**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 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 coach evaluates 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](https://docs.google.com/a/knowlabs.com/document/d/17-JwNQH1aRxtqMkJ6zpCL_68kh5F6uSbDXcJS26vZWY/pub)

Each question has one or more specific rubric items associated with it, so before you submit an answer, take a look at that the rubric.  If your response does not meet expectations, you will be asked to resubmit.

Once you’ve submitted your responses, your coach will take a look and 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 the project is to identify person’s of interest (POI) among Enron employees who may have committed fraud or broken the law. To do this, I will build a POI identifier that uses email and financial data released as part of the court case. Machine learning is useful for building a model that can determine whether someone is a POI. I will use Machine Learning techniques to build a classifier, to select the most important features to use for classification, and to test the accuracy of my classifier. Initial data discovery uncovered the following:

* total no. of data points = 146 people in the data set
* allocation across classes (POI/non-POI) = 18 POI, 128 non-POI
* number of features = 21 features are available
* are there features with many missing values? Yes.

There were many features with ‘NaN’ values. I removed these features from classification because the ‘NaN’ would be converted to 0, which could negatively affect my final algorithm. ‘NaN’, which means unknown, is a different value from 0. Features with large and moderate number of missing values: loan\_advances, restricted\_stock\_deferred, director\_fees, deferral\_payments, long\_term\_incentive, deferred\_income. I created scatter plots of certain features. A discovered an outlier - FREVERT MARK A – who was often grouped with clusters of POI but is not a POI. I did not remove him because I want to avoid over cleansing the data and remove useful information that might cause future new incoming data to be wrongly classified.

1. 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 doesn’t 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.)  If you used an algorithm like a decision tree, please also give the feature importance of the features that you use.  [relevant rubric items: “create new features”, “properly scale features”, “intelligently select feature”]

**Final feature set:** 'total\_payments', 'bonus', 'restricted\_stock\_deferred', 'director\_fees', ‘suspicious\_email\_ratio’

**Feature importances:** array([ 0.40618417, 0.45397068, 0. , 0. , 0.13984515])

**My selection process:** I began with the SelectKBest algorithm to determine which features were most important using an objective and systematic process. I took the top 8 highest scoring features and ran them through the Gaussian Naïve Bayes classifier one-by-one. I omitted the features that did not increase the accuracy, precision and recall to avoid overfitting.

Next, I engineered a new feature: the ratio of “suspicious” emails to total emails sent and received by a person. I created a text vectorizer from which I created a tf-idf matrix. Then I ran K-means with 60 clusters on the term document matrix to cluster emails. The top 5 email clusters with the greatest ratio of emails from POIs became the most “suspicious” email clusters. Ultimately, the new feature looks at how many emails someone has sent or received which appear in the same clusters as emails to and from other POIs. The logic is people who may be committing fraud could be emailing each other frequently and the content of their emails might provide clues that they are participating in fraudulent activities.

1. What algorithm did you end up using?  What other one(s) did you try? [relevant rubric item: “pick an algorithm”]

I used a Decision Tree algorithm. I tried a Naïve Bayes, C-Support Vector Classification, and KNearestNeighbor algorithm. I found Naïve Bayes and Decision Tree algorithm performed the best; the SVC and KNN both produced high accuracy but 0 precision and 0 recall. In the end, I chose to use the Decision Tree algorithm because I am limited in how much I can tune the parameters of a Naïve Bayes algorithm to optimize how it classified. I did not choose to scale my features because neither the Naïve Bayes nor Decision Tree requires feature scaling for better performance.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Naïve Bayes | Decision Tree | K Nearest Neighbors | SVM (SVC) |
| Accuracy | 0.35036 | 0.78871 | 0.84664 | 0.85714 |
| Precision | 0.15873 | 0.27015 | 0.12308 | 0.00000 |
| Recall | 0.82500 | 0.28150 | 0.01200 | 0.00000 |
| F1 | 0.26624 | 0.27571 | 0.02187 | 0.00000 |
| F2 | 0.44849 | 0.27916 | 0.00738 | 0.00000 |

1. 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 don’t 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 if you used, say, a decision tree classifier). [relevant rubric item: “tune the algorithm”]

Tuning parameters of an algorithm allows me to balance bias and variance. Depending on the data set, I tune the parameters of my Decision Tree algorithm to optimize its performance. For example, I can set the max\_depth of my Decision Tree classifier if I know that past a certain depth, going any further will hurt my classification because of the characteristics of my data set. I tuned my algorithm’s parameters by trying different parameter values and choosing the values that gave me the best validation scores in terms of precision and recall.

If I don’t tune parameters well, I will build a bad classifier that makes poor predictions. This can manifest itself in 2 ways: 1. I can create a bad classifier that doesn’t work at all with training or test data sets, and/or 2. I can fine-tune based on my training set, which ends up causing bad performance on my test set and overall generalizing for future incoming new data.

1. 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 checking I didn’t overfit to my training data set so my algorithm is generalizable to independent future incoming data. I validated my analysis by splitting my dataset into two groups: a training set, which is used to select features, train my classifier, and parameter tune my algorithm, and a test set which is used to check that I didn’t overfit my algorithm. In addition to splitting data into training and test sets, validation also involves looking at metrics to objectively gauge how well my algorithm is performing.

I validated my analysis by ensuring that precision and recall were at least 0.3. I discovered that accuracy alone was unreliable because it was often high (at least 0.8) but precision and recall would both equal 0, which meant the algorithm was not performing well. This happens because the test set was so skewed toward non-POI, partially because of the small data set and partially because we had few POI training examples. Precision and recall are more granular metrics concerned with true positives, false positives, false negatives that help me recognize if my algorithm is actually doing a good job classifying.

1. 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”]

**2 metrics:** precision = 0.32860, recall = 0.31250

Precision is the measure: of all the people I say are POIs, how many of them are actually POIs. Recall is the measure: of all the people who are truly POIs, how many of them was I able to identify as POI. The numbers provided (precision = 0.32860, recall = 0.31250) is the average metrics across multiple validation sets.