1) Dataset&Task Description:

Data Set Characteristics:	Multivariate	**Number of Instances:**	217	**Area:**	Computer
Attribute Characteristics:	Integer	**Number of Attributes:**	12	**Date Donated**	2017-10-11
Associated Tasks:	Classification, Regression	**Missing Values?**	Yes	**Number of Web Hits:**	29482

Attributes:

- Movie: Movie's name.
- Year: The year of the movie released
- Ratings: Movie's IMDB rating(X/10)
- Genre: Genre of the movie (Already encoded in the dataset)
- Gross: Total money earned from the movie
- Budget: Total money spent fort he movie.
- Screens: The number of screens (movie theater) in which the movie was released.
- Sequel: If the movie is a series; the order of the movie. If not the 'Sequel' = 1.
- Sentiment: Sentiment analysis of the movie which is obtained by social media.
- Views: How many people watced the movie.
- Likes: Number of likes (obtained by social media).
- Dislikes: Number of dislikes (obtained by social media).
- Comments: Number of comments (obtained by social media).
- Aggregate Followers: Total number of followers (obtained by social media).

(Dataset includes the movies which are released in 2014-2015)

Dataset URL :
https://archive.ics.uci.edu/ml/datasets/CSM+%28Conventional+and+Social+Media+Movies%29+Dataset+2014+and+2015

I created a new attribute named 'IsPayOffWithOnlyTicket' and this means; If the movie gain more than its budget with only the money which gained with the movie tickets. (Total movie gain is not just ticket gain). If yes I labeled it as 1 and if not I labeled it as 0.

The main task of this project is: try to classify the movies according to whether they are PayOffWithOnlyTicket or not with using the attributes : { Ratings, Budget, Screens, Sentiment, Views, Like-Dislike, Comments, TotalFollowers }.

The side task of this project is; try and compare various classification techniques.

Comparison of the classification techniques made by confussion matrix results.

2) Imputation:

In the first examination of the DataFrame, results of the missing value counts:

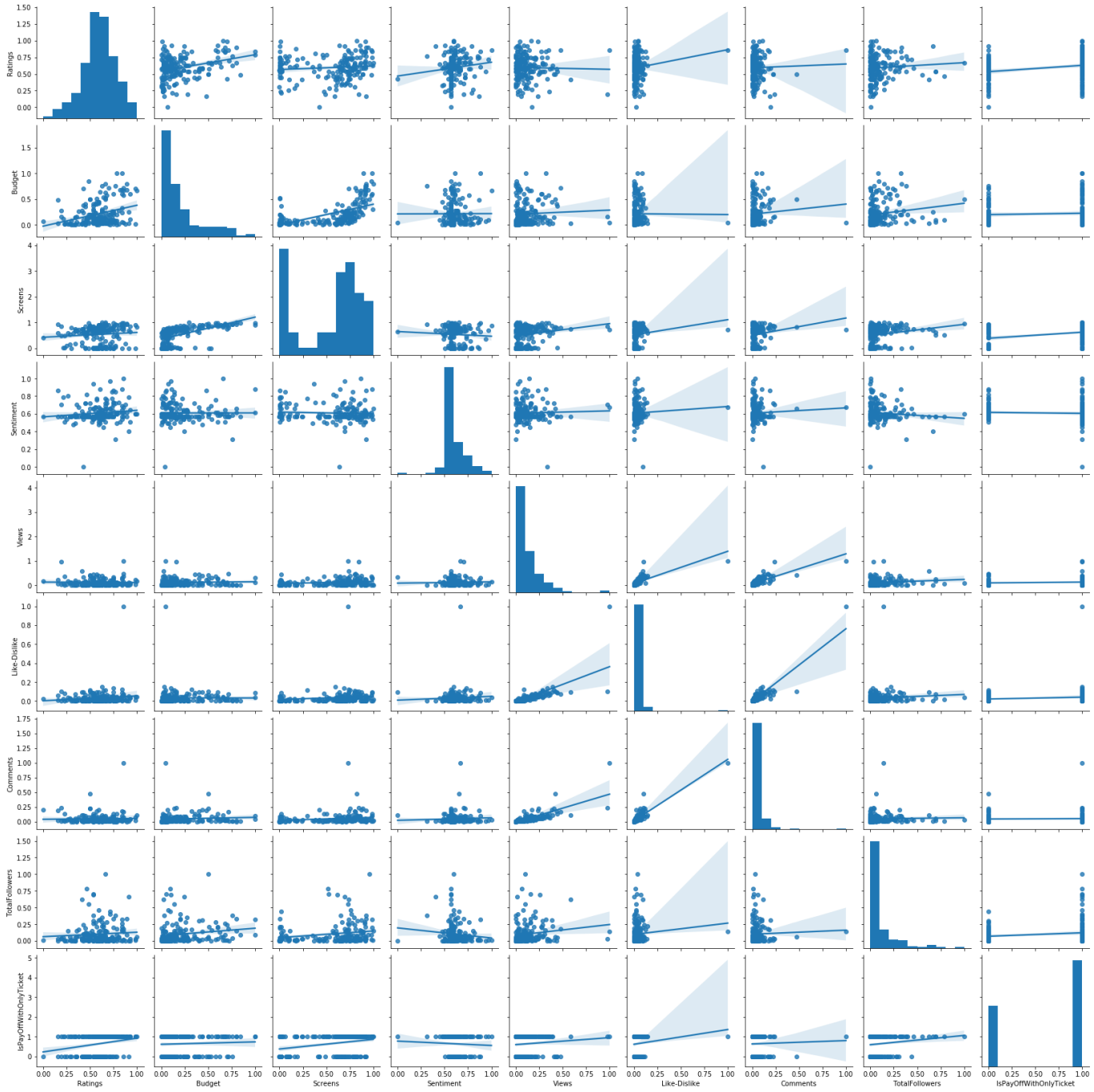
Movie	**0**
Year	
Ratings	0
Genre	0
Gross	0
Budget	1
Screens	10
Sequel	0
Sentiment	0
Views	0
Likes	0
Dislikes	0
Comments	0
Followers	35

We have missing values in only; {Budget, Screens, Followers} attributes. 'Budget' and 'Screens' missing value counts are very few so I dropped them. The Followers has '35' missing value and it's a bit high fort his dataset; A prediction tecnique can be used for predicting the missing Followers but I thought it is too hard to predict a usable value with this dataset because I have aproximately 200-250 rows in this dataset. So I decided to drop this missing values too. So at the end of this section I have no missing values in the DataFrame.

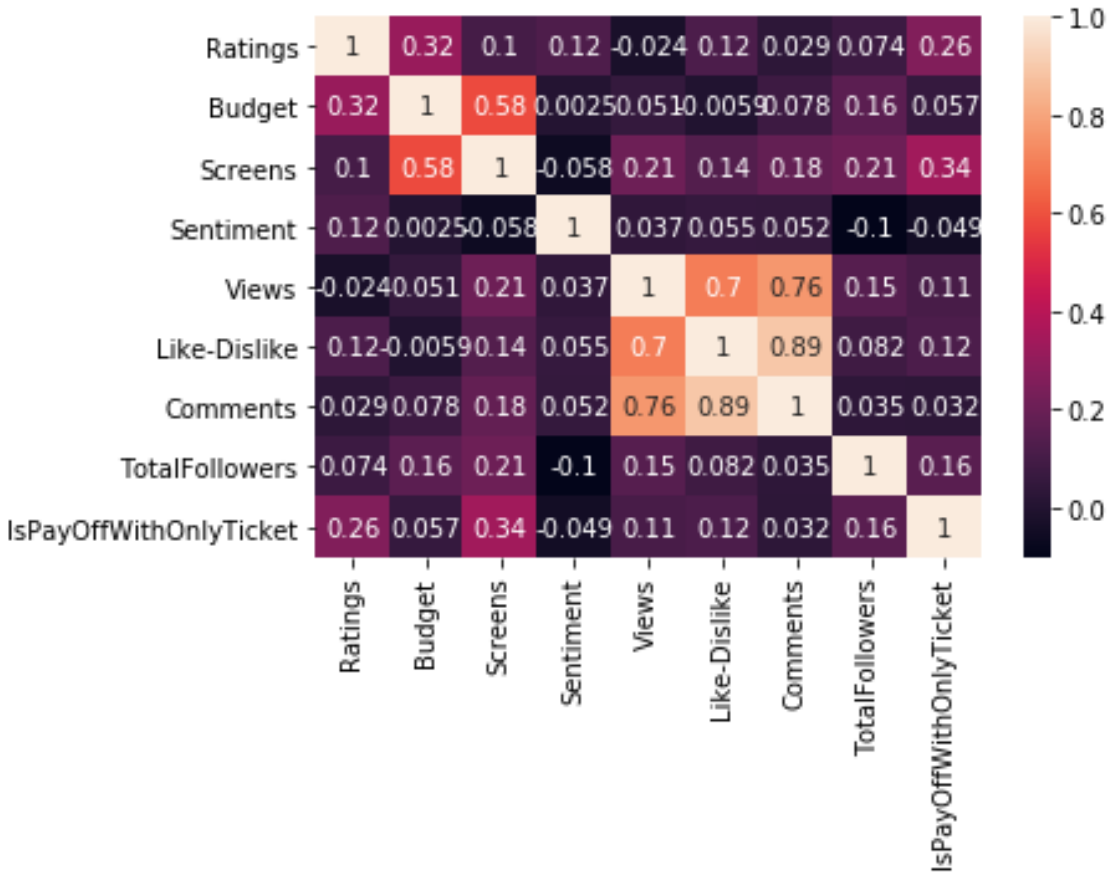
3) Encoding:

The categorical attributes like {Sentiment, Genre, …} are already encoded with LabelEncoder so the categorical values are represented in numbers.
I created a new DataFrame and filled this with attributes which I will use to train my model. And another DataFrame to target variable ('IsPayOffWithOnlyTicket').
I used MinMaxScaler for numerical variables.

4) Visualization:



(Pairplot of the DataFrame which will be used for training)



(Pairplot of the DataFrame which will be used for training)

5) **Classification Techniques:**

a) **Logistic Regression:**

Disadvantages: Works only when the predicted variable is binary, assumes all predictors are independent of each other, and assumes data is free of missing values.

Comment: It looks like the dataset is suitable for this technique and counters its disadvantages.

b) **Decision Tree Classifier:**

Advantages: Decision Tree is simple to understand and visualise, requires little data preparation, and can handle both numerical and categorical data.

Comment: Since I have both numerical and categorical attributes in the dataset, DTC might handle this well but also I have a few rows (200-250) and this might be cause bad results.

c) **K-Nearist Neighbours:**

Advantages: This algorithm is simple to implement, robust to noisy training data, and effective if training data is large.

Comment: Since my dataset includes a bit of noisy but on the other hand my training data is small, so this may lead to poor results.

d) **Random Forest:**

Disadvantages: Slow real time prediction, difficult to implement, and complex algorithm.

Comment: Since I have a small dataset and my task is not too much complex, and also the attributes are well prepared; It is possible to get very good results.

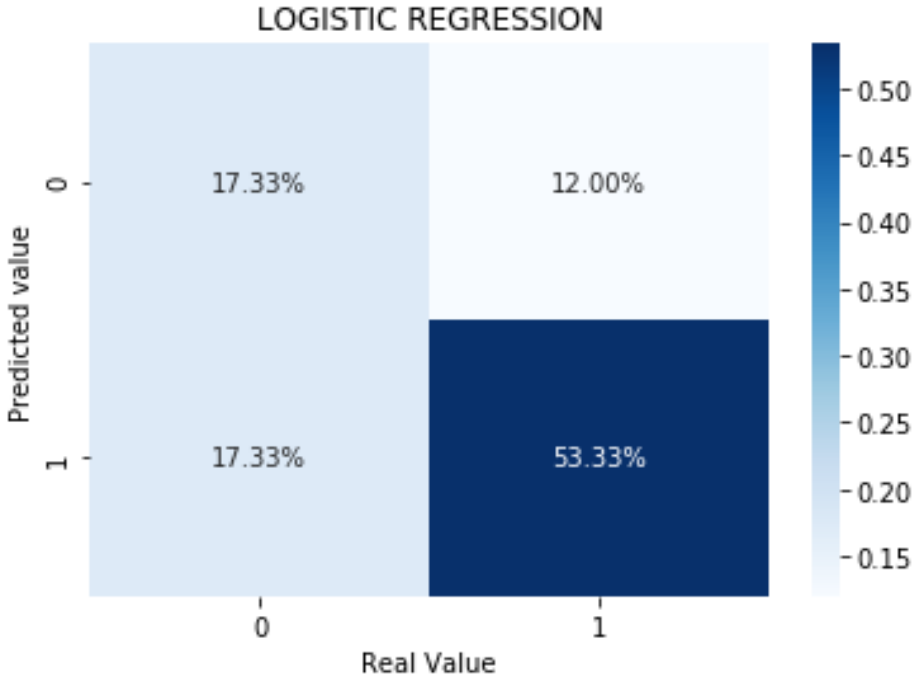
e) **Support Vector Machine:**

Definition: Support vector machine is a representation of the training data as points in space separated into categories by a clear gap that is as wide as possible. New examples are then mapped into that same space and predicted to belong to a category based on which side of the gap they fall.

Comment: Actually I do not have an idea about how it will be resulted.

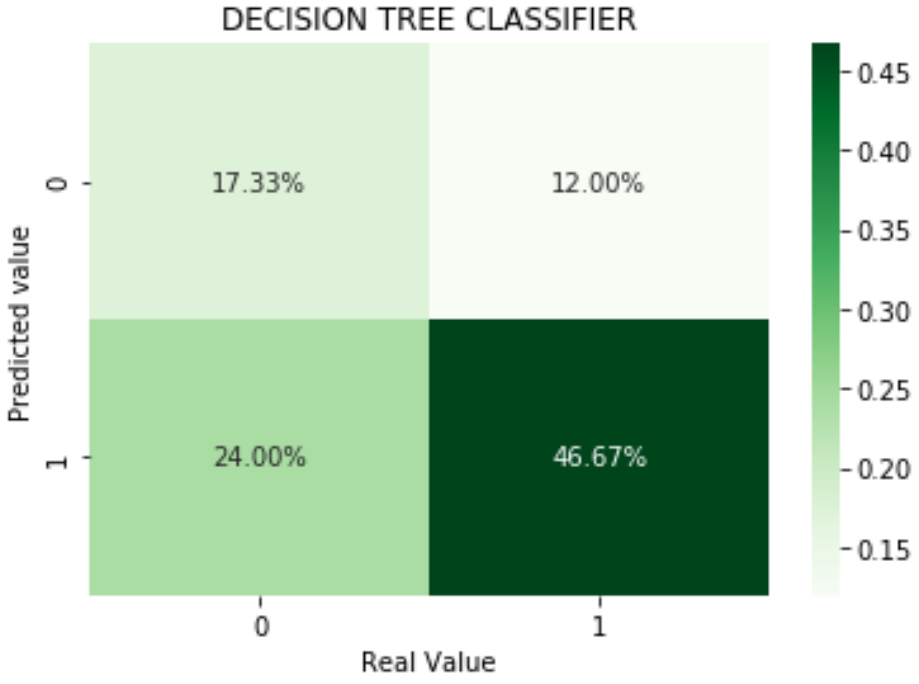
6) Evaluation and Comparison of Classification Techniques:

a) Logistic Regression:



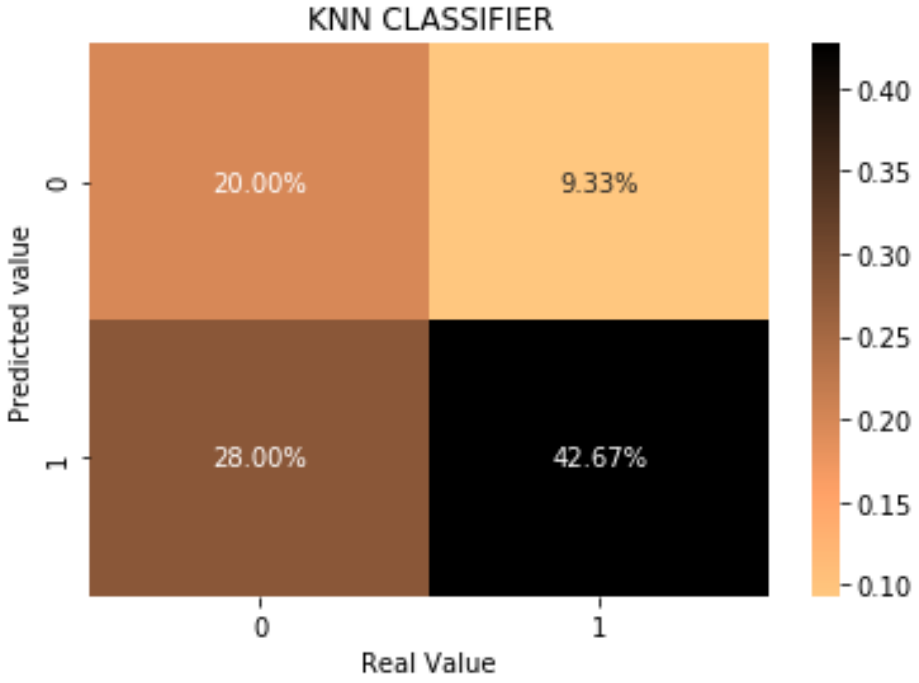

```
{'Sensitivity': 0.8163265306122449,
 'Specificity': 0.5,
 'Precision': 0.7547169811320755,
 'Accuracy': 0.7235374149659863,
 'F-Measure': 0.7843137254901962}
```

b) Decision Tree Classifier:



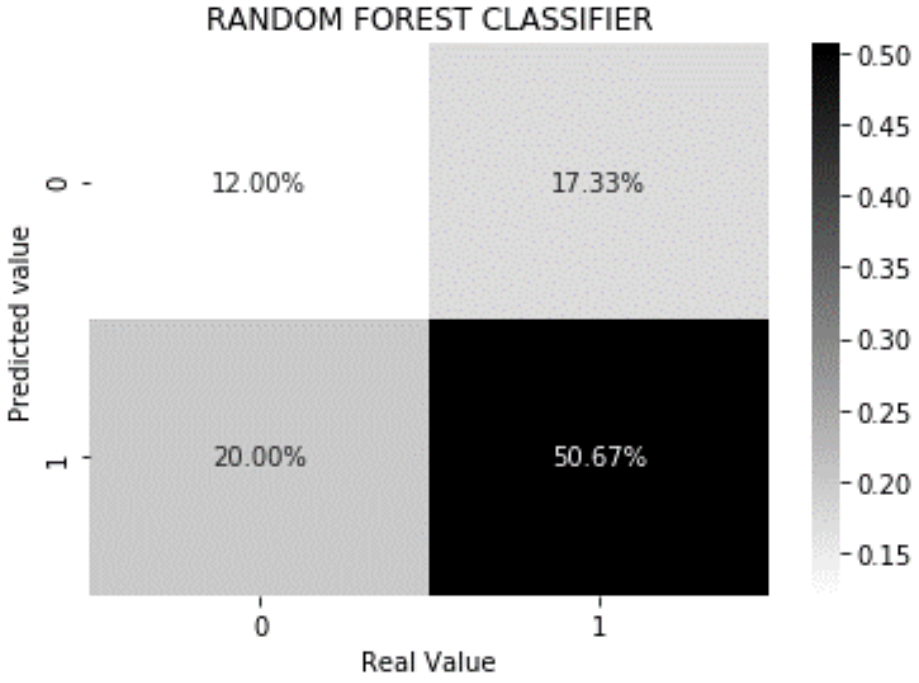

```
{'Sensitivity': 0.7954545454545454,
 'Specificity': 0.41935483870967744,
 'Precision': 0.660377358490566,
 'Accuracy': 0.6851319648093842,
 'F-Measure': 0.7216494845360825}
```

c) K-Nearist Neighbours:



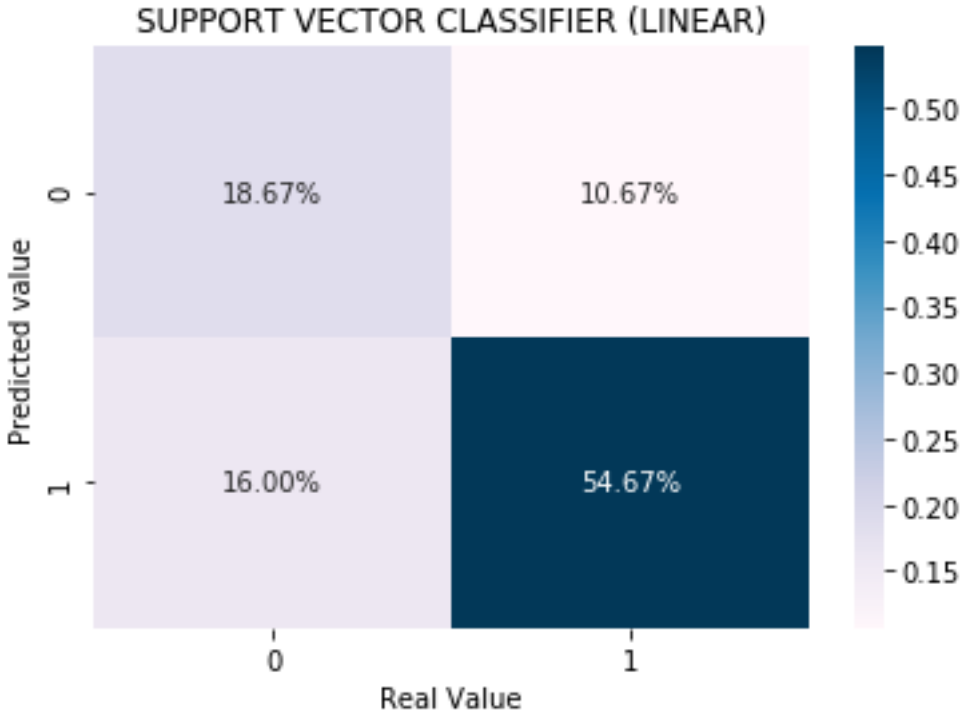

```
{'Sensitivity': 0.8205128205128205,
 'Specificity': 0.4166666666666667,
 'Precision': 0.6037735849056604,
 'Accuracy': 0.7020512820512821,
 'F-Measure': 0.6956521739130435}
```

d) Random Forest:



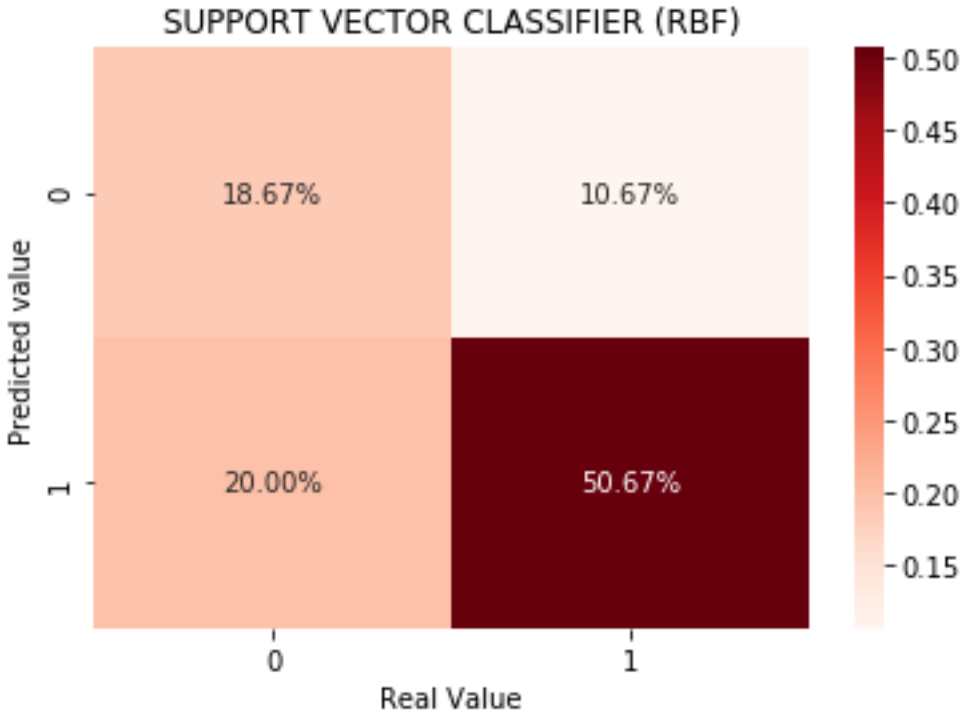

```
{'Sensitivity': 0.7450980392156863,
 'Specificity': 0.375,
 'Precision': 0.7169811320754716,
 'Accuracy': 0.6365359477124183,
 'F-Measure': 0.7307692307692308}
```

e) Support Vector Machine(Linear):




```
{'Sensitivity': 0.8367346938775511,
 'Specificity': 0.5384615384615384,
 'Precision': 0.7735849056603774,
 'Accuracy': 0.7492412349555206,
 'F-Measure': 0.803921568627451}
```

f) Support Vector Machine(RBF):




```
{'Sensitivity': 0.8260869565217391,
 'Specificity': 0.4827586206896552,
 'Precision': 0.7169811320754716,
 'Accuracy': 0.7253773113443279,
 'F-Measure': 0.7676767676767677}
```

Classification Algorithms	Accuracy	F-Measure
Logistic Regression	**72%**	**0.7843**
Decision Tree Classifier	**68%**	**0.7216**
K-Nearist Neighbours	**70%**	**0.6956**
Random Forest	**64%**	**0.7307**
Support Vector Machine(Linear)	**75%**	**0.8039**
Support Vector Machine(RBF)	**72%**	**0.7676**