

# Identifying Fraud from Enron Emails and Financial Data

Haleh Dolati

## Project Overview

Enron Corporation was an American energy, commodities, and services company, which was formed in 1985 by, Kenneth Lay. This corporation filed for bankruptcy on December 2001. Enron case is famous for being the largest bankruptcy reorganization the biggest audit failure in American history at that time. In this project, we are asked to use the Enron email data and identify employees who may have committed fraud and were involved in the Enron scam, using supervised machine learning.

Machine learning algorithms are very useful to accomplish these kinds of goals, because their output is a machine system that builds models without demanding human involvement which would be really time consuming. The goal is to make a model that uses some financial and email data to identify Persons of Interest among the executives. The Person of Interest (POI) is defined as *"individuals who were indicted, reached a settlement or plea deal with the government, or testified in exchange for prosecution immunity"*. The mentioned information became public after the federal investigation of Enron scandal. Enron Email Dataset was prepared by the CALO Project in MIT and is available here: (<https://www.cs.cmu.edu/~.enron/>).

This data has financial and email features as well as POI label which shows whether an executive was POI as well. The features are:

- **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
- **Email features:** to\_messages, email\_address, from\_poi\_to\_this\_person, from\_messages, from\_this\_person\_to\_poi, shared\_receipt\_with\_poi

## Data Cleaning and Outliers

The first step is to know the data and find and handle possible problems. First, I took a look at the name of the executives. Over all there are 146 executives in this data which 18 of them are POIs. The last two names **Total** and **The Travel Agency In The Park** are different from the rest. Total is the sum of all financial features therefore does not represent any executive just like The Travel Agency In The Park. Therefore, I excluded them from the data.

Each executive is presented by 21 features. I went through them and using my intuition and some online research about the Enron scandal, I excluded some of them. At the end, I chose a mixture of financial and email data to be used in the feature selection part. This table shows all the available features and the one that I used. In the table below, the red ones are the excluded ones:

Feature's Name	Included?	Feature's Name	Included?	Feature's Name	Included?
bonus	Yes	exercised_stock_options	Yes	deferral_payments	Yes
from_this_person_to_poi	Yes	from_poi_to_this_person	Yes	salary	Yes
email_address	No	deferred_income	No	director_fees	No
restricted_stock_deferred	Yes	total_payments	Yes	from_messages	No
loan_advances	No	expenses	No	other	No
shared_receipt_with_poi	No	to_messages	No	long_term_incentive	No
total_stock_value	No	restricted_stock	No		

I also looked into each features value to see whether there are missing values or anything out of ordinary. Of course there are features with very different value range, especially in financial features. However, I'm not going to exclude those extremely high values because many of them like extremely high bonuses actually can be an indicator of the fraud. Another step that I took was scaling the data. As mentioned before, there are two types of variables: financial and email. Email data has smaller numbers compared to financial data. Therefore, I've decided to scale them all.

Features	Values	Features	Values
poi	144 non-null	bonus	81 non-null
salary	94 non-null	salary_to_avg	94 non-null
deferral_payments	38 non-null	total_payments	123 non-null
restricted_stock_deferred	17 non-null	exercised_stock_options	101 non-null
from_poi_to_this_person	86 non-null	from_this_person_to_poi	86 non-null

In addition to the mentioned features, I made a new feature by dividing the salary of each executive to the average of all salaries. This ratio shows how ones salary is way above the average.

## Classifier Selection

Before the tuning step, I ran Gaussian Naive Bayes, K-Neighbors, and Decision Tree using their default parameters. While Gaussian Naive Bayes results were really underwhelming, the other two algorithms showed very good results.

Classifier	Accuracy	Recall	Precision
Gaussian Naive Bayes	0.28	0.16	1.0
K-Neighbors	0.82	0.34	0.25
Decision Tree	0.82	0.4	0.5

As you can see, with the default parameters, decision tree achieved the best accuracy, recall and precision. However, K-Neighbors has the same accuracy and the other two metrics are not

as bad as the Gaussian Naive Bayes. Based on these results, I decided to move forward with both K-Neighbors and Decision Tree and improve both of them with scaling and feature selection.

## Tuning

Tuning the parameters of a machine learning algorithm means to manually change and optimize the values of them in order to enable the algorithm to perform the best. These parameters have a default value that will be applied if no other values have been assigned manually. For example, K-Neighbors has a parameter named weights, with the default value of 'uniform'. This parameter defines the weighting system for the points. The default value of 'uniform' means all points in each neighborhood has the same weight. However, this parameter can take another pre-defined value: 'distance'. This value, 'distance', assigns the inverse of each point's distance as their weight. Here, I chose K-Neighbors and Decision Tree for tuning. For Decision tree, I chose the following parameters to tune:

### K-Neighbors

Parameter	Description	Values
leaf_size	Number of cases or observations in that leaf.	30, 50
n_neighbors	Number of neighbors to use by default for k_neighbors queries	5, 10
p	Power parameter for the Minkowski metric	1, 2
weights	Weighting system for the points	Distance, Uniform
algorithm	Algorithm used to compute the nearest neighbors	Auto, Ball_Tree, Kd_Tree, Brute

### Decision Tree

Parameter	Description	Values
criterion	The function to measure the quality of a split	Gini, Balanced
class_weight	Weights associated with classes	None, Balanced
min_samples_split	The minimum number of samples required to split an internal node	2, 3, 4, 5
max_leaf_nodes	Grow a tree with max_leaf_nodes in best-first fashion	None, 5
min_samples_leaf	The minimum number of samples required to be at a leaf node:	4, 2

The SelectKBest chose 6 best features to be included in the model. As it is shown in the following table, **bonus** and **exercised\_stock\_options** have significantly higher importance score compared to the rest:

- bonus: 0.35689672
- exercised\_stock\_options: 0.26079073
- from\_poi\_to\_this\_person: 0.18264055
- salary\_to\_avg: 0.11818132
- total\_payments: 0.08149068
- salary: 0.

The after tuning results shows although K neighbors has slightly higher accuracy after tuning, Decision Tree has much better Precision, Recall, F1, and F2. Therefore, I chose it as the final tuned algorithm.

Classifier	Accuracy	Precision	Recall	F1	F2
KNeighborsClassifier	0.87	0.54	0.18	0.27	0.21
DecisionTreeClassifier	0.82	0.35	0.43	0.38	0.41

Accuracy, recall and precision are the metric that i focused on for choosing my algorithm. these three metrics are using values from confusion matrix to evaluate a n algorithm. Accuracy shows the how many of the predicted labels were predicted correctly. In other words, accuracy is:

$$\text{accuracy} = (\text{true positives} + \text{true negatives}) / \text{total predictions}$$

However, depending on the scenario that we want to predict, the false positive or false negative predictions might be far more important for us. For example, when it comes to medical tests, specially in the first steps, the preference is to predict one might has a type of cancer and rule it out by further tests (false positive) rather than missing someone who has cancer and finding out about it when it is too late (false negative). For these cases, we can take advantage of two other metrics: precision and recall.

$$\text{precision} = \text{true positives} / (\text{true positives} + \text{false positives})$$

$$\text{recall} = \text{true positives} / (\text{true positives} + \text{false negatives})$$

The results of Decision Tree modes show the accuracy of the model is 82%. it mean out of 100 predictions, 82 of them will be predicted correctly. The precision is 35% which means 35% of the predictions that were identified as POIs, were actually POIs. Finally, the 43% value for Recall the model catches 43% of the POIs.

The SelectKBest chose 6 best features to be included in the model. As it is shown in the following table, bonus and exercised\_stock\_options have significantly higher importance score compared to the rest:

- bonus: 0.35689672
- exercised\_stock\_options: 0.26079073
- from\_poi\_to\_this\_person: 0.18264055
- salary\_to\_avg: 0.11818132
- total\_payments: 0.08149068 s
- alary: 0.