CensusPay.R

rharidas

2021-12-02

```
# Note: This script will take a while to run. In particular the knn and random forest algorithms with tuning grids will take
# more time. please be patient if you happen to execute it. The execution report is available in the github location as well

# Execute the given source code for the project
source("DatasetProcessingCode.R")
```

```
## Loading required package: tidyverse
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --
```

```
## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.2     v dplyr   1.0.7
## v tidyr   1.1.3     v stringr 1.4.0
## v readr   1.4.0     v forcats 0.5.1
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
## Loading required package: caret
```

```
## Loading required package: lattice
```

```
##
## Attaching package: 'caret'
```

```
## The following object is masked from 'package:purrr':
##
##     lift
```

```
## Loading required package: data.table
```

```
##
## Attaching package: 'data.table'
```

```
## The following objects are masked from 'package:dplyr':
##
##     between, first, last
```

```
## The following object is masked from 'package:purrr':
##
##     transpose
```

```
## Loading required package: gridExtra
```

```
##
## Attaching package: 'gridExtra'
```

```
## The following object is masked from 'package:dplyr':
##
##     combine
```

```
## Loading required package: kableExtra
```

```
##
## Attaching package: 'kableExtra'
```

```
## The following object is masked from 'package:dplyr':
##
##     group_rows
```

```
## Loading required package: epiDisplay
```

```
## Loading required package: foreign
```

```
## Loading required package: survival
```

```
##
## Attaching package: 'survival'
```

```
## The following object is masked from 'package:caret':
##
##     cluster
```

```
## Loading required package: MASS
```

```
##
## Attaching package: 'MASS'
```

```
## The following object is masked from 'package:dplyr':
##
##     select
```

```
## Loading required package: nnet
```

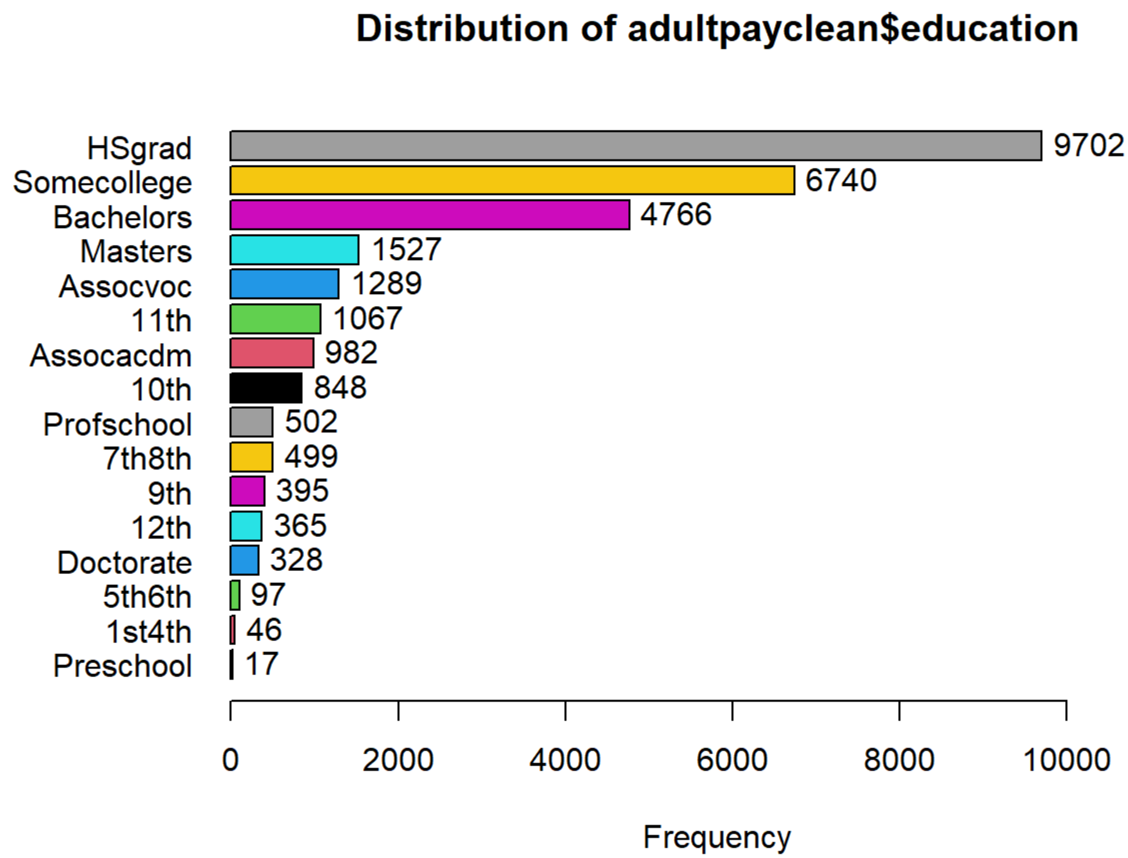
```
##
## Attaching package: 'epiDisplay'
```

```
## The following object is masked from 'package:lattice':
##
##     dotplot
```

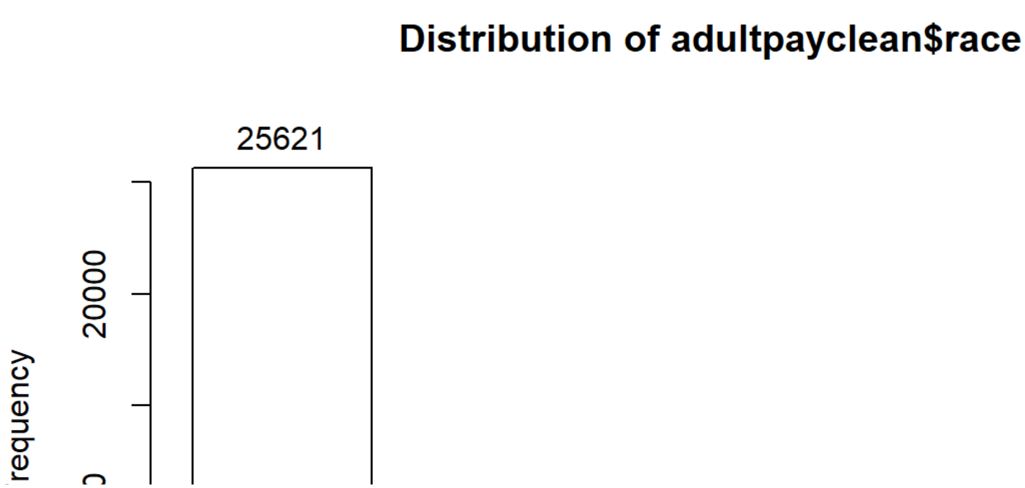
```
## The following object is masked from 'package:ggplot2':
##
##     alpha
```

```
## Warning in set.seed(1, sample.kind = "Rounding"): non-uniform 'Rounding' sampler
## used
```

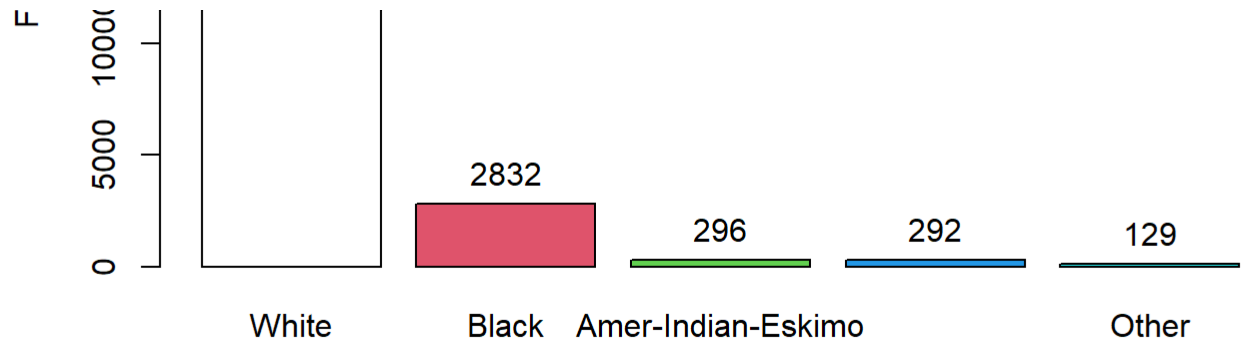
```
## Rows: 32,561
## Columns: 15
## $ age            <int> 90, 82, 66, 54, 41, 34, 38, 74, 68, 41, 45, 38, 52, 32,~
## $ workclass      <chr> "?", "Private", "?", "Private", "Private", "Private", "~
## $ fnlwgt         <int> 77053, 132870, 186061, 140359, 264663, 216864, 150601, ~
## $ education      <chr> "HS-grad", "HS-grad", "Some-college", "7th-8th", "Some-~
## $ education.num  <int> 9, 9, 10, 4, 10, 9, 6, 16, 9, 10, 16, 15, 13, 14, 16, 1~
## $ marital.status <chr> "Widowed", "Widowed", "Widowed", "Divorced", "Separated~
## $ occupation     <chr> "?", "Exec-managerial", "?", "Machine-op-inspct", "Prof~
## $ relationship   <chr> "Not-in-family", "Not-in-family", "Unmarried", "Unmarri~
## $ race           <chr> "White", "White", "Black", "White", "White", "White", "~
## $ sex            <chr> "Female", "Female", "Female", "Female", "Female", "Fema~
## $ capital.gain   <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0~
## $ capital.loss   <int> 4356, 4356, 4356, 3900, 3900, 3770, 3770, 3683, 3683, 3~
## $ hours.per.week <int> 40, 18, 40, 40, 40, 45, 40, 20, 40, 60, 35, 45, 20, 55,~
## $ native.country <chr> "United-States", "United-States", "United-States", "Uni~
## $ income         <chr> "<=50K", "<=50K", "<=50K", "<=50K", "<=50K", "<=50K", "~
## Rows: 29,170
## Columns: 13
## $ age           <int> 90, 82, 66, 54, 41, 34, 38, 74, 68, 45, 38, 52, 32, 51, ~
## $ fnlwgt        <int> 77053, 132870, 186061, 140359, 264663, 216864, 150601, 8~
## $ education     <fct> HSgrad, HSgrad, Somecollege, 7th8th, Somecollege, HSgrad~
## $ eduyears      <int> 9, 9, 10, 4, 10, 9, 6, 16, 9, 16, 15, 13, 14, 16, 15, 7,~
## $ maritalstatus <fct> Widowed, Widowed, Widowed, Divorced, Separated, Divorced~
## $ occupation    <fct> Prof-specialty, Execmanagerial, Prof-specialty, Machineo~
## $ relationship  <fct> Notinfamily, Notinfamily, Unmarried, Unmarried, Ownchild~
## $ race          <fct> White, White, Black, White, White, White, White, White, ~
## $ sex           <fct> Female, Female, Female, Female, Female, Female, Male, Fe~
## $ hoursperweek  <int> 40, 18, 40, 40, 40, 45, 40, 20, 40, 35, 45, 20, 55, 40, ~
## $ native        <chr> "UnitedStates", "UnitedStates", "UnitedStates", "UnitedS~
## $ income        <fct> AtBelow50K, AtBelow50K, AtBelow50K, AtBelow50K, AtBelow5~
## $ class         <fct> Private, Private, Private, Private, Private, Private, Pr~
```

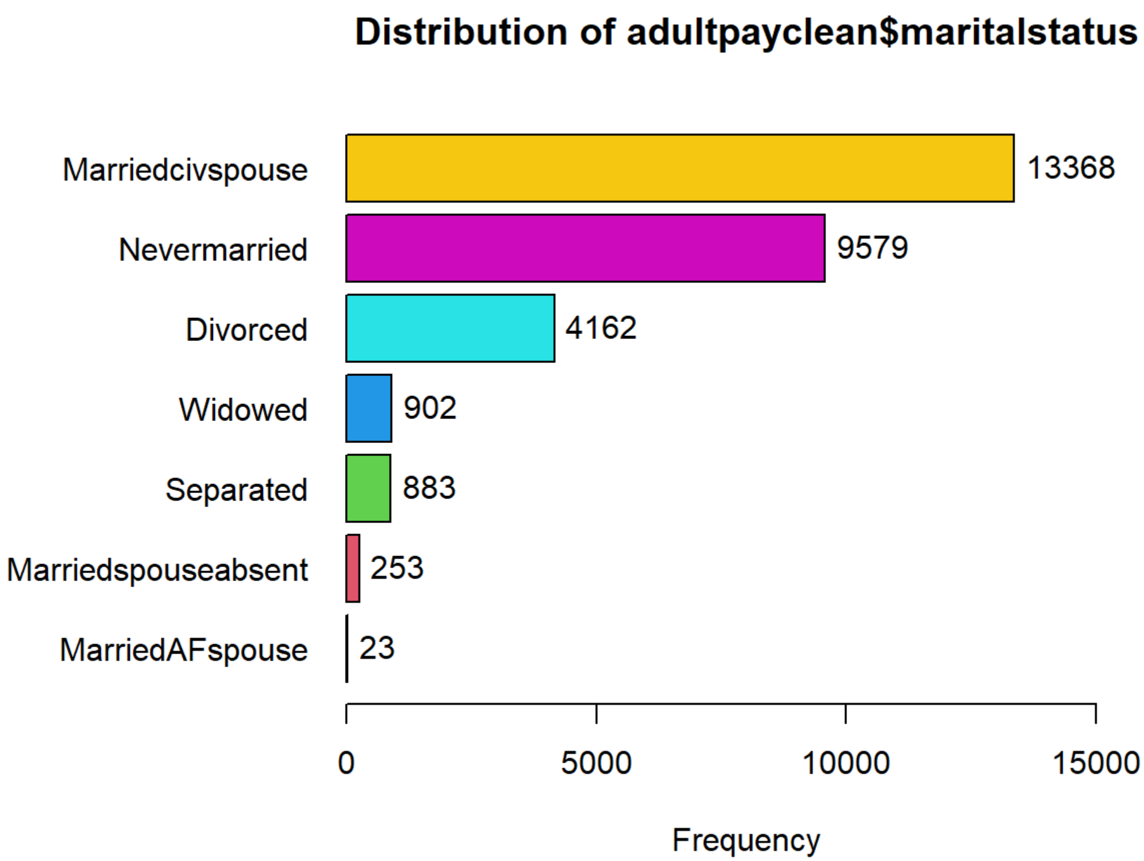


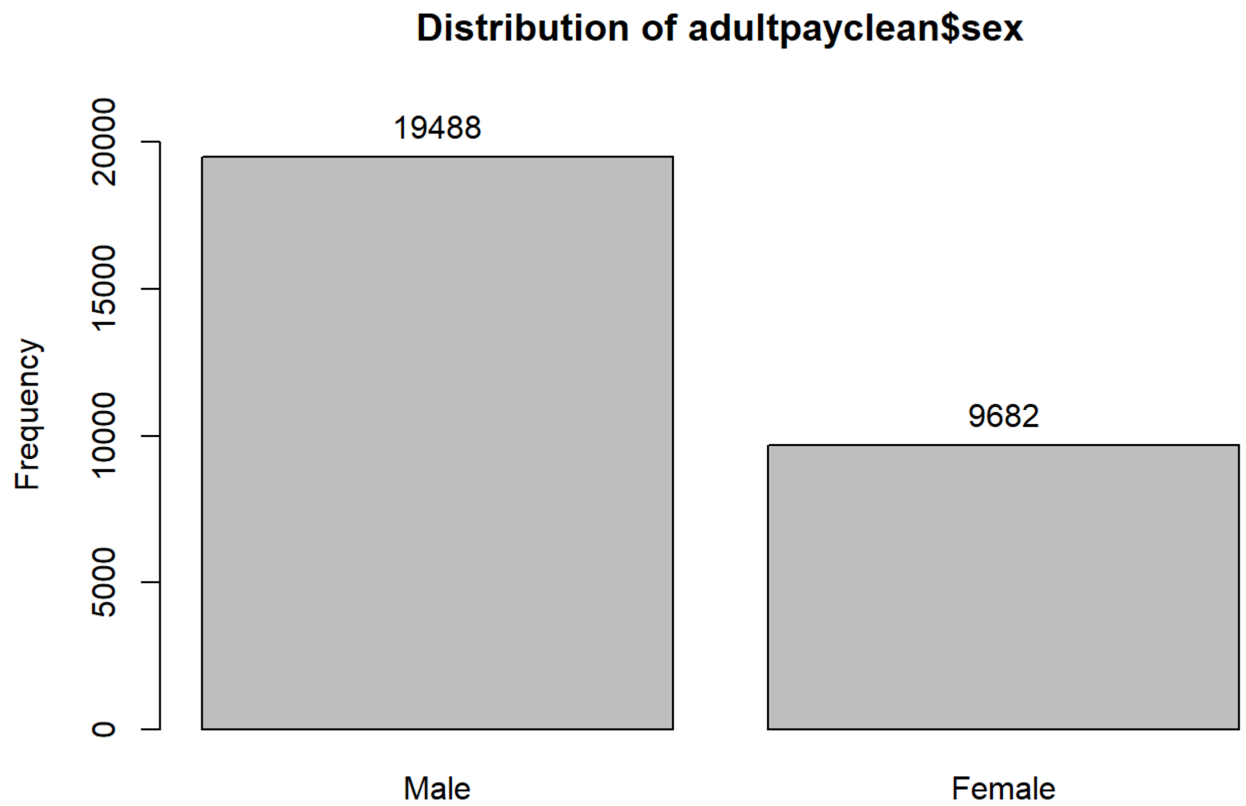
Distribution of adultpayclean$education
HSgrad
Somecollege
Bachelors
Masters
Assocvoc
11th
Assocacdm
10th
Profschool
7th8th
9th
12th
Doctorate
5th6th
1st4th
Preschool
9702
6740
4766
1527
1289
1067
982
848
502
499
395
365
328
97
46
17
0
2000
4000
6000
8000
10000
Frequency

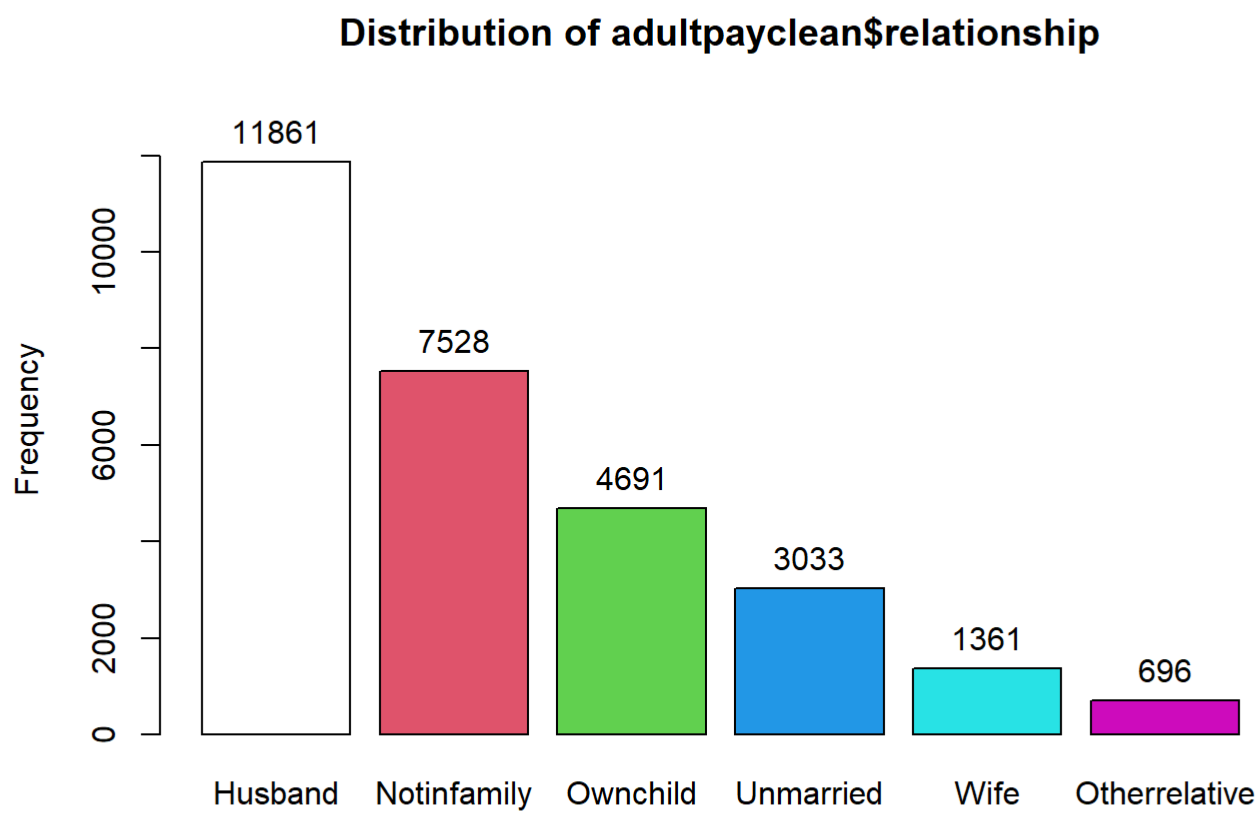


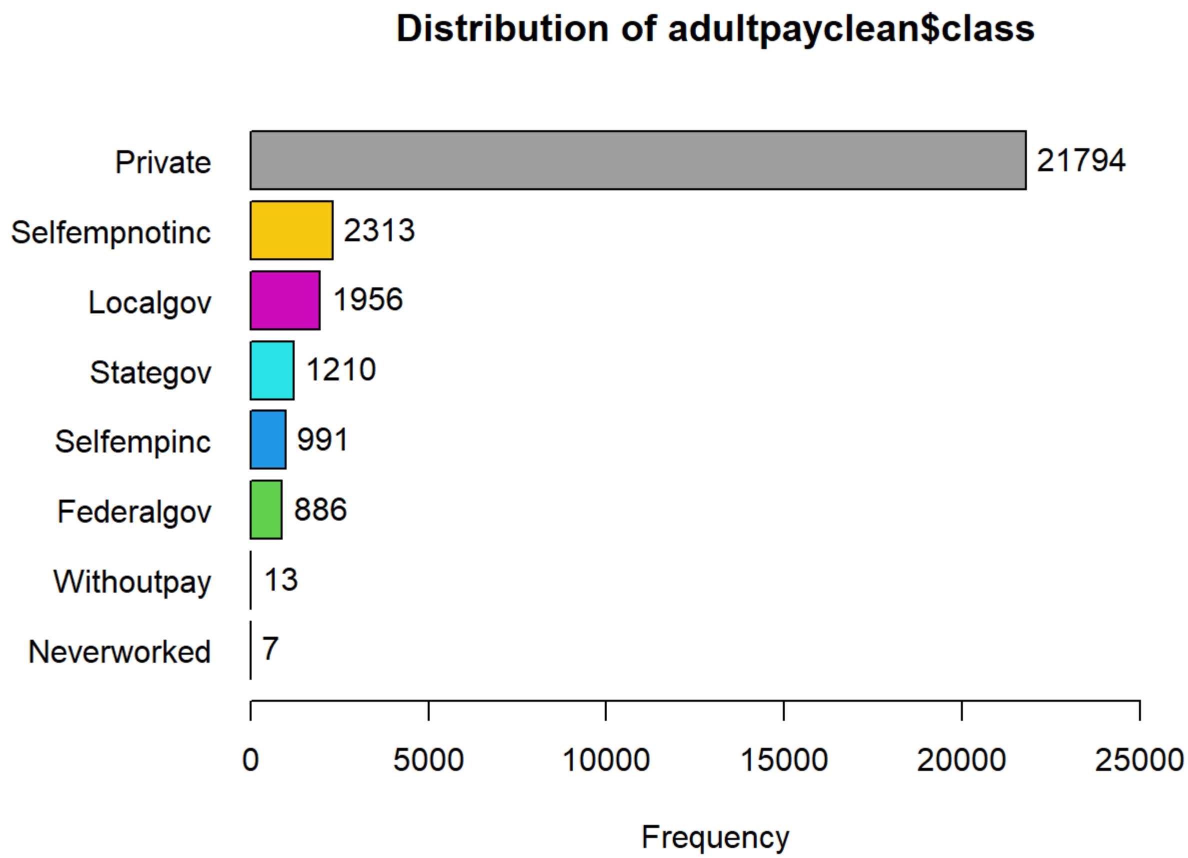
Distribution of adultpayclean$race
25621
20000

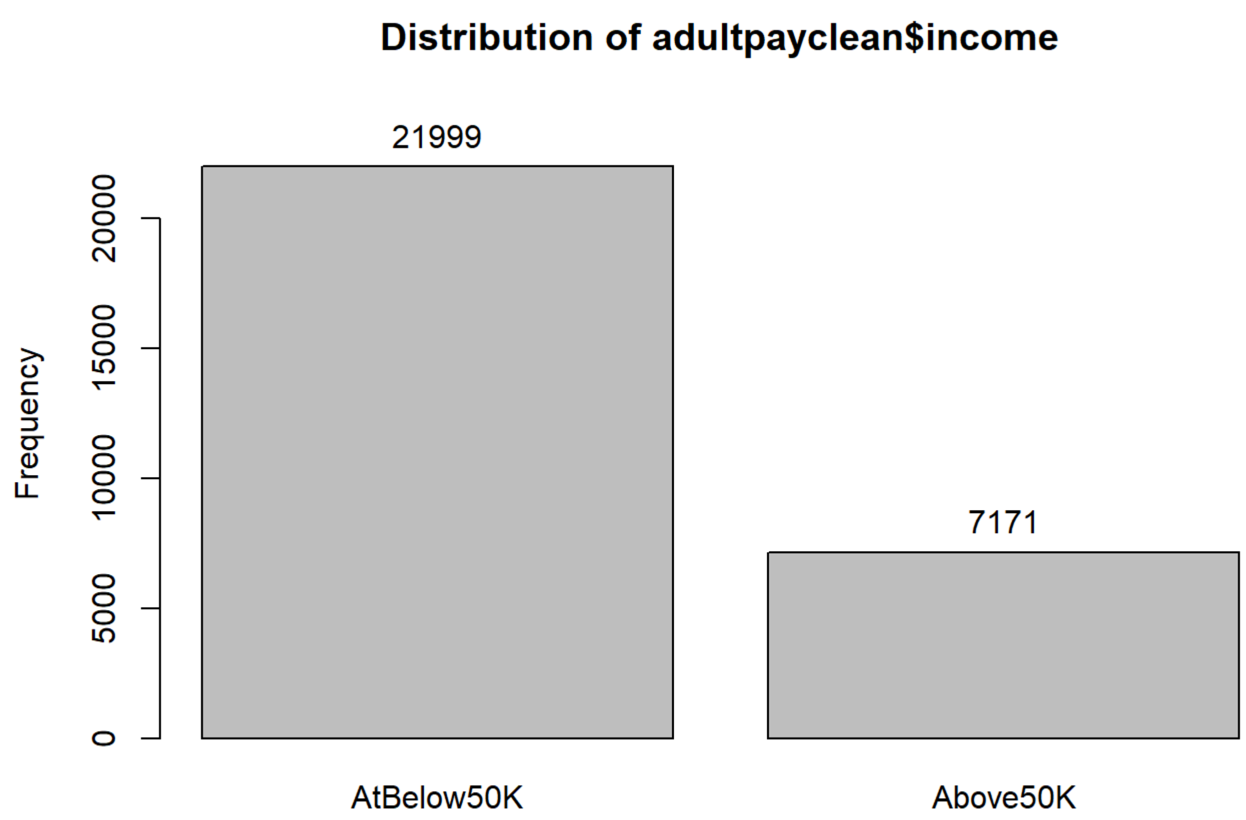

F
0
5000
1000(
2832
296
292
129
White
Black
Amer-Indian-Eskimo
Other


Distribution of adultpayclean$maritalstatus
Marriedcivspouse
Nevermarried
Divorced
Widowed
Separated
Marriedspouseabsent
MarriedAFspouse
13368
9579
4162
902
883
253
23
0
5000
10000
15000
Frequency


Distribution of adultpayclean$sex
Frequency
0
5000
10000
15000
20000
19488
9682
Male
Female


Distribution of adultpayclean$relationship
Frequency
0
2000
6000
10000
11861
7528
4691
3033
1361
696
Husband
Notinfamily
Ownchild
Unmarried
Wife
Otherrelative


Distribution of adultpayclean$class
Private
Selfempnotinc
Localgov
Stategov
Selfempinc
Federalgov
Withoutpay
Neverworked
21794
2313
1956
1210
991
886
13
7
0
5000
10000
15000
20000
25000
Frequency




```
if (!require(randomForest))
  install.packages("randomForest", repos = "http://cran.us.r-project.org")
```

```
## Loading required package: randomForest
```

```
## randomForest 4.6-14
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
##
## Attaching package: 'randomForest'
```

```
## The following object is masked from 'package:gridExtra':
##
##     combine
```

```
## The following object is masked from 'package:dplyr':
##
##     combine
```

```
## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
if (!require(purrr))
  install.packages("purrr", repos = "http://cran.us.r-project.org")

if (!require(e1071))
  install.packages("e1071")
```

```
## Loading required package: e1071
```

```
if (!require(pROC))
  install.packages("pRoc")
```

```
## Loading required package: pROC
```

```
## Type 'citation("pROC")' for a citation.
```

```
##
## Attaching package: 'pROC'
```

```
## The following object is masked from 'package:epiDisplay':
##
##     ci
```

```
## The following objects are masked from 'package:stats':
##
##     cov, smooth, var
```

```
if (!require(ROCit))
  install.packages("ROCit")
```

```
## Loading required package: ROCit
```

```
## Warning: package 'ROCit' was built under R version 4.1.2
```

```
library(caret)
library(gridExtra)
library(kableExtra)
library(randomForest)
library(purrr)
library(e1071)
library(caTools)
library(pROC)
library(ROCit)

#set the seed for reproducible results
set.seed(2008, sample.kind = "Rounding")
```

```
## Warning in set.seed(2008, sample.kind = "Rounding"): non-uniform 'Rounding'
## sampler used
```

```r
# the simplest possible machine algorithm: guessing the outcome
seat_of_the_pants <-
  sample(c("Above50K", "AtBelow50K"), length(test_index), replace = TRUE) %>% factor(levels = levels(adultpayclean_validatio
n$income))
# calculate the accuracy of this sampling
accuracy_guess <-
  mean(seat_of_the_pants == adultpayclean_validation$income)

# build a confusion matrix for this simple model
table(predicted = seat_of_the_pants, actual = adultpayclean_validation$income)
```

```
##               actual
## predicted      Above50K AtBelow50K
##   Above50K        361       1078
##   AtBelow50K      357       1122
```

```r
# tabulate accuracy by income levels
adultpayclean_validation %>%
  mutate(y_hat = seat_of_the_pants) %>%
  group_by(income) %>%
  summarize(accuracy = mean(y_hat == income))
```

```
## # A tibble: 2 x 2
##   income     accuracy
##   <fct>         <dbl>
## 1 Above50K      0.503
## 2 AtBelow50K    0.51
```

```r
# confusion matrix using R function
cm <-
  confusionMatrix(data = seat_of_the_pants , reference = adultpayclean_validation$income)
# display the confusion matrix
cm
```

```
## Confusion Matrix and Statistics
##
##             Reference
## Prediction   Above50K AtBelow50K
##   Above50K        361       1078
##   AtBelow50K      357       1122
##
##                Accuracy : 0.5082
##                  95% CI : (0.4899, 0.5265)
##     No Information Rate : 0.7539
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.0096
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.5028
##             Specificity : 0.5100
##          Pos Pred Value : 0.2509
##          Neg Pred Value : 0.7586
##              Prevalence : 0.2461
##          Detection Rate : 0.1237
##    Detection Prevalence : 0.4931
##       Balanced Accuracy : 0.5064
##
##        'Positive' Class : Above50K
##
```

```
#record the sensitivity, specificity, and prevalence
sensitivity_guess <- cm$byClass[["Sensitivity"]]
specificity_guess <- cm$byClass[["Specificity"]]
prevalence_guess <- cm$byClass[["Prevalence"]]
f1_guess <- cm$byClass[["F1"]]

#find the area under the curve/ROC
auc(ifelse(adultpayclean_validation$income == "Above50K",1,0), ifelse(seat_of_the_pants == "Above50K",1,0))
```

```
## Setting levels: control = 0, case = 1
```

```
## Setting direction: controls < cases
```

```
## Area under the curve: 0.5064
```

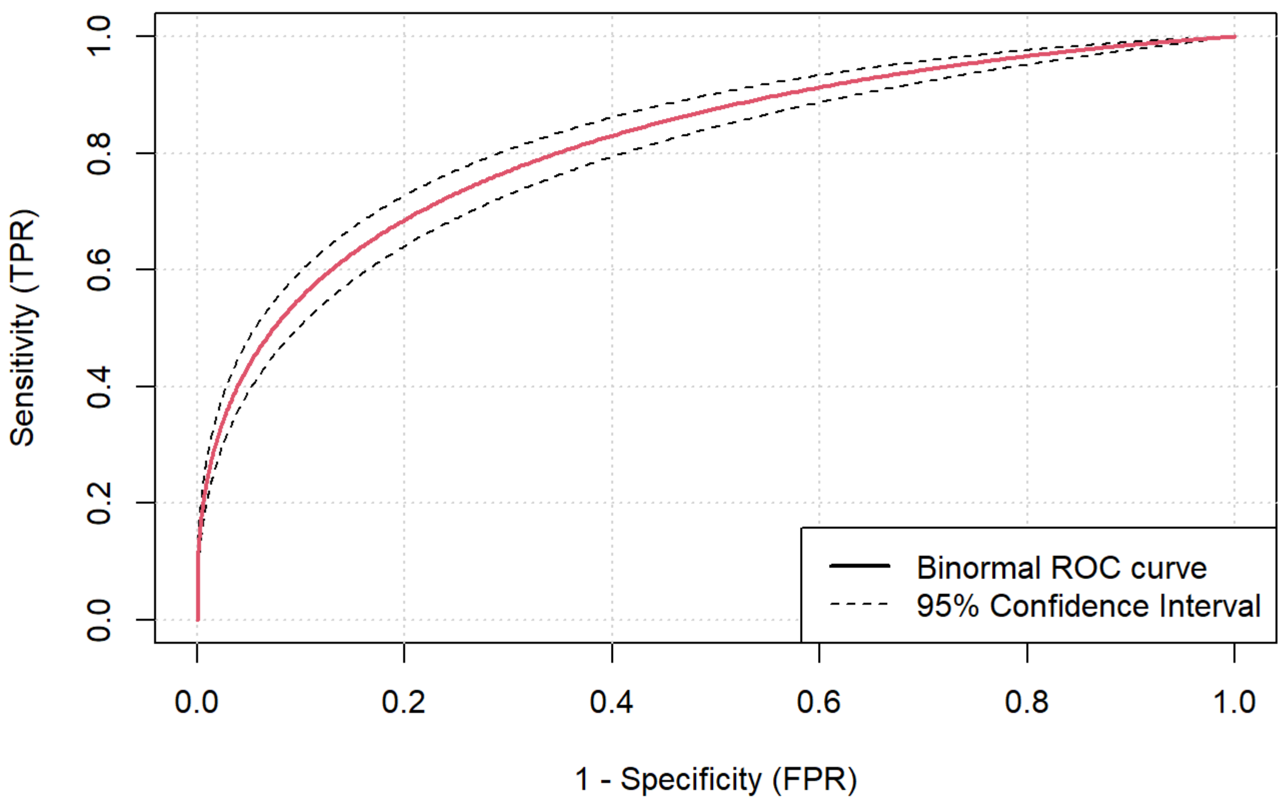
```
set.seed(2008)
#logistic linear model
# create the model
lm_fit <- adultpayclean_train %>%
  mutate(y = as.numeric(income == "Above50K")) %>%
  lm(y ~ age + eduyears + sex + race + hoursperweek + maritalstatus + relationship + education,
     data = .)

# predict using test set
p_hat_logit <- predict(lm_fit, newdata = adultpayclean_validation)
```

```
## Warning in predict.lm(lm_fit, newdata = adultpayclean_validation): prediction
## from a rank-deficient fit may be misleading
```

```
#translate predicted data into factor
y_hat_logit <-
  ifelse(p_hat_logit > 0.5, "Above50K", "AtBelow50K") %>% factor

#compare the predicted vs observed values and use confusionMatrix to get the accuracy and other metrics
cm_lm <-
  confusionMatrix(y_hat_logit, adultpayclean_validation$income)
accuracy_lm <-
  confusionMatrix(y_hat_logit, adultpayclean_validation$income)$overall[["Accuracy"]]

cm_lm
```

```
## Confusion Matrix and Statistics
##
##             Reference
## Prediction   Above50K AtBelow50K
##   Above50K        318        139
##   AtBelow50K      400       2061
##
##                Accuracy : 0.8153
##                  95% CI : (0.8007, 0.8292)
##     No Information Rate : 0.7539
##     P-Value [Acc > NIR] : 1.243e-15
##
##                   Kappa : 0.4327
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.4429
##             Specificity : 0.9368
##          Pos Pred Value : 0.6958
##          Neg Pred Value : 0.8375
##              Prevalence : 0.2461
##          Detection Rate : 0.1090
##    Detection Prevalence : 0.1566
##       Balanced Accuracy : 0.6899
##
##        'Positive' Class : Above50K
##
```

```
#record the sensitivity, specificity, and prevalence
sensitivity_lm <- cm_lm$byClass[["Sensitivity"]]
specificity_lm <- cm_lm$byClass[["Specificity"]]
prevalence_lm <- cm_lm$byClass[["Prevalence"]]
f1_lm <- cm_lm$byClass[["F1"]]

#Find the ROC and plot it. Show the AUC as well
pROC_bin <- ROCit::rocit(ifelse(adultpayclean_validation$income == "Above50K",1,0), ifelse(unname(y_hat_logit) == "Above50K"
,1,0),method="bin")
ciROC_bin95 <- ROCit::ciROC(pROC_bin,level = 0.95)
plot(ciROC_bin95, col = 1, values=TRUE)
lines(ciROC_bin95$TPR~ciROC_bin95$FPR, col = 2, lwd = 2)
```



```
ROCit::ciAUC(pROC_bin)
```

```
##
##    estimated AUC : 0.81709434414188
##    AUC estimation method : binormal
##
##    CI of AUC
##    confidence level = 95%
##    lower = 0.792575444461593     upper = 0.841613243822167
```

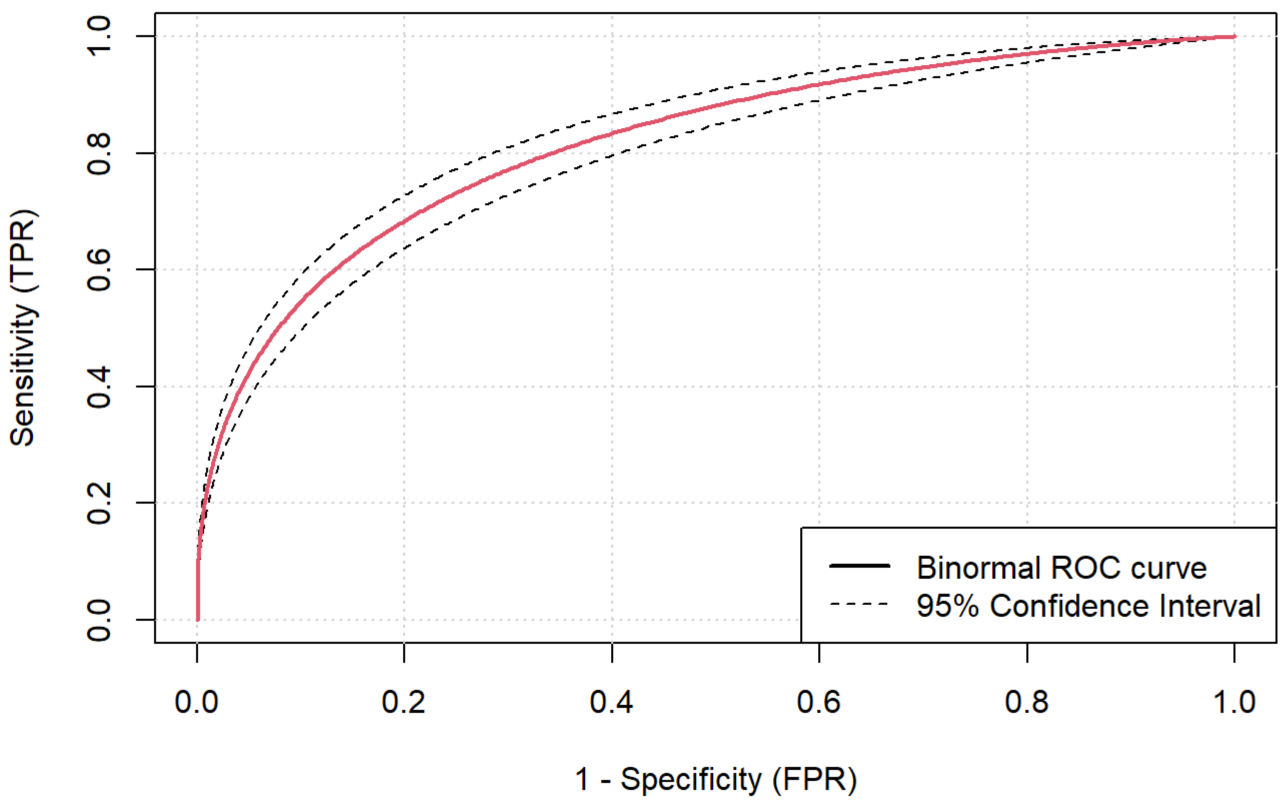
```
set.seed(2008)
#general linear model
#create the glm model
glm_fit <- adultpayclean_train %>%
  mutate(y = as.numeric(income == "Above50K")) %>%
  glm(
    y ~ age + eduyears + sex + race + hoursperweek + maritalstatus + relationship + education,
    data = .,
    family = "binomial"
  )

# predict using validation set
p_hat_logit <- predict(glm_fit, newdata = adultpayclean_validation)
```

```
## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type == :
## prediction from a rank-deficient fit may be misleading
```

```r
# translate the predicted data into factor
y_hat_logit <-
  ifelse(p_hat_logit > 0.5, "Above50K", "AtBelow50K") %>% factor

# compare the predicted vs observed values and use confusionMatrix to get the accuracy and other metrics for the glm model
cm_glm <-
  confusionMatrix(y_hat_logit, adultpayclean_validation$income)
accuracy_glm <-
  confusionMatrix(y_hat_logit, adultpayclean_validation$income)$overall[["Accuracy"]]

cm_glm
```

```
## Confusion Matrix and Statistics
##
##             Reference
## Prediction   Above50K AtBelow50K
##   Above50K        278        113
##   AtBelow50K      440       2087
##
##                Accuracy : 0.8105
##                  95% CI : (0.7958, 0.8246)
##     No Information Rate : 0.7539
##     P-Value [Acc > NIR] : 1.758e-13
##
##                   Kappa : 0.3967
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.38719
##             Specificity : 0.94864
##          Pos Pred Value : 0.71100
##          Neg Pred Value : 0.82588
##              Prevalence : 0.24606
##          Detection Rate : 0.09527
##    Detection Prevalence : 0.13400
##       Balanced Accuracy : 0.66791
##
##        'Positive' Class : Above50K
##
```

```r
#record the sensitivity, specificity, and prevalence
sensitivity_glm <- cm_glm$byClass[["Sensitivity"]]
specificity_glm <- cm_glm$byClass[["Specificity"]]
prevalence_glm <- cm_glm$byClass[["Prevalence"]]
f1_glm <- cm_glm$byClass[["F1"]]

#Find the ROC and plot it. Show the AUC as well
pROC_bin <- ROCit::rocit(ifelse(adultpayclean_validation$income == "Above50K",1,0), ifelse(unname(y_hat_logit) == "Above50K"
,1,0),method="bin")
ciROC_bin95 <- ROCit::ciROC(pROC_bin,level = 0.95)
plot(ciROC_bin95, col = 1, values=TRUE)
lines(ciROC_bin95$TPR~ciROC_bin95$FPR, col = 2, lwd = 2)
```



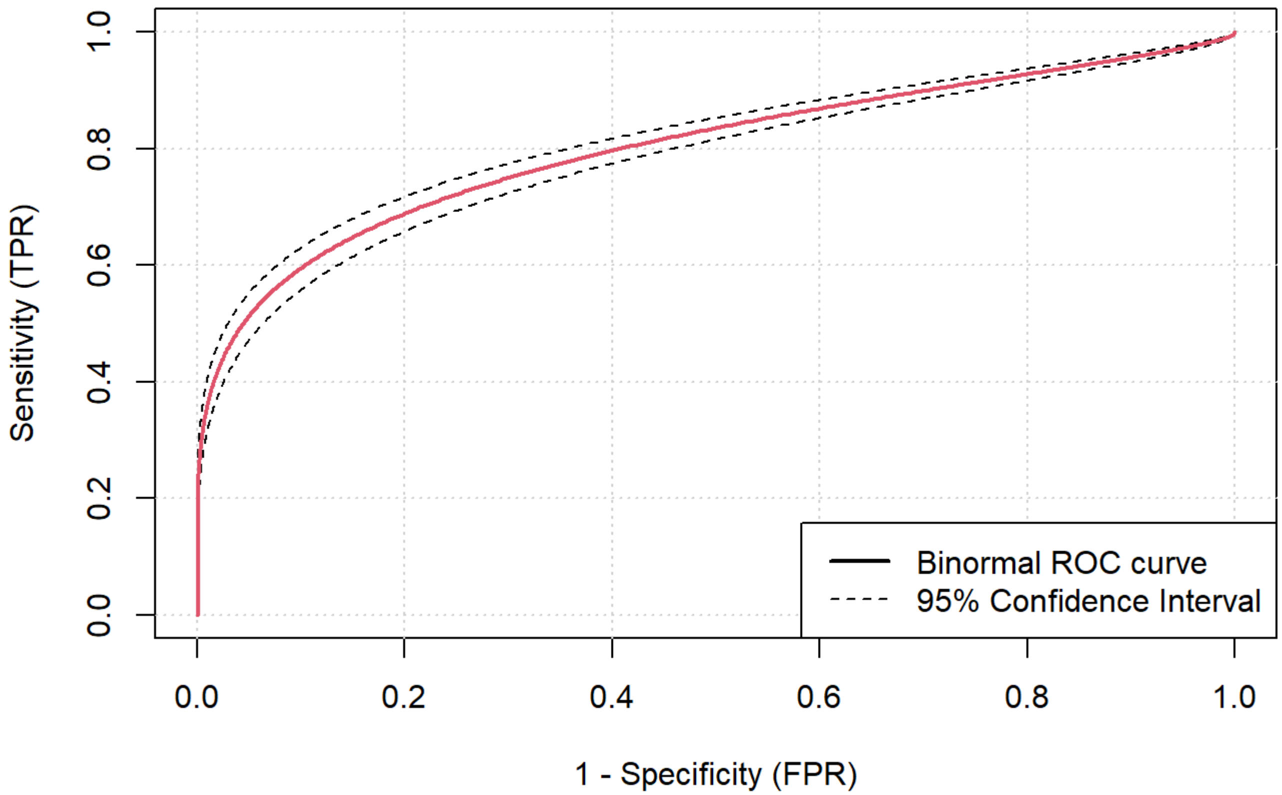
```
ROCit::ciAUC(pROC_bin)
```

```
##
##    estimated AUC : 0.818172245714073
##    AUC estimation method : binormal
##
##    CI of AUC
##    confidence level = 95%
##    lower = 0.791852436696456     upper = 0.844492054731691
```

```
#Naive bayes
set.seed(2008)
#create the naive bayes model
train_nb <- adultpayclean_train %>%
   mutate(y = as.factor(income == "Above50K")) %>%
   naiveBayes(y ~ age + eduyears + sex + race + hoursperweek + maritalstatus + relationship+education,data = .)

#predict using the validation dataset
y_hat_nb <- predict(train_nb, newdata = adultpayclean_validation)
#create the confusion matrix
cm_tab <- table(adultpayclean_validation$income == "Above50K", y_hat_nb)
cm_nb <- confusionMatrix(cm_tab)
cm_nb
```

```
## Confusion Matrix and Statistics
##
##        y_hat_nb
##         FALSE TRUE
##   FALSE  1794  406
##   TRUE    175  543
##
##                Accuracy : 0.8009
##                  95% CI : (0.7859, 0.8152)
##     No Information Rate : 0.6748
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.5158
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.9111
##             Specificity : 0.5722
##          Pos Pred Value : 0.8155
##          Neg Pred Value : 0.7563
##              Prevalence : 0.6748
##          Detection Rate : 0.6148
##    Detection Prevalence : 0.7539
##       Balanced Accuracy : 0.7417
##
##        'Positive' Class : FALSE
##
```

```r
#get the accuracy, sensitivity, specificity, prevalence and, F1 score
accuracy_nb <- cm_nb$overall[["Accuracy"]]
sensitivity_nb <- cm_nb$byClass[["Sensitivity"]]
specificity_nb <- cm_nb$byClass[["Specificity"]]
prevalence_nb <- cm_nb$byClass[["Prevalence"]]
f1_nb <- cm_nb$byClass[["F1"]]

#Find the ROC and plot it. Show the AUC as well
pROC_bin <- ROCit::rocit(ifelse(adultpayclean_validation$income == "Above50K",1,0), ifelse(unname(y_hat_nb) == "TRUE",1,0),m
ethod="bin")
ciROC_bin95 <- ROCit::ciROC(pROC_bin,level = 0.95)
plot(ciROC_bin95, col = 1, values=TRUE)
lines(ciROC_bin95$TPR~ciROC_bin95$FPR, col = 2, lwd = 2)
```



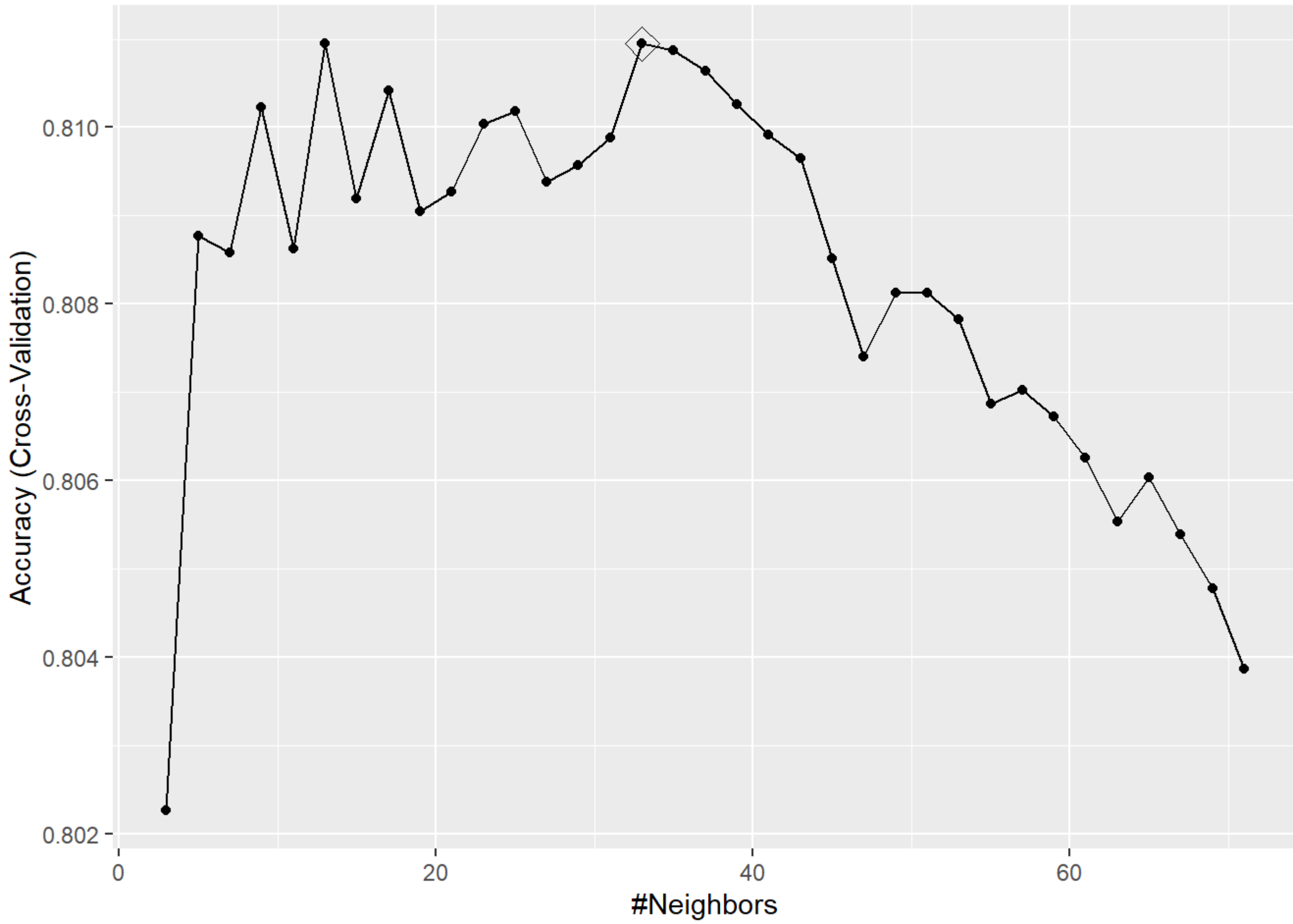

```
ROCit::ciAUC(pROC_bin)
```

```
##
##    estimated AUC : 0.801438459045964
##    AUC estimation method : binormal
##
##    CI of AUC
##    confidence level = 95%
##    lower = 0.783049833697349     upper = 0.819827084394579
```

```r
# translate income factor into binary outcome
temp <- adultpayclean_train %>%
  mutate(y = as.factor(income == "Above50K"))

#k-nearest neighbors with a train control and tuning
set.seed(2008)
# train control to use 10% of the observations each to speed up computations
control <- trainControl(method = "cv", number = 10, p = .9)
# train the model using knn. choose the best k value using tuning algorithm
train_knn <-
  train(
    y ~ age + eduyears + sex + race + hoursperweek + maritalstatus + relationship + education,
    method = "knn",
    data = temp,
    tuneGrid = data.frame(k = seq(3, 71, 2)),
    trControl = control
  )
```

```
## Warning in (function (kind = NULL, normal.kind = NULL, sample.kind = NULL) :
## non-uniform 'Rounding' sampler used
```

```r
#plot the resulting model
ggplot(train_knn, highlight = TRUE)
```




```
#verify which k value was used
train_knn$bestTune
```

```
##     k
## 16 33
```

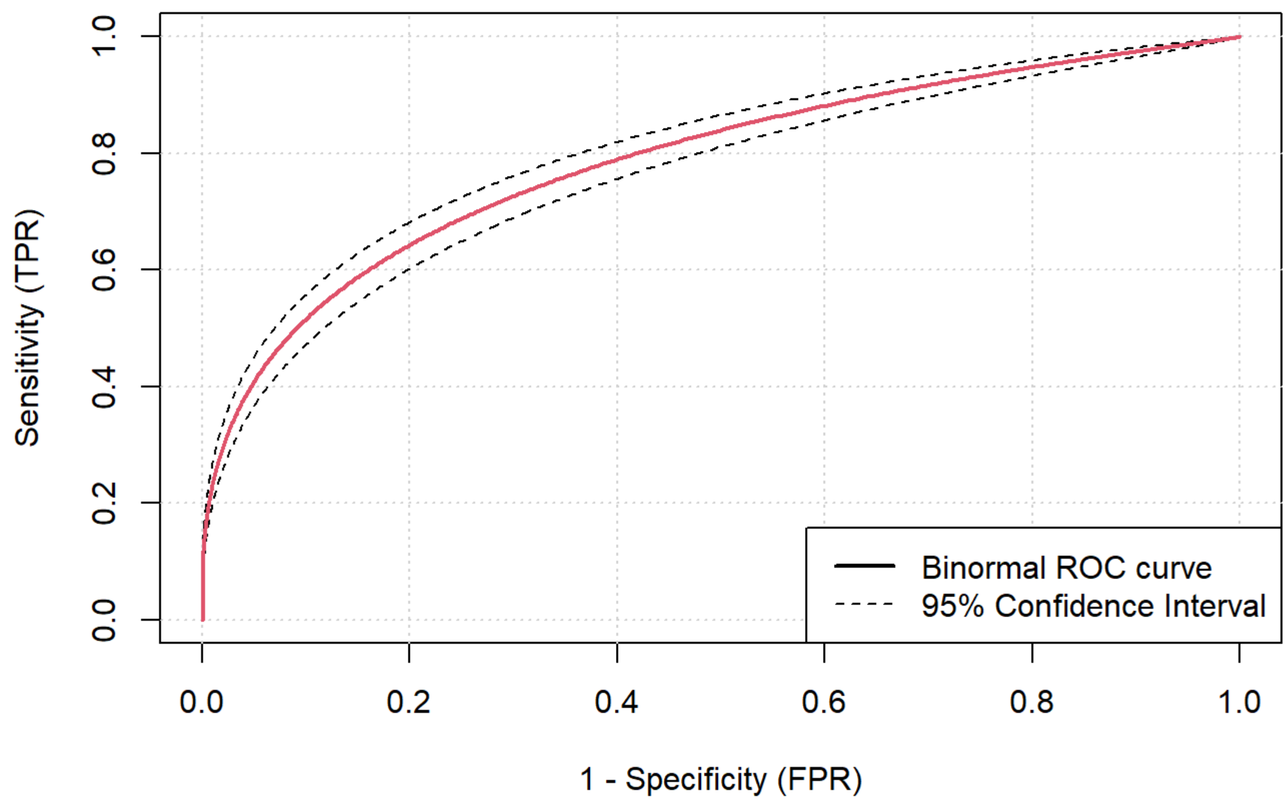
```
train_knn$finalModel
```

```
## 33-nearest neighbor model
## Training set outcome distribution:
##
## FALSE  TRUE
## 19799  6453
```

```
#use this trained model to predict raw knn predictions
y_hat_knn <-
  predict(train_knn, adultpayclean_validation, type = "raw")

# compare the predicted and observed values using confusionMatrix to get the accuracy and other metrics
cm_knn <-
  confusionMatrix(y_hat_knn,
                  as.factor(adultpayclean_validation$income == "Above50K"))
accuracy_knn <-
  confusionMatrix(y_hat_knn,
                  as.factor(adultpayclean_validation$income == "Above50K"))$overall[["Accuracy"]]

cm_knn
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction FALSE TRUE
##      FALSE  1992  360
##      TRUE    208  358
##
##                Accuracy : 0.8053
##                  95% CI : (0.7905, 0.8196)
##     No Information Rate : 0.7539
##     P-Value [Acc > NIR] : 2.195e-11
##
##                   Kappa : 0.4351
##
##  Mcnemar's Test P-Value : 2.361e-10
##
##             Sensitivity : 0.9055
##             Specificity : 0.4986
##          Pos Pred Value : 0.8469
##          Neg Pred Value : 0.6325
##              Prevalence : 0.7539
##          Detection Rate : 0.6827
##    Detection Prevalence : 0.8060
##       Balanced Accuracy : 0.7020
##
##        'Positive' Class : FALSE
##
```

```
#record the sensitivity, specificity, and prevalence
sensitivity_knn <- cm_knn$byClass[["Sensitivity"]]
specificity_knn <- cm_knn$byClass[["Specificity"]]
prevalence_knn <- cm_knn$byClass[["Prevalence"]]
f1_knn <- cm_knn$byClass[["F1"]]

#Find the ROC and plot it. Show the AUC as well
pROC_bin <- ROCit::rocit(ifelse(adultpayclean_validation$income == "Above50K",1,0), ifelse(unname(y_hat_knn) == "TRUE",1,0), method="bin")
ciROC_bin95 <- ROCit::ciROC(pROC_bin,level = 0.95)
plot(ciROC_bin95, col = 1, values=TRUE)
lines(ciROC_bin95$TPR~ciROC_bin95$FPR, col = 2, lwd = 2)
```



```r
ROCit::ciAUC(pROC_bin)
```

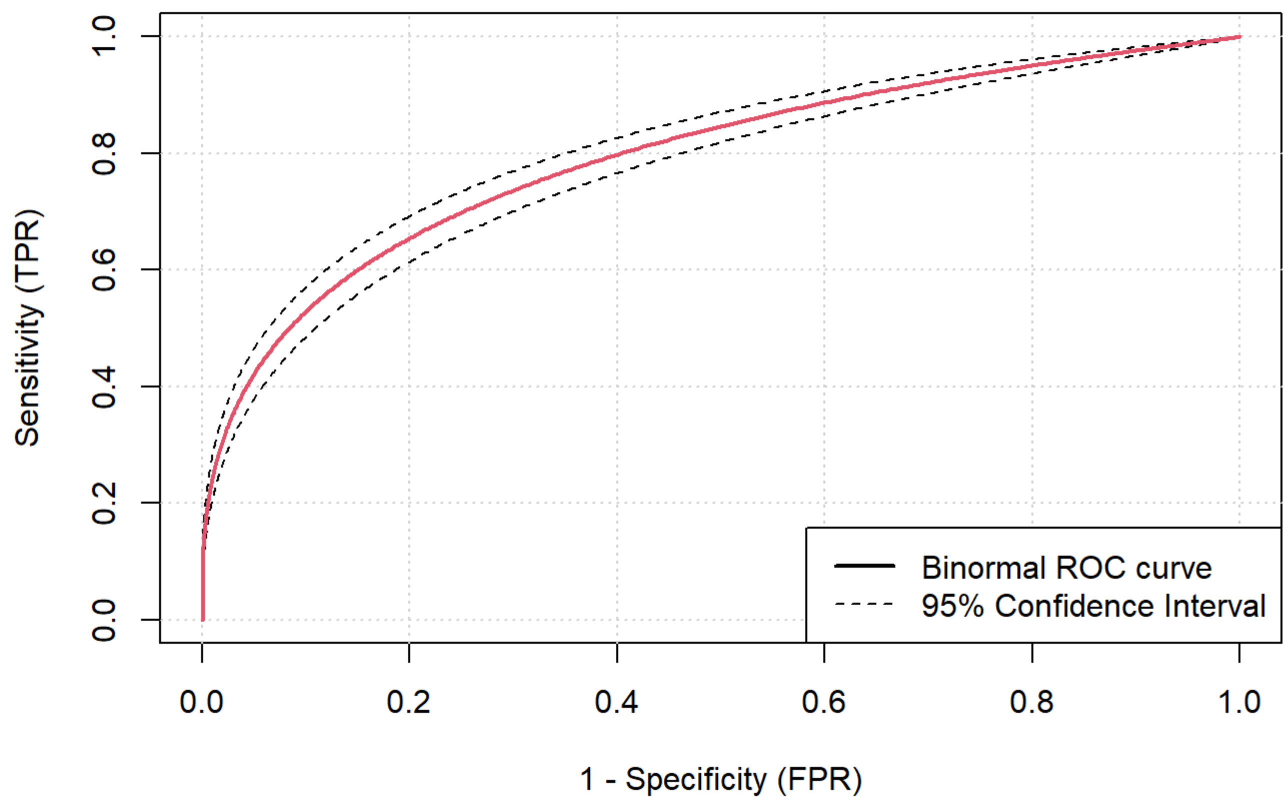
```
##
##    estimated AUC : 0.787212472866943
##    AUC estimation method : binormal
##
##    CI of AUC
##    confidence level = 95%
##    lower = 0.763782743778267     upper = 0.810642201955618
```

```r
#k-nearest classification using tuning function
set.seed(2008)

#train the model using knn3 classification
ks <- seq(3, 251, 2)
knntune <- map_df(ks, function(k) {
  temp <- adultpayclean_train %>%
    mutate(y = as.factor(income == "Above50K"))
  temp_test <- adultpayclean_validation %>%
    mutate(y = as.factor(income == "Above50K"))
  #create the kkn3 model
  knn_fit <-
    knn3(
      y ~ age + eduyears + sex + race + hoursperweek + maritalstatus + relationship+education,
      data = temp,
      k = k
    )
  #predict the model for the current k
  y_hat <- predict(knn_fit, temp, type = "class")
  #get the confusionmatrix for the current k
  cm_train <- confusionMatrix(y_hat, temp$y)
  train_error <- cm_train$overall["Accuracy"]
  #do the same for test model
  y_hat <- predict(knn_fit, temp_test, type = "class")
  cm_test <- confusionMatrix(y_hat, temp_test$y)
  test_error <- cm_test$overall["Accuracy"]

  tibble(train = train_error, test = test_error)
})
#get the accuracy for the k with maximum accuracy
accuracy_knntune <- max(knntune$test)
#get the confusion matrix for that k
knn_fit <-
  knn3(
    y ~ age + eduyears + sex + race + hoursperweek + maritalstatus + relationship+education,
    data = temp,
    k = ks[which.max(knntune$test)]
  )
#predict the knn tune using the model for the k neighbor
```

```r
y_hat_knntune <- predict(knn_fit, adultpayclean_validation, type = "class")
cm_knntune <- confusionMatrix(y_hat_knntune, as.factor(adultpayclean_validation$income == "Above50K"))

cm_knntune
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction FALSE TRUE
##      FALSE  1994  351
##      TRUE    206  367
##
##                Accuracy : 0.8091
##                  95% CI : (0.7944, 0.8232)
##     No Information Rate : 0.7539
##     P-Value [Acc > NIR] : 6.680e-13
##
##                   Kappa : 0.448
##
##  Mcnemar's Test P-Value : 1.051e-09
##
##             Sensitivity : 0.9064
##             Specificity : 0.5111
##          Pos Pred Value : 0.8503
##          Neg Pred Value : 0.6405
##              Prevalence : 0.7539
##          Detection Rate : 0.6833
##    Detection Prevalence : 0.8036
##       Balanced Accuracy : 0.7088
##
##        'Positive' Class : FALSE
##
```

```
#record the sensitivity, specificity, and prevalence
sensitivity_knntune <- cm_knntune$byClass[["Sensitivity"]]
specificity_knntune <- cm_knntune$byClass[["Specificity"]]
prevalence_knntune <- cm_knntune$byClass[["Prevalence"]]
f1_knntune <- cm_knntune$byClass[["F1"]]

#Find the ROC and plot it. Show the AUC as well
pROC_bin <- ROCit::rocit(ifelse(adultpayclean_validation$income == "Above50K",1,0), ifelse(unname(y_hat_knntune) == "TRUE",1
,0),method="bin")
ciROC_bin95 <- ROCit::ciROC(pROC_bin,level = 0.95)
plot(ciROC_bin95, col = 1, values=TRUE)
lines(ciROC_bin95$TPR~ciROC_bin95$FPR, col = 2, lwd = 2)
```



```r
ROCit::ciAUC(pROC_bin)
```

```
##
##    estimated AUC : 0.794126916519226
##    AUC estimation method : binormal
##
##    CI of AUC
##    confidence level = 95%
##    lower = 0.771082499427732     upper = 0.81717133361072
```

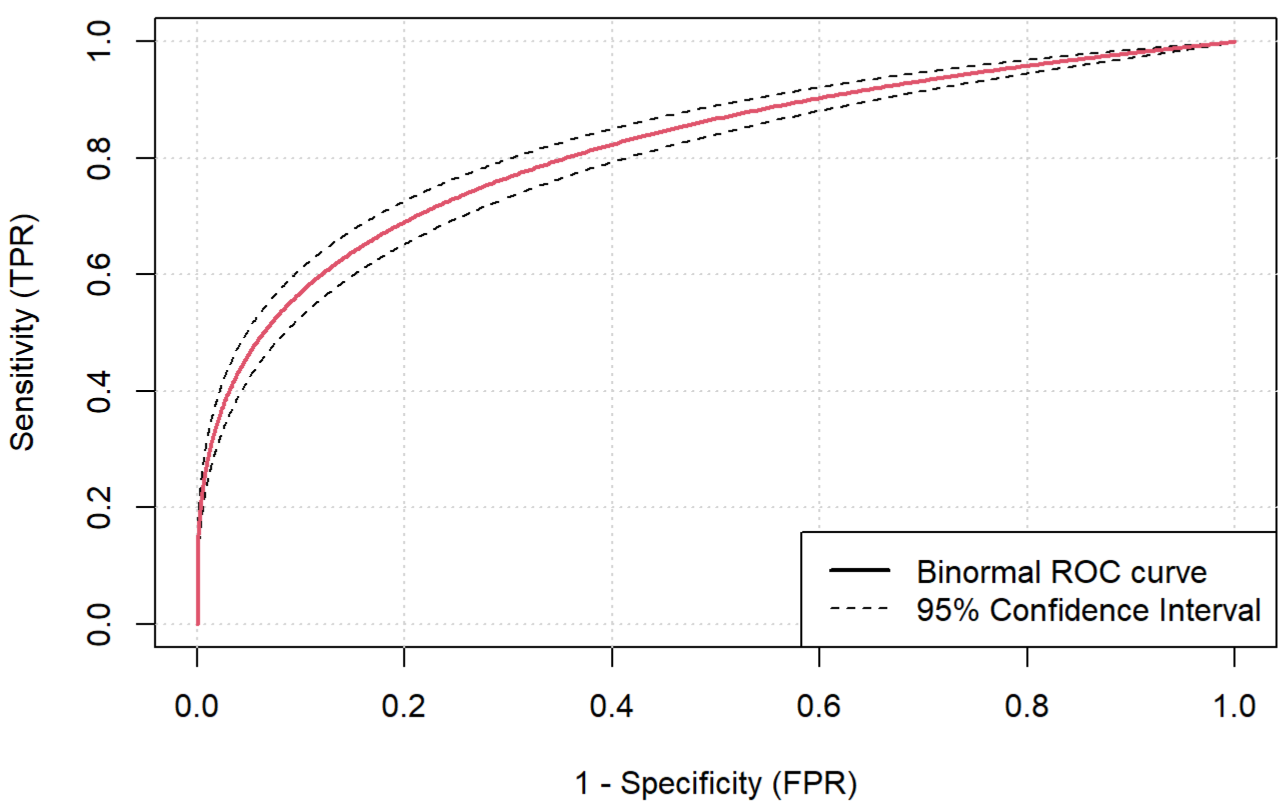
```r
#recursive partitioning using rpart
set.seed(2008)
#train the model with the recursive partitioning
train_rpart <-
  train(
    y ~ age + eduyears + sex + race + hoursperweek + maritalstatus + relationship+education,
    method = "rpart",
    tuneGrid = data.frame(cp = seq(0.0, 0.1, len = 25)),
    data = temp
  )
```

```
## Warning in (function (kind = NULL, normal.kind = NULL, sample.kind = NULL) :
## non-uniform 'Rounding' sampler used
```

```r
#predict the outcomes with this model
y_hat_rpart <- predict(train_rpart, adultpayclean_validation)
#confusion matrix for the rpart model
cm_rpart <-
  confusionMatrix(y_hat_rpart,
                as.factor(adultpayclean_validation$income  == "Above50K"))
#get the accuracy
accuracy_rpart <-
  confusionMatrix(y_hat_rpart,
                as.factor(adultpayclean_validation$income  == "Above50K"))$overall["Accuracy"]

cm_rpart
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction FALSE TRUE
##      FALSE  2002  324
##      TRUE    198  394
##
##                Accuracy : 0.8211
##                  95% CI : (0.8067, 0.8349)
##     No Information Rate : 0.7539
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.4876
##
##  Mcnemar's Test P-Value : 4.472e-08
##
##             Sensitivity : 0.9100
##             Specificity : 0.5487
##          Pos Pred Value : 0.8607
##          Neg Pred Value : 0.6655
##              Prevalence : 0.7539
##          Detection Rate : 0.6861
##    Detection Prevalence : 0.7971
##       Balanced Accuracy : 0.7294
##
##        'Positive' Class : FALSE
##
```

```
#record the sensitivity, specificity, and prevalence
sensitivity_rpart <- cm_rpart$byClass[["Sensitivity"]]
specificity_rpart <- cm_rpart$byClass[["Specificity"]]
prevalence_rpart <- cm_rpart$byClass[["Prevalence"]]
f1_rpart <- cm_rpart$byClass[["F1"]]

#Find the ROC and plot it. Show the AUC as well
pROC_bin <- ROCit::rocit(ifelse(adultpayclean_validation$income == "Above50K",1,0), ifelse(unname(y_hat_rpart) == "TRUE",1,0
),method="bin")
ciROC_bin95 <- ROCit::ciROC(pROC_bin,level = 0.95)
plot(ciROC_bin95, col = 1, values=TRUE)
lines(ciROC_bin95$TPR~ciROC_bin95$FPR, col = 2, lwd = 2)
```



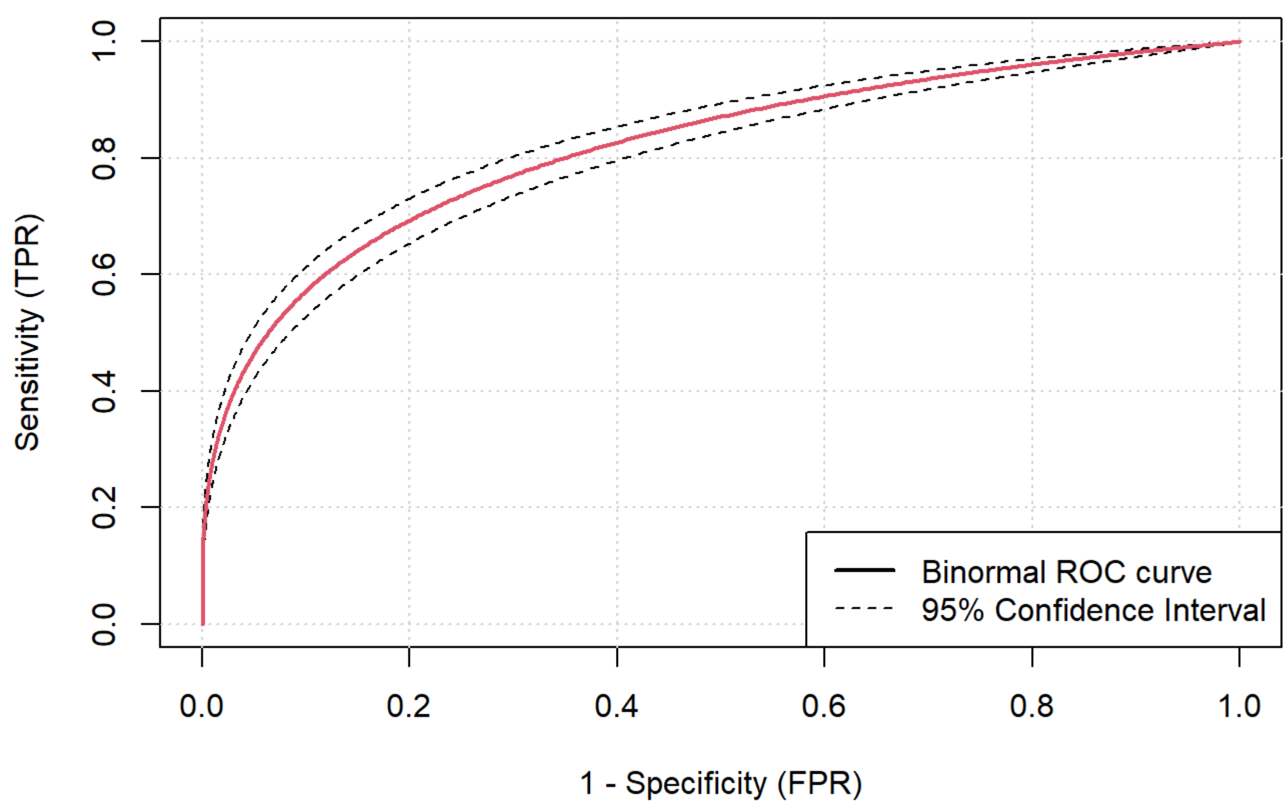
```r
ROCit::ciAUC(pROC_bin)
```

```
##
##    estimated AUC : 0.815732110644685
##    AUC estimation method : binormal
##
##    CI of AUC
##    confidence level = 95%
##    lower = 0.793892076612227     upper = 0.837572144677144
```

```r
#random forest
set.seed(2008)
#train the vanilla random forest model
train_rf <-
  randomForest(y ~ age + eduyears + sex + race + hoursperweek + maritalstatus + relationship+education,
               data = temp)

y_hat_rf <- predict(train_rf, adultpayclean_validation)

#create the confusionMatrix
cm_rf <-
  confusionMatrix(
    y_hat_rf,
    as.factor(adultpayclean_validation$income  == "Above50K")
  )
#get the accuracy
accuracy_rf <-
  confusionMatrix(
    y_hat_rf,
    as.factor(adultpayclean_validation$income  == "Above50K")
  )$overall["Accuracy"]

cm_rf
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction FALSE TRUE
##      FALSE  2012  332
##      TRUE    188  386
##
##                Accuracy : 0.8218
##                  95% CI : (0.8074, 0.8355)
##     No Information Rate : 0.7539
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.4849
##
##  Mcnemar's Test P-Value : 3.588e-10
##
##             Sensitivity : 0.9145
##             Specificity : 0.5376
##          Pos Pred Value : 0.8584
##          Neg Pred Value : 0.6725
##              Prevalence : 0.7539
##          Detection Rate : 0.6895
##    Detection Prevalence : 0.8033
##       Balanced Accuracy : 0.7261
##
##        'Positive' Class : FALSE
##
```

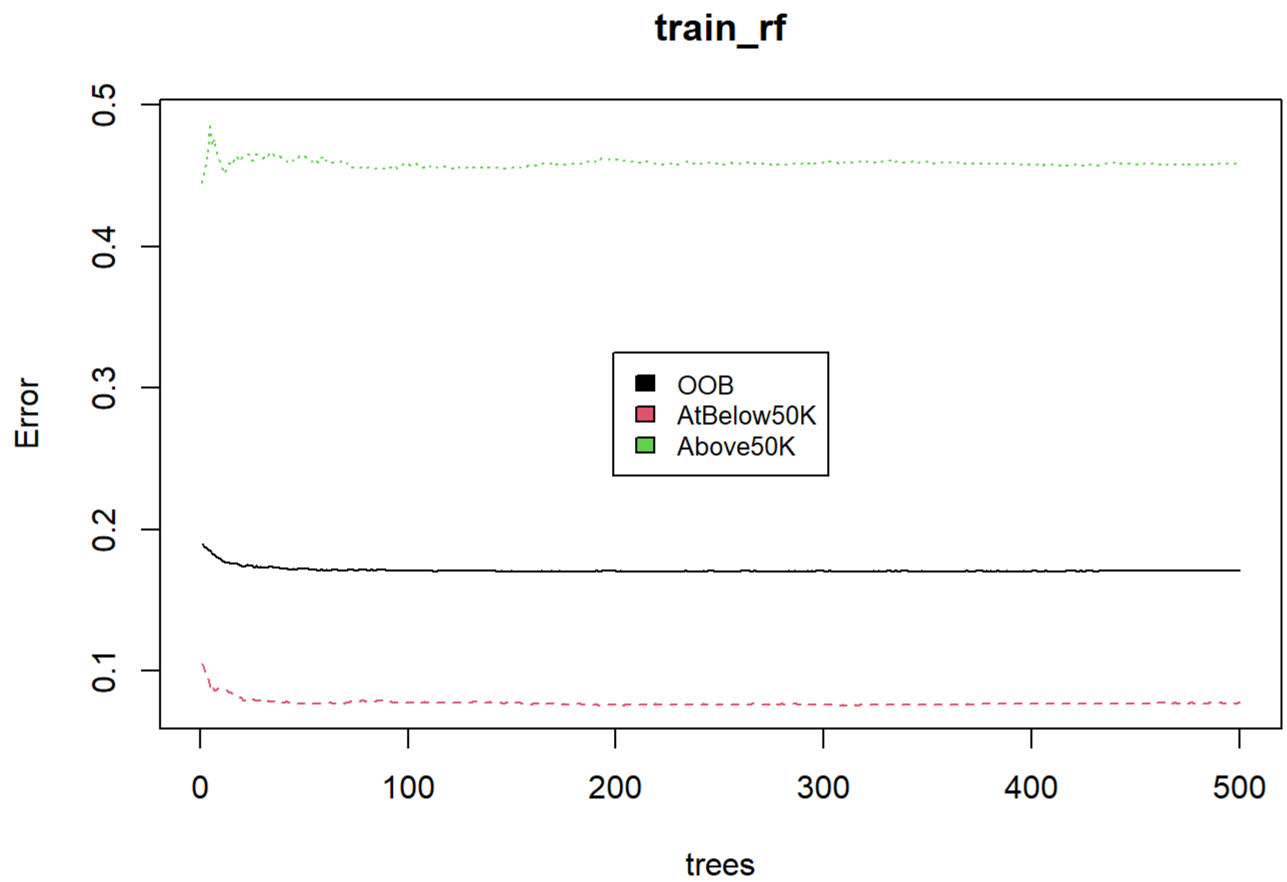
```
#record the sensitivity, specificity, and prevalence
sensitivity_rf <- cm_rf$byClass[["Sensitivity"]]
specificity_rf <- cm_rf$byClass[["Specificity"]]
prevalence_rf <- cm_rf$byClass[["Prevalence"]]
f1_rf <- cm_rf$byClass[["F1"]]

#Find the ROC and plot it. Show the AUC as well
pROC_bin <- ROCit::rocit(ifelse(adultpayclean_validation$income == "Above50K",1,0), ifelse(unname(y_hat_rf) == "TRUE",1,0),m
ethod="bin")
ciROC_bin95 <- ROCit::ciROC(pROC_bin,level = 0.95)
plot(ciROC_bin95, col = 1, values=TRUE)
lines(ciROC_bin95$TPR~ciROC_bin95$FPR, col = 2, lwd = 2)
```



```r
ROCit::ciAUC(pROC_bin)
```

```
##
##    estimated AUC : 0.818043983830059
##    AUC estimation method : binormal
##
##    CI of AUC
##    confidence level = 95%
##    lower = 0.795999705923417     upper = 0.840088261736701
```

```r
# Plot the error rate chart for the random forest
plot(train_rf)
legend("center", ifelse (colnames(train_rf$err.rate) == "FALSE","AtBelow50K",ifelse (colnames(train_rf$err.rate) == "TRUE",
"Above50K","OOB")),col=1:4,cex=0.8,fill=1:4)
```

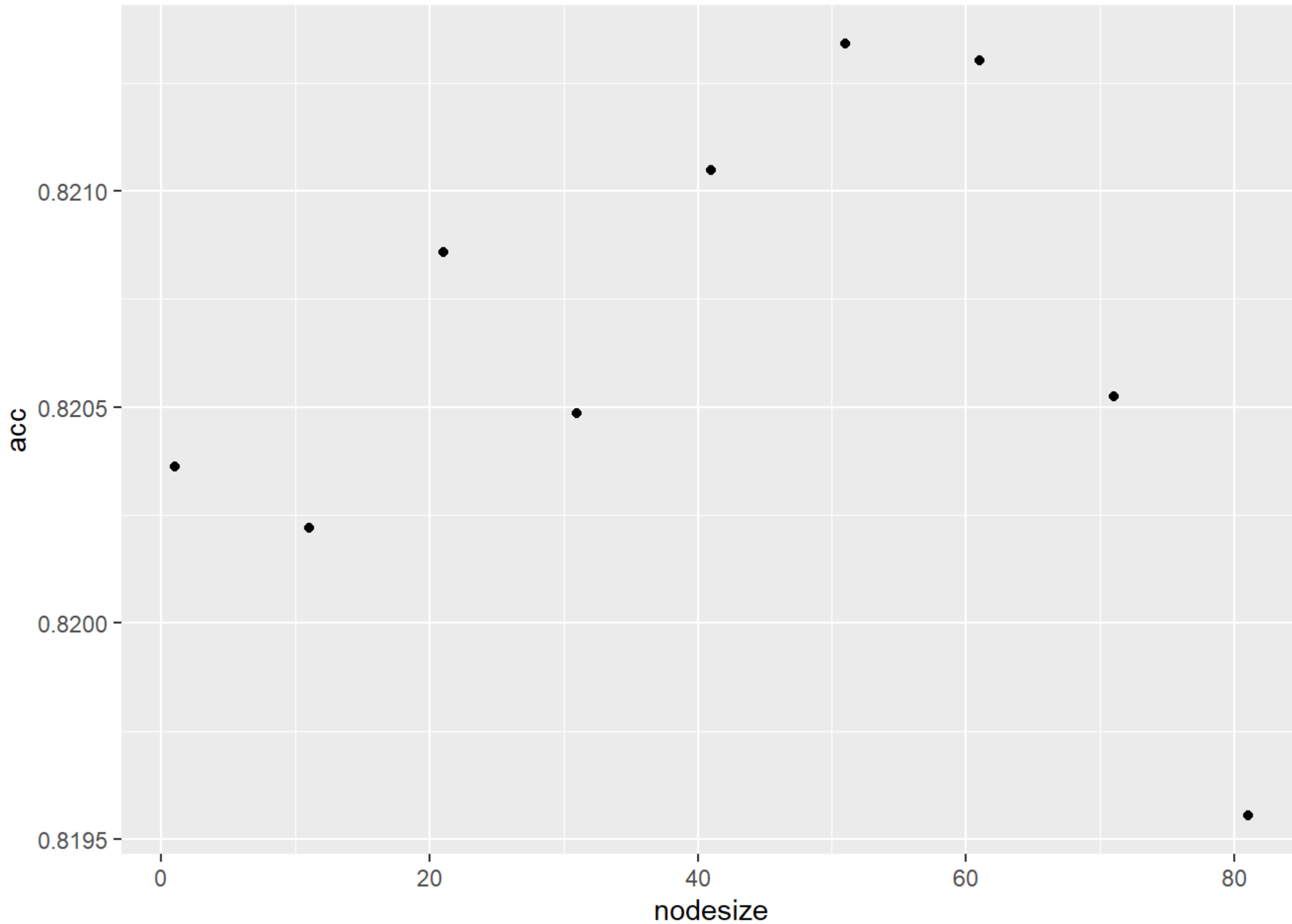

train_rf
Error
0.1
0.2
0.3
0.4
0.5
trees
0
100
200
300
400
500
OOB
AtBelow50K
Above50K

```r
set.seed(2008)
#random forest with tuning
nodesize <- seq(1, 90, 10)
acc <- sapply(nodesize, function(ns) {
  #train the model with tuning
  train(
    y ~ age + eduyears + sex + race + hoursperweek + maritalstatus + relationship + education + occupation + class,
    method = "rf",
    data = temp,
    tuneGrid = data.frame(mtry = 2),
    nodesize = ns
  )$results$Accuracy
})
```

```
## Warning in (function (kind = NULL, normal.kind = NULL, sample.kind = NULL) :
## non-uniform 'Rounding' sampler used
```

```
## Warning in (function (kind = NULL, normal.kind = NULL, sample.kind = NULL) :
## non-uniform 'Rounding' sampler used

## Warning in (function (kind = NULL, normal.kind = NULL, sample.kind = NULL) :
## non-uniform 'Rounding' sampler used

## Warning in (function (kind = NULL, normal.kind = NULL, sample.kind = NULL) :
## non-uniform 'Rounding' sampler used

## Warning in (function (kind = NULL, normal.kind = NULL, sample.kind = NULL) :
## non-uniform 'Rounding' sampler used

## Warning in (function (kind = NULL, normal.kind = NULL, sample.kind = NULL) :
## non-uniform 'Rounding' sampler used

## Warning in (function (kind = NULL, normal.kind = NULL, sample.kind = NULL) :
## non-uniform 'Rounding' sampler used

## Warning in (function (kind = NULL, normal.kind = NULL, sample.kind = NULL) :
## non-uniform 'Rounding' sampler used

## Warning in (function (kind = NULL, normal.kind = NULL, sample.kind = NULL) :
## non-uniform 'Rounding' sampler used
```

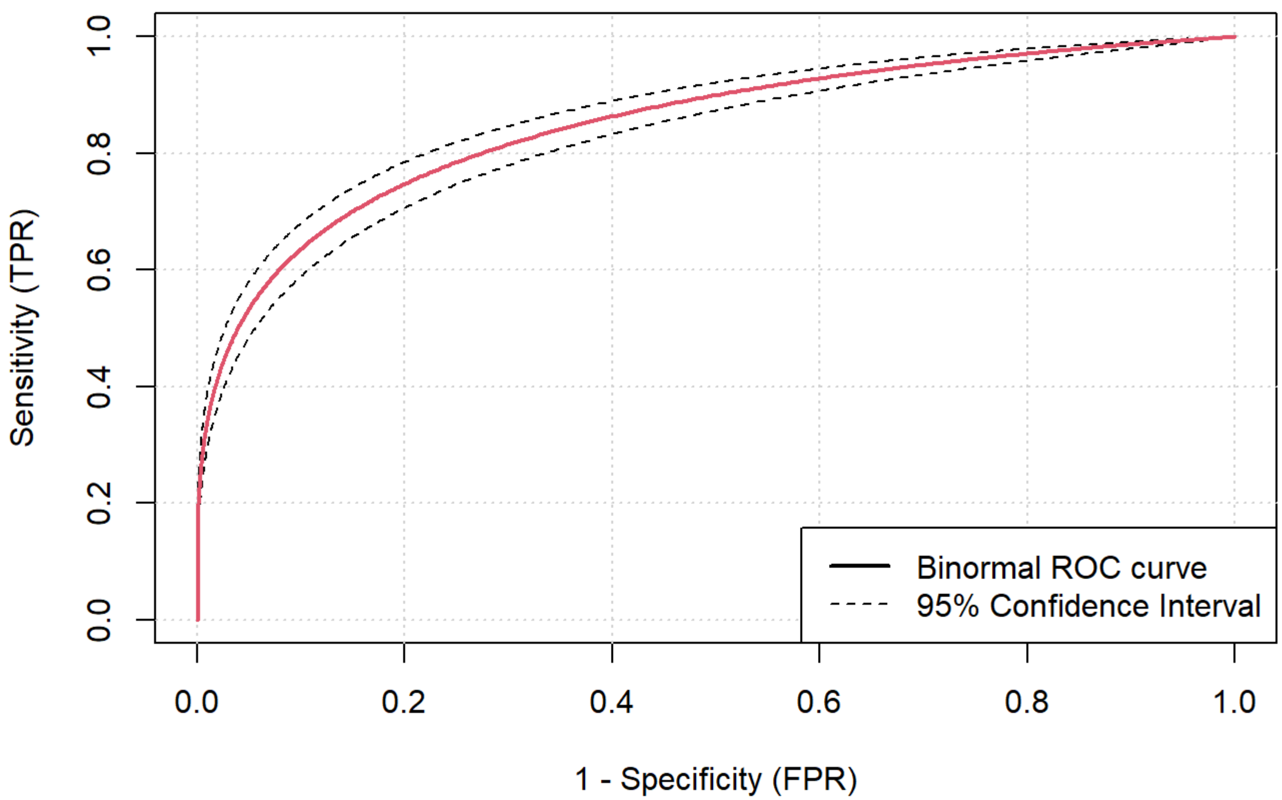
```
qplot(nodesize, acc)
```


0.8210
0.8205
0.8200
0.8195
acc
0
20
40
60
80
nodesize

```r
set.seed(2008)
#get the trained model for the max node size
train_rf_2 <-
  randomForest(
    y ~ age + eduyears + sex + race + hoursperweek + maritalstatus + relationship + education + occupation + class,
    data = temp,
    nodesize = nodesize[which.max(acc)]
  )
#predict the outcomes
y_hat_rf2 <- predict(train_rf_2, adultpayclean_validation)
#get the confusion matrix for random forest model
cm_rf2 <-
  confusionMatrix(
    y_hat_rf2,
    as.factor(adultpayclean_validation$income  == "Above50K")
  )
#get the accuracy
accuracy_rftune <-
  confusionMatrix(
    y_hat_rf2,
    as.factor(adultpayclean_validation$income  == "Above50K")
  )$overall["Accuracy"]

cm_rf2
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction FALSE TRUE
##      FALSE  2024  291
##      TRUE    176  427
##
##                Accuracy : 0.84
##                  95% CI : (0.8261, 0.8531)
##     No Information Rate : 0.7539
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.5441
##
##  Mcnemar's Test P-Value : 1.325e-07
##
##             Sensitivity : 0.9200
##             Specificity : 0.5947
##          Pos Pred Value : 0.8743
##          Neg Pred Value : 0.7081
##              Prevalence : 0.7539
##          Detection Rate : 0.6936
##    Detection Prevalence : 0.7934
##       Balanced Accuracy : 0.7574
##
##        'Positive' Class : FALSE
##
```

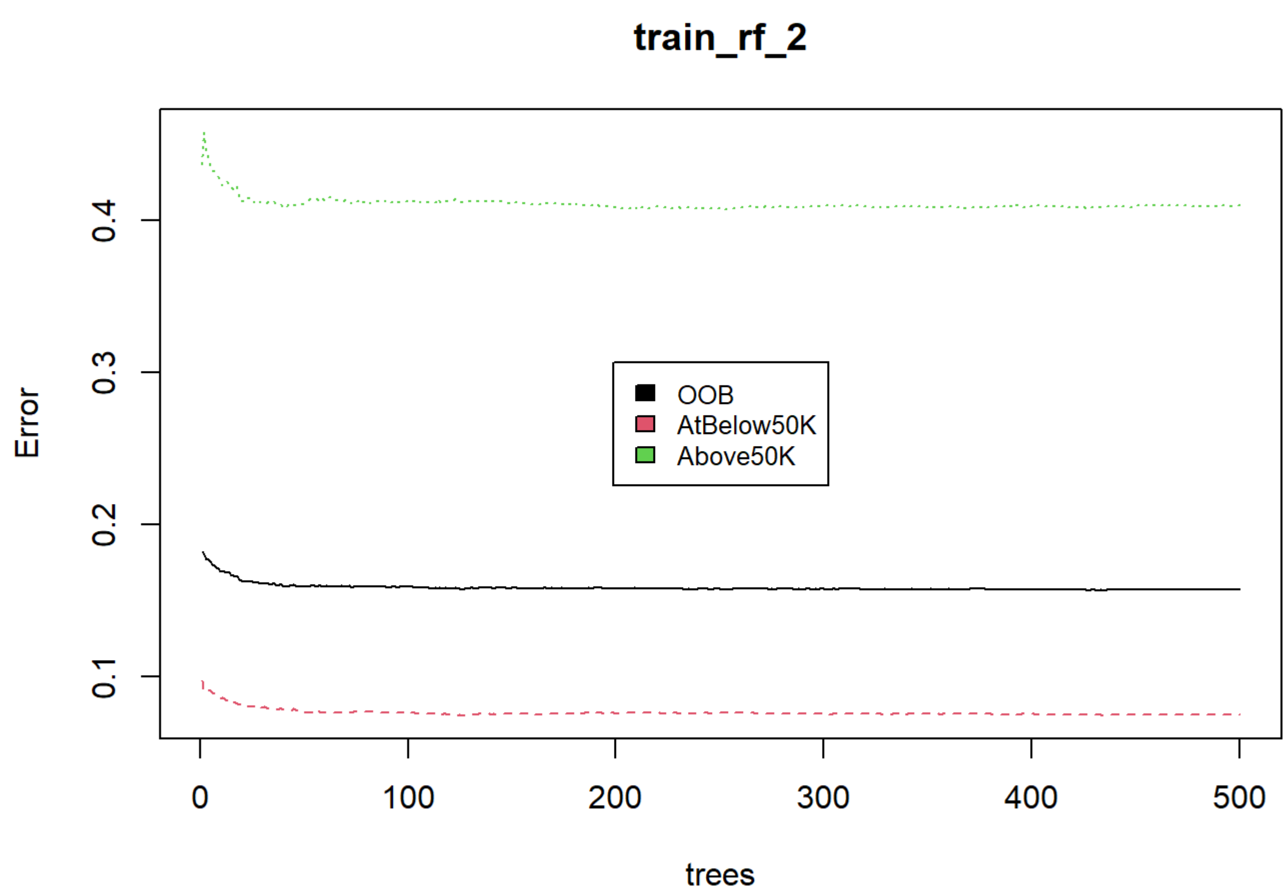
```
#record the sensitivity, specificity, and prevalence
sensitivity_rf2 <- cm_rf2$byClass[["Sensitivity"]]
specificity_rf2 <- cm_rf2$byClass[["Specificity"]]
prevalence_rf2 <- cm_rf2$byClass[["Prevalence"]]
f1_rf2 <- cm_rf2$byClass[["F1"]]

#Find the ROC and plot it. Show the AUC as well
pROC_bin <- ROCit::rocit(ifelse(adultpayclean_validation$income == "Above50K",1,0), ifelse(unname(y_hat_rf2) == "TRUE",1,0),
method="bin")
ciROC_bin95 <- ROCit::ciROC(pROC_bin,level = 0.95)
plot(ciROC_bin95, col = 1, values=TRUE)
lines(ciROC_bin95$TPR~ciROC_bin95$FPR, col = 2, lwd = 2)
```



```r
ROCit::ciAUC(pROC_bin)
```

```
##
##    estimated AUC : 0.849695675408264
##    AUC estimation method : binormal
##
##    CI of AUC
##    confidence level = 95%
##    lower = 0.829656180065837     upper = 0.869735170750691
```

```r
# Plot the error rate chart for the random forest
plot(train_rf_2)
legend("center", ifelse (colnames(train_rf_2$err.rate) == "FALSE","AtBelow50K",ifelse (colnames(train_rf_2$err.rate) == "TRU
E","Above50K","OOB")),col=1:4,cex=0.8,fill=1:4)
```


train_rf_2
Error
0.1
0.2
0.3
0.4
OOB
AtBelow50K
Above50K
0
100
200
300
400
500
trees

```r
# tabulate all the accuracy results with sensitivity and specificity
accuracy_results <-
  matrix(
    c(
      "Plain old guess",
      round(accuracy_guess, 5),
      round(sensitivity_guess, 5),
      round(specificity_guess, 5),
      round(prevalence_guess, 5),
      round(f1_guess, 5),
      "linear model",
      round(accuracy_lm, 5),
      round(sensitivity_lm, 5),
      round(specificity_lm, 5),
      round(prevalence_lm, 5),
      round(f1_lm, 5),
      "General linear model",
      round(accuracy_glm, 5),
      round(sensitivity_glm, 5),
      round(specificity_glm, 5),
      round(prevalence_glm, 5),
      round(f1_glm, 5),
      "naive bayes",
      round(accuracy_nb, 5),
      round(sensitivity_nb, 5),
      round(specificity_nb, 5),
      round(prevalence_nb, 5),
      round(f1_nb, 5),
      "knn",
      round(accuracy_knn, 5),
      round(sensitivity_knn, 5),
      round(specificity_knn, 5),
      round(prevalence_knn, 5),
      round(f1_knn, 5),
      "knn tune",
      round(accuracy_knntune, 5),
      round(sensitivity_knntune, 5),
      round(specificity_knntune, 5),
      round(prevalence_knntune, 5),
```

```
      round(f1_knntune, 5),
      "rpart",
      round(accuracy_rpart, 5),
      round(sensitivity_rpart, 5),
      round(specificity_rpart, 5),
      round(prevalence_rpart, 5),
      round(f1_rpart, 5),
      "rf",
      round(accuracy_rf, 5),
      round(sensitivity_rf, 5),
      round(specificity_rf, 5),
      round(prevalence_rf, 5),
      round(f1_rf, 5),
      "rf tune",
      round(accuracy_rftune, 5),
      round(sensitivity_rf2, 5),
      round(specificity_rf2, 5),
      round(prevalence_rf2, 5),
      round(f1_rf2, 5)
    ),
    nrow = 9,
    ncol = 6,
    byrow = TRUE,
    dimnames = list(
      c("1.", "2.", "3.", "4.", "5.", "6.", "7.", "8.", "9."),
      c(
        "Method",
        "Accuracy",
        "Sensitivity",
        "Specificity",
        "Prevalence",
        "F1"
      )
    )
  )
#style the table with knitr
accuracy_results %>% knitr::kable() %>%
  kable_styling(bootstrap_options = c("striped", "hover", "condensed"))
```

	Method	Accuracy	Sensitivity	Specificity	Prevalence	F1
1.	Plain old guess	0.50822	0.50279	0.51	0.24606	0.33472
2.	linear model	0.81528	0.4429	0.93682	0.24606	0.54128
3.	General linear model	0.81049	0.38719	0.94864	0.24606	0.50135
4.	naive bayes	0.80089	0.91112	0.57218	0.67478	0.86064
5.	knn	0.80535	0.90545	0.49861	0.75394	0.87522
6.	knn tune	0.8098	0.90636	0.51114	0.75394	0.87745
7.	rpart	0.82111	0.91	0.54875	0.75394	0.88467
8.	rf	0.8218	0.91455	0.5376	0.75394	0.88556
9.	rf tune	0.83996	0.92	0.59471	0.75394	0.89657