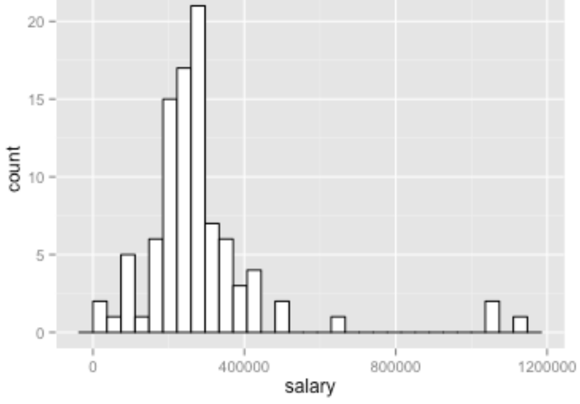
Identifying Fraud from Enron Email

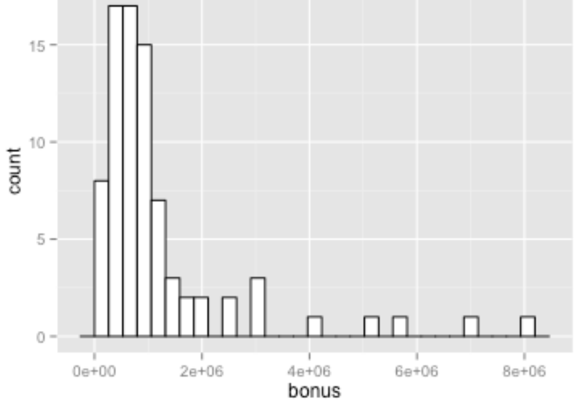
1. *Summarize for us the goal of this project and how machine learning is useful in trying to accomplish it. As part of your answer, give some background on the dataset and how it can be used to answer the project question. Were there any outliers in the data when you got it, and how did you handle those? [relevant rubric items: "data exploration", "outlier investigation"]*

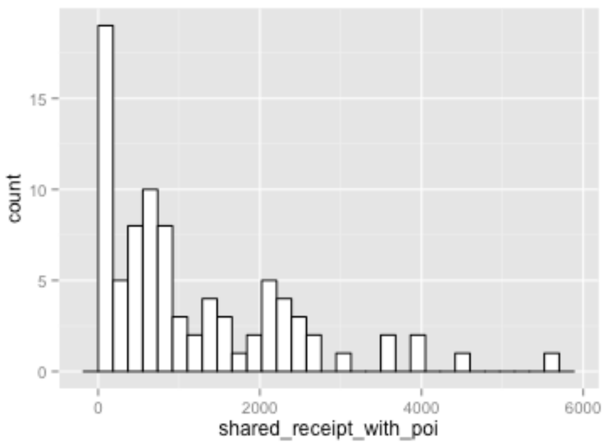
This project is to identify POI using financial and email data. This financial data includes Enron employees' compensation or expense related data (such as salary, bonus, stock options), and email data includes the number of messages from and to POI. We assume that POI have different characteristics and patterns that could affect their financial and email data. It is quite difficult to decide whether a person is POI or not by simply looking at one or two features manually, or it is hard to have logical ways to come up with rules that distinguish POI from non-POI. Therefore, machine learning is useful in finding the patterns that may be hidden in many features of financial and email data, and it can be used to predict other POIs.

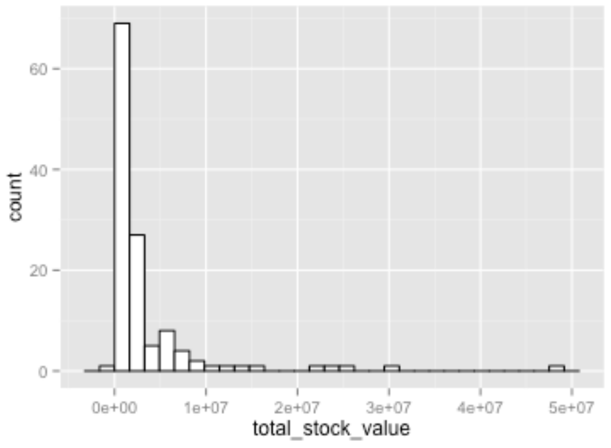
One outlier in the data set is the entry with name "TOTAL". This sample was removed from the data. There are other samples with relatively high values, but it is hard to conclude that they are outliers or valid samples, and they weren't excluded in this analysis. I plotted histograms of a few important features to show the distribution of each feature after removing "TOTAL" entry.

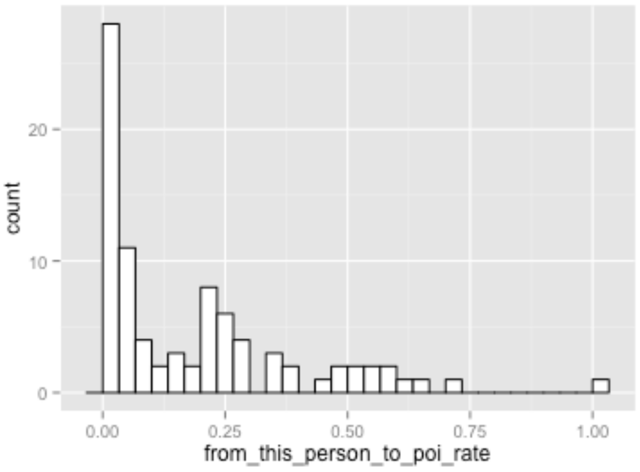
The total number of samples is 146, and after excluding the outlier, there are 145 samples in this data set. There are 21 features, and POI is the label indicating two classes (POI vs non-POI). There are 18 POI samples with POI=true, and 127 non-POI samples. The following features have too many missing values, so they were excluded from the analysis: 'deferral_payments', 'restricted_stock_deferred','loan_advances', 'director_fees', 'deferred_income'.

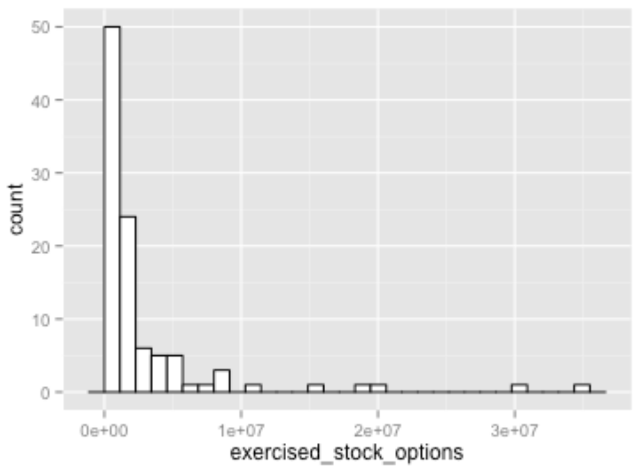










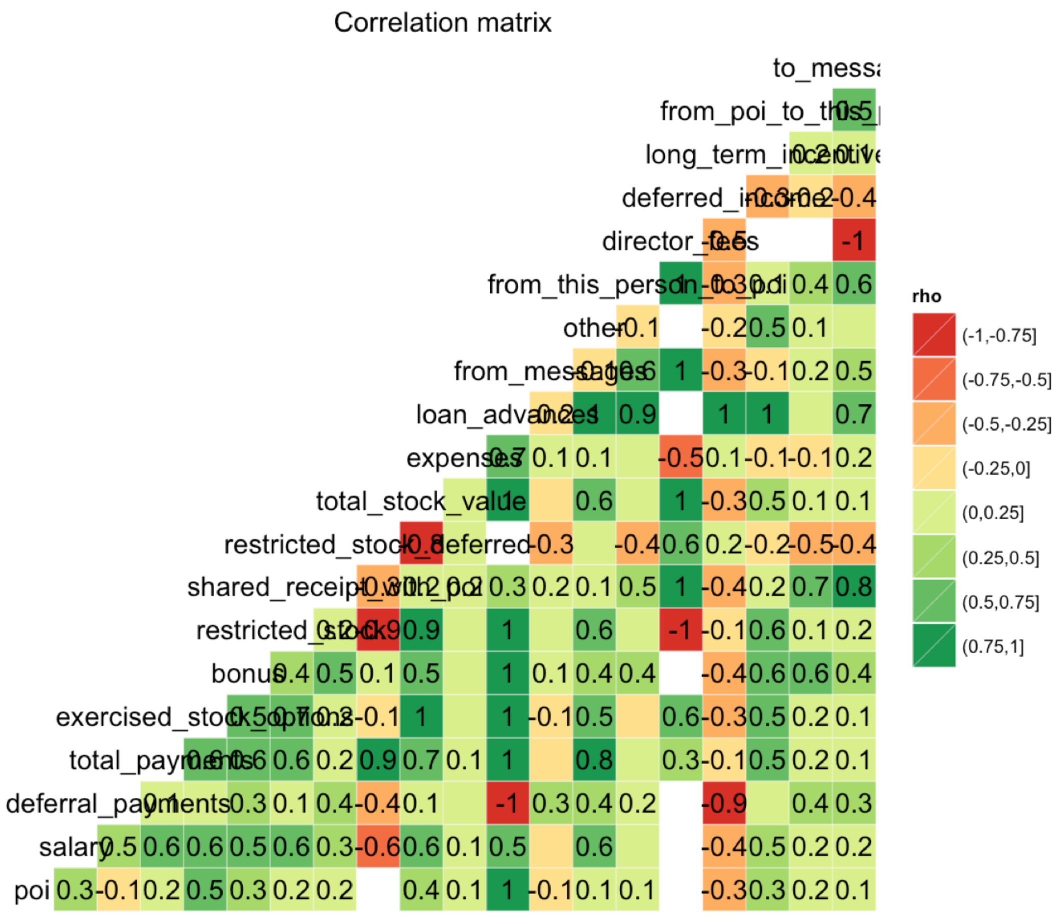


2. *What features did you end up using in your POI identifier, and what selection process did you use to pick them? Did you have to do any scaling? Why or why not? As part of the assignment, you should attempt to engineer your own feature that does not come ready-made in the dataset -- explain what feature 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 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”]*

In this analysis, I used four features -'salary', 'exercised_stock_options', 'bonus', 'shared_receipt_with_poi', 'total_stock_value', ‘from_this_person_to_poi_rate’. Initially, I tried a few different combinations to see which give the best scores. Then, I used selectKBest method to find the best features. Here is the score of selected features by selectKBest method.

bonus = 30.6522823057
salary = 15.8060900874
from_this_person_to_poi_rate = 13.7914132368
total_stock_value = 10.814634863
shared_receipt_with_poi = 10.6697373596
exercised_stock_options = 9.95616758208

To decide the number of features, I compared the score from test_classifier and decide the number of features. I also looked at the correlation matrix between poi and other features. The following figure shows the correlation matrix of all the features. The features that are highly correlated (positive or negatively) are the features that can be used in the algorithms. From the plots, salary, exercised_stock_option, total_stock_value, bonus, long_term_incentive have high correlation with poi. Note that "loan_advances" have a correlation with 1, but it is because of a very small number of samples that have valid "loan_advances" value. This helps me understand which features will be useful in the algorithms even though I eventually used selectKBest method to find four top features.



It was not necessary to do scaling for decision tree algorithm because thresholds for decision are decided accordingly.

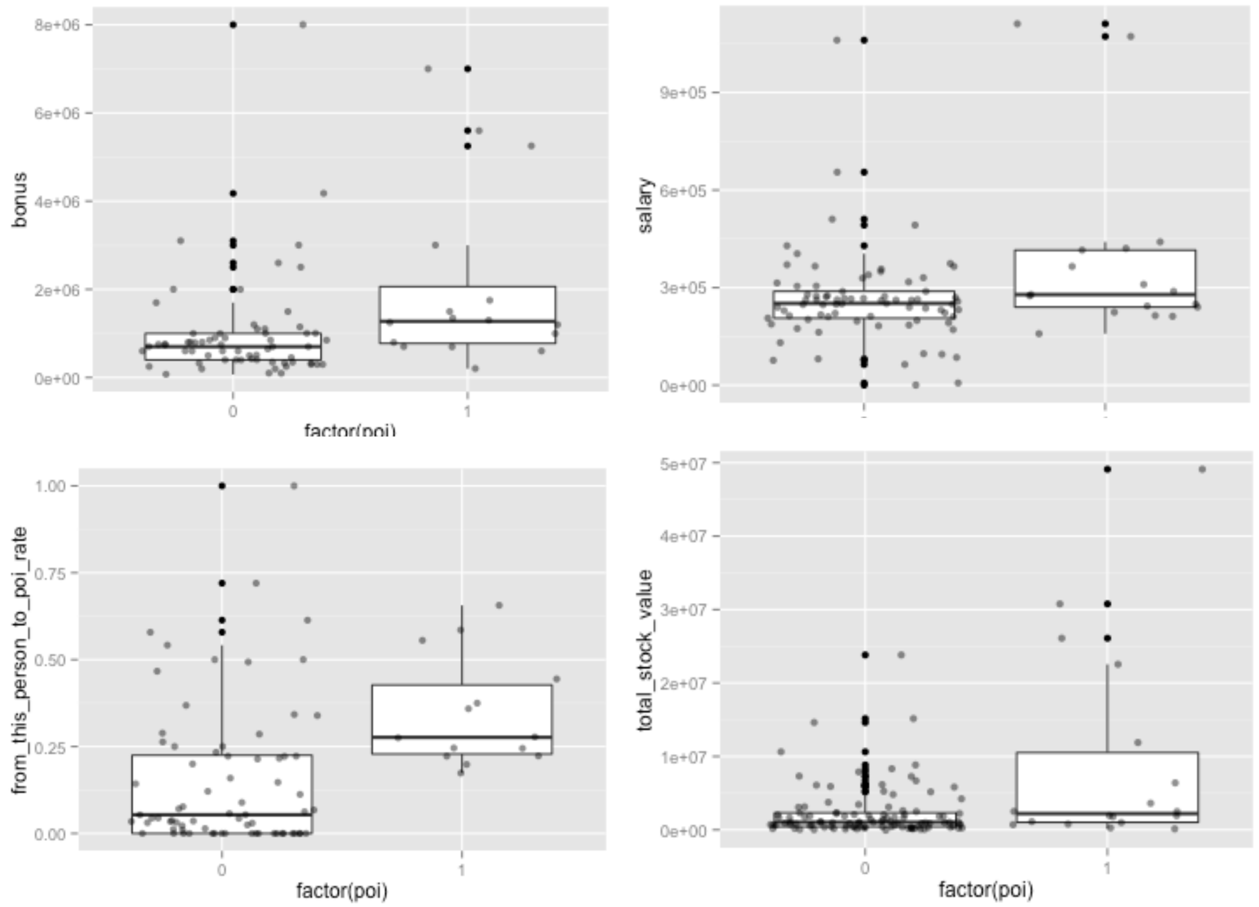
I added two additional features related to email communication. The original data include the absolute number of messages from/to poi, but it is more important to look at the rate of messages from/to poi by dividing them by the total number of messages. from_this_person_to_poi_rate was used in the final algorithm.

Decision tree algorithm also returns the feature importance. Here is the list of feature importance.

```
['salary', 'exercised_stock_options', 'bonus', 'shared_receipt_with_poi',
'total_stock_value', 'from_this_person_to_poi_rate']
[ 0.15437946  0.08492976  0.36543795  0.30923394  0.          0.08601888]
```

Please note that the importances of the features change at every run because of random state, and they were obtained with one fixed training set. In most cases, bonus, shared_receipt_with_poi, and salary have high importance.

The following box plots show the distribution of key features for each class (poi). Even though the number of samples with poi=1 is small, these plots show the different patterns of key financial and email data of each class.



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"]*

I selected Decision tree algorithm. I also tried random forest, but found that decision tree algorithm performs better in terms of F1 score for this data. Performance was measured with 1000-fold cross validation.

Decision tree algorithm
Accuracy: 0.82150   Precision: 0.38079   Recall: 0.39850   F1: 0.38945   F2: 0.39483

Random forest (min_samples_split = 1, n_estimators = 5)
Accuracy: 0.84214   Precision: 0.42268   Recall: 0.28700   F1: 0.34187   F2: 0.30669

Random forest (min_samples_split = 1, n_estimators = 1)
Accuracy: 0.81129   Precision: 0.33351   Recall: 0.32150   F1: 0.32739   F2: 0.32383

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? (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"]*

Classification algorithms have various input parameters that allow us to control the trade-off between bias and variance, or over-fitting/under-fitting. Classification algorithm will minimize the objective function for the given parameters, so if we don't tune the parameters well, we will end up having unexpected results. In this analysis, I used grid search to find the best parameters for the algorithms and select the parameters that give the highest score. In case of decision tree classification, I changed min_samples_split from 1 to 10 and selected the one give the best score. For the random forest, I tried to tune min_samples_split and n_estimators.

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 item: "validation strategy"]*

Training of machine learning algorithm is to minimize the objective function (such as errors) for the given training set. Because it is possible to minimize the error by overfitting, it is important to validate the algorithm with test data set which weren't seen by the algorithm during training.
In this analysis, I used 1000-fold cross validations using StratifiedShuffleSplit where the data was splitter into a training and test set. Stratification divides the samples into

homogenous subgroups so that the percentage of samples for each class is preserved. One of the classic mistakes is to use the training data set during the validation phase, which leads to incorrect conclusion of the performance of the algorithms.

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"]*

I used recall, precision and f1 score. As described above, I selected decision tree algorithm, and here is the performance score from 1000-folds cross validations. 0.38 precision means that when an algorithm predicts a person as POI, it is correct for 38% of samples. 0.4 recall means that among all the POIs, the algorithm predicts them as POI correctly for 40% of samples. For 60% of POI, the algorithm predicts as non-POI. f1 score considers both precision and recall because it is the harmonic mean of precision and recall.

Decision tree algorithm

```
Accuracy: 0.82150	Precision: 0.38079	Recall: 0.39850	F1: 0.38945	F2: 0.39483
```