**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!

**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 dataset was collected and prepared by the CALO Project (A Cognitive Assistant that Learns and Organizes). It contains data from about 150 users, mostly senior management of Enron, organized into folders. The corpus contains a total of about 0.5M messages. This data was originally made public, and posted to the web, by the Federal Energy Regulatory Commission during its investigation.
- In this project we will build a POI(person of interest) identifying tool based on the above dataset(By use email and financial data for 146 executives at Enron to identify POI in the fraud case) This report documents the machine learning techniques used in building a POI identifier. The dataset contained 14 financial features, 6 email features, and 1 labeled feature (POI). Of the 146 records, 18 were labeled as persons of interest. Some features (loan advances, directors fees) has many missing values, some not that much( Salary and total stock value).

- Using a manual strategy of inspection, I found that there were two obvious outliers. 'Total' and ''THE TRAVEL AGENCY IN THE PARK'.We can tell them based on our common senses.

**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"]**

- features end up using in my POI identifier:['poi', 'bonus', 'exercised_stock_options', 'total_stock_value']

- I used SelectBest to select best 3 features and use those features for all kinds of Algorithm. The score of feature importances is as below:

| | |
|---|---|
| bonus | 20.792252047181535 |
| exercised_stock_options' | 24.815079733218194 |
| total_stock_value | 24.182898678566879 |

- I did try with 5 features and 7 features under KNN. With 5 features, the result is only with a Precision: 0.41767 and Recall: 0.20800. With 7 features ,the result is even worse(Precision: 0.38045 Recall: 0.17900) The table is as below:

| Feature's Number | Precision | Recall |
|---|---|---|
| 3 | 0.59494 | 0.39950 |
| 5 | 0.41767 | 0.20800 |
| 7 | 0.38045 | 0.17900 |

- So I finally choose 3.In fact these 3 features are very consisitant with our common sense as well. I have read some materials and documentary film about the Enron scandal. Exercised stock options, bonus and total stock value can be highly relevant to the ethics of the management level guys.

- I didn't do scaling for the  3 features since the result of KNN is already ok.

**Feature Engineering**

After seeing your comment, I create a new features:"from_ratio" which is the ratio of "from_this_person_to_poi" in "from_messages".
The reason is that the ratio could imply the frequency difference of email to POI.
In the KNN method with 3 features, it does not have significant impact on the result.I guess it is not as important as the exercised stock options, bonus and total stock value

**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"]**

- After trying SVM(not get a good result), K-means, and KNN, I end up with using KNN.
- The difference between the result of K-means and KNN is as below:

```
KNeighborsClassifier(algorithm='auto', leaf_size=30, metric='minkowski',
        metric_params=None, n_jobs=1, n_neighbors=3, p=2,
        weights='distance')
            Accuracy: 0.86577   Precision: 0.59494   Recall: 0.39950       F1: 0.47801          F2: 0.42759
            Total predictions: 13000         True positives:  799 False positives:  544          False negatives: 1201          True
negatives: 10456

KMeans(copy_x=True, init='k-means++', max_iter=300, n_clusters=2, n_init=10,
   n_jobs=1, precompute_distances='auto', random_state=None, tol=0.001,
   verbose=0)
            Accuracy: 0.76485   Precision: 0.26584   Recall: 0.30000       F1: 0.28189          F2: 0.29248
            Total predictions: 13000         True positives:  600 False positives: 1657          False negatives: 1400          True
negatives: 9343
```

It can be easily found that, with the current features ,  the KNN is far better than the K-means.

**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 an algorithm or machine learning technique, can be simply thought of as process which one goes through in which they optimize the parameters that impact the model in order to enable the algorithm to perform the best as you wish.
- In this case, I found KNN was "lazy" enough without having to use more complicated methods The main parameter I tuned was actually k itself. k refers to the number of surrounding nearest neighbors to look at when voting on the majority class. I found that the optimal number to get the accuracy, precision, and recall I wanted was k = 3.
- I did tried with k =5 and k =7, the result is not as balanced as k =3. It is shown in the below tables

| K | Precision | Recall |
|---|---|---|
| 3 | 0.59494 | 0.39950 |
| 5 | 0.66115 | 0.33950 |
| 7 | 0.64562 | 0.15850 |

**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 performed to ensure that a machine learning algorithm generalizes well. A classic mistake is over-fitting, where the model is trained and performs very well on the training dataset, but markedly worse on the cross-validation and test datasets.

- I validated my dataset under the help of the test_classifier () function. The Stratified ShuffleSplit cross validation iterator provides train/test indices to split data in train test sets and run many times. Since our dataset is small and only 18 are labelled as POIs,we need to randomly split the data in many trials.

**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"]**

- **Precision** reflects the ratio of true positives to the records that are actually POIs, essentially describing how often 'false alarms' are (not) raised.Or how much percent is ture POIs within the POIs this model find ?1343 total positive predictions were made by

the model on the test data and the amount of these positive predictions that was were correct was 799. The precision of the model was just around 60%.

- **Recall** means the ratio of correct positive predictions made out of the actual total that were indeed positive (correct positive predictions + incorrect false negative predictions). Or how much percent this model can find of all real POIs in prediction?The model was able to achieve a recall of around 40%.

http://stackoverflow.com/questions/22903267/what-is-tuning-in-machine-learning
https://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm
Udacity DAND