Identify 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”]:

The goal of the project is to determine a Person of Interest (POI) identifier by using the Enron email and financial dataset for the creation of a model which will accurately predict if the subject is a person of interest in a fraud investigation.

Machine learning proved useful in accomplishing this goal due to its natural ability to categorize data, identify trends and the application of these learnings to new data.

Enron was one of the largest companies in the United States. However, by 2002, it collapsed into bankruptcy due to widespread corporate fraud. During a federal investigation, the Enron corpus, emails and financial information, which was retrieved/gathered together were released to the public and was utilized as a dataset in this project.
The dataset used for this project was pre-processed and assimilated by Katie Malone from Udacity.

Data exploration was the first step to understand the characteristics of the dataset. This involved determining the size of the data set, number of features to be utilized, etc.

While exploring the dataset, a number of outliers were discovered. The fate of the outliers was determined by further data analysis. Aware of the importance of outliers in representing the complete picture in every case, elimination of a few irrelevant, extreme outliers was undertaken. The eliminated outliers included a data point representing “**Total**” and not an actual observation, and the other was listed as "**The Travel Agency in The Park**". This outlier seems to be the account of a company. This elimination was executed after careful consideration of the consequence on the output.

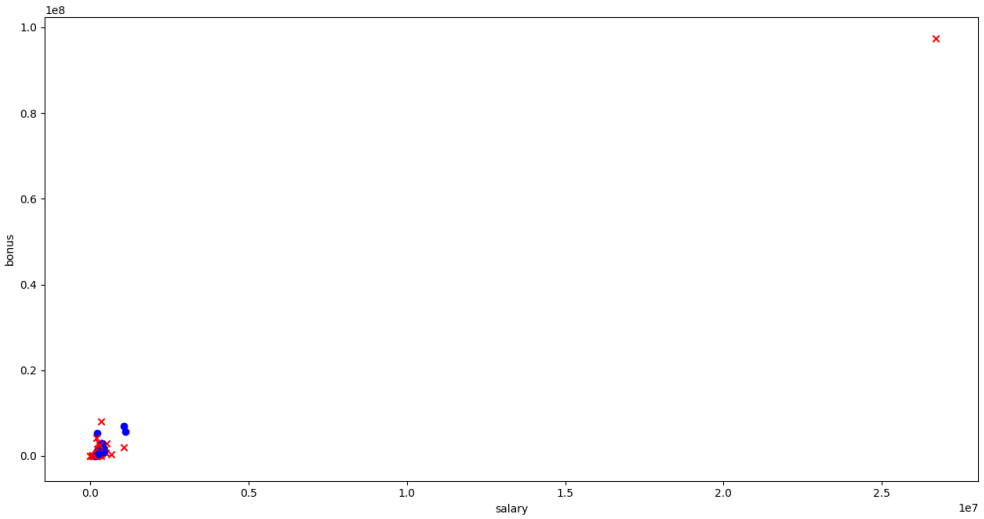
Incompleteness was also a problem with the dataset. Below is an overview of the data, missing values, and interesting data points.

Data Overview:

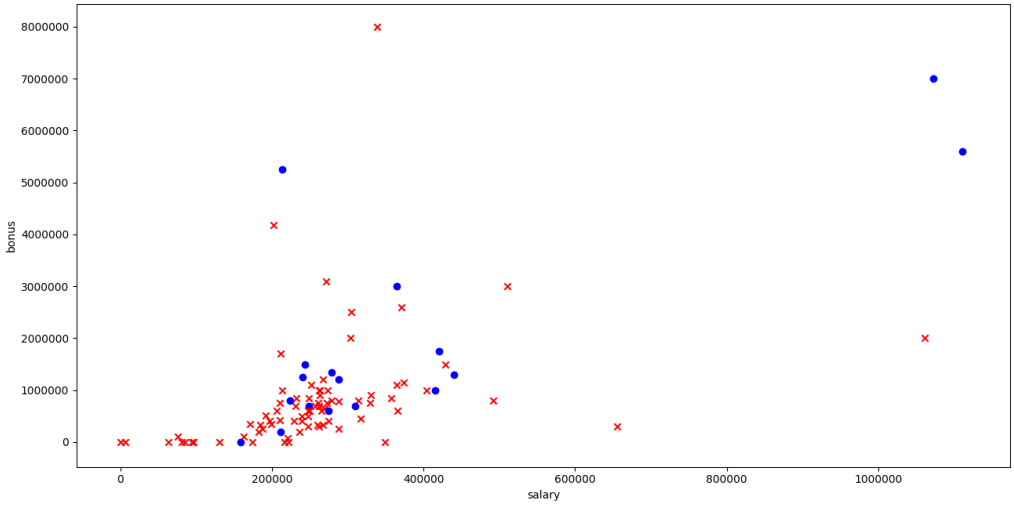
1. Number of people in the dataset: 146
2. Number of Persons of Interests (POIs) in the dataset: 18 out of 34 total POIs
3. Number of non-POIs in the dataset: 128
4. POIs with zero or missing to/from email messages in the dataset: 4
    a. KOPPER MICHAEL J
    b. FASTOW ANDREW S
    c. YEAGER F SCOTT
    d. HIRKO JOSEPH

5. Salary vs Bonus (before and after) dropping outliers:
    Here the outlier is "**Total**" that is the sum of all value. Hence it is removed.

Salary vs Bonus with outliers



Salary vs Bonus without outliers



6. Number of non-POIs in the dataset: 128
7. Salary Bonus Fortuner (Employees receiving a salary or bonus of 2M+ and 5M+, respectively):
    a. LAVORATO JOHN J
    b. LAY KENNETH L
    c. BELDEN TIMOTHY N
    d. SKILLING JEFFREY K

8. Incomplete data - NaN values in features:

Updating NaN values in features

Feature	NAN updated
salary	51
to_messages	60
deferral_payments	107
total_payments	21
loan_advances	142
bonus	64
email_address	0
restricted_stock_deferred	128
total_stock_value	20
shared_receipt_with_poi	60
long_term_incentive	80
exercised_stock_options	44
from_messages	60

other	53
from_poi_to_this_person	60
from_this_person_to_poi	60
poi	0
deferred_income	97
expenses	51
restricted_stock	36
director_fees	129

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 importance 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”]

Both univariate feature selection and engineering were performed and used in testing when creating the final model.
Feature scaling was also utilized as there were a number of outliers which could skew the results (be used as a primary predictor) but due to the validity of the data, these points could not be removed. Although performance was tested with and without feature scaling as a reassurance to the process, the final model does not utilize feature scaling.

Feature Selection:

Feature selection was performed by SelectKBest to obtain the best Precision and Recall scores during numerous testing. No features used were manually picked. Training test sets were created with the use of Stratified Shuffle Split cross validation due to the small size of the data and that of the POI list.

The final features utilized in the model are:

1. salary
2. deferred_income
3. total_stock_value
4. expenses
5. exercised_stock_options
6. long_term_incentive
7. restricted_stock
8. director_fees

Feature Engineering:

Three features were engineered for testing of the model:

1	**to_poi_fraction**	a fraction of the total 'to' emails that were sent to a POI
2	**from_poi_fraction**	a fraction of the total 'from' emails that were received from a POI
3	**salary_bonus_fraction**	a fraction of salary to bonus money

I believe that the quality of the project would be significantly improved by adding two additional features. These features are stated below

1. calculate the percentage/relationship of a POI with other employees at the company via their 'to' and 'from' email interaction. This will produce insightful and useful information, allowing the algorithm to use these values as predictors.

   E.g. if person A sends (or receives) a large portion of their total emails to/from a POI, there may be a greater likelihood that
   person A is also a POI.

2. Like-wise, after discovering the large salary and bonuses of some of the POIs, I believed knowing the fraction or multiplier between someone's salary and their bonus would help as a predictor for the algorithm.

GridSearchCV was initially used but being unable to easily identify k best features. It gave kbest = 8 for which the precision was very low whereas for kbest = 11 it was higher.
I did a number of separate SelectKBest testing, and viewed their precision and recall scores. With SelectKBest, the number 11 for k was decided upon after number of performance and

tuning which determined this to deliver the best mix of performance (timing), precision and recall.

kbest	Accuracy	Precision	Recall	F1 Score
8	0.791	0.27348	0.27348	0.27646
9	0.79653	0.27541	0.3225	0.2971
11	0.8352	0.3767	0.3605	0.36842
12	0.8392	0.3272	0.364	0.3342

Here are the few sample on different “kbest” values. And since for kbest=11 all parameter are high, hence it is chosen.

One of the above engineered feature “**to_poi_fraction**” topped the list of best features because of its high score.

Below is a table of features and their scores:

Feature	Score
salary	10.050053
bonus	4.505912
deferred_income	5.55813
total_stock_value	22.810947
expenses	3.223887
exercised_stock_options	18.938347
long_term_incentive	14.612171
restricted_stock	25.3775
from_this_person_to_poi	1.88522
to_poi_fraction	4.036608

Some engineered features are also added to the final list because of the score. It didn’t reflected every time in the top 11 list but it came many times with high score. So I kept it in the final feature list.

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”]*

The POI identifier model uses the AdaBoost and DecisionTree algorithm as it provided the best validation results.

The other main algorithms used were Random Forest and GaussianNB, and Logistic Regression. All of which performed adequately in one aspect or another. For example, Random Forest provided the best accuracy score but had low Precision and Recall scores.

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"]

Tuning the parameters of an algorithm is the process of changing, testing and updating the parameters in order to get the right mix or settings. The tuning once completed, results in parameters which are optimized to produce the best results. Most ML algorithms have parameters and in some cases, there are default values, so it's not always necessary to "tune" an algorithm but in a lot of cases it is deemed essential.

Great care is to be taken while tuning the parameters. As a shabby tuning could generate a model that seems correct but actually produces false data.
In the case of my final model, I used trial error to check for the best value I get out of the classifier. After updating different parameters it became intuitive and results are below.

	Accuracy	Precision	Recall	F1 Score	F2 Score
AdaBoostClassifier	0.8352	0.3767	0.3605	0.36842	0.36363
RandomForestClassifier	0.868	0.51253	0.2045	0.29235	0.23244
GaussianNB	0.84133	0.38415	0.315	0.34615	0.32676
LogisticRegression	0.60867	0.03597	0.075	0.04862	0.06163

The Precision is better for GaussianNB but due to less Recall value it is not taken up. AdaBoostClassifier has balance of all values hence it is taken for further tuning.

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 the process of checking your model's prediction against data that wasn't used during the algorithm/model training phase.

A classic mistake of overfitting occurs when you train the algorithm on all available data instead of splitting it into training and testing data. Overfitting causing the model to merely memorize classification and not 'learn' to generalize and apply this information to new datasets.

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"]*

The final model uses Precision, Recall and F1 scores to evaluate how good the model is in predicting POIs.
The raw data can be seen below. Each observation is a test, and each test made 15,000 predictions.

	Paramter n_estimators	**Parameter** learning_rate	**Accuracy**	**Precision**	**Recall**	**F1 Score**	**F2 Score**
AdaBoostClassifier	50	0.8	0.8352	0.3767	0.3605	0.36842	0.36363
AdaBoostClassifier	40	0.6	0.8275	0.3569	03660	0.36139	0.36414
AdaBoostClassifier	50	0.4	0.8173	0.3512	0.4370	0.38948	0.41667

Since, the parameter in the first row gives a good balance of accuracy, precision and recall. It is chosen for validating the results further.
This is determined using the "tester_classifier" by passing the dataset to it to get the accuracy, precision and recall for the final tuning of parameters.

Here, "n_estimaors" and "learning_rate" are variated to get the perfect tuning of the algorithm. Since both are interdependent hence both are changed together to get the result.

	Total Prediction 1500	Actual	
		Positive	Negative
Predicted	Positive	720	1192
	Negative	1280	11808

Precision is the measurement of how many selected items were identified as relevant
Recall is the measurement of how many relevant items were selected

The model's precision is approx. 38 % i.e from the people classified as POIs by the model, 38% of them are actual POIs. However, the model's recall is approx. 36 % i.e from the number of actual POIs in the total dataset, the model correctly identifies 36 % of them.

It can be concluded that although the model "spreads a wide net" it will capture over 36 % of actual POIs.

7. References:

1. http://scikit-learn.org/stable/modules/
2. https://en.wikipedia.org/wiki/Enron_Corpus