IMDB MOVIES ANALYSIS

Liza & Malu

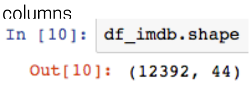
DATA

The IMDB Movies Dataset contains information about **14,762 movies and 44 columns**. The data has already been preprocessed and cleaned.
Columns:
title, wordsInTitle, url, imdbRating, ratingCount, duration, year, Type, nrOfWins, nrOfNominations, nrOfPhotos, nrOfNewsArticles, nrOfUserReviews, nrOfGenre,Other
Columns are for genre and they are dummy (0/1) variables:
Action, Adult, Adventure, Animation, Biography, Comedy, Crime, Documentary, Drama, Family, Fantasy, FilmNoir, GameShow, History, Horror, Music, Musical, Mystery, News, RealityTV, Romance, SciFi, Short, Sport, TalkShow, Thriller, War, Western.

DATA CLEANING

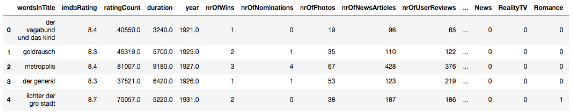
- Check for Duplicates
- Find Outliers
- Missing Values (remove)

After removing the missing values and dropping the unnecessary columns



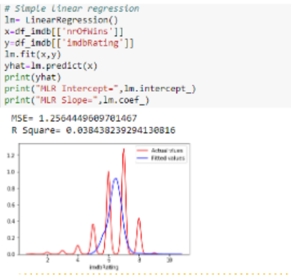
```
In [10]: df_imdb.shape

Out[10]: (12392, 44)
```



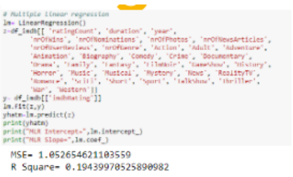
	wordsInTitle	imdbRating	ratingCount	duration	year	nrOfWins	nrOfNominations	nrOfPhotos	nrOfNewsArticles	nrOfUserReviews	...	News	RealityTV	Romance
0	der vagabund und das kind	8.4	40550.0	3240.0	1921.0	1	0	19	96	85	...	0	0	0
1	goldrausch	8.3	45319.0	5700.0	1925.0	2	1	35	110	122	...	0	0	0
2	metropolis	8.4	81007.0	9180.0	1927.0	3	4	67	428	376	...	0	0	0
3	der general	8.3	37521.0	6420.0	1926.0	1	1	53	123	219	...	0	0	0
4	lichter der gro stadt	8.7	70057.0	5220.0	1931.0	2	0	38	187	186	...	0	0	1

MODEL BUILDING



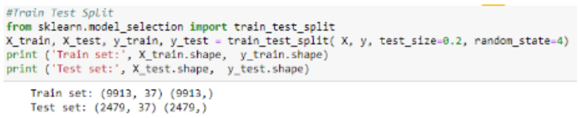
```
# Simple linear regression
lm= LinearRegression()
x=df_imdb[['nrOfWins']]
y=df_imdb[['imdbRating']]
lm.fit(x,y)
yhat=lm.predict(x)
print(yhat)
print("MLR Intercept=",lm.intercept_)
print("MLR Slope=",lm.coef_)
```

```
MSE= 1.2564449609701467
R Square= 0.038438239294130816
```



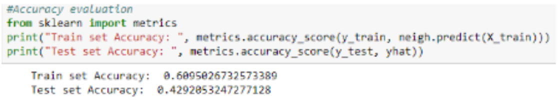
```
# Multiple linear regression
lm= LinearRegression()
z=df_imdb[[ 'ratingCount', 'duration', 'year',
        'nrOfWins', 'nrOfNominations', 'nrOfPhotos', 'nrOfNewsArticles',
        'nrOfUserReviews', 'nrOfGenre', 'Action', 'Adult', 'Adventure',
        'Animation', 'Biography', 'Comedy', 'Crime', 'Documentary',
        'Drama', 'Family', 'Fantasy', 'FilmNoir', 'GameShow', 'History',
        'Horror', 'Music', 'Musical', 'Mystery', 'News', 'RealityTV',
        'Romance', 'SciFi', 'Short', 'Sport', 'TalkShow', 'Thriller',
        'War', 'Western']]
y= df_imdb[['imdbRating']]
lm.fit(z,y)
yhatm=lm.predict(z)
print(yhatm)
print("MLR Intercept=",lm.intercept_)
print("MLR Slope=",lm.coef_)
```

```
MSE= 1.052654621103559
R Square= 0.19439970525890982
```



```
#Train Test Split
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split( X, y, test_size=0.2, random_state=4)
print ('Train set:', X_train.shape,  y_train.shape)
print ('Test set:', X_test.shape,  y_test.shape)
```

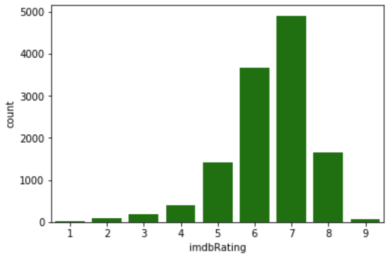
```
    Train set: (9913, 37) (9913,)
    Test set: (2479, 37) (2479,)
```

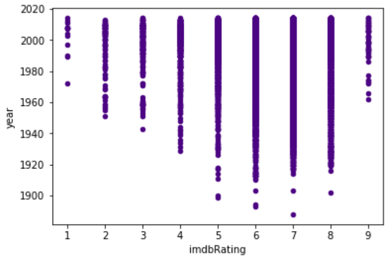


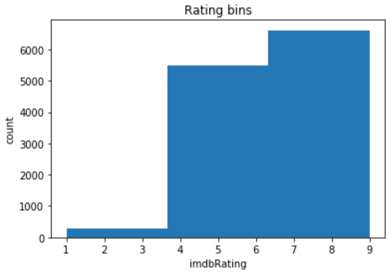
```
#Accuracy evaluation
from sklearn import metrics
print("Train set Accuracy: ", metrics.accuracy_score(y_train, neigh.predict(X_train)))
print("Test set Accuracy: ", metrics.accuracy_score(y_test, yhat))
```

```
    Train set Accuracy:  0.6095026732573389
    Test set Accuracy:  0.4292053247277128
```

VISUALIZATION







CONCLUSION

***3 models were tried to fit our data.
***Simple linear regression, multiple linear regression and K-NN (Clustering)
***Best result was from the KNN.
***More features to help suggest the movies: common actors, directors or the movies total gross
***Accuracy results did not match with our expectations for the possible movie recommender system