# Identify Fraud from Enron Email

Author: Sergio Comuñas

Date: 06/09/2017

## 1. Summary

**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 this project is make a procedure for identify employees from Enron who may have committed fraud as a POIs (Persons of Interest) based on his financial info and his e-mail activity. Machine Learning algorithms are strong recommended for this type of problems because can handle a lot of data and make accurate predictions.

## 1.1 Dataset description

There is a total of 146 employees. 18 of them are identified as POIs.

## 1.2 Dataset features

There are 14 financial features (all of it in US $):

- 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

There are 6 e-mail features (all of it in number of e-mails except *email-address* wich is a string with the mail of each person):

- to_messages
- email_address
- from_poi_to_this_person
- from_messages

- from_this_person_to_poi
- shared_receipt_with_poi

## 1.3 Target variable

There is a variable called *poi* that is a Boolean indicating if the person is POI or not (1 or 0).

## 1.4 Missing values in features

I've made an analisys of missing values in each feature with the next results:

- Financial
    - 'salary': 51
    - 'deferral_payments': 107
    - 'total_payments': 21
    - 'loan_advances': 142
    - 'bonus': 64
    - 'restricted_stock_deferred': 128
    - 'deferred_income': 97
    - 'total_stock_value': 20
    - 'expenses': 51
    - 'exercised_stock_options': 44
    - 'other': 53
    - 'long_term_incentive': 80
    - 'restricted_stock': 36
    - 'director_fees': 129
- E-mail
    - 'to_messages': 60
    - 'email_address': 35
    - 'from_poi_to_this_person': 60
    - 'from_messages': 60
    - 'from_this_person_to_poi': 60
    - 'shared_receipt_with_poi': 60
- Target
    - 'poi': 0

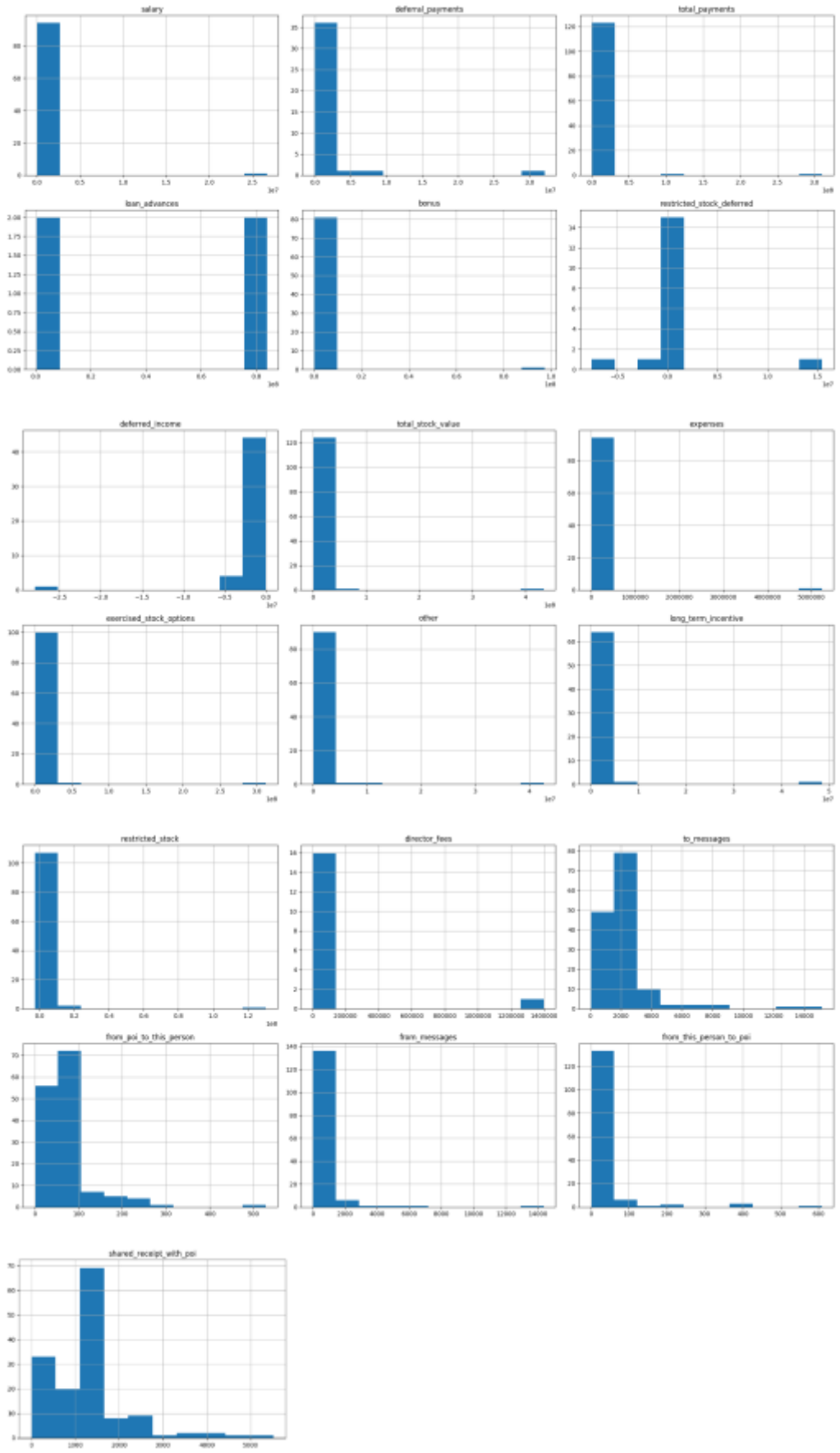
You can see there are NaN values in all the features but not in the target.

For the financial features we can assume that NaN values are 0 because it's been taken from the table in "enron61702insiderpay.pdf". Here you can find there's no 0, instead there are '-' character. We have not to do anything in *poi_id* because *featureFormat* function also do it.

E-mail features are different, you can see there are 60 people with no data in the e-mail features. If we left it may introduce some bias in the prediction. Also remove 60 people from 146 is not possible. I've imputed missing in e-mail features to the mean of each feature.

## 1.5 Outliers

I've made histograms for each feature in order to see if there are outliers:




As you can see in the financial features (for example salary and bonus) there are extreme outlier for investigate. I've made top3 of this 2 features:

- Salary
    - TOTAL: 26704229.0$
    - SKILLING JEFFREY K: 1111258.0$
    - LAY KENNETH L: 1072321.0$
- Bonus
    - TOTAL: 26704229.0$
    - SKILLING JEFFREY K: 1111258.0$
    - LAY KENNETH L: 1072321.0$

Here we can find our first outlier: TOTAL is not an employee. This is an import mistake.

Second exercise I've made is to look for all NaNs value in any row. And there's the second outlier: LOCKHART EUGENE E has no data in any feature.

Finally, looking at the table in PDF file you can find a row called: THE TRAVEL AGENCY IN THE PARK that also is not a person.

I've removed all three.

# 2. Feature Selection

**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 importances 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", "intelligently select features", "properly scale features"].**

## 2.1 New features created

I was thinking maybe someone has a poor activity on his mail but all of this is related to or from POIs. Then I think that maybe counting mails as absolute value is an error. Then, I've created: from_poi_perc and to_poi_perc that gives us the percentage of mails (from or to POIs) relative to the total amount of mails.

Finally you will see that is one of the feature picked for our model.

## 2.2 Scaling

I've used minmaxscaler function in order to scale features but only for SVM and K-nearest neighbours model types. I also try Decision Tree classifier but don't use scaling.

## 2.3 Feature Selection

I've find most important features using SelectKBest and Feature importance on a simple Decission Tree. These are the results:

| | scoreSKB | scoreDT |
|---|---|---|
| exercised_stock_options | 24,81508 | 0,19984 |
| total_stock_value | 24,182899 | 0,055633 |
| bonus | 20,792252 | 0 |
| salary | 18,289684 | 0 |
| to_poi_perc | 13,280903 | 0,115235 |
| deferred_income | 11,458477 | 0 |
| long_term_incentive | 9,922186 | 0,104762 |
| restricted_stock | 9,212811 | 0,025162 |
| total_payments | 8,772778 | 0,059016 |
| loan_advances | 7,184056 | 0 |
| expenses | 6,094173 | 0,047667 |
| shared_receipt_with_poi | 5,696352 | 0,126762 |
| other | 4,187478 | 0,19218 |
| from_poi_to_this_person | 3,01796 | 0,015213 |
| director_fees | 2,126328 | 0 |
| from_this_person_to_poi | 1,35263 | 0,05853 |
| from_poi_perc | 1,154316 | 0 |
| from_messages | 0,582371 | 0 |
| to_messages | 0,366008 | 0 |
| deferral_payments | 0,224611 | 0 |
| restricted_stock_deferred | 0,0655 | 0 |

Finally I've decided to take the top5 features: exercised_stock_options, total_stock_value, bonus, salary and to_poi_perc.

## 3. Algorithm Selection

**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"]**

I've decided to use: Decision Tree, SVM and K-nearest neighbours models and see wich one has great performance. I've also use Grid Search in order to help me tune the parameters for each model. The results are the next ones:

| | Decision Tree | SVM | K-nearest Neighbours |
|---|---|---|---|
| Accuracy | 0,81613 | 0,75707 | 0,81913 |
| Precision | 0,40011 | 0,26701 | 0,22682 |
| Recall | 0,759 | 0,471 | 0,148 |
| F1 | 0,52399 | 0,34081 | 0,17912 |
| Total predictions | 15000 | 15000 | 15000 |
| True positives | 1518 | 942 | 296 |
| False positives | 2276 | 2586 | 1009 |
| False negatives | 482 | 1058 | 1704 |
| True negatives | 10724 | 10414 | 11991 |

Finally I've decided to pick Decision Tree based on the results. I've read that SVM doesn't works very well on unbalanced datasets (like this because there's only 18 of 146 trues on the targe variable). K-nearest neighbours have great accuracy but Precision and Recall are so lower (there are so little True Positives as you can see).

## 4. Parameter Tunning

**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? What parameters did you tune? (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 items: "discuss parameter tuning", "tune the algorithm"]**

Tune the parameters of our model is basic to improve the performance. Each one has it's own particular ones and must be defined previous.

In this case we have 3 models and a lot of parameters to tune. Then, as I mention on previous point, I've decide to use GridSearch function to improve it.

Then, for example, for Decision Tree I've combined this parameters:

- criterion=['gini', 'entropy']
- max_depth=[None, 5, 10,15,20, 30, 40
- min_samples_split=[2, 5, 10, 20
- class_weight=[None, 'balanced']
- random_state=[42]
- splitter = ['best','random']
- max_leaf_nodes = [5,10,20, 30, 40]

Using GridSearch we finally choose the next parameters:

- criterion=['gini']
- max_depth=[None]
- min_samples_split=[20]
- class_weight=['balanced']
- random_state=[42]
- splitter = ['best']
- max_leaf_nodes = [5]

As you can see the most of the parameters are not default ones. This proves that tunning is a must in model selection.

## 5. Validation

**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 items: "discuss validation", "validation strategy"]**

Validation is the mode we have to see if the performance of our model is good enough for us.

There are 2 classic mistakes in this part:

- Take only accuracy parameter for validate our model. In this case you can see, for example, k-nearest neighbours have a great accuracy but the performance of the model is not great (all the rest of the indicators are so low)
- Not to split our data in Train and Test. If not done, our indicators will be great but in the real it will go down

I've split the data in train/test datasets (70% for train and 30% for test). Finally I've measured accuracy, precision, recall and f1 for myself and using *tester.py*.

## 6. Evaluation Metrics

**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"]**

In our final model we have this evaluation metrics and his explanation:

- Accuracy: 0,81613. This indicates our model is good fitting if a person is POI or not. It predicts correct over the 80% of the times
- Precision: 0,40011. This indicates our model have a medium performance on true results. That means 40% of the POIs predicted really are POIs
- Recall: 0,759. This indicates our model has a great performance on predict real POIs. The 75% of the real POIs are correctly predicted by our model
- F1: 0,52399. It's an indicator that relates Precision over Recall. It's like a weighted average of Precision and Recall. Then, our indicator is good because it's over 0.5