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”]

This project aims to apply machine learning into the famous Enron dataset to try and identify persons of interest (POIs) using the publicly available financial and email data of Enron’s top executives and employees. This is a real test of machine learning power to deal with real and non-perfect data. The data contains 146 records in the data set with 20 features and 1 label (POI). In addition to that, 18 people are identified as POIs.

When I started exploring the data, it has an extreme outlier across almost all the variables. When I checked it, I found that it is called “TOTAL”, which means it’s the sum of the results of all employees, not a real one, so I dropped it. In addition to that, I’ve deleted “THE TRAVEL AGENCY IN THE PARK” because it’s not a person and “LOCKHART EUGENE E” because all his values are missing.

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

I used MinMaxScaler as a scaler to normalize the dataset to have a fair comparison between the different features. In addition to that, I’ve engineers some features, that includes ['from_poi_ratio' , 'to_poi_ratio' , 'bonus_to_salary_ratio' , 'total_money']. While testing the different classifiers, I compared the original data with the one after adding the new features and found that the results are better with the newly added features, so I kept them.

After that, I used SelectKBest to select features and tuned the K value using GridSearchCv to test it on different sittings (all integers from 7 to 15) and ended up with 13 features, which are: ['salary', 'total_payments', 'loan_advances', 'bonus', 'deferred_income', 'total_stock_value', 'exercised_stock_options', 'long_term_incentive', 'restricted_stock', 'shared_receipt_with_poi', 'to_poi_ratio', 'bonus_to_salary_ratio', 'total_money'] (the last 3 are from the features I’ve added)

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 tested the following classifiers: DecisionTreeClassifier, SVC, KNeighborsClassifier, RandomForestClassifier and LogisticRegressionI. I compared their F1 score using train_test_split with a split of 40% to 60% between the training and testing sets. Based on that, I found that DecisionTreeClassifier was the best option to move forward with.

After tuning the parameters, I did cross-validation using StratifiedShuffleSplit with 1000 folds as it was set up in test_classifier in tester.py.

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"]

Like we do with the radio's frequency to get to the right channel, the parameters of an algorithm need to be tuned to improve its performance. Without parameters' tuning, we won't be able to get the best of the selected model.

For the DecisionTreeClassifier, I used the GridSearchCV to automatically search over specified values for two parameters: max_depth, which is the maximum depth of the decision tree, and min_samples_split, which is the minimum number of samples required to split an internal node.

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 aims to evaluate the developed model before using it. The risk usually is to overfit, which risks having a model that fits perfectly to the used dataset but can't be reliable if we try to use it on a separate dataset. So, the approach is train on a sub-set of data and then the model can be tested on the other sub-set.

For our dataset, the issue of having a small sample forces us to use cross-validation, which was the methodology I used. Stratified Shuffle Split shuffles the data every iteration and then splits it. Also, Stratification ensures having the same split between the two labels (POI vs. non-POI), so it was selected here due to the imbalance between them.

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"]

To evaluate the model, I used precision and recall where the model had an average precision score of more than .6 and a recall score of more than .5.
In this case, precision measures the portion of those who were identified by the model as POIs that are actually POIs. On the other hand, recall measures the portion of the actual POIs that were identified correctly by the model.