Project 5 - Enron Submission Free-Response Questions

1.- Summarize for us the goal of this project and how machine learning is useful in trying to accomplish it.

The goal is to use the Enron dataset to predict if a particular person is a person of interest in the fraud investigation. The dataset consists of the email and financial information of people involved in the investigation. Each data point corresponds to a person. And the features in the dataset corresponds to financial and email communication variables collected for each person. There are 146 data points and each data point has 21 features. Features that have no value are indicated with 'NaN'. Some features have lots of NaNs. In particular, Loan Advances (142), Director Fees (129) and Restricted Stock Deferred (128).

In particular one feature is 'poi' which indicates if the person is known to be a person of interest before hand (18 poi's in the dataset out of 35 known ones). From the initial data exploration an outlier was removed. The source of the data included a 'Total' data point which was including the addition of all other features. This was removed from the data dictionary right after the data load to avoid it influencing the classification.

This is a problem that can be tackled through machine learning because classification algorithms are suited to take those features and predict the class where a person may fall (poi or non-poi) following the patterns in the data.

2.- What features did you end up using in your POI identifier?

Features selection process was based on univariate feature selection. The chosen method was 'selectkbest' as the number of features is reasonably low. For the chosen algorithm the number of features (k parameter) were iteratively increased from 2 (minimum) to 18 (maximum). At each iteration the test script was ran to assess precision, recall and other evaluation metrics with Naive Bayes and Decision Tree algorithms.

For the Gaussian Naive Bayes the chosen parameter for k was 7. This is the point where adding more features made the evaluation metrics start to decrease in performance. For the Decision Tree this point was reached with 3 features.

The following are the scores for the selected features:

exercised_stock_options = 25.0975415287
total_stock_value = 24.4676540475
bonus = 21.0600017075
salary = 18.575703268
deferred_income = 11.5955476597
long_term_incentive = 10.0724545294
restricted_stock = 9.34670079105

No scaling of features was deemed necessary for the selected algorithm.

A new feature called ‘mentioned_by_poi’ was developed. The feature consists in the number of times a particular person's name is mentioned in the corpus of emails sent from email addresses corresponding to known persons of interest. The rationale being that maybe if you are a poi, other pois would mention you in emails more often than other people. Details of how the feature was engineered can be found in my_feature.py of the submission. The score of the feature by using SelectKBest was 0.0367. When introducing it into the classifier along with the top 3 features it actually decreased the performance of the algorithm as follows (see answer about algorithms for comparison):

Accuracy: 0.83787 Precision: 0.37158 Recall: 0.31250
F1: 0.33949 F2: 0.32276

3. What algorithm did you end up using?

Two algorithms were applied to the classification problem. Next the performance is presented:

Gaussian Naive Bayes:
Accuracy: 0.84300 Precision: 0.48581 Recall: 0.35100
F1: 0.40755 F2: 0.37163

Total predictions: 13000 True positives: 702 False positives: 743
False negatives: 1298 True negatives: 10257

Decision Tree:
The chosen parameter for k was 3. Here are the selected features and evaluation metrics:

exercised_stock_options = 25.0975415287
total_stock_value = 24.4676540475
bonus = 21.0600017075

Accuracy: 0.80062 Precision: 0.36116 Recall: 0.38500
F1: 0.37270 F2: 0.37998

Total predictions: 13000 True positives: 770 False positives: 1362
False negatives: 1230 True negatives: 9638

Based on the performance Naive Bayes was selected for the final analysis.

4. What does it mean to tune the parameters of an algorithm, and what can happen if you don’t do this well?

Tuning an algorithm is the process of adjusting the parameters that affect the algorithm behaviour in order to maximize its performance. If this is not done well, we would end up with poor performance in the classification or regression results of the algorithm.

The chosen algorithm Gaussian Naive Bayes has no parameters to tune. But if I had chosen the DecisionTreeClassifier I would have tuned the parameters using GridSearchCV to find optimal paramters which are tested using cross validation. The following is an example of such tuning:

```
parameters = {criterion':('gini', 'entropy'), 'max_features':[2,10,15]}
model = DecisionTreeClassifier()
clf = grid_search.GridSearchCV(model, parameters)
clf.fit(features, labels)
```

5. What is validation?

Validation is the process of testing the performance of your algorithm in a test data set which is different than the training data set. This ensures that you are truly testing how the algorithm will perform in new situations, and not just how it performs when it has overfitted around your train data.

This analysis has been validated using the strategy defined in the tester.py script. Which employes the Stratified ShuffleSplit cross validation technique to assign samples to training and test data respectively.

6. Give at least 2 evaluation metrics and your average performance for each of them.

Precision and Recall are the two evaluation metrics used in the analysis. Recall refers to the probability of positively identifying as POI a person that is known to be of interest. While Precision refers to the probability that once having identified someone as POI, that person is actually of interest in the investigation.

Here are the results of the algorithm:

Accuracy: 0.84300 Precision: 0.48581 Recall: 0.35100

F1: 0.40755 F2: 0.37163

Total predictions: 13000 True positives: 702 False positives: 743

False negatives: 1298 True negatives: 10257