Kathleen Cravotta **December 2016**

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. As part of your answer, give some background on the dataset and how it can be used to answer the project question. Were there any outliers in the data when you got it, and how did you handle those? [relevant rubric items: "data exploration", "outlier investigation"]

The goal of the project is to find patterns in the data that may be used to signal whether or not a person was involved in fraud at Enron. The data used in my code has financial data, such as bonus and salary, as well as summary email data, such as the number of emails from that person. This data was compiled by Katie or her colleagues using publicly available financial and email information of Enron employees or contractors.

I explored the data using 2-D visuals and max check to see if there were any outliers. The financial outlier 'TOTAL' popped up, which we discovered during the lessons. I removed it from the working dictionary before formatting the features for the machine learning algorithms.

2. What features did you end up using in your POI identifier, and what selection process did you use to pick them? Did you have to do any scaling? Why or why not? As part of the assignment, you should attempt to engineer your own feature that does not come ready-made in the dataset -- explain what feature you tried to make, and the rationale behind it. (You do not necessarily have to use it in the final analysis, only engineer and test it.) In your feature selection step, if you used an algorithm like a decision tree, please also give the feature importances of the features that you use, and if you used an automated feature selection function like SelectKBest, please report the feature scores and reasons for your choice of parameter values. [relevant rubric items: "create new features", "properly scale features", "intelligently select feature"]

I used some intuition based on the 2-D visuals (see NotebooksAndHTML/ All_POI_Classifiers_Code.html for details) and looked at the importance of features in some of the classifiers. Since the number of emails per person could be on different scales, I used a ratio in place of the number of emails to and from a person of interest so that these would be in balance with the person's email activity. I also used ratio in place of exercised stock options, because that activity would be bound by the individual's held stock value. Ultimately the ratio for exercised stock options didn't seem to be used by the classifiers. The SVM classifiers did not work, so I used a scaler on the features for them. It didn't help! In general, more features got better results.

Features: 'salary', 'bonus', 'total_payments', 'exercised_stock_options', 'shared_receipt_with_poi', 'expenses', 'email_to_poi_ratio', 'email_from_poi_ratio', 'exer_stock_ratio'

Grid Search Decision Tree Importance:

salary : 0.0799353623057
bonus : 0.369159274301
total_payments : 0.0

exercised_stock_options : 0.0807861302916
shared_receipt_with_poi : 0.0
expenses : 0.470119233101
email_to_poi_ratio : 0.0
email_from_poi_ratio : 0.0
exer_stock_ratio : 0.0

3. What algorithm did you end up using? What other one(s) did you try? How did model performance differ between algorithms? [relevant rubric item: "pick an algorithm"]

I used a grid search with a decision tree varying the depth and min samples split. It worked better than the default decision tree, and slightly better than the ada boost decision tree. It overfit. K-neighbors had less success. Some algorithms didn't classify at all, like the Gaussian NB and the SVM. There was an issue with the SVM.

4. What does it mean to tune the parameters of an algorithm, and what can happen if you don't do this well? How did you tune the parameters of your particular algorithm? (Some algorithms do not have parameters that you need to tune -- if this is the case for the one you picked, identify and briefly explain how you would have done it for the model that was not your final choice or a different model that does utilize parameter tuning, e.g. a decision tree classifier). [relevant rubric item: "tune the algorithm"]

The parameter tuning can help to prevent over-fitting with high variance, and underfitting with high bias. I used a grid search varying the depth and min samples split to see if I could find better parameters, but it was still hard to find a good fit for the data. I also ran some of the parameters manually so that I could set up different test sets. I thought about if the grid search I used was trying to find a good solution, so I specified the scoring solution to 'f1'. This ties in to validation below.

5. What is validation, and what's a classic mistake you can make if you do it wrong? How did you validate your analysis? [relevant rubric item: "validation strategy"]

Validation can be wrong when using the same data that trained the classifier, to test the classifier. When the classifier over-fits to the training data, it will appear to have high accuracy. Another thing we are looking at when validating our Enron data set is that the occurrence of positive nodes is sparse. In those cases, only looking at accuracy will make the algorithm appear to have good performance.

To do good validations, separate data should be used to test the algorithm, that wasn't used to train the algorithm. I used a fixed split of data as I explored different classifiers, and then I used k-fold testing to get a closer look at the performance with different splits of data. Then I tested the classifiers with the provided tester. When I looked at the performance, instead of looking at accuracy, I focused on Precision and Recall, below.

6. Give at least 2 evaluation metrics and your average performance for each of them. Explain an interpretation of your metrics that says something human-understandable about your algorithm's performance. [relevant rubric item: "usage of evaluation metrics"]

Precision and Recall are especially important metrics for the sparse POI classifier. Precision is measures how likely the classifier is to be correct when it identifies a person of interest. My Grid Search DT is on average 34% correct when it identifies a person of interest. Recall is how likely the classifier is to correctly identify a person of interest. My Grid Search DT finds on average 30% of the persons of interest it sees.

```
C:\Users\KCRAVOTT\AppData\Local\Continuum\Anaconda2\lib\site-packages\sklearn\me
trics\classification.py:1113: UndefinedMetricWarning: F-score is ill-defined and
 being set to 0.0 due to no predicted samples.
  'precision', 'predicted', average, warn_for)
GridSearchCV(cv=None, error_score='raise',
       estimator=DecisionTreeClassifier(class_weight=None, criterion='gini', max
_depth=None,
            max_features=None, max_leaf_nodes=None,
            min_impurity_split=1e-07, min_samples_leaf=1,
            min_samples_split=2, min_weight_fraction_leaf=0.0,
            presort=False, random_state=None, splitter='best'),
       fit_params={}, iid=True, n_jobs=1,
       param_grid={'min_samples_split': [2, 3, 5], 'criterion': ['gini', 'entrop
y'], 'max_depth': [2, 3, 5, 8, 10, 15]},
       pre_dispatch='2*n_jobs', refit=True, return_train_score=True,
       scoring='f1', verbose=0)
        Accuracy: 0.82893       Precision: 0.34065      Recall: 0.30250 F1: 0.32
044     F2: 0.30943
        Total predictions: 15000        True positives:  605    False positives:
 1171   False negatives: 1395   True negatives: 11829
```