P5: Identify Fraud from Enron Email

By William Autry

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”]

   Enron was an American energy, commodities, and Services Company based in Houston, TX. Before its bankruptcy in 2001 due to corporate fraud, Enron employed approx. 20,000 staff and was one of the world's major electricity, natural gas, communications and pulp and paper companies, with claimed revenues of nearly $101 billion during 2000. The Enron scandal data was made public by the US Federal Energy Regulatory Commission during its investigation. The data has been widely used in machine learning.

   For this project, I explored both email & financial data. The goal of this project is to build a classifier to predict whether a person was involved in the Enron fraud scandal. The identified individuals are referred to as Persons of Interest. The dataset consists of 146 employees, 18 POIS, & 22 features. The amount of POIs in the dataset account for 12.3% (18/146) of the total number of employees, making this dataset imbalanced with 87.7% non-POIs vs. 12.3% POIs.

   There were several observed outliers, financial feature outliers existed for TOTAL, as well as the top four salary earners: Jeffrey K Skilling, Kenneth L Lay, Mark A Frevert, & Mark R Pickering. These four individuals had the highest salaries for Enron. I removed TOTAL from the data, as this value was a culmination of all salaries/bonuses and added no value to my research.

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 importance 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”, “intelligently select features”, “properly scale features”]

   I created new features based on the premise that POI’s send emails to other POI’s at a rate higher than for the general population.

   Created features:
   ‘fraction_from_poi’: Fraction of all emails sent to a person by a POI.
   ‘fraction_to_poi’: Fraction of all emails a person sent directly to a POI.

In a previous submission I tried Linear SVC for feature selection, but determined that SelectKBest was most useful for this dataset. Univariate feature selection works by selecting the best features based on univariate statistical tests. SelectKBest removes all but the k highest scoring features. The features provided by this selection method were: 'total_payments', 'loan_advances', 'deferral_payments', & 'salary'. I did not perform any scaling.

I later revisited my feature selection to add in 'fraction_from_poi' & 'fraction_to_poi' to determine their effect on my algorithm. I compared my initial features with the larger feature set to determine the impact on the algorithms, in terms of performance metrics, when including my newly created features.

Including more features actually had a positive effect on the performance of the algorithm. The two created features were important as they greatly improved the precision & recall values.

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"]

To the best of my knowledge, Random Forest is a more advanced take o Decision Trees so I will avoid Decision Trees completely in favor of Random Forest. AdaBoost can be summarized as "adaptive" or "incremental" learning from mistakes. The AdaBoost model has a lower bias than an individual decision tree, which makes it less likely to underfit the training data.

I tried a few of the algorithms mentioned in the lesson.

SelectKBest Features				
Model	**Time**	**Precision**	**F1**	**Recall**
Naïve Bayes	0.077 s	0.132	0.12	0.115
AdaBoost	8.997 s	0.152	0.12	0.100
Random Forest	2.715 s	0.193	0.19	0.205

Random Forest has proven to provide the greatest precision.

SelectKBest Features + Created Features				
Model	**Time**	**Precision**	**F1**	**Recall**
Naïve Bayes	0.081 s	0.099	0.10	0.105
AdaBoost	8.823 s	0.174	0.15	0.135
Random Forest	2.900 s	0.303	0.26	0.240

For both sets of feature selections, the Random Forest classifier outperformed the others. The features from my initial feature selection prove that my created feature do hold some value. In comparison the larger feature selection that includes the created features outperforms the initial feature selection.

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? What parameters did you tune? (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 items: “discuss parameter tuning”, “tune the algorithm”]

Tuning the parameters of an algorithm indicates that an algorithm has been optimized to better fit the data set. For AdaBoost, the number of weak estimators and their complexity. For Random Forest, the parameters are the number of trees and the number of features. I was able to test several parameter changes and find the optimal values by viewing the average accuracy as an estimate of out-of-sample accuracy since I implemented ‘cross_val_score.’

If done wrong, you will negatively affect the precision and recall of your algorithm. I was able to tune the ‘max_depth’ and ’min_samples_leaf’ to improve the recall of my Random Forest algorithm. As the recall improved, the precision fell a little, but this had to be performed to get the recall value above 0.3.

Final feature selection: ‘total_payments’, ‘loan_advances’, ‘deferral_payments’, ‘salary’, ‘fraction_from_poi’ & ‘fraction_to_poi’

Final Performance				
Model	**Time**	**Precision**	**F1**	**Recall**
Random Forest	2.732 s	0.343	0.34	0.34

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 items: “discuss validation”, “validation strategy”]

   Validation is the process of checking the algorithm’s results against data that was not used for training the algorithm. This is important to avoid over/under fitting a model. Stratification is important to address defects in the classification algorithms, as they are too easily biased by over- or under-representation of classes.

   I initially chose to test K-fold cross-validation to split the data into K equal partitions. However, the imbalanced data ultimately forced me to choose the StratifiedShuffleSplit function to split the data into training & testing sets. The StratifiedShuffleSplit function was a better choice than other choices due to the ratio of POIs to non-POIs being so low. This cross-validation object is a merge of StratifiedKFold and ShuffleSplit, which returns stratified randomized folds. The folds are made by preserving the percentage of samples for each class.

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”]

   The final Random Forest model had a precision of 0.587, recall of 0.490, and F1 of 0.534. The precision value indicates that 59% of the individuals identified as POIs will be accurately

identified, while 41% will be falsely identified. The recall value refers to the true positive identification rate, which is how often the model makes a correct identification. The F1 score is a combination of the two metrics precision & recall. [F1 = $2 * \frac{(precision * recall)}{(precision + recall)}$]