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"]
    - The goal of this project is to create a list of persons of interest from the email and financial data provided regarding the Enron scandal in the early 2000s. Machine learning can be useful in this process because it is a way to apply statistical methods and algorithms to find patterns in the data. This is helpful because it can be faster and more precise. It can also find relations that are not as easily seen by just looking at the data.
    - This data set includes a list of Enron employees with some of their financial and email information such as salary, bonus, and to/from emails. We can combine these features using machine language to come up with a list of people who were possibly involved in the fraud that occurred at Enron. There are 18 persons of interest, 143 data points and 20 features. Three more features were created and two of them used. There are also 1352 total features with no values (NaN).
    - I found a few outliers. The biggest was that the Total column was included in the list of employees. This was really throwing off results because the amounts were so much larger than anyone else. Therefore, the people who really did have outsized salary or bonuses, for example, were not standing like they should. A couple other outliers were The Travel Agency in the Park and Eugene Lockhart. The travel agency is not a person and Mr. Lockhart had no data in his features. I removed them 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 importance 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 didn’t want to choose too many features to avoid overfitting. This could make the results in the training data look really god, but then, surprisingly have poor performance on new test data.
- I chose the top ten using the univariate selection tool SelectKBest. These are the features with their scores:
    1.

'total_payments'	8.77277773
'restricted_stock'	9.21281062
'long_term_incentive'	9.92218601
'deferred_income'	11.4584766
'to_poi_message_fraction'	16.4097126
'salary'	18.289684
'bonus'	20.7922521
'sum_salary_bonus'	22.5903575
'total_stock_value'	24.1828987
exercised_stock_options'	24.8150797

- I did not do feature scaling because it did not apply to the algorithms I tried.

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 tried using two algorithms, Gaussian Naive Bayes and Decision Tree. The accuracy scores for both were 0.8837209302325582. The difference showed when I ran the predictions, precision and recall scores. The Gaussian Naïve Bayes algorithm performed as expected, but the Decision Tree returned zero scores. Also the Gaussian Naïve Bayes algorithm had a faster training time at .009 seconds compared to .012 seconds for the Decision Tree.
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”]
    - Tuning the parameters of an algorithm means adjusting the parameters to find the best combination that results in the highest accuracy. If it is not tuned correctly you could have misleading or less than optimal results. The Gaussian Naïve Bayes algorithm is generally simple and straightforward which, I think, makes it a great tool. But this also means there’s not much, if any, tuning to be done.

- I tried using GridSearchCV but it did not find any parameters that would provide better performance. Then I went back to the features to see if making changes there would help tune my results. First, I realized that I was adding from_poi_message_fraction to the features list even though it was not in my top ten list from running SelectKBest. Then I tried removing salary and bonus because I was creating sum_salary_bonus and having both seemed redundant. There wasn't much difference in the scores, so I kept the salary and bonus features.

File used	features changed	Accuracy	Precision	Recall
poi_id.py	start-no changes	0.88372093	0.5	0.6
tester.py	start-no changes	0.8448	0.39474	0.3075
poi_id.py	removing from_poi_message_fraction	0.88372093	0.5	0.6
tester.py	removing from_poi_message_fraction	0.8448	0.39474	0.3075
poi_id.py	removing salary, bonus	0.860465116	0.4	0.4
tester.py	removing salary, bonus	0.84627	0.39908	0.3025

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 the process of splitting your data into train and test groups, then running your processes and tuning thorough the train set, and then testing your choices by running them through the test set. A common mistake is overfitting by choosing the settings that give the result you are looking for specific to just the train set.
   - This analysis was validated using the Stratified ShuffleSplit cross validator. This shuffles the data and then splits it into a train and test set. The multiple shuffling and then splitting helps ensure that data isn't skewed in either the train or test set which could cause a result that isn't as accurate as it appears to be.
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"]
   - The average performance with the Gaussian Naïve Bayes algorithm was 0.39474 for precision and 0.30705 for recall. The precision refers to the how many of the results are relevant, so 39% of results in this data set were relevant. The recall refers to how many of the results were classified correctly, which is 30%. This means that this algorithm is identifying about 30% of the persons of interest (poi). The precision score means the person of interest identified is correct about 39% of the time.