# Udacity Machine Learning Course:

# Enron Person of Interest Identifier Project Report

Cem Alper

**Data Exploration**

The original dataset includes a dictionary of 20 features with 140 data points. Among these, 18 are persons of interest (POI). The features ‘deferral payments’ (100 missing entries), ‘director\_fees’ (123 missing entries) and ‘restricted\_stock\_deferred’ (123 missing entries) have a large percentage of missing points. Therefore, they are primary feature candidates to be removed though we have utilized principal component analysis where we found the number of features (*n\_components*) by cross-validated Grid Search to perform the removal automatically.

**Outlier Investigation**

For outlier detection, I have utilized a visual approach since the number of data points are small, where I plotted each of the features w. r. t. poi labels in a scatter plot. Two data points with dictionary keys ‘TOTAL’ and ‘THE TRAVEL AGENCY IN THE PARK’ are removed right away, due to them not corresponding to a real person. For the remaining outliers, see the plots and the table below for removed data points (for the plots below blue points are the non-POI, and red points the POI):

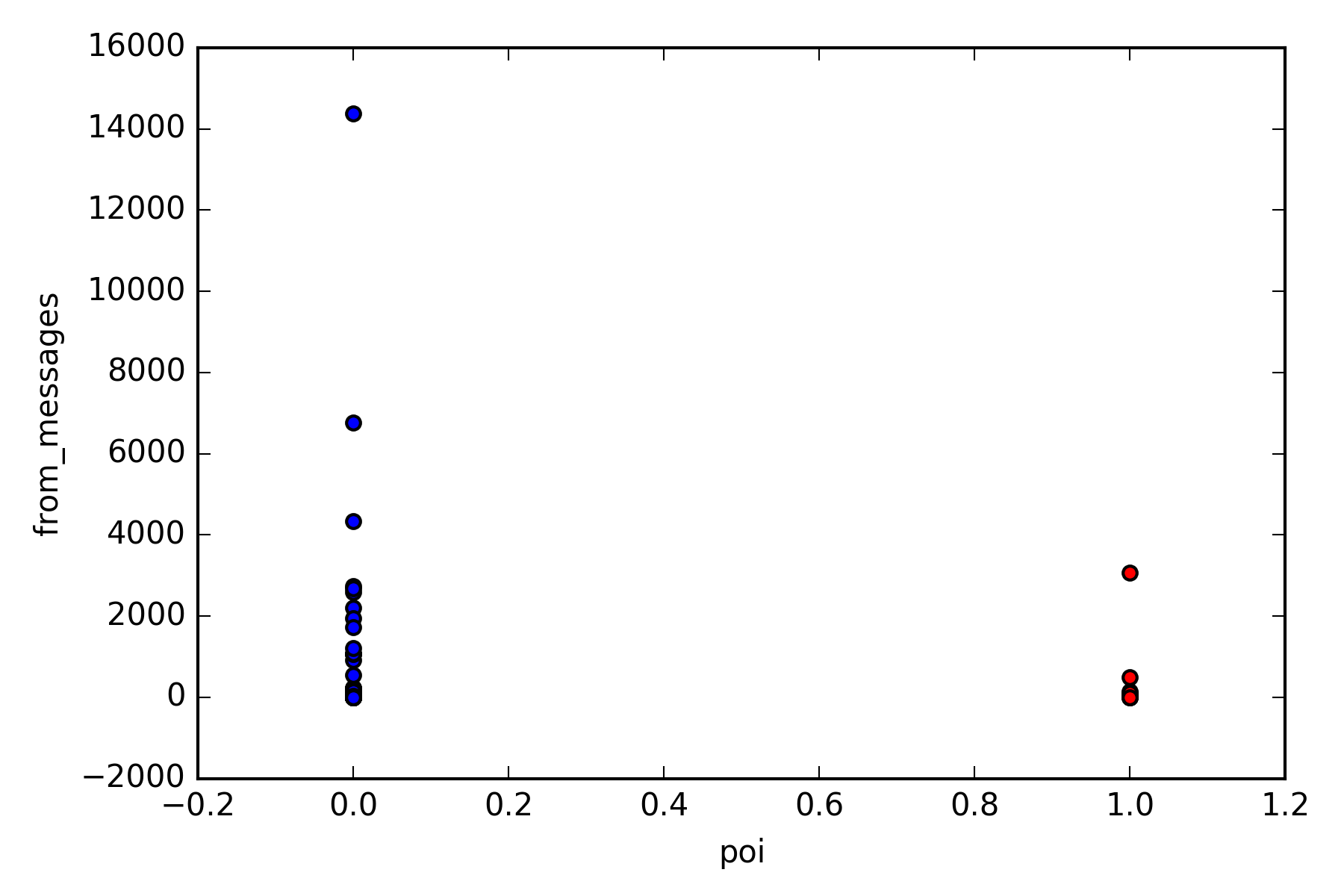
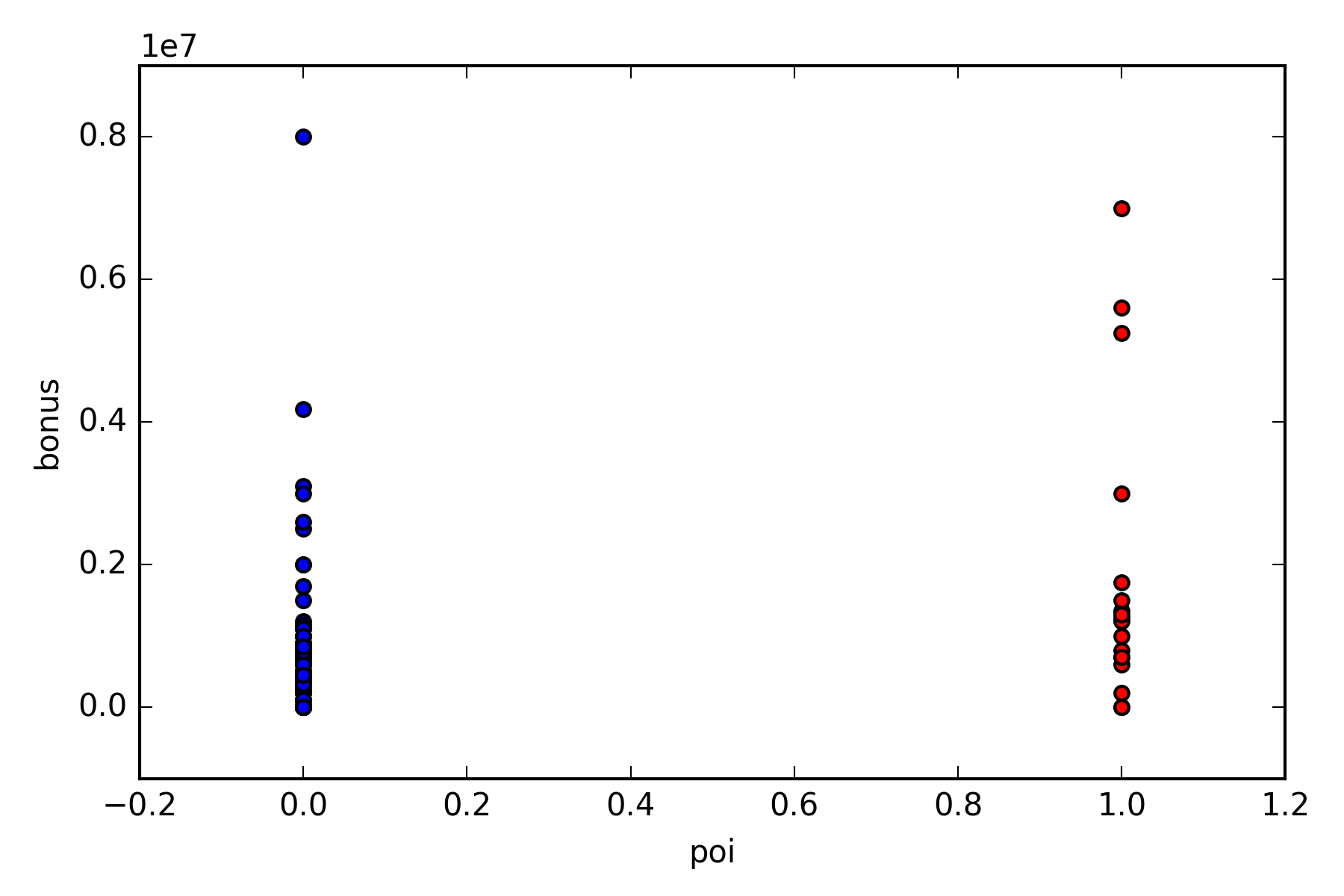


Figure - Visual identification of the outliers. For both cases I have removed the largest non-POI data points.

The removed outlier points are given below in the table:

Table - The dictionary keys of the removed data points.

|  |
| --- |
| KEY |
| TOTAL |
| THE TRAVEL AGENCY IN THE PARK |
| LOWRY CHARLES P |
| KAMINSKI WINCENTY J |
| BHATNAGAR SANJAY |
| LAVORATO JOHN J |

To assess the improvement of our model after performing the outlier removal, we train the following pipeline of models:

**Feature Creation**

I have created two new features from the already existing ones. The first one is called ‘from\_to\_ratio’ which indicates the ratio between the number of mails received from the POI and the number of mails sent to the POI. The second one is called ‘poi\_ratio’ which indicates the ratio of the mails sent or received to/from the POI over the total number of mails sent and received.

The rationale for creating the ‘from\_to\_ratio’ is to deduce whether the nature of communication changes depending the POI status of the person (i.e., whether the e-mail conversation is predominantly one-sided when it is between a POI and non-POI and perhaps more two-sided between POIs.). ‘poi\_ratio’ tries to assess whether POIs are more likely to communicate with each other than non-POIs.

**Feature Creation**

For feature selection, univariate feature selection (*SelectKBest*) is used with ANOVA f-score as the scoring function. ANOVA f-score compares each feature and assigns them an f-value which is the sum of difference of means for each group (i.e., POI and non-POI) for each feature. The features that exhibit the highest difference of means have high scores. Here is the f-scores for each feature given in the dataset + newly created ones:

Table - ANOVA f-scores for all the considered features.

|  |  |  |
| --- | --- | --- |
| Ranking | Feature Name | ANOVA f-score |
| 1 | bonus | 24.2088610537 |
| 2 | exercised\_stock\_options | 16.1940646191 |
| 3 | total\_stock\_value | 14.3331324085 |
| 4 | shared\_receipt\_with\_poi | 10.3092648697 |
| 5 | deferred\_income | 9.53916232556 |
| 6 | long\_term\_incentive | 9.47670139255 |
| 7 | salary | 8.20829503011 |
| 8 | from\_poi\_to\_this\_person | 7.76917226481 |
| 9 | total\_payments | 6.58614692476 |
| 10 | poi\_ratio | 6.54084329234 |
| 11 | loan\_advances | 5.67102200261 |
| 12 | expenses | 5.45144951395 |
| 13 | restricted\_stock | 3.72650827556 |
| 14 | other | 3.58020144479 |
| 15 | from\_this\_person\_to\_poi | 3.04124489958 |
| 16 | director\_fees | 1.36152820243 |
| 17 | restricted\_stock\_deferred | 1.31089941269 |
| 18 | from\_to\_ratio | 0.0772364044866 |
| 19 | deferral\_payments | 0.0711425783689 |

Even though the feature ‘poi ratio’ seems to rank high in this score table, we also note the possibility of the newly created features being highly correlated with the ones that are used to create them. So, use principal component analysis (PCA) could be beneficial which will detect such correlations (i.e., repeated information) and allow us to remove them by reducing the dimensionality thus saving computation time and possibly increasing accuracy.

**Feature Scaling**

Since we will use PCA, we will have to scale our features such that they all have scaled standard deviations. Otherwise, the scale differences between features will misguide the algorithm such that it will choose the ones with higher scales. In our dataset, this might correspond to financial features which are on the order of , compared to e-mail features which are on much smaller orders of magnitude.

**Picking an Algorithm**

I have considered three different algorithms: Naïve Bayes with Gaussian distribution (*GaussianNB*), Support Vector Machines (*SVC*) and Decision Trees (*DecisionTreeClassifier*). Since we are using scaling and PCA in the previous steps, I make use of Pipelines for easier parameter investigation. As for why I did not pick K-Means clustering algorithm, I note the inherent imbalance in different class types (i.e., we have 18 POIs vs. 122 non-POI) in which case K-Means is not ideal. For the same reason, I have utilized the ‘class\_weight’ option ‘balanced’ which makes the algorithm assign a higher weight to the less frequent class types (POIs in our case) in SVC and Decision Tree classifier.

For each case we have optimized the possible parameters of the algorithm using cross-validated grid search (*GridSearchCV*) using ‘f1’ function as the scoring function again due to the fact that the imbalance in the dataset. we list the parameters that we optimized for each algorithm in the table below:

Table - Optimized Parameters for each algorithm in trial.

|  |  |  |
| --- | --- | --- |
| SelectKBest + PCA + Naïve Bayes | SelectKBest + PCA + SVC | SelectKBest + PCA + Decision Tree |
| k | k | k |
| n\_components | kernel | min\_samples\_split |
|  | C | min\_samples\_leaf |
|  | gamma | n\_components |
|  | n\_components |  |

I list below the best estimators found by grid-search for each case with their precision and recall metric.

Table - The performance of optimized algorithms.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Algorithm | Accuracy | Precision | Recall | F1 |
| PCA + Naïve Bayes | 0.82414 | 0.382322 | 0.37900 | 0.38110 |
| PCA + SVC | 0.84221 | 0.42728 | 0.30700 | 0.35729 |
| PCA + Decision Tree | 0.79443 | 0.33546 | 0.44750 | 0.38346 |