EE660 Project

HR Analytics: Employee Attrition Classification

Manqi Wang, manqiwan@usc.edu

December 3th, 2017

Instructor: Professor B. Keith Jenkins

1. Abstract

The project aims to predict if an employee will leave the company or not using two datasets. Dataset A is provided by IBM and dataset B is provided by another company. They have different number of samples and use unlike features. To solve the attrition classification problem, I use three machine learning methods, i.e. Logistic Regression, Random Forest and Adaboost. In each method, I use cross-validation to choose the best parameters and use the best parameters to predict the attrition of unseen employees. Logistic regression performs best on dataset A while random forest performs best on dataset B. I get a good predictive score on dataset B but a 'bad' score on dataset A. By comparing the features and predictive scores of two datasets, I give some conjecture about the results.

2. Problem Statement and Goals

"You don't build a business. You build people, and people build the business." - Zig Ziglar

Employees are important to companies since they are the ones who do the work and shape the company's culture. When an employee you have invested so much time leaves, it can lead the company to huge time and monetary losses to hire someone else. The goal of this project is to create a model that can predict if a certain employee will leave the company or not. Therefore, the company could create or improve their retention strategies on targeted employees.

This is a binary classification problem and I'm going to predict if an employee will leave or not. It is a difficult problem since 1) it's inherently complicated because we need to treat employees as samples and predict their thought; 2) it's hard to extract features that affect employee's attrition, so the features might cause different results.

3. Prior and Related Work – None

4. Project Formulation and Setup

After analyzing the problem statement and goals, I decide to implement three algorithms to do the classification:

4.1 Logistic Regression

Logistic regression is a simple kind of discriminative model which is the model of the form:

$$p(y|X, w) = \mathrm{Ber}(y|\mathrm{sigm}(w^T X))$$

Here, I prefer MAP estimation for logistic regression to computing the MLE.

Parameters to tune for logistic regression:

Table 1 Parameters within logistic regression classifier

C	0.01, 0.1, 1, 10, 100
penalty	'l1', 'l2'

where, 'C' is the inverse of regularization strength;

'penalty' specify the norm used in the penalization.

Set other parameters as default.

4.2 Random Forest Classification

Random forest is a meta estimator that fits a number of decision tree classifiers on various sub-samples of the dataset and averaging to improve the predictive accuracy and control over-fitting.

The sub-sample size is always the same as the original input sample size but the samples are drawn with replacement.

Parameters to tune for random forest classifier:

Table 2 Parameters within random forest classifier

max_depth ()	6, 8, 12, 16, 20, 24, 28, 32
n_estimators	50, 100, 200, 400, 800

where, 'max_depth()' is the maximum depth of the tree.

'n_estimators' is the number of trees in the forest.

Set other parameters as default.

Figure 1 is the framework of random forest.

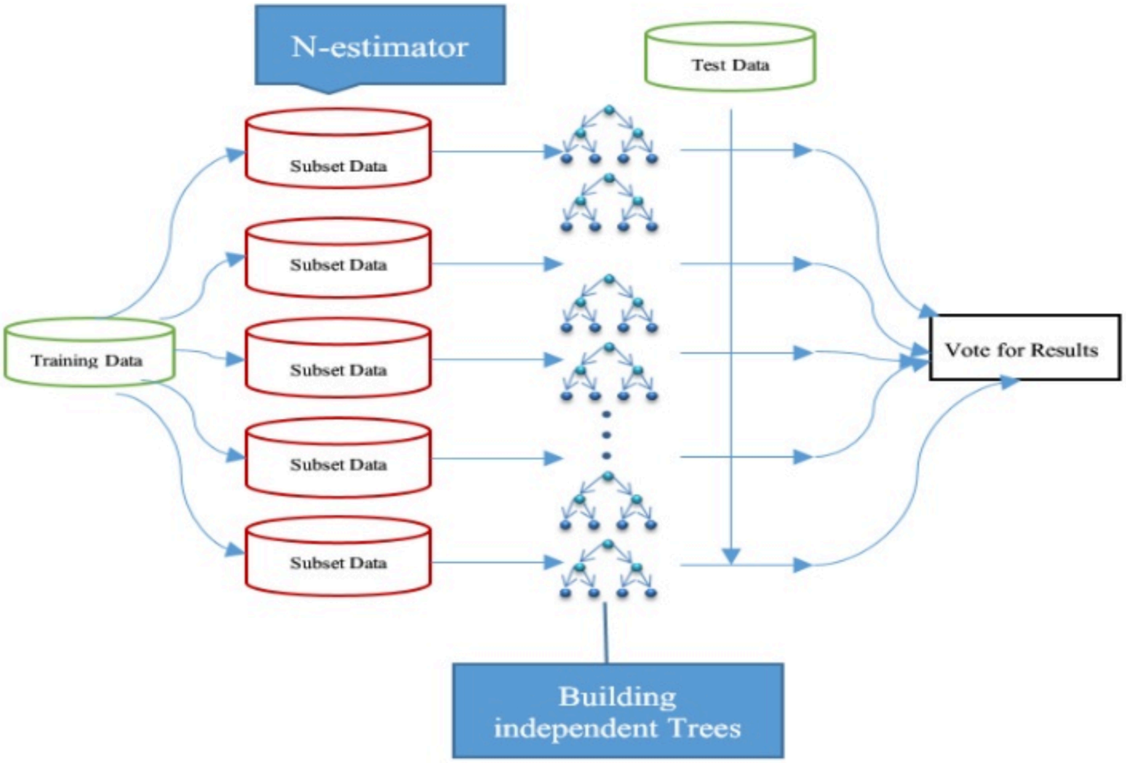


Figure 1 The Framework of Random Forest

4.3 Adaboost Classification

Adaboost classifier is a meta-estimator that begins by fitting a classifier on the original dataset and then fits additional copies of the classifier on the same dataset but where the weights of incorrectly instances are adjusted such that subsequent classifiers focus more on difficult cases.

Parameters to tune for Adaboost classifier:

Table 2 Parameters within Adaboost classifier

learning_rate	0.6, 0.8, 1, 1.2
n_estimators	50, 100, 200, 400, 800

where, 'learning_rate' shrinks the contribution of each classifier by learning_rate;

'n_estimators' is the maximum number of estimators at which boosting is terminated.

Set other parameters as default.

5. Methodology

The procedure of this project includes preprocessing, training and evaluation.

5.1 Preprocessing

Step 1: Check if there is any missing data in dataset.
Step 2: Check shape, types of features and statistical overview of dataset.
Step 3: Using **one hot encoding** for categorical features.
Step 4: Split the dataset into **training and test data**
Step 5: **Standardize** the numeric features in training set and test set.

The reason I segment the dataset in step 2 is we can't standardize the test set, we need to apply the mean and standard deviation of features in training set to features in the test set.

5.2 Training process

Once we finish preprocessing and segment the dataset into two parts: training set and test set, we need to set aside and never look at the test side in the whole training process.

Step 6: Apply **cross-validation** method to the training process. The training set is divided into **five** equal-size sets, each time four of them are used for training and the rest one is used for testing the performance for the specified model and parameters. In the project, there are three machine learning methods: logistic regression, random forest and adaboost. I'll use cross-validation to find the best parameters for each method.

Especially, we need to consider the **hypothesis set** of different methods since it is related to the feasibility and performance of machine learning. For random forest and adaboost, the basic idea is using decision tree as weak classifier. For each tree, it should have the unit hypothesis set $h\{i\}$, the depth within the unit hypothesis is decided by its required depth and halting situation. In general, for depth = d and node = n, the $h\{i\} = D^n$, where D is the feature dimension. Therefore, for the whole system, the hypothesis set $H = \bigcup_i^N h\{i\}$, where N is the number of decision trees.

5.3 Evaluation

After the training process, we get the optimal model for each method. In this evaluation section, test set is brought into the picture.

Step 7: Use the **test set** to evaluate the performance of the optimal models for each method.
Step 8: Compare the in-sample error and out-of-sample error of these models, select the classifier with the best performance as the final classification system.

6. Implementation

6.1 Feature Space

There are two datasets in this project. Dataset A is provided by IBM and dataset B is provided by another company. Table 3 and 4 are features in dataset A and b respectively.

Table 3 Features in dataset A

Number	Feature's Name	Type	Cardinality/Range	Description
1	Age	integer	18--60	
2	Business Travel	categorical	3	
3	DailyRate	integer	102--1499	
4	Department	categorical	3	
5	DistanceFromHome	integer	1--29	

6	Education	integer	1--5	1 'Below College' 2 'College' 3 'Bachelor' 4 'Master' 5 'Doctor'
7	EducationField	categorical	6	
8	EnvironmentSatisfication	integer	1--4	1 'Low' 2 'Medium' 3 'High' 4 'Very High'
9	Gender	categorical	2	
10	HourlyRate	integer	30--100	
11	JobInvolvement	integer	1--4	1 'Low' 2 'Medium' 3 'High' 4 'Very High'
12	JobLevel	integer	1--5	
13	JobRole	categorical	9	
14	JobSatisfication	integer	1--4	1 'Low' 2 'Medium' 3 'High' 4 'Very High'
15	MaritalStatus	categorical	3	
16	MonthlyIncome	integer	1009--19999	
17	MonthlyRate	integer	2094--26999	
18	NumCompaniesWorked	integer	0--9	
19	PercentSalaryHike	integer	11--25	
20	PerformanceRating	integer	1--4	1 'Low' 2 'Good' 3 'Excellent' 4 'Outstanding'
21	RelationshipSatisfaction	integer	1--4	1 'Low' 2 'Medium' 3 'High' 4 'Very High'
22	StockOptionLevel	integer	0--3	
23	TotalWorkingYears	integer	0--40	
24	TrainingTimesLastYear	integer	0--6	
25	WorkLifeBalance	Integer	1--4	1 'Bad' 2 'Good' 3 'Better' 4 'Best'
26	YearsAtCompany	Integer	0--40	
27	YearsInCurrentRole	Integer	0--18	
28	YearsSinceLastPromotion	Integer	0--15	
29	YearsWithCurrManager	Integer	0--17	

Note: Here, the type of features 6, 8, 11, 14, 20, 21, 25 are integer. Because in the original dataset, they are represented by numbers. For example, for feature ‘WorkLifeBalance’, 1is ‘Bad’, 2 is ‘Good’, 3 is ‘Better’ and 4 is ‘Best’. There is a logical ordering between these integers. In lecture 9, professor said the type of ‘Quality of Location’ is integer, which is {1, 2, 3}.

Table 4 Features in dataset B

Number	Feature’s Name	Type	Cardinality/Range	Description
1	satisfaction_level	Real	0.09—1.00	JobSatisfication
2	last_evaluation	Real	0.36—1.00	
3	number_project	integer	2—7	
4	average_montly_hours	Integer	96—310	
5	time_spend_company	Integer	2—10	YearsAtCompany
6	Work_accident	Integer	0—1	0 ‘No work accident’ 1 ‘Have work accident’
7	promotion_last_5years	Integer	0—1	0 ‘No promotion’ 1 ‘Have promotion’
8Salary	sales	categorical	10	department
9	salary	categorical	3	Salary level

6.2 Pre-processing

Step 1: Using **one hot encoding** for categorical features.

After using one hot encoding, the number of features of dataset A is **51**; the number of features of dataset B is **20**.

Step 2: Split the dataset into **training and test data**

For dataset A: there are **1176** samples in training set and **294** samples in test set, **51** features;

For dataset B: there are **11999** samples in training set and **3000** samples in test set, **20** features.

Step 3: **Standardize** the numeric features in training set and test set.

The reason I segment the dataset in step 2 is we can’t standardize the test set, we need to apply the mean and standard deviation of features in training set to features in the test set.

There is no missing data in both dataset. I didn’t do feature extraction in this project.

6.3 Training Process

I implement three methods: logistic regression, random forest and adaboost. For each method, I use cross-validation to select the optimal parameters. Here, I use function **GridSearchCV()** to implement cross-validation.

Table 5 is the complexity of hypothesis sets.

Table 5 Complexity of Hypothesis Sets

Dataset	Number of samples in original dataset	Number of samples in training set	Number of samples in test set	Dimension of pre-processed feature space
A	1470	1176	294	51
B	14999	11999	3000	20

Table 6 is the libraries and functions I used for each method.

Table 6 Libraries and Function within Scikit

	Logistic Regression	Random Forest	Adaboost
Library	sklearn.linear_model. LogisticRegression	sklearn.ensemble. RandomForestClassifier	sklearn.ensemble. AdaBoostClassifier
Training	Model.fit		
Predicting	Model.predict		

6.4 Testing, Validation and Model Selection

Firstly, for dataset A

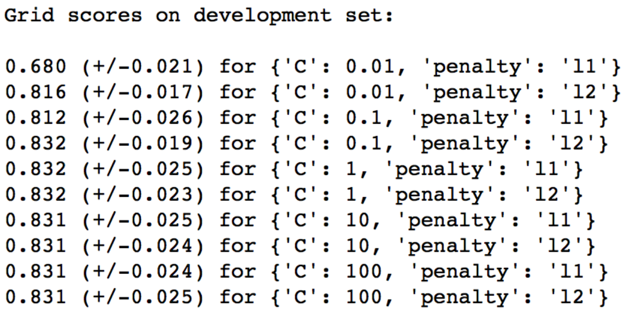
Logistic Regression

Tuned parameters:

Table 7 Parameters within logistic regression classifier

C	0.01, 0.1, 1, 10, 100
penalty	'l1', 'l2'

The cross-validation results are shown below:



```
Grid scores on development set:

0.680 (+/-0.021) for {'C': 0.01, 'penalty': 'l1'}
0.816 (+/-0.017) for {'C': 0.01, 'penalty': 'l2'}
0.812 (+/-0.026) for {'C': 0.1, 'penalty': 'l1'}
0.832 (+/-0.019) for {'C': 0.1, 'penalty': 'l2'}
0.832 (+/-0.025) for {'C': 1, 'penalty': 'l1'}
0.832 (+/-0.023) for {'C': 1, 'penalty': 'l2'}
0.831 (+/-0.025) for {'C': 10, 'penalty': 'l1'}
0.831 (+/-0.024) for {'C': 10, 'penalty': 'l2'}
0.831 (+/-0.024) for {'C': 100, 'penalty': 'l1'}
0.831 (+/-0.025) for {'C': 100, 'penalty': 'l2'}
```

So, we can choose the best parameters based on results: **{'C': 0.1, 'penalty': 'l2'}**

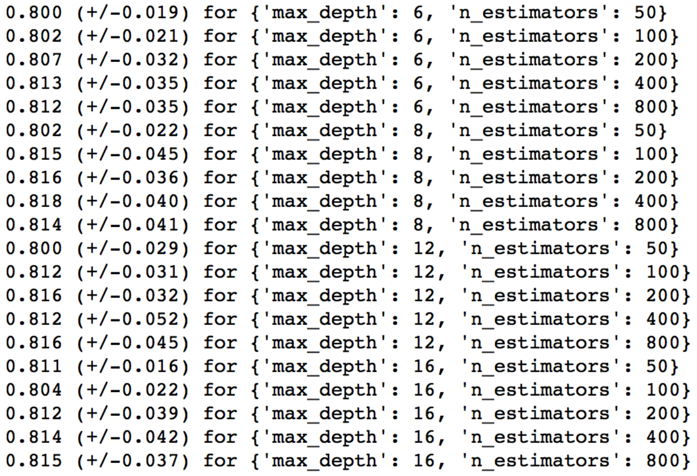
Random Forest:

Tuned parameters:

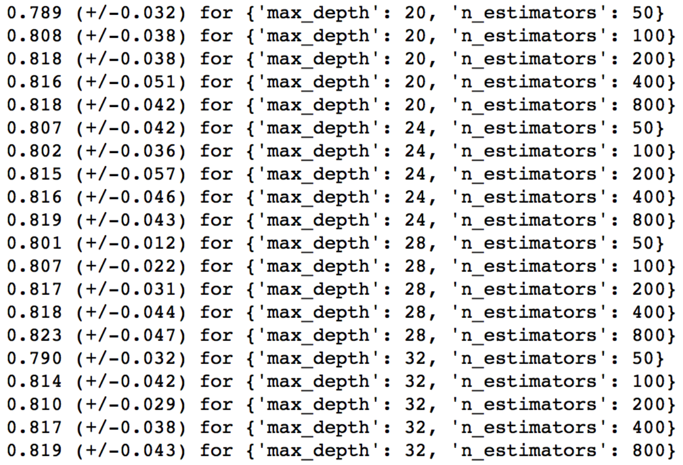
Table 8 Parameters within random forest classifier

max_depth ()	6, 8, 12, 16, 20, 24, 28, 32
n_estimators	50, 100, 200, 400, 800

The cross-validation results are shown below:



```
0.800 (+/-0.019) for {'max_depth': 6, 'n_estimators': 50}
0.802 (+/-0.021) for {'max_depth': 6, 'n_estimators': 100}
0.807 (+/-0.032) for {'max_depth': 6, 'n_estimators': 200}
0.813 (+/-0.035) for {'max_depth': 6, 'n_estimators': 400}
0.812 (+/-0.035) for {'max_depth': 6, 'n_estimators': 800}
0.802 (+/-0.022) for {'max_depth': 8, 'n_estimators': 50}
0.815 (+/-0.045) for {'max_depth': 8, 'n_estimators': 100}
0.816 (+/-0.036) for {'max_depth': 8, 'n_estimators': 200}
0.818 (+/-0.040) for {'max_depth': 8, 'n_estimators': 400}
0.814 (+/-0.041) for {'max_depth': 8, 'n_estimators': 800}
0.800 (+/-0.029) for {'max_depth': 12, 'n_estimators': 50}
0.812 (+/-0.031) for {'max_depth': 12, 'n_estimators': 100}
0.816 (+/-0.032) for {'max_depth': 12, 'n_estimators': 200}
0.812 (+/-0.052) for {'max_depth': 12, 'n_estimators': 400}
0.816 (+/-0.045) for {'max_depth': 12, 'n_estimators': 800}
0.811 (+/-0.016) for {'max_depth': 16, 'n_estimators': 50}
0.804 (+/-0.022) for {'max_depth': 16, 'n_estimators': 100}
0.812 (+/-0.039) for {'max_depth': 16, 'n_estimators': 200}
0.814 (+/-0.042) for {'max_depth': 16, 'n_estimators': 400}
0.815 (+/-0.037) for {'max_depth': 16, 'n_estimators': 800}
```



```
0.789 (+/-0.032) for {'max_depth': 20, 'n_estimators': 50}
0.808 (+/-0.038) for {'max_depth': 20, 'n_estimators': 100}
0.818 (+/-0.038) for {'max_depth': 20, 'n_estimators': 200}
0.816 (+/-0.051) for {'max_depth': 20, 'n_estimators': 400}
0.818 (+/-0.042) for {'max_depth': 20, 'n_estimators': 800}
0.807 (+/-0.042) for {'max_depth': 24, 'n_estimators': 50}
0.802 (+/-0.036) for {'max_depth': 24, 'n_estimators': 100}
0.815 (+/-0.057) for {'max_depth': 24, 'n_estimators': 200}
0.816 (+/-0.046) for {'max_depth': 24, 'n_estimators': 400}
0.819 (+/-0.043) for {'max_depth': 24, 'n_estimators': 800}
0.801 (+/-0.012) for {'max_depth': 28, 'n_estimators': 50}
0.807 (+/-0.022) for {'max_depth': 28, 'n_estimators': 100}
0.817 (+/-0.031) for {'max_depth': 28, 'n_estimators': 200}
0.818 (+/-0.044) for {'max_depth': 28, 'n_estimators': 400}
0.823 (+/-0.047) for {'max_depth': 28, 'n_estimators': 800}
0.790 (+/-0.032) for {'max_depth': 32, 'n_estimators': 50}
0.814 (+/-0.042) for {'max_depth': 32, 'n_estimators': 100}
0.810 (+/-0.029) for {'max_depth': 32, 'n_estimators': 200}
0.817 (+/-0.038) for {'max_depth': 32, 'n_estimators': 400}
0.819 (+/-0.043) for {'max_depth': 32, 'n_estimators': 800}
```

So, we can choose the best parameters based on results: **{'max_depth': 28, 'n_estimators': 800}**

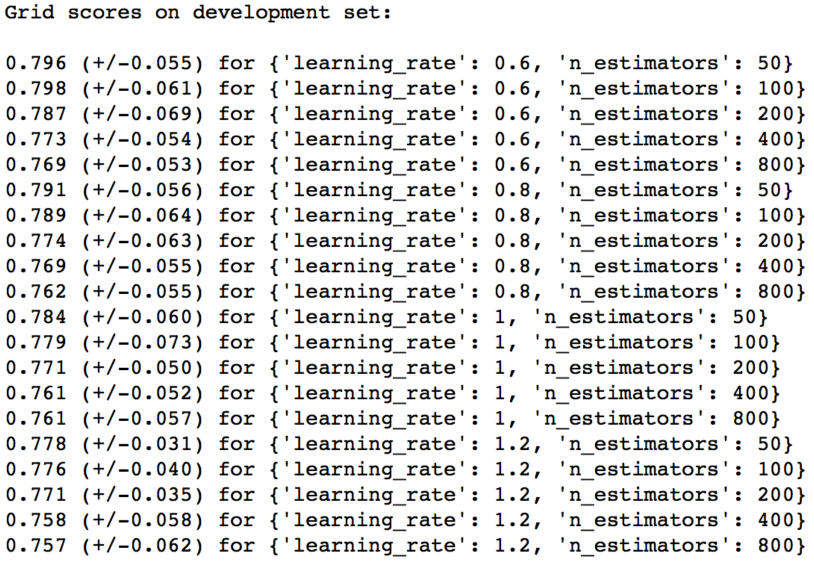
Adaboost:

Tuned parameters:

Table 9 Parameters within Adaboost classifier

learning_rate	0.6, 0.8, 1, 1.2
n_estimators	50, 100, 200, 400, 800

The cross-validation results are shown below:



```
Grid scores on development set:

0.796 (+/-0.055) for {'learning_rate': 0.6, 'n_estimators': 50}
0.798 (+/-0.061) for {'learning_rate': 0.6, 'n_estimators': 100}
0.787 (+/-0.069) for {'learning_rate': 0.6, 'n_estimators': 200}
0.773 (+/-0.054) for {'learning_rate': 0.6, 'n_estimators': 400}
0.769 (+/-0.053) for {'learning_rate': 0.6, 'n_estimators': 800}
0.791 (+/-0.056) for {'learning_rate': 0.8, 'n_estimators': 50}
0.789 (+/-0.064) for {'learning_rate': 0.8, 'n_estimators': 100}
0.774 (+/-0.063) for {'learning_rate': 0.8, 'n_estimators': 200}
0.769 (+/-0.055) for {'learning_rate': 0.8, 'n_estimators': 400}
0.762 (+/-0.055) for {'learning_rate': 0.8, 'n_estimators': 800}
0.784 (+/-0.060) for {'learning_rate': 1, 'n_estimators': 50}
0.779 (+/-0.073) for {'learning_rate': 1, 'n_estimators': 100}
0.771 (+/-0.050) for {'learning_rate': 1, 'n_estimators': 200}
0.761 (+/-0.052) for {'learning_rate': 1, 'n_estimators': 400}
0.761 (+/-0.057) for {'learning_rate': 1, 'n_estimators': 800}
0.778 (+/-0.031) for {'learning_rate': 1.2, 'n_estimators': 50}
0.776 (+/-0.040) for {'learning_rate': 1.2, 'n_estimators': 100}
0.771 (+/-0.035) for {'learning_rate': 1.2, 'n_estimators': 200}
0.758 (+/-0.058) for {'learning_rate': 1.2, 'n_estimators': 400}
0.757 (+/-0.062) for {'learning_rate': 1.2, 'n_estimators': 800}
```

So, we can choose the best parameters based on results: **{'learning_rate': 0.6, 'n_estimators': 100}**

In sum, the best parameters that I choose for each method are:

Table 10 The Best Parameters of Each Method for **Dataset A**

Logistic Regression	{'C': 0.1, 'penalty': 'l2'}
Random Forest Classification	{'max_depth': 28, 'n_estimators': 800}
Adaboost Classification	{'learning_rate': 0.6, 'n_estimators': 100}

Secondly, for dataset B, I use the same methods as the prior ones. So, I'll just show the best parameters I choose for each method based on results.

Table 11 The Best Parameters of Each Method for **Dataset B**

Logistic Regression	{'C': 0.1, 'penalty': 'l1'}
Random Forest Classification	{'max_depth': 28, 'n_estimators': 800}
Adaboost Classification	{'learning_rate': 1, 'n_estimators': 800}

Finally, I use the test set, which have not been 'looked' through the whole training process, to do evaluation on the different models with the best parameters and get the predictive accuracy.

7. Final Results

Table 12 Final Results for Dataset A

	ROC_AUC _training	ROC_AUC _test	precision	recall	f1 score	Computaion Time(/s)	Best Parameters
Base Rate Model	0.5	0.5	0.71	0.84	0.77	0.004	
Logistic Regression	**0.832**	**0.67**	**0.81**	**0.71**	**0.74**	**1.499**	**{'C': 0.1, 'penalty': 'l2'}**
Random Forest	0.823	0.55	0.83	0.85	0.80	74.555	{'max_depth': 28, 'n_estimators': 800}
Adaboost	0.798	0.61	0.85	0.86	0.83	40.021	{'learning_rate': 0.6, 'n_estimators': 100}

Table 13 Final Results for Dataset B

	ROC_AUC _training	ROC_AUC _test	precision	recall	f1 score	Computaion Time(/s)	Best Parameters
Base Rate Model	0.5	0.5	0.58	0.76	0.66	0.005	
Logistic Regression	0.828	0.78	0.82	0.77	0.78	3.797	{'C': 0.1, 'penalty': 'l1'}
Random Forest	**0.993**	**0.98**	**0.99**	**0.99**	**0.99**	**204.803**	**{'max_depth': 28, 'n_estimators': 800}**
Adaboost	0.983	0.94	0.95	0.95	0.95	104.467	{'learning_rate': 1, 'n_estimators': 800}

Figure 2 and 3 is the ROC graph of dataset A and B.

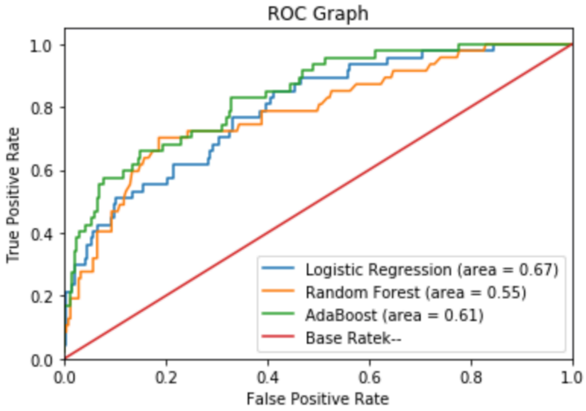


Figure 2 ROC graph of dataset A

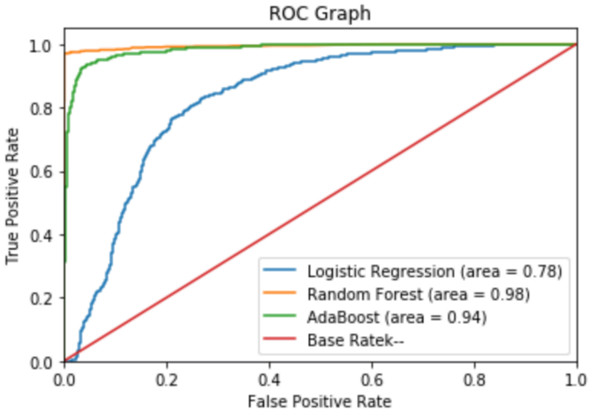


Figure 3 ROC graph of dataset B

Depending on the results above, for **dataset A**, the **logistic regression** performs the best; for **dataset B**, **random forest** performs the best.

8. Interpretation

I use same method on different datasets. The results of dataset A and B have huge difference. So, I have two conjecture about it:
First, models for A might have over-fitting problem.
Second, features extracted in the dataset A are bad.

To verify the first conjecture, I check Table 12 Final Results for Dataset A. Clearly, the training accuracy is higher than test accuracy. So, there isn't an overfitting.
As for the second conjecture, there are huge difference between the features between dataset A and B. The features selected by each company might be the reason that results of B are better than A.