Machine Learning Project

|  |
| --- |
| **DATA EXPLORATION** |
| 146 Total No. of Data Points |
| 18/144 Allocation Across Class (POI/non-POI) |
| 14.38% NaN for Total Payments |
| 35 Total Gathered POIs |
| 0 of 18 POIs with NaN for Total Payments |
| 95 Qualified Salary |
| 21 Number of Features |
| 111 Known Email Address |

**Identify Fraud from Enron Email**

In 2002, the Harvard’s Ig Nobel Prize for the ’Most Creative Use of Imaginary Numbers’ went to a company that two years prior had received its sixth, ‘America’s Most Innovative Company’ award from Fortune magazine[[1]](#footnote-1). In 2000, Enron was the seventh-largest company in America with over $100 Billion in revenue.[[2]](#footnote-2) The next year as the result of fraudulent accounting practices, they were bankrupt.[[3]](#footnote-3) During the Federal Energy Regulatory Commission’s investigation, internal company emails were released to the general public.[[4]](#footnote-4) Combining this dataset with the Securities and Exchange Commission financial data, along with Persons of Interest derived

We will answer the question from newspaper, ‘**Can we identify Persons Of Interest (POI) solely from this dataset?**’

**Features with Many Missing Values**

Loan Advances

Director Fees

Restricted Stock Deferred

# OUTLIER INVESTIGATION

|  |
| --- |
| The financial dataset are missing some POIs. |

Machine learning algorithms were performed to make some predictions and calculate its accuracy. The worst case of outliers uncovered was due to a TOTAL in the financial statement which was read in as a line item. It was removed by hand. Four other outliers were considered to be valid data points so they were left in.

A technique to remove outliers was to discard 10% of the points with the largest errors which was introduced and used to help identify additional potential outliers, however it was determined that these points were valid. A careful reading of the field names revealed, ‘THE TRAVEL AGENCY IN THE PARK’ which, not being an individual, was removed.

# OPTIMIZE FEATURE SELECTION

Supervised Machine Learning algorithms needs re-training at periodic intervals. The Netflix prize which was

“Best” algorithm was presented with $1 Million, though it wasn’t used in production because of the engineering effort required for its minimal gains.[[5]](#footnote-5) The goal in this case was to accomplish recall and precision values of 0.3+. In an effort to simulate a real-world scenario and utilize as few resources as possible, minimum features were selected.

Exercised Stock Options, Total Stock Value and Bonus were chosen in combination with sklearn’s

|  |
| --- |
| **Sample SelectPercentage Results**  Percentage: 15  Precision: 0.36  Recall: 0.38  TheFeatures:   1. exercised\_stock 2. total\_stock 3. bonus |

Univariate feature selection **SelectPercentile,** which selects features based on the highest score percentage.

|  |
| --- |
| ***sample impact of new features using final algorithm (Decision Tree)***  **Feature Precision Recall** |
| neither feature 0.37 0.38 |
| milk 0.38 0.40 |
| fraction of deferred 0.40 0.39 income to total payments |
| combined features 0.40 0.41 |

# New Features fraction of deferred income to total payments

This feature assumes that non-POIs believed that the company would keep growing and be in a better place to make those payments. POIs knew that the foundations were falling down and should be paid before the company ran out of funds.

**The value “milk” was scaled using the following ratio.**

(Expenses + Deferral Payments)

—————————————————————————

1 + (Loan Advances + Long Term Incentive + Deferred Income)

They believed that POIs have an incentive to get as much out of the company as quickly as possible since they don’t see a long-term future in the company, drove the decision to create a new label called“milk”. Expenses include consulting, reimbursements from the company and deferral payments are distributions from deferred compensation. The company must pay these now whereas loan advances, long term incentives and deferred income are future payments. If employees believe that the company won’t have the money to pay them in the future, they will want to collect what they can now.

|  |
| --- |
| **Feature Importance**  **with sample algorithm performance**   * **.30** Exercised Stock Options * **.27** Bonus * **.19** Total Stock Value * **.14** Fraction of   Deferred Income to  Total Payments   * **.11** Milk |

# FEATURE IMPORTANCE

As seen in the table to the left, each feature adds to the final result. They are sorted by their values respectively. Though these values changes with each iteration, you can count on Exercised Stock Options always provide more weight to the final algorithm than that of our created features.

# PICK AN ALGORITHM

The goal is to classify individuals as either a POI or non-POI. Further to this, it’s not a “Big Data”

Issue as there are less than 100,000 samples, so there was no need to apply an SGD Classifier or kernel approximation. This required testing of the following algorithms:

|  |  |
| --- | --- |
| 1. Naive Bayes (GaussianNB) 2. Support Vector Machines 3. LinearSVC | 1. Decision Trees |

|  |  |  |
| --- | --- | --- |
| **Algorithm** | **Precision** | **Recall** |
| GaussianNB | 0.15 | 0.75 |
| SVC | 0.71 | 0.00 |
| LinearSVC | 0.20 | 0.30 |
| Decisiontree | 0.33 | 0.34 |

**A .GaussianNB**: was simple to implement but even with excellent recall numbers, no amount of tuning afforded respectable precision results.

**B .Support Vector Machines:** gave us great precision values and only 0.0025 for recall.

**C. LinearSVC**: returned both precision and recall numbers that looked more even, but didn’t reach the desired results.

**D**.**SVM algorithms**: are not scale invariant and therefore required scaling as they deal with Euclidian distances

**E. Decision Trees:** gave back very fast results and the defaults were very close to the desired precision and recall targets.

# TUNE AN ALGORITHM

Algorithm tuning is vital to improving performance as the attributes for a given method help to refine how that algorithm is processed. After tuning the algorithms above, the DecisionTree was chosen and tuned using the following parameters:

**GridCV Trials of Tuning Settings on DecisionTreeClassifier**

*several parameters were tuned and included the following…*

**criterion** gini, entropy **splitter** best, random **max\_features** 1, 2, 3, 5, 9, 0.1, 0.2, 0.25, 0.5, 0.75, 0.8, 0.9, 0.99, auto, sqrt, log2, None **max\_features** 0.1, 0.2, 0.25, 0.5, 0.75, 0.8, 0.9, 0.95 **class\_weight** None, auto

**max\_leaf\_nodes** None, 2, 3, 4, 5, 6, 7, 8, 9, 10

After starting with grid search (GridCV) the parameters were manually tuned for the decision tree. Interestingly, adding *max\_leaf\_nodes* as a parameter lead to worse results**.** In this case, the default parameters gave the best results, with the exception of changing the **splitter** attribute from *best* to *random*. The best algorithm-tune combination from these results were selected for the final analysis.

# CLASSIFIER

A decision tree classifier was used in the final product. This decision depends on the balance between precision and recall values. For instance, though the **SVC** algorithm returned great precision results their recall values were terrible, on the other hand **GaussianNB** returned great recall but limited precision values**. LinearSVC** was tested and removed because the overall numbers were too low.

# EVALUATION MATRICS

Recall that the goal was to achieve higher than 0.3 for both precision and recall. In this case the average precision value was 0.39. which translates to mean that 39% of the time we don’t label a POI as a Non-POI. Recall looks at the other side of the equation and in plain English means that 40% of the time we correctly classify POIs.

|  |
| --- |
| **Precision and Recall Values** |
| **Precision Recall** |
| **mean 0.39 0.40** |
| median 0.39 0.40 |
| maximum 0.42 0.42 |
| minimum 0.38 0.38 |
| first quart 0.39 0.39 |
| third quart 0.40 0.40 |
| **The DecisionTreeClassifier was run 1,000 times where it recorded the precision and recall values each time. These statistics were generated using this data.** |

# VALIDATION STRATEGY

To validate the data which was split into two groups which were labeled as training and testing sets. The algorithm was used to train the data and validate using the testing data. A trivial mistake would be to use the testing data to train, because even though it will generate great results in a simulated environment, those results will not translate to a real-world environment.

Since the dataset is small, a **StratifiedShuffleSplit** as part of the model\_selection package (sklearn) is the best choice. It will return stratified randomized folds (1,000) which are produced by preserving the percentage of samples of each class.

# ALGORITHM PERFORMANCE

Code was added to the *poi\_id.py* script to run the algorithm with the tuning parameters 1,000 times.

The table to the left clearly shows that precision and recall results above 0.3 for each iteration were achieved.

# PERSONAL INSIGHTS

Enron’s upper management didn’t write their own emails. In fact, according to this dataset, Ken Lay didn’t write a single email as he had a personal assistant write and send the mails on his behalf. This means that we should not use word choice as part of our analysis, as it would only be analyzing his assistant’s vocabulary. The metadata, “to” information is slightly more useful as we can assume that management gave direction as to where their emails were sent. Though limited information could be garnered from the emails under the assumption that non-POIs emailed each other more than they emailed POIs, it proved possible to manipulate the financial dataset with enough insight to make that analysis outdated. This is why I dropped the email metadata from this analysis.

I did run across a video[[6]](#footnote-6) the other day that claimed that you can tell who a director is at Enron by comparing the number of the emails they sent to the number of nodes.

1. <https://en.wikipedia.org/wiki/Enron> [↑](#footnote-ref-1)
2. <http://www.forbes.com/2002/01/15/0115enron.html> [↑](#footnote-ref-2)
3. <https://en.wikipedia.org/wiki/Enron_scandal> [↑](#footnote-ref-3)
4. <https://www.cs.cmu.edu/~./enron/> [↑](#footnote-ref-4)
5. [http://www.forbes.com/sites/ryanholiday/2012/04/16/what-the-failed-1m-netflix-prize-tells-us-about-businessadvice/](http://www.forbes.com/sites/ryanholiday/2012/04/16/what-the-failed-1m-netflix-prize-tells-us-about-business-advice/) [↑](#footnote-ref-5)
6. <https://www.youtube.com/watch?v=GBzoNgqF-gQ&t=38m17s> [↑](#footnote-ref-6)