MA679 Midterm Exam

Zhiwei Liang

3/18/2021

Context

A company wants to hire data scientists from pool of people enrolled in the courses conduct by the company. The company wants to know which of these candidates are looking to change their job. Information related to demographics, education, experience are in hands from candidates signup and enrollment. In this exam, your goal is to predict if the candidate is looking for a new job or will work for the current company.

- uid : Unique ID for candidate
- city: City code
- city_dev_index : Development index of the city (scaled)
- gender: Gender of candidate
- relevant_experience: Relevant experience of candidate
- enrolled_university: Type of University course enrolled if any
- education_level: Education level of candidate
- major_discipline :Education major discipline of candidate
- experience_years: Candidate total experience in years
- company_size: No of employees in current employer's company
- company_type : Type of current employer
- lastnewjob: Difference in years between previous job and current job
- training_hours: training hours completed
- change_job: 0 – Not looking for job change, 1 – Looking for a job change

My Work

Data Processing

- Observe the train data

```r
# check the type of each variable
summary(train)
```

```
##        V1             uid            city_id           city_dev_index
##  Min.   :   1   Min.   :    1   Length:8000        Min.   :0.4480
##  1st Qu.:2001   1st Qu.: 8295   Class :character   1st Qu.:0.7430
##  Median :4000   Median :16660   Mode  :character   Median :0.9030
##  Mean   :4000   Mean   :16734                      Mean   :0.8293
##  3rd Qu.:6000   3rd Qu.:25081                      3rd Qu.:0.9200
##  Max.   :8000   Max.   :33377                      Max.   :0.9490
##     gender          relevant_experience enrolled_university education_level
##  Length:8000        Length:8000         Length:8000         Length:8000
##  Class :character   Class :character    Class :character    Class :character
##  Mode  :character   Mode  :character    Mode  :character    Mode  :character
##
##
##
##  major_discipline   experience_years   company_size       company_type
##  Length:8000        Length:8000        Length:8000        Length:8000
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##  last_new_job       training_hours     change_job
##  Length:8000        Min.   :  1.00   Min.   :0.0000
##  Class :character   1st Qu.: 23.00   1st Qu.:0.0000
##  Mode  :character   Median : 47.00   Median :0.0000
##                     Mean   : 65.02   Mean   :0.2432
##                     3rd Qu.: 88.00   3rd Qu.:0.0000
##                     Max.   :336.00   Max.   :1.0000
```

```r
# cancel some useful variable
train = train %>% select(-city_id,-uid,-V1)
test= test %>% select(-city_id)
```

- Convert character variable to the factor

To better achieve the results, I bin below features into different intervals. For `experience_years`, I bin into four categories based on the quantiles: Y1 to Y4.

```r
##################################### convert train data #################################
train$gender = factor(train$gender)
train$relevant_experience = factor(train$relevant_experience)
train$enrolled_university = factor(train$enrolled_university)
train$education_level = factor(train$education_level)
train$major_discipline = factor(train$major_discipline)
train$company_type = factor(train$company_type)
train$last_new_job = factor(train$last_new_job)
train$change_job = factor(train$change_job)
train$company_size = factor(train$company_size)
train$city_dev_index = as.numeric(train$city_dev_index)

# year
#unique(train$experience_years)
train$experience_level = train$experience_years
train$experience_years[train$experience_years == ">20"] = "22"
```

```r
train$experience_years[train$experience_years == "<1"] = "0"
train$experience_years = as.numeric(train$experience_years)
#hist(train$experience_years)
#summary(train$experience_years)
train$experience_level[train$experience_years>=0 & train$experience_years <4] = "Y1"
train$experience_level[train$experience_years>=4 & train$experience_years <10] = "Y2"
train$experience_level[train$experience_years>=10 & train$experience_years <16] = "Y3"
train$experience_level[train$experience_years>=16] = "Y4"
train$experience_level = factor(train$experience_level)



################################### convert test data #################################
test$gender = factor(test$gender)
test$relevant_experience = factor(test$relevant_experience)
test$enrolled_university = factor(test$enrolled_university)
test$education_level = factor(test$education_level)
test$major_discipline = factor(test$major_discipline)
test$company_type = factor(test$company_type)
test$last_new_job = factor(test$last_new_job)
test$company_size = factor(test$company_size)
test$city_dev_index = as.numeric(test$city_dev_index)

# year
#unique(test$experience_years) #add a col experience_level
test$experience_level = test$experience_years
test$experience_years[test$experience_years == ">20"] = "22"
test$experience_years[test$experience_years == "<1"] = "0"
test$experience_years = as.numeric(test$experience_years)
#hist(train$experience_years)
#summary(train$experience_years)
test$experience_level[test$experience_years>=0 & test$experience_years <4] = "Y1"
test$experience_level[test$experience_years>=4 & test$experience_years <10] = "Y2"
test$experience_level[test$experience_years>=10 & test$experience_years <16] = "Y3"
test$experience_level[test$experience_years>=16] = "Y4"
test$experience_level = factor(test$experience_level)
```

- Deal with the missing value

I use `skimr` and `forcats` packages to detect the missing value and handle them by imputation.

```r
library(skimr)
library(ggplot2)
library(forcats)

############################################### train data imputation ##############################
library(missForest)
set.seed(12)

train_impute <- missForest(xmis = train, maxiter = 2, ntree = 20)
```

```
##   missForest iteration 1 in progress...done!
##   missForest iteration 2 in progress...done!
```

```r
train_impute  <- train_impute$ximp



############################################## test data imputation ###########################################
test_impute <- missForest(xmis = test, maxiter = 2, ntree = 20)
```

```
##   missForest iteration 1 in progress...done!
##   missForest iteration 2 in progress...done!
```

```r
test_impute  <- test_impute$ximp
```

- Split the train data into two part

In order to validate the model later, I put 80% train dataset into `train_set` and 20% into`test_set`. The `train_set` is used to train the model and `test_set` is used to test the model.

```r
#write.csv(train_impute,"train_impute.csv")
#write.csv(test_impute,"test_impute.csv")
#train_impute = read.csv("train_impute.csv")
#test_impute = read.csv("test_impute.csv")

# split the train_imput into the train_set and test_set
library(caTools)
set.seed(12)
split = sample.split(train_impute$change_job, SplitRatio = 0.8)

train_set = subset(train_impute, split == TRUE) # for modeling
test_set = subset(train_impute, split == FALSE) # for validation

#nrow(train_set)
#nrow(test_set)
```

Model Selection

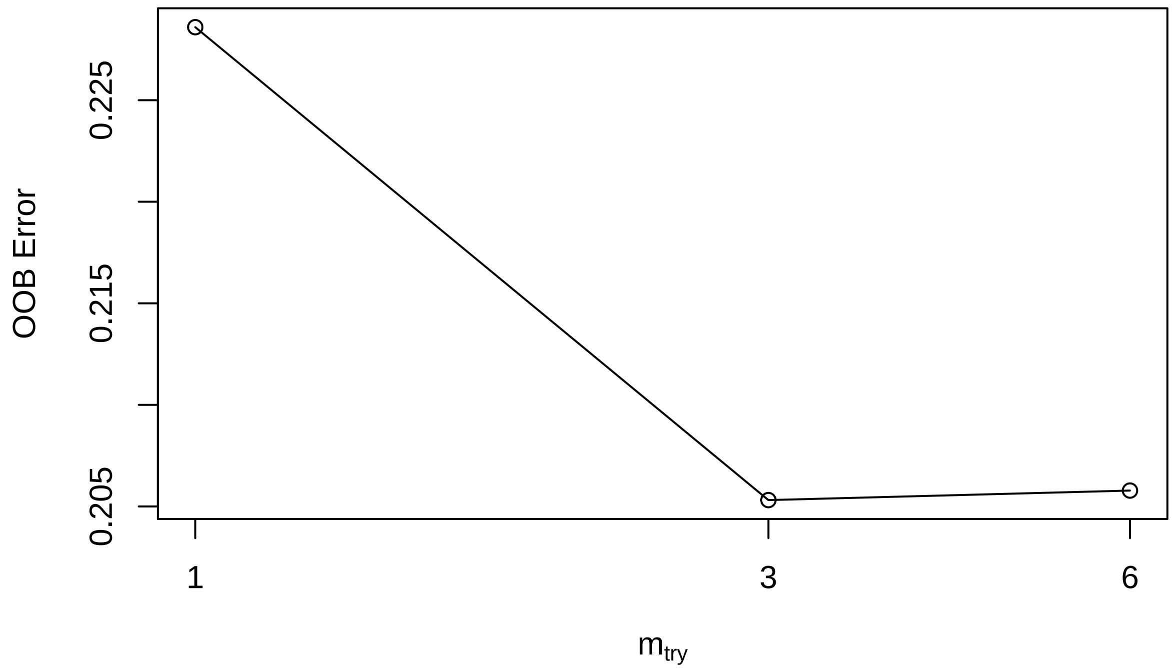
Random Forest

- Choose the best `mtry`

```r
library(randomForest)
set.seed(12)

train_mytry <- tuneRF(x = train_set[,-12], y = train_set$change_job,
       stepFactor = 0.5,
       plot = TRUE,
       ntreeTry = 200,
       trace = TRUE,
       improve = 0.05)
```

```
## mtry = 3  OOB error = 20.53%
## Searching left ...
## mtry = 6     OOB error = 20.58%
## -0.002283105 0.05
## Searching right ...
## mtry = 1     OOB error = 22.86%
## -0.1133942 0.05
```



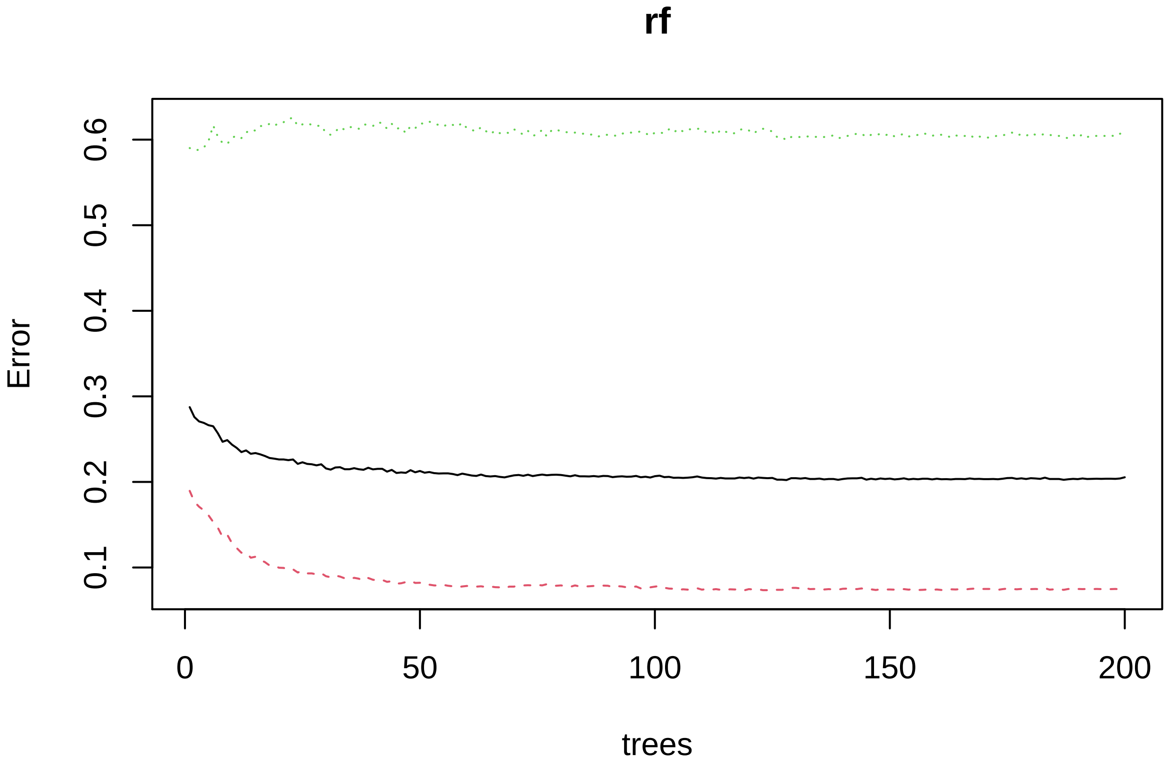

```
# mtry=3
```

- Build random forest model, using the `train_set`

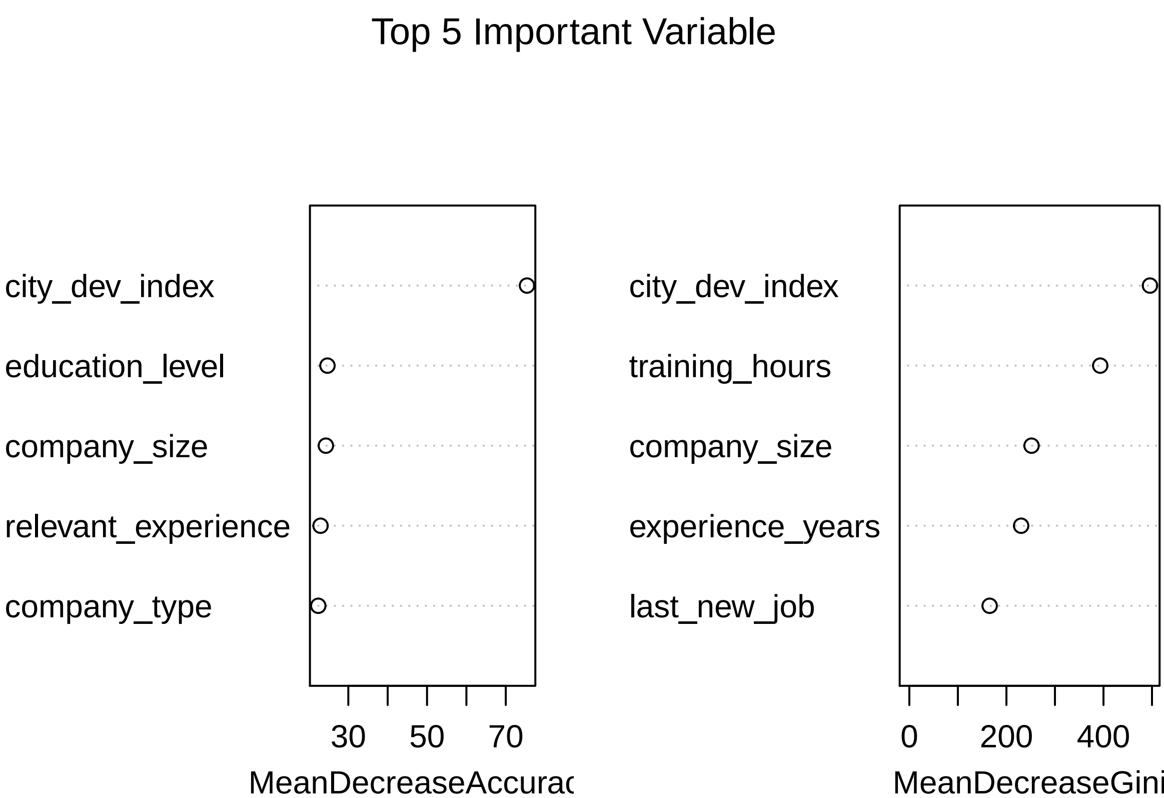
```
set.seed(12)
rf = randomForest::randomForest(
  change_job ~ .,
  data = train_set,
  ntree = 200,
  mtry = 3,
  importance = TRUE,
  type = 'class'
  )
rf
```

```
##
## Call:
##  randomForest(formula = change_job ~ ., data = train_set, ntree = 200,      mtry = 3, importance = TI
##                Type of random forest: classification
##                      Number of trees: 200
## No. of variables tried at each split: 3
##
##         OOB estimate of  error rate: 20.55%
## Confusion matrix:
##      0   1 class.error
## 0 4474 369  0.07619244
## 1  946 611  0.60757868
```

```
plot(rf)
```




```r
# find the most important variables
varImpPlot(rf,
           sort = T,
           n.var = 5,
           main = "Top 5 Important Variable")
```



We can see that the level of city development and people's training hours are the most important influence variables to the possibility of changing jobs.

Logistic Regression

```
lr=glm(formula = change_job ~ ., data = train_set,family=binomial(link="logit"))

summary(lr)
```

```
##
## Call:
## glm(formula = change_job ~ ., family = binomial(link = "logit"),
##     data = train_set)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.1363  -0.6724  -0.4932  -0.3054   2.5969
##
## Coefficients:
##                                               Estimate Std. Error z value
## (Intercept)                                  4.7966798  0.3979890  12.052
## city_dev_index                              -5.9498406  0.2686298 -22.149
## genderMale                                  -0.1511968  0.1209984  -1.250
## genderOther                                 -0.0295112  0.3111831  -0.095
## relevant_experienceNo relevent experience    0.5022444  0.0841259   5.970
## enrolled_universityno_enrollment            -0.2753093  0.0859857  -3.202
## enrolled_universityPart time course         -0.3040838  0.1454568  -2.091
## education_levelHigh School                  -0.8573962  0.1208942  -7.092
## education_levelMasters                      -0.2702643  0.0816098  -3.312
```

```
## education_levelPhd                          -0.6190298  0.2705785  -2.288
## education_levelPrimary School               -0.9825326  0.2696699  -3.643
## major_disciplineBusiness Degree             -0.6951705  0.3498977  -1.987
## major_disciplineHumanities                  -0.6443399  0.3040828  -2.119
## major_disciplineNo Major                    -0.6154018  0.3709767  -1.659
## major_disciplineOther                       -0.7962022  0.3483441  -2.286
## major_disciplineSTEM                        -0.6305183  0.2526469  -2.496
## experience_years                            -0.0168652  0.0188922  -0.893
## company_size10/49                            0.5604972  0.1301840   4.305
## company_size100-500                          0.0604435  0.1317923   0.459
## company_size1000-4999                        0.2712762  0.1394520   1.945
## company_size10000+                           0.3665681  0.1353197   2.709
## company_size50-99                           -0.0704077  0.1293125  -0.544
## company_size500-999                          0.3440339  0.1522066   2.260
## company_size5000-9999                        0.6590269  0.1586010   4.155
## company_typeFunded Startup                  -0.4186613  0.1938852  -2.159
## company_typeNGO                             -0.0226880  0.2080803  -0.109
## company_typeOther                            0.0301017  0.4080396   0.074
## company_typePublic Sector                    0.5694539  0.1748748   3.256
## company_typePvt Ltd                         -0.1456110  0.1464486  -0.994
## last_new_job1                                0.1150830  0.1069242   1.076
## last_new_job2                                0.0188048  0.1219456   0.154
## last_new_job3                               -0.0020004  0.1693992  -0.012
## last_new_job4                               -0.0014832  0.1721994  -0.009
## last_new_jobnever                           -0.2086531  0.1387745  -1.504
## training_hours                              -0.0008911  0.0005546  -1.607
## experience_levelY2                          -0.1997095  0.1172025  -1.704
## experience_levelY3                          -0.0302361  0.2154390  -0.140
## experience_levelY4                          -0.0849316  0.3543447  -0.240
##                                             Pr(>|z|)
## (Intercept)                                  < 2e-16 ***
## city_dev_index                               < 2e-16 ***
## genderMale                                  0.211454
## genderOther                                 0.924446
## relevant_experienceNo relevent experience 2.37e-09 ***
## enrolled_universityno_enrollment            0.001366 **
## enrolled_universityPart time course         0.036569 *
## education_levelHigh School                  1.32e-12 ***
## education_levelMasters                      0.000927 ***
## education_levelPhd                          0.022149 *
## education_levelPrimary School               0.000269 ***
## major_disciplineBusiness Degree             0.046947 *
## major_disciplineHumanities                  0.034094 *
## major_disciplineNo Major                    0.097142 .
## major_disciplineOther                       0.022273 *
## major_disciplineSTEM                        0.012573 *
## experience_years                            0.372014
## company_size10/49                           1.67e-05 ***
## company_size100-500                         0.646502
## company_size1000-4999                       0.051739 .
## company_size10000+                          0.006751 **
## company_size50-99                           0.586113
## company_size500-999                         0.023802 *
## company_size5000-9999                       3.25e-05 ***
```

```
## company_typeFunded Startup                0.030825 *
## company_typeNGO                           0.913175
## company_typeOther                         0.941192
## company_typePublic Sector                 0.001129 **
## company_typePvt Ltd                       0.320086
## last_new_job1                             0.281791
## last_new_job2                             0.877447
## last_new_job3                             0.990578
## last_new_job4                             0.993127
## last_new_jobnever                         0.132700
## training_hours                            0.108092
## experience_levelY2                        0.088387 .
## experience_levelY3                        0.888386
## experience_levelY4                        0.810573
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 7101.9  on 6399  degrees of freedom
## Residual deviance: 6070.1  on 6362  degrees of freedom
## AIC: 6146.1
##
## Number of Fisher Scoring iterations: 4
```

Model Validation

- **Confusion Matrix function** evaluates data accuracy,sensitivity, specificity and F-Score. The parameters represents the confusion matrix of a prediction.

```
Confusion_Matrix = function(confusion) {
  TP = confusion[4]
  TN = confusion[1]
  FP = confusion[2]
  FN = confusion[3]

  accuracy = round((TP+TN)/(TP+TN+FP+FN),4)
  sensitivity = round(TP/(TP+FN),4)
  specificity = round(TN/(TN+FP),4)
  F1Score = round((2*TP)/(2*TP+FP+FN),4)
  PPV = TP/(TP+FP)
  NPV = TN/(TN+FN)

  print(confusion)
  print(c("accuracy:", round(accuracy,4)))
  print(c("sensitivity:", round(sensitivity,4)))
  print(c("specificity:", round(specificity,4)))
  print(c("F1Score:", round(F1Score,4)))
  print(c("PPV:", round(PPV,4)))
  print(c("NPV:", round(NPV,4)))
  return(accuracy)
}
```

- **Random Forest Validation**

```r
# random forest prediction, predict the train_set
pred_train_rf = predict(rf,train_set)
pred_train_t = table(actual = train_set$change_job,predicted = pred_train_rf)
Confusion_Matrix(pred_train_t)
```

```
##        predicted
## actual    0    1
##      0 4831   12
##      1  119 1438
## [1] "accuracy:" "0.9795"
## [1] "sensitivity:" "0.9917"
## [1] "specificity:" "0.976"
## [1] "F1Score:" "0.9564"
## [1] "PPV:"   "0.9236"
## [1] "NPV:"   "0.9975"

## [1] 0.9795
```

```r
# random forest validation, predict the test_set
pred_test_rf = predict(rf,test_set)
pred_test_t=table(actual = test_set$change_job,predicted = pred_test_rf)
Confusion_Matrix(pred_test_t)
```

```
##        predicted
## actual    0    1
##      0 1141   70
##      1  233  156
## [1] "accuracy:" "0.8106"
## [1] "sensitivity:" "0.6903"
## [1] "specificity:" "0.8304"
## [1] "F1Score:" "0.5073"
## [1] "PPV:"  "0.401"
## [1] "NPV:"   "0.9422"

## [1] 0.8106
```

- **Logistic Regression Validation**

```r
# logistic regression, predict the train_set
p_train_lr = predict(lr,test_set,type = "response")
p_train_lr [p_train_lr >=0.5]=1
p_train_lr [p_train_lr <0.5]=0
t_train=table(actual = test_set$change_job,predicted =p_train_lr )
Confusion_Matrix(t_train)
```

```
##        predicted
## actual    0    1
##      0 1149   62
##      1  279  110
## [1] "accuracy:" "0.7869"
## [1] "sensitivity:" "0.6395"
## [1] "specificity:" "0.8046"
## [1] "F1Score:" "0.3922"
## [1] "PPV:"   "0.2828"
## [1] "NPV:"   "0.9488"

## [1] 0.7869
```

```r
# logistic regression, predict the test_set
p_test_lr = predict(lr,test_set,type = "response")
p_test_lr [p_test_lr >=0.5]=1
p_test_lr [p_test_lr <0.5]=0
t_test=table(actual = test_set$change_job,predicted =p_test_lr )
Confusion_Matrix(t_test)
```

```
##       predicted
## actual    0    1
##      0 1149   62
##      1  279  110
## [1] "accuracy:" "0.7869"
## [1] "sensitivity:" "0.6395"
## [1] "specificity:" "0.8046"
## [1] "F1Score:" "0.3922"
## [1] "PPV:"   "0.2828"
## [1] "NPV:"   "0.9488"

## [1] 0.7869
```

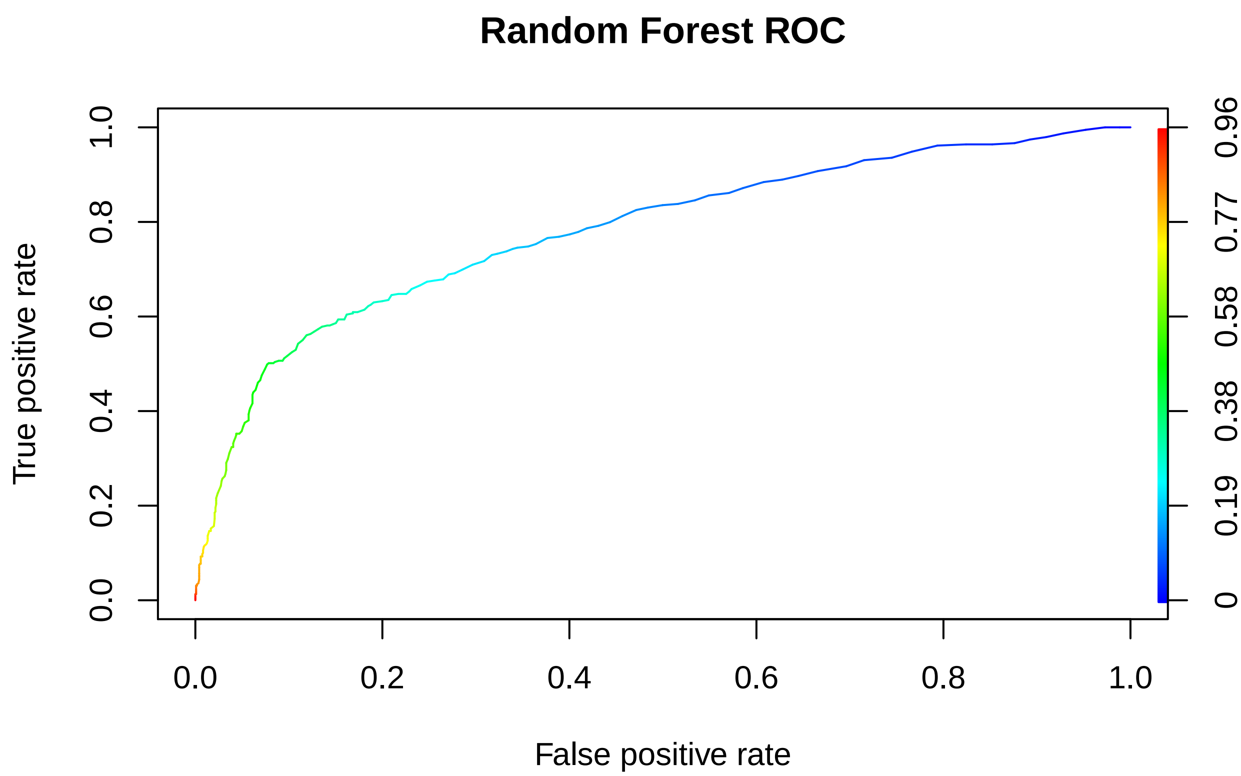
Through validation, we can clearly see the **accuracy** of Random Forest Model is far higher than Logistic Regression Model. The **sensitivity** and **specificity** of Random Forest Model is also a little bit higher than Logistic Regression Model.

So, I decide to use Random Forest Model.

Model Evaluation

- Draw **ROC** and calculate the **AUC** of Random Forest Model

```r
library(ROCR)
# random forest validation, predict the test_set
pred_test_rf_prob=predict(rf,test_set,type = "prob")
pred_test_rf_y=prediction(pred_test_rf_prob[,2],test_set$change_job)
perf_test<-performance(pred_test_rf_y,"tpr","fpr")
plot(perf_test,colorize=TRUE,main = "Random Forest ROC")
```




```
auc_test_rf <- performance(pred_test_rf_y,'auc')
auc_test_rf =unlist(slot(auc_test_rf,"y.values"))
auc_test_rf
```

```
## [1] 0.7807141
```

It seems like the model is significant since the AUC of test_set is **0.7807141**.

- Predict the test_sample

```
pred_rf = predict(rf,test_impute)
submit=cbind(test_impute,pred_rf)
write.csv(submit,"my_submit.csv")
```

Discussion

We can get a lot of informations from these two models, such as the following:

- The higher level of city development, people are less likely to change jobs.
- Those with less relevant job experience are more likely to change jobs.
- People in very small size companies are more likely to change jobs.

In summary, the accuracy and AUC scores of the Random Forest Model are both good. So, the prediction of the test sample is significant for us.

Limitations

- In both train and test sample, `change_job` which equal to 0 are much more than those equal to 1. I think this imbalance will influence the accuracy of the prediction model. SMOTE algrithom can solve the imbalance problem but I don't know how to use it in R.
- The sample size is not big enough. I worry the model will be overfitting.

Reference

An Introduction to Statistical Learning with Applications in R