**Rating Prediction Using Reviews (Yelp Dataset)**

**-Adrita Dutta(axd172930), Chirag Shahi(cxs180005), Pratima(pxl180030), Santhosh Medide(sxm174930)**

**Introduction:**

Yelp is a popular online website that provides review of local businesses based on different parameters. It lets its users review a business based on their own experiences. Over the years Yelp has collected a humungous repository of user reviews which ultimately conclude how a business is doing. Using yelp dataset (provided as part of yelp dataset challenge) as the source, we have tried to target the specific problem of “predicting user ratings based on the review typed by user” via different machine learning algorithms we learnt as a part of our coursework.

We have learnt a lot about different classifiers in our Machine Learning class. We took the dataset from yelp challenge, cleansed the data, did feature engineering and finally applied several machine learning algorithms (tuned different parameters). The content given below contains a detailed explanation of our approach and the results achieved.

**Goal:**

Predicting the ratings given to a business based on the reviews and other useful credentials of the user and the business. This is a project based on a Yelp Challenge.

**Motivation:**

The motivation behind this project is 2 fold:

1) Users sometimes enter wrong ratings by mistake

2) Writing reviews and giving a star rating is repeated work, thus this saves time.

**Dataset used**:

YELP dataset (<https://www.yelp.com/dataset>)

**Original Dataset Description**:

The original dataset contains 6 json files,

1.business.json

2.reviews.json

3.user.json

4.checkin.json

5.tip.json

6.photo.json

**Dataset Components used**:

For the purpose of our project we have used the following json files from the yelp dataset

1.business.json

{**"business\_id", "name”, "neighborhood", "address", "city", "state", “postal code”, “latitude”, "longitude","stars","review\_count","is\_open","attributes”, "categories", "hours"}**

2.reviews.json

{**"review\_id", "user\_id"**, **"business\_id"**, **"stars"**, **"date"**, **"text"**, **"useful"**, **"funny"**, **"cool"**}

3.user.json

{**"user\_id", "name", "review\_count", "yelping\_since", "friends"**, **"useful”, "funny"**, **"cool", "fans", "elite", "average\_stars"**, **"compliment\_hot", "compliment\_more", "compliment\_profile"**, **"compliment\_cute", "compliment\_list", "compliment\_note", "compliment\_plain", "compliment\_cool", "compliment\_funny"**,**"compliment\_writer", "compliment\_photos"}**

**Dataset size**: 3GB

**Final Features Calculated**:

['review\_id', 'review\_stars', 'review\_funny\_upvotes', 'review\_useful\_upvotes', 'review\_cool\_upvotes', 'total\_tokens', 'compound\_score\_review', 'noun\_count', 'pos\_noun\_count', 'neg\_noun\_count', 'neutral\_noun\_count', 'adverb\_count', 'pos\_adverb\_count', 'neg\_adverb\_count', 'neutral\_adverb\_count','verb\_count', 'pos\_verb\_count', 'neg\_verb\_count', 'neutral\_verb\_count', 'adjective\_count', 'pos\_adjective\_count', 'neg\_adjective\_count', 'neutral\_adjective\_count', 'tot\_pos\_words\_count', 'tot\_neg\_words\_count', 'tot\_neu\_words\_count', 'user\_avg\_stars', 'user\_yelping\_since', 'user\_review\_count']

**Final Features used:**

['review\_stars', 'review\_useful\_upvotes', 'review\_cool\_upvotes', 'total\_tokens', 'compound\_score\_review', 'noun\_count', 'verb\_count', 'adjective\_count', 'tot\_pos\_words\_count', 'tot\_neg\_words\_count', 'tot\_neu\_words\_count', 'user\_avg\_stars', 'user\_yelping\_since', 'user\_review\_count']

**Features explanation for used features:**

1. review\_stars:

Found in: review.json

  Explanation: This is what we will use for ground truth

1. review\_useful\_upvotes:

Found in: review.json

Explanation: Upvotes that the review received

1. review\_cool\_upvotes:

Found in: review.json

Explanation: Upvotes that the review received

1. total\_tokens:

Calculated using reviews found in reviews.json

Explanation: Count of number of words in a review

1. compound\_score\_review:

Calculated using reviews found in reviews.json

Explanation: sentimental score of the review

1. noun\_count:

Calculated using reviews found in reviews.json

Explanation: Total no of nouns in review

1. verb\_count:

Calculated using reviews found in reviews.json

Explanation: Total no of verbs in review

1. adjective\_count:

Calculated using reviews found in reviews.json

Explanation: Total no of adjectives in review

1. tot\_pos\_words\_count:

Calculated using reviews found in reviews.json

Explanation: Total no of positive words in review, calculated using sentiment analysis

1. tot\_neg\_words\_count:

Calculated using reviews found in reviews.json

Explanation: Total no of negative words in review, calculated using sentiment analysis

1. tot\_neu\_words\_count:

Calculated using reviews found in reviews.json

Explanation: Total no of neutral words in review, calculated using sentiment analysis

1. user\_avg\_stars:

Found in: review.json

Explanation: Upvotes that the review received

1. user\_yelping\_since:

Found in : user.json

Foreign Key: User\_id

Calculated using the date given in users.json

Explanation: User has been member of yelp (number of days)

1. user\_review\_count:

Found in : user.json

Foreign Key: User\_id

Explanation: Number of reviews given by user

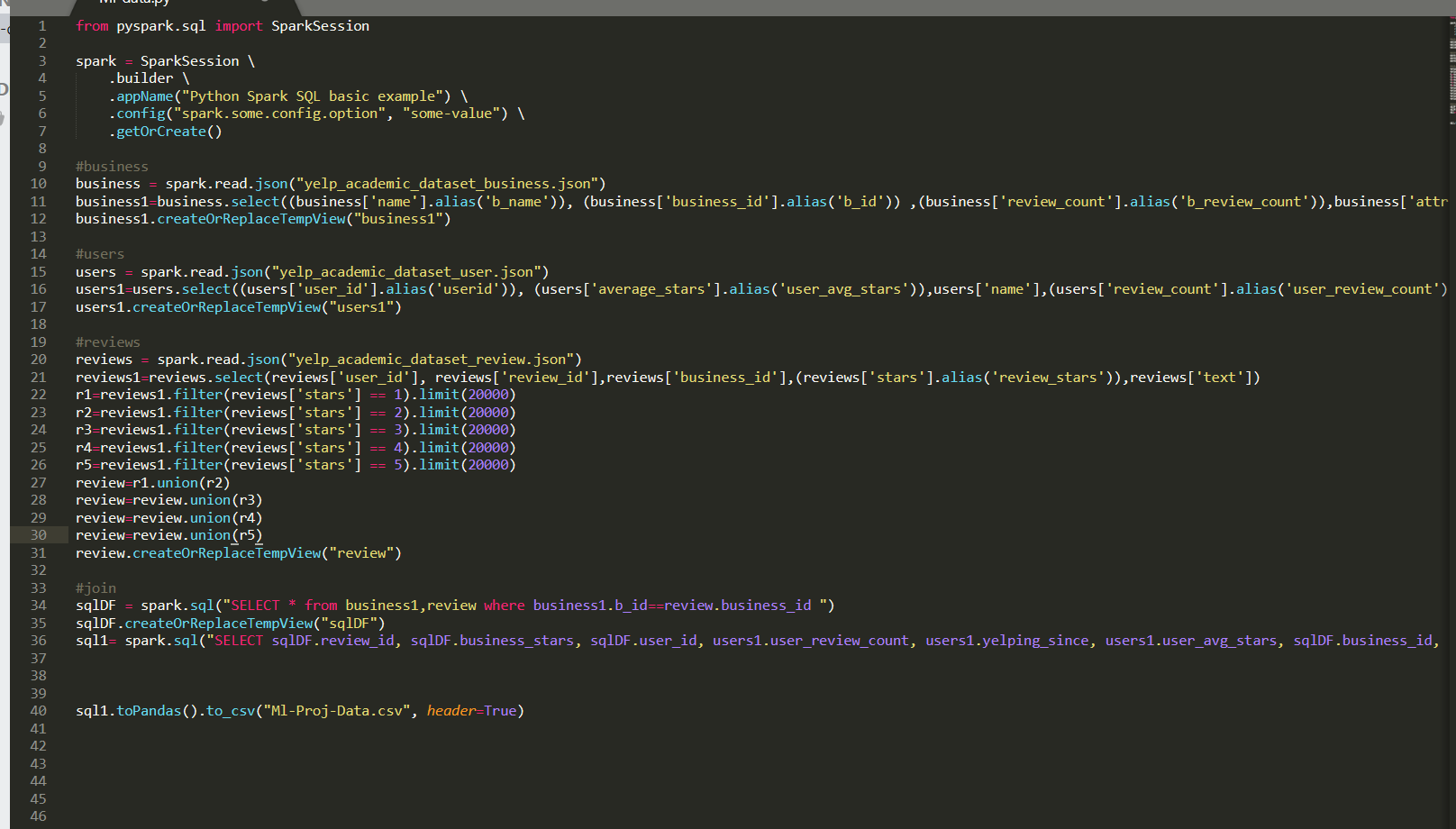
**PRE-PROCESSING THE DATA:**

1. Extraction of relevant data from the 3 datasets (users, business and reviews)

2. Joining the relevant columns using Python and Spark SQL to form a single dataset. The features were extracted in an .csv file.

Features:

* review\_id : ID of the review given by user
* business\_stars : Number of starts business has received on an average
* user\_id: ID of user
* user\_review\_count: Number of reviews given by user
* yelping\_since: Number of days since user has joined yelp
* user\_avg\_stars: Users average rating of all reviews
* business\_id: Business ID for which user has reviewed
* review\_stars : Review stars given by user for the current review ID
* text: Comments given by user for the current review ID
* useful: Number of useful upvotes achieved by review
* funny: Number of funny upvotes achieved by review
* cool : Number of cool upvotes achieved by review



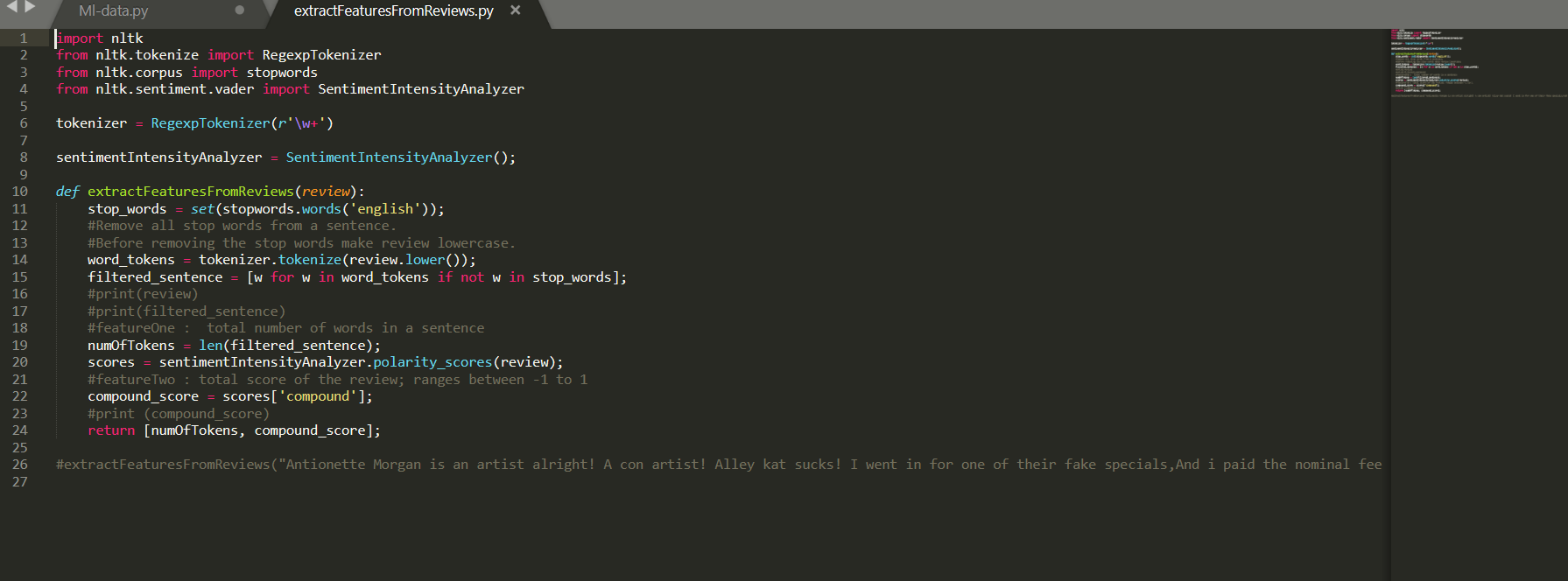
3. Running python scripts the data from the previous step .csv file was extracted, and further sentiment analysis was done over different columns to extract different features. The extracted features were stored in a sql table called: yelpdatasetfeatures.features. Final list of features:

* review\_id: ID of review given by user
* review\_stars: Number of stars given by user in the review (our ground truth)
* review\_funny\_upvotes: Number of funny upvotes achieved by review
* review\_useful\_upvotes: Number of useful upvotes achieved by review
* review\_cool\_upvotes: Number of cool upvotes achieved by review
* total\_tokens: Total number of important words in the review
* compound\_score\_review: Affinity of the review (neutral, negative, positive)
* noun\_count: Total number of noun in review
* pos\_noun\_count': Total number of noun in review with positive affinity
* neg\_noun\_count: Total number of noun in review with negative affinity
* neutral\_noun\_count: Total number of noun in review with neutral affinity
* adverb\_count: Total number of adverb in review
* pos\_adverb\_count: Total number of adverb with positive sentiment in review
* neg\_adverb\_count: Total number of adverb with negative sentiment in review
* neutral\_adverb\_count: Total number of adverb with negative sentiment in review
* verb\_count: Total number of verb in review
* pos\_verb\_count: Total number of verb with positive sentiment in review
* neg\_verb\_count: Total number of verb with negative sentiment in review
* neutral\_verb\_count: Total number of verb with negative sentiment in review
* adjective\_count: Total number of adjective count in review
* pos\_adjective\_count: Total number of adjective with positive sentiment in review
* neg\_adjective\_count: Total number of adjective with negative sentiment in review
* neutral\_adjective\_count: Total number of adjective with neutral sentiment in review
* tot\_pos\_words\_count: Total number of words with positive affinity
* tot\_neg\_words\_count: Total number of words with negative affinity
* tot\_neu\_words\_count: Total number of words with neutral affinity
* user\_avg\_stars: Users average rating of all reviews
* user\_yelping\_since: Number of days since user has joined yelp
* user\_review\_count: Number of reviews given by user

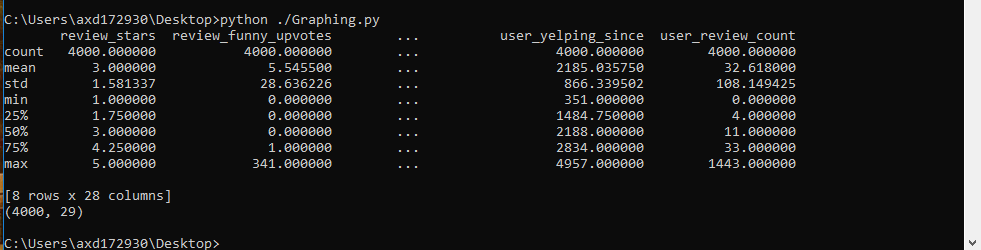
4. Choosing the training set of size 4000 data points and saving the data in MySQL database

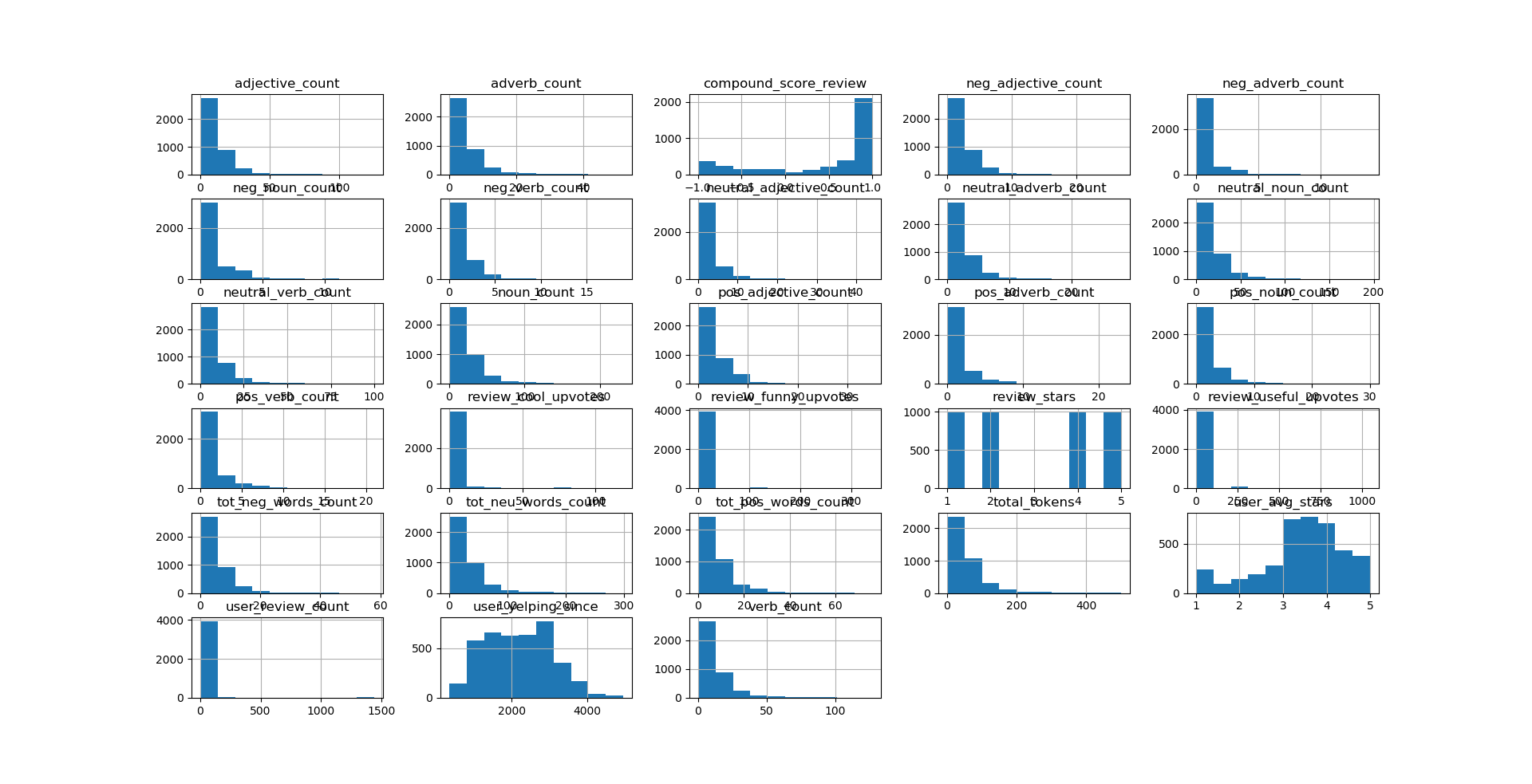
for better querying

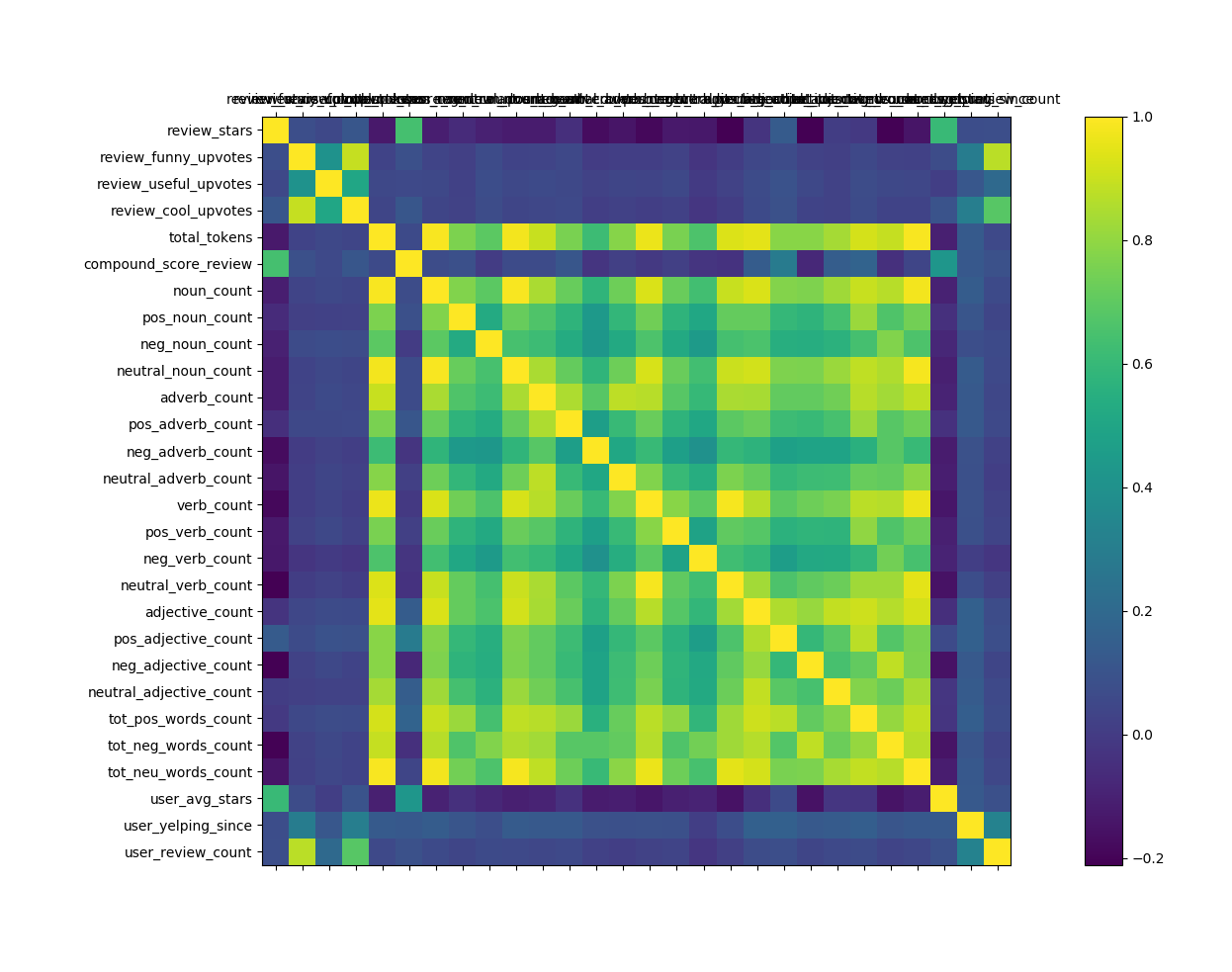
The final training was done on 400000 data points, removing the reviews that had stars=3.



5. Graphed correlation matrix and scatterplot to draw insights from the data.







6. To refine the feature selection, used FeatureSelection library of sklearn.

o F-classif : Compute the ANOVA F-value for the provided sample. ANOVA

stands for Analysis of Variance.

o Select\_K\_best : Computes the best k features

o Chi2 : Compute chi-squared stats between each non-negative feature and

class. This score can be used to select the n\_features features with the highest

values for the test chi-squared statistic from X.

o RFE : Feature ranking with recursive feature elimination.

o Mutual\_info\_classif : Estimate mutual information for a discrete target variable

7. Model Evaluations were done on different models.

**Models Used:**

1)Decision Tree Classifier:

2)Neural Network:

3)K-Nearest Neighbors:

4)Random Forest:

5)Bagging:

6)AdaBoost:

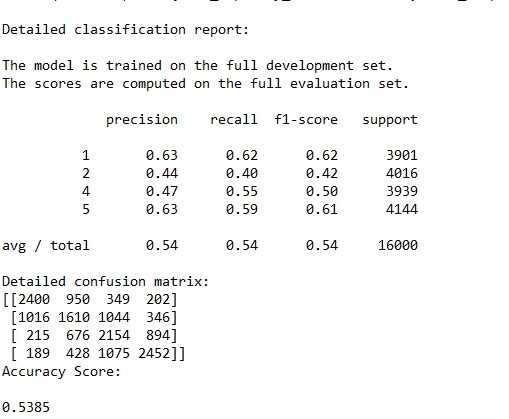
7)Logistic Regression:

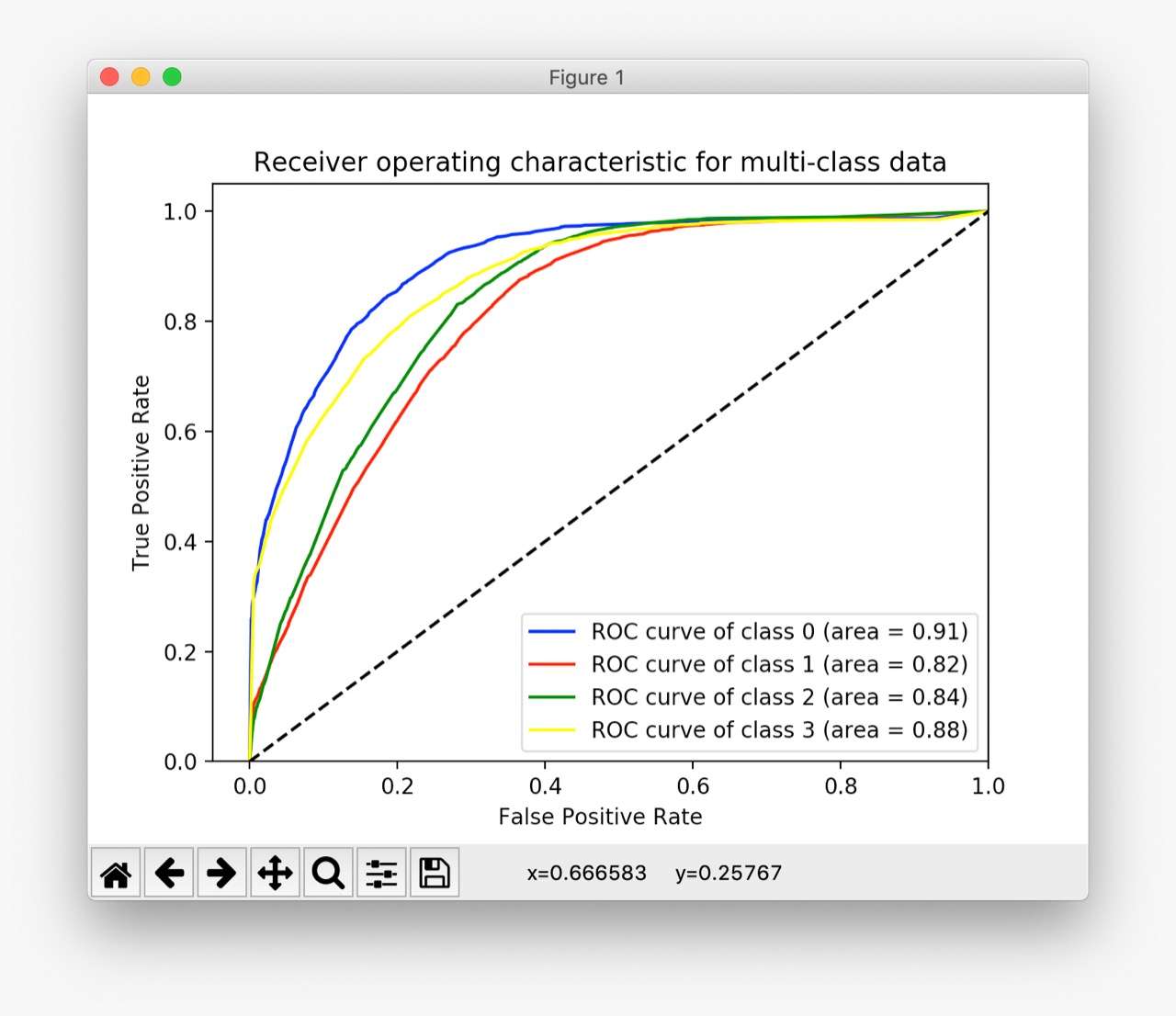
8) Gradient Naïve Bayes:

9)Gradient Boosting:

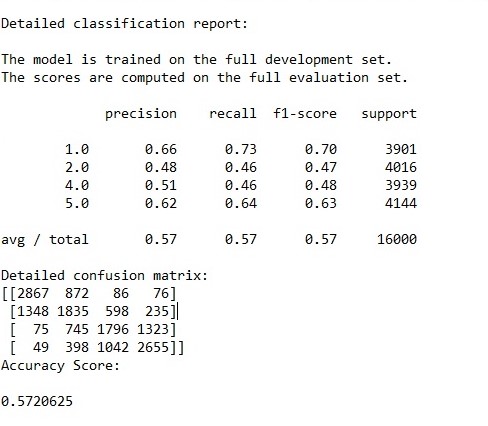
**Experimental Results and Analysis:**

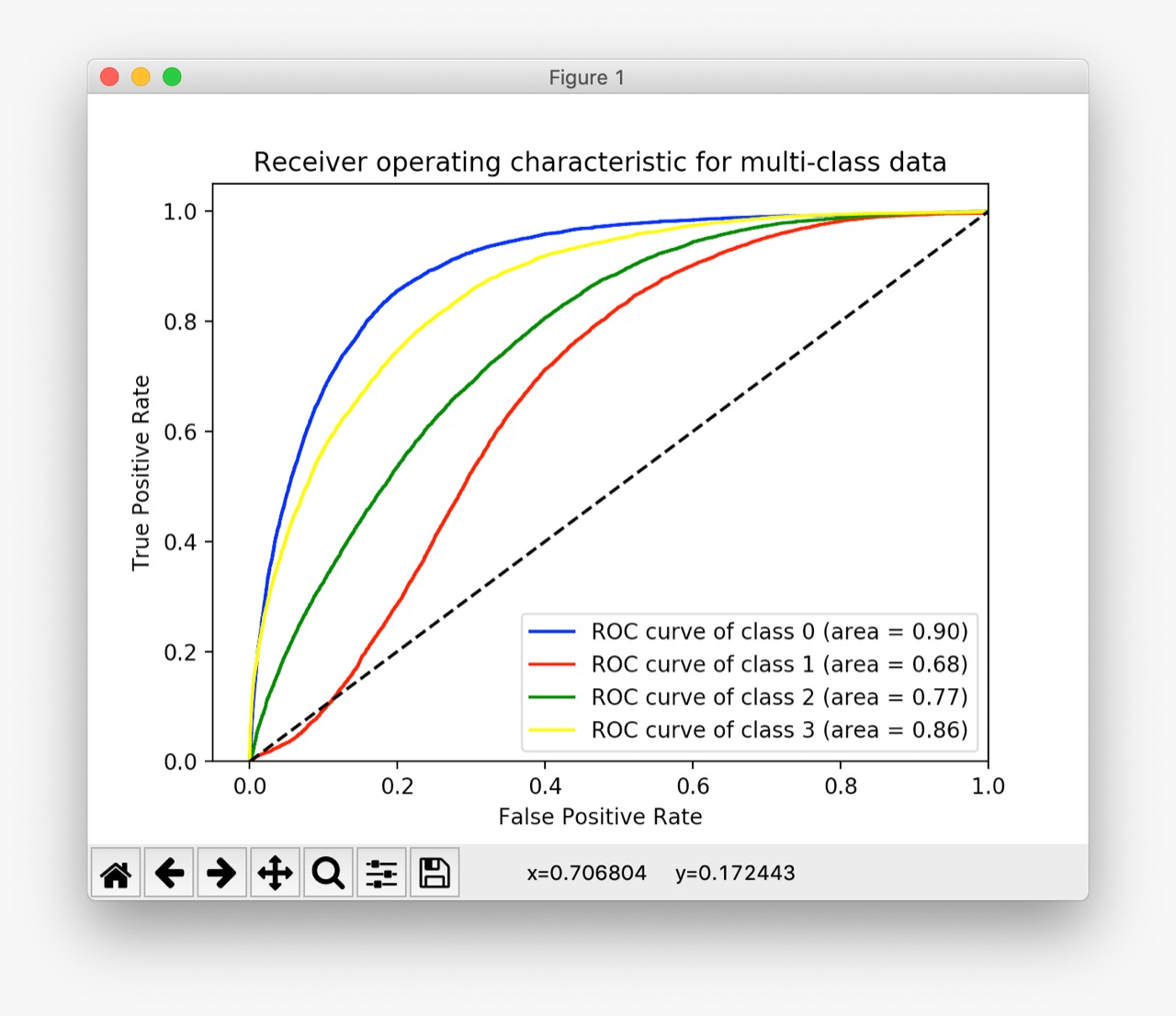
1)Decision Tree:



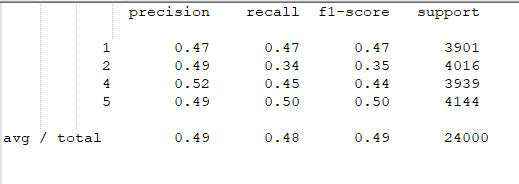


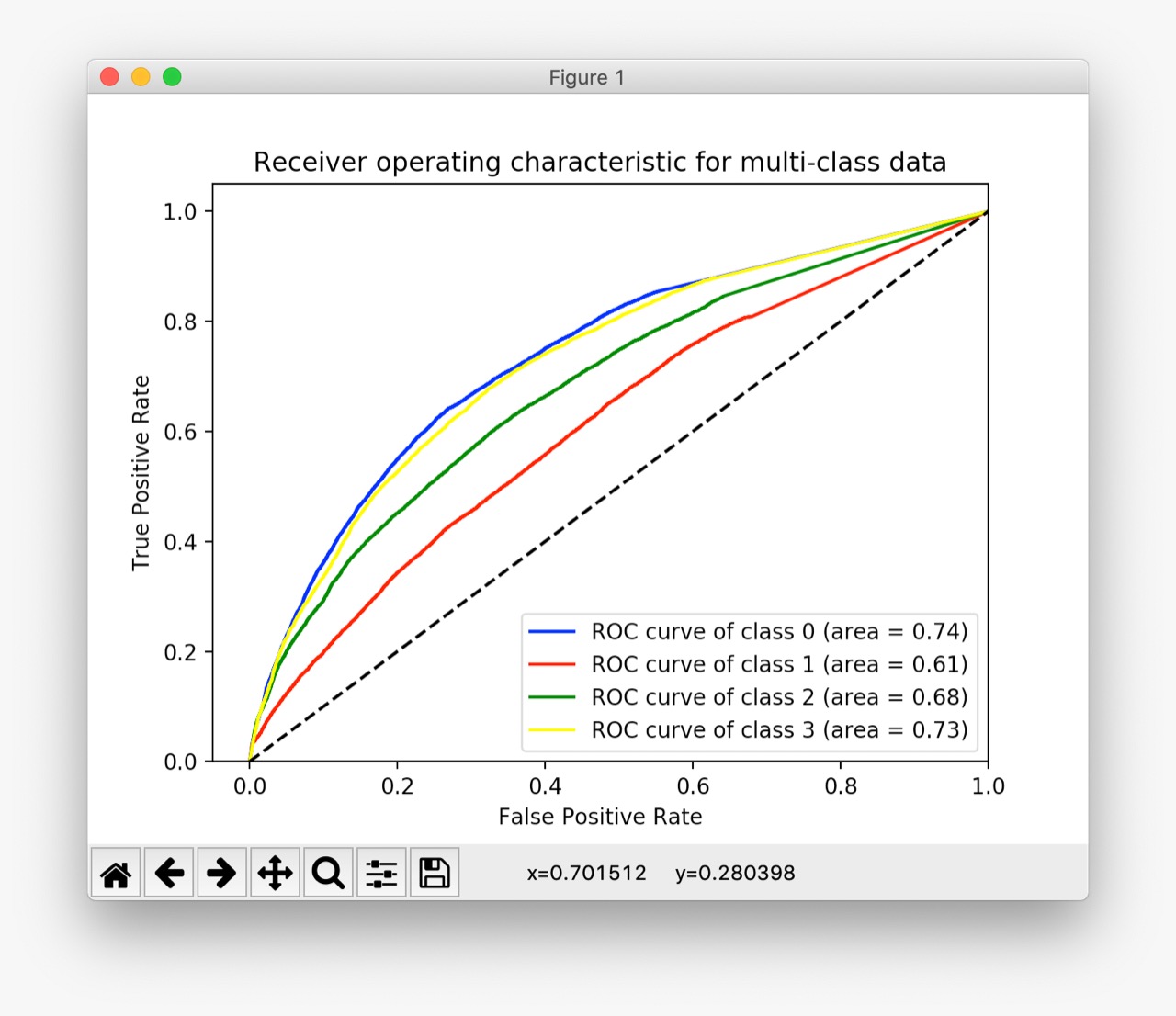
2)Neural Network:



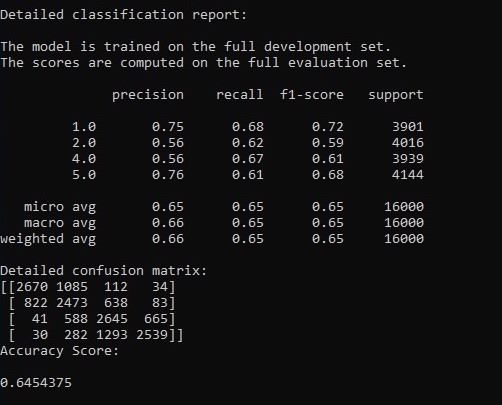


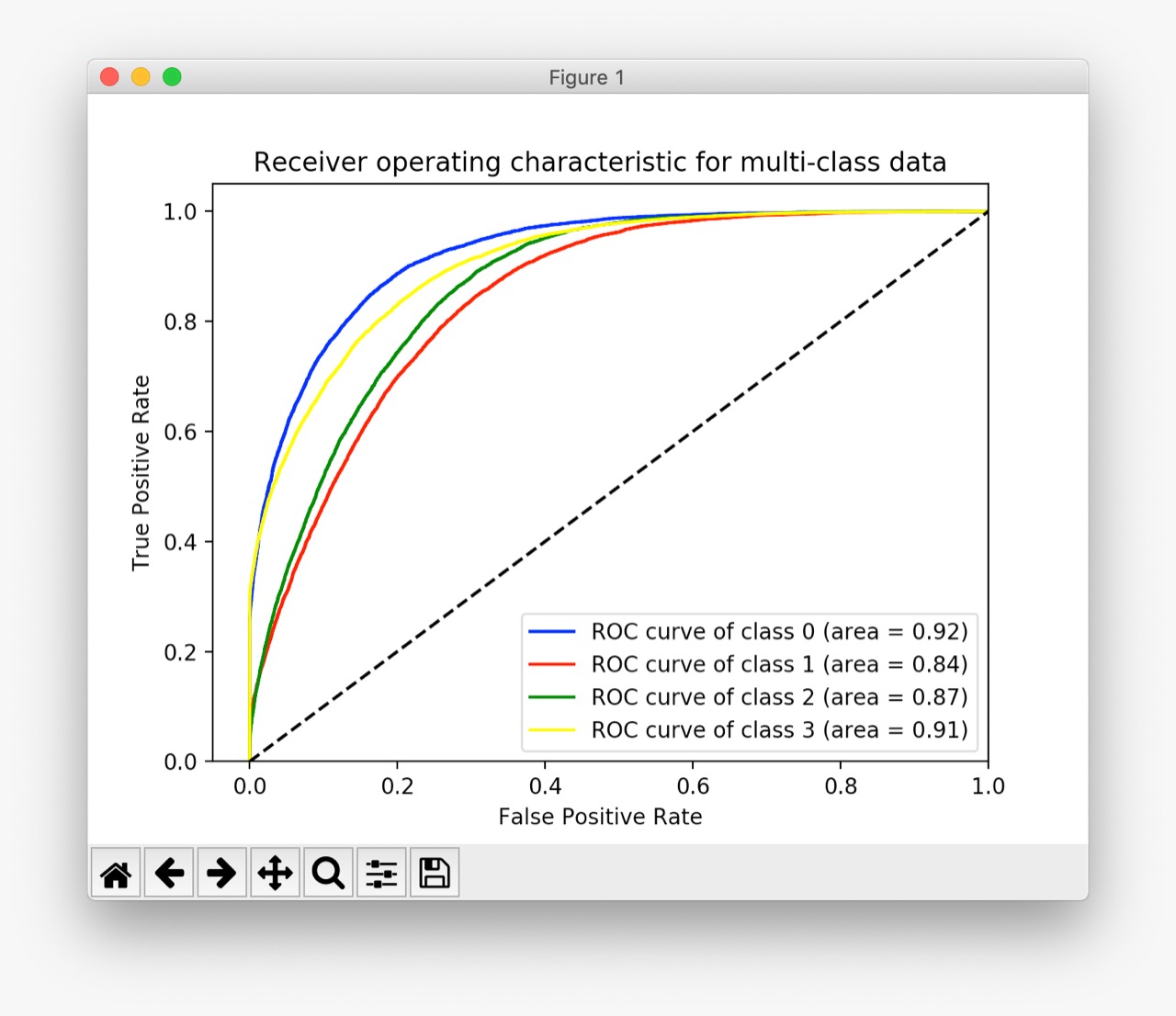
3)KNN:



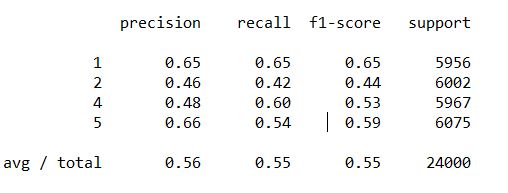


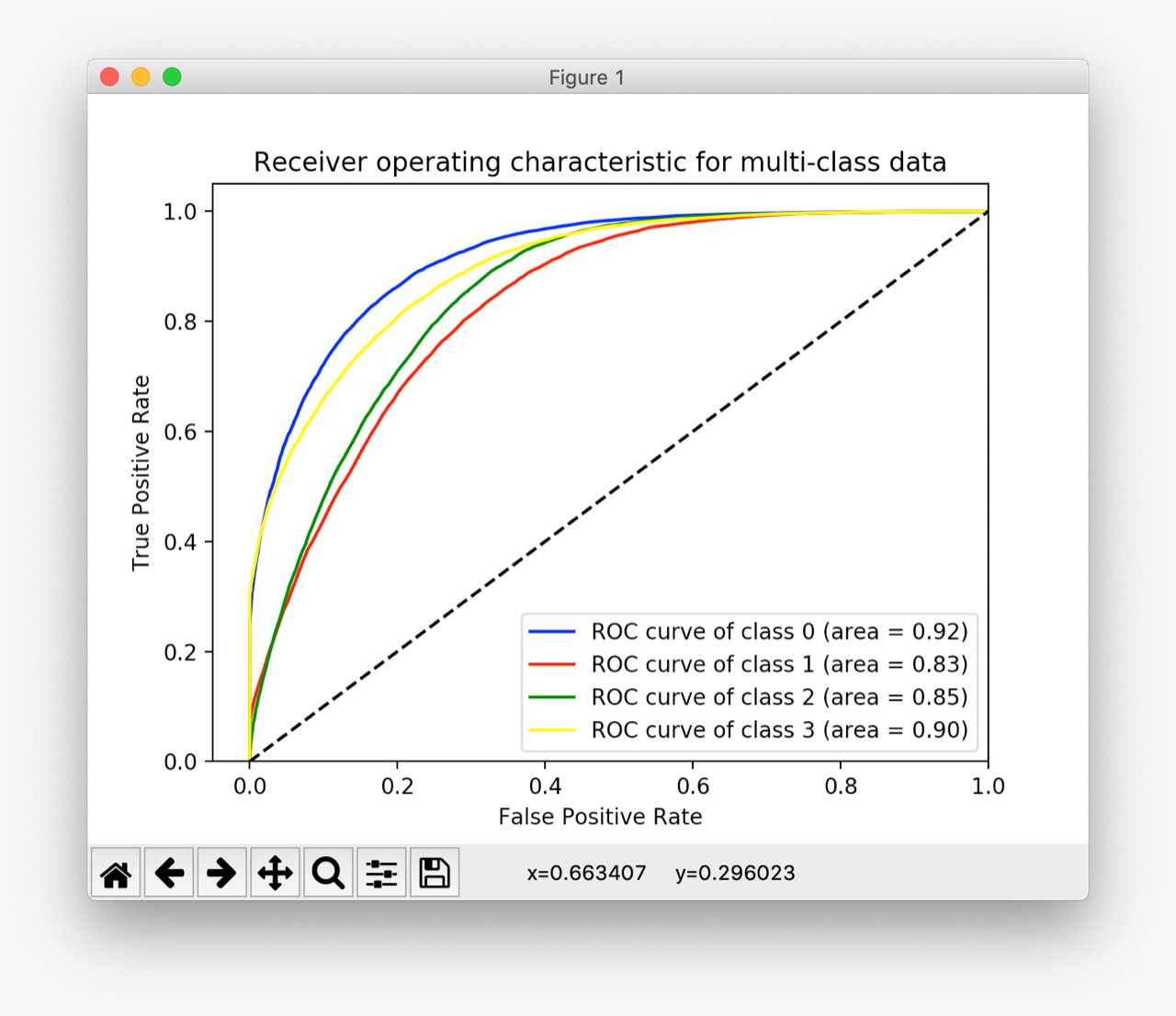
4)Random Forest:



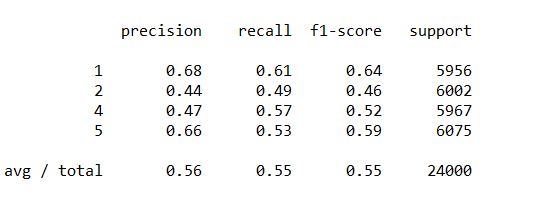


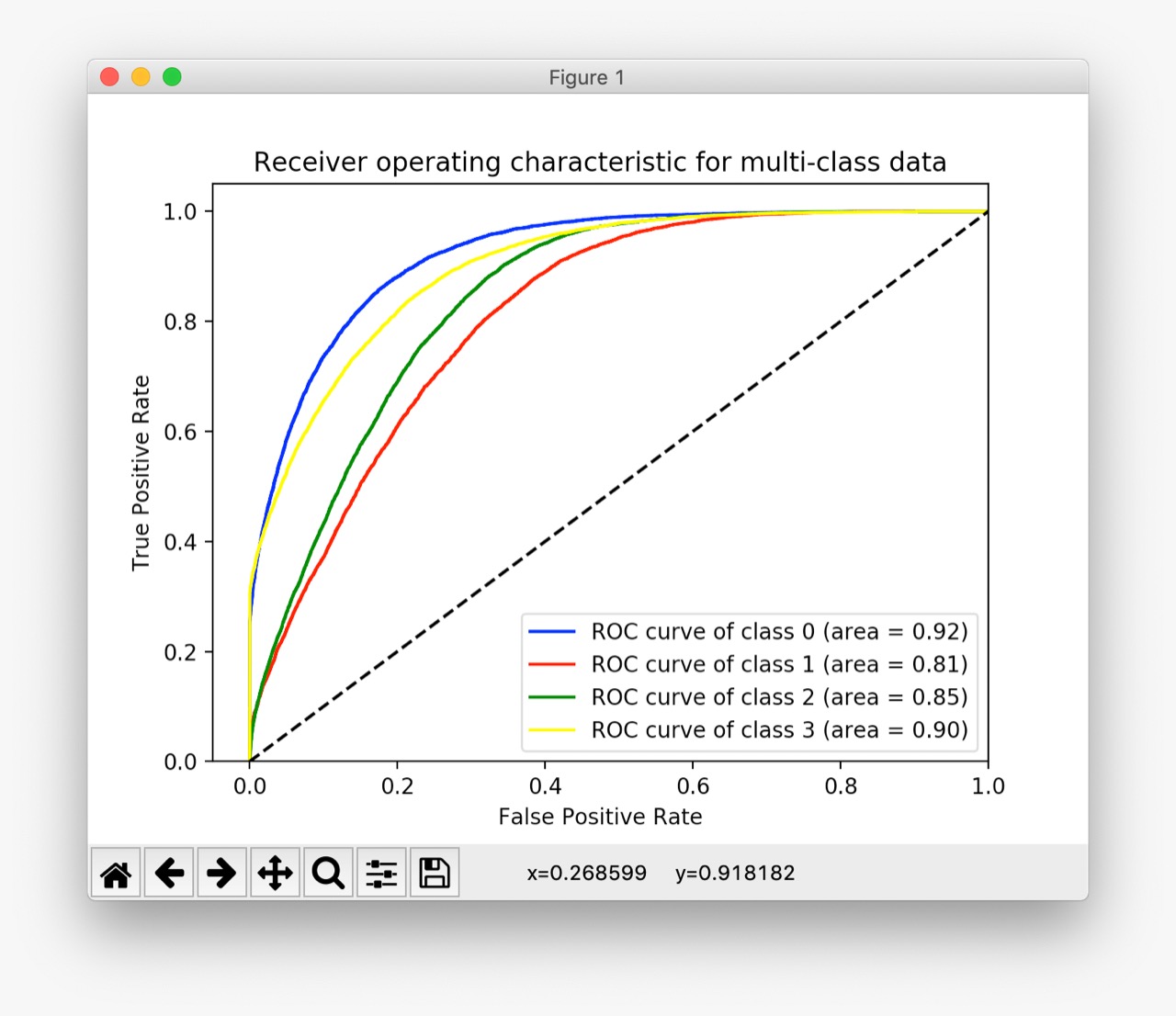
5)Bagging:



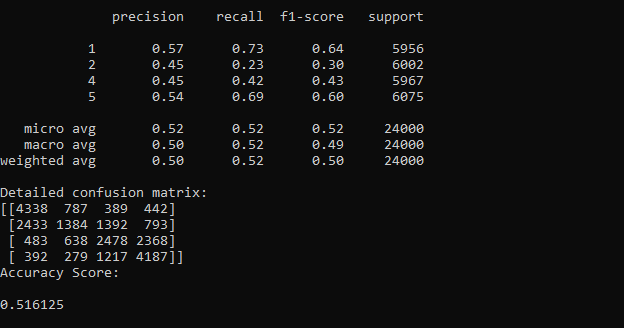


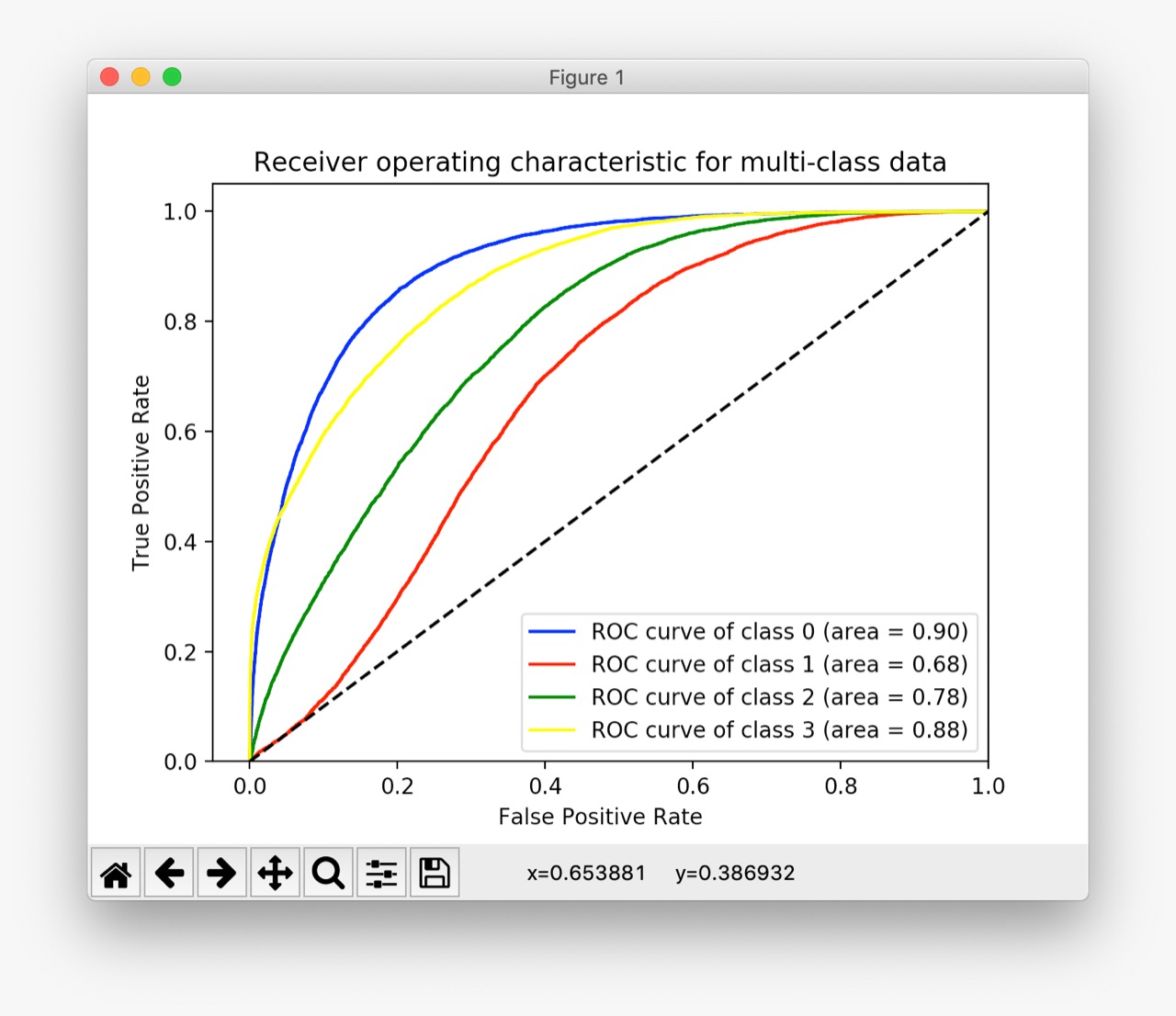
6)AdaBoost:



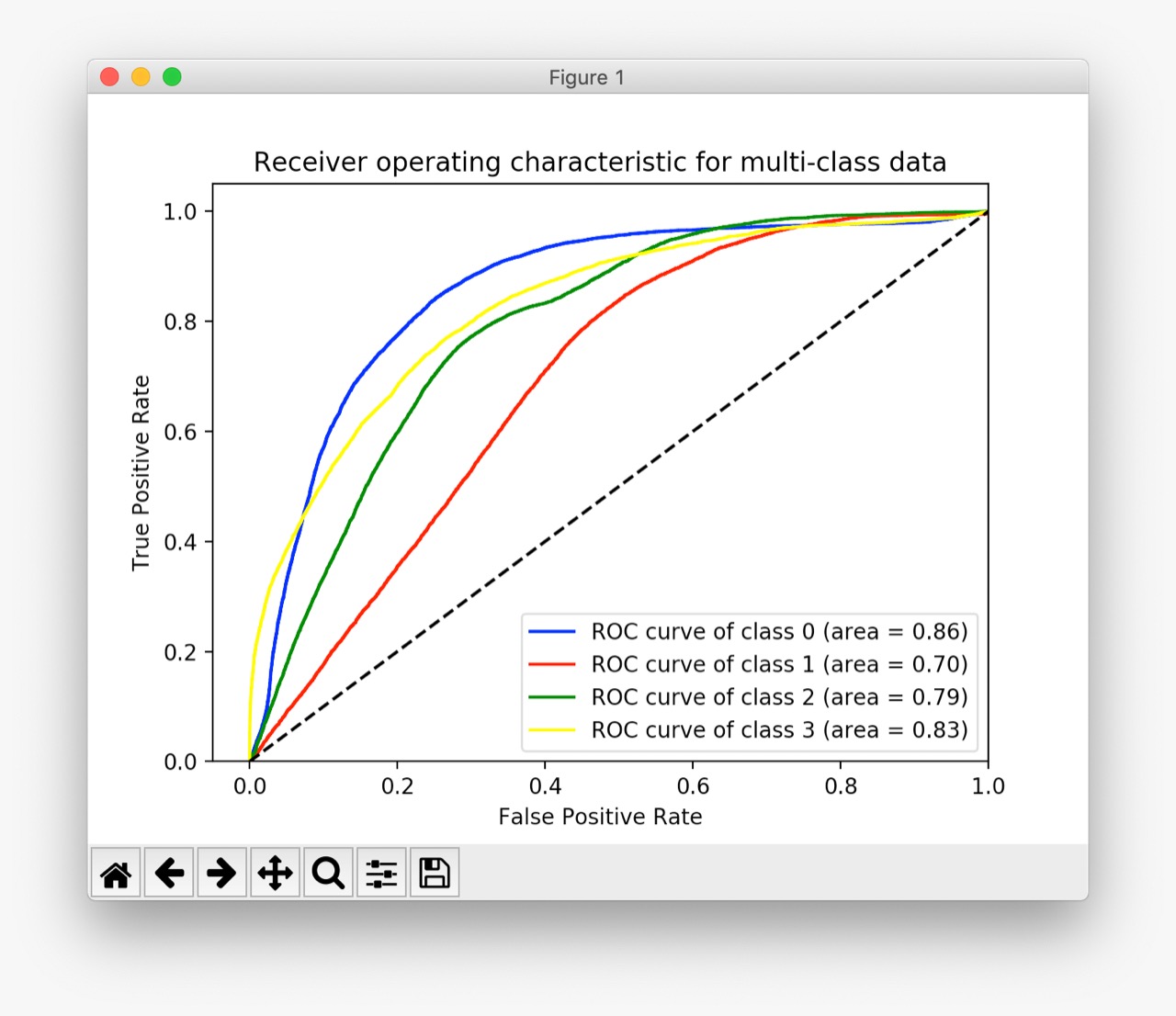


7)Logistic Regression:

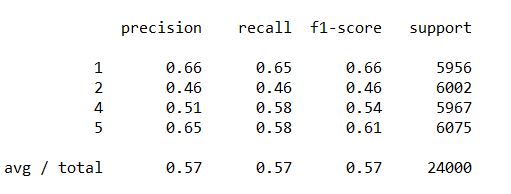


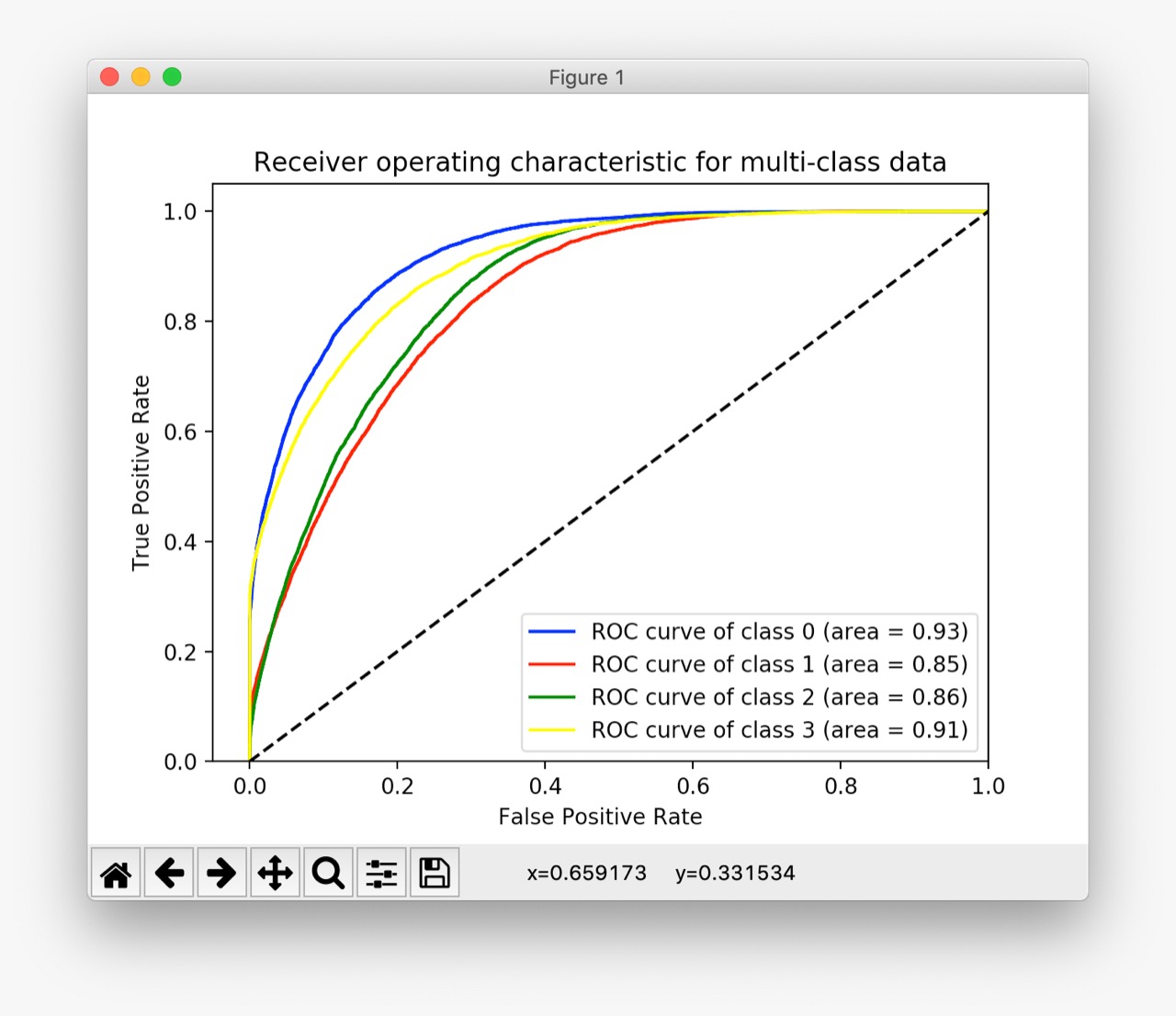


8)Gradiant Naïve Bayes:



9)Gradiant Boosting:





**SUMMARIZING THE RESULTS:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Algorithm | Best Parameters | Avg-Precision | Avg-Recall | Avg-F-1 | Accuracy |
| Decision Tree | {'min\_impurity\_decrease': 0.0, 'min\_samples\_leaf': 3, 'min\_samples\_split': 2, 'random\_state': 2, 'max\_features': None, 'max\_depth': 10} | 0.54 | 0.54 | 0.54 | 0.5385 |
| Neural Network | {'random\_state': 5, 'activation': 'identity', 'max\_iter': 100, 'learning\_rate': 'constant', 'hidden\_layer\_sizes': (10, 2)} | 0.57 | 0.57 | 0.57 | 0.5720625 |
| KNN | {'algorithm': 'brute', 'n\_neighbors': 5, 'p': 1, 'weights': 'distance'} | 0.49 | 0.48 | 0.48 | 49.1 |
| Random Forest | {'criterion': 'gini', 'max\_depth': 90, 'min\_samples\_split': 30, 'n\_estimators': 80} | 0.66 | 0.65 | 0.65 | 0.6454375 |
| Bagging | {'max\_features': 12, 'max\_samples': 600, 'n\_estimators': 190, 'random\_state': 1} | 0.56 | 0.55 | 0.55 | 0.5536666666666666 |
| AdaBoost | {'algorithm': 'SAMME.R', 'learning\_rate': 0.5, 'n\_estimators': 225, 'random\_state': 10} | 0.56 | 0.55 | 0.55 | 0.5487916666666667 |
| Logistic  Regression | {'C': 0.1, 'fit\_intercept': 'True', 'max\_iter': 100, 'penalty': 'l2'} | 0.50 | 0.52 | 0.50 | 0.516125 |
| Gradiant Naïve Bayes | {‘priors’:(0.05,0.24,0.7,0.01)} | 0.5 | 0.52 | 0.49 | 0.5168333333333334 |
| Gradiant Boosting | {'loss': 'deviance', 'max\_features': 'auto', 'n\_estimators': 110, 'random\_state': 3} | 0.57 | 0.57 | 0.57 | 0.5665416666666667 |

**Conclusion:**

On trying a few different combinations for feature sets and observing correlation matrix, histograms, RFE classification scores and other metrics a set of 13 features.

It can be observed from the above data that the **best result of prediction is obtained in case of the Random Forest Classifier** both on the basis of ROC curve and Accuracy score.

The classifiers ranked in the order of decreasing accuracy:

1) Random Forest: 0.6454375

2)Neural Network: 0.5720625

3) Gradiant Boosting: 0.5665416666666667

4) Bagging: 0.5536666666666666

5) AdaBoost: 0.5487916666666667

6) Decision Tree Classifier: 0.5385

7) Gradiant Naïve Bayes: 0.5168333333333334

8)Logistic Regression: 0.516125

9) KNN: 49.1

**References:**

1. <https://stackoverflow.com/questions/50941223/plotting-roc-curve-with-multiple-classes>
2. <https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html>
3. <https://cseweb.ucsd.edu/classes/wi17/cse258-a/reports/a041.pdf>