Identifying Fraud from Enron Email

Machine Learning Final Project

by Dirk Kalmbach

(September 2015)

***Executive summary**: Enron was on of the largest corporate frauds in US history. This report describes the implementation of an machine learning alghorithm to identify persons involved in the fraud by analyzing their email and financial information. It is the final project in the Udacity Machine Learning course. All code to perform the alghorithm can be found in the corresponding python file* `poi_id.py`.

1 Summary

About Enron

In 2000, Enron was one of the largest companies in the United States. By 2002, it had collapsed into bankruptcy due to widespread corporate fraud. In the resulting Federal investigation, there was a significant amount of typically confidential information entered into public record, including tens of thousands of emails and detailed financial data for top executives.

Goal of this project

The goal of this project is to investigate this dataset and to implement an machine learning algorithm which identifies whether a particular email was written by a person involved in the Enron fraud („person of interest" or POI) or not.[1] Machine Learning is an adequate method for this as it "*explores the study and construction of algorithms that can learn from and make predictions on data.*" (wikipedia)

Dataset

This dataset consists of personal **financial information** like salary, bonuses, loan advances, total payments as well as **email information** like how many emails from and to POIs and how many emails in total were sent to / received from that person.[2]

After identifying visually one outlier which probably resulted from an incorrect data transfer and eliminating it manually, the dataset contains 145 persons. 18 among them are POIs.

2 Features selection

The dataset contains 7 features representing email information and 13 features representing financial information. One feature (`other`) is not assignable. Another feature indicates whether the person is an POI or not. A table with all features and missing values can be found in the Appendix.

I started with some obvious features like salary and bonus, added successively other features and tried different machine learning algorithm with different tuning parameters and documented the performance in a spreadsheet. Although this strategy gave me some feeling about which features seems to be important and which not, I did not achieve a recall and precision score higher than 0.3.[3]

I then switched to the automatic feature detection tool SelectKBest (k=7) which suggested to keep the following features

- `bonus`
- `total_stock_value`
- `salary`
- `exercised_stock_options`
- `deferred_income`, and

---

1. We define a POI as a person who was either indicted, settled without admitting or testified in exchange for immunity for prosecution during the Enron trial. Find out more about Enron participants at *USATODAY.com (2015)*.

2. We extracted these information in class from „The Enron Email Corpus“ *(Cohen 2015)* which can be also investigated online at http://www.enron-mail.com/email/.

3. One of the requirements to pass the project.

- `long_term_incentive`, as well as `strength_of_email_conn_to_POI`,

The latter feature was constructed by hand and indicates the strength of connections between a person and a POI measured by all emails to/ received from a POI divided by all emails.

It was not necessary to scale features as Naive Bayes does not require this.

3 Alghorithm

I used the Gaussian Naive Bayes algorithm. I also implemented a Decision Tree, Random Forest and Adaboost but Naive Bayes performed best. The following table shows the recall and precision scores of these algorithms:

Algorithm	Parameters	Recall	Precision
Naive Bayes	*criterion='gini'*	*0.6*	*0.6*
Decision Tree	criterion="gini", min_samples_split=2	0.5	0.2
Random Forest	min_samples_split=11, n_estimators=8	0.4	0.4
Adaboost	algorithm='samme'	0.34	0.2

Table 1: Recall and Precision for different algorithm

4 Tuning

Parameter tuning (i.e.: finding the best combination of parameters of an algorithm) usually is an important step in machine learning as this often results in higher validation scores. On the other side, many machine learning algorithm have a lot of parameters which - especially when different algorithm are compared - often leads to many possible combinations, or like Domingos (2012) writes: "*Machine learning algorithms have lots of knobs, and success often comes from twiddling them a lot, so this is a real concern.*"

As Naive Bayes does not have any tunable parameters, this step was omitted.

Nevertheless, I also run a decision tree algorithm (among other algorithm) to see wether it beats the Naive Bayes or not. I tried different values for the `min_sample` parameter[4] and compared the results to the Naive Bayes scores.[5] But none of these beat the Naive Bayes algorithm so I kept the latter.

5 Validation

Validation means proofing or examining wether an algorithm works properly (i.e. predicting the expected outcome). In machine learning this can be done by splitting the dataset into a training and testing set. The former is used to select features and to develop and tune the algorithm, whereas the latter is used to validate the algorithm by comparing the predicted results with the data in the test set.

Validation is one of the most important parts in machine learning. Without validation it is possible that an algorithm memorizes all features, i.e. showing high performance scores only at the training set.

I deployed a 3-fold cross validation to validate the results.

I should also mention that the 3-Fold Cross Validation showed better results than the shuffle split cross validation implemented in `tester.py`. Testing the algorithm with the latter resulted in a precision score of 0.489 and a recall score of 0.367 for the Naive Bayes.

6 Evaluation

As Table 1 shows both precision and recall were 0.6, accuracy was 0.85 and the f1-score was

4. How many persons in a node are required to split a node.

5. Other parameters are the splitting criteria (`gini` vs `entropy`), the `max_depth` to limit the size of the tree and therefore the processing time, `min_samples_leaf`, `min_weight_fraction_leaf` and `max_leaf_nodes`.

Accuracy is the fraction of correctly identified POIs and Non-POIs from all persons, i.e.: the probability that a randomly selected email was classified correctly either as written by a POI or a Non-POI.

Recall is the fraction of correctly identified POIs among all (real) POIs, i.e. the probability that a randomly selected email written by an POI. That means that, nearly every time a POI shows up in my test set, the likelihood is 60% that I am able to identify him or her.

Precision is the fraction of correctly identified POIs among all (correct or wrong) identified POIs, i.e. the probability that a randomly selected email among all as POI-identified emails was actually written by an POI.

Accuracy, precision, recall, and f1-score all range from 0 to 1, with 1 being optimal.

That means that whenever a POI gets flagged in my test set, the probability is 60% that it is a real POI and not a false alarm.

7 Reflection

The alghorithm I implemented used only a few features. Better results could be achieved by digging deeper into the email information, e.g. using text learning to analyze the content as well as analyzing the time structure of the email communication. However, I did not implement any natural language processing alghorithms as the goal of this project was to achieve a precision and recall score above 0.3.

Some minor critics about the quality of the original datasets are that the emails does not include attachments, and that some messages have been deleted "as part of a redaction effort due to requests from affected employees" (Cohen 2015).

8 References

Cohen, William. W. „Enron Email Dataset.“ May 8, 2015. Accessed September 13, 2015. http://www.cs.cmu.edu/~./enron/.

Domingos, Pedro. A few useful things to know about machine learning. Commun. ACM. 55 (10):78–87, 2012.

USATODAY.com, „A look at those involved in the Enron scandal.“ December 28, 2005, Accessed September 13, 2015. http://usatoday30.usatoday.com/money/industries/energy/2005-12-28-enron-participants_x.htm

Wikipedia contributors, "Machine learning," Wikipedia, The Free Encyclopedia, Accessed October 14, 2015. https://en.wikipedia.org/w/index.php?title=Machine_learning&oldid=685498675.

9 Appendix

List of all features

		missing values		
	name	POI	Non-POI	*total*
1	salary	50	0	*50*
2	bonus	62	0	*62*
3	total stock value	20	0	*20*
4	exercised_stock_options	38	0	*38*
5	deferred income	90	0	*90*
6	exercised_stock_options	38	0	*38*
7	long_term_incentive	74	0	*74*
8	director_fees	111	0	*111*
9	total_payments	21	0	*21*
10	loan_advances	125	0	*125*
11	expenses	51	0	*51*
12	deferral_payments	94	0	*94*
13	restricted_stock_deferred	110	0	*110*
14	from_messages	56	0	*56*
15	to_messages	56	0	*56*
16	from_this_person_to_poi	56	0	*56*

17	from_poi_to_this_person	56	0	*56*
18	shared_receipt_with_poi	56	0	*56*
19	shared_receipt_with_poi	56	0	*56*
20	email_adress	35	0	*35*
21	other	53	0	*53*

Table 2: Variables and missing values

I hereby confirm that this submission is my work. I have cited above the origins of any parts of the submission that were taken from Websites, books, forums, blog posts, github repositories, etc.