Enron Submission Free-Response Questions

> A critical part of machine learning is making sense of your analysis process and communicating it to others. The questions below will help us understand your decision-making process and allow us to give feedback on your project. Please answer each question; your answers should be about 1-2 paragraphs per question.
>
> When your evaluator looks at your responses, he or she will use a specific list of rubric items to assess your answers. Here is the link to that rubric: Link to the rubric.

Q1：

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”]

A1：

The goal of this project is to classify POIs. We have 145 data points with 21 features, including 18 POIs and 127 no-POIs. There is an obvious class imbalance phenomenon.

59 data points missed all email features except email address (email address is not so useful here). I remove three outliers, 'TOTAL' and 'THE TRAVEL AGENCY IN THE PARK' don't look like a person's name. Person named 'LOCKHART EUGENE E' has all its value missing, which is useless to train model.

Finally, I choose Adaptive Boosting algorithm to build my classifier model.

Q2:

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”]

A2:

feature selection

I used SelectKBest method of Sklearn to pick up four features ('exercised_stock_options', 'loan_advances', 'total_stock_value', 'bonus') . The fifth feature salary only get 3.05 scores , much less than the former four . I also tuning parameter k in GridSerachCV, which shows using top four features can have the best metric defined by myself. It takes both recall and precision into account. See more detailes in next part **evaluation metric**.

feature	score (two decimals)	rank
exercised_stock_options	6.85	1
loan_advances	6.69	2
total_stock_value	5.48	3
bonus	5.12	4
salary	3.05	5

evaluation metric

As precision and recall of final classifier are both supposed no less than 0.3, I define an evaluation metric that take both into account. This metric returns f1_score score, if precision and recall both over 0.3; returns 0.3 if one is less than 0.3; returns 0 if neiher of them over 0.3.

```
# fucntion: define an evaluation metric
# in order to get more performant model whose recall and precision are both over
0.3
def scorer_r_p(estimator, features_test, labels_test):
    labels_pred = estimator.predict(features_test)
    pre= precision_score(labels_test, labels_pred, average='micro')
    rec = recall_score(labels_test, labels_pred, average='micro')
    if pre>0.3 and rec>0.3:
        return f1_score(labels_test, labels_pred, average='macro')
    elif  pre>0.3 and rec<0.3:
        return 0.3
    elif rec >0.3 and pre<0.3:
        return 0.3
    return 0
```

scale

Before feature selection, I scaled features to a range from 0 to 1. We will compute chi-squared value in feature selection, which only can be computed for non-negative values.

new features

I add three new features:

'email_features_miss' is a dummy variable that indicates whether this data point miss email features (yes: 1; no: 0). The feature_format function will transformed 'NAN' to 0, but 'NAN' of email features is not equal to the value 0. I think that missing value could possibly be a new feature.

'poi_rate_to_messages' compute the proportion of the number of messages sent to poi in the total number of messages sent ($poi\_rate\_to\_messages = \frac{from\_this\_person\_to\_poi}{to\_messages}$).

And 'poi_rate_from_messages' compute the proportion of the number of messages received from poi in the total number of messages received ($poi\_rate\_from\_messages = \frac{from\_poi\_to\_this_{p}erson}{from\_messages}$).

When the number of messages of two person is very different, the percentage is more meaningful than the absolute number.

The left part of the below picture shows the features' scores for the original feature set and the right part shows the features' scores added with the new features.

The rankings of 'email_features_miss', 'poi_rate_from_messages' and 'poi_rate_to_messages' are 11th, 9th and 15th respectively. The median of the rankings is 1.55 scores and the mean is 2.19 scores. Though the scores of 'email_features_miss' and 'poi_rate_from_messages' are higher then the median, the scores of the three features are all less then the mean. Due to the low scores, they won't be used in the final model.

```
Feature ranking:
1. 'exercised_stock_options' score: 6.845509
2. 'loan_advances' score: 6.688782
3. 'total_stock_value' score: 5.476610
4. 'bonus' score: 5.120754
5. 'salary' score: 3.052787
6. 'total_payments' score: 2.784779
7. 'long_term_incentive' score: 2.538485
8. 'shared_receipt_with_poi' score: 2.432220
9. 'other' score: 1.715951
10. 'director_fees' score: 1.501131
11. 'expenses' score: 1.486103
12. 'from_poi_to_this_person' score: 1.370059
13. 'from_this_person_to_poi' score: 1.000808
14. 'restricted_stock' score: 0.589535
15. 'to_messages' score: 0.436398
16. 'deferred_income' score: 0.340099
17. 'from_messages' score: 0.068739
18. 'deferral_payments' score: 0.060697
19. 'restricted_stock_deferred' score: 0.003507
```

```
Feature ranking:
1. 'exercised_stock_options' score: 6.845509
2. 'loan_advances' score: 6.688782
3. 'total_stock_value' score: 5.476610
4. 'bonus' score: 5.120754
5. 'salary' score: 3.052787
6. 'total_payments' score: 2.784779
7. 'long_term_incentive' score: 2.538485
8. 'shared_receipt_with_poi' score: 2.432220
9. 'poi_rate_from_messages' score: 1.785305
10. 'other' score: 1.715951
11. 'email_features_miss' score: 1.607142
12. 'director_fees' score: 1.501131
13. 'expenses' score: 1.486103
14. 'from_poi_to_this_person' score: 1.370059
15. 'poi_rate_to_messages' score: 1.269784
16. 'from_this_person_to_poi' score: 1.000808
17. 'restricted_stock' score: 0.589535
18. 'to_messages' score: 0.436398
19. 'deferred_income' score: 0.340099
20. 'from_messages' score: 0.068739
21. 'deferral_payments' score: 0.060697
22. 'restricted_stock_deferred' score: 0.003507
```

Q3:

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”]

A3:

I ended up using Adaptive Boosting and I tried DecisionTree, Naive Bayes. Adaptive Boosting got the highest accuracy and precision. And the recall of Adaptive Boosting is quite close to 0.3.

The other two algorithms did not perform so well. Although the recall of Naive Bayes is the highest, but the accuracy and the precision are both the lowest. The precision and the recall of Decision Tree is far less than 0.3.

	Accuracy	**Precision**	**Recall**	**F1**
DecisionTree	0.79540	0.23760	0.24200	0.23978
Naive Bayes	0.74607	0.23467	0.40000	0.29580
Adaptive Boosting	0.83660	0.35536	0.27700	0.31132

Q4:

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? [relevant rubric item: “tune the algorithm”]

A4:

Tuning the parameters could improve the performance of an algorithm. If I don't do this well, the performance of the model could be very poor.

I use the combination of GridSearchCV and Pipeline to tune parameters. The tuned parameters are k (number of selected features) and score function in feature selection process, iteration times , learning rate, base estimator (min_samples_split of DecisionTree) of Adaptive Boosting algorithm.

Following are relative codes:

```
# function: build model and tuning parameters
def train_classifier(features,labels,n=50):
    # Stratified ShuffleSplit cross-validator
    sss = StratifiedShuffleSplit(labels, n_iter=n, test_size = 0.2,
random_state=42)
    # Pipeline of transforms with a final estimator.
    pipe = Pipeline([('scaler',MinMaxScaler(feature_range=(0,1))),
                     ('selector', SelectKBest()),
                       ('classifier', AdaBoostClassifier(random_state=32))])
    # Specify a parameter grid.
    # The tuned parameters are k (number of selected features) and score function
in feature selection process,iteration times , learning rate, base estimator
(min_samples_split of DecisionTree) of Adaptive Boosting algorithm.
    param_grid = dict( selector__score_func=[chi2,f_classif]
         ,selector__k=range(4,11)
         ,classifier__n_estimators=[50,80,100]
         ,classifier__learning_rate=[0.5,1,1.5,2]
         ,classifier__base_estimator=[DecisionTreeClassifier(min_samples_split=2)

     ,DecisionTreeClassifier(min_samples_split=3)
                                    ,DecisionTreeClassifier(min_samples_split=4)])
    # Exhaustive search over specified parameter values for AdaBoostClassifier.
    clf_grid = GridSearchCV(pipe, param_grid, scoring=scorer_r_p, cv=sss,
n_jobs=-1)
    clf_grid.fit(features, labels)
    clf = clf_grid.best_estimator_
    return clf
```

Q5:

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"]

A5:

Validation is a method to estimate how well is the trained model.

Without validation, small training errors will overestimate the performace of the model. If I use much data for validation, the model could be underfitting due to lacking enough training data.

Another problem is that when dealing with small imbalanced datasets (like 18 pois vs 127 no-pois) with corss-validation, it is possible that some folds contain almost none (or even none!) instances of the minority class.

Here, I use StratifiedShuffleSplit method in SKlearn, which returns stratified randomized folds. The folds are made by preserving the percentage of samples for each class, which can fix the above problem and help to estimate the performace of the model more precisely.

Q6:

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"]

A6:

The final classifier model get accuracy 0.85, precision 0.43, recall 0.38.

Accuracy is the fraction of total predictions that are predicted correctly.

$$\mathbf{Accuracy} = \frac{\mathit{True\ pois} + \mathit{True\ no\_pois}}{\mathit{Total\ predictions}}$$

Precision is the fraction of total predicted poi that are predicted correctly.

$$\mathbf{Precision} = \frac{\mathit{True\ pois}}{\mathit{Total\ predicted\ pois}}$$

Recall is the fraction of total pois that are predicted correctly.

$$\mathbf{Recall} = \frac{\mathit{True\ pois}}{\mathit{Total\ pois}}$$