Enron Submission Free-Response Questions

By Michaela Wonacott

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?

In 2000, Enron was one of the largest companies in the United States. By 2002, it had collapsed into bankruptcy due to widespread corporate fraud. In the resulting Federal investigation, a significant amount of typically confidential information entered into the public record, including tens of thousands of emails and detailed financial data for top executives. In this project, I will play detective, and put my new skills to use by building a person of interest (POI) identifier based on financial and email data made public as a result of the Enron scandal.

The total number of data points in the enron data is 146. There are 18 POIs and 128 non-POI(s). There are 21 features used. All the features had some amount of missing data. But features that had many missing data ('many' as in greater than 50% of data values missing) are the features deferral_payments, long_term_incentive, restricted_stock_deferred, loan_advances, deferred_income, and director_fees.

To find any outliers in the data set I decided to create a scatter plot using the features 'salary' and 'bonus.' From this I was able to see one striking outlier. This outlier was the name 'TOTAL' is the total of each feature for every enron employee (or data point) examined in the data set. Because 'TOTAL' is not a person we can examine, I removed it from the data set.
Further exploration of the dataset revealed two more items of interest. The first being that since the feature 'email_address' is unique to every person, it will not help determine which feature correlates to a POI. The second was that 'THE TRAVEL AGENCY IN THE PARK' is not a person but used by employees to make payments on the account of business related travel. Because of this, I decided to remove the feature 'email_adress' and the data point 'THE TRAVEL AGENCY IN THE PARK' from the dataset.

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?

I decided to create and implement five new features. These new features are 'bonus_to_salary', 'bonus_to_total', 'from_poi_ratio', 'to_poi_ratio', 'shared_poi_ratio'. Receiving a bonus could be an indicator of fraud depending on how much it made up a portion of their salary or total payments.The amount of emails to, from, and shared with POIs is an important metric, but I thought that it would be a better indicator of fraud to see out of what portion of emails are to,

from, and shared from a POI out of all their emails. This to see if people who communicated with POIs had a certain portion of communication out of that person's total communication.

To find the number of features I would use, I used kBest with a k-value = 15. Feature scaling is performed to normalize the range of each feature so that a feature cannot dominate another feature. I only applied scaling to Naive Bayes since Decision Tree is not affected by feature scaling.

What algorithm did you end up using? What other one(s) did you try? How did model performance differ between algorithms?

The algorithms I tried were Naive Bayes and Decision Tree. I used the test_classifier function for the tester.py to score each algorithm after they were fit. The scores I focused on from this function were the accuracy, precision, recall and F1 scores. After the initial performance and performance after turning, I decided to pick Naive Bayes since it has higher accuracy, precision, recall and F1 scores.

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?

Tuning the parameters of an algorithm can allow you to achieve the best performance for your algorithm. This is because algorithms depend on the selected parameter. I used GridSearchCV with the parameter scoring set to 'f1' for parameter tuning. For Naive Bayes, I used a pipeline to find the best k value to see if there was a more optimal amount of features for the algorithm. For Decision Trees, I focused on the parameters 'criterion' and 'max_depth' since they have a significant impact on a Decision Trees performance.

What is validation, and what's a classic mistake you can make if you do it wrong? How did you validate your analysis?

Validation makes sure that the algorithm does what it is intended to do by testing it. Validating, or testing data, gives an estimate of performance on an independent dataset and services as a check on overfitting. A class mistake is to tune the algorithm to the training data and not the test set. This can result in overfitting, leading to an algorithm that does not generalize well. To prevent overfitting, I used the cross validation technique Stratified Shuffle Split. I chose Stratified Shuffle Split because of the small size of the dataset.

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.

My average performance for precision was 0.38. My average performance for recall was 0.35. Precision is the probability the classifier identified is a real POIs (true positives) out of all the identified POIs (sum of true positives and false positives). For my algorithm, this means that the classifier identifies 38% of the real POIs out of the total POIs identified by the classifier. Recall is the probability that the classifier will identify a real POI if it is indeed a real POI. This means out of all the real POIs in the dataset, only 35% of them were identified. This performance leads me to believe that there are probably better parameters and/or algorithms that could better identify a POI.