# Identify Fraud from Enron Email

Daniel Suruagy
November, 28 2017

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”]

This project goal is to try to predict if an Enron employee is a POI - Person of Interest - or someone who may have committed fraud that has collapsed this company. Some examples of POI are the company president and its CEO. To do this prediction, it was required to analyse some financial data, like salary, stocks, bonus, and email data, like number of messages sent or received, from POIs or not, for example.
The dataset is formed by 146 data points, identified by the employee name and his/her data. Part of the information was acquired from a payments report. From newspaper informations, 18 (eighteen) people were marked as POI, because were investigated or involved with fraud. From the e-mail corpus, which was made public, the number of emails sent or received was acquired. One of the data points was identified as an outlier, because corresponds to the Total sum of the other financial data points, as part of the payments report. This data point was removed from the dataset before the analysis.

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

I’ve created three new features, which are comparisons between the number of emails sent or received by people to or from POIs on this dataset. They are: **sent_to_poi_rate**, the rate of emails sent from this person to poi (from_this_person_to_poi), in comparison to the total messages sent (from_messages); **shared_poi_rate**, the number of emails that are also received by POIs(shared_receipt_with_poi), related to the total messages received (to_messages); **received_poi_rate**, the rate of emails received from POI (from_poi_to_this_person) in comparison to the total of received messages (to_messages).

Because features values are in a different range, such as financial values and number of emails exchanged, I thought it would be important to do some scale before doing feature selection. I've used Min Max Scale because I thought it would do a better data standardization, and would show a better performance with the classifiers I've tried to use. After that, I've used SelectKBest, using values of k equal to 5 or 7, because they provided better performance too. These are the selected features: ['loan_advances' 'bonus' 'total_stock_value' 'exercised_stock_options' 'sent_to_poi_rate'], with the following scores: [ 6.74274761 5.19334926 5.54483965 6.92747711 4.71795058].

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've obtained the required performance, with a score of above 0.3 in precision and recall, with AdaBoost Classifier, with a Decision Tree Classifier as the base estimator. I've tried a standalone Decision Tree Classifier and Random Forest Classifier, which presented the second and third best performance respectively, all above 0.3 in the same metrics shown above. The differences were very small, with the parameters I've used.

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"]

The SciKit Learn library comes with a large variety of algorithms, which have default values for a number of parameters. These default values could present a not so good performance in some situations, so it is important to explore other values before decide if the algorithm is a good choice or not. I think it is important to explore values carefully. If we provide a wide range of parameter values, the analysis of better performance can be an exhaustive work, when made manually. However, if few of them were experimented, before doing a better analysis, they could give a worse performance.

I've used GridSearchCV, to explore various parameters combinations, before decide which one presents a better performance. I was exploring specific parameters of Decision Tree algorithms, which I've used in three different ways, like criterion and max_depth. Afterwards, I decided to explore other parameters of pipeline components, like PCA number of components or the number of estimators used by AdaBoost or RandomForest Classifiers. The number of features used for SelectKBest also was explored. I've reached a large number of parameters options that, together with the StratifiedShuffleSplit number of splits, has increased the running time, so I decided to select only some values that provided better results for the analysis.

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 activity of check if one algorithm has really learned from training data and if it can predict correctly, based on testing data. A classic mistake occurs when data used for training is also used for testing the model. In this case, it will result in a overfit model, with a high bias, that will not predict well when receive new data incomes. In my analysis, I've used Grid Search cross validation, with Stratified ShuffleSplit, which search for optimized parameters for the algorithm, using different combinations of datasets for training and testing. This approach guarantees that there will be always a POI in the training and testing data, which is important, because there are less POI information than non-POI.

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"]

In my work, the first evaluation metric used was precision, which is a number that represents how many value predictions were correct (true positive) between all that was assigned to this specific value (true positive added to false positive). The average value was 0.35303 + 0.35961 + 0.33333 = 0.34865.
The second evaluation metric was recall, which is the number that represents how many value predictions were correct (true positive) between all that is really true, but ended assigned to another class (true positive added to false negative). The average value was 0.34950 + 0.35350 + 0.32550 = 0.34283.