Markdown_fa.R

yaxin

2021-04-12

```
# Uncomment if packages not installed
## install.packages("psych")
## install.packages("caret")
## install.packages("randomForest")
## install.packages("MLmetrics")
## install.packages("doParallel")
## install.packages("kernlab")
## install.packages("glmnet")



# Load data
setwd('D:\\Yaxin\\HKBU BM\\Courses\\Sem 2\\ECON7860 Big Data Analytics for Business (S11)\\Group Projec
rawData <- read.csv2("HR_comma_sep.csv", sep = ',')
colnames(rawData)
```

```
##  [1] "satisfaction_level"    "last_evaluation"       "number_project"
##  [4] "average_montly_hours"  "time_spend_company"    "Work_accident"
##  [7] "left"                  "promotion_last_5years" "sales"
## [10] "salary"
```

```
# Move the target variable "left" after "time_spend_company"
library(dplyr)
```

```
## Warning: package 'dplyr' was built under R version 4.0.4
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```r
rawData <- rawData %>% relocate(left, .after = time_spend_company)



# Transform feature types
transform_feature <- function(X) {
  X$satisfaction_level <- as.numeric(X$satisfaction_level)
  X$last_evaluation <- as.numeric(X$last_evaluation)
  X$Work_accident <- as.factor(X$Work_accident)
  X$promotion_last_5years <- as.factor(X$promotion_last_5years)
  X$sales <- as.factor(X$sales)
  X$salary <- as.factor(X$salary)
  X$left <- factor(ifelse(X$left == 0, 'no', 'yes'), levels = c('yes', 'no'))
  return(X)
}

rawData <- transform_feature(rawData)
summary(rawData)
```

```
##  satisfaction_level last_evaluation  number_project  average_montly_hours
##  Min.   :0.0900     Min.   :0.3600   Min.   :2.000   Min.   : 96.0
##  1st Qu.:0.4400     1st Qu.:0.5600   1st Qu.:3.000   1st Qu.:156.0
##  Median :0.6400     Median :0.7200   Median :4.000   Median :200.0
##  Mean   :0.6128     Mean   :0.7161   Mean   :3.803   Mean   :201.1
##  3rd Qu.:0.8200     3rd Qu.:0.8700   3rd Qu.:5.000   3rd Qu.:245.0
##  Max.   :1.0000     Max.   :1.0000   Max.   :7.000   Max.   :310.0
##
##  time_spend_company  left       Work_accident promotion_last_5years
##  Min.   : 2.000     yes: 3571   0:12830       0:14680
##  1st Qu.: 3.000     no :11428   1: 2169       1:  319
##  Median : 3.000
##  Mean   : 3.498
##  3rd Qu.: 4.000
##  Max.   :10.000
##
##          sales         salary
##  sales      :4140   high  :1237
##  technical  :2720   low   :7316
##  support    :2229   medium:6446
##  IT         :1227
##  product_mng: 902
##  marketing  : 858
##  (Other)    :2923
```

```r
# Separate features and target variable
X <- rawData
y <- X$left
tag <- colnames(X)
tag
```

```
##  [1] "satisfaction_level"    "last_evaluation"       "number_project"
##  [4] "average_montly_hours"  "time_spend_company"    "left"
##  [7] "Work_accident"         "promotion_last_5years" "sales"
## [10] "salary"
```

```
# Feature engineering
## Define the function for factor extraction
require(psych)
```

```
## Loading required package: psych
```

```
## Warning: package 'psych' was built under R version 4.0.4
```

```
fe <- function(M, n) {
  # The numeric feature matrix M needs to be normalized beforehand
  fa1 <- fa(M, n)
  fa.diagram(fa1)
  return(list('scores' = fa1$scores, 'weights' = fa1$weights))
}



## Create dummy variables for "sales" and "salary"
dummySales <- dummy.code(X$sales)
dummySalary <- dummy.code(X$salary)
colnames(dummySales)
```

```
##  [1] "sales"       "technical"   "support"     "IT"          "product_mng"
##  [6] "marketing"   "RandD"       "accounting"  "hr"          "management"
```

```
colnames(dummySalary)
```

```
## [1] "low"    "medium" "high"
```

```
### Set "sales" and "low" as the default values respectively
dummySales <- dummySales[ , -c(1)]
dummySalary <- dummySalary[ , -c(1)]

X_dummy <- cbind(X[ , -c(9, 10)], dummySales, dummySalary)
tag_dummy <- colnames(X_dummy)
tag_dummy
```

```
##  [1] "satisfaction_level"    "last_evaluation"       "number_project"
##  [4] "average_montly_hours"  "time_spend_company"    "left"
##  [7] "Work_accident"         "promotion_last_5years" "technical"
## [10] "support"               "IT"                    "product_mng"
## [13] "marketing"             "RandD"                 "accounting"
## [16] "hr"                    "management"            "medium"
## [19] "high"
```

```
# Train(80%)-test(20%)-split (stratified as "left" is unbalanced)
library(caret)
```

```
## Warning: package 'caret' was built under R version 4.0.4

## Loading required package: lattice

## Loading required package: ggplot2

##
## Attaching package: 'ggplot2'

## The following objects are masked from 'package:psych':
##
##     %+%, alpha
```

```
## Set seed for replication purpose
set.seed(7860)
index <- createDataPartition(y, p = 0.8, list = FALSE)
X_train <- X[index, ]
X_test <- cbind(X[-index, 1 : 5], X[-index, 7 : length(X)])
X_dummy_train <- X_dummy[index, ]
X_dummy_test <- cbind(X_dummy[-index, 1 : 5], X_dummy[-index, 7 : length(X_dummy)])
y_test <- y[-index]



# Modeling with extracted factors, 5-fold nested CV with random search
n_fold <- 5
cv <- createFolds(X_train$left, n_fold)
#models <- c('glmnet', 'svmLinear', 'rf', 'knn')
models <- c('glmnet', 'rf', 'knn')
tune <- 15
n_cluster <- 5 ## Please set the number of multiprocessing slaves accordingly


for (m in models) {
  assign(paste0(m, '_cv'),
         list('1' = list('model' = c(), 'f1_val' = c(), 'confm' = c()),
              '2' = list('model' = c(), 'f1_val' = c(), 'confm' = c()),
              '3' = list('model' = c(), 'f1_val' = c(), 'confm' = c()),
              '4' = list('model' = c(), 'f1_val' = c(), 'confm' = c()),
              '5' = list('model' = c(), 'f1_val' = c(), 'confm' = c())))

  for (i in 1 : n_fold) {
    ## Extract factors and weights from EFA
    fa_result <- fe(scale(X_train[-cv[[i]], 1 : 5]), 2)
    performance <- fa_result$scores[ , 1]
    satisfaction <- fa_result$scores[ , 2]
    w <- fa_result$weights
    colnames(w) <- c('performance', 'satisfaction')
    y_val <- X_train[cv[[i]], 'left']

    control <- trainControl(method = 'repeatedcv', number = 5, repeats = 2,
                            summaryFunction = prSummary, classProbs = TRUE,
                            search="random", verboseIter = TRUE)
```

```r
set.seed(7860)

require(doParallel)
cl <- makePSOCKcluster(n_cluster, outfile = '')
registerDoParallel(cl)

if (m == 'rf') {
  X_dev <- cbind(performance, satisfaction,
                 X_train[-cv[[i]], 6 : length(X_train)])
  X_val <- cbind(scale(as.matrix(X_train[cv[[i]], 1 : 5]) %*% w),
                 X_train[cv[[i]], 7 : length(X_train)])

  m1 <- train(left ~ ., data = X_dev, method = m, metric = 'F',
              tuneLength = tune, trControl = control)

  rf_cv[[i]][['model']] <- m1
  rf_cv[[i]][['f1_val']] <- F_meas(predict(m1, X_val), y_val)
  rf_cv[[i]][['confm']] <- confusionMatrix(predict(m1, X_val), y_val)

} else {
  X_dev <- cbind(performance, satisfaction,
                 X_dummy_train[-cv[[i]], 6 : length(X_dummy_train)])
  X_val <- cbind(scale(as.matrix(X_dummy_train[cv[[i]], 1 : 5]) %*% w),
                 X_dummy_train[cv[[i]], 7 : length(X_dummy_train)])

  if (m == 'glmnet') {
    m1 <- train(left ~ ., data = X_dev, method = m, family = 'binomial',
                metric = 'F', tuneLength = tune, trControl = control)

    glmnet_cv[[i]][['model']] <- m1
    glmnet_cv[[i]][['f1_val']] <- F_meas(predict(m1, X_val), y_val)
    glmnet_cv[[i]][['confm']] <- confusionMatrix(predict(m1, X_val), y_val)

  } else if (m == 'knn') {
    m1 <- train(left ~ ., data =  X_dev, method = m, metric = 'F',
                tuneLength = tune, trControl = control,
                tuneGrid = expand.grid(k = c(2, 3, 4, 5, 10)))

    knn_cv[[i]][['model']] <- m1
    knn_cv[[i]][['f1_val']] <- F_meas(predict(m1, X_val), y_val)
    knn_cv[[i]][['confm']] <- confusionMatrix(predict(m1, X_val), y_val)

  } else if (m == 'svmLinear') {
    m1 <- train(left ~ ., data =  X_dev, method = m, metric = 'F',
                tuneLength = tune, trControl = control)

    svmLinear_cv[[i]][['model']] <- m1
    svmLinear_cv[[i]][['f1_val']] <- F_meas(predict(m1, X_val), y_val)
    svmLinear_cv[[i]][['confm']] <- confusionMatrix(predict(m1, X_val), y_val)
  }
}

stopImplicitCluster()
```

```
    stopCluster(cl)
  }
}
```

```
## Loading required namespace: GPArotation
```

```
## Loading required package: doParallel
```

```
## Warning: package ’doParallel’ was built under R version 4.0.4
```

```
## Loading required package: foreach
```

```
## Warning: package ’foreach’ was built under R version 4.0.4
```

```
## Loading required package: iterators
```

```
## Warning: package ’iterators’ was built under R version 4.0.4
```

```
## Loading required package: parallel
```

```
## Warning in nominalTrainWorkflow(x = x, y = y, wts = weights, info = trainInfo, :
## There were missing values in resampled performance measures.
```

```
## Aggregating results
```

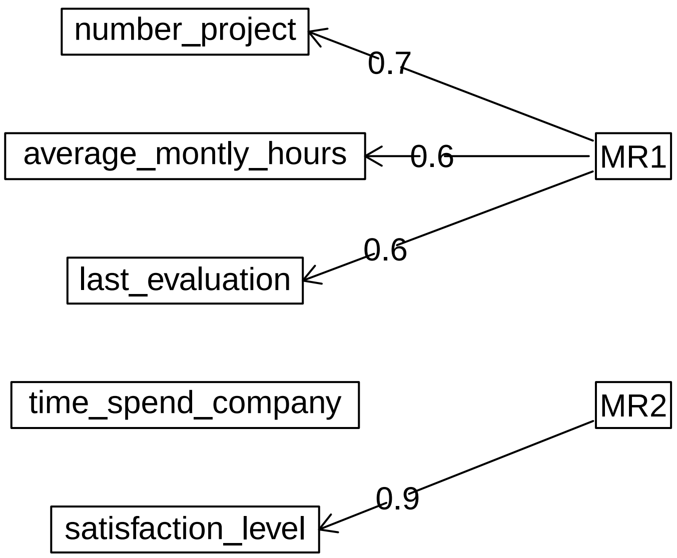
```
## Warning in train.default(x, y, weights = w, ...): missing values found in
## aggregated results
```

```
## Selecting tuning parameters
## Fitting alpha = 0.153, lambda = 0.00124 on full training set
```

```
## Warning in nominalTrainWorkflow(x = x, y = y, wts = weights, info = trainInfo, :
## There were missing values in resampled performance measures.
```

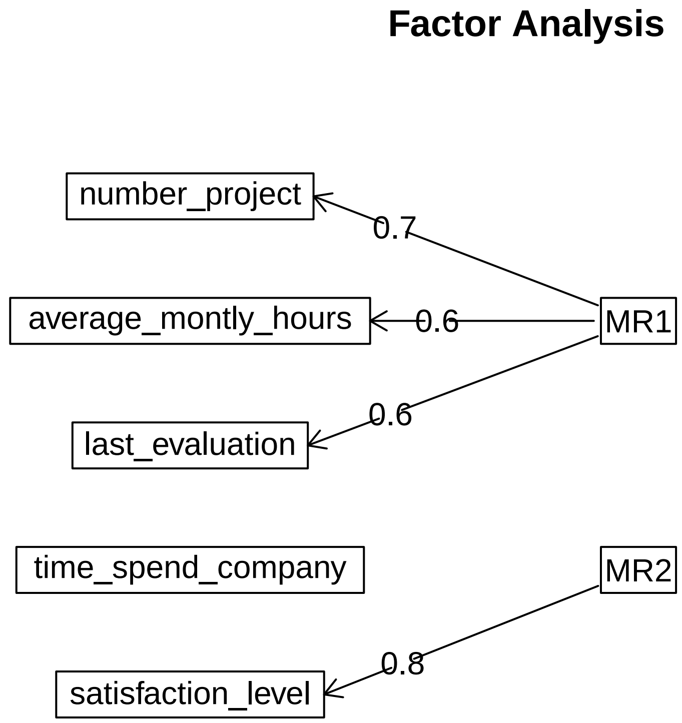
```
## Aggregating results
```

```
## Warning in train.default(x, y, weights = w, ...): missing values found in
## aggregated results
```

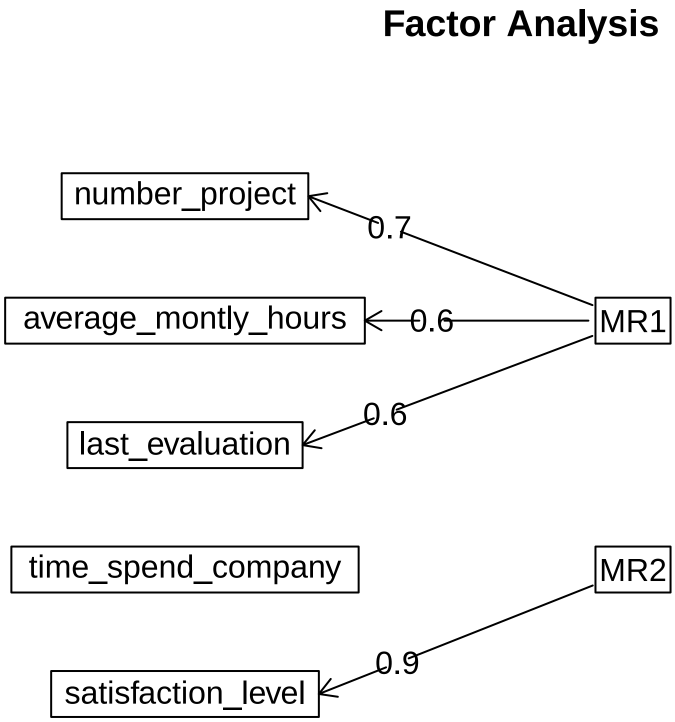


Factor Analysis

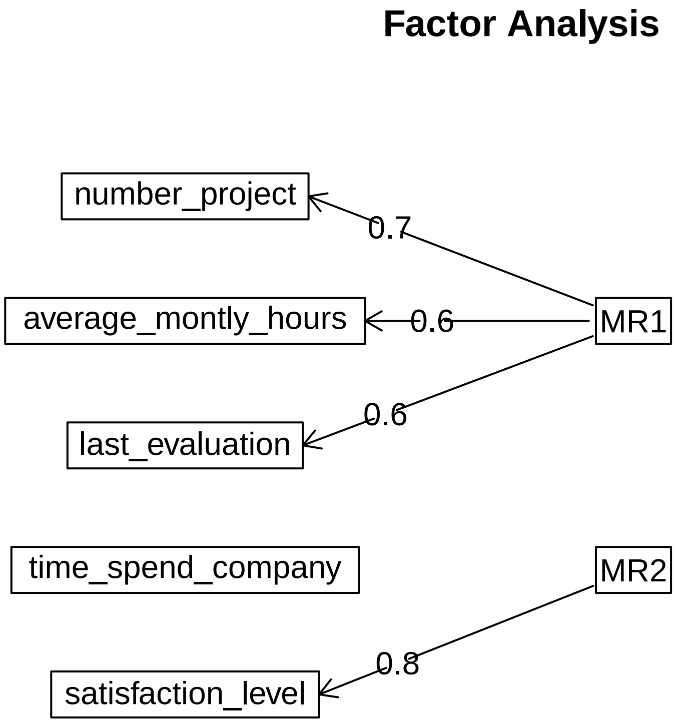
```
## Selecting tuning parameters
## Fitting alpha = 0.153, lambda = 0.00124 on full training set

## Warning in nominalTrainWorkflow(x = x, y = y, wts = weights, info = trainInfo, :
## There were missing values in resampled performance measures.

## Aggregating results

## Warning in train.default(x, y, weights = w, ...): missing values found in
## aggregated results
```



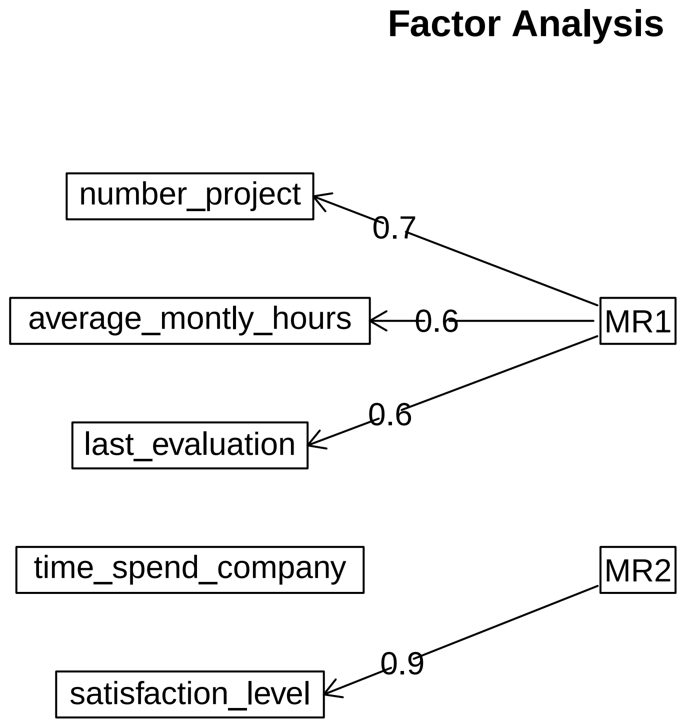
```
## Selecting tuning parameters
## Fitting alpha = 0.153, lambda = 0.00124 on full training set

## Warning in nominalTrainWorkflow(x = x, y = y, wts = weights, info = trainInfo, :
## There were missing values in resampled performance measures.

## Aggregating results

## Warning in train.default(x, y, weights = w, ...): missing values found in
## aggregated results

## Selecting tuning parameters
## Fitting alpha = 0.153, lambda = 0.00124 on full training set

## Warning in nominalTrainWorkflow(x = x, y = y, wts = weights, info = trainInfo, :
## There were missing values in resampled performance measures.

## Aggregating results

## Warning in train.default(x, y, weights = w, ...): missing values found in
## aggregated results
```



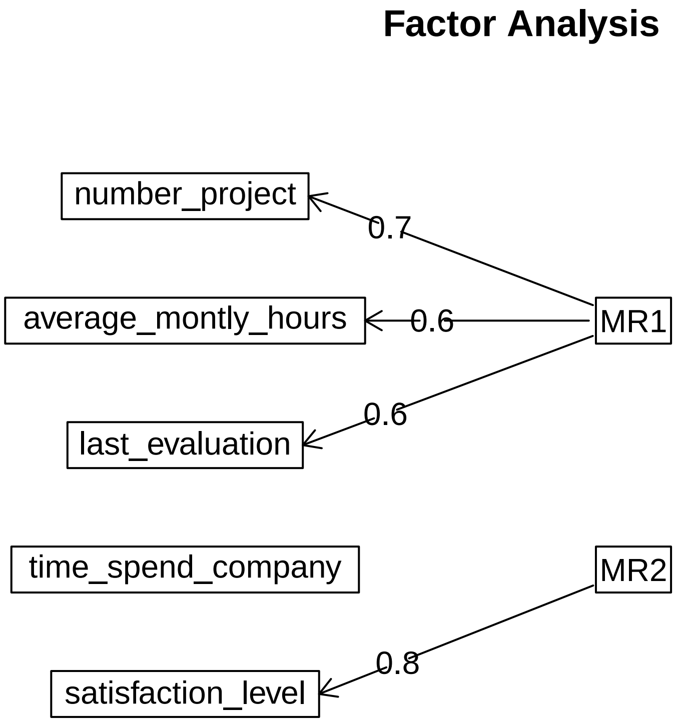
```
## Selecting tuning parameters
## Fitting alpha = 0.572, lambda = 0.042 on full training set

## Warning in nominalTrainWorkflow(x = x, y = y, wts = weights, info = trainInfo, :
## There were missing values in resampled performance measures.

## Aggregating results
## Selecting tuning parameters
## Fitting mtry = 10 on full training set

## Warning in nominalTrainWorkflow(x = x, y = y, wts = weights, info = trainInfo, :
## There were missing values in resampled performance measures.

## Aggregating results
## Selecting tuning parameters
## Fitting mtry = 10 on full training set

## Warning in nominalTrainWorkflow(x = x, y = y, wts = weights, info = trainInfo, :
## There were missing values in resampled performance measures.
```



```
## Aggregating results
## Selecting tuning parameters
## Fitting mtry = 10 on full training set

## Warning in nominalTrainWorkflow(x = x, y = y, wts = weights, info = trainInfo, :
## There were missing values in resampled performance measures.

## Aggregating results
## Selecting tuning parameters
## Fitting mtry = 8 on full training set

## Warning in nominalTrainWorkflow(x = x, y = y, wts = weights, info = trainInfo, :
## There were missing values in resampled performance measures.

## Aggregating results

## Warning in train.default(x, y, weights = w, ...): missing values found in
## aggregated results
```

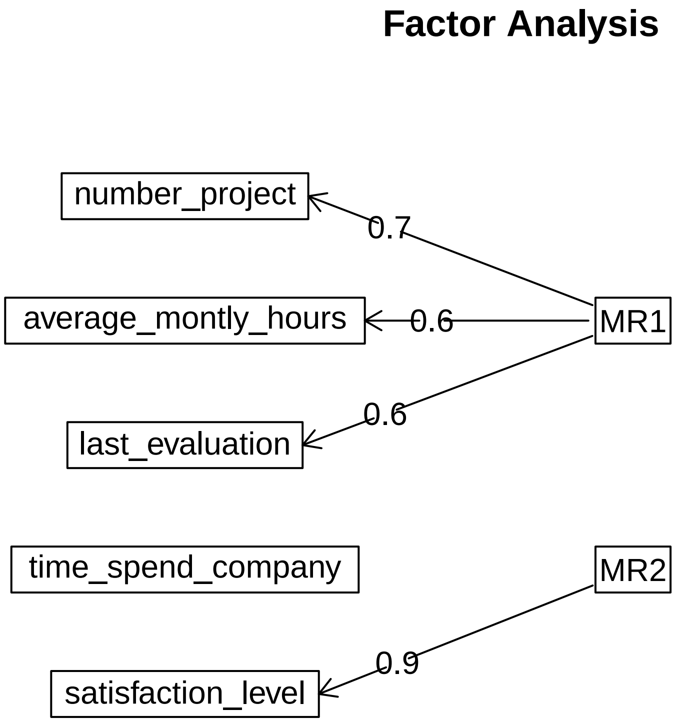


```
## Selecting tuning parameters
## Fitting mtry = 12 on full training set

## Aggregating results
## Selecting tuning parameters
## Fitting k = 3 on full training set

## Aggregating results
## Selecting tuning parameters
## Fitting k = 3 on full training set
```



```
## Aggregating results
## Selecting tuning parameters
## Fitting k = 3 on full training set
```

```
## Aggregating results
## Selecting tuning parameters
## Fitting k = 3 on full training set
```



```
## Aggregating results
## Selecting tuning parameters
## Fitting k = 5 on full training set
```

```
# Print CV results
for (m in models) {
  cat(rep('\n', 3))
  print(get(paste0(m, '_cv')))
  cat(rep('\n', 3))
}
```

```
##
##
##
## $‘1‘
## $‘1‘$model
## glmnet
##
## 9601 samples
##   15 predictor
##    2 classes: ’yes’, ’no’
##
## No pre-processing
## Resampling: Cross-Validated (5 fold, repeated 2 times)
## Summary of sample sizes: 7680, 7681, 7681, 7681, 7681, 7681, ...
## Resampling results across tuning parameters:
##
##   alpha       lambda       AUC        Precision  Recall     F
##   0.03431740  0.089848573  0.5772080  0.7876377  0.1854710  0.2998691
##   0.08249613  0.360573416  0.5833627        NaN  0.0000000        NaN
##   0.11032045  5.835644047  0.0000000        NaN  0.0000000        NaN
##   0.14298028  5.900723701  0.0000000        NaN  0.0000000        NaN
##   0.15333117  0.001237096  0.5727423  0.5898577  0.2865355  0.3853157
##   0.24414545  0.041843375  0.5832865  0.6731357  0.2303149  0.3429387
##   0.24717296  0.005363840  0.5746734  0.5736787  0.2567858  0.3545295
##   0.29186267  0.038180467  0.5838735  0.6618070  0.2318462  0.3432183
##   0.37615343  0.002088687  0.5737165  0.5849002  0.2786604  0.3771301
##   0.57222747  0.041985838  0.5869645  0.7086693  0.2318457  0.3491847
##   0.73301844  2.449870885  0.0000000        NaN  0.0000000        NaN
##   0.77100396  0.191331587  0.5736713        NaN  0.0000000        NaN
##   0.78001638  0.014048926  0.5836553  0.5934982  0.2392822  0.3409883
##   0.96830053  0.005986254  0.5790393  0.5678478  0.2440952  0.3413153
##   0.98979898  1.565802338  0.0000000        NaN  0.0000000        NaN
##
## F was used to select the optimal model using the largest value.
## The final values used for the model were alpha = 0.1533312 and lambda
##  = 0.001237096.
##
## $‘1‘$f1_val
## [1] 0.4175084
##
## $‘1‘$confm
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  yes   no
##        yes  186  134
##        no   385 1694
##
##                Accuracy : 0.7837
##                  95% CI : (0.7666, 0.8)
##     No Information Rate : 0.762
##     P-Value [Acc > NIR] : 0.006362
##
##                   Kappa : 0.2974
##
##  Mcnemar’s Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.32574
##             Specificity : 0.92670
##          Pos Pred Value : 0.58125
##          Neg Pred Value : 0.81481
##              Prevalence : 0.23802
##          Detection Rate : 0.07753
##    Detection Prevalence : 0.13339
##       Balanced Accuracy : 0.62622
##
##        ’Positive’ Class : yes
##
```

```
##
##
## $‘2‘
## $‘2‘$model
## glmnet
##
## 9600 samples
##   15 predictor
##    2 classes: ’yes’, ’no’
##
## No pre-processing
## Resampling: Cross-Validated (5 fold, repeated 2 times)
## Summary of sample sizes: 7680, 7680, 7680, 7680, 7680, 7680, ...
## Resampling results across tuning parameters:
##
##   alpha       lambda       AUC        Precision  Recall     F
##   0.03431740  0.089848573  0.5807717  0.8006113  0.2107221  0.3332538
##   0.08249613  0.360573416  0.5792088        NaN  0.0000000        NaN
##   0.11032045  5.835644047  0.0000000        NaN  0.0000000        NaN
##   0.14298028  5.900723701  0.0000000        NaN  0.0000000        NaN
##   0.15333117  0.001237096  0.5761376  0.5941547  0.2820569  0.3821086
##   0.24414545  0.041843375  0.5841563  0.6925909  0.2413567  0.3576818
##   0.24717296  0.005363840  0.5780074  0.5851413  0.2621444  0.3617453
##   0.29186267  0.038180467  0.5844313  0.6814771  0.2415755  0.3564603
##   0.37615343  0.002088687  0.5770560  0.5890117  0.2746171  0.3741422
##   0.57222747  0.041985838  0.5843376  0.7184135  0.2413567  0.3610837
##   0.73301844  2.449870885  0.0000000        NaN  0.0000000        NaN
##   0.77100396  0.191331587  0.5660670        NaN  0.0000000        NaN
##   0.78001638  0.014048926  0.5844165  0.6008996  0.2461707  0.3490103
##   0.96830053  0.005986254  0.5812564  0.5764578  0.2503282  0.3488117
##   0.98979898  1.565802338  0.0000000        NaN  0.0000000        NaN
##
## F was used to select the optimal model using the largest value.
## The final values used for the model were alpha = 0.1533312 and lambda
##  = 0.001237096.
##
## $‘2‘$f1_val
## [1] 0.372549
##
## $‘2‘$confm
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  yes   no
##        yes  171  175
##        no   401 1653
##
##                Accuracy : 0.76
##                  95% CI : (0.7424, 0.777)
##     No Information Rate : 0.7617
##     P-Value [Acc > NIR] : 0.5869
##
##                   Kappa : 0.2351
##
```

```
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.29895
##             Specificity : 0.90427
##          Pos Pred Value : 0.49422
##          Neg Pred Value : 0.80477
##              Prevalence : 0.23833
##          Detection Rate : 0.07125
##    Detection Prevalence : 0.14417
##       Balanced Accuracy : 0.60161
##
##        'Positive' Class : yes
##
##
##
## $`3`
## $`3`$model
## glmnet
##
## 9600 samples
##   15 predictor
##    2 classes: 'yes', 'no'
##
## No pre-processing
## Resampling: Cross-Validated (5 fold, repeated 2 times)
## Summary of sample sizes: 7679, 7680, 7680, 7680, 7681, 7680, ...
## Resampling results across tuning parameters:
##
##   alpha       lambda       AUC        Precision  Recall     F
##   0.03431740  0.089848573  0.5839719  0.8064638  0.2174085  0.3417013
##   0.08249613  0.360573416  0.5843289        NaN  0.0000000        NaN
##   0.11032045  5.835644047  0.0000000        NaN  0.0000000        NaN
##   0.14298028  5.900723701  0.0000000        NaN  0.0000000        NaN
##   0.15333117  0.001237096  0.5796851  0.5938845  0.2882631  0.3872137
##   0.24414545  0.041843375  0.5866031  0.6882912  0.2482461  0.3639817
##   0.24717296  0.005363840  0.5812199  0.5852592  0.2679273  0.3668093
##   0.29186267  0.038180467  0.5868749  0.6769617  0.2497778  0.3639797
##   0.37615343  0.002088687  0.5803097  0.5906427  0.2830143  0.3818531
##   0.57222747  0.041985838  0.5878423  0.7165020  0.2471539  0.3665754
##   0.73301844  2.449870885  0.0000000        NaN  0.0000000        NaN
##   0.77100396  0.191331587  0.5695331        NaN  0.0000000        NaN
##   0.78001638  0.014048926  0.5867639  0.6015270  0.2543697  0.3568058
##   0.96830053  0.005986254  0.5844679  0.5821560  0.2594006  0.3581493
##   0.98979898  1.565802338  0.0000000        NaN  0.0000000        NaN
##
## F was used to select the optimal model using the largest value.
## The final values used for the model were alpha = 0.1533312 and lambda
##  = 0.001237096.
##
## $`3`$f1_val
## [1] 0.3917749
##
## $`3`$confm
## Confusion Matrix and Statistics
```

```
##
##           Reference
## Prediction  yes   no
##        yes  181  172
##        no   390 1657
##
##                Accuracy : 0.7658
##                  95% CI : (0.7484, 0.7827)
##     No Information Rate : 0.7621
##     P-Value [Acc > NIR] : 0.3431
##
##                   Kappa : 0.2566
##
##  Mcnemar’s Test P-Value : <2e-16
##
##             Sensitivity : 0.31699
##             Specificity : 0.90596
##          Pos Pred Value : 0.51275
##          Neg Pred Value : 0.80948
##              Prevalence : 0.23792
##          Detection Rate : 0.07542
##    Detection Prevalence : 0.14708
##       Balanced Accuracy : 0.61147
##
##        ’Positive’ Class : yes
##
##
##
## $‘4‘
## $‘4‘$model
## glmnet
##
## 9600 samples
##   15 predictor
##    2 classes: ’yes’, ’no’
##
## No pre-processing
## Resampling: Cross-Validated (5 fold, repeated 2 times)
## Summary of sample sizes: 7679, 7680, 7680, 7680, 7681, 7680, ...
## Resampling results across tuning parameters:
##
##   alpha       lambda       AUC        Precision  Recall     F
##   0.03431740  0.089848573  0.5846583  0.7855777  0.2038523  0.3231638
##   0.08249613  0.360573416  0.5846747        NaN  0.0000000        NaN
##   0.11032045  5.835644047  0.0000000        NaN  0.0000000        NaN
##   0.14298028  5.900723701  0.0000000        NaN  0.0000000        NaN
##   0.15333117  0.001237096  0.5811456  0.5988584  0.2928717  0.3928077
##   0.24414545  0.041843375  0.5887475  0.6769320  0.2410275  0.3550853
##   0.24717296  0.005363840  0.5829132  0.5885299  0.2703396  0.3700639
##   0.29186267  0.038180467  0.5891996  0.6713757  0.2445276  0.3580867
##   0.37615343  0.002088687  0.5820901  0.5944076  0.2856535  0.3853012
##   0.57222747  0.041985838  0.5890402  0.7036780  0.2421225  0.3597007
##   0.73301844  2.449870885  0.0000000        NaN  0.0000000        NaN
##   0.77100396  0.191331587  0.5727108        NaN  0.0000000        NaN
##   0.78001638  0.014048926  0.5888988  0.6018243  0.2517458  0.3547284
##   0.96830053  0.005986254  0.5857254  0.5808015  0.2580901  0.3570937
##   0.98979898  1.565802338  0.0000000        NaN  0.0000000        NaN
##
## F was used to select the optimal model using the largest value.
## The final values used for the model were alpha = 0.1533312 and lambda
##  = 0.001237096.
##
## $‘4‘$f1_val
## [1] 0.3471616
##
## $‘4‘$confm
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  yes   no
##        yes  159  186
##        no   412 1643
##
##                Accuracy : 0.7508
##                  95% CI : (0.733, 0.768)
##     No Information Rate : 0.7621
##     P-Value [Acc > NIR] : 0.9058
##
##                   Kappa : 0.2046
##
##  Mcnemar’s Test P-Value : <2e-16
##
##             Sensitivity : 0.27846
##             Specificity : 0.89831
##          Pos Pred Value : 0.46087
##          Neg Pred Value : 0.79951
##              Prevalence : 0.23792
##          Detection Rate : 0.06625
##    Detection Prevalence : 0.14375
##       Balanced Accuracy : 0.58838
##
##        ’Positive’ Class : yes
##
##
##
## $‘5‘
## $‘5‘$model
## glmnet
##
## 9599 samples
##   15 predictor
##    2 classes: ’yes’, ’no’
##
## No pre-processing
## Resampling: Cross-Validated (5 fold, repeated 2 times)
## Summary of sample sizes: 7679, 7679, 7679, 7679, 7680, 7679, ...
## Resampling results across tuning parameters:
##
```

```
##   alpha       lambda       AUC        Precision  Recall     F
##   0.03431740  0.089848573  0.5870402  0.8210220  0.2284464  0.3570610
##   0.08249613  0.360573416  0.5911446        NaN  0.0000000        NaN
##   0.11032045  5.835644047  0.0000000        NaN  0.0000000        NaN
##   0.14298028  5.900723701  0.0000000        NaN  0.0000000        NaN
##   0.15333117  0.001237096  0.5832537  0.5727365  0.2724289  0.3689496
##   0.24414545  0.041843375  0.5933157  0.6930946  0.2501094  0.3672583
##   0.24717296  0.005363840  0.5855347  0.5762286  0.2636761  0.3615385
##   0.29186267  0.038180467  0.5940984  0.6801366  0.2507659  0.3661709
##   0.37615343  0.002088687  0.5844254  0.5725664  0.2704595  0.3670794
##   0.57222747  0.041985838  0.5950446  0.7274197  0.2487965  0.3703174
##   0.73301844  2.449870885  0.0000000        NaN  0.0000000        NaN
##   0.77100396  0.191331587  0.5819488        NaN  0.0000000        NaN
##   0.78001638  0.014048926  0.5939276  0.6109304  0.2562363  0.3607058
##   0.96830053  0.005986254  0.5905502  0.5808563  0.2606127  0.3595125
##   0.98979898  1.565802338  0.0000000        NaN  0.0000000        NaN
##
## F was used to select the optimal model using the largest value.
## The final values used for the model were alpha = 0.5722275 and lambda
##  = 0.04198584.
##
## $`5`$f1_val
## [1] 0.06586826
##
## $`5`$confm
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  yes   no
##        yes   22   74
##        no   550 1755
##
##                Accuracy : 0.7401
##                  95% CI : (0.7221, 0.7576)
##     No Information Rate : 0.7618
##     P-Value [Acc > NIR] : 0.9937
##
##                   Kappa : -0.0028
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.038462
##             Specificity : 0.959541
##          Pos Pred Value : 0.229167
##          Neg Pred Value : 0.761388
##              Prevalence : 0.238234
##          Detection Rate : 0.009163
##    Detection Prevalence : 0.039983
##       Balanced Accuracy : 0.499001
##
##        'Positive' Class : yes
##
##
##
```

```
##
##
##
##
##
##
## $‘1‘
## $‘1‘$model
## Random Forest
##
## 9601 samples
##    6 predictor
##    2 classes: ’yes’, ’no’
##
## No pre-processing
## Resampling: Cross-Validated (5 fold, repeated 2 times)
## Summary of sample sizes: 7680, 7681, 7681, 7681, 7681, 7681, ...
## Resampling results across tuning parameters:
##
##   mtry  AUC        Precision  Recall      F
##    1    0.8340516  1.0000000  0.01356005  0.06386678
##    3    0.9395373  0.9378831  0.84011543  0.88624213
##    4    0.9462432  0.9343864  0.86701480  0.89937838
##    7    0.7274992  0.9333987  0.91207275  0.92255994
##    8    0.6474083  0.9345026  0.92257365  0.92845568
##   10    0.5594492  0.9337276  0.92607426  0.92983167
##   11    0.5533944  0.9330670  0.92607331  0.92951403
##   12    0.5311951  0.9333159  0.92607379  0.92962965
##   14    0.4890939  0.9302047  0.92585497  0.92796681
##   15    0.4739264  0.9289500  0.92585497  0.92735754
##
## F was used to select the optimal model using the largest value.
## The final value used for the model was mtry = 10.
##
## $‘1‘$f1_val
## [1] 0.109834
##
## $‘1‘$confm
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  yes   no
##        yes   43  169
##        no   528 1659
##
##                Accuracy : 0.7095
##                  95% CI : (0.6908, 0.7276)
##     No Information Rate : 0.762
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : -0.0219
##
##  Mcnemar’s Test P-Value : <2e-16
##
```

```
##             Sensitivity : 0.07531
##             Specificity : 0.90755
##          Pos Pred Value : 0.20283
##          Neg Pred Value : 0.75857
##              Prevalence : 0.23802
##          Detection Rate : 0.01792
##    Detection Prevalence : 0.08837
##       Balanced Accuracy : 0.49143
##
##        ’Positive’ Class : yes
##
##
##
## $‘2‘
## $‘2‘$model
## Random Forest
##
## 9600 samples
##    6 predictor
##    2 classes: ’yes’, ’no’
##
## No pre-processing
## Resampling: Cross-Validated (5 fold, repeated 2 times)
## Summary of sample sizes: 7680, 7680, 7680, 7680, 7680, 7680, ...
## Resampling results across tuning parameters:
##
##   mtry  AUC        Precision  Recall       F
##    1    0.8302486  1.0000000  0.002625821  0.01731016
##    3    0.9377880  0.9414006  0.834354486  0.88449297
##    4    0.9434182  0.9364035  0.865426696  0.89937172
##    7    0.7928893  0.9369335  0.911597374  0.92398344
##    8    0.7038235  0.9367652  0.921663020  0.92902274
##   10    0.6356841  0.9345435  0.925820569  0.93004998
##   11    0.6125494  0.9333441  0.925164114  0.92909673
##   12    0.6000030  0.9314347  0.924945295  0.92805959
##   14    0.5702973  0.9290084  0.924507659  0.92662005
##   15    0.5584111  0.9283400  0.924070022  0.92607973
##
## F was used to select the optimal model using the largest value.
## The final value used for the model was mtry = 10.
##
## $‘2‘$f1_val
## [1] 0.118239
##
## $‘2‘$confm
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  yes   no
##        yes   47  176
##        no   525 1652
##
##                Accuracy : 0.7079
##                 95% CI : (0.6893, 0.7261)
##     No Information Rate : 0.7617
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : -0.0179
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.08217
##             Specificity : 0.90372
##          Pos Pred Value : 0.21076
##          Neg Pred Value : 0.75884
##              Prevalence : 0.23833
##          Detection Rate : 0.01958
##    Detection Prevalence : 0.09292
##       Balanced Accuracy : 0.49294
##
##        'Positive' Class : yes
##
##
##
## $`3`
## $`3`$model
## Random Forest
##
## 9600 samples
##    6 predictor
##    2 classes: 'yes', 'no'
##
## No pre-processing
## Resampling: Cross-Validated (5 fold, repeated 2 times)
## Summary of sample sizes: 7679, 7680, 7680, 7680, 7681, 7680, ...
## Resampling results across tuning parameters:
##
##   mtry  AUC        Precision  Recall       F
##    1    0.8335961  1.0000000  0.002625821  0.02590872
##    3    0.9355564  0.9414090  0.840330425  0.88786102
##    4    0.9461599  0.9343939  0.864607321  0.89806793
##    7    0.7930380  0.9338374  0.912724910  0.92309770
##    8    0.7239428  0.9340612  0.922347663  0.92809013
##   10    0.6574238  0.9329622  0.927160234  0.92996755
##   11    0.6388138  0.9322938  0.926285439  0.92919648
##   12    0.6149003  0.9303613  0.926723075  0.92844170
##   14    0.5833562  0.9296924  0.926941416  0.92824319
##   15    0.5676825  0.9276230  0.926503301  0.92699225
##
## F was used to select the optimal model using the largest value.
## The final value used for the model was mtry = 10.
##
## $`3`$f1_val
## [1] 0.1985112
##
## $`3`$confm
## Confusion Matrix and Statistics
##
```

```
##           Reference
## Prediction  yes   no
##        yes   80  154
##        no   491 1675
##
##                Accuracy : 0.7312
##                  95% CI : (0.713, 0.7489)
##     No Information Rate : 0.7621
##     P-Value [Acc > NIR] : 0.9998
##
##                   Kappa : 0.0701
##
##  Mcnemar’s Test P-Value : <2e-16
##
##             Sensitivity : 0.14011
##             Specificity : 0.91580
##          Pos Pred Value : 0.34188
##          Neg Pred Value : 0.77331
##              Prevalence : 0.23792
##          Detection Rate : 0.03333
##    Detection Prevalence : 0.09750
##       Balanced Accuracy : 0.52795
##
##        ’Positive’ Class : yes
##
##
##
## $‘4‘
## $‘4‘$model
## Random Forest
##
## 9600 samples
##    6 predictor
##    2 classes: ’yes’, ’no’
##
## No pre-processing
## Resampling: Cross-Validated (5 fold, repeated 2 times)
## Summary of sample sizes: 7679, 7680, 7680, 7680, 7681, 7680, ...
## Resampling results across tuning parameters:
##
##   mtry  AUC        Precision  Recall      F
##    1    0.8298945  1.0000000  0.01159499  0.04495334
##    3    0.9331156  0.9373460  0.83508404  0.88302029
##    4    0.9423735  0.9311381  0.86832771  0.89847954
##    7    0.7982230  0.9277492  0.90747852  0.91739694
##    8    0.7065197  0.9287166  0.91491405  0.92168138
##   10    0.6305422  0.9255611  0.91600910  0.92064919
##   11    0.6072451  0.9264572  0.91469666  0.92042937
##   12    0.5854844  0.9228110  0.91425951  0.91838695
##   14    0.5548390  0.9213188  0.91535312  0.91817190
##   15    0.5464007  0.9202180  0.91425807  0.91708677
##
## F was used to select the optimal model using the largest value.
## The final value used for the model was mtry = 8.
```

```
##
## $‘4‘$f1_val
## [1] 0.08994709
##
## $‘4‘$confm
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  yes   no
##        yes   34  152
##        no   537 1677
##
##                Accuracy : 0.7129
##                  95% CI : (0.6944, 0.731)
##     No Information Rate : 0.7621
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : -0.0307
##
##  Mcnemar’s Test P-Value : <2e-16
##
##             Sensitivity : 0.05954
##             Specificity : 0.91689
##          Pos Pred Value : 0.18280
##          Neg Pred Value : 0.75745
##              Prevalence : 0.23792
##          Detection Rate : 0.01417
##    Detection Prevalence : 0.07750
##       Balanced Accuracy : 0.48822
##
##        ’Positive’ Class : yes
##
##
##
## $‘5‘
## $‘5‘$model
## Random Forest
##
## 9599 samples
##    6 predictor
##    2 classes: ’yes’, ’no’
##
## No pre-processing
## Resampling: Cross-Validated (5 fold, repeated 2 times)
## Summary of sample sizes: 7679, 7679, 7679, 7679, 7680, 7679, ...
## Resampling results across tuning parameters:
##
##   mtry  AUC        Precision  Recall     F
##    1    0.8325350        NaN  0.0000000        NaN
##    3    0.9382353  0.9332901  0.8439825  0.8862211
##    4    0.9485859  0.9284373  0.8778993  0.9023586
##    7    0.7914873  0.9334663  0.9164114  0.9248003
##    8    0.7227009  0.9324627  0.9199125  0.9260960
##   10    0.6230053  0.9326191  0.9223195  0.9273996
##   11    0.6042595  0.9325230  0.9242888  0.9283550
##   12    0.5859090  0.9317753  0.9253829  0.9285325
##   14    0.5573494  0.9302556  0.9238512  0.9270012
##   15    0.5367351  0.9286138  0.9238512  0.9261866
##
## F was used to select the optimal model using the largest value.
## The final value used for the model was mtry = 12.
##
## $‘5‘$f1_val
## [1] 0.110971
##
## $‘5‘$confm
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  yes   no
##        yes   44  177
##        no   528 1652
##
##                Accuracy : 0.7064
##                  95% CI : (0.6877, 0.7245)
##     No Information Rate : 0.7618
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : -0.0252
##
##  Mcnemar’s Test P-Value : <2e-16
##
##             Sensitivity : 0.07692
##             Specificity : 0.90323
##          Pos Pred Value : 0.19910
##          Neg Pred Value : 0.75780
##              Prevalence : 0.23823
##          Detection Rate : 0.01833
##    Detection Prevalence : 0.09204
##       Balanced Accuracy : 0.49007
##
##        ’Positive’ Class : yes
##
##
##
##
##
##
##
##
##
## $‘1‘
## $‘1‘$model
## k-Nearest Neighbors
##
## 9601 samples
##   15 predictor
##    2 classes: ’yes’, ’no’
```

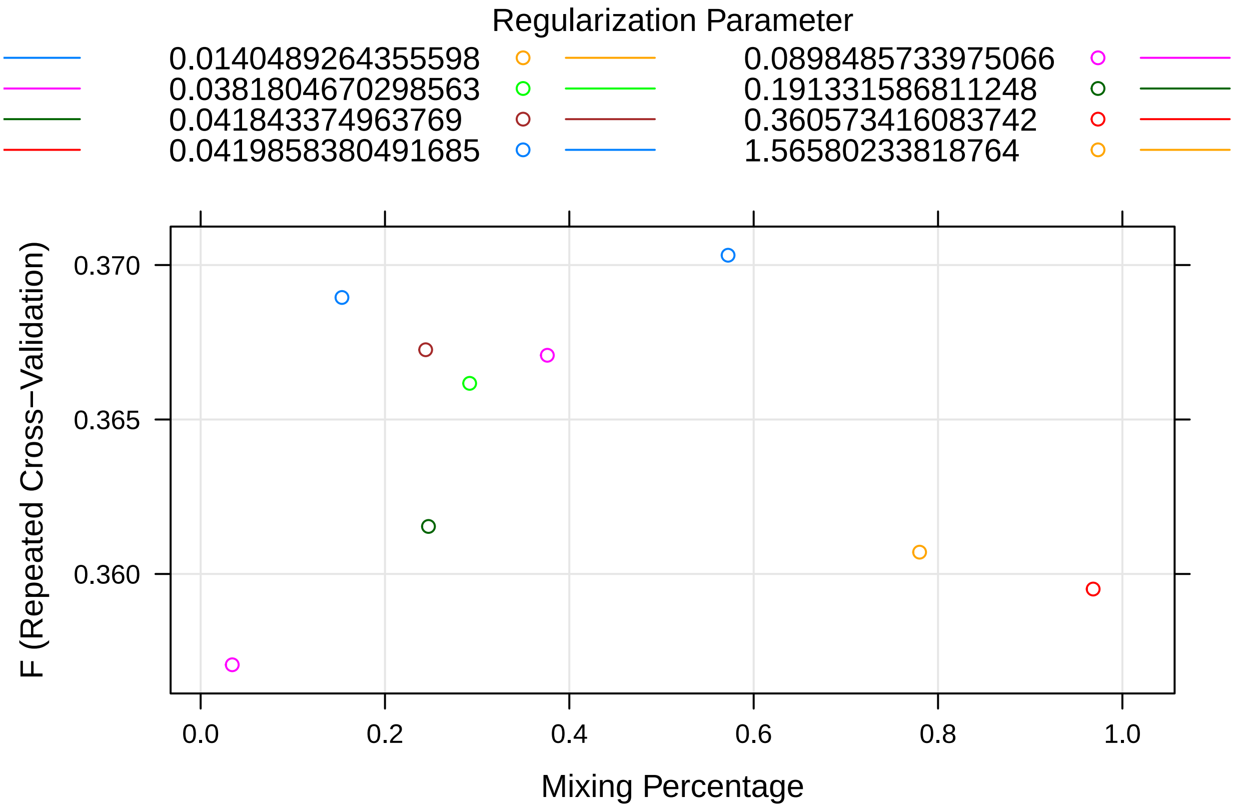
```
##
## No pre-processing
## Resampling: Cross-Validated (5 fold, repeated 2 times)
## Summary of sample sizes: 7680, 7681, 7681, 7681, 7681, 7681, ...
## Resampling results across tuning parameters:
##
##   k   AUC        Precision  Recall     F
##    2  0.1361611  0.8233147  0.8805792  0.8508726
##    3  0.2020149  0.8326205  0.8707285  0.8511398
##    4  0.2417961  0.8305367  0.8543276  0.8421786
##    5  0.2683598  0.8453635  0.8464545  0.8458770
##   10  0.4036427  0.8410745  0.8475471  0.8442668
##
## F was used to select the optimal model using the largest value.
## The final value used for the model was k = 3.
##
## $‘1‘$f1_val
## [1] 0.3675958
##
## $‘1‘$confm
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  yes   no
##        yes  214  365
##        no   357 1463
##
##                Accuracy : 0.699
##                  95% CI : (0.6802, 0.7174)
##     No Information Rate : 0.762
##     P-Value [Acc > NIR] : 1.0000
##
##                   Kappa : 0.1743
##
##  Mcnemar’s Test P-Value : 0.7945
##
##             Sensitivity : 0.3748
##             Specificity : 0.8003
##          Pos Pred Value : 0.3696
##          Neg Pred Value : 0.8038
##              Prevalence : 0.2380
##          Detection Rate : 0.0892
##    Detection Prevalence : 0.2414
##       Balanced Accuracy : 0.5876
##
##        ’Positive’ Class : yes
##
##
##
## $‘2‘
## $‘2‘$model
## k-Nearest Neighbors
##
## 9600 samples
##   15 predictor
##    2 classes: 'yes', 'no'
##
## No pre-processing
## Resampling: Cross-Validated (5 fold, repeated 2 times)
## Summary of sample sizes: 7680, 7680, 7680, 7680, 7680, 7680, ...
## Resampling results across tuning parameters:
##
##   k   AUC        Precision  Recall     F
##    2  0.1441727  0.8172311  0.8704595  0.8427944
##    3  0.2084413  0.8314812  0.8634573  0.8469245
##    4  0.2498527  0.8352245  0.8560175  0.8453306
##    5  0.2793481  0.8428885  0.8492341  0.8457637
##   10  0.4087602  0.8438838  0.8485777  0.8460734
##
## F was used to select the optimal model using the largest value.
## The final value used for the model was k = 3.
##
## $`2`$f1_val
## [1] 0.4007156
##
## $`2`$confm
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  yes   no
##        yes  221  330
##        no   351 1498
##
##                Accuracy : 0.7163
##                  95% CI : (0.6977, 0.7342)
##     No Information Rate : 0.7617
##     P-Value [Acc > NIR] : 1.0000
##
##                   Kappa : 0.2085
##
##  Mcnemar's Test P-Value : 0.4434
##
##             Sensitivity : 0.38636
##             Specificity : 0.81947
##          Pos Pred Value : 0.40109
##          Neg Pred Value : 0.81017
##              Prevalence : 0.23833
##          Detection Rate : 0.09208
##    Detection Prevalence : 0.22958
##       Balanced Accuracy : 0.60292
##
##        'Positive' Class : yes
##
##
##
## $`3`
## $`3`$model
## k-Nearest Neighbors
```

```
##
## 9600 samples
##   15 predictor
##    2 classes: ’yes’, ’no’
##
## No pre-processing
## Resampling: Cross-Validated (5 fold, repeated 2 times)
## Summary of sample sizes: 7679, 7680, 7680, 7680, 7681, 7680, ...
## Resampling results across tuning parameters:
##
##   k   AUC        Precision  Recall     F
##    2  0.1363338  0.8200137  0.8790426  0.8484538
##    3  0.2078433  0.8324206  0.8722631  0.8518093
##    4  0.2527100  0.8349515  0.8584823  0.8464787
##    5  0.2896776  0.8414663  0.8460144  0.8436588
##   10  0.4180243  0.8501025  0.8348638  0.8423424
##
## F was used to select the optimal model using the largest value.
## The final value used for the model was k = 3.
##
## $‘3‘$f1_val
## [1] 0.4333895
##
## $‘3‘$confm
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  yes   no
##        yes  261  356
##        no   310 1473
##
##                Accuracy : 0.7225
##                  95% CI : (0.7041, 0.7403)
##     No Information Rate : 0.7621
##     P-Value [Acc > NIR] : 1.00000
##
##                   Kappa : 0.2554
##
##  Mcnemar’s Test P-Value : 0.08121
##
##             Sensitivity : 0.4571
##             Specificity : 0.8054
##          Pos Pred Value : 0.4230
##          Neg Pred Value : 0.8261
##              Prevalence : 0.2379
##          Detection Rate : 0.1087
##    Detection Prevalence : 0.2571
##       Balanced Accuracy : 0.6312
##
##        ’Positive’ Class : yes
##
##
##
## $‘4‘
## $‘4‘$model
## k-Nearest Neighbors
##
## 9600 samples
##   15 predictor
##    2 classes: ’yes’, ’no’
##
## No pre-processing
## Resampling: Cross-Validated (5 fold, repeated 2 times)
## Summary of sample sizes: 7679, 7680, 7680, 7680, 7681, 7680, ...
## Resampling results across tuning parameters:
##
##   k   AUC        Precision  Recall     F
##    2  0.1382653  0.8136408  0.8637297  0.8377682
##    3  0.2066693  0.8325459  0.8661362  0.8488567
##    4  0.2465014  0.8350916  0.8482007  0.8414494
##    5  0.2836692  0.8444190  0.8407614  0.8423856
##   10  0.4175974  0.8415670  0.8492943  0.8451582
##
## F was used to select the optimal model using the largest value.
## The final value used for the model was k = 3.
##
## $‘4‘$f1_val
## [1] 0.4320675
##
## $‘4‘$confm
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  yes   no
##        yes  261  354
##        no   310 1475
##
##                Accuracy : 0.7233
##                  95% CI : (0.705, 0.7412)
##     No Information Rate : 0.7621
##     P-Value [Acc > NIR] : 0.99999
##
##                   Kappa : 0.2567
##
##  Mcnemar’s Test P-Value : 0.09517
##
##             Sensitivity : 0.4571
##             Specificity : 0.8065
##          Pos Pred Value : 0.4244
##          Neg Pred Value : 0.8263
##              Prevalence : 0.2379
##          Detection Rate : 0.1087
##    Detection Prevalence : 0.2562
##       Balanced Accuracy : 0.6318
##
##        ’Positive’ Class : yes
##
##
```

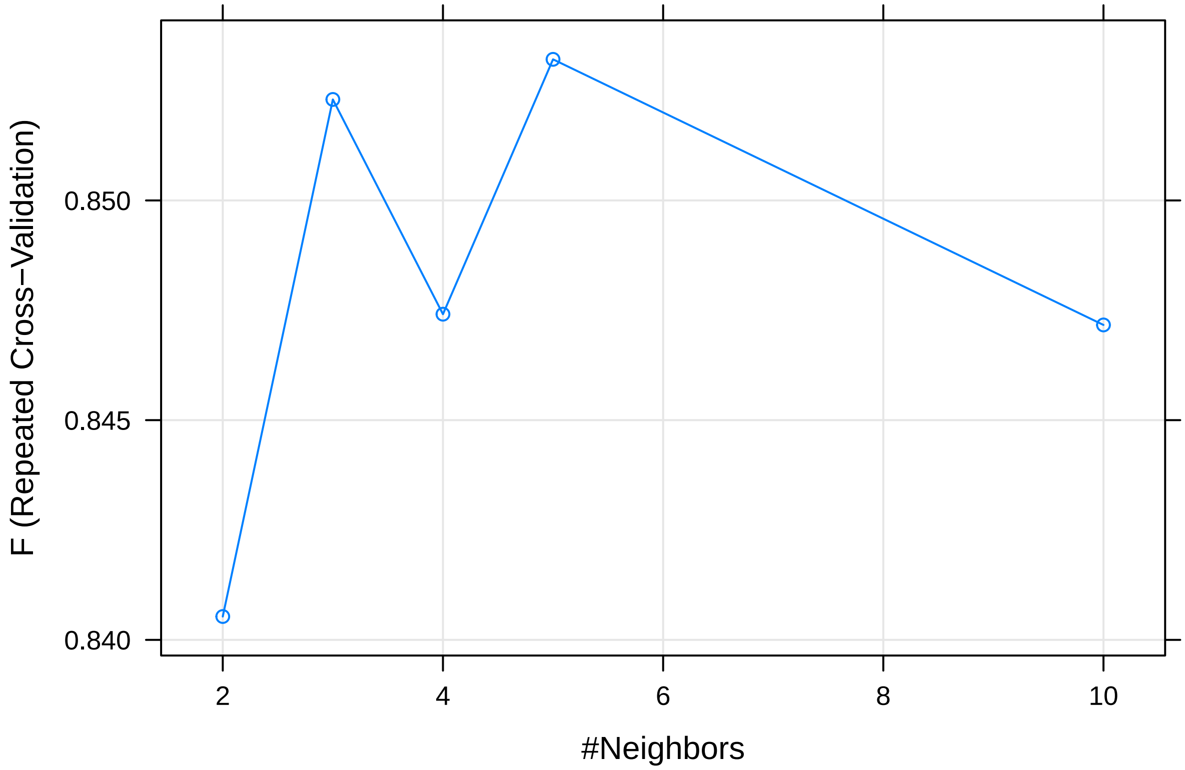
```
##
## $`5`
## $`5`$model
## k-Nearest Neighbors
##
## 9599 samples
##   15 predictor
##    2 classes: 'yes', 'no'
##
## No pre-processing
## Resampling: Cross-Validated (5 fold, repeated 2 times)
## Summary of sample sizes: 7679, 7679, 7679, 7679, 7680, 7679, ...
## Resampling results across tuning parameters:
##
##   k   AUC        Precision  Recall     F
##    2  0.1421366  0.8161043  0.8667396  0.8405315
##    3  0.2004090  0.8382867  0.8669584  0.8522979
##    4  0.2486435  0.8381891  0.8571116  0.8474112
##    5  0.2869835  0.8489578  0.8577681  0.8532106
##   10  0.4402926  0.8390127  0.8557987  0.8471668
##
## F was used to select the optimal model using the largest value.
## The final value used for the model was k = 5.
##
## $`5`$f1_val
## [1] 0.4092466
##
## $`5`$confm
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  yes   no
##        yes  238  364
##        no   334 1465
##
##                Accuracy : 0.7093
##                  95% CI : (0.6907, 0.7274)
##     No Information Rate : 0.7618
##     P-Value [Acc > NIR] : 1.0000
##
##                   Kappa : 0.2132
##
##  Mcnemar's Test P-Value : 0.2724
##
##             Sensitivity : 0.41608
##             Specificity : 0.80098
##          Pos Pred Value : 0.39535
##          Neg Pred Value : 0.81434
##              Prevalence : 0.23823
##          Detection Rate : 0.09913
##    Detection Prevalence : 0.25073
##       Balanced Accuracy : 0.60853
##
##        'Positive' Class : yes
```

```
##
##
##
##
##
##
```

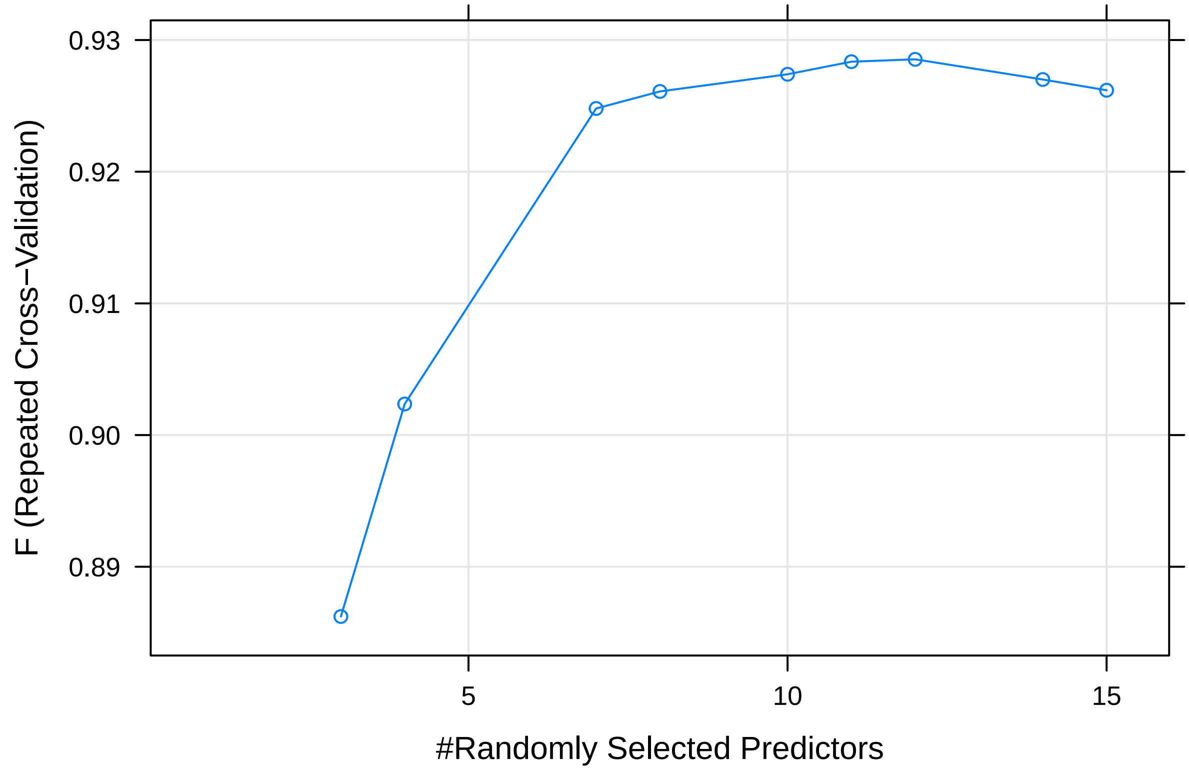
```
# Choose best model in terms of F1 score on test dataset
for (m in models) {
  f <- get(paste0(m, '_cv'))[[1]]$f1_val
  n <- 1

  for (i in 2 : n_fold) {
    f1 <- get(paste0(m, '_cv'))[[i]]$f1_val
    if (f1 > f) {
      f <- f1
      n <- i
    }
  }

  assign(paste0(m, '_best'), get(paste0(m, '_cv'))[[i]])
}



# Record best model for each method
#results <- as.data.frame(cbind(glmnet_best, svmLinear_best, knn_best, rf_best))
results <- as.data.frame(cbind(glmnet_best, knn_best, rf_best))

plot(results$glmnet_best$model)
```



```
#plot(results$svmLinear_best$model)
plot(results$knn_best$model)
```



```
plot(results$rf_best$model)
```



```r
for (i in 1 : 3) {
  cat(rep('\n', 3))
  print(results[[i]])
  cat(rep('\n', 3))
}
```

```
##
##
##
## $model
## glmnet
##
## 9599 samples
##   15 predictor
##    2 classes: ’yes’, ’no’
##
## No pre-processing
## Resampling: Cross-Validated (5 fold, repeated 2 times)
## Summary of sample sizes: 7679, 7679, 7679, 7679, 7680, 7679, ...
## Resampling results across tuning parameters:
##
##   alpha       lambda       AUC        Precision  Recall     F
##   0.03431740  0.089848573  0.5870402  0.8210220  0.2284464  0.3570610
##   0.08249613  0.360573416  0.5911446        NaN  0.0000000        NaN
##   0.11032045  5.835644047  0.0000000        NaN  0.0000000        NaN
##   0.14298028  5.900723701  0.0000000        NaN  0.0000000        NaN
##   0.15333117  0.001237096  0.5832537  0.5727365  0.2724289  0.3689496
##   0.24414545  0.041843375  0.5933157  0.6930946  0.2501094  0.3672583
##   0.24717296  0.005363840  0.5855347  0.5762286  0.2636761  0.3615385
##   0.29186267  0.038180467  0.5940984  0.6801366  0.2507659  0.3661709
##   0.37615343  0.002088687  0.5844254  0.5725664  0.2704595  0.3670794
##   0.57222747  0.041985838  0.5950446  0.7274197  0.2487965  0.3703174
##   0.73301844  2.449870885  0.0000000        NaN  0.0000000        NaN
##   0.77100396  0.191331587  0.5819488        NaN  0.0000000        NaN
##   0.78001638  0.014048926  0.5939276  0.6109304  0.2562363  0.3607058
##   0.96830053  0.005986254  0.5905502  0.5808563  0.2606127  0.3595125
##   0.98979898  1.565802338  0.0000000        NaN  0.0000000        NaN
##
## F was used to select the optimal model using the largest value.
## The final values used for the model were alpha = 0.5722275 and lambda
##  = 0.04198584.
##
## $f1_val
## [1] 0.06586826
##
## $confm
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  yes   no
##        yes   22   74
##        no   550 1755
##
##                Accuracy : 0.7401
##                  95% CI : (0.7221, 0.7576)
##     No Information Rate : 0.7618
##     P-Value [Acc > NIR] : 0.9937
##
##                   Kappa : -0.0028
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.038462
##             Specificity : 0.959541
##          Pos Pred Value : 0.229167
##          Neg Pred Value : 0.761388
##              Prevalence : 0.238234
##          Detection Rate : 0.009163
##    Detection Prevalence : 0.039983
##       Balanced Accuracy : 0.499001
##
##        'Positive' Class : yes
##
##
##
##
##
##
##
##
```

```
## $model
## k-Nearest Neighbors
##
## 9599 samples
##   15 predictor
##    2 classes: 'yes', 'no'
##
## No pre-processing
## Resampling: Cross-Validated (5 fold, repeated 2 times)
## Summary of sample sizes: 7679, 7679, 7679, 7679, 7680, 7679, ...
## Resampling results across tuning parameters:
##
##   k   AUC        Precision  Recall     F
##    2  0.1421366  0.8161043  0.8667396  0.8405315
##    3  0.2004090  0.8382867  0.8669584  0.8522979
##    4  0.2486435  0.8381891  0.8571116  0.8474112
##    5  0.2869835  0.8489578  0.8577681  0.8532106
##   10  0.4402926  0.8390127  0.8557987  0.8471668
##
## F was used to select the optimal model using the largest value.
## The final value used for the model was k = 5.
##
## $f1_val
## [1] 0.4092466
##
## $confm
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  yes   no
##        yes  238  364
##        no   334 1465
##
##                Accuracy : 0.7093
##                  95% CI : (0.6907, 0.7274)
##     No Information Rate : 0.7618
##     P-Value [Acc > NIR] : 1.0000
##
##                   Kappa : 0.2132
##
##  Mcnemar's Test P-Value : 0.2724
##
##             Sensitivity : 0.41608
##             Specificity : 0.80098
##          Pos Pred Value : 0.39535
##          Neg Pred Value : 0.81434
##              Prevalence : 0.23823
##          Detection Rate : 0.09913
##    Detection Prevalence : 0.25073
##       Balanced Accuracy : 0.60853
##
##        'Positive' Class : yes
##
##
```

```
##
##
##
##
##
##
## $model
## Random Forest
##
## 9599 samples
##    6 predictor
##    2 classes: ’yes’, ’no’
##
## No pre-processing
## Resampling: Cross-Validated (5 fold, repeated 2 times)
## Summary of sample sizes: 7679, 7679, 7679, 7679, 7680, 7679, ...
## Resampling results across tuning parameters:
##
##   mtry  AUC        Precision  Recall     F
##    1    0.8325350        NaN  0.0000000        NaN
##    3    0.9382353  0.9332901  0.8439825  0.8862211
##    4    0.9485859  0.9284373  0.8778993  0.9023586
##    7    0.7914873  0.9334663  0.9164114  0.9248003
##    8    0.7227009  0.9324627  0.9199125  0.9260960
##   10    0.6230053  0.9326191  0.9223195  0.9273996
##   11    0.6042595  0.9325230  0.9242888  0.9283550
##   12    0.5859090  0.9317753  0.9253829  0.9285325
##   14    0.5573494  0.9302556  0.9238512  0.9270012
##   15    0.5367351  0.9286138  0.9238512  0.9261866
##
## F was used to select the optimal model using the largest value.
## The final value used for the model was mtry = 12.
##
## $f1_val
## [1] 0.110971
##
## $confm
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  yes   no
##        yes   44  177
##        no   528 1652
##
##                Accuracy : 0.7064
##                  95% CI : (0.6877, 0.7245)
##     No Information Rate : 0.7618
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : -0.0252
##
##  Mcnemar’s Test P-Value : <2e-16
##
##             Sensitivity : 0.07692
##             Specificity : 0.90323
##          Pos Pred Value : 0.19910
##          Neg Pred Value : 0.75780
##              Prevalence : 0.23823
##          Detection Rate : 0.01833
##    Detection Prevalence : 0.09204
##       Balanced Accuracy : 0.49007
##
##        'Positive' Class : yes
##
##
##
##
##
```

```
save.image("D:/Yaxin/HKBU BM/Courses/Sem 2/ECON7860 Big Data Analytics for Business (S11)/Group Project/
```