Identify Fraud from Enron Email

By Andre Luis Borges

Introduction

The Enron scandal, publicized in October 2001, eventually led to the bankruptcy of the Enron Corporation, an American energy company based in Houston, Texas, and the de facto dissolution of Arthur Andersen, which was one of the five largest audit and accountancy partnerships in the world. In addition to being the largest bankruptcy reorganization in American history at that time, Enron was cited as the biggest audit failure. [1]

In the resulting Federal investigation, a significant amount of typically confidential information entered into the public record, including tens of thousands of emails and detailed financial data for top executives. [2]

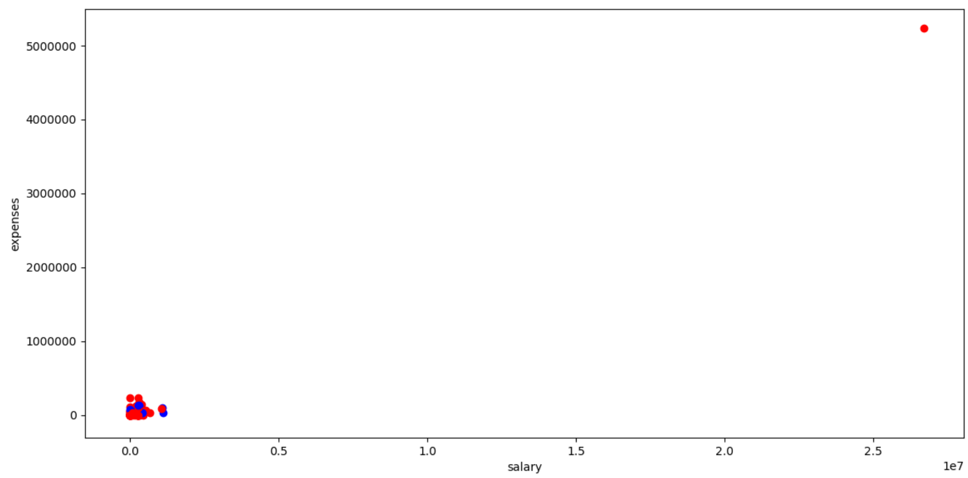
Who are the persons of interest in the fraud case? Based financial data available, we are going to build an identifier to find out the persons of interest.

Data analysis

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”]

My goal in this project is to build an identifier to find out the persons of interest. The dataset has 146 observations and the numbers of Persons of interest identified is 20, which means the allocation across classes is 0.14.

An outlier is an observation that lies an abnormal distance from other values in a random sample from a population. In a sense, this definition leaves it up to the analyst (or a consensus process) to decide what will be considered abnormal. Before abnormal observations can be singled out, it is necessary to characterize normal observations [4].



After plot the salary vs expense, I figured out an outlier value.
A commonly used rule says that a data point is an outlier if it is more than 1.5 * IQR above the third quartile or below the first quartile.[6]

IQR = Q3 – Q1, where Q3 = median of the n largest entries and Q1 = median of the n smallest entries.[7]

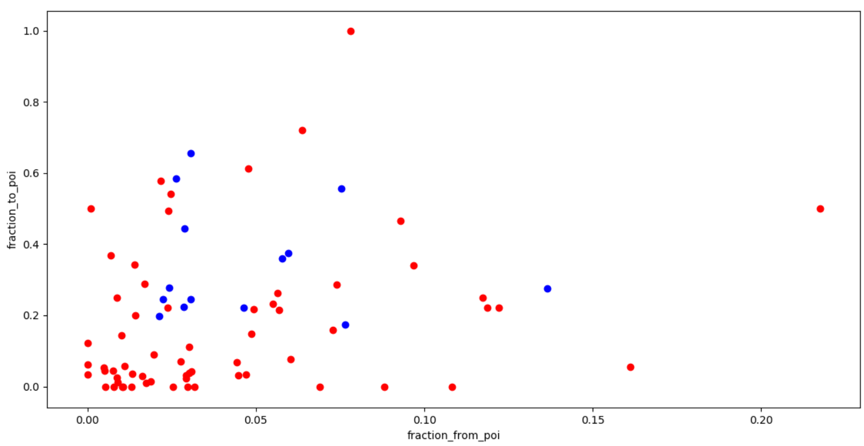
I count how many outliers there were for each data entry and print the top 14.

Name	Number of outliers
TOTAL	14
FREVERT MARK A	10
BELDEN TIMOTHY N	10
LAY KENNETH L	10
WHALLEY LAWRENCE G	8
LAVORATO JOHN J	8
SKILLING JEFFREY K	8
ALLEN PHILLIP K	6
BAXTER JOHN C	6
RICE KENNETH D	6
KITCHEN LOUISE	6
DELAINEY DAVID W	6
DERRICK JR. JAMES V	6
SHAPIRO RICHARD S	5

The first one is the TOTAL. According the official pdf documentation [8], Total is the sum of the columns and we can drop it from the dataset.

However, there are 3 person with 10 outlier feature. They are FREVERT MARK A, BELDEN TIMOTHY N and LAY KENNETH L. According the dataset, FREVERT MARK A is not a POI, and BELDEN TIMOTHY N and LAY KENNETH L both are. I don't think we should remove them from the dataset because they are important for our analyses.

Next plot is going to show the salary vs expense without the TOTAL row.

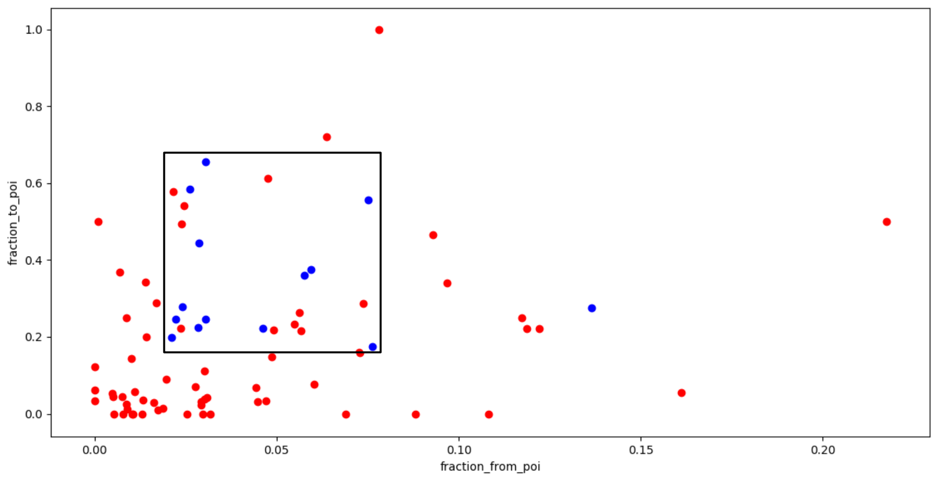


There is payments were made by Enron employees of business-related travel to 'THE TRAVEL AGENCY IN THE PARK' [8]. The data in the dataset is about Enrom employees and this company will be dropped from the dataset.

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 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"]

Feature scaling is a method used to standardize the range of independent variables or features of data[15]. For example, in the dataset, the HANNON KEVIN P have salary of 243,293.00 and the total stock value is 6,391,065.00. In order to use the features to identify the POI, I scale all the value from my feature list.

I create the Fraction to poi and Fraction from poi new features to verify if the number of e-mails send and received from POI could help us to identify a POI. Fraction to poi is the number emails sent to a POI divided by the total of email sent. Fraction from poi is the number of emails received from a POI divided by the total of e-mail received.



In the last chart, blue are the POI and red is the no POI. I think these new features are important. They will have a positive impact on precision metric because the most part of the POI are concentrated between 0.2 and 0.6 fraction_to_poi feature value and 0.03 and 0.08 fraction from POI and the most part of no POI are out of this zone what make it good for an classifier algorithm identify a POI. Therefore, I added these features to my feature list.

Feature selection is the process of selecting a subset of relevant features (variables, predictors) for use in model construction. Feature selection techniques are used for four reasons: simplification of models to make them easier to interpret, shorter training times, to avoid the curse of dimensionality, enhanced generalization by reducing overfitting[5]. There are 20 features in the dataset and I put away the features with missing value features higher than 50%, what reduce the amount of features to 10, plus 2 features I have created.

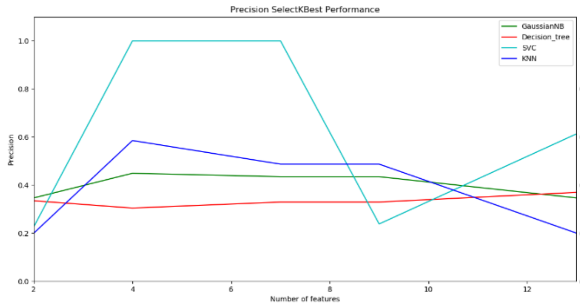
Feature	Number	Percent of missing value
salary	51	34.93 %
to_messages	60	41.10 %
deferral_payments	107	73.29 %
total_payments	21	14.38 %
exercised_stock_options	44	30.14 %
bonus	64	43.84 %
restricted_stock	36	24.66 %
restricted_stock_deferred	128	87.67 %
total_stock_value	20	13.70 %
director_fees	129	88.36 %
from_poi_to_this_person	60	41.10 %
loan_advances	142	97.26 %
from_messages	60	41.10 %
other	53	36.30 %
expenses	51	34.93 %
from_this_person_to_poi	60	41.10 %
deferred_income	97	66.44 %
shared_receipt_with_poi	60	41.10 %
email_address	35	23.97 %
long_term_incentive	80	54.79 %

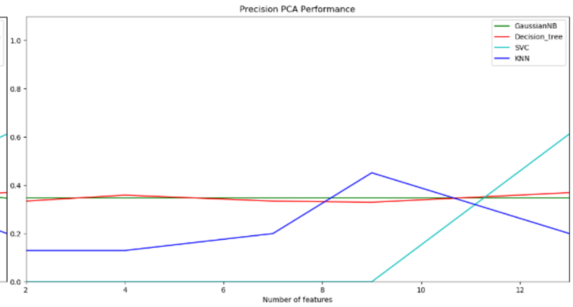
The last table showed the percent of missing value by feature.

In order to continue reducing the number of feature, I applied the SelectKBest and PCA algorithms with 2, 4, 7 and 9 features. SelectKBest selects the features according to the k highest score [9]. PCA is linear dimensionality reduction using Singular Value Decomposition of the data to project it to a lower dimensional space [12].

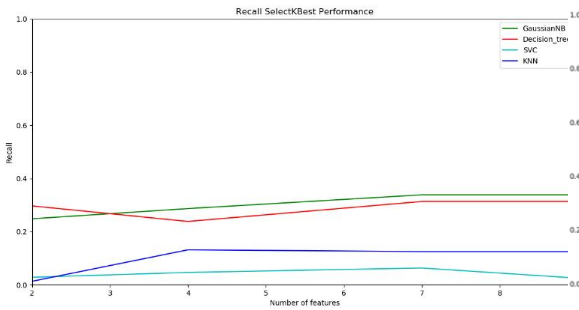
After apply the PCA and SelectKBest algorithm, I compared the precision and recall metrics from Guassian Naïve Bayes, decision tree, SVM and k nearest neighbor algorithm to find out the best one to identify the POI.

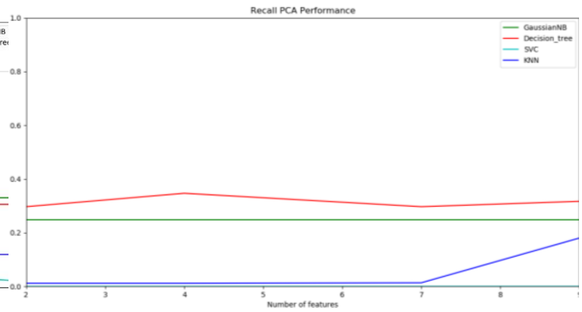
Analyzing the precision, three models showed a better performance using SelectKBest than PCA function.





The recall metric showed a performance behavior for SelectKBest and PCA function.





I've chosen the SelectKBest method because the recall and precision metrics for had the best performance to find out the POI working with GaussianNB model and 4 features.

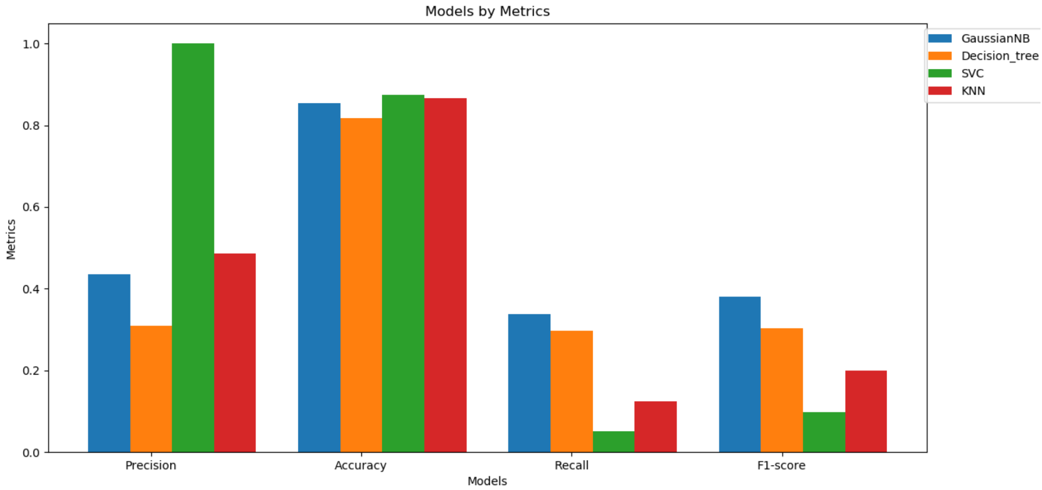
SelectKBest showed its best performance to find out the POI with 4 features: salary, exercised_stock_options, bonus and total_stock_value. The process to reduce the number of feature is important because the smaller is the number of features, faster is the process to train and predict.

My final list will contain the features salary, exercised_stock_options, bonus and total_stock_value.

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"]

I tried the Guassian Naïve Bayes, decision tree, SVM and K Nearest Neighbor algorithm.

In order to find out the best performance, I based my analysis on precision and recall metrics, however I collected data from accuracy, precision, recall and f1 score metrics. The next chart show the performance from the metrics by models.



All these algorithms were tuned. For precision metric, the best result was for SVC model. On the other hand, for recall metric, SVC was the worst result. The best average for precision and recall was the GaussianNB model.

This analysis make me believe I should use the GaussianNB algorithm because it were the best precision and recall average performance.

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? 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"]

In Machine Learning, tuning is the problem of choosing a set of optimal parameters values for a learning algorithm. [14]

Tuning the parameter value is important because an optimized parameters value can improve the algorithm to better predict or classify a data when you set it correctly, on the other hand, making the wrong setting, the prediction can get worse or overfitting.

For example, the Decision Tree classifier with salary, total_payments, exercised_stock_options, bonus, restricted_stock, total_stock_value, fraction_to_poi features had a better performance. The accuracy metric varied from 0.25 to 0.30 after tuning.

Due the difficulties to figure out the best parameters values, I used the GridSearchCV automation to make exhaustive search over specified parameters values to find the best estimator [11]. GridSearchCV was used in decision tree, SVC and KNeighborsClassifier algorithm. I tried to search the best parameters values for Decision Tree (min_samples_split and criterion), SVC (kernel, C, gamma), KNeighborsClassifier (n_neighbors, weights). Gaussian Naïve Bayes was not possible to tune because it does not have parameter to set.

During my tests, I figured out when I tuned the model, there were few improvements and the time spent to train the data is slower. For example, the time average spent for a decision tree model to be trained are 0.00088, and after I added the GridSearchCV, the average time rose to 0.1472, what means 167 time slower. Working with decision tree, after 9 training and prediction test with different number of features, just once there was improvement from default to the tuned algorithm.

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

Validation is the process of deciding whether the numerical results quantifying hypothesized relationships between variables, are acceptable as descriptions of the data [16]. The main mistake we can make is to use the same data for training and testing because you will overfitting the data.

In my case, I used two methods to split the data to train and test. The first one was the train_test_split method. Train_test_split split arrays or matrices into random train and test subsets[17]. I set 10% for testing and 90% for training. During this training and testing I used the SelectKBest and PCA function to find the best model, to tune the algorithm and to reduce the number of features. After that, I used the StratifiedShuffleSplit function to train and the test again with different data. StratifiedShuffleSplit returns stratified randomized folds. The folds are made by preserving the percentage of samples for each class[18]. Due the size of the dataset, using just the train_test_split method, the model was overfitting the data, which means the model I choose had good performance with known data and a bad performance with different training and test data.

Working with these two methods allowed me to choose my best algorithm, to reduce the number of features and to better understand the behavior of the models I was trying.

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"]

Follow the formulas for all metrics I showed during the project:

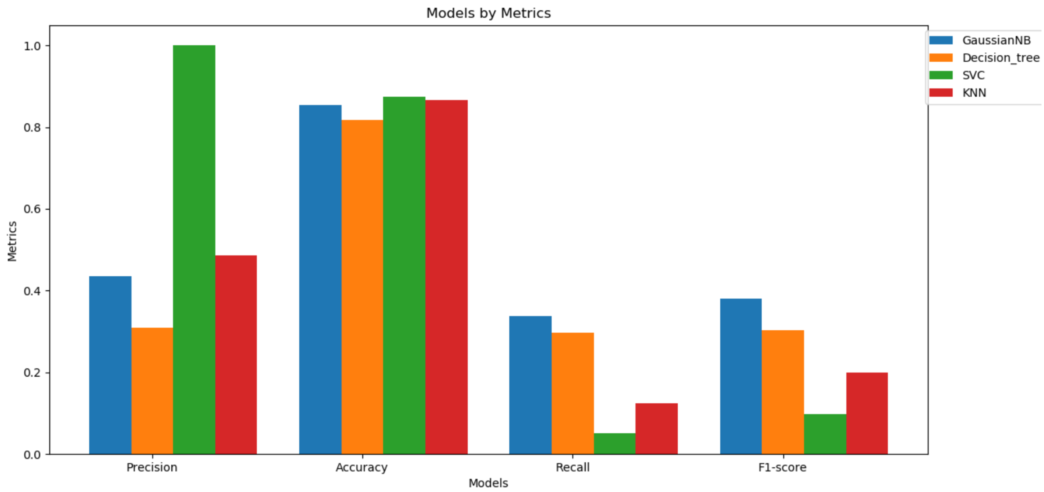
Accuracy is the number of items in a class labeled correctly divided by all the items in the class[2].

Precision is the number of true positive divided by the number of true positive plus false positive. (True positive / true positive + false positive) [2]

Recall is the number of true positive divided by the number of true positive plus false negative. (True positive / true positive + false negative) [2]

F1-score can be interpreted as a weighted average of the precision and recall, where an F1 score reaches its best value at 1 and worst score at 0. F1 = 2 * (precision * recall) / (precision + recall)[10]

The 2 evaluation metrics I used were the precision and the recall. Precision is the fraction of the relevant POI successfully identified. Recall is the fraction of the POI identification that are relevant for the query.



There is 14% of POI in the dataset. There is 84% of chance for an identifier to find a no POI, what means the accuracy metric will be higher because it is easier to identify a no POI, as we can see in the last chart for all the models. A higher precision performance means the model is correctly identifying the POI and NO POI labels, and the SVC models stood out at this point. A bad recall performance means there are many POI and NO POI label identified incorrectly (many false negative) and SVC model also stood out in this point.

We are looking for a model that identified the POI and NO POI label correctly with few mistakes, GaussianNB models had the closer performance from what I'm looking for, and this was the reason I choose the GaussianNB model instead of SVC or KNN model.

Reflection

During this project, I learned how to deal with imperfect dataset, validate a machine learning result using test data, evaluate the result using quantitative metrics, compare the performance of different algorithms, tune the algorithm to get the maximum performance and communicate the result.

I struggled on two moments. The first one was when I was trying to understand pipeline and how to use it to reduce the number of features and the second was when I was trying to reach a precision and recall performance higher that 0.3. The first one I could resolve after I read a book and watch again the last 3 module from the Machine Learning but the second I had to change the source code and follow the same idea from the tester.py routine to calculate the metric values. This routine change changed the model that I choose to identifier the POI labels, from KNN model with 2 features to GaussianNB with 4 features.

I figured out after I tuned an algorithm, I got better performance to predict. Because of this, I tried to tune all the algorithms I was working using GridSearchCV and this make my code slower. Working with this dataset, the code worked, however with a dataset larger than that, probably my code will take hours to be executed. I think, the best approach is to tune the algorithm after find the number of features and the best algorithm.

It is not easy to find the best numbers of features to work with. After I learn how to work with PCA and SelectKBest I figured out the importance of this function. When I use all the features I choose, the result was horrible, however after I reduce the number of features the result improved. I was surprised that four features was enough for my identifier.

I think it was amazing to work in this project and I learned a lot.

References

[1] - https://en.wikipedia.org/wiki/Enron_scandal

[2] - https://classroom.udacity.com/nanodegrees/nd115/

[4] - https://www.itl.nist.gov/div8988/handbook/prc/section1/prc16.htm

[5] - https://en.wikipedia.org/wiki/Feature_selection

[6] - https://www.khanacademy.org/math/statistics-probability/summarizing-quantitative-data/box-whisker-plots/a/identifying-outliers-iqr-rule

[7] - https://en.wikipedia.org/wiki/Interquartile_range

[8] - https://github.com/udacity/ud120-projects/blob/master/final_project/enron61702insiderpay.pdf

[9] - http://scikit-learn.org/stable/modules/generated/sklearn.feature_selection.SelectKBest.html#sklearn.feature_selection.SelectKBest

[10] - http://scikit-learn.org/stable/modules/generated/sklearn.metrics.f1_score.html

[11] - http://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html

[12] - http://scikit-learn.org/stable/modules/generated/sklearn.decomposition.PCA.html

[13] - https://en.wikipedia.org/wiki/Cross-validation_(statistics)#k-fold_cross-validation

[14] - https://en.wikipedia.org/wiki/Hyperparameter_optimization

[15] - https://en.wikipedia.org/wiki/Feature_scaling

[16] - https://towardsdatascience.com/cross-validation-in-machine-learning-72924a69872f

[17] - http://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html

[18] - http://scikit-learn.org/stable/modules/generated/sklearn.model_selection.StratifiedShuffleSplit.html