Enron Emails and Financial Data Analyze

By Xinlin Feng

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, 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.

In this final project, I will use machine learning to find POI based on financial and email data revealed in Enron scandal, as well as a labeled list of individuals.

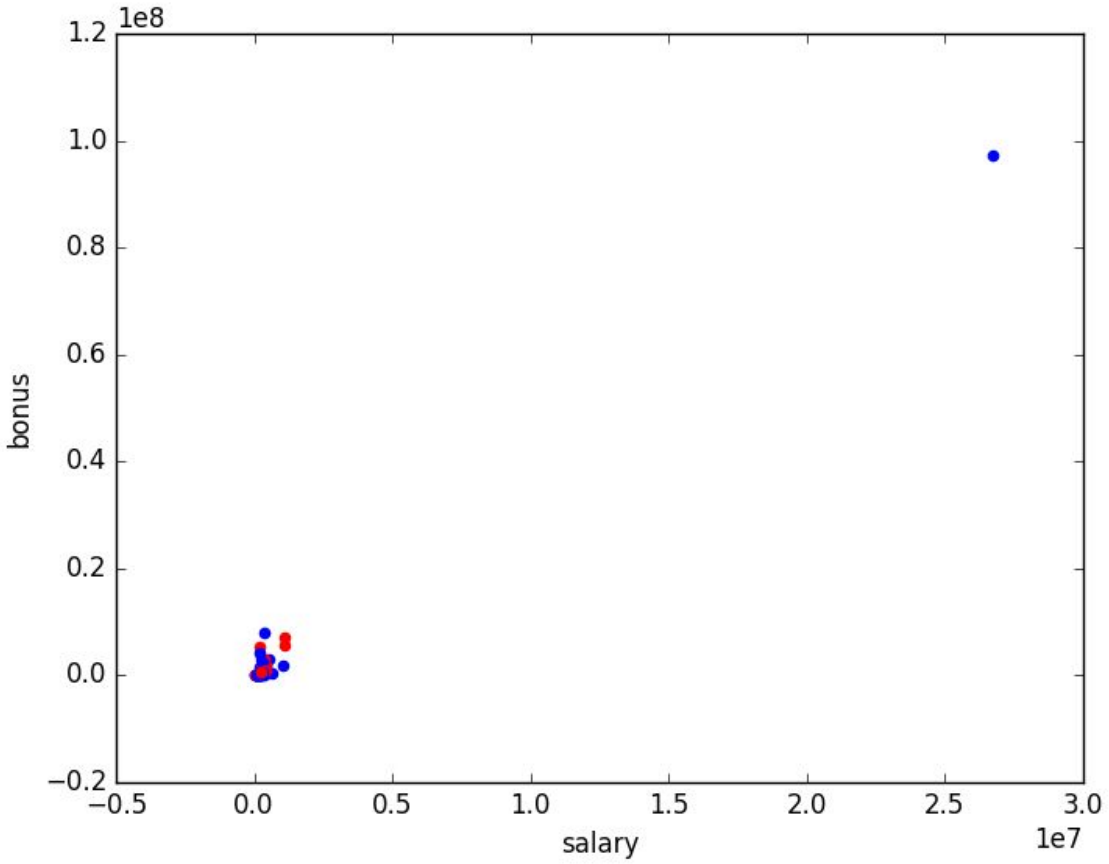
1.Goal and outlier of this project:

The goal of this project is to use machine learning skills to identify POI based on features in the Enron dataset. Since this data is widely used for various machine learning and although it has been labeled already, the value is the potential application for similar cases in other companies or spam filtering application.
After read file, here is the summary of data set:
There are 146 entries, with 18 POIs and 128 non-POIs; each of them has 21 features.
As for missing value, here is the summary of each feature:

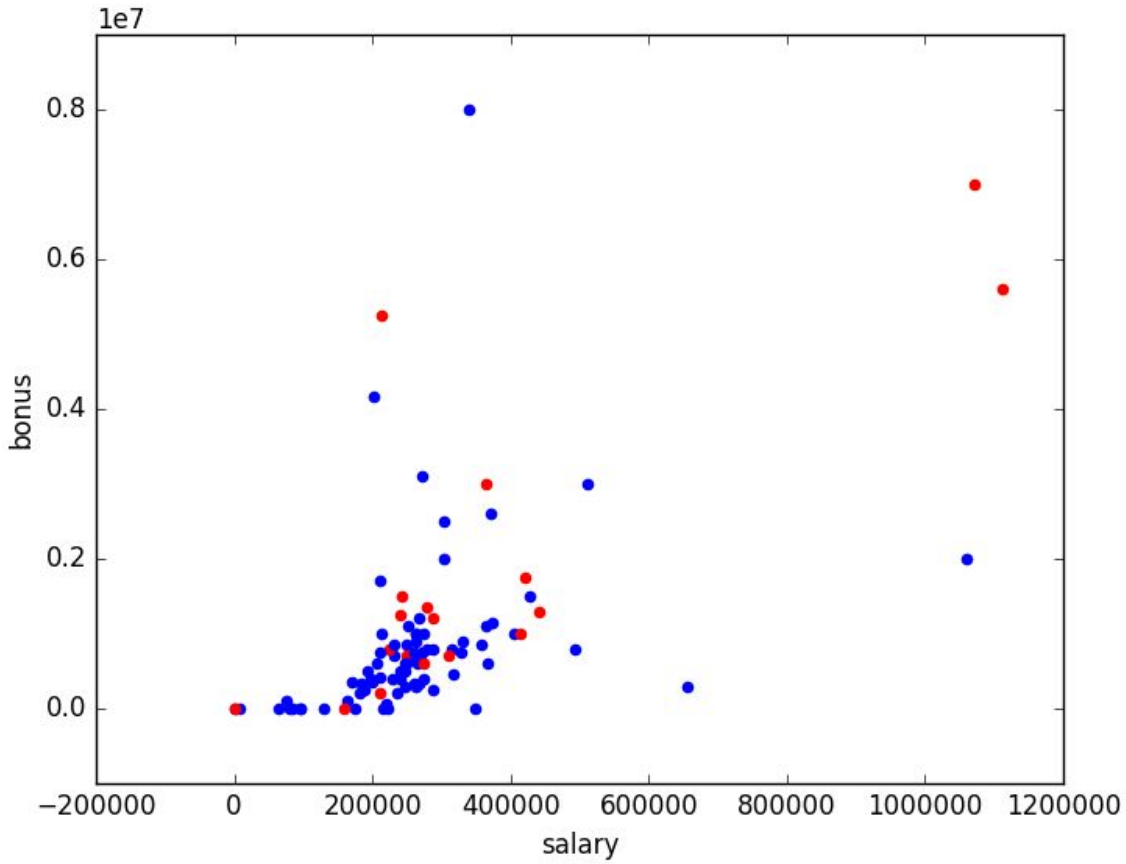
Ming value Table	
Items	**Missing Quantity**
Salary	51
To messages	60
Deferral payments	107
Total payments	21
Exercised stock options	44
Bonus	64
Restricted stock	36
Shared receipt with POI	60
Restricted stock deferred	128
Total stock value	20
Expenses	51
Loan advances	142
From messages	60
other	53

From this person to POI	60
POI	0
Director fee	129
Deferred income	97
Long term incentive	80
Email address	35
From POI to this person	60

The outlier is labeled with “TOTAL”, according to data plot and filter. We removed it,



and I removed 2 other entries: one is THE TRAVEL AGENCY IN THE PARK, it isn't represent an individual, another is LOCKHART EUGENE E, who contains only NaN values.
After that, I plot again:



Then I checked outliers with salary > 1000000 and bonus > 0.5, all results are normal persons, so I keep them in data still.

2. Features Selection

What features did you end up using in your POI identifier, and what selection process did you use to pick them?
Besides 21 features, I have created 2 features additionally by using existing features which I think should help to find more accurate results.

- `Fraction_poi_communication`: fraction of emails related to POIs. This new feature helps to determine what proportion of messages are communicated between POIs compared to overall messages.
- `total_wealth`: salary, bonus, total stock value, exercised stock options. It helps to find the sum of all values which is related to person wealth.

I will use sklearn to help me fit the data set. Especially , I will choose `SelectKBest` to select top 10 influential features.

Feature	**Score**
Exercised stock options	24.815079733218194

Total stock value	24.182898678566879
Bonus	20.792252047181535
Salary	18.289684043404513
Total wealth	15.369276864584535
Deferred income	11.458476579280369
Long term incentive	9.9221860131898225
Restricted stock	9.2128106219771002
Total payments	8.7727777300916792
Shared Receipt with POI	8.589420731682381

As the table above, all 10 features are related to finance factors rather communication factors. Unsurprisingly, the feature 'total wealth' is listed on table. The reason of choosing 10 features to display is based on 2 reasons:
First, too few features might cause the bias while too many features might cause overfitting. In this sense, since there are 21 + 2 created features, select top 10 influential features is reasonable.
Second, during tuning parameters, I found the Logistic Regression has the best performance, and the result varied according to features selected:

Logistic Regression Features Selected				
Features Selected	**Precision**	**Recall**	**F1-score**	**F2-score**
7	0.45138	0.40850	0.42887	0.41641
8	0.44211	0.44100	0.44155	0.44122
9	0.43741	0.44900	0.44313	0.44663
10	0.37708	0.43100	0.40224	0.41902
11	0.37731	0.42900	0.40150	0.41756

We see that best F1 score are in 9 Features selected, so that is another reason for listing top 10 influential features above.

Did you have to do any scaling?
In sklearn, I used `StandardScaler()` function to standardize the features in the dataset.
Why or why not?
Because there might be some abnormal data in dataset which will influence the complete data distribution. Further, some algorithm such as SVM required a scaling process.

3. Algorithm and compare

What algorithm did you end up using?
I used Logistic Regression
What other one(s) did you try?
I also tried Gaussian Naive Bayes, SVM, Decision Tree and Random Forest.
How did model performance differ between algorithms?
Logistic Regression gives overall best performance using evaluation matrices among all others, detailed discussion is in part 6.

4. Tune parameters

What does it mean to tune the parameters of an algorithm, and what can happen if you don't do this well?
Tune parameters in an algorithm, according to my understanding, is to give the algorithm several sets of candidates for different parameters and compare those candidates' performance and find the best parameters combination choice among those candidates.
Some of the algorithm are pretty parameter sensitive, if you don't handle parameters well, you are away from the best performance.

How did you tune the parameters of your particular algorithm?
I used `Pipeline` function, where parameters tuning is done using `GridSearchCV` and `StratifiedShuffleSplit`.Following parameters are used to tune an algorithm:

1. SelectKBest: k(for all except Naive Bayes)
2. Principal Components Analysis (PCA): n_components, whiten
3. Logistic Regression: C, tol, penalty, random_state
4. Support Vector Classifier: C, gamma, kernel
5. Decision Tree: min_samples_split, min_samples_leaf, criterion, max_depth, random_state
6. Random Forest: n_estimators, max_depth, random_state

PCA is for transforming input features into principal components, It helps in dimensionality reduction, reduce noise and make algorithms to work better with fewer inputs. Maximum PCs in the data is equal to number of features.

5. Validation

What is validation, and what's a classic mistake you can make if you do it wrong?
Validation means that a machine learning algorithm generalizes well, and to prevent overfitting.
if you don't handle it well, say you considered too many factors and designed a too complexed algorithm, overfitting will occur, in this case, it is hard to use this algorithm to predict new case.
How did you validate your analysis?
I use `Stratified shufflesplit` from sklearn to provide train/test indices to split data in train test sets.

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.
In the final comparison, I evaluated Precision, Recall, and F1-score, F2-score.

Precision is defined as TP / (TP + FP), a high precision means POIs identified by an algorithm tended to be correct.
Recall is defined as TP /(TP + FN), a high recall means if there are POIs in the dataset, an algorithm has good chance to identify them.
F1-score is defined as (2*recall*precision) / (recall + precision).
F2-score is defined as (5*recall*precision) / (4*precision + recall).

Algorithm	Precision	Recall	F1-score	F2-score
Logistic Regression	0.43741	0.44900	0.44313	0.44663
Gaussian Naive Bayes	0.37401	0.32950	0.35035	0.33753
Random Forest	0.46379	0.16650	0.24503	0.19098
Decision Tree	0.25762	0.21550	0.23469	0.22279
SVM	0.35698	0.08050	0.23089	0.09525

In table above, Logistic Regression gives the highest F score. That's the reason why I use Logistic Regression finally.

7.File illustration

poi_id.py : This is the main part of this project.
enron.py : This is the helper class, including select top k, print tune information.

8.Reference

Udacity Data Analyst Nanodegree
http://scikit-learn.org/stable/modules/pipeline.html
http://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html
http://scikit-learn.org/0.17/modules/generated/sklearn.cross_validation.StratifiedShuffleSplit.html
https://clusteval.sdu.dk/1/clustering_quality_measures/5