Enron Submission Free-Response Questions

A critical part of machine learning is making sense of your analysis process and communicating it to others. The questions below will help us understand your decision-making process and allow us to give feedback on your project. Please answer each question; your answers should be about 1-2 paragraphs per question. If you find yourself writing much more than that, take a step back and see if you can simplify your response!

When your evaluator looks at your responses, he or she will use a specific list of rubric items to assess your answers. Here is the link to that rubric: Link to the rubric Each question has one or more specific rubric items associated with it, so before you submit an answer, take a look at that part of the rubric. If your response does not meet expectations for all rubric points, you will be asked to revise and resubmit your project. Make sure that your responses are detailed enough that the evaluator will be able to understand the steps you took and your thought processes as you went through the data analysis.

Once you've submitted your responses, your coach will take a look and may ask a few more focused follow-up questions on one or more of your answers.

We can't wait to see what you've put together for this project!

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 this project is to generate an algorithm that can evaluate a set of data for a given individual from the Enron dataset to determine if the given person was or was not a person of interest in the case. Machine learning is helpful to this as we can use the dataset provided to teach the computer how to evaluate the data to come to a conclusion which is as repeatable and precise as we can manage. Since the dataset is fairly comprehensive in depth, even if it is limited in length was can use the features to create a machine learning algorithm that will help us predict if a given person is likely a person of interest.

    In this case the data is organized as a set of nested dictionaries with each person's set of data being in a dictionary with their name as the key. We can iterate over these dictionaries to evaluate the data we are getting back. In looking at the data there is one strong outlier that stands out among the rest for its extreme values, that is the Total line. To be able to properly evaluate the data this line will be removed from the data set.

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"]

   In creating my POI Identifier, I went through a number of iterations of feature combinations. In the end my feature list was, poi, salary, director fees, total stock value, total payments, exercised stock options, from poi to this person, and deferral payments. To develop this list, I added features one at a time, testing to see if their addition increased my Precision and Recall scores. If the addition increased the scores it stayed in the list; those that contributed to stagnant or reduced scores were removed. I then went through and tested initial list to see if it could be refined, during that process I removed a couple of features leaving me with the list previously presented.

   I engineered a feature that created a ratio of emails from this person to poi compared to from poi to this person. This was an attempting to represent back and forth conversation as opposed to generic corporate emails. In the end this feature did not have any impact on the validity scores and I left it off the feature list.

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 ended up using KNeighborsClassifier as I was able to gain the desired results using this algorithm. Along the way I tried a number of other algorithms including:

   GaussianNB ()
   Precision: 0.22688 Recall: 0.31150

   DecisionTreeClassifier(max_depth = 10)
   Precision: 0.32687 Recall: 0.34550

   AdaBoostClassifier(n_estimators = 100)
   Precision: 0.27503 Recall: 0.21700

KNieghborsClassifier(n_neighbors=3, weights='uniform')"

Precision: 0.55754   Recall: 0.32700

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"]

   Tuning the parameters of an algorithm allows you refine the results provided by the code. Since the parameters can have quite an effect of the results, it is important to spend time tuning the parameters to get the best results. In tuning the parameters for my algorithm I tested several options to see which provided the best validation results. In this case I found that setting n_neighbors to 3 and weights to uniform yielded the strongest validation scores for Precision and Recall.

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 is a means to test if your algorithm can be used on unknown data sets to yield an expected result. Common validation testing metrics are Accuracy (correct to incorrect data point classifications), Precision (low ratio of false positives) and Recall (low ratio of false negatives). In validating your code, it is important that you validate on data that is different than what you trained your algorithm on. Testing on the same data points you trained on is a classic mistake that can cause you to overstate the confidence in your algorithm and lead to poor results when run on unknown data point.

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"]

   My final code provides an accuracy rating of .87567 which is a fairly high accuracy rating. This means that roughly 87 percent of the time my algorithm will accurately predict if a person is a Person of Interest (POI) in the Enron dataset. The code also has a Precision of .55754, meaning that just over half of the people identified as POI's are true positives. Additionally, my algorithm has a Recall score of .32700 this would mean that among those that are actually POI's I was able to identify roughly 33% of them correctly. Overall this dataset is designed to error on the side of caution for investigating and prosecuting those that played an active role in the Enron fraud activity, leaning towards confidence in those being charged instead of casting the net wide and risk charging more innocent people than should be. Ideally I would prefer to have both Precision and Recall but higher but was unable to achieve that despite multiple approaches over the course of the project.