Udacity IntroML Project

Tianzhixi Yin

January 2016

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 this project is to identify persons of interest in the Enron fraud case. Machine learning could be helpful in trying to accomplish it because machine learning might be able to find the underlying relationship between the features we have and whether a person is guilty or not. Enron fraud was one of the biggest company frauds in the United States, after its investigation, a large amount of confidential data were released.

   Our dataset contains Enron email and financial data. There are records like salary, bonus, number of emails from a person to POIs etc. There are totally 20 ready-made features we can use for prediction (however, email address is not useful so I did not include it). Each person is labelled as a POI or non-POI. We have 146 data points, but one of them is clearly an outlier, which has the key "TOTAL". I removed it from the dataset because it is a spreadsheet quirk. Among the 145 persons, 18 of them are labelled POI. There are several features with many missing values, for example salary, bonus, total payments. Since these are missing values, I do not think replacing them with 0s is appropriate.

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", "properly scale features", "intelligently select feature"]*

   The features I ended up using are: salary, deferral payments, total payments, bonus, deferred income, total stock value, expenses, exercised stock options, other, long term incentive, restricted stock, to messages, from poi to this person, from messages, from this person to poi, and shared receipt with poi. Totally 16 features. I selected them based on feature importances by Random Forest. I removed the features considered not important by feature importances: loan advances, restricted stock deferred, and director fees. Although from poi to this person is also considered not important, this is possibly because of the new feature (from to poi) I made. Because my average recall is not good enough, I removed the new feature and used from poi to this person which gave me a better average recall.

I did not do scaling because both Random Forest and AdaBoost are tree-based methods, they do not need feature scaling.

The new feature that I made is the sum of from poi to this person and from this person to poi. My reason for doing this is that the new feature would be a better summary to describe the connection of a person to the POIs. The new feature has a importance of 0.04701573 evaluated by Random Forest. However, in the end I did not use my new feature.

The feature importances given by Random Forest:

Feature	**Importance**
salary	0.05679454
deferral payments	0.00733264
total payments	0.06545864
loan advances	0
bonus	0.06467299
restricted stock deferred	0
deferred income	0.1176804
total stock value	0.08255272
expenses	0.06882014
exercised stock options	0.04388871
other	0.11881023
long term incentive	0.02612813
restricted stock	0.0650501
director fees	0
to messages	0.05349905
from poi to this person	0
from messages	0.02500633
from this person to poi	0.07923048
shared receipt with poi	0.07805916
from to poi	0.04701573

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 ended up using the AdaBoost algorithm. I also tried Random Forest. Before parameter tuning, Random Forest has a high precision but very low recall(less than 0.2), while the result by AdaBoost is more balanced, precision greater than 0.3 and recall less than but close to 0.3. Therefore I picked AdaBoost for further parameter tuning. (I also tried tuning Random Forest, however the recall is always too low.)

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 like making a special algorithm best suited for the prediction task of the data we have. If we do not tune the parameters, every parameter value is by default. These predefined values are set by programmers who have never seen our data and for very general cases. Tuning the parameters can make the algorithm different but better for the data we have.

I first did a GridSearchCV for three parameters in AdaBoost: algorithm, learning rate, and number of estimators. I also set the scoring function to be 'f1'. However, although the resulting parameter values give me the highest 'f1' value, the recall is still below 0.3 (though very close). Therefore I decided to manually tune the parameters to see if I can find a case where both precision and recall are greater than 0.3. Finally I found one case, with algorithm being 'SAMME', learning rate being 2 and number of estimators being 15.

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 a way to access whether the built algorithm is actually good at predicting future data. Normally we separate the data into training set and testing set. If we train the algorithm with all the data we have, it would result in over-fitting: the algorithm is very good at predicting the data we have but not very good at predicting the cases we have not seen yet. How to choose training data is also important, a classic mistake you can make is selecting training data sequentially that most training data belong to the same category. Therefore it is best to randomly pick training data from the whole dataset.

   I first validated my analysis by separating 30% of the data for testing to see whether the algorithms I choose are any good. The main validation work is done by the stratified shuffle split cross validation in the tester code, where more reliable prediction results are generated.

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"]*

   My average precision is 0.31569, and average recall is 0.32500. Precision is also called positive predictive value, meaning among all the persons that we predict to be POIs, what is the percentage that they are truly POIs. Recall is also called true positive, meaning if a person is a POI, the probability that the algorithm can identify that person as a POI.

   I would not say my algorithm is great at predicting POI, but I think I have tuned the algorithm close to the best it can do, to have a balanced precision and recall. Perhaps the data we have for now are not enough to build a very good algorithm, especially considering the limited number of POIs we have in our dataset.