Predicting Movie Success

Machine Learning Project

Team Members

Wael Mohamed Abd ElMoaaty

Mahmoud Hamed Mohamed Deif

Asmaa Sobhy Mostafa

Gehad Khaled Sayed

Asmaa Hamed Mohamed

Team ID : 5

Preprocessing

Preprocessing Steps:

1-Drop non-effective features (`'homepage'`, `'id'`, `'original_language'`, `'title'`, `'overview'`, `'original_title'`, `'status'`, `'tagline'`, `'movie_id'`).

2- Use one hot encoded to deal with JSON features

- Count how many this value of a feature appears in column
- Create columns for Top 100 in each feature and consider each one from this new columns as a new feature
- Then take each element in this feature and check if this value from top 100 put 1 in this cell and 0 in others cells
- After that drop the JSON columns

3- Replace missing values (Null or 0) in each feature with the average of this feature.

4- Normalize old features by MinMax Scaler

Finally, we do the same preprocessing for Regression and Classification

Regression Phase

Regression Models

- **Multivariable Linear Regression.**
- **Decision Tree Regressor.**

	Linear Regression	**Decision Tree Regressor**
Size of (train/ Test)	Train = 80% Test = 20%	Train = 80% Test = 20%
MSE	0.46	0.29
score	0.434	0. 652

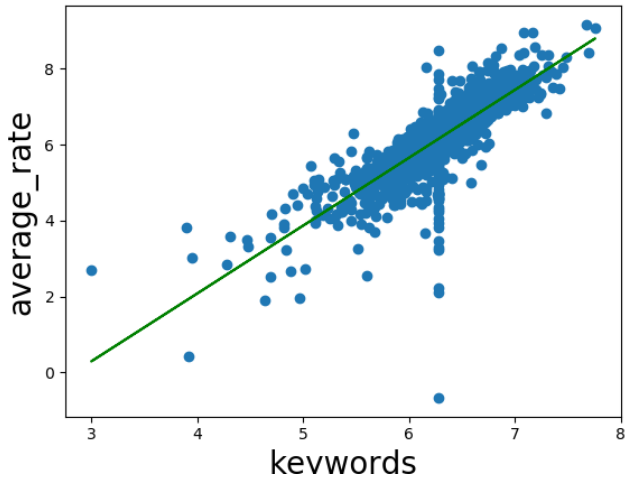
Screen Shots :

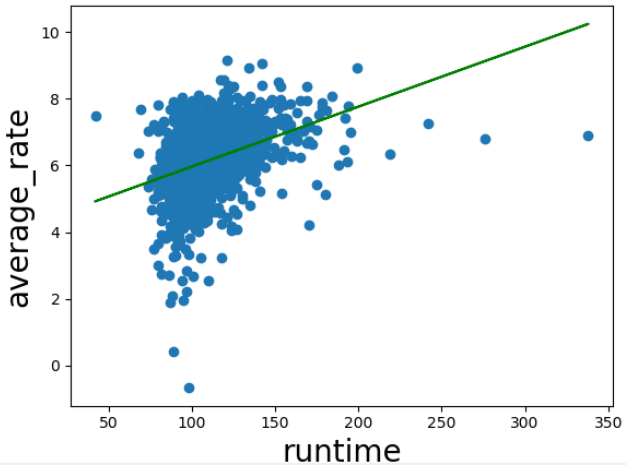
Output:

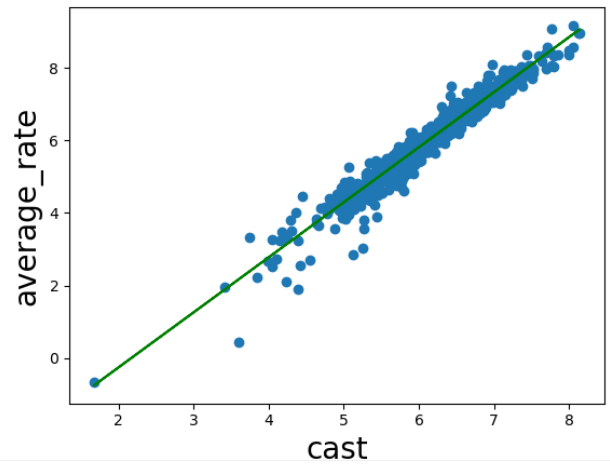
```
___________Linear Regression_____________
Intercept of linear regression model 6.389858788357235
Mean Square Error 0.46625916315203414
Score of linear regression 0.4342596703579731
_____________________________________________
___________Decision Tree Regression_____________
mean squared error (Tree)=  0.39010524849708056
Score =  0.5266618024451357
_____________________________________________
Least MSE 0.39010524849708056
The Best Model fitting the data is DecisionTreeRegressor(criterion='mse', max_depth=None, max_features=None,
                      max_leaf_nodes=None, min_impurity_decrease=0.0,
                      min_impurity_split=None, min_samples_leaf=10,
                      min_samples_split=10, min_weight_fraction_leaf=0.0,
                      presort=False, random_state=100, splitter='best')
```

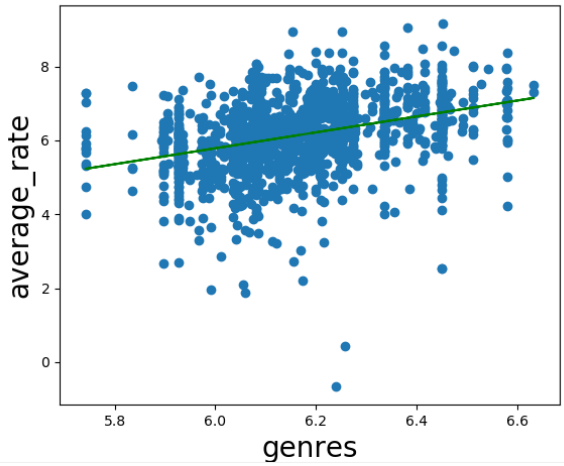
Plotting:

➢ Linear Regression









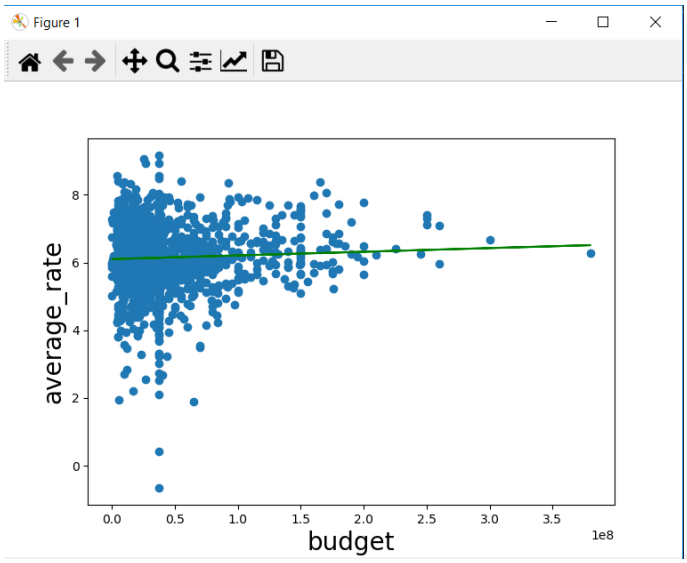
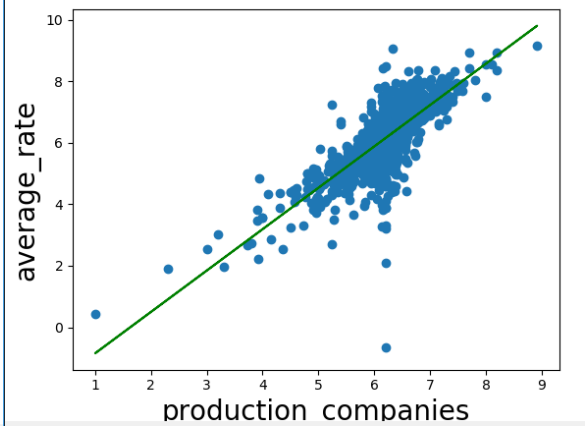
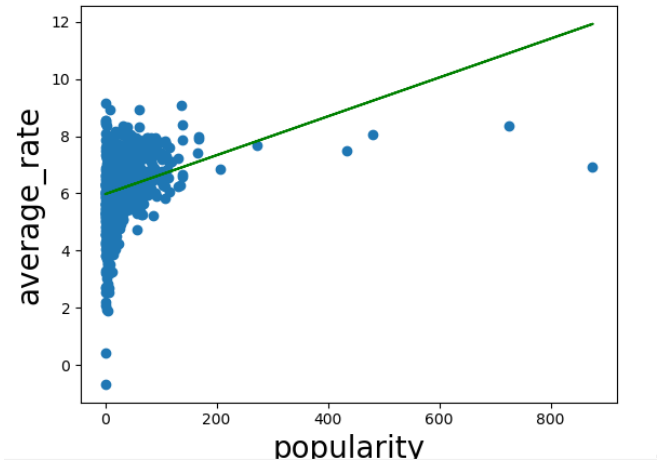
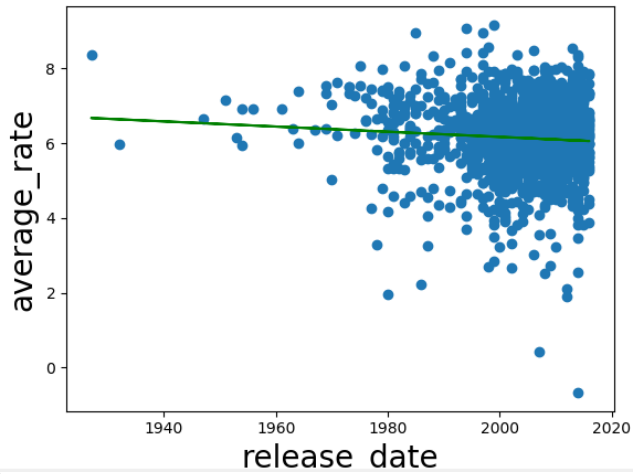
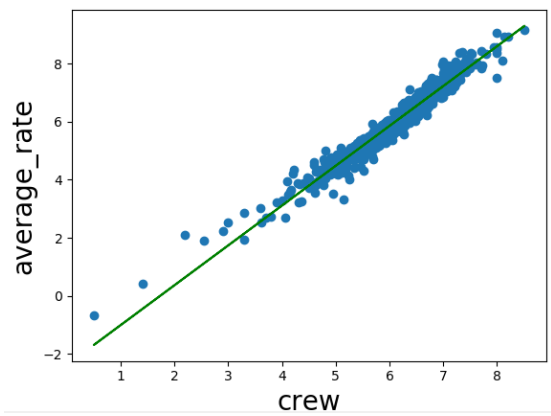


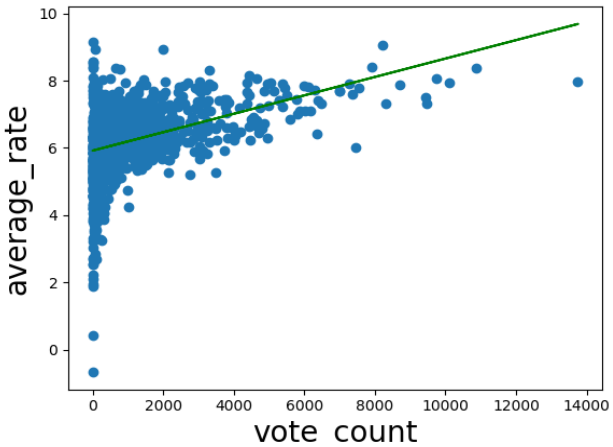
Figure 1
average_rate
budget
1e8

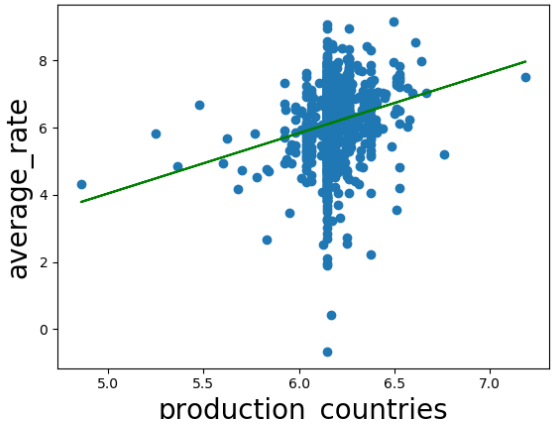

average_rate
production_companies

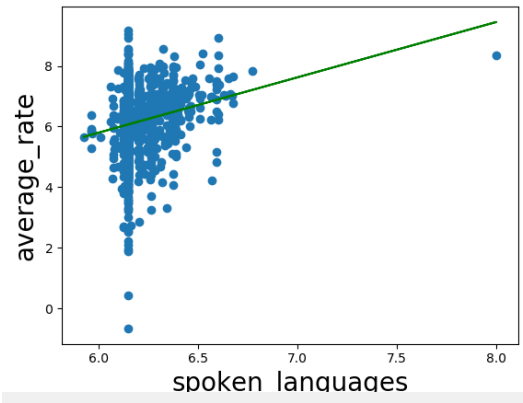

average_rate
popularity

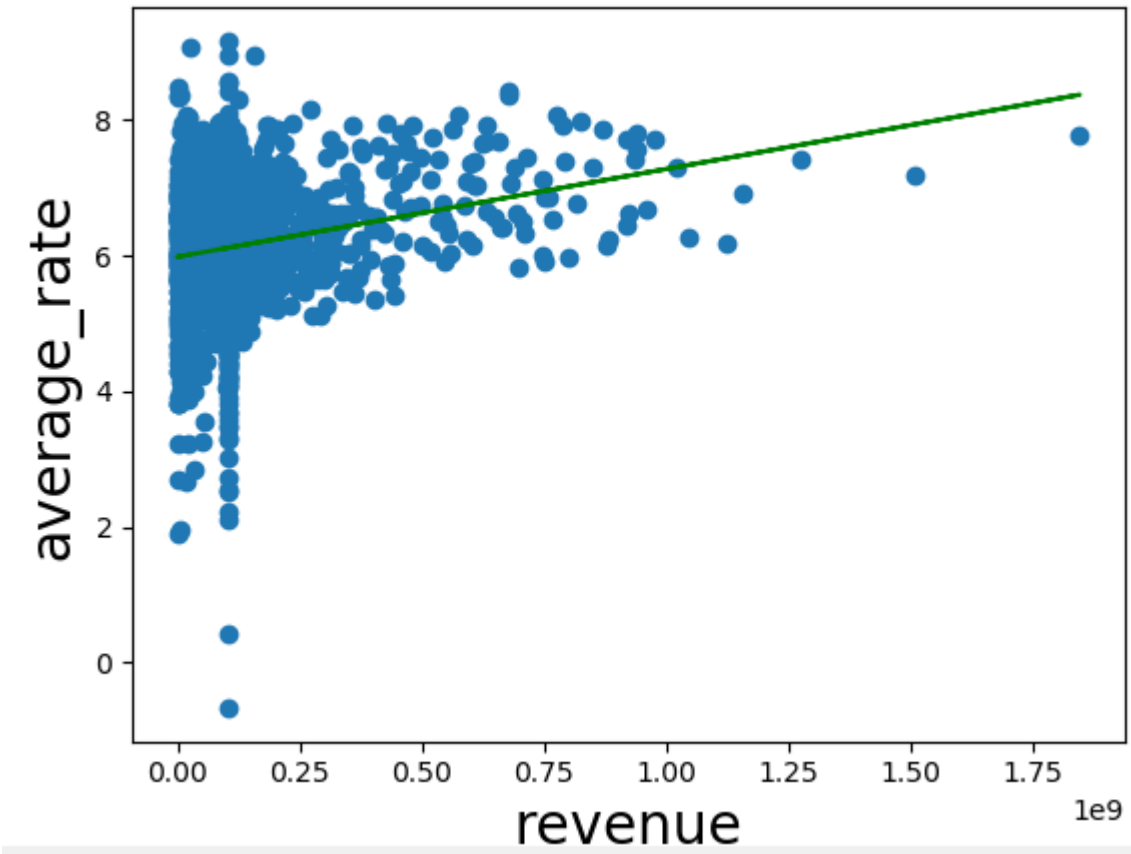

average_rate
release_date


average_rate
crew

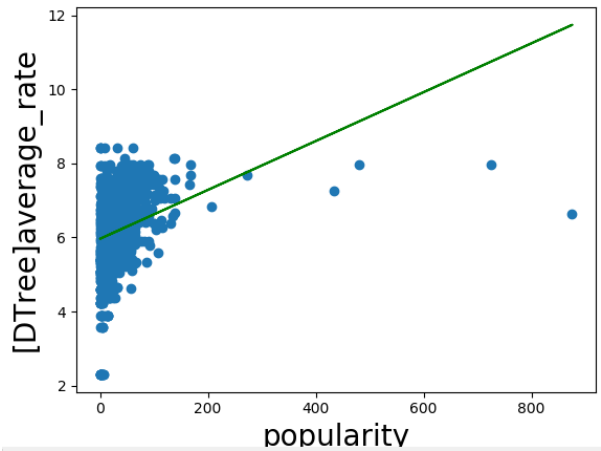

average_rate
vote_count

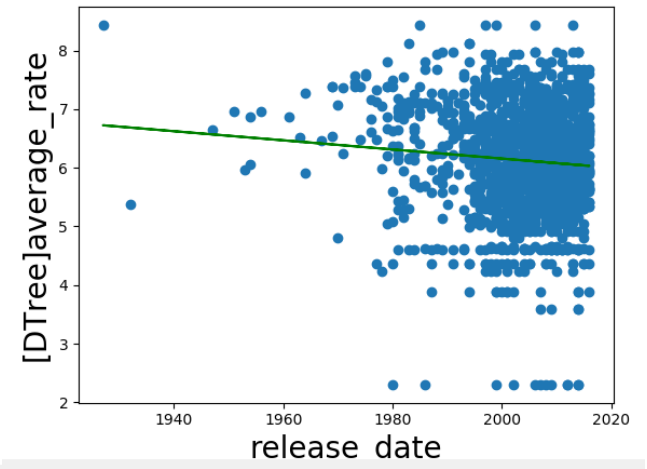

average_rate
production_countries

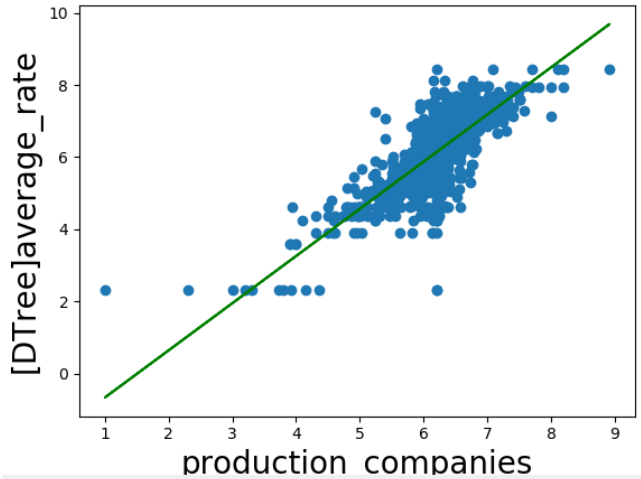

average_rate
spoken_languages

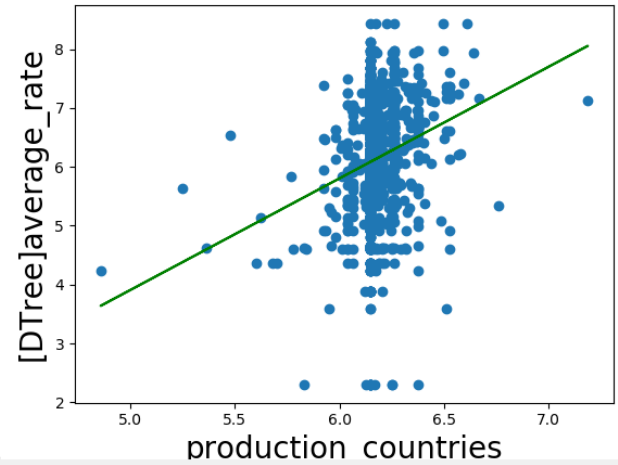

average_rate
revenue
1e9

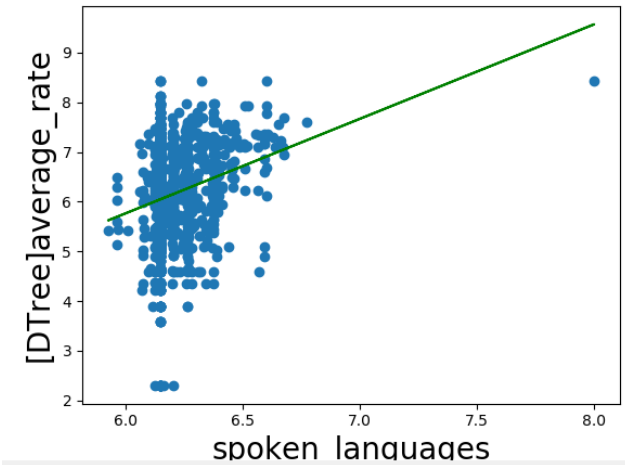
➢ Decision Tree Regressor

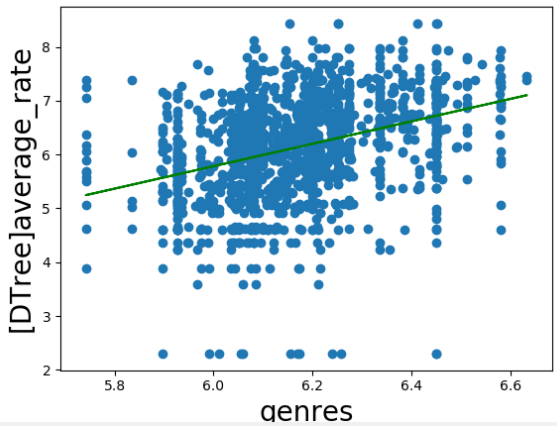


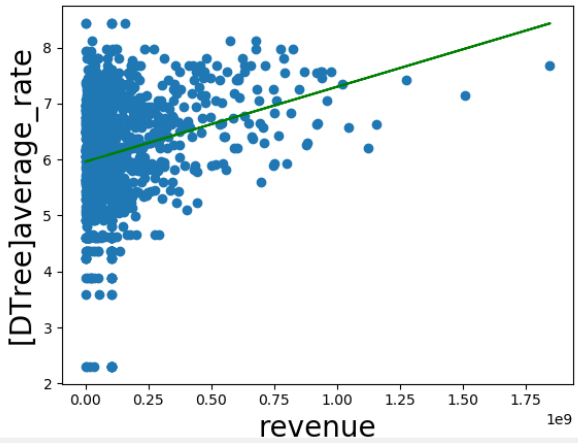


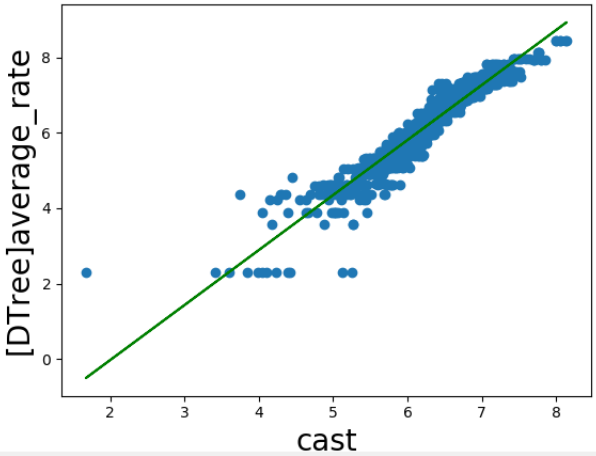


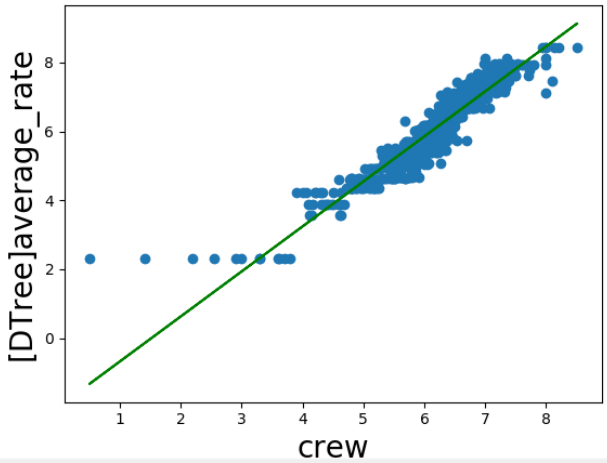


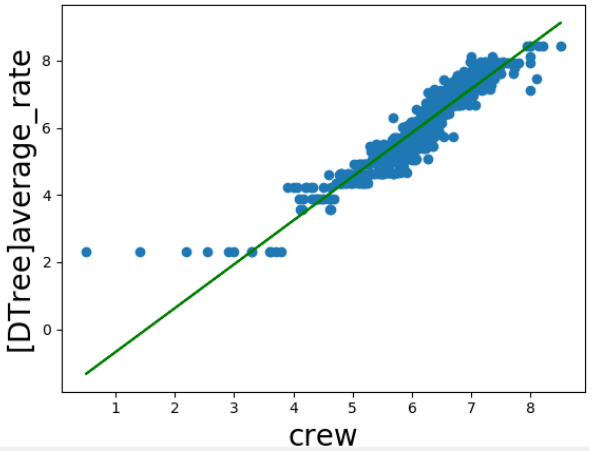


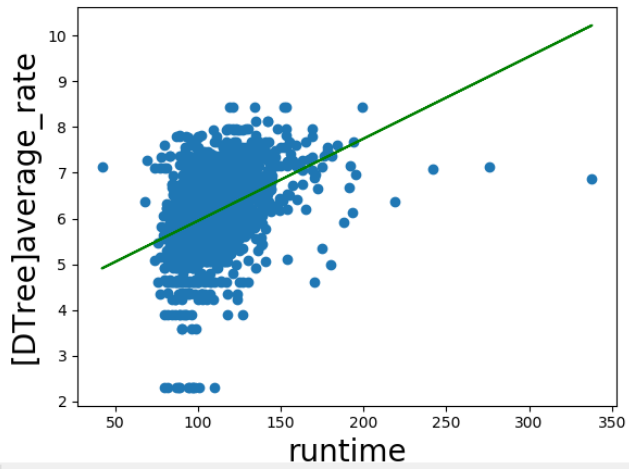


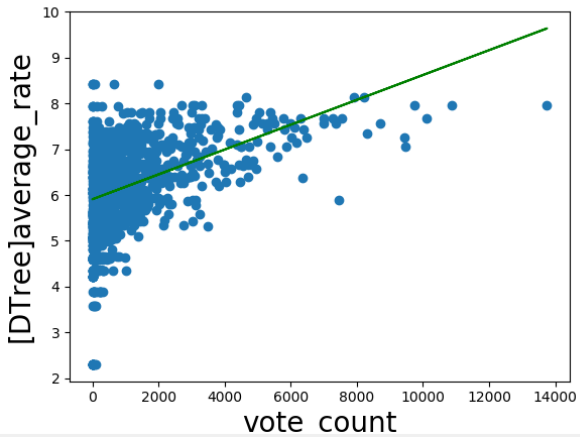

[DTree]average_rate
revenue
1e9

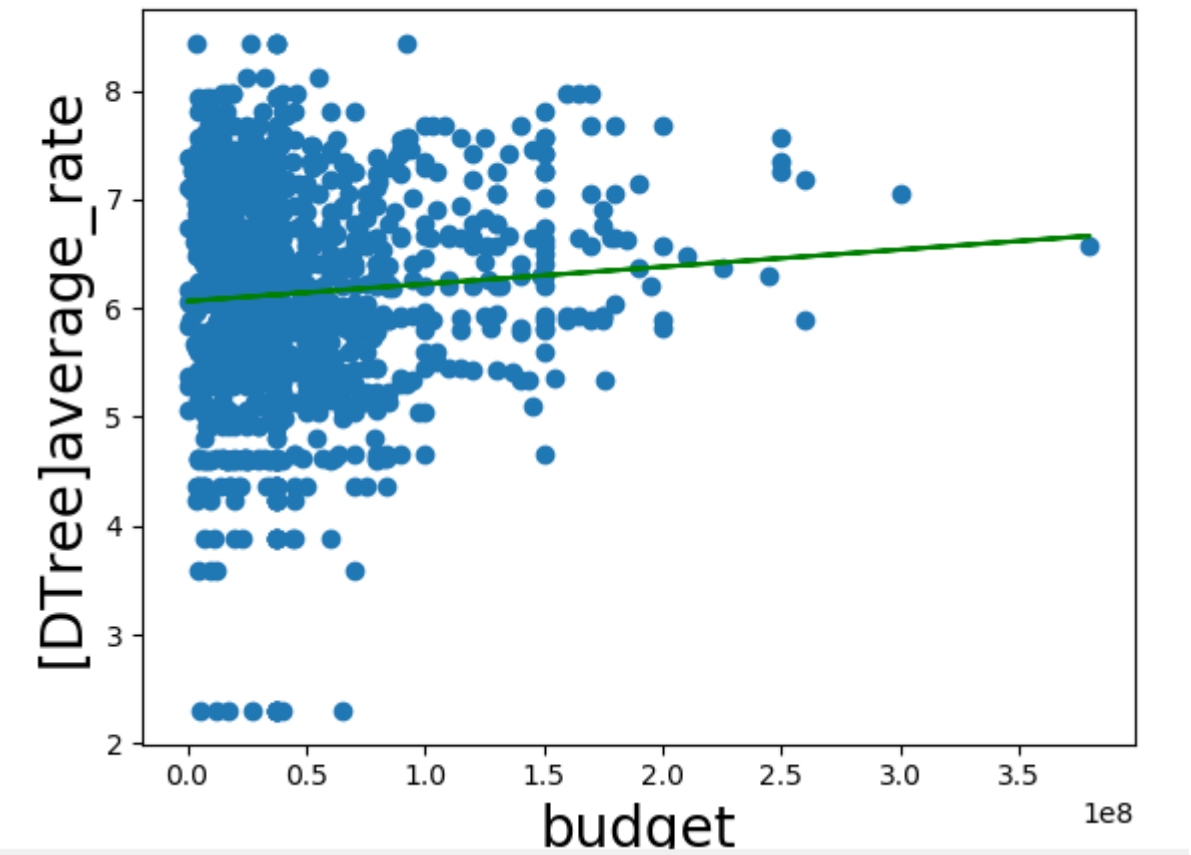

[DTree]average_rate
cast


[DTree]average_rate
crew


[DTree]average_rate
crew


[DTree]average_rate
runtime


[DTree]average_rate
vote_count


[DTree]average_rate
budget
1e8

CONCLUSION

First We Made Preprocessing To Clean Our Dataset To Be Ready Before Use It In Our Models, By One Hot encoded

The Best Model It Was The Decision Tree With MSE = 0.39

We Apply Four Models On Regression Multivariable Linear Regression , Decision Tree Finally We Notice That The Best Model When We Testing Several Parts On Data Is Decision Tree With MSE = 0.39 , Score 0.5.

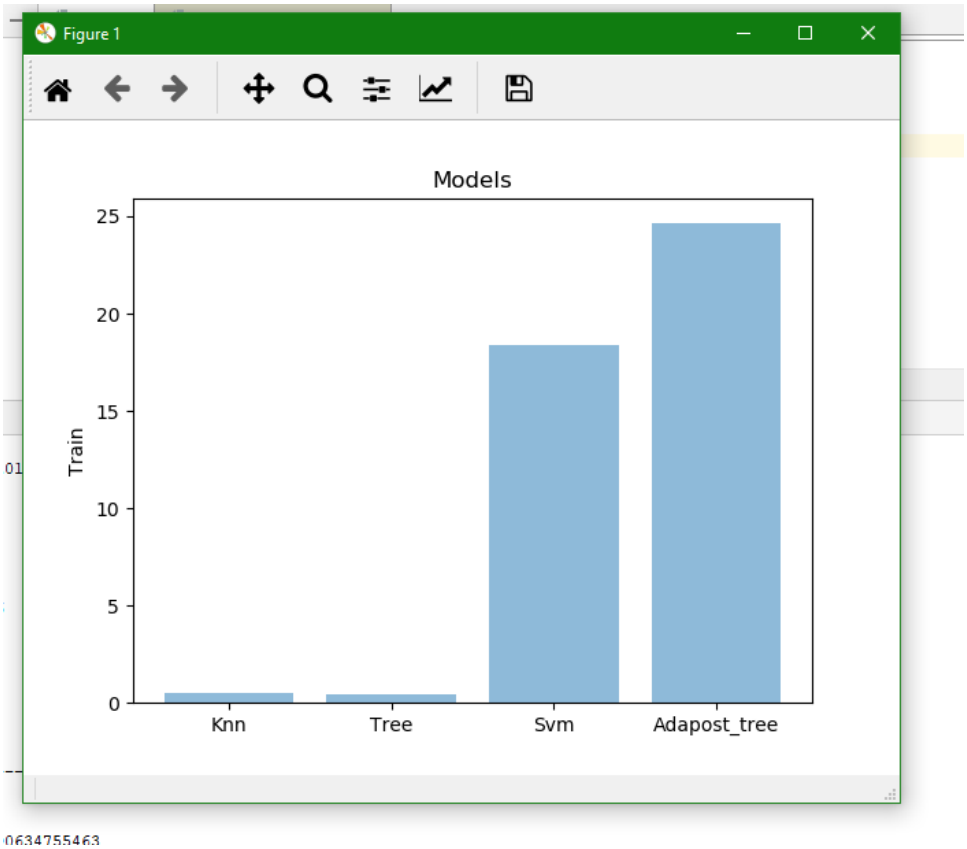
Classification Phase

1- Classification Models

- **Logistic Regression**
- **KNN Classifier.**
- **Decision Tree Classifier.**
- **SVM Classifier.**
- **AdaBoost Classifier**

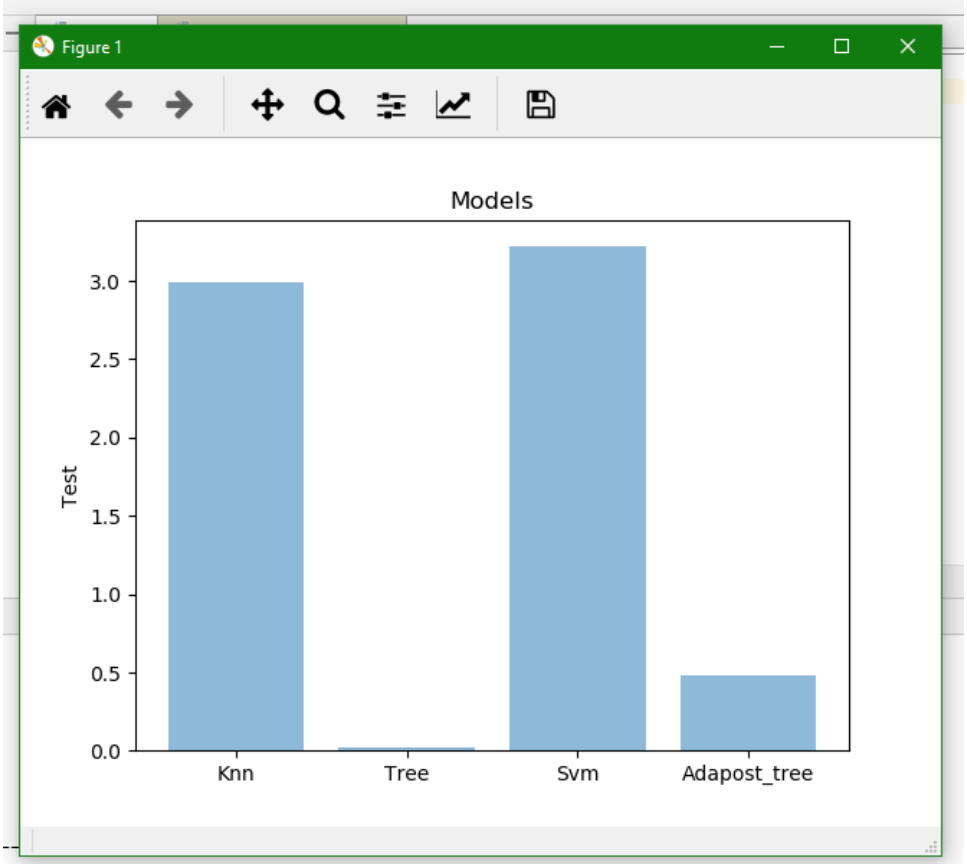
When we Split dataset into 80% training and 20% testing and try previous models the result in (training time, testing time, accuracy) be as mentioned

Total Training Time in seconds

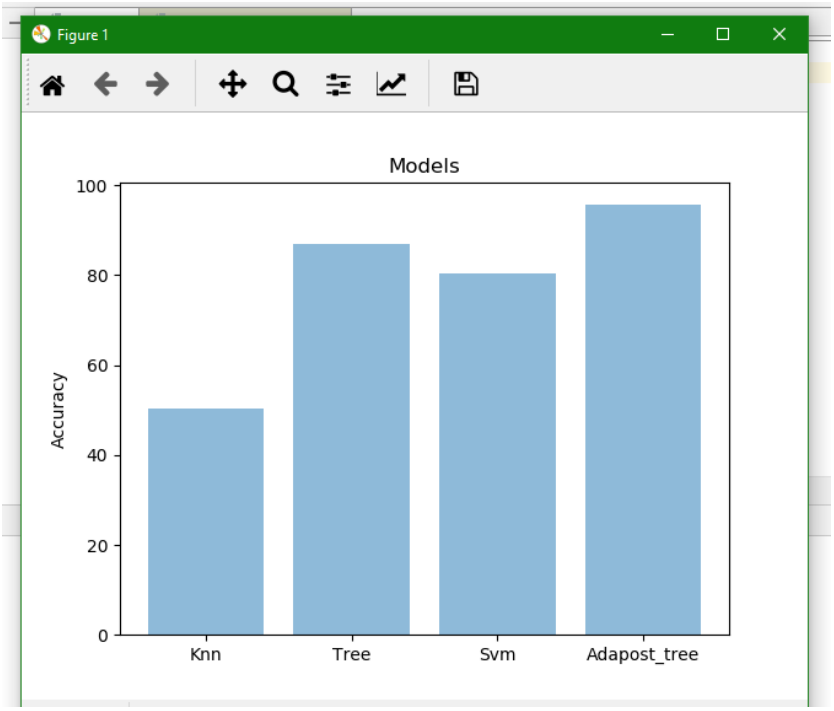


0634755463

Total Testing Time in seconds



Models Accuracy:



2- Hyperparameter tuning:

- **Kernel function.**
- **Max depth.**

We take kernel function in SVM model as a first tuning parameter then we check how this affected your models' performance (training time, test time, Accuracy)

	Linear	**Rbf**	**Sigmoid**
Trian time(second)	22.3	28.830	23.73
Test time(second)	2.651	6.28	5.58
Accuracy (%)	80.64	61.18	61.70

Linear kernel function has least train time and best accuracy

- Second tuning we take max-depth in a decision tree model

	Depth=15	**Depth=5**	**Depth=20**
Trian time(second)	0.834	0.214	0.909
Test time(second)	0.026	0.015	0.022
Accuracy (%)	85.74	60.145	59.83

Depth 15 for Decision Tree has best accuracy.

When the Depth tends to be greater than 15 or less than 10, the accuracy decreases

3- Principle Component Analysis (PCA):

- After Applying PCA With n_components **= 0.98**

  The data features was **(, 566)** has been reduced to **(, 508)**

- After Applying PCA With n_components = **0.90**

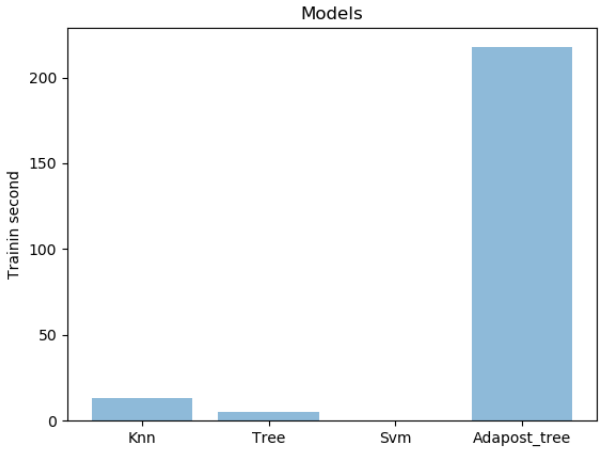
  The data features was **(, 567)** has been reduced to **(, 408)**

- After Applying PCA With n_components = **0.85**

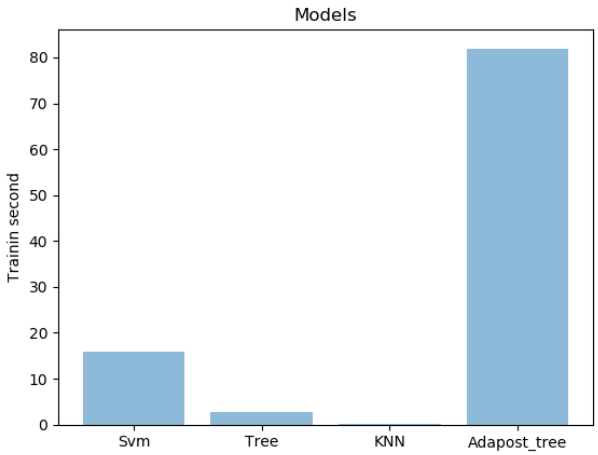
  The data features was **(, 555)** has been reduced to **(, 355)**

Total Training Time for all models (in seconds)

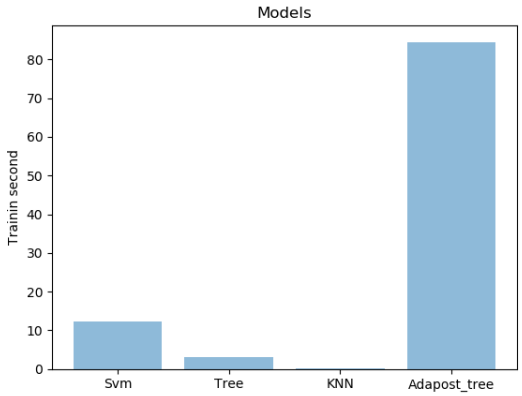
When n_components = 0.98



When n_components = 0.90

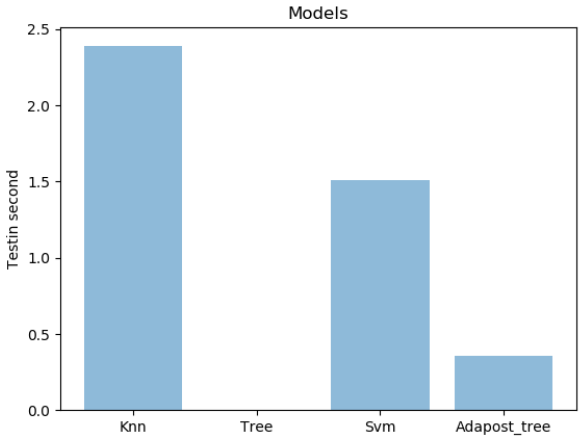


When n_components = 0.85

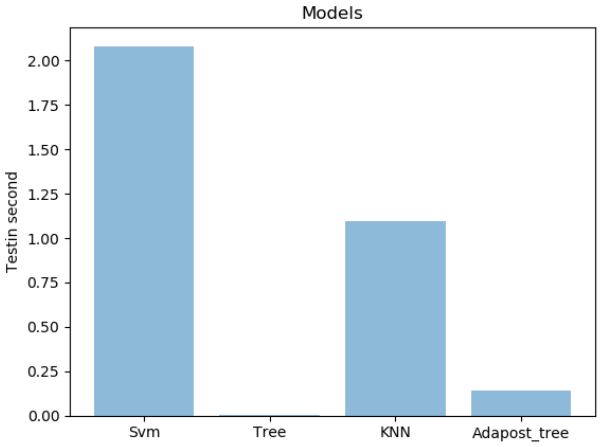


Total Testing Time for all models (in seconds)

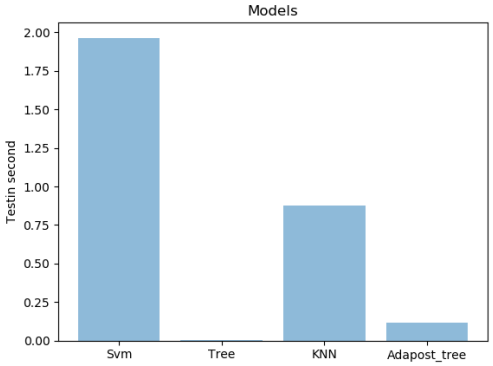
When n_components = 0.98



When n_components = 0.90

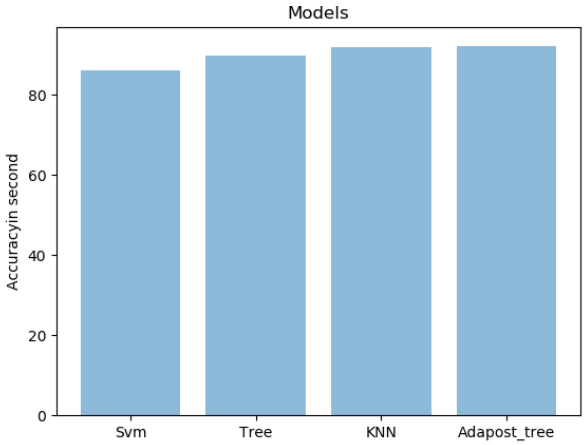


When n_components = 0.85

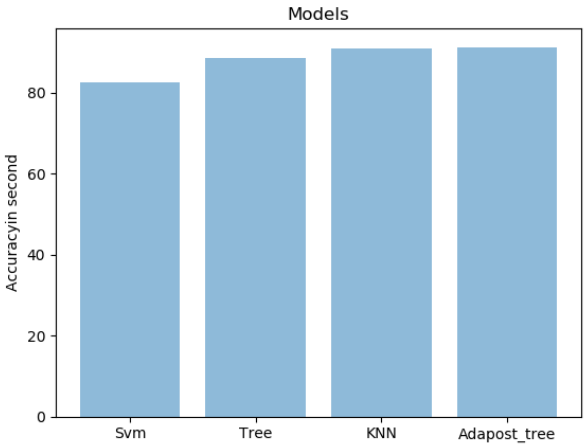


Accuracy

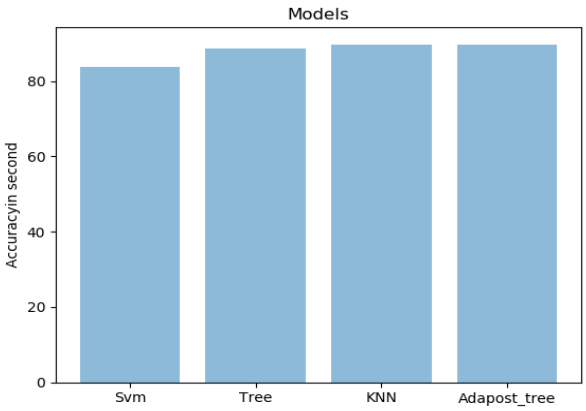
When n_components = 0.98



When n_components = 0.90



When n_components = 0.85



CONCLUSION

After doing one hot encoding in preprocessing, we trained 5 classifier models, we noticed that the validation accuracy was acceptable as SVM Classier with accuracy = 80 %

, 92 % for Adaboost Tree Classifier

We applied PCA on preprocessed data set to reduce the number of features

, we noticed that PCA improved accuracy and fitting time