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**] 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"]

For this specific project, machine learning can be used to predict the people of interest based on a provided dataset and algorithms, if a person of interest has the likelihood of committing fraud under certain conditions a human has more trouble understanding or predicting on their own.

There were no outliers under the investigation that was made that were removed.

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's 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"

The features used in this algorithm were the default feature set provided by Udacity. Reviewing the features provided by Udacity, all of them seemed to be important for their choice of being selected to run against the algorithms. I did scale the data so computation will be reduced using *scaller.fit_transform*.

I have engineered a new featured called *pct_amount_paid_from_bonus* that took the features from *bonus* and *total_payments* and used these to see how much by percentage of the bonus was possibly used to pay off the total payments. I did actually run and use the newly made feature against Adaboost and the results came back perfect and the accuracy and recall were above their needed score still so I kept the feature to be used.

No algorithm was used for the new feature, and it was a calculation made against two default features.

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”]

***Adaboost* was the algorithm of choice that was picked with the best results. I compared the best algorithms that carried the best results by using pycaret to compare against different algorithms simultaneously and Adaboost came back with the best score.**

Using pycaret, it provided us with a list of multiple other algorithms and their performance, with Adaboost giving us the best results.

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 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 parameters of an algorithm is a process which one goes through in which you optimize the parameters that impact the model in order to enable the algorithm to perform the best.

We selected the Adaboost algorithm, and tuned its parameters using pycaret.

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 a model validation technique that assesses how the results of a statistical analysis will generalize to an independent data set. A classic mistake I often make is overfitting a model.

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”]

I used 3 evaluation metrics: accuracy, precision, and recall. The best algorithm I applied to this project was Adaboost.

Accuracy demonstrates how close a measured value is to the actual (true) value. Precision is a metric that measures an algorithm's power to classify true positives from all cases that are classified as positives. Recall is a metric that measures an algorithm's power to classify true positives over all cases that are actually positives.