Prediction on the rating of fine food and evaluation for the helpfulness of review based on the Amazon Fine Food Review

Capstone Project by Liyan Cen

1 INTRODUCTION

Customer reviews is an important part of the Amazon shopping experience. It is so far the major reference and indicator for Amazon to evaluate the quality of their products, and the performance of their sellers.

Additionally, it is a great tool for customers to compare between products. Amazon always tend to place the most helpful reviews at the first place of the list as a reference for customers, and also as a standard for reviewers to follow. The company carefully considers any changes to the ratings and review system to ensure customers and sellers continue to trust it.

In order to improve the quality of the review system, Amazon started to distribute free products to quality reviewers years ago. To be more specific, Amazon sends free products to their top reviewers. Once received, all they need to do is to review the product on the website, and then keep the product themselves. Amazon receives a ranking that says how helpful review was to potential consumers. The more helpful the review is, the more likely the reviewer could become a top reviewer.

In our project, we divided up the analysis and prediction into two parts with two iPython notebooks, they are “Ratings vs. Reviews”, and “Helpfulness vs. Reviews”. For “Ratings vs. Reviews” section, we use classification models to examine ratings based on the reviews, and to predict the pattern of how the reviewers rate products based on written comments. This model could help amazon to evaluate the reasonableness of the ratings. For “Helpful vs. Reviews” section, we use classification models as well to predict which review is more helpful.

2 Data Acquisition and Cleaning

This dataset consists of reviews of fine foods from amazon. The data span a period of more than 10 years, including all ~500,000 reviews up to October 2012. Reviews include product and user information, ratings, and a plain text review.

Because there is memory limitation of our CPU, we first cross out the “HelpfulnessDenominator” value that is smaller than 10. Also, we cross out all rows which has null values, and make sure there is no duplicate contents. As a result, the length of data changes from 568,454 to 21,463.

In order to make prediction models with better performance along with less noises or outliers, we first reprocess the text to make our feature set more correlated with the target variable. By doing that, we create a function called “normalize_corpus” to remove punctuations, digits, html tags, accented characters, special characters,and stop words. At the same time, we lower all strings of each text.

For both the classification and regression models, we use text as our feature set by converting each word from each text into numerators by “TfidfVectorizer”. Frequent words across all documents may tend to overshadow other terms in the feature set. The TF-IDF model is functioned to solve this issue by using a scaling or normalizing factor in its computation.

2.1 Data Wrangling and Cleaning - “Ratings vs. Reviews” section

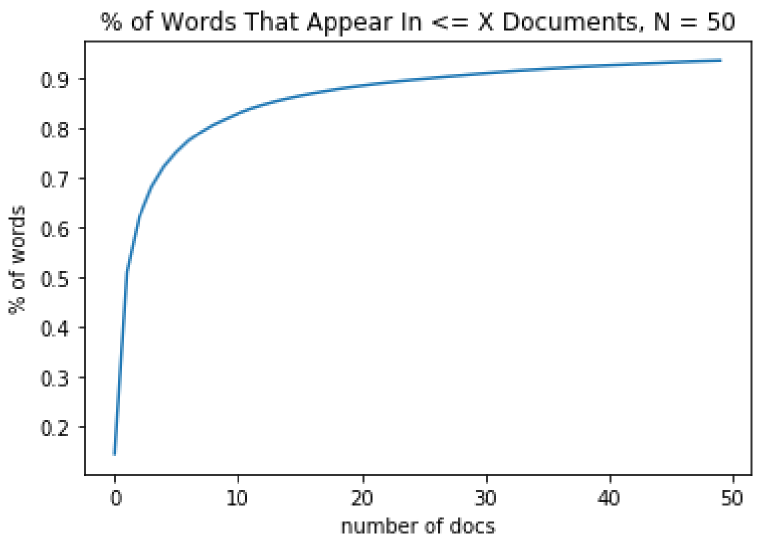
For this section, we use text feature as our feature set, and score as our target variable. Classification models are used because our model predicts and classifies reviews into a score ranging from 1 to 5.

2.2 Data Wrangling and Cleaning - “Helpfulness vs. Reviews” section

For this section, we use text feature as our feature set, and “Helpful” as our target variable. We get “Helpfulness ratio” by having “HelpfulnessNumerator” divided by “HelpfulnessDenominator”. Classification models are used because our model predicts and classifies reviews into either “0” or “1”, while ratio greater than the mean will be equal to “1”, and less than the mean will be equal to “0”.

More details on the data cleaning process can be found from the IPython notebook. The cleaned dataset is then ready for explorations.

3 Data Exploration



Based on the graph, we can see that around 70% of the text features appeared in about 4 or 5 documents, so we set our min_df to be “5”, which means we cut off the words that occur in 5 documents or less.

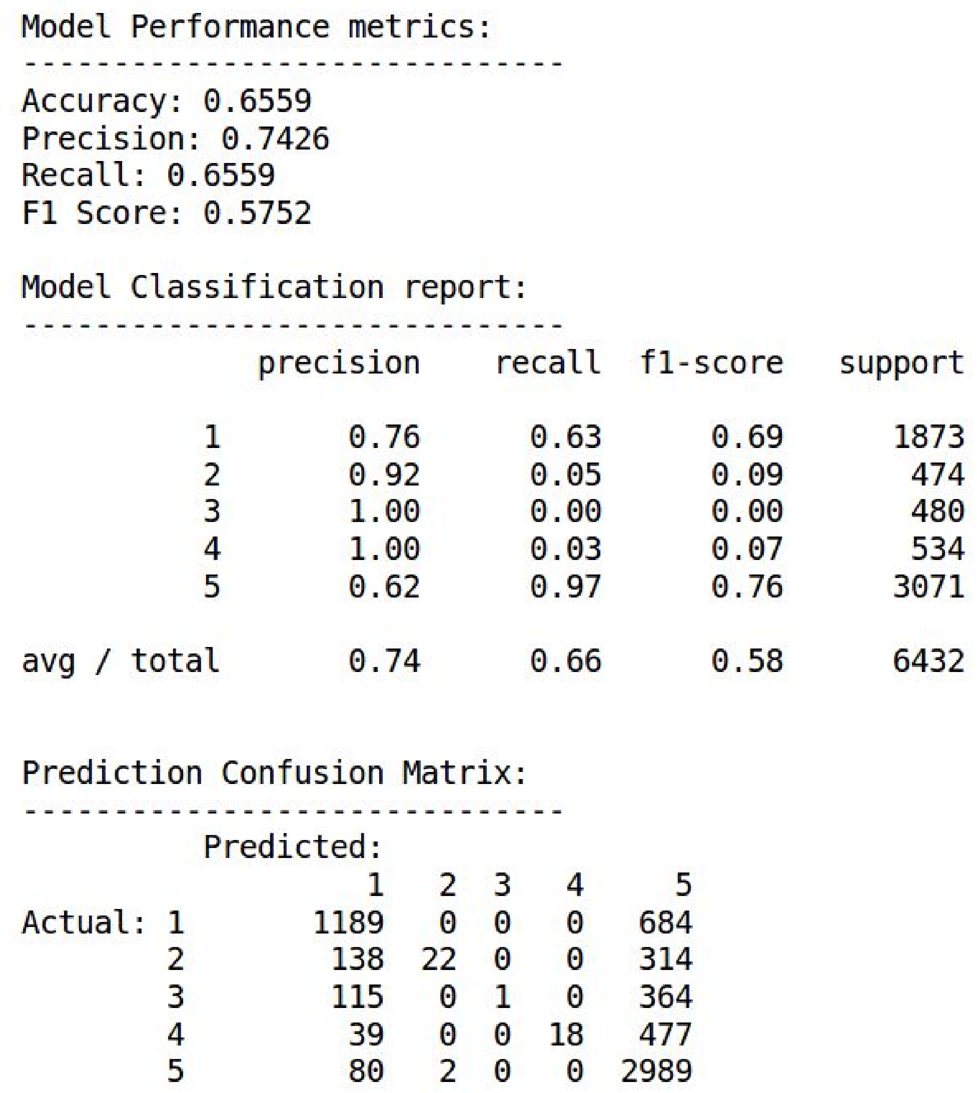
4 Modeling

We use supervised learning algorithms to build predictive models, and find out the best model based on their performance, specifically regarding to each Accuracy and Precision on testing sets. As we mentioned earlier, we will have the cleaned and vectorized “reviews” as our feature set, and the “ score” and “Helpful” as our predictors. We perform Logistics regression, Naives Bayes, LinearSVC, Random Forests,Gradient Boosting and SGDClassifier as our models, and compare accuracy for each to figure out the best model, and parameter.

4.1 Ratings vs. Reviews

4.1.1 Naive Bayes

Accuracy on training set: 0.6767744085304899
Accuracy on testing set: 0.6559390547263682



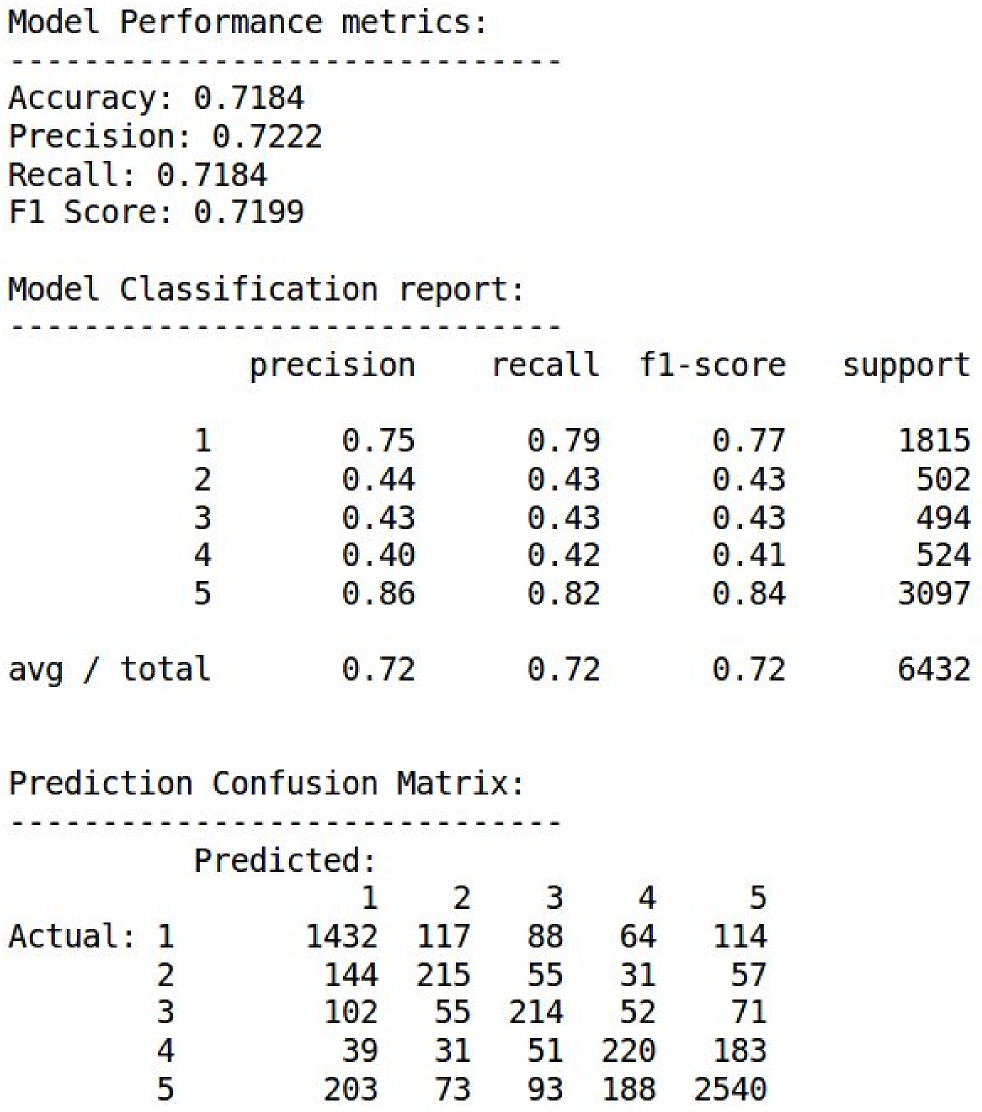
```
Model Performance metrics:
------------------------------
Accuracy: 0.6559
Precision: 0.7426
Recall: 0.6559
F1 Score: 0.5752

Model Classification report:
------------------------------
             precision    recall  f1-score   support

          1       0.76      0.63      0.69      1873
          2       0.92      0.05      0.09       474
          3       1.00      0.00      0.00       480
          4       1.00      0.03      0.07       534
          5       0.62      0.97      0.76      3071

avg / total       0.74      0.66      0.58      6432


Prediction Confusion Matrix:
------------------------------
          Predicted:
                   1   2  3   4     5
Actual: 1       1189   0  0   0   684
        2        138  22  0   0   314
        3        115   0  1   0   364
        4         39   0  0  18   477
        5         80   2  0   0  2989
```

We first start off with Naive Bayes, because it is a popular method for text categorization. This model is based on the assumption that features are independent with each other.

By referring to the data above, we can tell that the model does not perform very well, this might be caused by the oversimplified assumption.

4.1.2 Logistic Regression

Accuracy on training set: 0.87350883038987
Accuracy on testing set: 0.7184390547263682



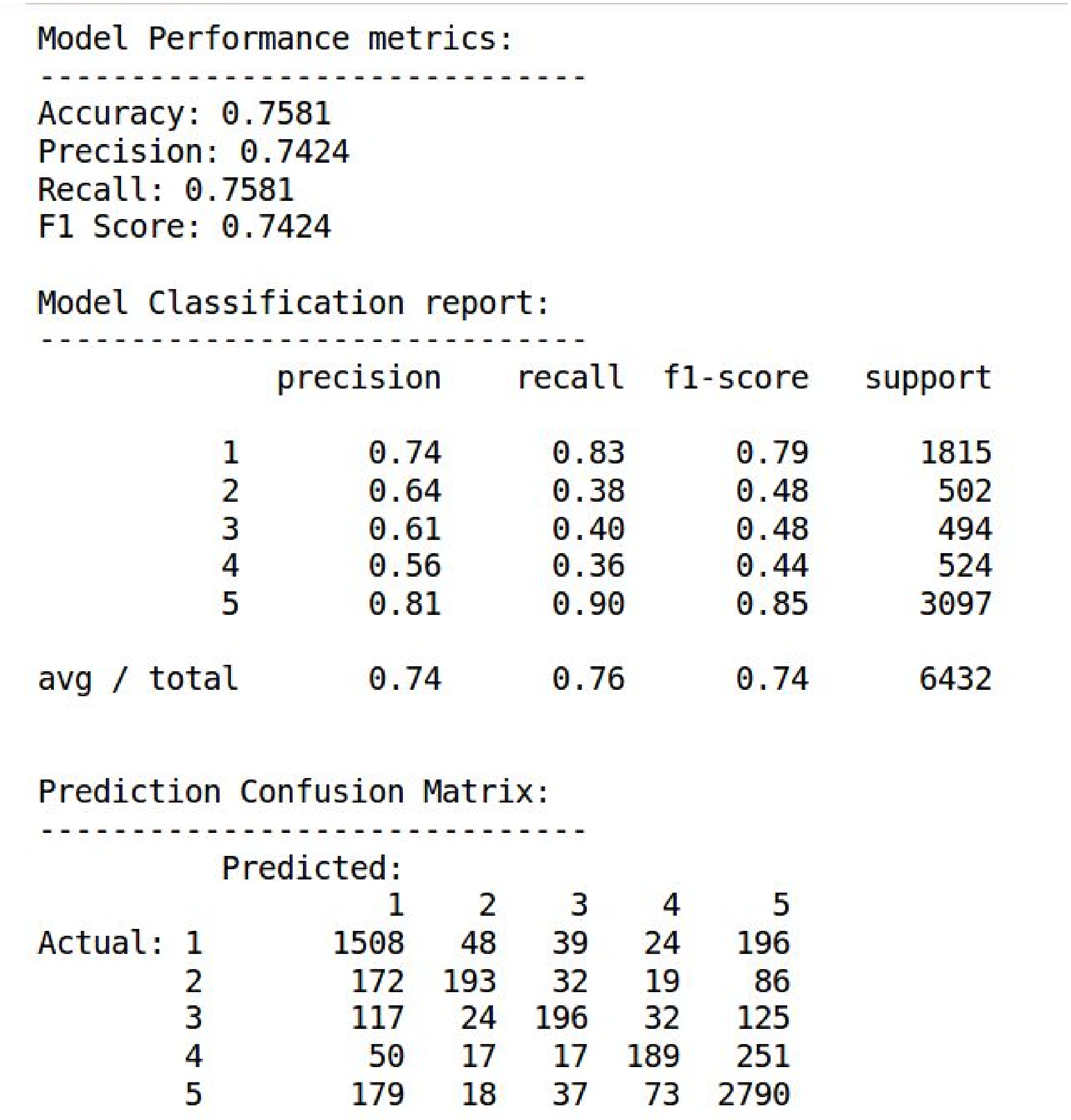
```
Model Performance metrics:
------------------------------
Accuracy: 0.7184
Precision: 0.7222
Recall: 0.7184
F1 Score: 0.7199

Model Classification report:
------------------------------
             precision    recall  f1-score   support

          1       0.75      0.79      0.77      1815
          2       0.44      0.43      0.43       502
          3       0.43      0.43      0.43       494
          4       0.40      0.42      0.41       524
          5       0.86      0.82      0.84      3097

avg / total       0.72      0.72      0.72      6432


Prediction Confusion Matrix:
------------------------------
          Predicted:
                   1    2    3    4     5
Actual: 1       1432  117   88   64   114
        2        144  215   55   31    57
        3        102   55  214   52    71
        4         39   31   51  220   183
        5        203   73   93  188  2540
```

Given the rejected oversimplified assumption, we first assume that the feature sets and the target variables are roughly linear. Accordingly, we perform the Logistic Regression because it can help us turn most non-linear features into linear pretty easily. At the same time, we set the penalty level to be “l2”, which helps the model to ease the overfitting issue.

By referring to the data above, we can tell that the model performs pretty well, especially for score 1 and score 5. Comparatively, score 2,3, and 4 do not perform as well as the other ones; this might be caused by the fact we do not have as many reviews with score 2 to 4 as we have for score 1 and 5 from our dataset.

4.1.3 LinearSVC

Accuracy on training set: 0.9538820393202266
Accuracy on testing set: 0.7580845771144279



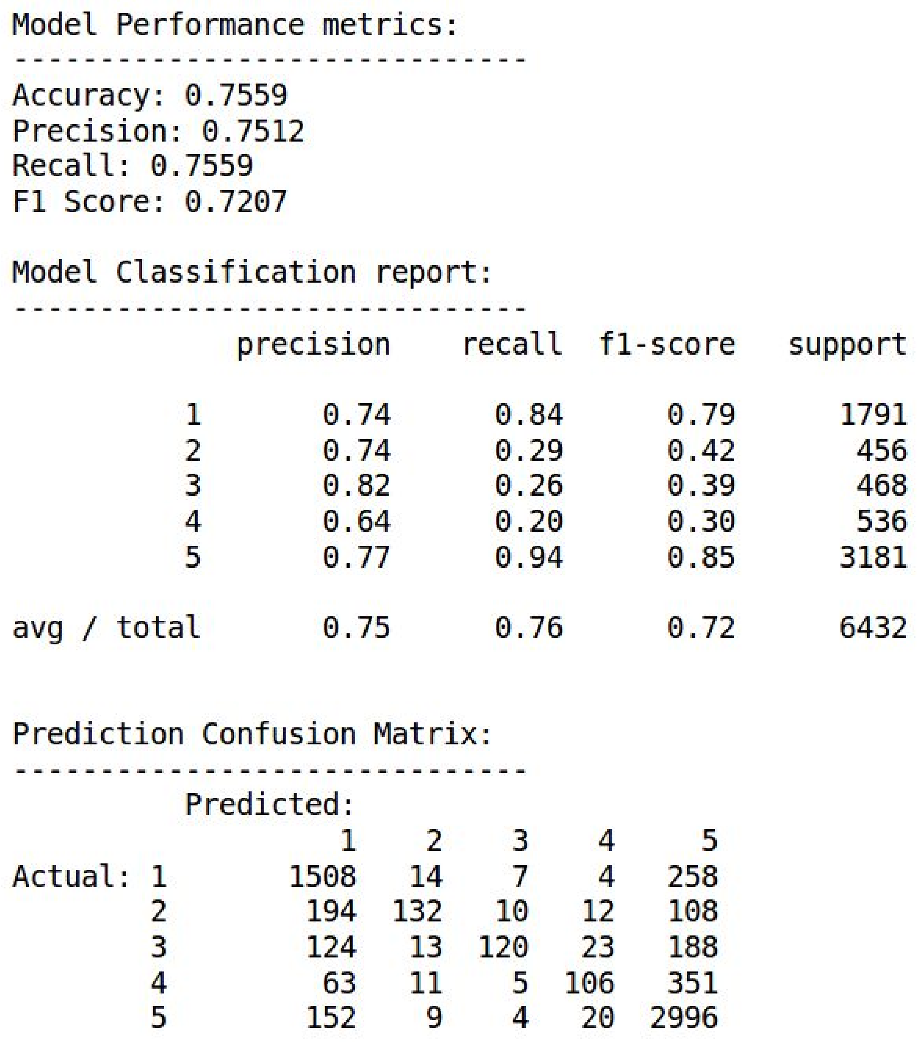
```
Model Performance metrics:
------------------------------
Accuracy: 0.7581
Precision: 0.7424
Recall: 0.7581
F1 Score: 0.7424

Model Classification report:
------------------------------
             precision    recall  f1-score   support

          1       0.74      0.83      0.79      1815
          2       0.64      0.38      0.48       502
          3       0.61      0.40      0.48       494
          4       0.56      0.36      0.44       524
          5       0.81      0.90      0.85      3097

avg / total       0.74      0.76      0.74      6432


Prediction Confusion Matrix:
------------------------------
          Predicted:
                  1    2    3    4     5
Actual: 1      1508   48   39   24   196
        2       172  193   32   19    86
        3       117   24  196   32   125
        4        50   17   17  189   251
        5       179   18   37   73  2790
```

Given the fact that this project is to predict on text classification, the feature sets and the target variables may not be linearly related. Also, due to the large amount of unique words in the dataset, 44,755 features in specific, we choose to perform SVC models, one of models in SVM, with a non-linear kernel, RBF.

By referring to the data above, we can tell that the LinearSVC model performs pretty well, and actually a little bit better than logistic regression.

4.1.3 SGDClassifier

Accuracy on training set: 0.8421859380206598
Accuracy on testing set: 0.755907960199005



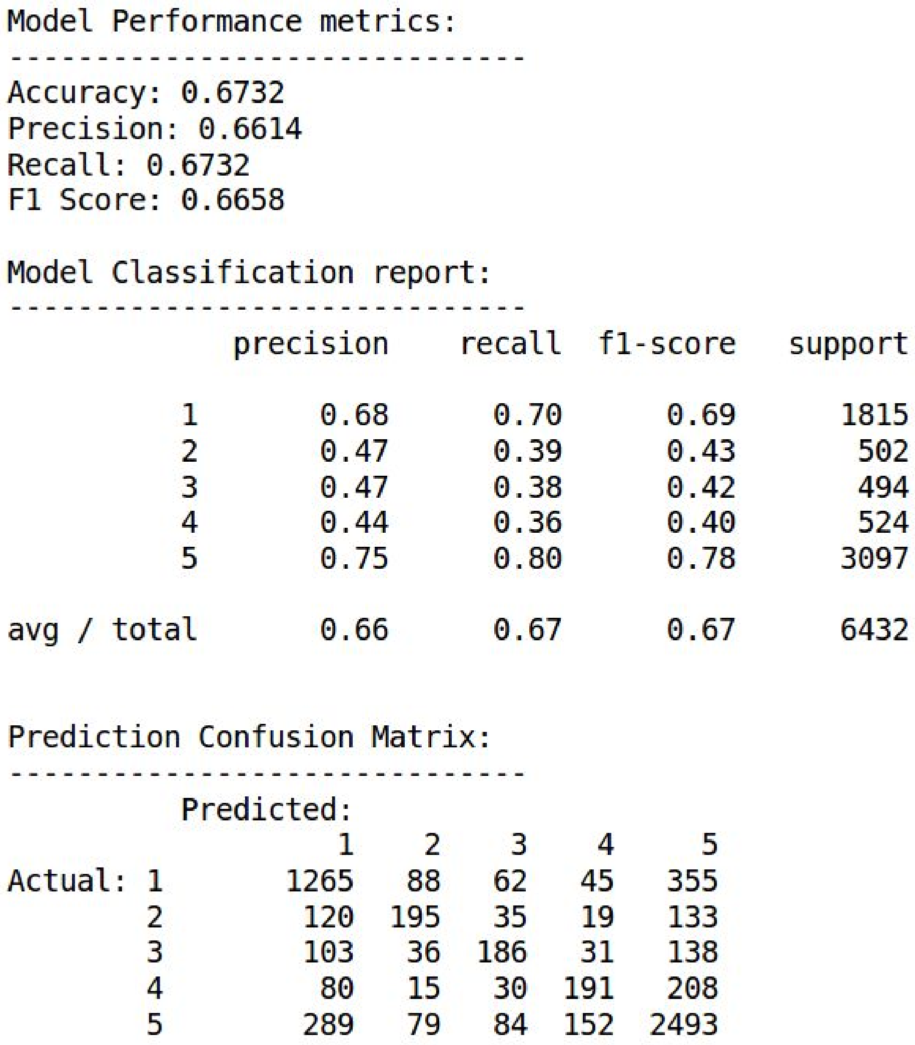
```
Model Performance metrics:
------------------------------
Accuracy: 0.7559
Precision: 0.7512
Recall: 0.7559
F1 Score: 0.7207

Model Classification report:
------------------------------
             precision    recall  f1-score   support

          1       0.74      0.84      0.79      1791
          2       0.74      0.29      0.42       456
          3       0.82      0.26      0.39       468
          4       0.64      0.20      0.30       536
          5       0.77      0.94      0.85      3181

avg / total       0.75      0.76      0.72      6432


Prediction Confusion Matrix:
------------------------------
          Predicted:
                  1    2    3    4     5
Actual: 1      1508   14    7    4   258
        2       194  132   10   12   108
        3       124   13  120   23   188
        4        63   11    5  106   351
        5       152    9    4   20  2996
```

While Logistics Regression and Support Vector Machines have a disadvantage of discriminating linear classifiers under convex loss functions, we try out SGDClassifier model to compensate this issue.

Based on the Accuracy and Precision on the testing set, this model performs just as well as LinearSVC.

4.1.4 Decision Tree

Accuracy on training set: 1.0
Accuracy on testing set: 0.6731965174129353



```
Model Performance metrics:
------------------------------
Accuracy: 0.6732
Precision: 0.6614
Recall: 0.6732
F1 Score: 0.6658

Model Classification report:
------------------------------
             precision    recall  f1-score   support

          1       0.68      0.70      0.69      1815
          2       0.47      0.39      0.43       502
          3       0.47      0.38      0.42       494
          4       0.44      0.36      0.40       524
          5       0.75      0.80      0.78      3097

avg / total       0.66      0.67      0.67      6432


Prediction Confusion Matrix:
------------------------------
          Predicted:
                 1    2    3    4    5
Actual: 1     1265   88   62   45  355
        2      120  195   35   19  133
        3      103   36  186   31  138
        4       80   15   30  191  208
        5      289   79   84  152 2493
```

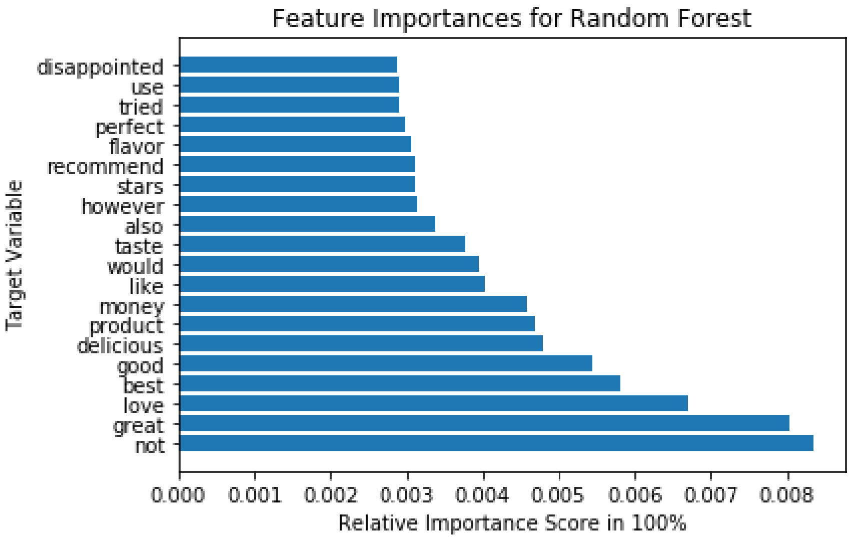
Besides examining the dataset as a whole, we also perform decision tree models to make the most optimal decision at each step. This model helps us to examine how each feature in specific has an effect on the target variables.

Given the fact that there are 44,755 features in the feature set, and the fact that written reviews could contain quite a lot of “noises” and “outliers”, the model tend to be overfitting. Hence, we set the max_df to be “None”, which helps to limit the risk of overfitting.

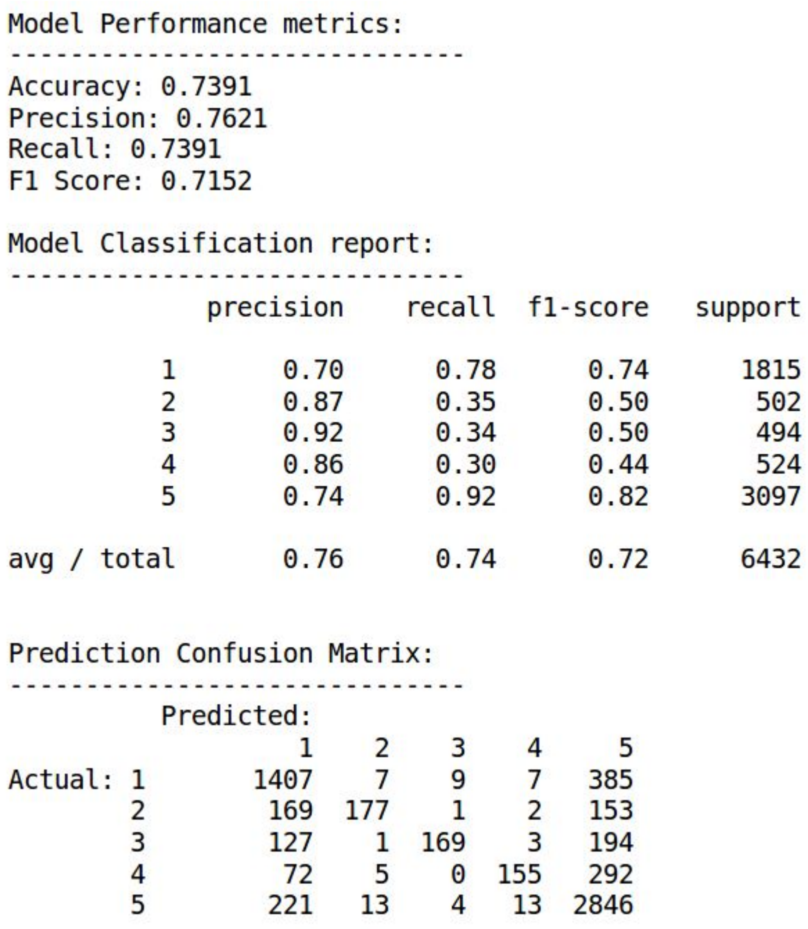
The criteria "None" means nodes are expanded until all leaves are pure or until all leaves contain less than min_samples_split samples.

Based on the "Accuracy on testing set" above, we can tell that the Decision Tree model does not perform as well as the other ones. This might be due to the nature of decision tree model, which it tends to make the most optimal decision at each step, instead of taking into account the global optimum. The other reason may also be that decision tree trains data based on the results of the last training data.

4.1.5 Random Forests



Accuracy on training set: 0.9910696434521826
Accuracy on testing set: 0.7391169154228856



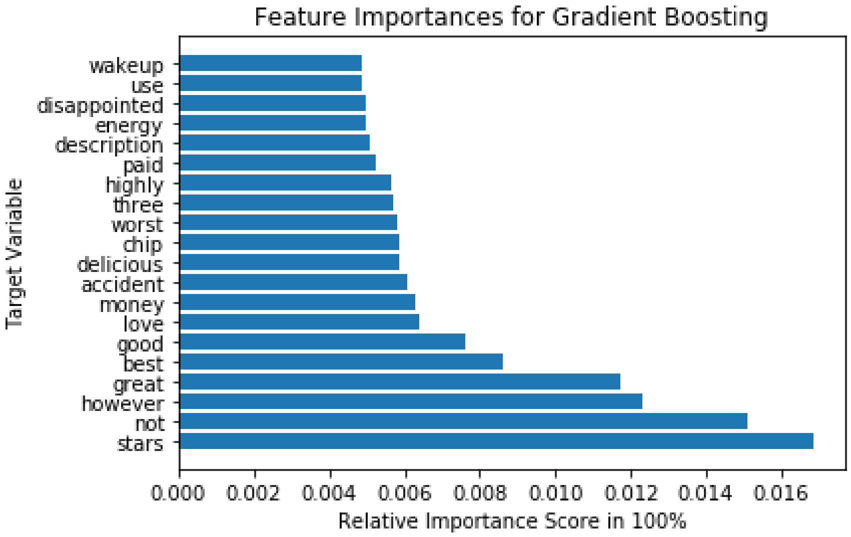
```
Model Performance metrics:
------------------------------
Accuracy: 0.7391
Precision: 0.7621
Recall: 0.7391
F1 Score: 0.7152

Model Classification report:
------------------------------
             precision    recall  f1-score   support

          1       0.70      0.78      0.74      1815
          2       0.87      0.35      0.50       502
          3       0.92      0.34      0.50       494
          4       0.86      0.30      0.44       524
          5       0.74      0.92      0.82      3097

avg / total       0.76      0.74      0.72      6432


Prediction Confusion Matrix:
------------------------------
          Predicted:
                   1    2    3    4     5
Actual: 1       1407    7    9    7   385
        2        169  177    1    2   153
        3        127    1  169    3   194
        4         72    5    0  155   292
        5        221   13    4   13  2846
```

Instead of running all features in a dataset, randomly selecting a subset of features for each step and running each one independently might ease this overfitting issue. Accordingly, we try out the Random Forests model to fix the problem. As a result, the output shows that Random Forests performs stronger than a single decision tree.

Based on the top 20 feature importances listed above, we get a sense of which word variables have the most effect in our models. As we can tell, the model performs pretty well because all the words listed above are the common words we see and use in reviews.

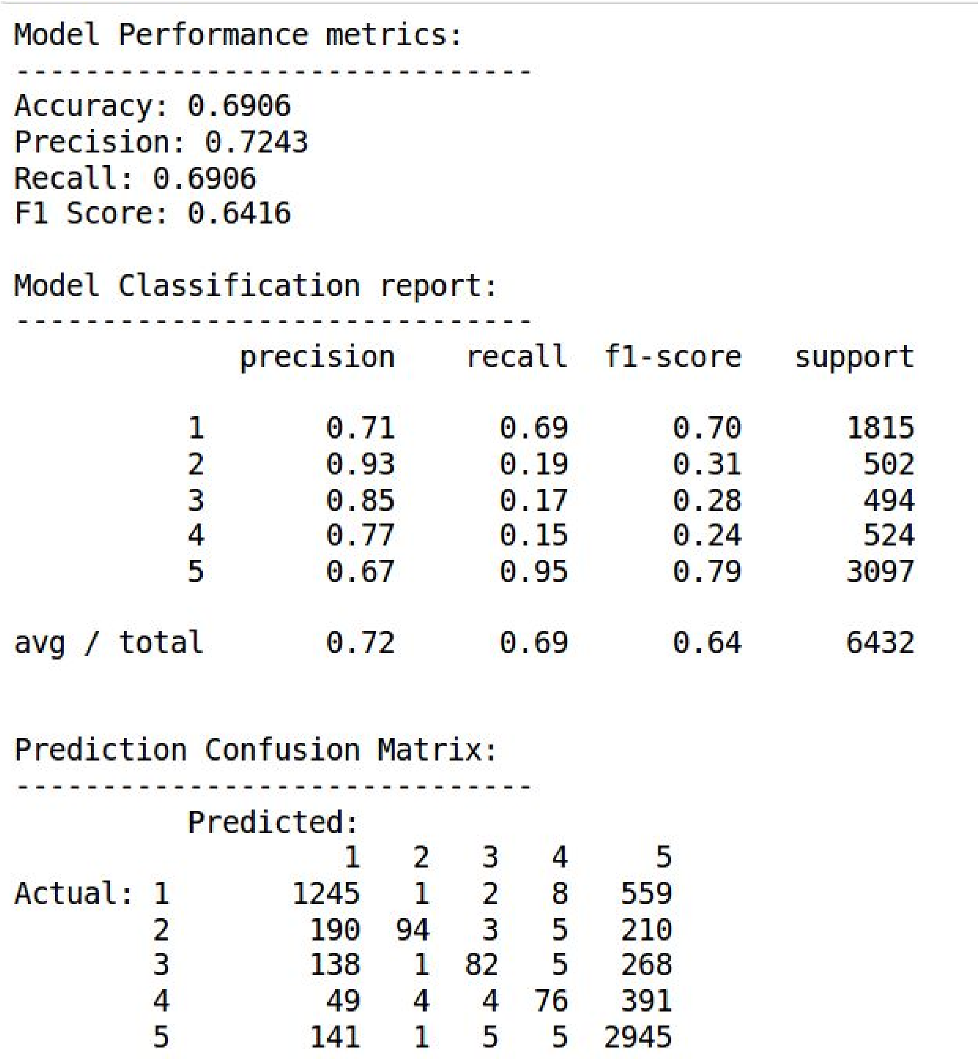
Also, by referring to the accuracy, we can tell the this model performs pretty well, and is actually the third best model of all besides LinearSVC and SGDClassifier. Additionally, score 2,3, and 4 perform well in this model.

4.1.6 Gradient Boosting



Accuracy on training set: 0.738487170943019

Accuracy on testing set: 0.6906094527363185



```
Model Performance metrics:
------------------------------
Accuracy: 0.6906
Precision: 0.7243
Recall: 0.6906
F1 Score: 0.6416

Model Classification report:
------------------------------
             precision    recall  f1-score   support

          1       0.71      0.69      0.70      1815
          2       0.93      0.19      0.31       502
          3       0.85      0.17      0.28       494
          4       0.77      0.15      0.24       524
          5       0.67      0.95      0.79      3097

avg / total       0.72      0.69      0.64      6432


Prediction Confusion Matrix:
------------------------------
          Predicted:
                   1   2   3   4     5
Actual: 1       1245   1   2   8   559
        2        190  94   3   5   210
        3        138   1  82   5   268
        4         49   4   4  76   391
        5        141   1   5   5  2945
```

Even though Random Forests performs well on this dataset, it does have its limitations like the features were chosen randomly with replacement. Hence, we try out another popular decision tree model, Gradient Boosting, to fix this issue by training on data instances that had been modeled poorly in the overall system before.

According to the output, Gradient Boosting does not perform as well as Random Forests. This might be due to the fact that Gradient Boosting' training is based on last training result whereas Random Forests is trained independently from the rest.

4.1.7 Hyperparameter tuning for the best parameter

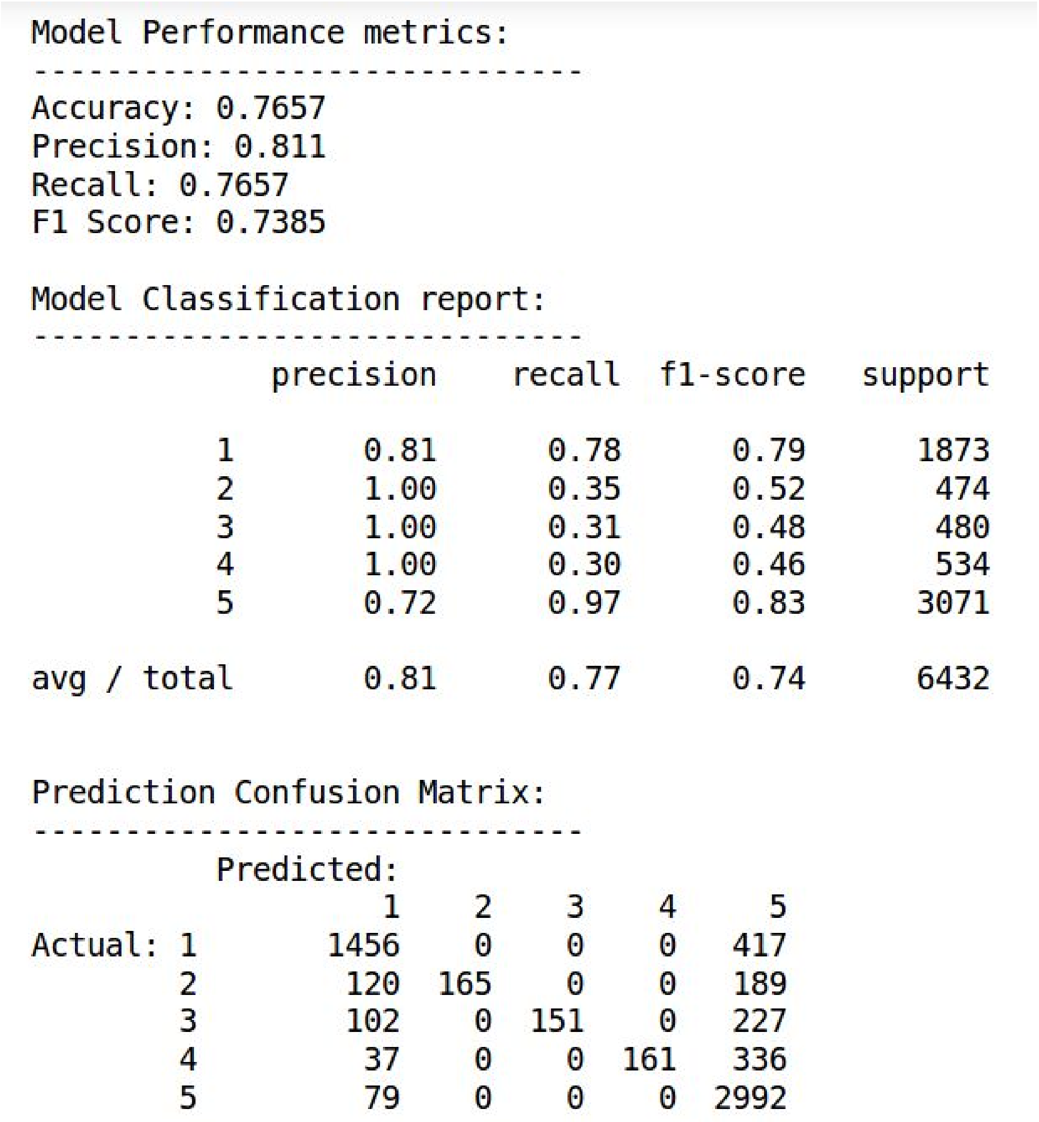
Based on all the models listed above, it turns out that the Random Forest, SGDClassifier, and LinearSVC give us the best results. In order to re-confirm which model is the best for

our dataset, we use grid search to figure out the best hyperparameter by performing each model one more time.

By referring to the output for each model, it turns out that the Random Forest gives us the best parameter.

R-Squared::0.7559480173275575
Best Hyperparameters::
{'n_estimators': 200}

Accuracy on training set: 1.0
Accuracy on testing set: 0.7657027363184079



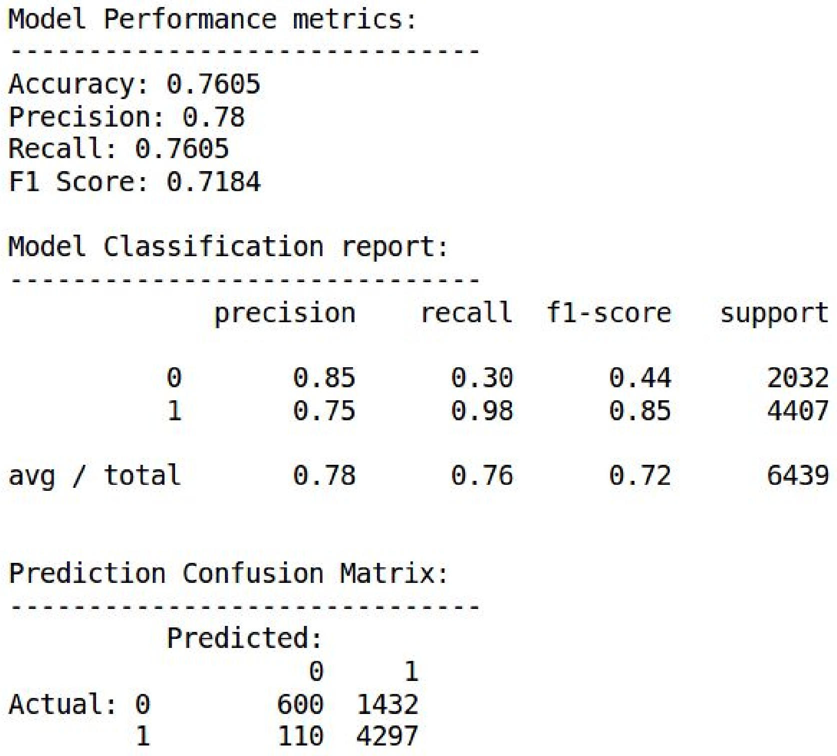
```
Model Performance metrics:
------------------------------
Accuracy: 0.7657
Precision: 0.811
Recall: 0.7657
F1 Score: 0.7385

Model Classification report:
------------------------------
             precision    recall  f1-score   support

          1       0.81      0.78      0.79      1873
          2       1.00      0.35      0.52       474
          3       1.00      0.31      0.48       480
          4       1.00      0.30      0.46       534
          5       0.72      0.97      0.83      3071

avg / total       0.81      0.77      0.74      6432


Prediction Confusion Matrix:
------------------------------
          Predicted:
                 1    2    3    4     5
Actual: 1     1456    0    0    0   417
        2      120  165    0    0   189
        3      102    0  151    0   227
        4       37    0    0  161   336
        5       79    0    0    0  2992
```

4.2 Helpful vs. Reviews

For this section, we perform all the model as we do for “ Ratings vs. Reviews” section.

4.2.1 Naive Bayes

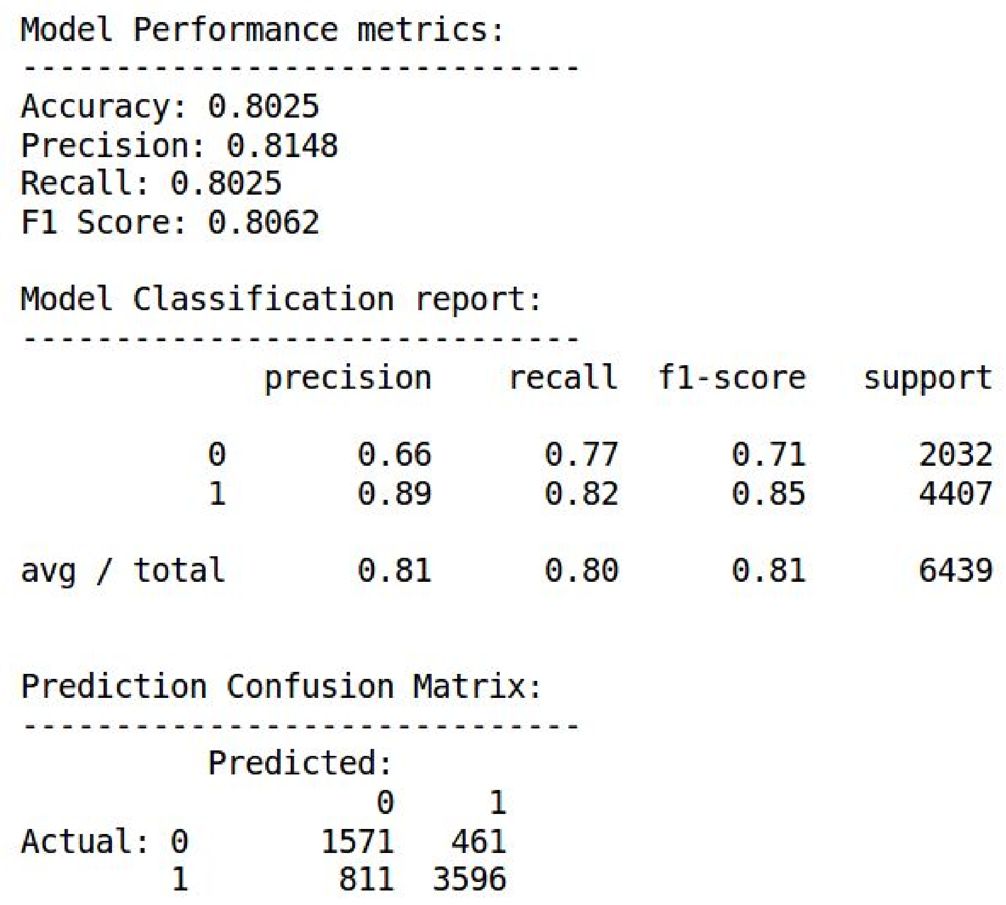
Accuracy on training set: 0.7872736954206603
Accuracy on testing set: 0.7605218201584097



```
Model Performance metrics:
------------------------------
Accuracy: 0.7605
Precision: 0.78
Recall: 0.7605
F1 Score: 0.7184

Model Classification report:
------------------------------
             precision    recall  f1-score   support

          0       0.85      0.30      0.44      2032
          1       0.75      0.98      0.85      4407

avg / total       0.78      0.76      0.72      6439


Prediction Confusion Matrix:
------------------------------
          Predicted:
                   0     1
Actual: 0        600  1432
        1        110  4297
```

4.2.1 Logistic Regression

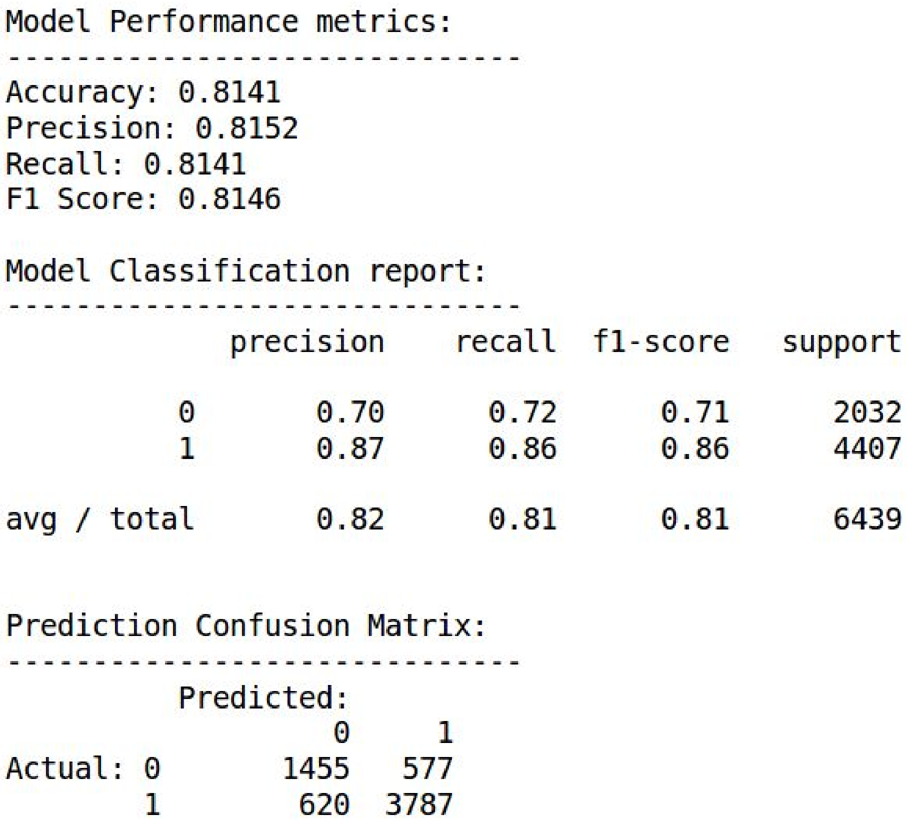
Accuracy on training set: 0.86395101171459
Accuracy on testing set: 0.8024537971734741



```
Model Performance metrics:
------------------------------
Accuracy: 0.8025
Precision: 0.8148
Recall: 0.8025
F1 Score: 0.8062

Model Classification report:
------------------------------
             precision    recall  f1-score   support

          0       0.66      0.77      0.71      2032
          1       0.89      0.82      0.85      4407

avg / total       0.81      0.80      0.81      6439


Prediction Confusion Matrix:
------------------------------
          Predicted:
                   0     1
Actual: 0       1571   461
        1        811  3596
```

4.2.2 LinearSVC

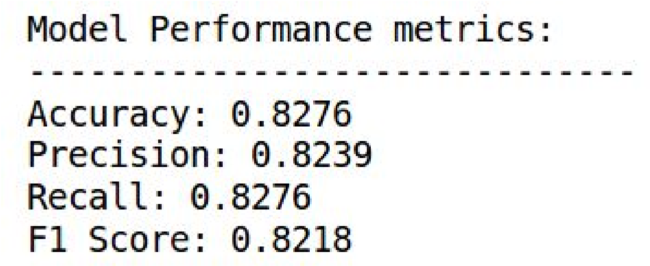
Accuracy on training set: 0.9959398296059638
Accuracy on testing set: 0.8141015685665476



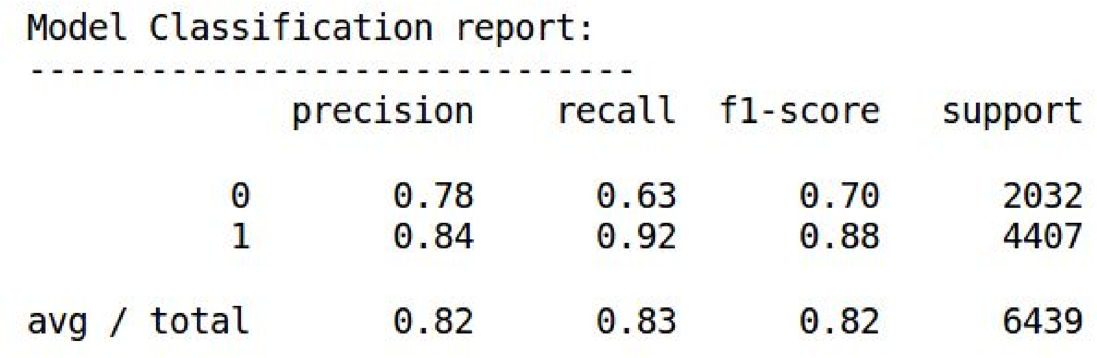
```
Model Performance metrics:
------------------------------
Accuracy: 0.8141
Precision: 0.8152
Recall: 0.8141
F1 Score: 0.8146

Model Classification report:
------------------------------
             precision    recall  f1-score   support

          0       0.70      0.72      0.71      2032
          1       0.87      0.86      0.86      4407

avg / total       0.82      0.81      0.81      6439


Prediction Confusion Matrix:
------------------------------
          Predicted:
                   0     1
Actual: 0       1455   577
        1        620  3787
```

4.2.3 SGDClassifier

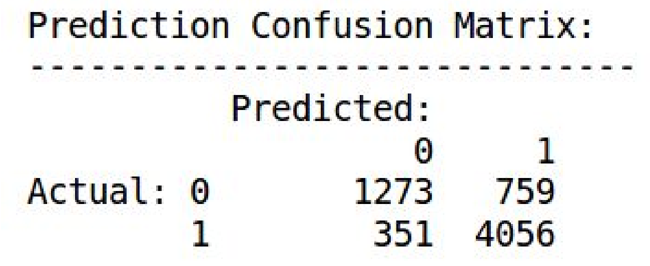
Accuracy on training set: 0.8893104366347178
Accuracy on testing set: 0.8276129833825128



```
Model Performance metrics:
------------------------------
Accuracy: 0.8276
Precision: 0.8239
Recall: 0.8276
F1 Score: 0.8218
```



```
Model Classification report:
------------------------------
             precision    recall  f1-score   support

          0       0.78      0.63      0.70      2032
          1       0.84      0.92      0.88      4407

avg / total       0.82      0.83      0.82      6439
```

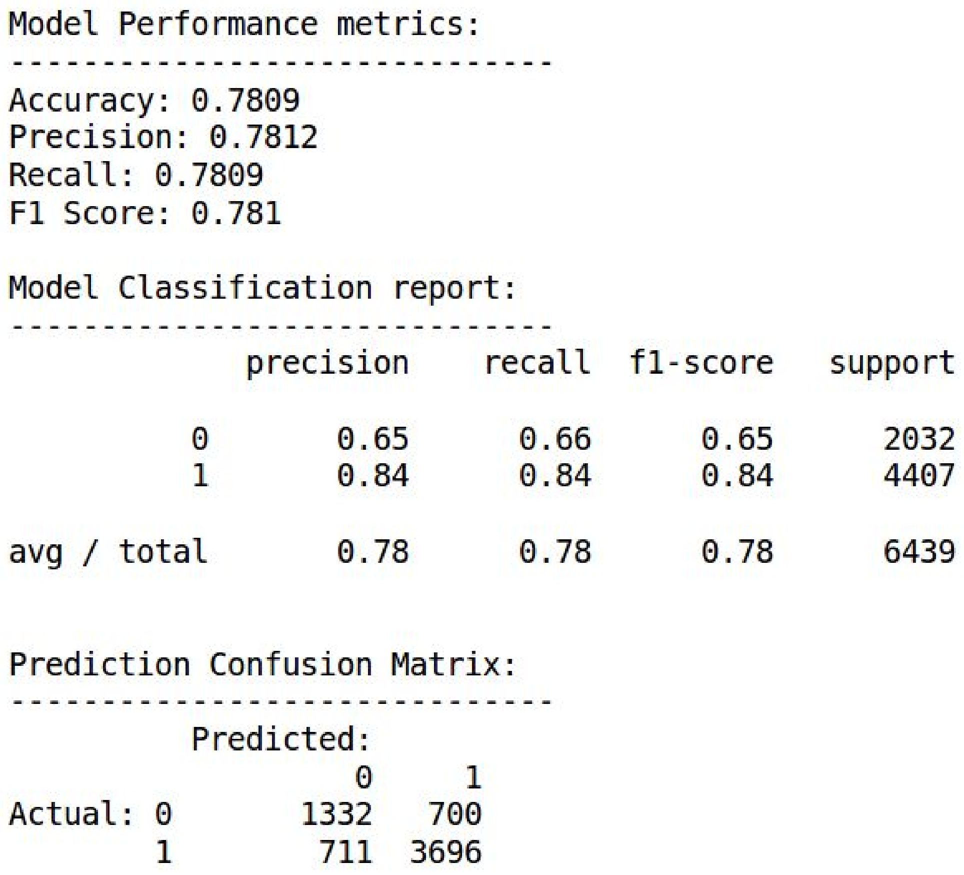


```
Prediction Confusion Matrix:
------------------------------
          Predicted:
                   0     1
Actual: 0       1273   759
        1        351  4056
```

4.2.4 Decision Tree

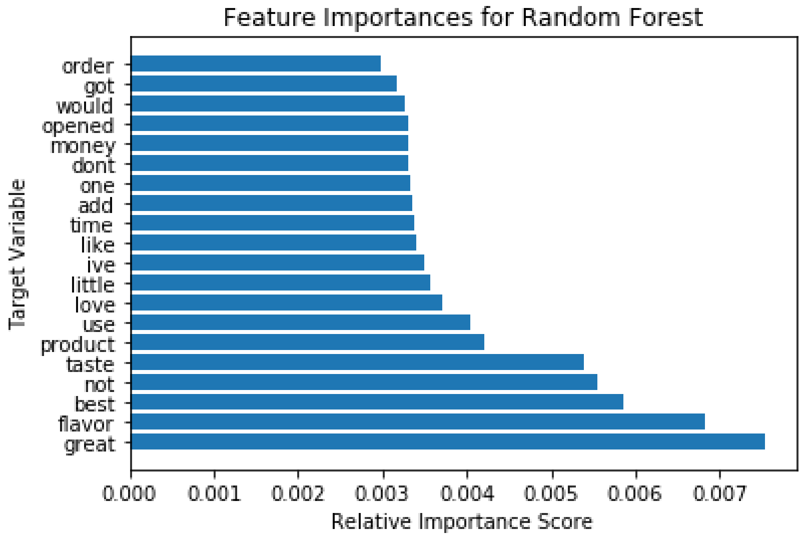
Tree One, Accuracy on training set: 0.9987353567625133
Tree One, Accuracy on testing set: 0.7808665941916446

Tree Two, Accuracy on training set: 0.6845047923322684
Tree Two, on testing set: 0.6903245845628203



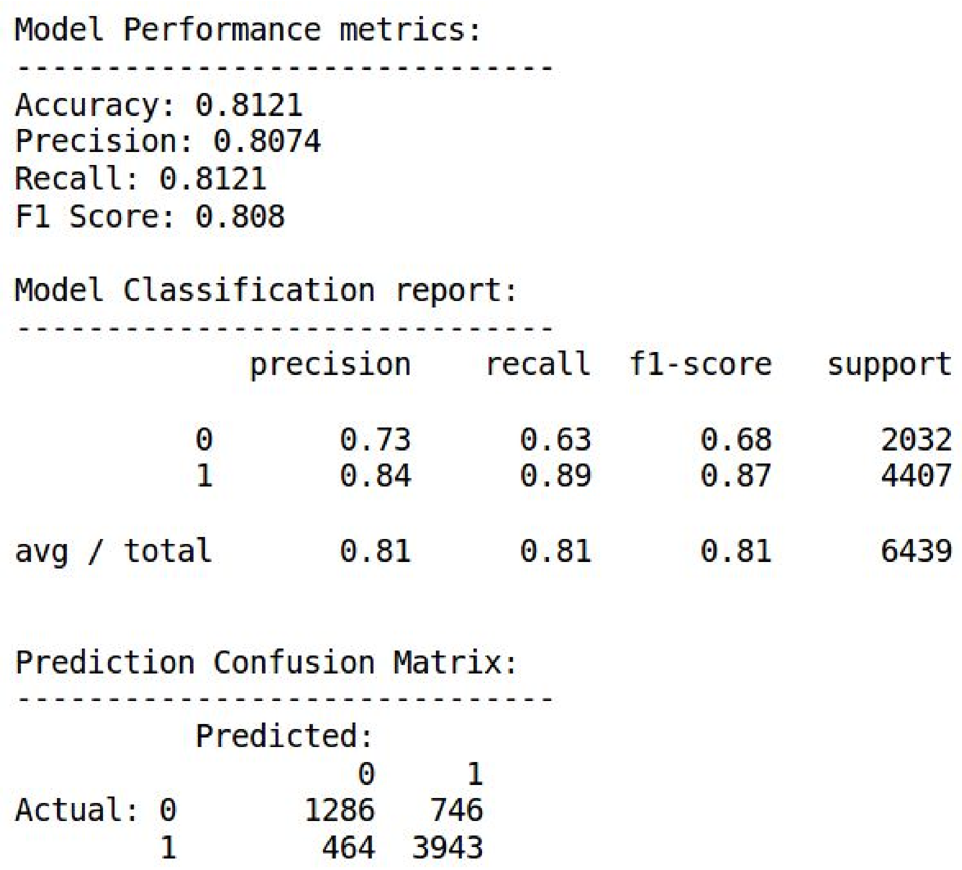
```
Model Performance metrics:
------------------------------
Accuracy: 0.7809
Precision: 0.7812
Recall: 0.7809
F1 Score: 0.781

Model Classification report:
------------------------------
             precision    recall  f1-score   support

          0       0.65      0.66      0.65      2032
          1       0.84      0.84      0.84      4407

avg / total       0.78      0.78      0.78      6439


Prediction Confusion Matrix:
------------------------------
          Predicted:
                   0     1
Actual: 0       1332   700
        1        711  3696
```

4.2.5 Random Forests



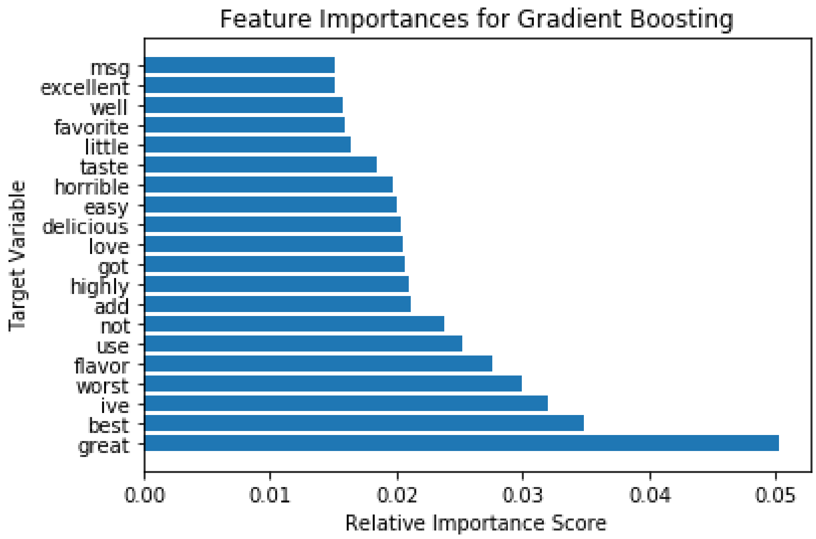
Accuracy on training set: 0.9932774227902024

Accuracy on testing set: 0.8120826215250815



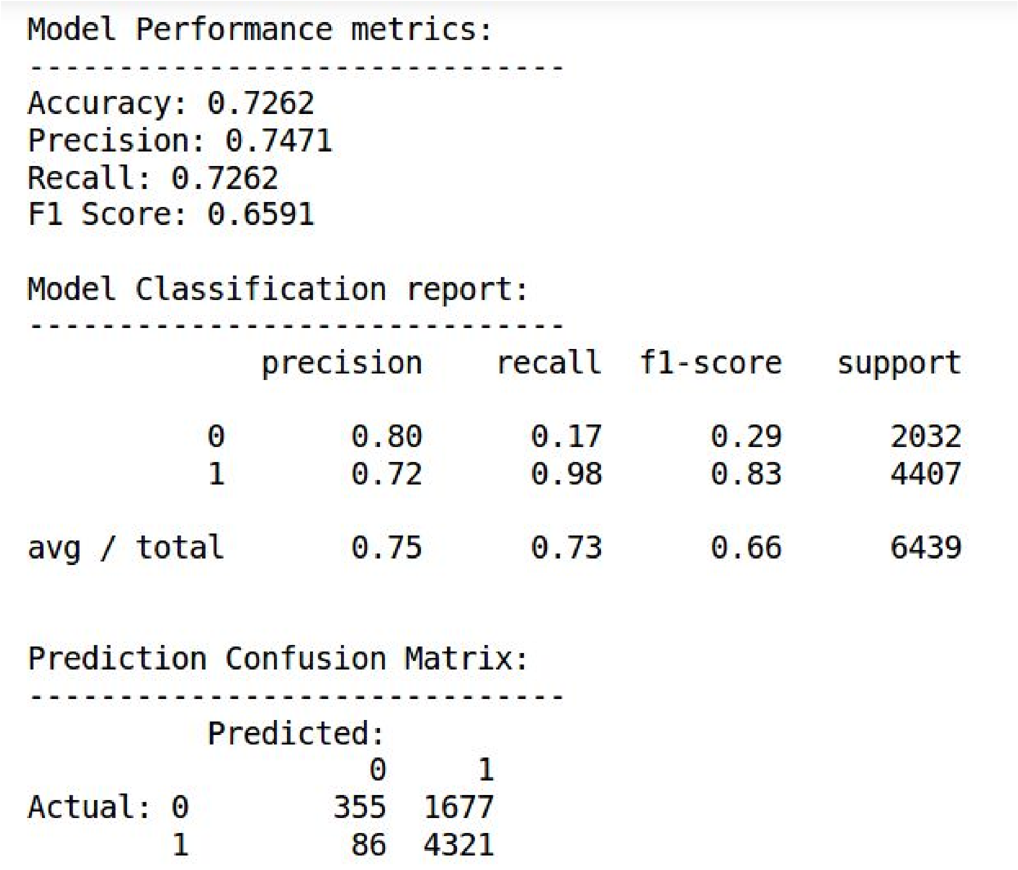
```
Model Performance metrics:
------------------------------
Accuracy: 0.8121
Precision: 0.8074
Recall: 0.8121
F1 Score: 0.808

Model Classification report:
------------------------------
             precision    recall  f1-score   support

          0       0.73      0.63      0.68      2032
          1       0.84      0.89      0.87      4407

avg / total       0.81      0.81      0.81      6439


Prediction Confusion Matrix:
------------------------------
          Predicted:
                 0     1
Actual: 0     1286   746
        1      464  3943
```

4.2.6 Gradient Boosting



Accuracy on training set: 0.7279685835995741

Accuracy on testing set: 0.7261997204534866



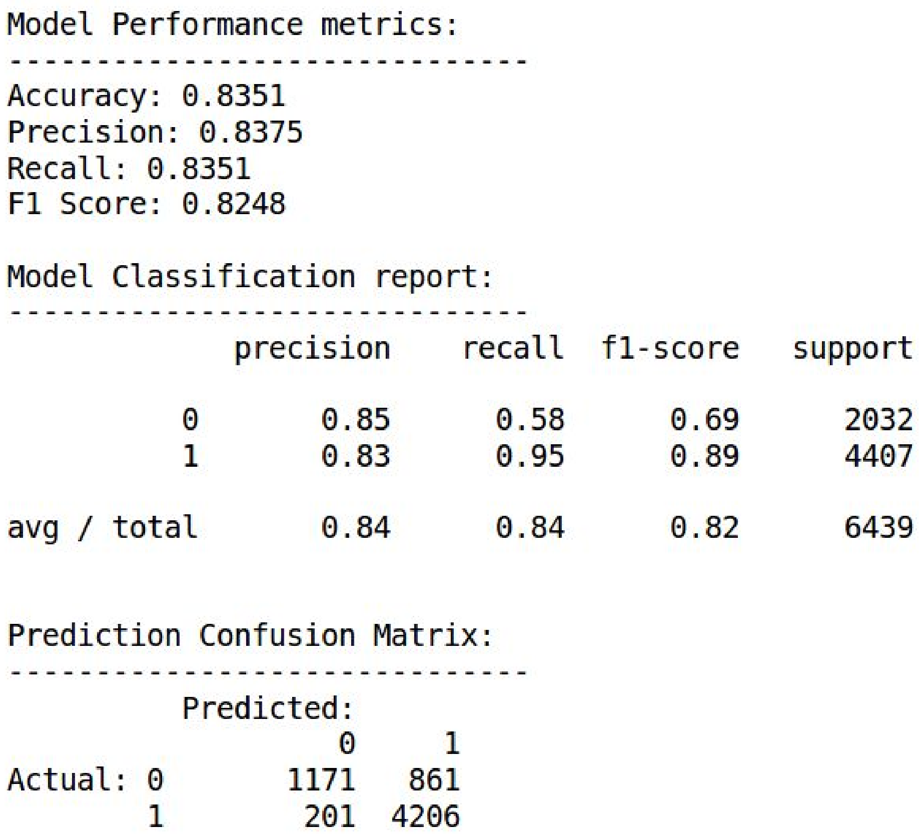
```
Model Performance metrics:
------------------------------
Accuracy: 0.7262
Precision: 0.7471
Recall: 0.7262
F1 Score: 0.6591

Model Classification report:
------------------------------
             precision    recall  f1-score   support

          0       0.80      0.17      0.29      2032
          1       0.72      0.98      0.83      4407

avg / total       0.75      0.73      0.66      6439


Prediction Confusion Matrix:
------------------------------
          Predicted:
                  0     1
Actual: 0       355  1677
        1        86  4321
```

4.2.7 Hyperparameter tuning for the best model

Based on all the models listed above, it turns out that the Random Forest, SGDClassifier, Logistic Regression and LinearSVC give us the best results. In order to re-confirm which model is the best for our dataset, we use grid search to figure out the best hyperparameter by performing each model one more time.

By referring to the output for each model, it turns out that the Random Forest gives us the best parameter.

Accuracy on training set: 0.9987353567625133
Accuracy on testing set: 0.8350675570740799



```
Model Performance metrics:
------------------------------
Accuracy: 0.8351
Precision: 0.8375
Recall: 0.8351
F1 Score: 0.8248

Model Classification report:
------------------------------
             precision    recall  f1-score   support

          0       0.85      0.58      0.69      2032
          1       0.83      0.95      0.89      4407

avg / total       0.84      0.84      0.82      6439


Prediction Confusion Matrix:
------------------------------
          Predicted:
                   0     1
Actual: 0       1171   861
        1        201  4206
```

5 Limitations

We lack the knowledge of the major market for each category of fine food. In other words, the reviewers are not narrowed down in ages, jobs, or other backgrounds. Hence, the degree of helpfulness could be somewhat biased.

While positive reviews are helping Amazon and sellers generating more profits, fake reviews could be a concern that hurt their reputations. For example, a Washington Post examination found that for some popular product categories, such as Bluetooth headphones and speakers, the vast majority of reviews appear to violate Amazon’s prohibition on paid reviews. Such reviews have certain characteristics, such as repetitive wording that people probably cut and paste in. Regarding to this issue, we lack the knowledge of whether the review is from a verified user.

6 Conclusions

For both sections, the RandomForestRegressor gives us the best R^2 value among all the models we have performed. Also, it give us the best hyperparameter and model.