Enron Submission Free Response Questions (1-6)

Eric Fraser

1. The goal of this project is to use machine learning algorithms to analyze the Enron Email Corpus. Machine learning is useful for this project because the mission is to use the data in the Corpus to see if we can determine whether a person in the data set is a person of interest based on the other features of that person. This type of discovery is explored, and this type of data is analyzed much more easily using computerized algorithms than trying to synthesize the data utilizing a human brain. The data contains thousands of emails as data points and offers the viewer a unique look into the biggest scandal involving one of the most notorious organizations of the 21st century. There were a few outliers, including names of potential POIs that did not really fit with the others. These were cleaned out of the list.
2. I initially sorted my features using K-Best to determine which features should be kept and which should be discarded. No scaling was required. I figured that it would be interesting to see the proportion of all e-mails to and from a person went to POIs as opposed to just looking at the raw number sent. As it turned out, the ‘ratio of messages to poi’ feature had the 5th highest K-Value of the features, and the other created feature, ‘ratio of messages from poi’ had a K-Value about 8 times lower than the ‘messages to’ variant. In the final analysis, 10 features were selected ranging in K-Values from 24+ (Exercised Stock Options and Total Stock Value) to around 7 (Loan Advances). No features selected had a K-Value of less than 7.
3. I ended up using the Decision Tree algorithm. I also tried Gaussian Naïve Bayes, but the metrics for Decision Tree (especially at test size 0.25) outperformed the metrics for GNB. The accuracy for my model was .89, the recall was .67, and the precision was .4. GNB offered .33 accuracy, 1.0 recall, and .11 precision. Clearly the GNB algorithm recalled the positive data perfectly, but the accuracy and precision were lacking to the point where Decision Tree was the better option on the whole.
4. Tuning the parameters of an algorithm means changing the input variables to better classify and match the data. For example, you can adjust the subset of the data set that is used for training and testing the algorithm at the front end. Because I used decision tree there were several parameters that could be played around with. For example, the ‘splitter’ parameter is how the algorithm will choose how to split each node – the options being randomly or by what the algorithm determines is the ‘best’ split. Additionally, you may determine the parameter min\_samples\_leaf, which identifies the minimum number of samples required to be at a leaf node, effectively smoothing the model out.
5. Validation is how you determine that your algorithm is doing what you want it to do. In general, we want our algorithms to work to classify new data properly. One common error in the validation step is called ‘overfitting’. Overfitting occurs when your algorithm is tuned so perfectly to your training data set, the lack of ambiguity makes the algorithm ineffective in dealing with new data. I validated my data set by utilizing the cross-validation function and adjusting the test size sample and the training size sample.
6. The evaluation metrics I used were Precision and Recall. Precision is the number of True Positives divided by the number of all items labeled positive. Recall on the other hand gives you the number of truly positive items that were correctly classified as positive. Precision and recall are extremely common metrics and offer a lot of insight into how effective an algorithm can be. In this project, the Decision Tree algorithm achieved a measurement of .67 for recall and .4 for precision. This means that of all the people the algorithm said were Persons of Interest, 40% were in fact labeled as POIs. Also, the Decision tree algorithm correctly identified, or recalled, 67% of the POIs who were in fact POIs.

Resources:

<https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html>

<https://www.w3schools.com/python/ref_func_round.asp>

<https://www.tutorialspoint.com/How-to-wrap-long-lines-in-Python>

<https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html>

<https://scikit-learn.org/stable/modules/generated/sklearn.naive_bayes.GaussianNB.html>

<https://scikit-learn.org/stable/modules/generated/sklearn.metrics.accuracy_score.html>

<https://scikit-learn.org/stable/modules/generated/sklearn.metrics.recall_score.html>

<https://scikit-learn.org/stable/modules/generated/sklearn.metrics.precision_score.html>

<https://scikit-learn.org/stable/modules/generated/sklearn.feature_selection.SelectKBest.html>

<https://towardsdatascience.com/why-and-how-to-do-cross-validation-for-machine-learning-d5bd7e60c189>