

## Enron Submission Free-Response Questions

- \* 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 project uses “supervised machine learning” to predict if a person is involved in the ENRON fraud case. The data is financial and email texts, collected and organized for teaching purposes and shared widely over the internet. The dataset consists of 145 employee names with their financial information and emails log for each employee. Every employee will be labeled for being person of interest(POI) or not ( True/False) i.e. indicted or not. The algorithm will try to predict if the person is POI by learning from the data using the important features like emails, salary bonus..etc

There are multiple outliers; however only one was to be removed which is “TOTAL”. There are some employees who have an outlying values for salary and bonuses; but that is significantly related to the fraud case; so we kept them.  
For example SKILLING JEFFREY K.

- 
- \* 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”, “intelligently select features”, “properly scale features”]**

I ended up using the following features: ['bonus', 'deferred\_income', 'long\_term\_incentive', 'total\_stock\_value']. The selection process started by a guess that failed; then I used the SelectKbest function for automated process but that again didn't work well. It gave me one feature that can have high score which is excersised\_stock\_options, but as I increase number of features the prediction accuracy will decrease. However building a model on one feature is not intelligent so I started to pick manually from those features that had high score on the SelectKbest function. The features I used had the following scores: ( ['bonus': 21.327, 'deferred\_income': 11.732, 'long\_term\_incentive': 10.222, 'total\_stock\_value': 24.752]

Also in the SelectKbest function I used the parameter `score_func = f_classif` because the `chi2` wouldn't work for the negative values in the dataset. Regarding the `k` parameter I tried 1,2,3,4,5,6,7,8,9,11 and 15.

Regarding the scaling I had to scale the features because I thought that their scales are different (money vs number of emails); of course after deleting text data (i.e. email address). The feature that I tried to add was to look at the email numbers as percentage rather than real numbers; so I added `from_poi_percentage` and `to_poi_percentage` of every person total email messages. However it didn't affect the result .

---

**\* 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 ended up using the gaussianNB.

I tried Decision tree classifier and SVM and also I tried to use the PCA to pipe the result to SVM and DTC.

The best result I got was the gaussianNB. I noticed also that SVM is really time consuming (poorest performance) and not easily managed; that is why I ended up using GridSearchCV

---

**\* 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"]**

Algorithm parameters tuning is an essential step in machine learning as the performance and accuracy are dependent on the correct parameters chosen by the programmer. Notice that these parameters are sometimes named hyper parameter while parameters are referred to values learned by the estimator from the data.

Tuning the parameter can take many ways: random, try-and-error, and brute-force search. My algorithm gaussianNB didn't have parameter tuning however when I tried the SVM I used the gridSearchCV that takes all possible parameter and try then in a the classifier to find the best combination. Needless to say that will be time consuming.

---

**\* 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 for assessing how the results of an algorithm will generalize to an independent data set. The classic mistake that is done with cross-validation is applying the cross validation on the whole data without keeping some untouched dataset for testing; this is known as data leaking in validation.

I used the kFold cross validation dividing the training sets into 7 folds and trained my estimator on them then I tested it on the testing dataset.

---

**\* 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: 0.481      Recall: 0.365

The accuracy of a test can't be represented by a simple number. There must be a number that can represent the ability of the algorithm to detect the positive cases; and we call it the RECALL. In my algorithm this number is 0.36; i.e. the algorithm can detect 36% of the true POIs.

So if the algorithm can detect about 36% of the true POIs in the sample; and we know that it might consider some negative as positive (false positive); then comes another metric called PRECISION that detects how much is the ability to include only the true positive POI by the algorithm and that will be the percentage of the true positive / (true + false positive); which is 0.48 in my algorithm. In other words; we can say that the algorithm can get 48% true positive POIs from all the cases that was labelled as true.

The concept of recall and precision is used in multiple scientific fields sometimes with different names. In biomedical field this is called sensitivity and specificity.

RECALL represent how much of the positive cases the algorithm can detect, regardless of how many wrong positive detection ( because it might detect negative cases and consider them positive cases).

PRECISION measures how many 'true positive cases' did the algorithm find out of all cases that the algorithm considered as positive.