

Q 1 :

Enron was one of the largest companies in the United States. In 2002, it had collapsed into bankruptcy due to widespread corporate fraud.

As result of authorities' investigation, thousands of Enron email data, and detailed financial data for top executives were made public.

Combing email data, financial data and list of persons of interest (POI) who were indicted or testified in exchange for prosecution immunity; I want to use machine learning algorithms to identify POI based on the mentioned data. I want to build, explore and train different classifiers where I feed the classifiers with set of features that will be extracted from the dataset. At the end, I will test the classifiers to measure their performance and see how well they are labeling the person either POI or Not-POI according to test data.

The dataset contains:

164 data point

21 Feature

18 POI

Intuitively, if we have outliers I'd assume they will somewhere around financial data. I find "salary" and "bonus" features most compelling for outliers, I used scatter plot to visualize those features and found out we have an outlier in the dataset. Accordingly, I removed the found outlier, named **TOTAL**

Q 2 :

I've started with below features which I find most interesting features, most of them financial:

[**POI**, **salary**, **bonus**, **total_stock_value**, **expenses**, **long_term_incentive**, **director_fees**, **from_this_person_to_poi**, **shared_receipt_with_poi**]

I've created a new feature called **earned_cash** which is sum of salary and bonus.

After preparing my initial feature list, I used an automated feature selection function, SelectKBest.

I got below scores:

Feature	Score
<u>salary</u>	18.003739993113935
<u>bonus</u>	20.524645181851792
<u>earned_cash</u>	22.292899935251594
<u>total_stock_value</u>	23.898259813869416
<u>expenses</u>	5.9545442921972933
<u>long_term_incentive</u>	9.7721035384082544
director_fees	2.1453342495720547
from_this_person_to_poi	2.3388361146462624
<u>shared_receipt_with_poi</u>	8.4326354230246814

Finally, I chose top 7 features with highest score ()

Q 3 :

I've tried 2 algorithms, and used GridSearchCV to tune parameters for each algorithm.

	GaussianNB	Decision Trees
<u>parameters</u>	Default	criterion="entropy", max_depth=None, min_samples_split=40
<u>Accuracy</u>	0.74167	0.84953
<u>Precision</u>	0.29436	0.41540
<u>Recall</u>	0.67100	0.31550

I chose to processed with GaussianNB since it shows better recall, which I believe it's more important in our case of investigation than precision. **We want to be able capture more POI tolerating precision, as we able to do manually** sanity check.

Q 4 :

Turning the parameters of ML algorithms is very essential in building our model, it means trying different values for set of parameters that available for that algorithm. Picking the right parameters has very critical impact on algorithm performance and can help us to achieve the optimized performance for our model.

Failing to tune those parameters will negatively impact our algorithm performance, and will lead to overfitting.

As I mentioned before, used GridSearchCV to tune parameters for each algorithm I've used when it's possible. For example, for Decision Tree algorithm I created dictionary with 3 different parameters, each parameters with set of values and then used GridSearchCV to get the best values for those parameters :

Parameter	Values
criterion	["gini", "entropy"]
max_depth	[None, 2, 4, 6,8,10]
min_samples_split	[5, 10, 15,20,25,30,35,40,45,50]

Finally, I've got below as best parameters :

```
{'min_samples_split': 40, 'criterion': 'entropy', 'max_depth': None}
```

Q 5 :

Validation is the processing of testing our trained model against test dataset. Basically, we split our dataset into two group, training set and testing set. We first start by training our model using our training set, once we achieved the optimized performance for our classifier we test it using testing set.

One common classic mistake could happen here is using the same dataset to train and test our model. Doing so will cause overfitting which will make the model to memorize not learn to generalize as it's supposed to do.

I used cross-validation by splitting my dataset into two parts where 30% as testing set and 70% as training set.

Q 6 :

I've used two evaluation metrics to evaluate my classifier performance which are precision and recall.

My GaussianNB classifier achieved average precision of 0.29436, and average recall of 0.67100.

What we mean by precision here is how precisely our classifier predicts the correct class (POI vs. Not-POI). On the other hand, recall means how many correct predictions our classifier makes from total number of true labels.