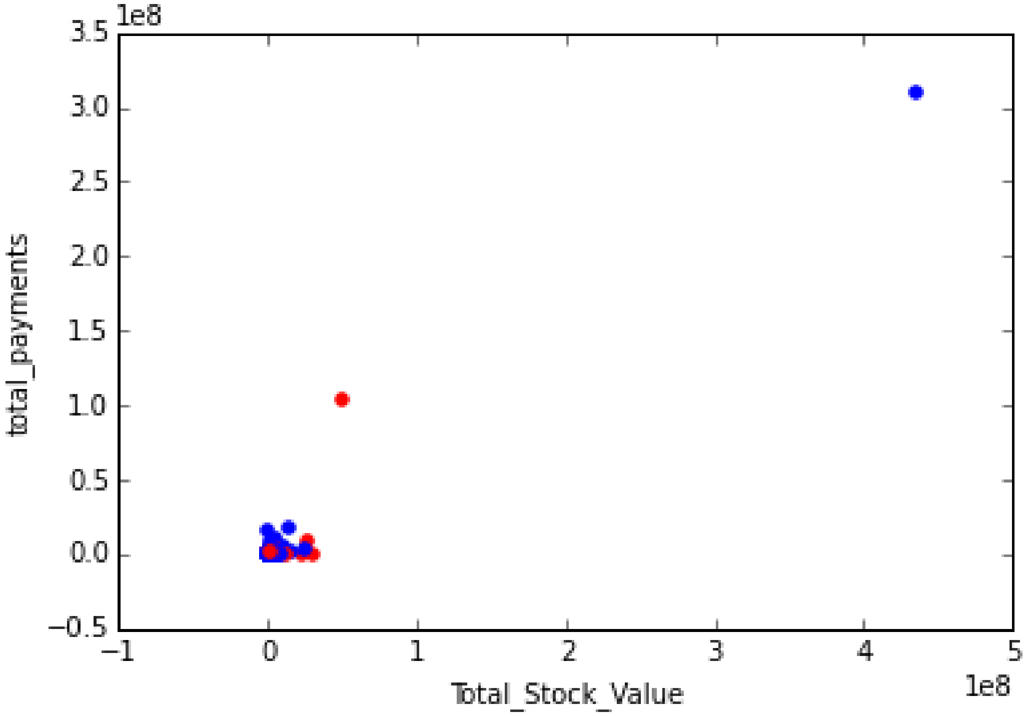
Enron POI Identifier Report

Introduction

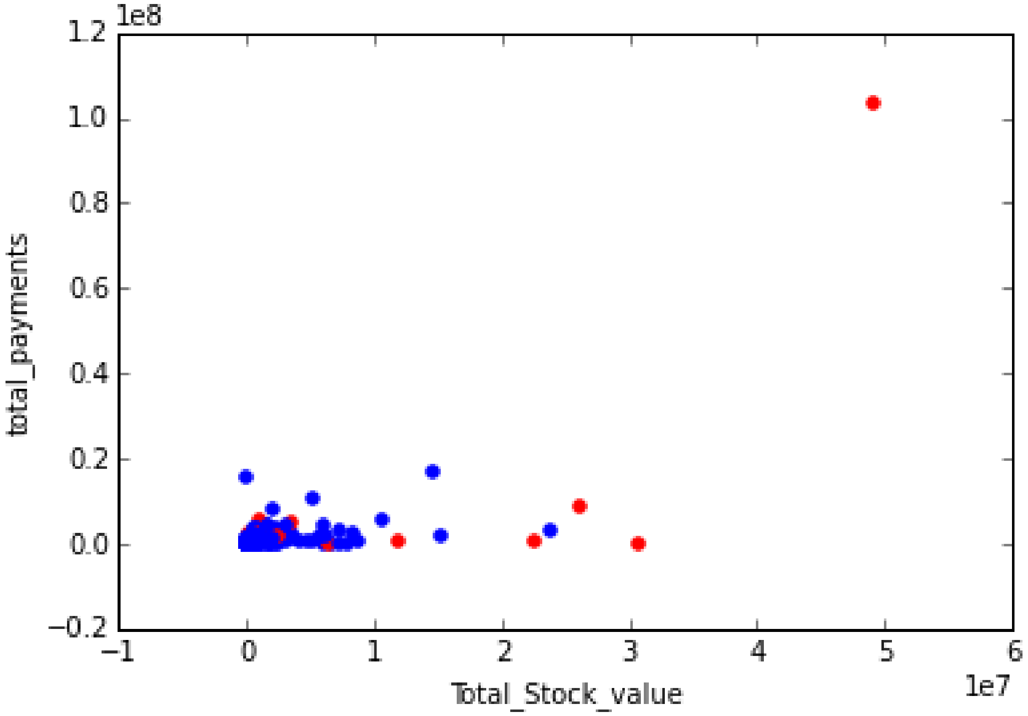
Enron is one of the largest corporate fraud cases in American history and also is one of the well-documented ones. It has a large corpus of corporate e-mail database available for the public for study and research purpose. In this project, the e-mail data and the financial data has been used to identify the Person of Interest (POI). POI is someone who is indicted for fraud, settled with the government or testified in exchange for immunity. This report documents the machine learning techniques used in building the POI identifier

The Enron Data

The Enron Dataset has 146 records. Of these 18 are labeled as POI and the rest are labeled as non-POI. There are 21 features for each Person. Some of the fields like loan_advances, director_fees and restricted_stock_deferred are sparsely populated. loan_advances field has values only for 4 records, director_fees has values only for 17 records and restricted_stock_deferred has values only for 18 records.



On examining the data points plotted for total_stock_value and total_payments, one outlier is distinct. This is the “Total” and has been removed from the data set. The other data point which stood out while examining the PDF for financial data was “THE TRAVEL AGENCY IN THE PARK”. This data point has only data for “Other” and “total_payments” fields. The rest of the data is not available for it. Also, it appears to be an agency and not one of the Enron Executives. After removing these 2 data points we get the graph as shown below



There is still a very distinct outlier standing out for the data point of Ken Lay. The total payment for Ken Lay is around 103 million and the Total Stock Value is around 49 million which stands out as compared to rest of the data points. There are a few more outliers like Joseph Hirko with over 30 million of total stock value, Jeff Skilling with over 26 million of total stock value. But, we would like to retain these data points as they are Enron executives and the fact that so much money was taken out of the company by them and they are identified as the POIs, we want to retain them in the dataset

Feature Processing

After Outlier removal, the next step was to assess if any new feature was to be created from the given features. We created the following 3 new features

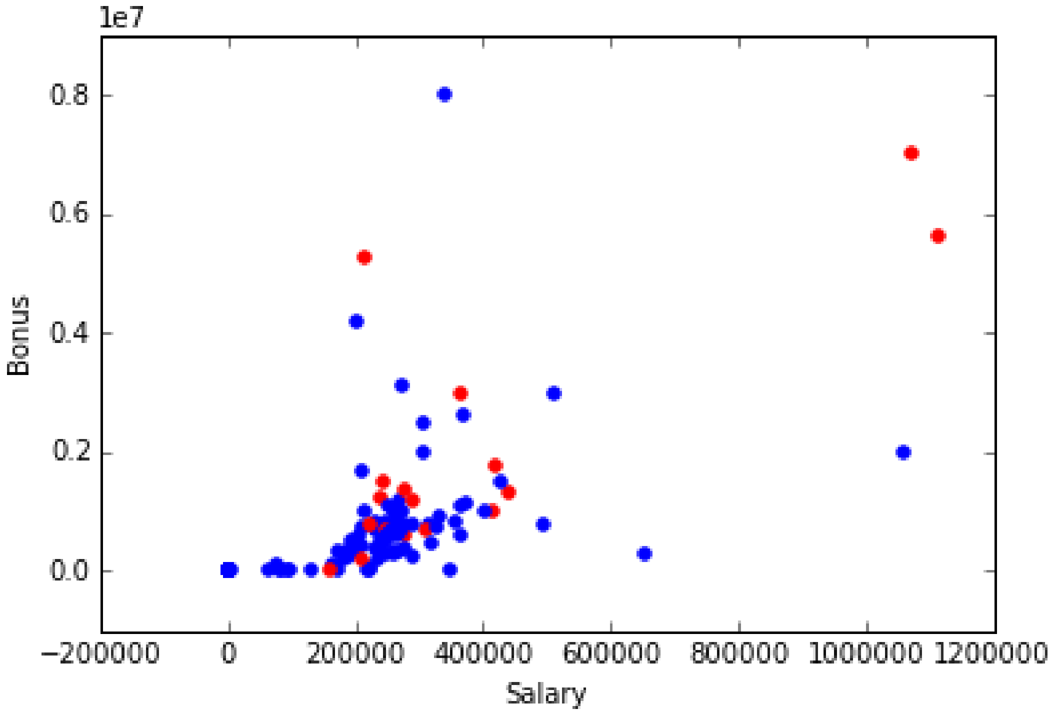
1. **fraction_shared_receipt_with_poi:** Created as a fraction of **shared_receipt_with_poi** over **to_messages**
2. **fraction_from_poi:** Created as a fraction of **from_poi_to_this_person** over **to_messages**
3. **fraction_to_poi:** Created as a fraction of **from_this_person_to_poi** over **from_messages**

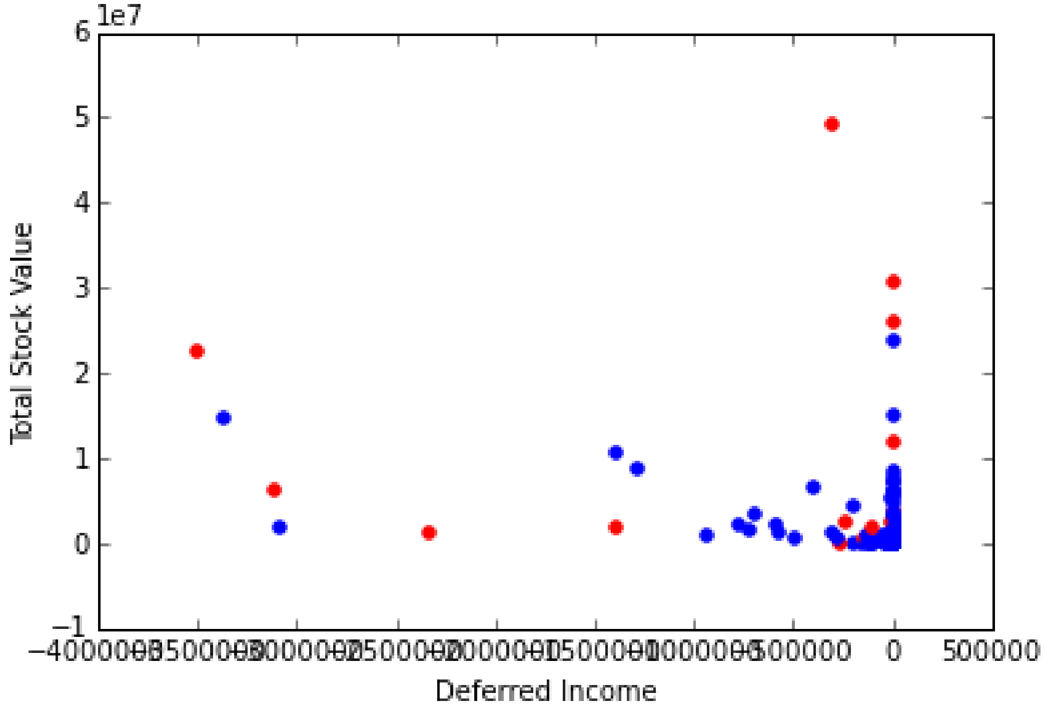
The premise for creating these features was that POIs among themselves would be having a strong e-mail connection among each other and the fraction will give the percentage of the total e-mails received / sent /shared between the person and the POI and if this fraction is high then it might imply that the person has strong connection with the POI

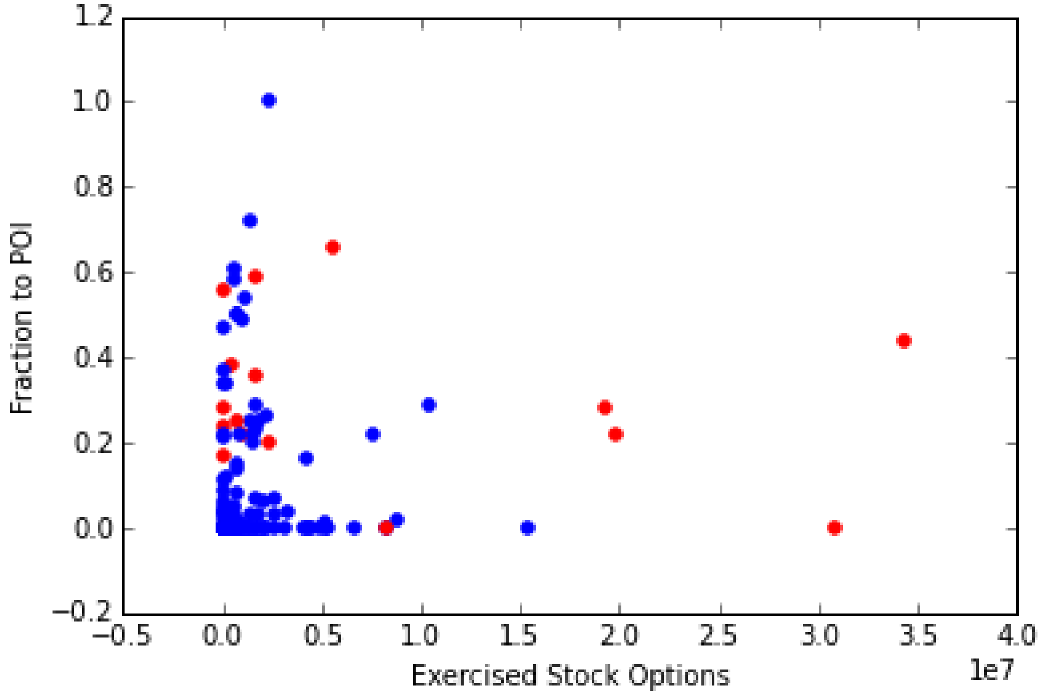
Once the data was cleaned of outliers, the next step was selecting the best features to use in the classifier. This was an iterative process, where we used the “SelectKBest” algorithm from the sklearn package. We first started with 9 features and started decreasing the count and found that for 6 features we had the best results for the Gaussian Naïve Bayes classifier. The 6 best features identified by the K-Best algorithm are as follows:

salary
bonus
deferred_income
total_stock_value
exercised_stock_options
fraction_to_poi

We then plotted the visualizations for these 6 features identified by the K-Best algorithm as having some discriminatory power.







From the visualizations all of these features seemed to have discriminatory power. But, found that the feature deferred_income has value for only 49 records. I thought that the **fraction_shared_receipt_with_poi** and / or **fraction_from_poi** should give better results than deferred_income. Replaced the deffered_income with a combination of these 2 fields and then individually and found that by using fraction_shared_receipt_with_poi I am getting better results.

Hence, the deferred_income feature was replaced by fraction_shared_receipt_with_poi and the final list of features retained are as below:
salary
bonus
total_stock_value
exercised_stock_options
fraction_to_poi
fraction_shared_receipt_with_poi

These features were then used in the next steps

Algorithm Selection and Tuning

Started with the Naïve Bayes algorithm. For the features selected this Algorithm had the following values:
Precision: 0.44915
Recall: 0.29150

Then, I used the support vector machine, SVC classifier from sklearn algorithm. For using this classifier, I had to scale the features. I used the MinMaxScaler to scale the features and passed the scaled features to the SVC algorithm. I used Pipelining to achieve this. MinMaxScaler was used as the first stage of the pipeline and SVC algorithm was used as the second stage.
Tried the following combinations:

Kernel	Gamma	C	Precision	Recall
rbf	1.0	10	0.691	0.056
rbf	1.0	100	0.38782	0.0605
rbf	1.0	1000	0.33106	0.097
rbf	1.0	10000	0.25505	0.139
rbf	2.0	10	0.4187	0.0515
rbf	2.0	100	0.35859	0.071
rbf	2.0	1000	0.30268	0.1185
rbf	5.0	10	0.33861	0.0535
rbf	5.0	100	0.32268	0.106
rbf	5.0	1000	0.20167	0.157
linear	1.0	10	0.47143	0.099
linear	1.0	100	0.52224	0.135
linear	1.0	1000	0.54647	0.147
linear	2.0	10	0.47143	0.099
linear	2.0	100	0.52224	0.135
linear	2.0	1000	0.5464	0.147
linear	5.0	10	0.47143	0.099
linear	5.0	100	0.52224	0.135

linear	5.0	1000	0.54647	0.147

I am getting good precision scores but very poor recall scores

Then, the decision tree classifier was selected as the algorithm for the POI identification and min_samples_split parameter was tuned. Since, for cross-validation we are using StratifiedSuffleSplit and the results for the Decision Tree algorithm varies slightly for each run. I took average of 5 readings for each min_samples_split. The Precision and Recall values for different min_samples_split parameter are as below:

min_samples_split	**Precision**	**Recall**
2	0.30362	0.356
	0.31523	0.3685
	0.31048	0.3645
	0.30866	0.369
	0.31811	0.383
Average for 2	**0.31122**	**0.3682**
3	0.30643	0.317
	0.30313	0.3145
	0.29324	0.306
	0.30176	0.317
	0.30144	0.3135
Average for 3	**0.3012**	**0.3136**
4	0.28743	0.288
	0.28734	0.2895
	0.27924	0.2805
	0.27417	0.2765
	0.2865	0.2865
Average for 4	**0.282936**	**0.2842**

Since the best average precision and recall was found when min_samples_split=2 for the Decision tree classifier, this algorithm and parameter has clearly emerged as the best case POI identifier classifier. The Naïve Bayes algorithm and SVC has a high Precision but lower Recall than the Decision Tree classifier. We want high recall as compared to the high precision because high recall signifies identifying a true POI as POI with a higher probability. Precision lower than Naïve Bayes / SVC means erring and misclassifying more number of Non-POIs as POI. We would want to trade off the erring towards misclassifying more Non-POIs as POIs rather than not identifying a True POI as a POI. This is because once we have the POI flagged then in the further steps of Investigation we can drop off a Non-POI identified as a POI from further investigations if evidences states so.

From above we see that parameter tuning is very important. For the same dataset, and same algorithm the results of the classifier varies for the different parameters selected.

Hence we need to tune our parameters well so that our classifier gives the best possible metrics for the features which we believe explains the trend in our data

Since the Dataset is imbalanced we won't be able to use GridCV for parameter tuning. Hence manual approach has been used for parameter tuning

Validation and Performance

Validation is a means to measure the performance of the classifier on the features selected for the dataset. The classifier is trained using a set of data points (training set) and is tested on a different set of data points (testing set). We perform predictions on this testing set and compare it to the labels to check for the performance metrics of the classifier. It also serves as a check on over-fitting

The Enron dataset has 146 data points. Of these only 18 are POI and the rest are non-POI. The distribution of the 2 classes is imbalanced with almost 88% being Non-POI and only 12% are POI. Hence, if we employ simple K-fold method to split the data points into training and testing sets the ratio of POI to Non-POI in each of the folds might be very different. This is because it will depend on the order of the data points and how they get distributed in each of the folds. Hence, the metrics for the predictions which we get for our classifier might vary significantly for each of the folds. In order to overcome this problem we have used StratifiedShuffleSplit function to split the dataset into training and testing sets. We have set the folds to 1000 which denotes the number of re-shuffling and splitting iterations and random_state to 42 used for random sampling. This function provides the indices to split data into training and testing sets and we use it to train our classifier and perform the testing

Evaluation Metrics

The distribution of the POIs and Non-POIs in the Enron dataset is imbalanced. Hence, if we use accuracy as a metric to measure the performance of the classifier then even if the classifier classifies all of the data points as Non-POI still the accuracy will be around 88% since around 88% of the data are of Non-POI. Hence, we can't rely on accuracy metric to measure the performance of our classifier.

The metrics Precision and Recall turns out to be good metrics for such imbalanced datasets. The Precision determines the likelihood if a person is identified as POI by the classifier he truly is POI. Given that a person is POI, Recall measures the likelihood that our classifier flags it as POI

Hence Precision and Recall have been used as the metrics to measure the performance of the POI identifier classifier. The average Precision and Recall scores for our Decision Tree Classifier is 0.31and 0.37 respectively

Discussion and Conclusions

The precision score of 0.31 indicates that if our classifier identifies a person as POI, the likelihood that the person is truly POI is 31%

The recall score of 0.37 indicates that given that a person is POI, the likelihood that our classifier will flag the person as POI 37%

Given the fact that only 18 out of the 146 data points are POI, these scores are good. But, there seems to be further scope for improvement in the scores. The e-mail data in the Enron starter dataset were financial data and the aggregated count of the e-mail messages. We didn’t use the actual e-mail text data in our analysis. By exploring the actual e-mails we could extract Text features. There is a possibility that the text features might provide us with some more patterns in the data which might help in classifying the POIs better