CensusPay.R

rharidas

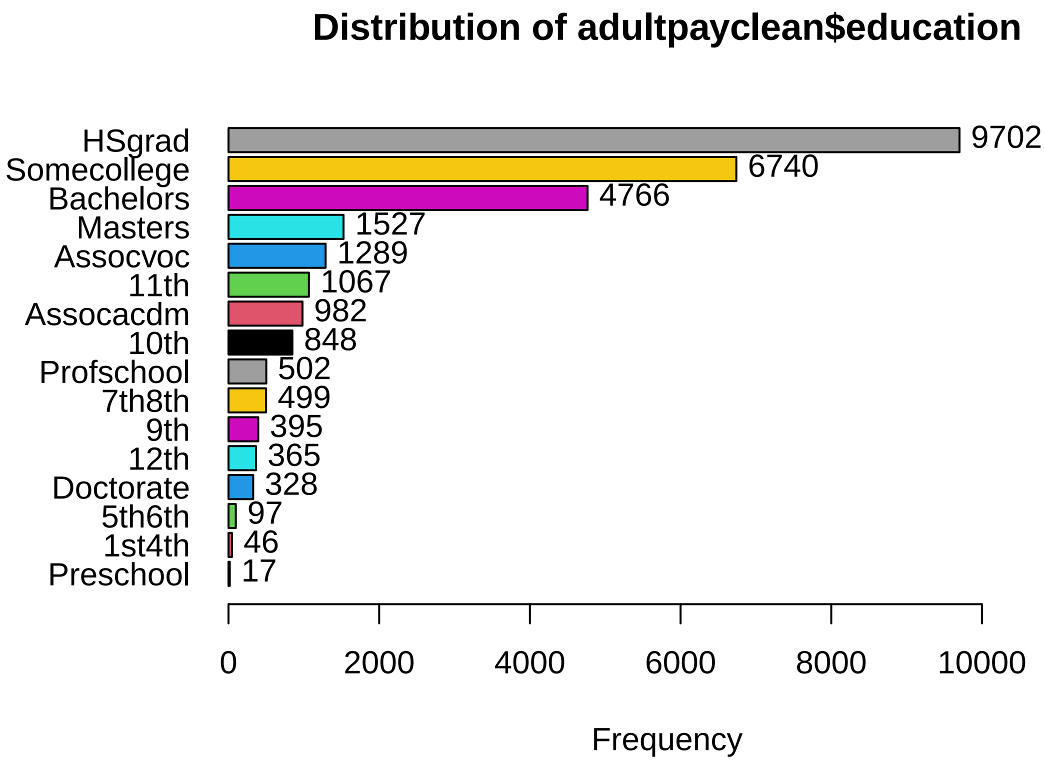
2021-11-20

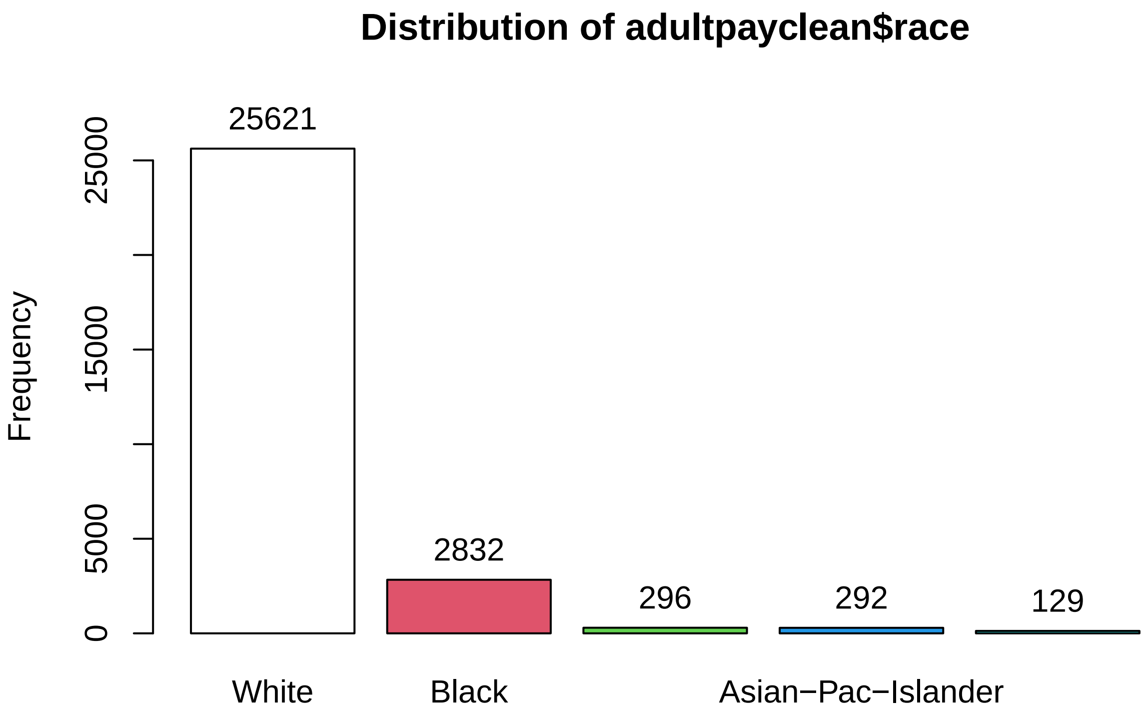
```r
#Note: This script will take a while to run. In particular the knn and random forest algorithms with tu
# more time. please be patient if you happen to execute it. The execution report is available in the gi

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

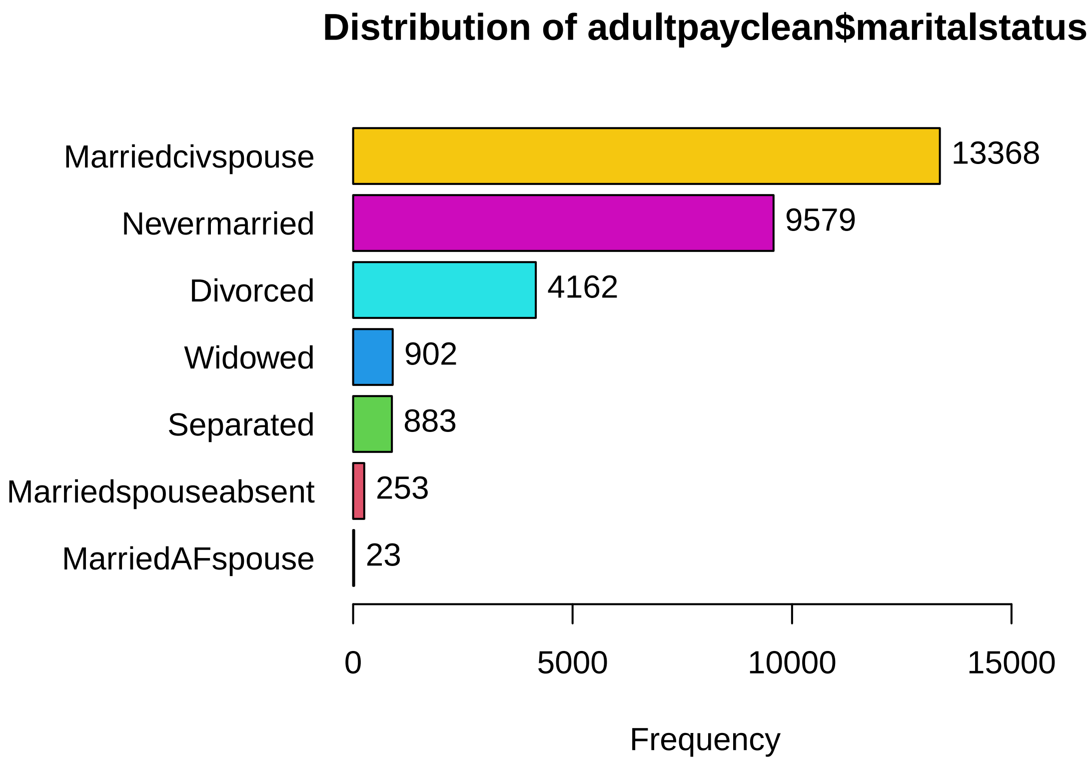
```
## Loading required package: tidyverse

## -- 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'
```

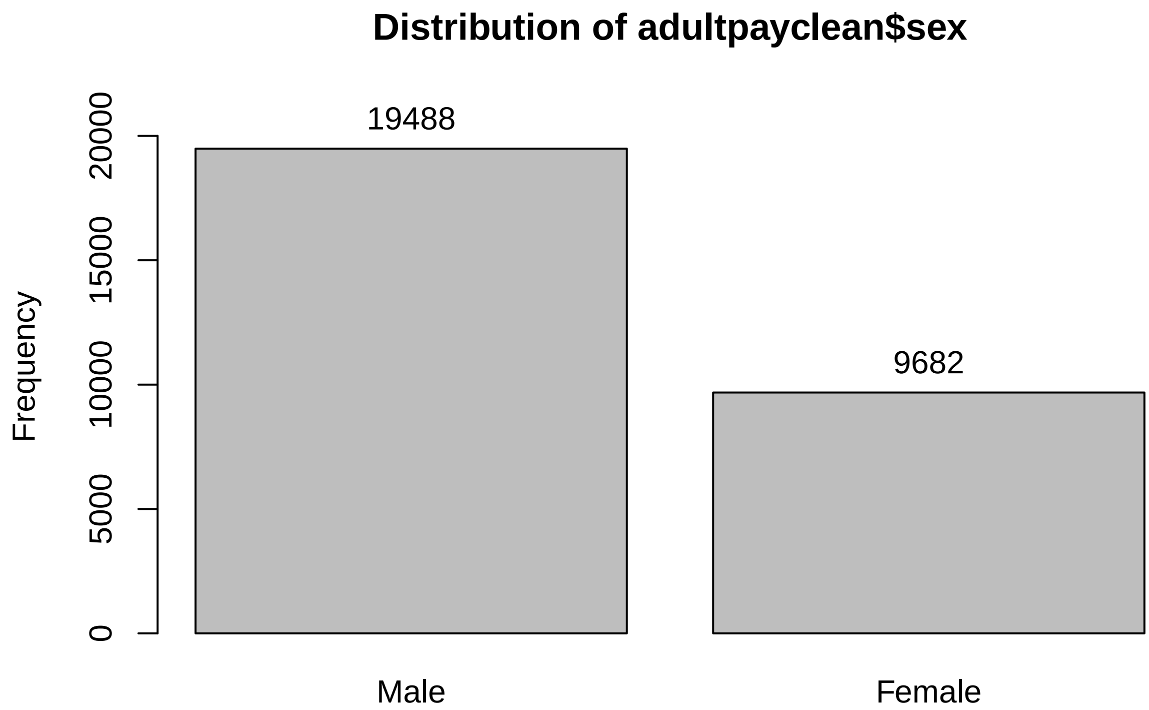
```
## The following object is masked from 'package:lattice':
##
##     dotplot

## 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~
## $ 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> Unknown, Execmanagerial, Unknown, Machineopinspct, Profs~
## $ 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> Unknown, Private, Unknown, Private, Private, Private, Pr~
```

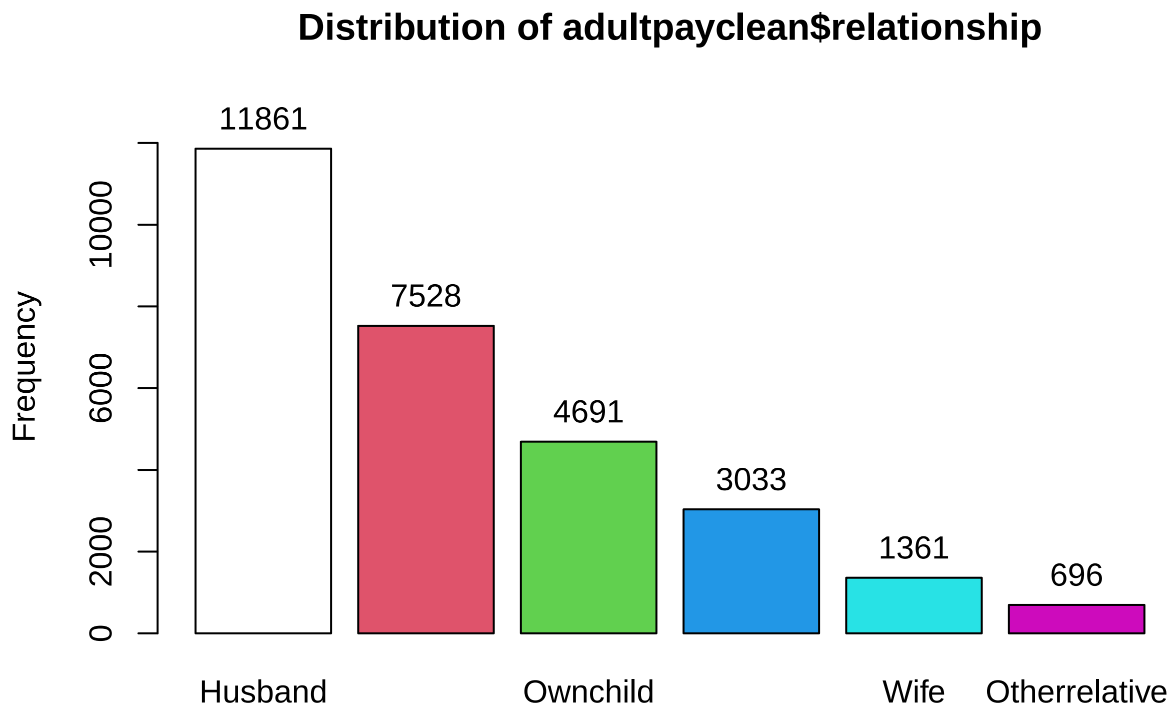

Distribution of adultpayclean$education
HSgrad
Somecollege
Bachelors
Masters
Assocvoc
11th
Assocacdm
10th
Profschool
7th8th
9th
12th
Doctorate
5th6th
1st4th
Preschool
9702
6740
4766
1527
1289
1067
982
848
502
499
395
365
328
97
46
17
0
2000
4000
6000
8000
10000
Frequency

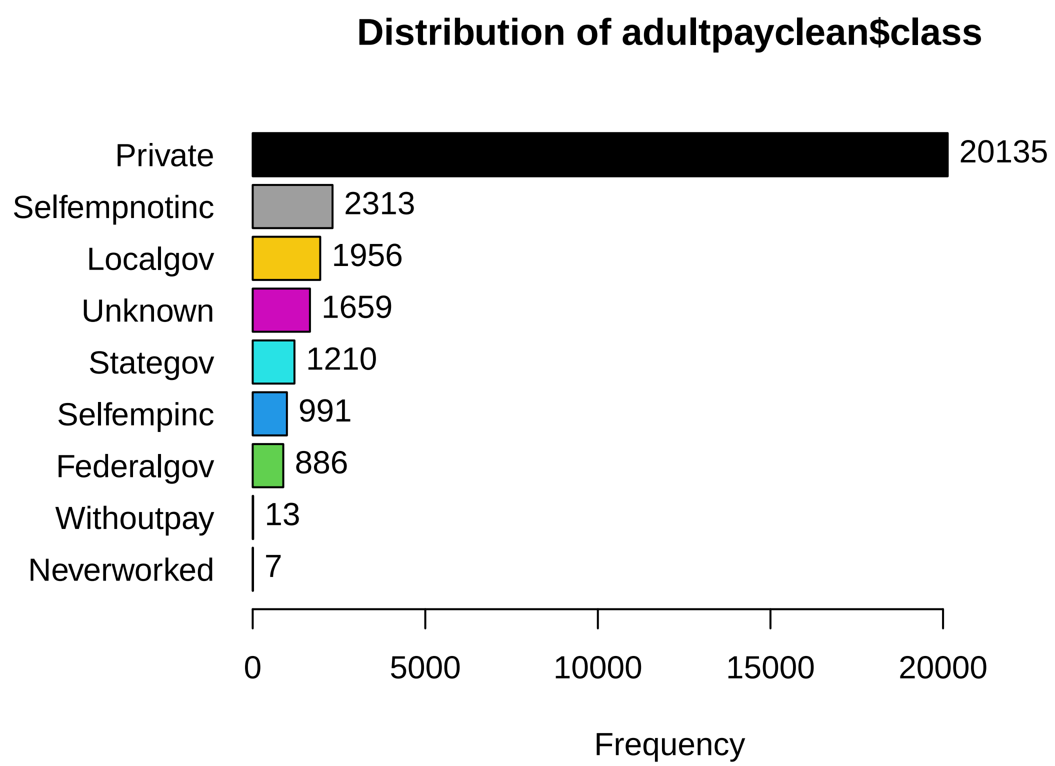

Distribution of adultpayclean$race
Frequency
0
5000
15000
25000
25621
2832
296
292
129
White
Black
Asian−Pac−Islander



Distribution of adultpayclean$maritalstatus
Marriedcivspouse
13368
Nevermarried
9579
Divorced
4162
Widowed
902
Separated
883
Marriedspouseabsent
253
MarriedAFspouse
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
Ownchild
Wife
Otherrelative




```r
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
```

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

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

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

```r
library(caret)
library(gridExtra)
library(kableExtra)
library(randomForest)
library(purrr)
library(e1071)
library(caTools)


set.seed(1996, sample.kind = "Rounding")
```

```
## Warning in set.seed(1996, 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(adu
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          347       1087
##   AtBelow50K        371       1113
```

```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      0.483
## 2 AtBelow50K    0.506
```

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

```
## Confusion Matrix and Statistics
##
##             Reference
## Prediction   Above50K AtBelow50K
##   Above50K        347       1087
##   AtBelow50K      371       1113
##
##                Accuracy : 0.5003
##                  95% CI : (0.482, 0.5186)
##     No Information Rate : 0.7539
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : -0.0081
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.4833
##             Specificity : 0.5059
##          Pos Pred Value : 0.2420
##          Neg Pred Value : 0.7500
##              Prevalence : 0.2461
##          Detection Rate : 0.1189
##    Detection Prevalence : 0.4914
##       Balanced Accuracy : 0.4946
##
##        'Positive' Class : Above50K
##
```

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

#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,
     data = .)

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

#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        344        161
##   AtBelow50K      374       2039
##
##                Accuracy : 0.8167
##                  95% CI : (0.8021, 0.8305)
##     No Information Rate : 0.7539
##     P-Value [Acc > NIR] : 2.788e-16
##
##                   Kappa : 0.451
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.4791
##             Specificity : 0.9268
##          Pos Pred Value : 0.6812
##          Neg Pred Value : 0.8450
##              Prevalence : 0.2461
##          Detection Rate : 0.1179
##    Detection Prevalence : 0.1731
##       Balanced Accuracy : 0.7030
##
##        '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"]]

#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,
    data = .,
    family = "binomial"
  )

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

# 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
cm_glm <-
  confusionMatrix(y_hat_logit, adultpayclean_validation$income)
accuracy_glm <-
```

```r
  confusionMatrix(y_hat_logit, adultpayclean_validation$income)$overall[["Accuracy"]]

cm_glm
```

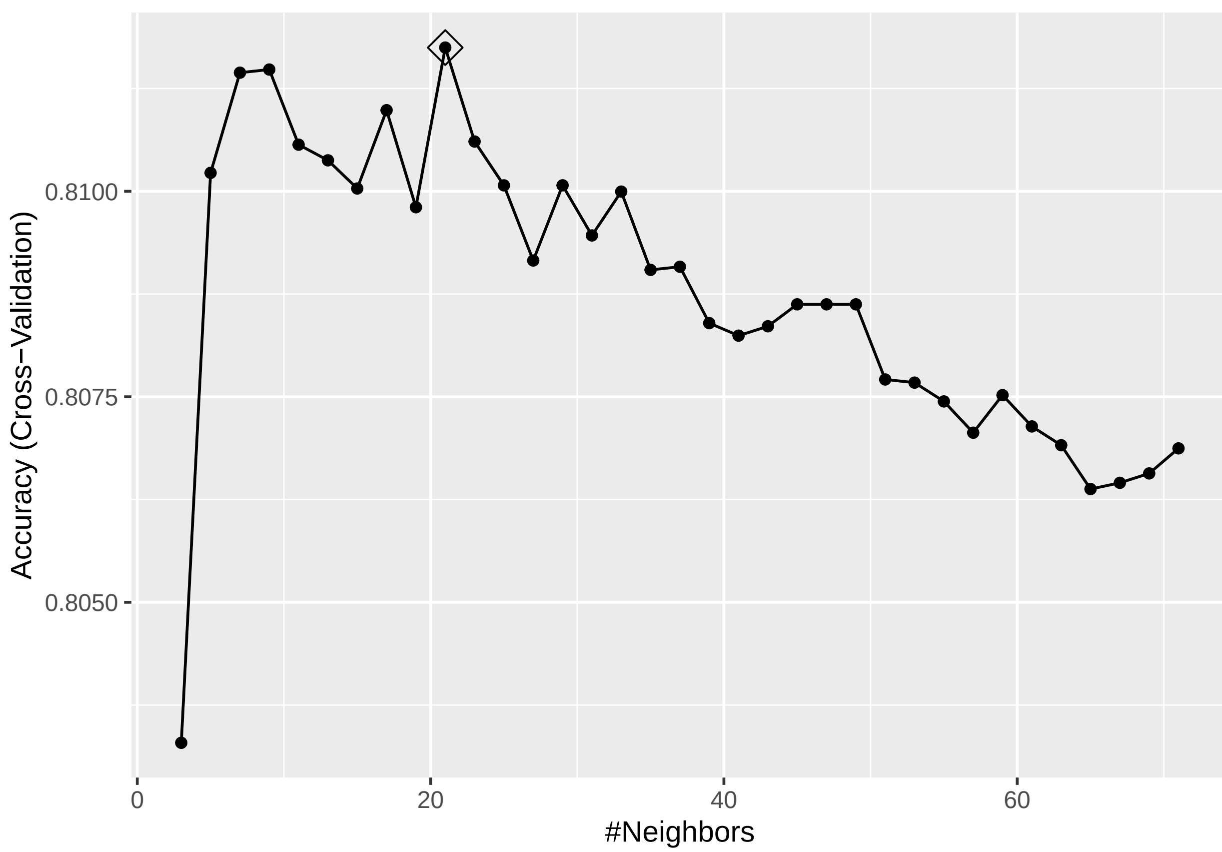
```
## Confusion Matrix and Statistics
##
##             Reference
## Prediction   Above50K AtBelow50K
##   Above50K        279        116
##   AtBelow50K      439       2084
##
##                Accuracy : 0.8098
##                  95% CI : (0.7951, 0.8239)
##     No Information Rate : 0.7539
##     P-Value [Acc > NIR] : 3.442e-13
##
##                   Kappa : 0.3958
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.38858
##             Specificity : 0.94727
##          Pos Pred Value : 0.70633
##          Neg Pred Value : 0.82600
##              Prevalence : 0.24606
##          Detection Rate : 0.09561
##    Detection Prevalence : 0.13537
##       Balanced Accuracy : 0.66793
##
##        '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"]]

#Naive bayes

train_nb <- adultpayclean_train %>%
   mutate(y = as.factor(income == "Above50K")) %>%
   naiveBayes(y ~ age + eduyears + sex + race + hoursperweek + maritalstatus + relationship,data = .)

y_hat_nb <- predict(train_nb, newdata = adultpayclean_validation)
cm_tab <- table(adultpayclean_validation$income == "Above50K", y_hat_nb)
cm_nb <- confusionMatrix(cm_tab)

accuracy_nb <- cm_nb$overall[["Accuracy"]]
sensitivity_nb <- cm_nb$byClass[["Sensitivity"]]
specificity_nb <- cm_nb$byClass[["Specificity"]]
prevalence_nb <- cm_nb$byClass[["Prevalence"]]

# translate income factor into binary outcome
```

```r
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,
    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)
```



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

```
##     k
## 10 21
```

```r
train_knn$finalModel
```

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

```r
#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  1988  348
##      TRUE    212  370
##
##                Accuracy : 0.8081
##                  95% CI : (0.7933, 0.8222)
##     No Information Rate : 0.7539
##     P-Value [Acc > NIR] : 1.777e-12
##
##                   Kappa : 0.4475
##
##  Mcnemar's Test P-Value : 1.165e-08
##
##             Sensitivity : 0.9036
##             Specificity : 0.5153
##          Pos Pred Value : 0.8510
##          Neg Pred Value : 0.6357
##              Prevalence : 0.7539
##          Detection Rate : 0.6813
##    Detection Prevalence : 0.8005
##       Balanced Accuracy : 0.7095
```

```
##
##        'Positive' Class : FALSE
##
```

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

#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,
      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,
    data = temp,
    k = 17
  )

y_hat <- predict(knn_fit, temp, type = "class")
cm_knntune <- confusionMatrix(y_hat, temp$y)

cm_knntune
```

```
## Confusion Matrix and Statistics
```

```
##
##            Reference
## Prediction FALSE  TRUE
##      FALSE 18001  2772
##      TRUE   1798  3681
##
##                Accuracy : 0.8259
##                  95% CI : (0.8213, 0.8305)
##     No Information Rate : 0.7542
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.5053
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.9092
##             Specificity : 0.5704
##          Pos Pred Value : 0.8666
##          Neg Pred Value : 0.6718
##              Prevalence : 0.7542
##          Detection Rate : 0.6857
##    Detection Prevalence : 0.7913
##       Balanced Accuracy : 0.7398
##
##        'Positive' Class : FALSE
##
```

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

#k-nearest using knn3
set.seed(2008)
knn3_fit <-
  knn3(
    y ~ age + eduyears + sex + race + hoursperweek + maritalstatus + relationship,
    data = temp,
    k = 17
  )
y_hat_knn3 <-
  predict(knn3_fit, adultpayclean_validation, type = "class")

cm_knn3 <-
  confusionMatrix(y_hat_knn3,
                  as.factor(adultpayclean_validation$income  == "Above50K"))
accuracy_knn3 <-
  confusionMatrix(y_hat_knn3,
                  as.factor(adultpayclean_validation$income  == "Above50K"))$overall["Accuracy"]

cm_knn3
```

```
## Confusion Matrix and Statistics
##
```

```
##           Reference
## Prediction FALSE TRUE
##      FALSE  1983  342
##      TRUE    217  376
##
##                Accuracy : 0.8084
##                  95% CI : (0.7937, 0.8226)
##     No Information Rate : 0.7539
##     P-Value [Acc > NIR] : 1.285e-12
##
##                   Kappa : 0.4515
##
##  Mcnemar's Test P-Value : 1.566e-07
##
##             Sensitivity : 0.9014
##             Specificity : 0.5237
##          Pos Pred Value : 0.8529
##          Neg Pred Value : 0.6341
##              Prevalence : 0.7539
##          Detection Rate : 0.6796
##    Detection Prevalence : 0.7968
##       Balanced Accuracy : 0.7125
##
##        'Positive' Class : FALSE
##
```

```
#record the sensitivity, specificity, and prevalence
sensitivity_knn3 <- cm_knn3$byClass[["Sensitivity"]]
specificity_knn3 <- cm_knn3$byClass[["Specificity"]]
prevalence_knn3 <- cm_knn3$byClass[["Prevalence"]]


#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,
    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
```

```
#predict the outcomes with this model
y_hat <- predict(train_rpart, adultpayclean_validation)
#confusion matrix for the rpart model
cm_rpart <-
  confusionMatrix(y_hat,
                  as.factor(adultpayclean_validation$income  == "Above50K"))
#get the accuracy
```

```r
accuracy_rpart <-
  confusionMatrix(y_hat,
                  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
##
```

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

#random forest
set.seed(2008)
#train the vanilla random forest model
train_rf <-
  randomForest(y ~ age + eduyears + sex + race + hoursperweek + maritalstatus + relationship,
               data = temp)
#create the confusionMatrix
cm_rf <-
  confusionMatrix(
    predict(train_rf, adultpayclean_validation),
    as.factor(adultpayclean_validation$income  == "Above50K")
  )
#get the accuracy
accuracy_rf <-
```

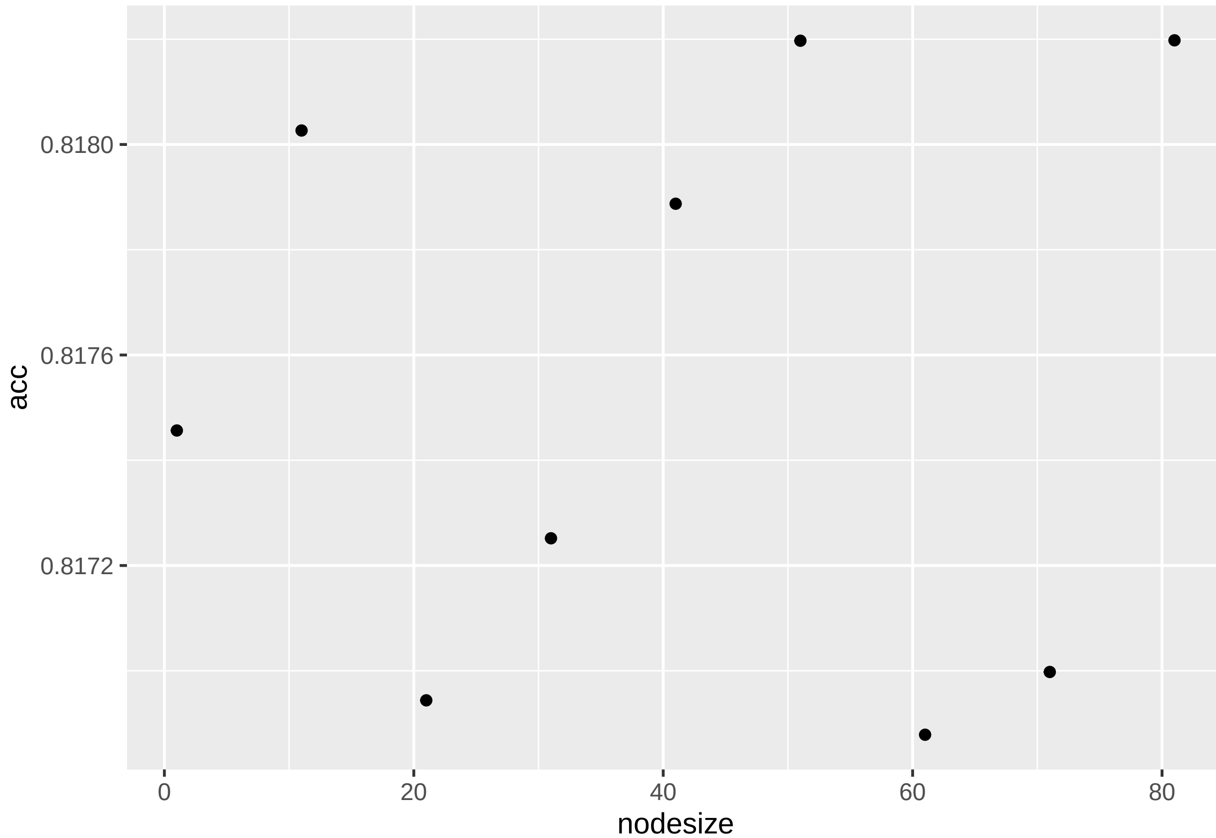
```r
  confusionMatrix(
    predict(train_rf, adultpayclean_validation),
    as.factor(adultpayclean_validation$income  == "Above50K")
  )$overall["Accuracy"]

cm_rf
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction FALSE TRUE
##      FALSE  2016  331
##      TRUE    184  387
##
##                Accuracy : 0.8235
##                  95% CI : (0.8092, 0.8372)
##     No Information Rate : 0.7539
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.4891
##
##  Mcnemar's Test P-Value : 1.247e-10
##
##             Sensitivity : 0.9164
##             Specificity : 0.5390
##          Pos Pred Value : 0.8590
##          Neg Pred Value : 0.6778
##              Prevalence : 0.7539
##          Detection Rate : 0.6909
##    Detection Prevalence : 0.8043
##       Balanced Accuracy : 0.7277
##
##        'Positive' Class : FALSE
##
```

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


#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,
    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)
```




```
#get the trained model for the max node size
train_rf_2 <-
  randomForest(
    y ~ age + eduyears + sex + race + hoursperweek + maritalstatus + relationship,
    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(
    predict(train_rf_2, adultpayclean_validation),
    as.factor(adultpayclean_validation$income  == "Above50K")
  )
#get the accuracy
accuracy_rftune <-
  confusionMatrix(
    predict(train_rf_2, adultpayclean_validation),
    as.factor(adultpayclean_validation$income  == "Above50K")
  )$overall["Accuracy"]

cm_rf2
```

```
## Confusion Matrix and Statistics
##
```

```
##           Reference
## Prediction FALSE TRUE
##      FALSE  2031  348
##      TRUE    169  370
##
##                Accuracy : 0.8228
##                  95% CI : (0.8085, 0.8365)
##     No Information Rate : 0.7539
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.4787
##
##  Mcnemar's Test P-Value : 4.94e-15
##
##             Sensitivity : 0.9232
##             Specificity : 0.5153
##          Pos Pred Value : 0.8537
##          Neg Pred Value : 0.6865
##              Prevalence : 0.7539
##          Detection Rate : 0.6960
##    Detection Prevalence : 0.8153
##       Balanced Accuracy : 0.7193
##
##        'Positive' Class : FALSE
##
```

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

# 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),
      "linear model",
      round(accuracy_lm, 5),
      round(sensitivity_lm, 5),
      round(specificity_lm, 5),
      round(prevalence_lm, 5),
      "General linear model",
      round(accuracy_glm, 5),
      round(sensitivity_glm, 5),
      round(specificity_glm, 5),
      round(prevalence_glm, 5),
      "naive bayes",
      round(accuracy_nb, 5),
      round(sensitivity_nb, 5),
      round(specificity_nb, 5),
```

```r
      round(prevalence_nb, 5),
      "knn",
      round(accuracy_knn, 5),
      round(sensitivity_knn, 5),
      round(specificity_knn, 5),
      round(prevalence_knn, 5),
      "knn3",
      round(accuracy_knn3, 5),
      round(sensitivity_knn3, 5),
      round(specificity_knn3, 5),
      round(prevalence_knn3, 5),
      "knn tune",
      round(accuracy_knntune, 5),
      round(sensitivity_knntune, 5),
      round(specificity_knntune, 5),
      round(prevalence_knntune, 5),
      "rpart",
      round(accuracy_rpart, 5),
      round(sensitivity_rpart, 5),
      round(specificity_rpart, 5),
      round(prevalence_rpart, 5),
      "rf",
      round(accuracy_rf, 5),
      round(sensitivity_rf, 5),
      round(specificity_rf, 5),
      round(prevalence_rf, 5),
      "rf tune",
      round(accuracy_rftune, 5),
      round(sensitivity_rf2, 5),
      round(specificity_rf2, 5),
      round(prevalence_rf2, 5)
    ),
    nrow = 10,
    ncol = 5,
    byrow = TRUE,
    dimnames = list(
      c("1.", "2.", "3.", "4.", "5.", "6.", "7.", "8.", "9.","10."),
      c(
        "Method",
        "Accuracy",
        "Sensitivity",
        "Specificity",
        "Prevalence"
      )
    )
  )
#style the table with knitr
accuracy_results %>% knitr::kable() %>%
  kable_styling(bootstrap_options = c("striped", "hover", "condensed"))
```

	Method	Accuracy	Sensitivity	Specificity	Prevalence
1.	Plain old guess	0.50034	0.48329	0.50591	0.24606
2.	linear model	0.81666	0.47911	0.92682	0.24606
3.	General linear model	0.8098	0.38858	0.94727	0.24606
4.	naive bayes	0.77176	0.91192	0.52462	0.63811
5.	knn	0.80809	0.90364	0.51532	0.75394
6.	knn3	0.80843	0.90136	0.52368	0.75394
7.	knn tune	0.81151	0.90919	0.57043	0.75419
8.	rpart	0.82111	0.91	0.54875	0.75394
9.	rf	0.82351	0.91636	0.539	0.75394
10.	rf tune	0.82248	0.92318	0.51532	0.75394