         0 
       content_rating                    budget                title_year 
                   11                        40                        11 
actor_2_facebook_likes               imdb_score              aspect_ratio 
                    0                         0                         7 
 movie_facebook_likes                    userId                   movieId 
                    0                         0                         0 
                  tag                      year                     month 
                    0                         0                         0 
                 date      total_facebook_likes 
                    0                        13 
```

- Here the missing values of numeric data type director_facebook_likes, total_facebook_likes, actor_3_facebook_likes will be replaced with zero.

Hide

```
merged_data$director_facebook_likes <- impute(merged_data$director_facebook_likes, fun = 0)
merged_data$total_facebook_likes <- impute(merged_data$total_facebook_likes, fun = 0)
merged_data$actor_3_facebook_likes <- impute(merged_data$actor_3_facebook_likes, fun = 0)
```

- The color is a categorical attribute which has two values Black and White , Color. It will be replaced with mode which is the Color value because majority of the movies are Color in the dataset.

Hide

```
merged_data$color <- impute(merged_data$color, fun = mode)
```

- The other columns with missing values like director_name,content_rating,budget,gross, title_year, aspect_ratio will be omitted. As these cannot be replaced with a value. Because replacing a missing value of director_name with some other name does not make sense. Similarly replacing the missing value of the budget of a movie with mean or median of the budget column does not make any sense. So all these remaining missing values will be omitted.

Hide

```
# check the dimension of the dataset before dropping the rows with missing values
dim(merged_data)
```

```
[1] 1828   35
```

Hide

```
## dimension of the dataset before dropping the values = (1828,35)
## drop the rows with missing values using na.omit()
merged_data <- na.omit(merged_data)
## check the dimension of the data after dropping the missing values
dim(merged_data)
```

```
[1] 1703   35
```

Hide

```
## validate that there are no missing values present in the data
colSums(is.na(merged_data))
```

```
                 color            director_name   num_critic_for_reviews
                     0                        0                        0
              duration  director_facebook_likes   actor_3_facebook_likes
                     0                        0                        0
          actor_2_name   actor_1_facebook_likes                    gross
                     0                        0                        0
                genres             actor_1_name              movie_title
                     0                        0                        0
       num_voted_users cast_total_facebook_likes            actor_3_name
                     0                        0                        0
  facenumber_in_poster            plot_keywords          movie_imdb_link
                     0                        0                        0
  num_user_for_reviews                 language                  country
                     0                        0                        0
        content_rating                   budget               title_year
                     0                        0                        0
actor_2_facebook_likes               imdb_score             aspect_ratio
                     0                        0                        0
  movie_facebook_likes                   userId                  movieId
                     0                        0                        0
                   tag                     year                    month
                     0                        0                        0
                  date     total_facebook_likes
                     0                        0
```

Less than 7% of data is lost after omitting the remaining missing values.

Checking the dataframe for special values: The number of special values in each of the columns and the missing will be detected using an user defined function and sum(is.na()) and sapply to apply the function on each column in the dataframe.

Hide

```
## create a user defined function to check for special values , nan and na
function(x){
  if (is.numeric(x)) (is.infinite(x) | is.nan(x) | is.na(x))
}
```

```
function(x){
  if (is.numeric(x)) (is.infinite(x) | is.nan(x) | is.na(x))
}
```

Hide

```
## using sapply call the user defined function to detect the special values , nan and na in #
#each of the columns
sapply(merged_data, function(x) sum(is.na(x)) )
```

```
                 color            director_name   num_critic_for_reviews 
                     0                        0                        0 
              duration  director_facebook_likes   actor_3_facebook_likes 
                     0                        0                        0 
          actor_2_name   actor_1_facebook_likes                    gross 
                     0                        0                        0 
                genres             actor_1_name              movie_title 
                     0                        0                        0 
       num_voted_users cast_total_facebook_likes            actor_3_name 
                     0                        0                        0 
  facenumber_in_poster            plot_keywords          movie_imdb_link 
                     0                        0                        0 
  num_user_for_reviews                 language                  country 
                     0                        0                        0 
        content_rating                   budget               title_year 
                     0                        0                        0 
actor_2_facebook_likes               imdb_score             aspect_ratio 
                     0                        0                        0 
  movie_facebook_likes                   userId                  movieId 
                     0                        0                        0 
                   tag                     year                    month 
                     0                        0                        0 
                  date     total_facebook_likes 
                     0                        0 
```

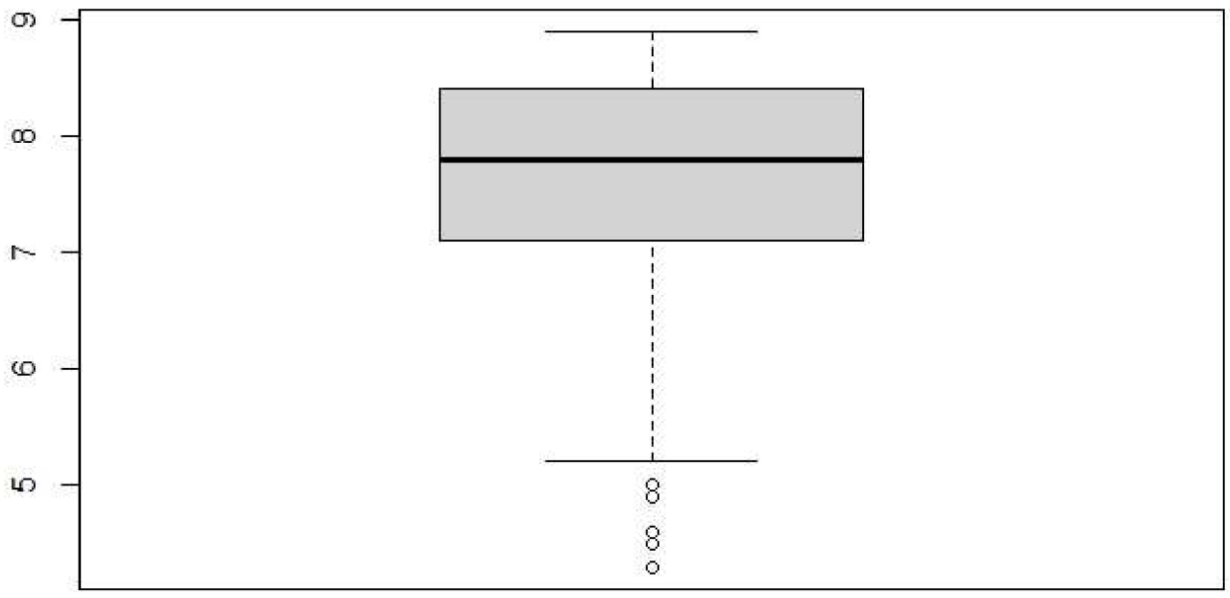
There are no special values in the data.

Scan II

The important numeric variables to be explored for outliers are the IMDB rating and the num_voted_users. Because the outliers in these attributes will affect the analysis. I will check the outliers in the IMDB_rating attribute using the boxplot.

Hide

```
## Boxplot to check for outliers in imdb_score
merged_data$imdb_score %>% boxplot()
```

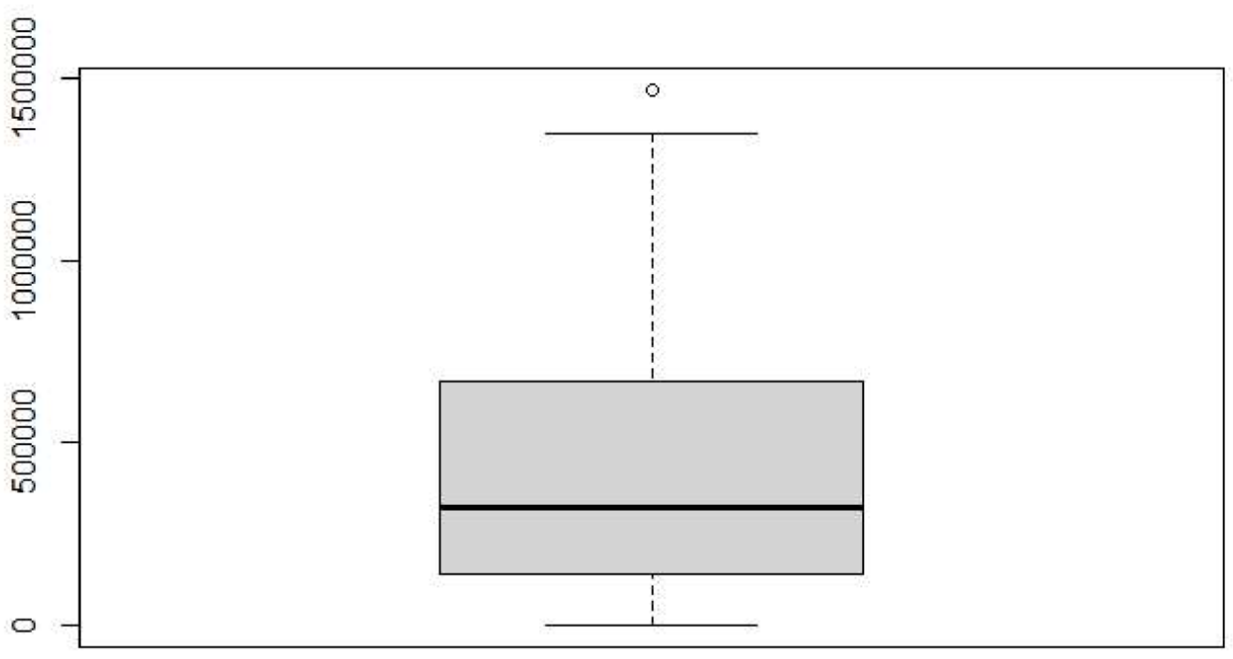


As seen from the above boxplot there are some outliers in lower region of the Boxplot. I cannot omit these outliers as these correspond to the movies with less imdb_score, because these movies are important for analysis as to check why a particular movie was a flop. I will apply a transformation method for the imdb_score attribute.

Similarly I will check for the outliers in num_voted_users using the boxplot

Hide

```
## Boxplot to check for outliers in num_voted_users
merged_data$num_voted_users %>% boxplot()
```



There is only one outlier in the num_voted_users. The movie Inception which has 1468200 users who voted for this movie. There are options to deal with this outlier like imputing the outliers with mean , median or dropping the outlier values. Imputing the outlier with mean or median for num_voted_users does not make sense. As this is the only outlier value I will omit it using the Capping method.

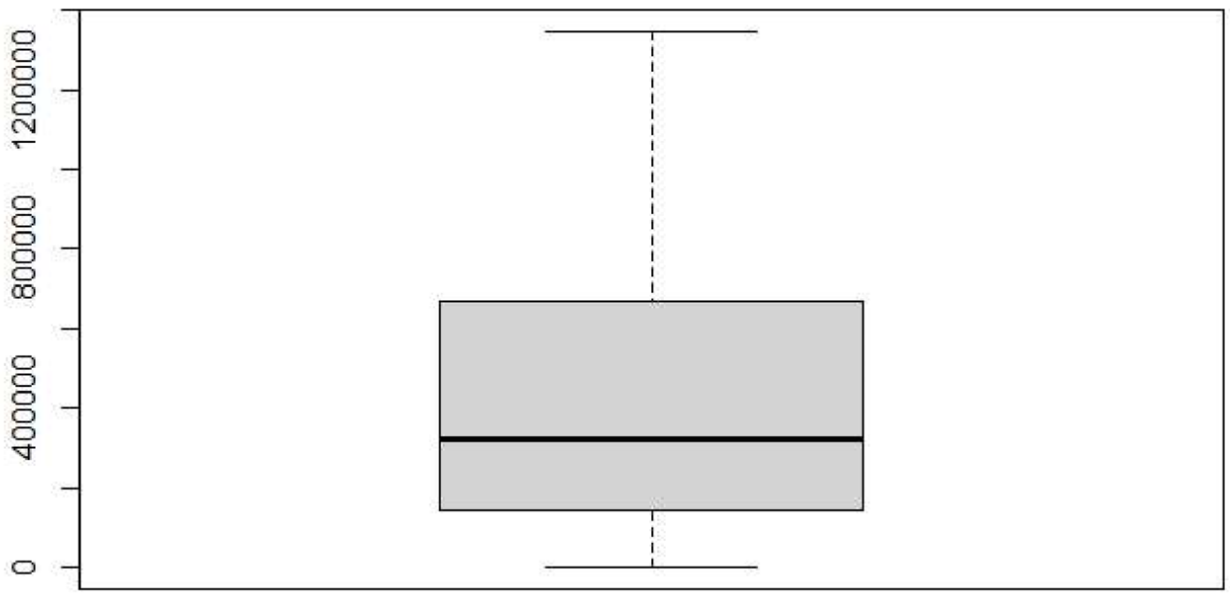
Hide

```
# user defined function to delete the outlier using the capping method
cap <- function(x){
    quantiles <- quantile( x, c(.05, 0.25, 0.75, .95 ) )
    x[ x < quantiles[2] - 1.5*IQR(x) ] <- quantiles[1]
    x[ x > quantiles[3] + 1.5*IQR(x) ] <- quantiles[4]
    x
}

## call the cap function to omit the outliers
num_voted_users_capped <- merged_data$num_voted_users %>% cap()

## check the summary to see that the outlier is dropped
summary(num_voted_users_capped)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
   3135  142619  324671  484281  666937 1347461 
```

Hide

```
## Also plot the boxplot to validate the outlier is omitted
num_voted_users_capped %>% boxplot()
```



From the above summary and boxplot we can see that the outlier is dropped and the max_value for the num_voted_users is 1347461.

Transform

As observed from the previous section the imdb_score attribute has outliers in the lower region of the boxplot which corresponds to movies with less imdb_score. The column imdb_score will be transformed using the Equal width binning with 3 bins or intervals. So these bins can depict movies with imdb_score (4.3,5.83), (5.83,7.36), (5.83, 8.90) as flop, average and Hit respectively. So binning the attribute imdb_score removes the outliers.

Hide

```
## applying equal width binning for imdb_score
binned_score <-discretize(merged_data$imdb_score, disc = "equalwidth")

## bind the binned_score with imdb_score to compare
merged_data$imdb_score %>% bind_cols(binned_score) %>% head()
```

```
New names:
* NA -> ...1
```

	...1 <dbl>	X <int>
1	7.9	9
2	7.9	9
3	7.9	9
4	7.9	9

	...1 <dbl>	X <int>
5	7.9	9
6	7.9	9

6 rows