**Irena Trendafilova**
**Intro to Machine Learning**

**Source code:**
https://github.com/irenatrend/Data_Analyst_Nanodegree_Udacity/tree/master/Intro_To_Machine_Learning/final_project

**Enron Submission Free-Response Questions**

# Intro

**Enron** was an American energy trading company, based in Houston Texas, which in around the year 2000 was one of the top ten largest companies in America. But, by the middle of 2002 the company was bankrupt, tens of thousands of people's lost their jobs and there were dozens of people in jail. The Enron scandal is one of the largest corporate fraud cases in history. When something like this happens, there must be some kind of fraud. Enron Company was involved in different schemas, such as selling assets to shell companies at the end of each month, and buying them back at the beginning of the next month to hide accounting losses, causing electrical grid failures in California and a plan in collaboration with Blockbuster movies to stream movies over the internet. There was a lot of fraud!

# The goal of this project & the Enron data set

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"]**

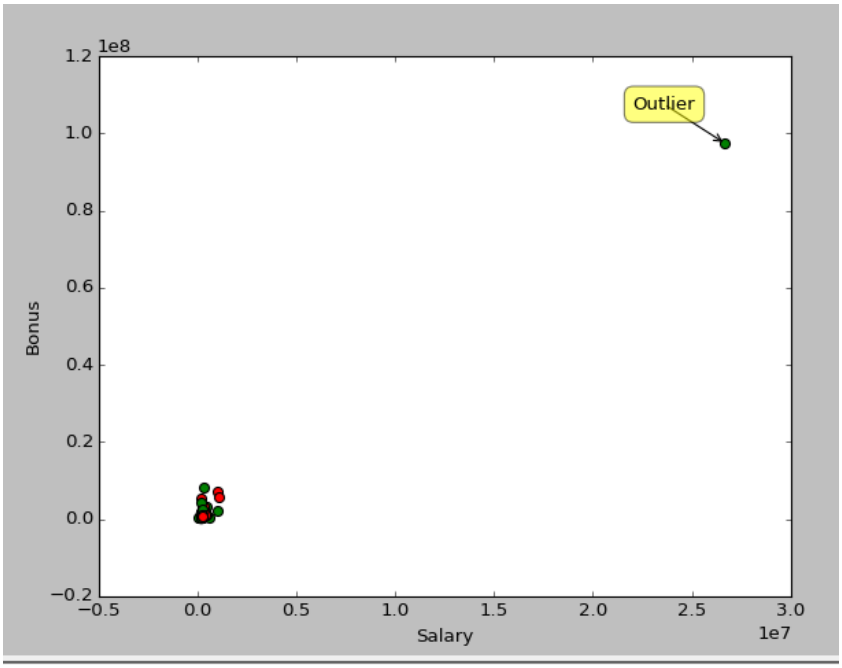
The objective of this project is, using the email and financial data from 145 executives at Enron to build a predictive model that could identify whether an individual could be considered as a "person of interest" (POI). A person of interest (POI) is someone who was charged with a crime, settled without admitting guilt or someone who testified for the government in exchange for immunity from prosecution. In this project I'll try to build POI identifier using the power of machine learning algorithms.

The Enron dataset includes 146 data points. For each person there are 20 features available (14 financial features, 6 email features) and 1 target label.

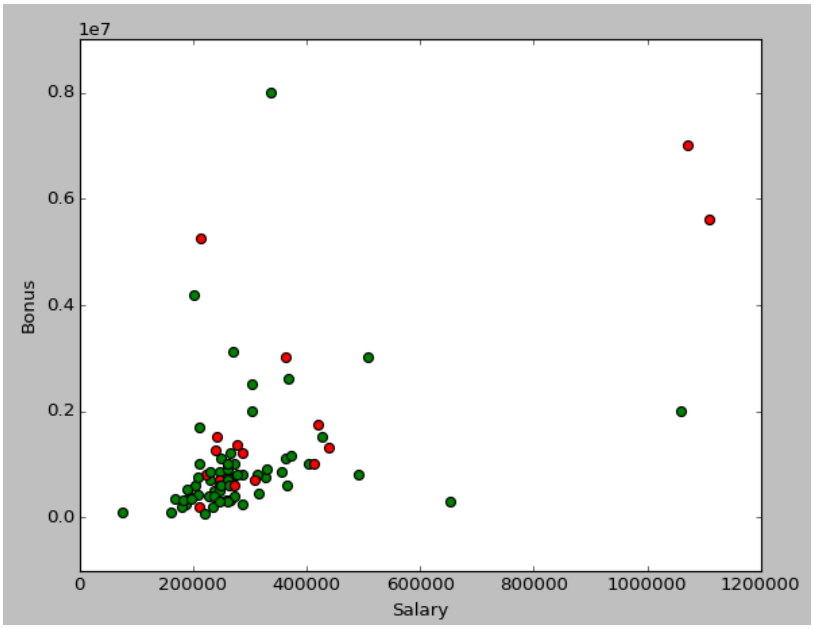
- **financial features**: ['salary', 'deferral_payments', 'total_payments', 'loan_advances', 'bonus', 'restricted_stock_deferred', 'deferred_income', 'total_stock_value', 'expenses', 'exercised_stock_options', 'other', 'long_term_incentive', 'restricted_stock', 'director_fees']
  (All units are in US dollars)
- **email features**: ['email_address', 'to_messages', 'from_poi_to_this_person', 'from_messages', 'from_this_person_to_poi', 'shared_receipt_with_poi']
  (Units are generally number of emails messages; notable exception is 'email_address', which is a text string)
- **POI label**: ['poi'] (Boolean, integer, is POI or not. Target variable.)

In the Enron dataset, there are 18 people who are indicated as POI and 127 as NON-POI. In total there were 35 POIs, but some POI are missing in the Enron dataset. Not having enough data to really learn the patterns can be a potential problem.

During the class and in my analysis I've noticed that there is one outlier. The name of that data point is 'Total', which was not an executive, and contained the sum of each financial feature for all 145 executives. The 'Total' outlier is represented in the scatterplot below.




After removing the 'Total' outlier I've created a new scatterplot.



Four points in this scatterplot seemed to be outliers (high salary or high bonus), but since they are real executive's data I decided to keep them because they are potential POIs.

## Feature Processing

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 doesn't 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.) If you used an algorithm like a decision tree, please also give the feature importances of the features that you use. [relevant rubric items: "create new features", "properly scale features", "intelligently select feature"]**

### Feature selection

After cleaning the data set from outliers, the next step was going through the process of feature selection. Initially I've used SelectKBest to select the 10 best features, and here are the results I've got for the top 10 features:

[('exercised_stock_options', 25.097541528735491), ('total_stock_value', 24.467654047526398),
('bonus', 21.060001707536571), ('salary', 18.575703268041785),
('deferred_income', 11.595547659730601), ('long_term_incentive', 10.072454529369441),
('restricted_stock', 9.3467007910514877), ('total_payments', 8.8667215371077752),
('shared_receipt_with_poi', 8.7464855321290802), ('loan_advances', 7.2427303965360181)]

After I've tested my model with the 10 best features the precision and recall were below 0.3 so I've continued to test my model by deleting the features manually or based on their feature importance score. I've played a lot here. I've tried a lot of different combinations of features. In the table below we can see different values I've got for precision and recall. Finally I picked the following features for my final model: 'exercised_stock_options', 'total_stock_value', 'bonus', 'salary'.

| Num. of features | features_list |
|---|---|
| 10 | 'exercised_stock_options', 'total_stock_value', 'bonus', 'salary', 'deferred_income', 'long_term_incentive', 'restricted_stock', 'total_payments', 'shared_receipt_with_poi', 'loan_advances' |
| 6 | 'exercised_stock_options', 'total_stock_value', 'bonus', 'salary', 'deferred_income', 'long_term_incentive' |
| 4 | **'exercised_stock_options', 'total_stock_value', 'bonus', 'salary'** |
| 3 | 'exercised_stock_options', 'total_stock_value', 'bonus' |

**DecisionTreeClassifier Results**

| Num. of features | Average Precision | Average Recall | DecisionTreeClassifier | PCA | scaler |
|---|---|---|---|---|---|
| 10 | 0.1 | 0.111 | | | |
| 6 | 0.161 | 0.166 | | | |
| 4 | **0.301** | **0.277** | | | |
| 3 | 0.507 | 0.277 | | | |
| 10 | 0.231 | 0.166 | min_samples_split=2, random_state=10 | | |

| | | | | | |
|---|---|---|---|---|---|
| 6 | 0.111 | 0.111 | min_samples_split=2, random_state=10 | | |
| 4 | **0.273** | **0.277** | min_samples_split=2, random_state=10 | | |
| 3 | 0.340 | 0.277 | min_samples_split=2, random_state=10 | | |
| 10 | 0.125 | 0.111 | min_samples_split=2, random_state=10 | n_components=2, copy=True, whiten=False | StandardScaler |
| 6 | 0.133 | 0.111 | min_samples_split=2, random_state=10 | n_components=2, copy=True, whiten=False | StandardScaler |
| **4** | **0.420** | **0.333** | min_samples_split=2, random_state=10 | n_components=2, copy=True, whiten=False | StandardScaler |
| 3 | 0.290 | 0.444 | min_samples_split=2, random_state=10 | n_components=2, copy=True, whiten=False | StandardScaler |

I've created two new features for this project.

- **emails_fraction_from_poi** → the fraction of all emails that a person received that were sent from a persons of interest.
- **emails_fraction_to_poi** → the fraction of all emails that a person sent that were addressed to a persons of interest.

Scaling features improved my model precision and recall. During my testing I've scaled the features using MinMaxScaler and StandardScaler scalers. Since StandardScaler give me better results I've selected StandardScaler for my final model. I've also applied PCA to my model, and had improvements on on recall and precision.

**Importance of the features used in the final model**

| | |
|---|---|
| **total_stock_value** | 0.3296 |
| **exercised_stock_options** | 0.2754 |
| **bonus** | 0.2422 |
| **salary** | 0.1526 |

## Algorithm Selection

3. What algorithm did you end up using? What other one(s) did you try? [relevant rubric item: "pick an algorithm"]

I've tried different algorithms, the one like GaussianNB, SVC, DecisionTreeClassifier , AdaBoostClassifier and RandomForestClassifier. For my final model I've ended up using DecisionTreeClassifier. Here are the precision and recall results of DecisionTreeClassifier and RandomForestClassifier.

| Classifier | Precision | Recall |
|---|---|---|
| DecisionTreeClassifier | 0.420 | 0.333 |
| RandomForestClassifier | 0.611 | 0.166 |

## Tuning

**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 don't 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 if you used, say, a decision tree classifier). [relevant rubric item: "tune the algorithm"]**

Some machine learning algorithms have parameters that affect the fit of the model. Algorithms may perform differently using different parameters. Tuning parameters can help us to ensure that the model neither over fits nor under fits on the training data and allow us to tradeoff between precision and recall. If we don't tune the parameters well, we can over-tune an algorithm to predict great on the training data, but fail on test/new data. While tuning the parameters my main goal was to achieve precision and recall > 0.3.

During my analysis and testing I've tuned the parameters of different algorithms using GridSearchCV or manually. GridSearchCV is a way of systematically working through multiple combinations of parameter tunes to determine which tune gives the best performance. After selecting features and algorithm I manually tuned min_samples_split parameter of the Decision Tree Classifier. Here are the results of the manually tuning the min_samples_split parameter of the Decision Tree algorithm.

| min_samples_split | Precision | Recall |
|---|---|---|
| **2** | **0.420** | **0.333** |
| 4 | 0.388 | 0.277 |
| 6 | 0.412 | 0.333 |
| 8 | 0.353 | 0.222 |
| 10 | 0.325 | 0.277 |

## Validation

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"]**

Validation is a way to measure the predictive performance of a statistical model, by running the model against a new dataset. One way to measure performance is to split the original dataset into **training and testing datasets**, train the model using the training dataset, and test the model using the test dataset. Since the training set can end up containing most of one category data points, while the test set contains another category data points, to avoid this problem the data can be **randomly shuffled** before creating training and testing set.

**Over fitting** is the classic mistake one can do when not performing validation correctly. Several reasons which can cause a model to over fit are: too many features relative to the number of records, small dataset used for training or testing the model using training data. Using testing data gives us an estimate of the performance of the classifier or regression on an independent data set, and also helps us to prevent over fitting.

This model was validated using **3-fold cross-validation**. The idea behind cross-validation is selecting part of data as the training set (67% of the data in this 3-fold cross-validation), and the rest of it is test set

(33% of the data). In case of Enron dataset using accuracy for validation is a mistake, because we have an imbalance problem where the number of POIs is small compared to the total number of examples in the dataset. In the tester.py code the precision and recall were averaged among these 3 folds to quantify the performance of different algorithms.

## Performance

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've used Precision and Recall as evaluation metrics in this project. The average performance for precision was 0.420 and for recall was 0.333. It is clear that with this performance this model cannot be the only source POI identification.

| Metric | Value | Explanation |
|---|---|---|
| Accuracy | X | Measure how good the algorithm predicts weather or not someone is a POI. |
| Precision | 0.420 | The precision can be interpreted as the likelihood that a person who is identified as a POI is actually a true POI. |
| Recall | 0.333 | The Recall measures correctly identified real life POIs. |