Enron Submission Free-Response Questions

Question 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”]

Answer: The goal of this project is simple. We need to find the POIs out of many employees at Enron. We have wide range of variable of each employee and out of these we need to also find that which variables contributes most to the POI detection. Only a machine learning approach would be able to decide if the person is an POI or not because this is impossible for human eyes to find the pattern into the data. The dataset is very biased as it should be for all the financial frauds. Hence prediction for a machine learning algorithm is also not straight forward. We need a lot more feature engineering and data munging before we can pass it to the classifier. There is also a lot of hyper parameter tuning. While data was being explored, one outlier was found which is removed from Features base class. All the basic exploratory data analysis was given in **eda.ipynb** file.

Question 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”]

Answer: There were two steps for feature selection. One is during classifier testing and the other is during fine tuning of the classifiers. During the classifier testing phase, 3 different types of feature selection is used. These feature selection is not done manually but rather using feature importance scores from classifiers like **xgboost** and random forest and feature selection

methods like **SelectKBest**. **XGBoost** with cross validation was also used to using **GridSearchCV** with an evaluation metric for F1 score. These feature selection was pretty successful because even without any classifier tuning, a score of more than the prescribed score of 0.3 was achieved for almost every classifier for precision and recall. During fine tuning we have also used **PCA** and **SelectKBest** with cross validation with different types of scaling like **MaxAbsScaler**(), **StandardScaler**(), **MinMaxScaler**(). However it is worth nothing that most of the cases, **MaxAbsScaler**() performed best.
5 features of the following

a. **poi_interaction,**
b. **income_ratio,**
c. **expenses_std,**
d. **deferral_payments_std,**
e. **other_std**

The reasons for creating these features were mentioned in the actual code but to put in brief
poi_interaction – If a person receives and sends a lot of emails from and to the POIs, chances are that they are POI themselves
income_ratio – Shows the ratio of an employee's total income standardized by the total payments. Lesser the value, more mismatch in his/her income and more chances of being POI
expenses_std, deferral_payments_std, other_std different payments standardized by the total payments.

Look at eda.ipynb for feature importance plots.

Question 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"]

Answer: I ended up using **NearestCentroid**(), a different type of **KNN**. This gave the best performance for both **F1** and **F2** score. I have also tried **SVC** with class penalty. This is because **SVC** assumes an unbiased class and this was not an unbiased class. Naïve Bayes was also used with **PCA** and cross validation. Because **PCA** would reduce the assumption that the features are related and keeps the **Naïve Bayes** as Independent as possible. The other algorithms that I tried is

A. **GradientBoostingClassifier**
B. **AdaBoostClassifier**

They perform poorly. For more info look at the **README.md** file.

Question 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"]

Answer: Tuning the parameters means adjusting the parameters of an algorithm so that it can handle a particular dataset better. Different parameter settings will result in different
decision boundaries. If we don't tune the parameters well, the algorithm won't be
able to generalize a dataset well and the final classification result might be less accurate.

While exploring for algorithms, I used **GridSearchCV** for parameter tuning. For
example, for **NearestCentroid**, I tried tuning two parameters: **shrink_threshold [None, 0.1, 0.6, 0.7, 0.8, 0.9, 1, 2, 5, 10] metric**': **["euclidean", "manhattan"]**. With different number of features used with **PCA** and **KBest** and with different number of scalars and tuned the classifier for precision.

Question 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"]

Answer: Validation is a way to assess the performance of a machine learning algorithm using the given dataset. This is done by splitting a dataset into training and testing
datasets, and comparing our machine learning results with the labels in the testing
dataset. A classic mistake is to use all observations available in a dataset to train the
machine learning algorithm, and then to end up overfitting the given dataset. To assess the performance of my final algorithm, I used the Stratified Shuffle Split
cross-validation. This is because the Enron dataset is small and unbalanced (many

more non-POIs than POIs). Stratified shuffle split cross-validation could construct
new instances from the dataset to ensure better representation of POI class in both
training and testing datasets. Then, it does a randomized k-fold cross validation on
the dataset.

Question 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"]

Answer: For my final algorithm choice, **NearestCentroid**, the average performance is as follows:

- **Precision: 0.40778**
  This means that out of those identified as POIs by the model, only 40.8% are true POIs
- **Recall: 0.71300**
  This means that the model is only able to point out 71.3% of POIs accurately.
- **F1: 0.51883**
  This is a weighted average of true positives, false positives and false negatives scores. Higher F1 score means better prediction. My score was 51.8%
- **F2: 0.62016**
  This is a weighted average of precision and recall scores. Higher F1 score indicates higher precision and recall scores, and hence a better model. My score was 62%