R Notebook

The following is your first chunk to start with. Remember, you can add chunks using the menu above (Insert -> R) or using the keyboard shortcut Ctrl+Alt+I. A good practice is to use different code chunks to answer different questions. You can delete this comment if you like.

Other useful keyboard shortcuts include Alt- for the assignment operator, and Ctrl+Shift+M for the pipe operator. You can delete these reminders if you don't want them in your report.

```
#setwd("C:/") #Don't forget to set your working directory before you start!

library("tidyverse")

## ── Attaching packages ───────────────────────────────────────────────────────
── tidyverse 1.3.0 ──

## ✔ ggplot2 3.2.1     ✔ purrr   0.3.3
## ✔ tibble  2.1.3     ✔ dplyr   0.8.3
## ✔ tidyr   1.0.0     ✔ stringr 1.4.0
## ✔ readr   1.3.1     ✔ forcats 0.4.0

## ── Conflicts ────────────────────────────────────────────────────────────────
───── tidyverse_conflicts() ──
## ✖ dplyr::filter() masks stats::filter()
## ✖ dplyr::lag()    masks stats::lag()

library("tidymodels")

## ── Attaching packages ───────────────────────────────────────────────────────
─ tidymodels 0.0.3 ──

## ✔ broom     0.5.3     ✔ recipes   0.1.9
## ✔ dials     0.0.4     ✔ rsample   0.0.5
## ✔ infer     0.5.1     ✔ yardstick 0.0.4
## ✔ parsnip   0.0.5

## ── Conflicts ────────────────────────────────────────────────────────────────
──── tidymodels_conflicts() ──
## ✖ scales::discard()   masks purrr::discard()
## ✖ dplyr::filter()     masks stats::filter()
## ✖ recipes::fixed()    masks stringr::fixed()
## ✖ dplyr::lag()        masks stats::lag()
## ✖ dials::margin()     masks ggplot2::margin()
```

```
## ✖ yardstick::spec()   masks readr::spec()
## ✖ recipes::step()     masks stats::step()
## ✖ recipes::yj_trans() masks scales::yj_trans()
```

```r
library("plotly")
```

```
##
## Attaching package: 'plotly'

## The following object is masked from 'package:ggplot2':
##
##     last_plot

## The following object is masked from 'package:stats':
##
##     filter

## The following object is masked from 'package:graphics':
##
##     layout
```

```r
library("skimr")
library("caret")
```

```
## Loading required package: lattice

##
## Attaching package: 'caret'

## The following objects are masked from 'package:yardstick':
##
##     precision, recall

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

```r
# Q1. a)


dfc <- read_csv("assignment3Carvana.csv")
```

```
## Parsed with column specification:
## cols(
##   Auction = col_character(),
##   Age = col_double(),
##   Make = col_character(),
##   Color = col_character(),
##   WheelType = col_character(),
##   Odo = col_double(),
##   Size = col_character(),
##   MMRAauction = col_double(),
```

```
##   MMRAretail = col_double(),
##   BadBuy = col_double()
## )
```

```r
#dfc

skim(dfc)
```

Data summary

Name	dfc
Number of rows	10061
Number of columns	10
_______________________	
Column type frequency:	
character	5
numeric	5
________________________	
Group variables	None

Variable type: character

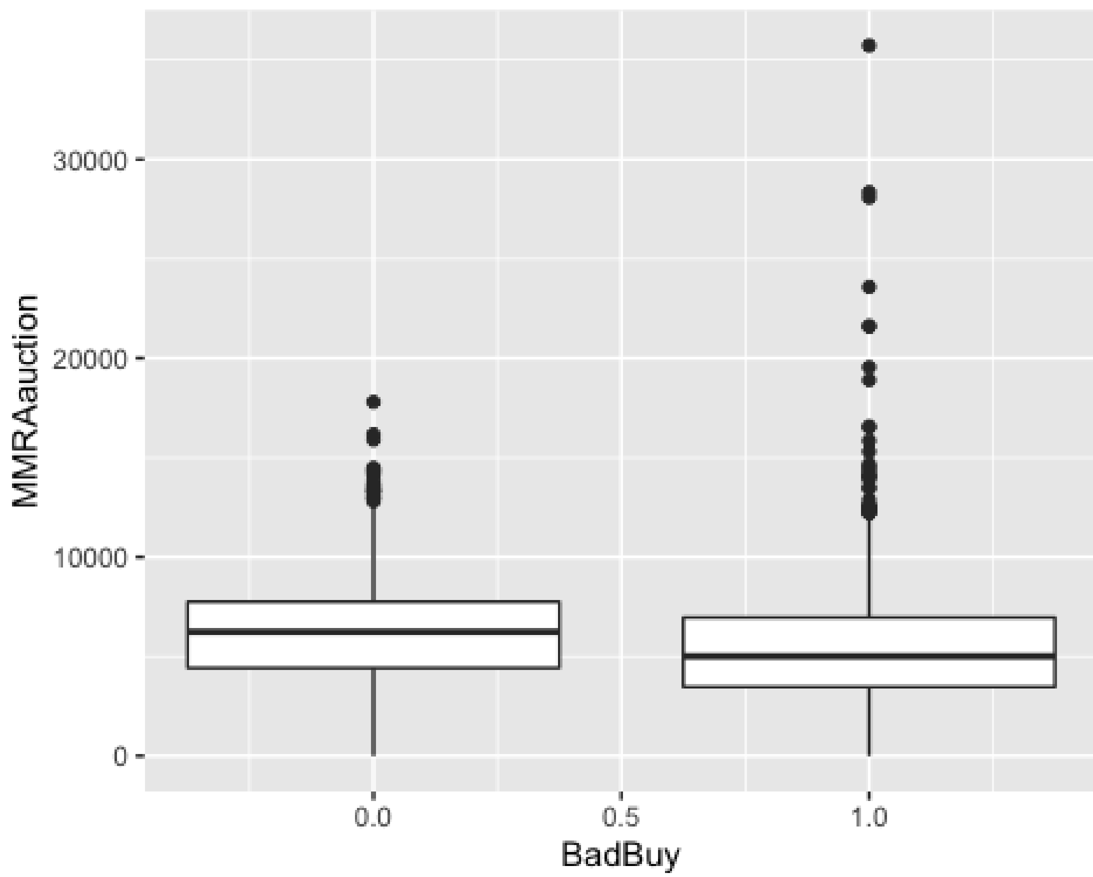
skim_variable	n_missing	complete_rate	min	max	empty	n_unique	whitespace
Auction	0	1	5	7	0	3	0
Make	0	1	3	10	0	30	0
Color	0	1	3	8	0	17	0
WheelType	0	1	4	7	0	4	0
Size	0	1	3	10	0	12	0

Variable type: numeric

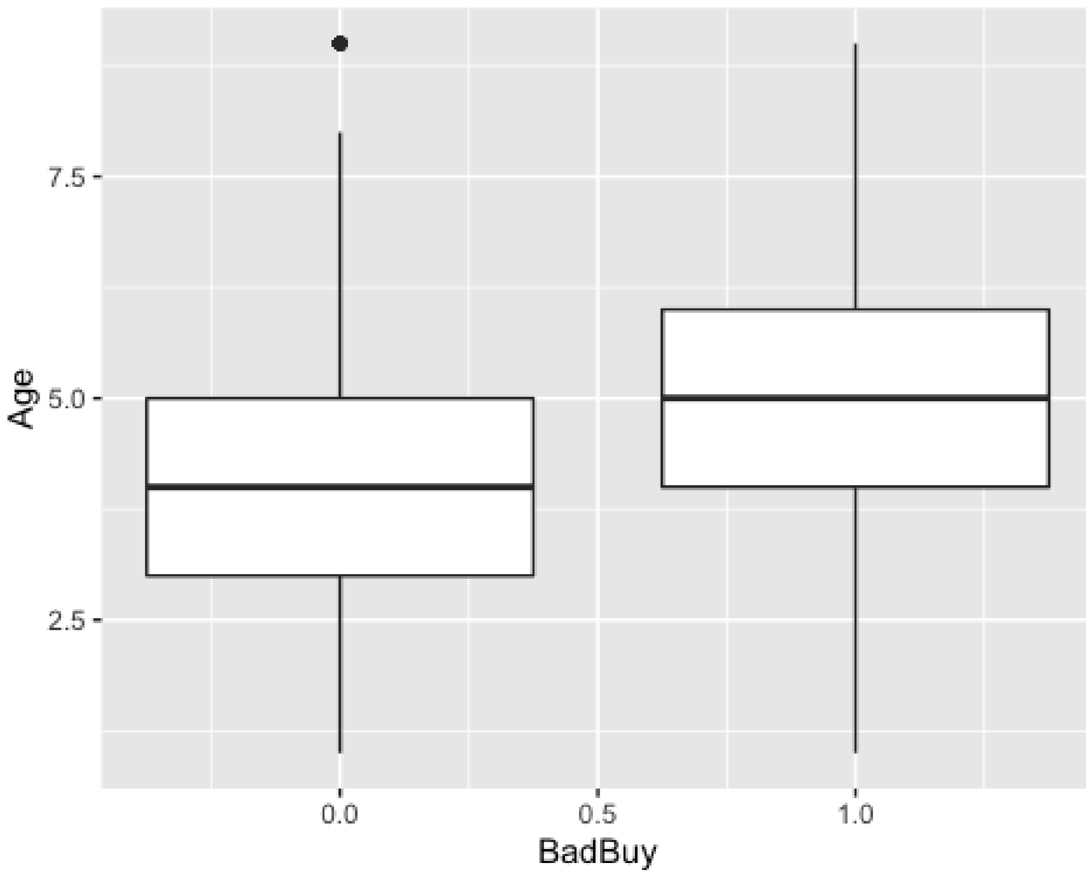
skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
Age	0	1	4.50	1.77	1	3	4	6	9	▂▇▃▅▁
Odo	0	1	72903.87	14498.87	9446	63488	74942	83663	115717	▁▂▆▇▁
MMRAauction	0	1	5812.38	2578.85	0	3877	5588	7450	35722	▇▃▁▁▁
MMRAretail	0	1	8171.51	3257.19	0	5872	8052	10315	39080	▇▇▁▁▁
BadBuy	0	1	0.50	0.50	0	0	0	1	1	▇▁▁▁▇

```r
# Q1 b)

set.seed(52156)
dfcTrain <- dfc %>% sample_frac(0.65)
dfcTest <- dplyr::setdiff(dfc, dfcTrain)
```

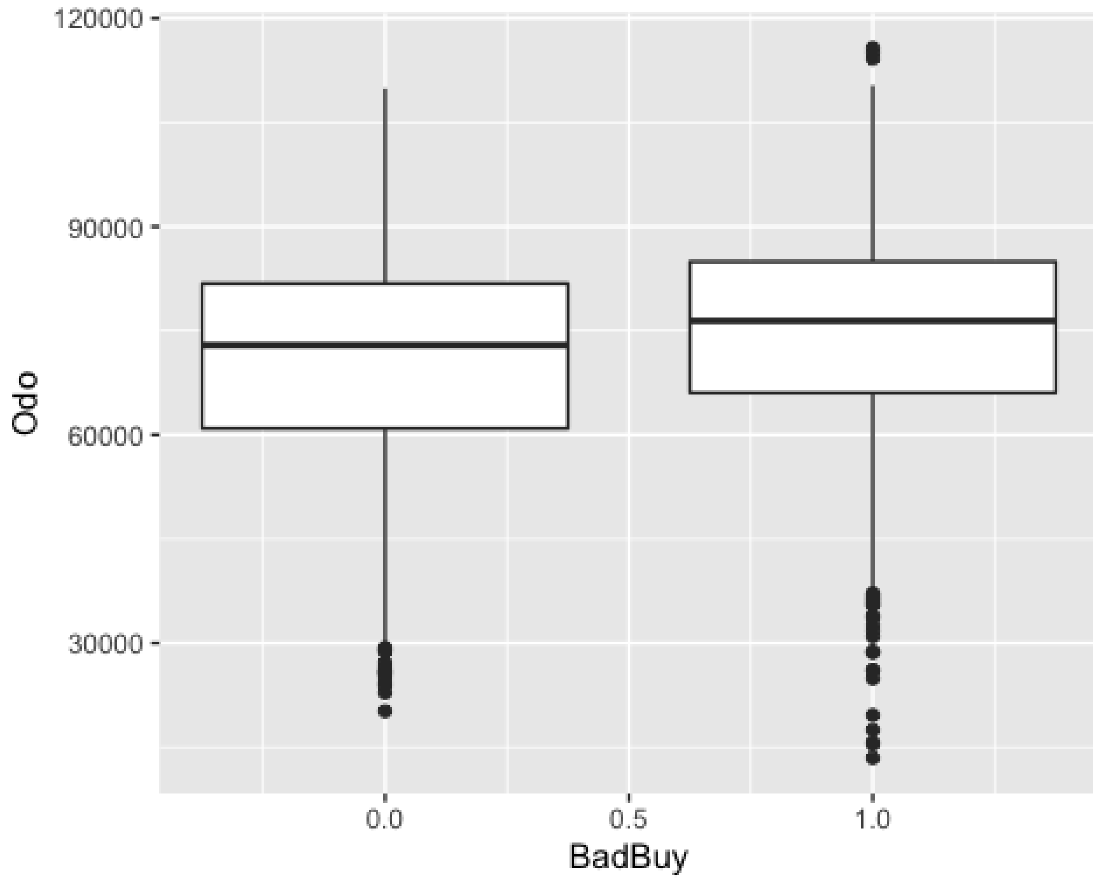
```r
# Q2 a) 1)

box1 <- dfcTrain %>%  ggplot(mapping = aes(x = BadBuy , y = MMRAauction,
group = BadBuy)) + geom_boxplot()
plot(box1)
```




```r
# Q2 a) 2)

box2 <- dfcTrain %>%  ggplot(mapping = aes(x = BadBuy , y = Age, group =
BadBuy)) + geom_boxplot()
plot(box2)
```




```
# Q2 a) 3)

box3 <- dfcTrain %>%  ggplot(mapping = aes(x = BadBuy , y = Odo, group =
BadBuy)) + geom_boxplot()
plot(box3)
```




```
# Q2 b)

 dfcTrain %>%
  group_by(Size) %>%
count(BadBuy) %>%
  mutate(pct = 100*n/sum(n))
```

```
## # A tibble: 24 x 4
## # Groups:   Size [12]
##    Size        BadBuy     n   pct
##    <chr>        <dbl> <int> <dbl>
##  1 COMPACT          0   309  40.8
##  2 COMPACT          1   448  59.2
##  3 CROSSOVER        0    88  57.1
##  4 CROSSOVER        1    66  42.9
##  5 LARGE            0   423  59.8
##  6 LARGE            1   284  40.2
##  7 LARGESUV         0    53  41.1
##  8 LARGESUV         1    76  58.9
##  9 LARGETRUCK       0   156  55.3
## 10 LARGETRUCK       1   126  44.7
## # … with 14 more rows
```

```r
#Q3 a')

dfcLPMTrain <- lm(dfcTrain, formula = BadBuy ~ . )
summary(dfcLPMTrain)

##
## Call:
## lm(formula = BadBuy ~ ., data = dfcTrain)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.2353 -0.3934 -0.1635  0.4658  0.9587
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)         -1.996e-01  2.394e-01  -0.834  0.40434
## AuctionMANHEIM       4.065e-02  1.490e-02   2.728  0.00638 **
## AuctionOTHER         2.287e-02  1.706e-02   1.341  0.18008
## Age                  5.154e-02  5.619e-03   9.172  < 2e-16 ***
## MakeBUICK            2.392e-01  2.360e-01   1.013  0.31089
## MakeCADILLAC         2.664e-01  5.045e-01   0.528  0.59756
## MakeCHEVROLET        1.861e-01  2.299e-01   0.810  0.41820
## MakeCHRYSLER         2.944e-01  2.297e-01   1.282  0.19993
## MakeDODGE            2.384e-01  2.293e-01   1.040  0.29853
## MakeFORD             2.620e-01  2.298e-01   1.140  0.25427
## MakeGMC              1.398e-01  2.379e-01   0.588  0.55685
## MakeHONDA            1.114e-01  2.374e-01   0.469  0.63904
## MakeHYUNDAI          2.099e-01  2.321e-01   0.904  0.36578
## MakeINFINITI         3.671e-01  3.201e-01   1.147  0.25141
## MakeISUZU            1.764e-01  2.747e-01   0.642  0.52082
## MakeJEEP             2.537e-01  2.331e-01   1.089  0.27638
## MakeKIA              2.190e-01  2.316e-01   0.946  0.34440
## MakeLEXUS            8.805e-01  3.221e-01   2.733  0.00629 **
## MakeLINCOLN          3.712e-01  2.577e-01   1.440  0.14980
## MakeMAZDA            2.567e-01  2.329e-01   1.102  0.27036
## MakeMERCURY          2.980e-01  2.337e-01   1.275  0.20229
## MakeMINI             3.301e-01  3.082e-01   1.071  0.28422
## MakeMITSUBISHI       1.179e-01  2.338e-01   0.504  0.61396
## MakeNISSAN           2.310e-01  2.313e-01   0.999  0.31801
## MakeOLDSMOBILE       3.261e-01  2.441e-01   1.336  0.18156
## MakePONTIAC          2.181e-01  2.306e-01   0.946  0.34427
## MakeSATURN           2.800e-01  2.316e-01   1.209  0.22684
## MakeSCION            1.091e-01  2.669e-01   0.409  0.68272
## MakeSUBARU           2.432e-01  3.922e-01   0.620  0.53520
## MakeSUZUKI           3.696e-01  2.335e-01   1.583  0.11354
## MakeTOYOTA           1.638e-01  2.341e-01   0.700  0.48414
## MakeVOLKSWAGEN       2.630e-01  2.613e-01   1.007  0.31409
## MakeVOLVO           -1.809e-01  3.906e-01  -0.463  0.64322
## ColorBLACK           2.220e-02  4.160e-02   0.534  0.59365
## ColorBLUE            1.890e-02  4.055e-02   0.466  0.64111
```

```
## ColorBROWN        1.819e-02  7.917e-02   0.230  0.81826
## ColorGOLD         5.438e-02  4.271e-02   1.273  0.20298
## ColorGREEN        2.264e-02  4.620e-02   0.490  0.62408
## ColorGREY         3.804e-02  4.137e-02   0.919  0.35793
## ColorMAROON       7.248e-02  5.097e-02   1.422  0.15503
## ColorNOTAVAIL    -4.753e-02  1.265e-01  -0.376  0.70717
## ColorNULL        -1.179e-01  4.546e-01  -0.259  0.79543
## ColorORANGE       4.598e-02  8.977e-02   0.512  0.60852
## ColorOTHER       -1.388e-01  9.958e-02  -1.394  0.16327
## ColorPURPLE       1.955e-02  8.259e-02   0.237  0.81289
## ColorRED          6.169e-02  4.214e-02   1.464  0.14326
## ColorSILVER       4.814e-02  3.960e-02   1.216  0.22418
## ColorWHITE        6.047e-02  4.013e-02   1.507  0.13186
## ColorYELLOW      -6.072e-02  1.016e-01  -0.597  0.55031
## WheelTypeCovers  -3.534e-02  1.395e-02  -2.533  0.01134 *
## WheelTypeNULL     5.096e-01  1.861e-02  27.379  < 2e-16 ***
## WheelTypeSpecial -8.805e-03  5.743e-02  -0.153  0.87815
## Odo               2.888e-06  4.327e-07   6.675 2.69e-11 ***
## SizeCROSSOVER    -1.783e-01  4.404e-02  -4.048 5.23e-05 ***
## SizeLARGE        -1.475e-01  2.616e-02  -5.640 1.77e-08 ***
## SizeLARGESUV     -1.379e-01  4.893e-02  -2.817  0.00486 **
## SizeLARGETRUCK   -1.916e-01  3.669e-02  -5.224 1.81e-07 ***
## SizeMEDIUM       -9.926e-02  2.020e-02  -4.913 9.18e-07 ***
## SizeMEDIUMSUV    -9.874e-02  2.840e-02  -3.477  0.00051 ***
## SizeSMALLSUV     -1.333e-01  4.231e-02  -3.149  0.00164 **
## SizeSMALLTRUCK   -1.449e-01  5.170e-02  -2.803  0.00508 **
## SizeSPECIALTY    -7.220e-02  4.718e-02  -1.530  0.12599
## SizeSPORTS       -1.081e-01  5.064e-02  -2.135  0.03277 *
## SizeVAN          -1.136e-01  2.727e-02  -4.164 3.16e-05 ***
## MMRAauction       1.595e-06  7.264e-06   0.220  0.82626
## MMRAretail       -1.126e-06  4.514e-06  -0.249  0.80302
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.4502 on 6474 degrees of freedom
## Multiple R-squared:  0.1975, Adjusted R-squared:  0.1894
## F-statistic: 24.51 on 65 and 6474 DF,  p-value: < 2.2e-16

dfcLPMresultsTrain <- lm(dfcTrain, formula = BadBuy ~ . ) %>%
  predict(.,dfcTrain) %>%
  bind_cols(dfcTrain, predictedProb =.)

#dfcLPMresultsTrain

dfcLPMresultsTest <- lm(dfcTrain, formula = BadBuy ~ . ) %>%
  predict(.,dfcTest) %>%
  bind_cols(dfcTest, predictedProb =.)

#dfcLPMresultsTest
```

```r
performance <- metric_set(rmse, mae)
performance
```

```
## function (data, truth, estimate, na_rm = TRUE, ...)
## {
##     call_args <- quos(data = data, truth = !!enquo(truth), estimate =
!!enquo(estimate),
##         na_rm = na_rm, ... = ...)
##     calls <- lapply(fns, call2, !!!call_args)
##     metric_list <- mapply(FUN = eval_safely, calls, names(calls),
##         SIMPLIFY = FALSE, USE.NAMES = FALSE)
##     bind_rows(metric_list)
## }
## <bytecode: 0x7ffa1f84e938>
## <environment: 0x7ffa1f84fee0>
## attr(,"class")
## [1] "numeric_metric_set" "metric_set"         "function"
## attr(,"metrics")
## attr(,"metrics")$rmse
## function (data, ...)
## {
##     UseMethod("rmse")
## }
## <bytecode: 0x7ffa381eb930>
## <environment: namespace:yardstick>
## attr(,"class")
## [1] "numeric_metric" "function"
## attr(,"direction")
## [1] "minimize"
##
## attr(,"metrics")$mae
## function (data, ...)
## {
##     UseMethod("mae")
## }
## <bytecode: 0x7ffa381967b0>
## <environment: namespace:yardstick>
## attr(,"class")
## [1] "numeric_metric" "function"
## attr(,"direction")
## [1] "minimize"
```

```r
performance(data= dfcLPMresultsTrain, truth= BadBuy, estimate= predictedProb)
```

```
## # A tibble: 2 x 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 rmse    standard       0.448
## 2 mae     standard       0.410
```

```r
performance <- metric_set(rmse, mae)
performance
```

```
## function (data, truth, estimate, na_rm = TRUE, ...)
## {
##     call_args <- quos(data = data, truth = !!enquo(truth), estimate = 
!!enquo(estimate),
##         na_rm = na_rm, ... = ...)
##     calls <- lapply(fns, call2, !!!call_args)
##     metric_list <- mapply(FUN = eval_safely, calls, names(calls),
##         SIMPLIFY = FALSE, USE.NAMES = FALSE)
##     bind_rows(metric_list)
## }
## <bytecode: 0x7ffa1f84e938>
## <environment: 0x7ffa1de0e190>
## attr(,"class")
## [1] "numeric_metric_set" "metric_set"         "function"
## attr(,"metrics")
## attr(,"metrics")$rmse
## function (data, ...)
## {
##     UseMethod("rmse")
## }
## <bytecode: 0x7ffa381eb930>
## <environment: namespace:yardstick>
## attr(,"class")
## [1] "numeric_metric" "function"
## attr(,"direction")
## [1] "minimize"
##
## attr(,"metrics")$mae
## function (data, ...)
## {
##     UseMethod("mae")
## }
## <bytecode: 0x7ffa381967b0>
## <environment: namespace:yardstick>
## attr(,"class")
## [1] "numeric_metric" "function"
## attr(,"direction")
## [1] "minimize"
```

```r
performance(data= dfcLPMresultsTest, truth= BadBuy, estimate= predictedProb)
```

```
## # A tibble: 2 x 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 rmse    standard       0.453
## 2 mae     standard       0.415
```

```r
# Q3 c)
dfcLPMresultTest2 <-
    dfcLPMresultsTest %>%
    mutate(predictedClass = as.factor(ifelse(predictedProb > 0.5, 1, 0)))
#dfcLPMresultTest2

#Q3 d)


dfcLPMresultTest2 %>%
  xtabs(~BadBuy+predictedClass, .) %>%
  confusionMatrix(positive='1')

## Confusion Matrix and Statistics
##
##       predictedClass
## BadBuy    0    1
##      0 1374  408
##      1  743  996
##
##                Accuracy : 0.6731
##                  95% CI : (0.6573, 0.6886)
##     No Information Rate : 0.6012
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.3446
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.7094
##             Specificity : 0.6490
##          Pos Pred Value : 0.5727
##          Neg Pred Value : 0.7710
##              Prevalence : 0.3988
##          Detection Rate : 0.2829
##    Detection Prevalence : 0.4939
##       Balanced Accuracy : 0.6792
##
##        'Positive' Class : 1
##

# Q3 e)

compute <- data.frame(Auction="ADESA", Age=1, Make="HONDA",
Color="SILVER",WheelType="Covers", Odo=10000, Size="LARGE", MMRAauction=8000,
MMRAretail=10000)

predict(dfcLPMTrain, compute, type= "response")
```

```
##           1
## -0.1410712

# Q4 a)

colsToFactor <- c('BadBuy')
colsToFactor

## [1] "BadBuy"

dfc <- dfc %>%
  mutate_at(colsToFactor, ~factor(.))
#dfc

dfcTrain <- dfcTrain %>%
  mutate_at(colsToFactor, ~factor(.))
#dfcTrain

dfcTest <- dfcTest %>%
  mutate_at(colsToFactor, ~factor(.))
#dfcTest

logit <-
  glm(BadBuy ~., family = 'binomial', data = dfcTrain)
summary(logit)

##
## Call:
## glm(formula = BadBuy ~ ., family = "binomial", data = dfcTrain)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.0710  -0.9782  -0.4260   1.0916   2.1903
##
## Coefficients:
##                      Estimate Std. Error z value Pr(>|z|)
## (Intercept)        -3.500e+00  1.270e+00  -2.756 0.005856 **
## AuctionMANHEIM      1.793e-01  7.504e-02   2.390 0.016845 *
## AuctionOTHER        1.001e-01  9.045e-02   1.106 0.268565
## Age                 2.668e-01  2.914e-02   9.155  < 2e-16 ***
## MakeBUICK           1.112e+00  1.255e+00   0.885 0.375971
## MakeCADILLAC        1.126e+01  5.354e+02   0.021 0.983227
## MakeCHEVROLET       8.248e-01  1.225e+00   0.673 0.500782
## MakeCHRYSLER        1.349e+00  1.224e+00   1.102 0.270477
## MakeDODGE           1.074e+00  1.222e+00   0.878 0.379780
## MakeFORD            1.200e+00  1.225e+00   0.980 0.327196
## MakeGMC             6.066e-01  1.261e+00   0.481 0.630493
## MakeHONDA           4.812e-01  1.265e+00   0.381 0.703541
## MakeHYUNDAI         9.347e-01  1.236e+00   0.757 0.449327
## MakeINFINITI        1.547e+00  1.736e+00   0.891 0.372912
## MakeISUZU           7.751e-01  1.427e+00   0.543 0.587130
```

```
## MakeJEEP          1.145e+00  1.240e+00   0.924 0.355648
## MakeKIA           9.983e-01  1.233e+00   0.809 0.418260
## MakeLEXUS         1.537e+01  2.430e+02   0.063 0.949544
## MakeLINCOLN       1.802e+00  1.393e+00   1.293 0.195875
## MakeMAZDA         1.141e+00  1.239e+00   0.921 0.357014
## MakeMERCURY       1.374e+00  1.243e+00   1.105 0.269084
## MakeMINI          1.440e+00  1.588e+00   0.907 0.364557
## MakeMITSUBISHI    4.306e-01  1.245e+00   0.346 0.729402
## MakeNISSAN        1.042e+00  1.232e+00   0.846 0.397814
## MakeOLDSMOBILE    1.569e+00  1.305e+00   1.203 0.229068
## MakePONTIAC       9.857e-01  1.228e+00   0.802 0.422313
## MakeSATURN        1.301e+00  1.233e+00   1.055 0.291232
## MakeSCION         4.789e-01  1.405e+00   0.341 0.733220
## MakeSUBARU        1.096e+00  1.882e+00   0.582 0.560402
## MakeSUZUKI        1.761e+00  1.242e+00   1.418 0.156157
## MakeTOYOTA        6.691e-01  1.246e+00   0.537 0.591252
## MakeVOLKSWAGEN    1.152e+00  1.354e+00   0.850 0.395072
## MakeVOLVO        -1.216e+01  3.689e+02  -0.033 0.973702
## ColorBLACK        1.504e-01  2.159e-01   0.697 0.485965
## ColorBLUE         1.243e-01  2.104e-01   0.591 0.554676
## ColorBROWN        1.349e-01  3.893e-01   0.347 0.728926
## ColorGOLD         3.050e-01  2.202e-01   1.385 0.165955
## ColorGREEN        1.703e-01  2.370e-01   0.719 0.472429
## ColorGREY         2.324e-01  2.140e-01   1.086 0.277401
## ColorMAROON       4.099e-01  2.596e-01   1.579 0.114381
## ColorNOTAVAIL    -3.050e-01  7.564e-01  -0.403 0.686743
## ColorNULL         9.528e+00  5.354e+02   0.018 0.985801
## ColorORANGE       2.938e-01  4.657e-01   0.631 0.528038
## ColorOTHER       -1.169e+00  6.442e-01  -1.815 0.069558 .
## ColorPURPLE       1.916e-01  4.249e-01   0.451 0.652092
## ColorRED          3.381e-01  2.178e-01   1.553 0.120491
## ColorSILVER       2.804e-01  2.058e-01   1.363 0.172916
## ColorWHITE        3.385e-01  2.083e-01   1.625 0.104210
## ColorYELLOW      -2.873e-01  5.003e-01  -0.574 0.565802
## WheelTypeCovers  -1.147e-01  6.705e-02  -1.711 0.087051 .
## WheelTypeNULL     3.487e+00  1.727e-01  20.188  < 2e-16 ***
## WheelTypeSpecial -5.611e-02  2.665e-01  -0.211 0.833232
## Odo               1.518e-05  2.192e-06   6.926 4.33e-12 ***
## SizeCROSSOVER    -8.961e-01  2.225e-01  -4.027 5.64e-05 ***
## SizeLARGE        -7.431e-01  1.322e-01  -5.621 1.90e-08 ***
## SizeLARGESUV     -7.374e-01  2.464e-01  -2.993 0.002762 **
## SizeLARGETRUCK   -9.680e-01  1.834e-01  -5.277 1.31e-07 ***
## SizeMEDIUM       -5.145e-01  1.018e-01  -5.053 4.35e-07 ***
## SizeMEDIUMSUV    -5.130e-01  1.432e-01  -3.584 0.000339 ***
## SizeSMALLSUV     -6.720e-01  2.083e-01  -3.226 0.001255 **
## SizeSMALLTRUCK   -7.125e-01  2.521e-01  -2.826 0.004716 **
## SizeSPECIALTY    -3.607e-01  2.316e-01  -1.557 0.119395
## SizeSPORTS       -5.310e-01  2.552e-01  -2.081 0.037438 *
## SizeVAN          -5.845e-01  1.363e-01  -4.287 1.81e-05 ***
## MMRAauction       1.280e-05  3.684e-05   0.348 0.728168
```

```
## MMRAretail       -5.316e-06  2.247e-05  -0.237 0.812969
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 9066.3  on 6539  degrees of freedom
## Residual deviance: 7516.5  on 6474  degrees of freedom
## AIC: 7648.5
##
## Number of Fisher Scoring iterations: 12
```

```
resultsLog <- train(BadBuy ~ ., family=binomial, data= dfcTrain, method=
'glm' )
```

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

    #predict(dfcTest, type = "response") %>%
    #bind_cols(dfcTest, predictedProb=.) %>%
summary(resultsLog)

##
## Call:
## NULL
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.0710  -0.9782  -0.4260   1.0916   2.1903
##
```

```
## Coefficients:
##                     Estimate Std. Error z value Pr(>|z|)    
## (Intercept)       -3.500e+00  1.270e+00  -2.756 0.005856 ** 
## AuctionMANHEIM     1.793e-01  7.504e-02   2.390 0.016845 *  
## AuctionOTHER       1.001e-01  9.045e-02   1.106 0.268565    
## Age                2.668e-01  2.914e-02   9.155  < 2e-16 ***
## MakeBUICK          1.112e+00  1.255e+00   0.885 0.375971    
## MakeCADILLAC       1.126e+01  5.354e+02   0.021 0.983227    
## MakeCHEVROLET      8.248e-01  1.225e+00   0.673 0.500782    
## MakeCHRYSLER       1.349e+00  1.224e+00   1.102 0.270477    
## MakeDODGE          1.074e+00  1.222e+00   0.878 0.379780    
## MakeFORD           1.200e+00  1.225e+00   0.980 0.327196    
## MakeGMC            6.066e-01  1.261e+00   0.481 0.630493    
## MakeHONDA          4.812e-01  1.265e+00   0.381 0.703541    
## MakeHYUNDAI        9.347e-01  1.236e+00   0.757 0.449327    
## MakeINFINITI       1.547e+00  1.736e+00   0.891 0.372912    
## MakeISUZU          7.751e-01  1.427e+00   0.543 0.587130    
## MakeJEEP           1.145e+00  1.240e+00   0.924 0.355648    
## MakeKIA            9.983e-01  1.233e+00   0.809 0.418260    
## MakeLEXUS          1.537e+01  2.430e+02   0.063 0.949544    
## MakeLINCOLN        1.802e+00  1.393e+00   1.293 0.195875    
## MakeMAZDA          1.141e+00  1.239e+00   0.921 0.357014    
## MakeMERCURY        1.374e+00  1.243e+00   1.105 0.269084    
## MakeMINI           1.440e+00  1.588e+00   0.907 0.364557    
## MakeMITSUBISHI     4.306e-01  1.245e+00   0.346 0.729402    
## MakeNISSAN         1.042e+00  1.232e+00   0.846 0.397814    
## MakeOLDSMOBILE     1.569e+00  1.305e+00   1.203 0.229068    
## MakePONTIAC        9.857e-01  1.228e+00   0.802 0.422313    
## MakeSATURN         1.301e+00  1.233e+00   1.055 0.291232    
## MakeSCION          4.789e-01  1.405e+00   0.341 0.733220    
## MakeSUBARU         1.096e+00  1.882e+00   0.582 0.560402    
## MakeSUZUKI         1.761e+00  1.242e+00   1.418 0.156157    
## MakeTOYOTA         6.691e-01  1.246e+00   0.537 0.591252    
## MakeVOLKSWAGEN     1.152e+00  1.354e+00   0.850 0.395072    
## MakeVOLVO         -1.216e+01  3.689e+02  -0.033 0.973702    
## ColorBLACK         1.504e-01  2.159e-01   0.697 0.485965    
## ColorBLUE          1.243e-01  2.104e-01   0.591 0.554676    
## ColorBROWN         1.349e-01  3.893e-01   0.347 0.728926    
## ColorGOLD          3.050e-01  2.202e-01   1.385 0.165955    
## ColorGREEN         1.703e-01  2.370e-01   0.719 0.472429    
## ColorGREY          2.324e-01  2.140e-01   1.086 0.277401    
## ColorMAROON        4.099e-01  2.596e-01   1.579 0.114381    
## ColorNOTAVAIL     -3.050e-01  7.564e-01  -0.403 0.686743    
## ColorNULL          9.528e+00  5.354e+02   0.018 0.985801    
## ColorORANGE        2.938e-01  4.657e-01   0.631 0.528038    
## ColorOTHER        -1.169e+00  6.442e-01  -1.815 0.069558 .  
## ColorPURPLE        1.916e-01  4.249e-01   0.451 0.652092    
## ColorRED           3.381e-01  2.178e-01   1.553 0.120491    
## ColorSILVER        2.804e-01  2.058e-01   1.363 0.172916    
## ColorWHITE         3.385e-01  2.083e-01   1.625 0.104210    
```

```
## ColorYELLOW      -2.873e-01  5.003e-01  -0.574 0.565802    
## WheelTypeCovers  -1.147e-01  6.705e-02  -1.711 0.087051 .  
## WheelTypeNULL     3.487e+00  1.727e-01  20.188  < 2e-16 ***
## WheelTypeSpecial -5.611e-02  2.665e-01  -0.211 0.833232    
## Odo               1.518e-05  2.192e-06   6.926 4.33e-12 ***
## SizeCROSSOVER    -8.961e-01  2.225e-01  -4.027 5.64e-05 ***
## SizeLARGE        -7.431e-01  1.322e-01  -5.621 1.90e-08 ***
## SizeLARGESUV     -7.374e-01  2.464e-01  -2.993 0.002762 ** 
## SizeLARGETRUCK   -9.680e-01  1.834e-01  -5.277 1.31e-07 ***
## SizeMEDIUM       -5.145e-01  1.018e-01  -5.053 4.35e-07 ***
## SizeMEDIUMSUV    -5.130e-01  1.432e-01  -3.584 0.000339 ***
## SizeSMALLSUV     -6.720e-01  2.083e-01  -3.226 0.001255 ** 
## SizeSMALLTRUCK   -7.125e-01  2.521e-01  -2.826 0.004716 ** 
## SizeSPECIALTY    -3.607e-01  2.316e-01  -1.557 0.119395    
## SizeSPORTS       -5.310e-01  2.552e-01  -2.081 0.037438 *  
## SizeVAN          -5.845e-01  1.363e-01  -4.287 1.81e-05 ***
## MMRAauction       1.280e-05  3.684e-05   0.348 0.728168    
## MMRAretail       -5.316e-06  2.247e-05  -0.237 0.812969    
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## (Dispersion parameter for binomial family taken to be 1)
## 
##     Null deviance: 9066.3  on 6539  degrees of freedom
## Residual deviance: 7516.5  on 6474  degrees of freedom
## AIC: 7648.5
## 
## Number of Fisher Scoring iterations: 12
```

```
library("plyr")
```

```
## ---------------------------------------------------------------------------
----

## You have loaded plyr after dplyr - this is likely to cause problems.
## If you need functions from both plyr and dplyr, please load plyr first,
then dplyr:
## library(plyr); library(dplyr)

## ---------------------------------------------------------------------------
----

## 
## Attaching package: 'plyr'

## The following objects are masked from 'package:plotly':
## 
##     arrange, mutate, rename, summarise

## The following objects are masked from 'package:dplyr':
## 
```

```
##     arrange, count, desc, failwith, id, mutate, rename, summarise,
##     summarize

## The following object is masked from 'package:purrr':
##
##     compact

dfc$Color <- revalue(dfc$Color, c("NULL"="NULL", "NOTAVAIL"="NULL"))

dfc$Make <- revalue(dfc$Make, c("ACURA"="OTHER",
"CADILLAC"="OTHER","LEXUS"="OTHER","MINI"="OTHER","SUBARU"="OTHER","VOLVO"="O
THER"))

set.seed(52156)
dfcTrain2 <- dfc %>% sample_frac(0.65)
dfcTest2 <- dplyr::setdiff(dfc, dfcTrain2)

# Q4 d)

resultsLog2 <- train(BadBuy ~ ., family=binomial, data= dfcTrain2, method=
'glm' )

    #predict(dfcTest, type = "response") %>%
    #bind_cols(dfcTest, predictedProb=.) %>%
summary(resultsLog2)

##
## Call:
## NULL
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.0725  -0.9782  -0.4717   1.0946   2.1705
##
## Coefficients:
##                     Estimate Std. Error z value Pr(>|z|)
## (Intercept)       -2.472e+00  4.513e-01  -5.478 4.30e-08 ***
## AuctionMANHEIM     1.735e-01  7.493e-02   2.316 0.020579 *
## AuctionOTHER       9.519e-02  9.037e-02   1.053 0.292217
## Age                2.785e-01  2.887e-02   9.647  < 2e-16 ***
## MakeCHEVROLET     -2.774e-01  2.895e-01  -0.958 0.337982
## MakeCHRYSLER       2.527e-01  3.011e-01   0.839 0.401419
## MakeDODGE         -2.483e-02  2.966e-01  -0.084 0.933287
## MakeFORD           1.020e-01  2.945e-01   0.346 0.729155
## MakeGMC           -5.054e-01  4.193e-01  -1.205 0.228054
## MakeHONDA         -6.530e-01  4.317e-01  -1.512 0.130433
## MakeHYUNDAI       -1.623e-01  3.381e-01  -0.480 0.631275
## MakeINFINITI       3.727e-01  1.280e+00   0.291 0.771007
## MakeISUZU         -3.227e-01  7.887e-01  -0.409 0.682408
## MakeJEEP           3.121e-02  3.496e-01   0.089 0.928850
## MakeKIA           -9.342e-02  3.281e-01  -0.285 0.775823
```

```
## MakeLINCOLN       6.866e-01  7.410e-01   0.927 0.354146    
## MakeMAZDA         3.015e-02  3.530e-01   0.085 0.931925    
## MakeMERCURY       2.670e-01  3.632e-01   0.735 0.462313    
## MakeMITSUBISHI   -6.722e-01  3.692e-01  -1.821 0.068664 .  
## MakeNISSAN       -7.824e-02  3.213e-01  -0.243 0.807645    
## MakeOLDSMOBILE    4.725e-01  5.397e-01   0.875 0.381344    
## MakeOTHER         3.109e-01  6.256e-01   0.497 0.619240    
## MakePONTIAC      -1.156e-01  3.039e-01  -0.380 0.703748    
## MakeSATURN        2.040e-01  3.293e-01   0.620 0.535513    
## MakeSCION        -6.429e-01  7.485e-01  -0.859 0.390426    
## MakeSUZUKI        6.756e-01  3.578e-01   1.888 0.058974 .  
## MakeTOYOTA       -4.609e-01  3.718e-01  -1.240 0.215081    
## MakeVOLKSWAGEN    3.278e-02  6.815e-01   0.048 0.961638    
## ColorBLACK        1.502e-01  2.157e-01   0.696 0.486312    
## ColorBLUE         1.197e-01  2.103e-01   0.569 0.569124    
## ColorBROWN        1.348e-01  3.891e-01   0.346 0.729074    
## ColorGOLD         3.066e-01  2.201e-01   1.393 0.163652    
## ColorGREEN        1.723e-01  2.369e-01   0.727 0.466976    
## ColorGREY         2.307e-01  2.139e-01   1.078 0.280903    
## ColorMAROON       4.114e-01  2.596e-01   1.585 0.112963    
## ColorNULL        -2.898e-01  7.521e-01  -0.385 0.700011    
## ColorORANGE       2.922e-01  4.655e-01   0.628 0.530251    
## ColorOTHER       -1.168e+00  6.442e-01  -1.812 0.069933 .  
## ColorPURPLE       1.899e-01  4.250e-01   0.447 0.655029    
## ColorRED          3.374e-01  2.177e-01   1.550 0.121257    
## ColorSILVER       2.850e-01  2.057e-01   1.386 0.165860    
## ColorWHITE        3.409e-01  2.083e-01   1.636 0.101745    
## ColorYELLOW      -2.904e-01  4.947e-01  -0.587 0.557141    
## WheelTypeCovers  -1.082e-01  6.698e-02  -1.615 0.106304    
## WheelTypeNULL     3.489e+00  1.727e-01  20.202  < 2e-16 ***
## WheelTypeSpecial -5.363e-02  2.663e-01  -0.201 0.840390    
## Odo               1.484e-05  2.184e-06   6.796 1.08e-11 ***
## SizeCROSSOVER    -9.331e-01  2.220e-01  -4.203 2.63e-05 ***
## SizeLARGE        -7.613e-01  1.319e-01  -5.770 7.91e-09 ***
## SizeLARGESUV     -7.972e-01  2.454e-01  -3.249 0.001157 ** 
## SizeLARGETRUCK   -1.013e+00  1.827e-01  -5.547 2.90e-08 ***
## SizeMEDIUM       -5.260e-01  1.015e-01  -5.181 2.21e-07 ***
## SizeMEDIUMSUV    -5.453e-01  1.425e-01  -3.826 0.000130 ***
## SizeSMALLSUV     -6.989e-01  2.079e-01  -3.361 0.000776 ***
## SizeSMALLTRUCK   -7.329e-01  2.520e-01  -2.908 0.003632 ** 
## SizeSPECIALTY    -4.271e-01  2.274e-01  -1.878 0.060352 .  
## SizeSPORTS       -5.701e-01  2.545e-01  -2.240 0.025066 *  
## SizeVAN          -5.982e-01  1.362e-01  -4.394 1.11e-05 ***
## MMRAauction       2.895e-05  3.634e-05   0.797 0.425670    
## MMRAretail       -8.784e-06  2.241e-05  -0.392 0.695044    
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## (Dispersion parameter for binomial family taken to be 1)
## 
```

```
##     Null deviance: 9066.3  on 6539  degrees of freedom
## Residual deviance: 7528.1  on 6480  degrees of freedom
## AIC: 7648.1
##
## Number of Fisher Scoring iterations: 5
```

```r
levels(as.factor(dfc$Color))
```

```
##  [1] "BEIGE"  "BLACK"  "BLUE"   "BROWN"  "GOLD"   "GREEN"  "GREY"
"MAROON"
##  [9] "NULL"   "ORANGE" "OTHER"  "PURPLE" "RED"    "SILVER" "WHITE"
"YELLOW"
```

```r
# Q4 d)

resultsLog2Caret<-  resultsLog2 %>%
  predict(., dfcTest2) %>%
  bind_cols(dfcTest2, predictedProb = .)
#resultsLog2Caret

resultsLog2Caret %>%
  xtabs(~BadBuy+predictedProb, .) %>%
  confusionMatrix(positive='1')
```

```
## Confusion Matrix and Statistics
##
##       predictedProb
## BadBuy    0    1
##      0 1341  441
##      1  721 1018
##
##                Accuracy : 0.67
##                  95% CI : (0.6542, 0.6855)
##     No Information Rate : 0.5856
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.3386
##
##  Mcnemar's Test P-Value : 2.731e-16
##
##             Sensitivity : 0.6977
##             Specificity : 0.6503
##          Pos Pred Value : 0.5854
##          Neg Pred Value : 0.7525
##              Prevalence : 0.4144
##          Detection Rate : 0.2891
##    Detection Prevalence : 0.4939
##       Balanced Accuracy : 0.6740
##
##        'Positive' Class : 1
##
```

```r
# Q4 e)


compute <- data.frame(Auction="ADESA", Age=1, Make="HONDA",
Color="SILVER",WheelType="Covers", Odo=10000, Size="LARGE", MMRAauction=8000,
MMRAretail=10000)

predict(resultsLog2, compute)

## [1] 0
## Levels: 0 1

# Q5 a)

set.seed(123)

dfcLda  <-
    train(BadBuy ~ ., data= dfcTrain2, method=
'lda',trControl=trainControl(method='cv', number=10))
summary(dfcLda)

##              Length Class      Mode
## prior          2    -none-     numeric
## counts         2    -none-     numeric
## means        118    -none-     numeric
## scaling       59    -none-     numeric
## lev            2    -none-     character
## svd            1    -none-     numeric
## N              1    -none-     numeric
## call           3    -none-     call
## xNames        59    -none-     character
## problemType    1    -none-     character
## tuneValue      1    data.frame list
## obsLevels      2    -none-     character
## param          0    -none-     list

resultsLda <-
    dfcLda %>%
    predict(dfcTest2, type= 'raw') %>%
    bind_cols(dfcTest2, predictedClass=.)
#resultsLda

resultsLda %>%
  xtabs(~BadBuy+predictedClass, .) %>%
  confusionMatrix(positive = '1')

## Confusion Matrix and Statistics
##
##         predictedClass
## BadBuy    0    1
##      0 1377  405
```

```
##      1  749  990
##
##                Accuracy : 0.6723
##                  95% CI : (0.6565, 0.6878)
##     No Information Rate : 0.6038
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.3428
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.7097
##             Specificity : 0.6477
##          Pos Pred Value : 0.5693
##          Neg Pred Value : 0.7727
##              Prevalence : 0.3962
##          Detection Rate : 0.2812
##    Detection Prevalence : 0.4939
##       Balanced Accuracy : 0.6787
##
##        'Positive' Class : 1
##
```

```r
#Q5 b)

set.seed(123)

dfcknn  <-
    train(BadBuy ~ ., data= dfcTrain2, method= 'knn',
trControl=trainControl(method='cv', number=10), tuneLength=24,
preProcess=c("center","scale"))
summary(dfcknn)
```

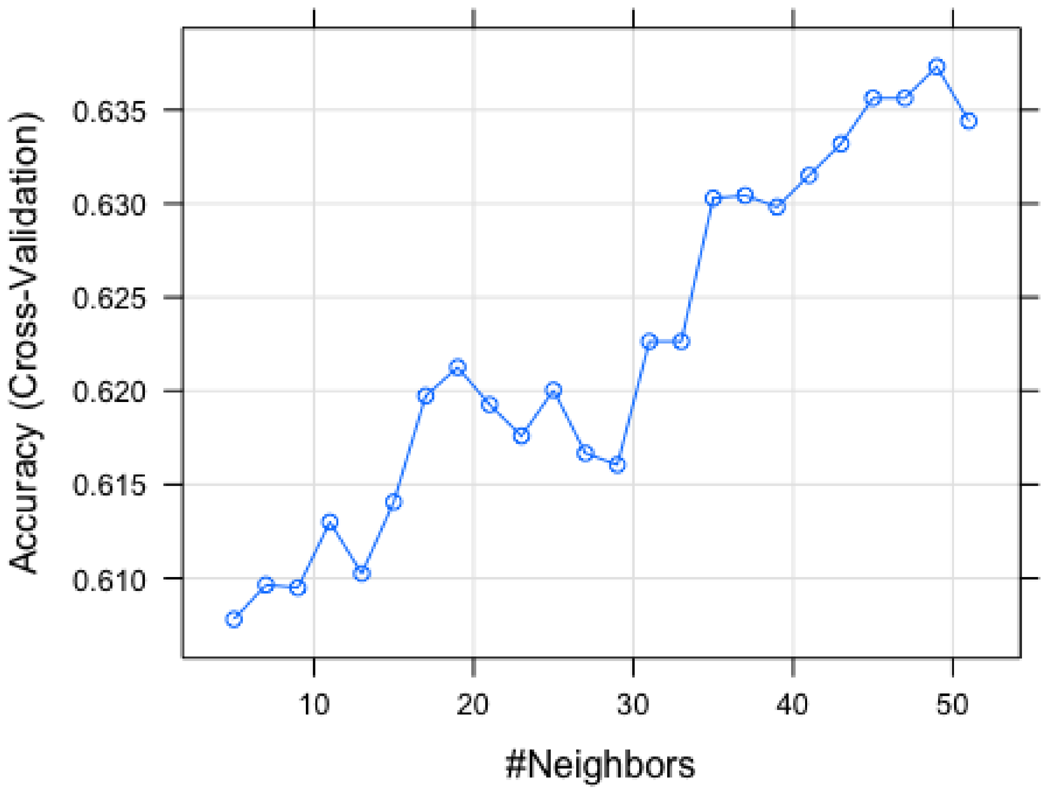
```
##             Length Class      Mode
## learn        2     -none-     list
## k            1     -none-     numeric
## theDots      0     -none-     list
## xNames      59     -none-     character
## problemType  1     -none-     character
## tuneValue    1     data.frame list
## obsLevels    2     -none-     character
## param        0     -none-     list
```

```r
resultsknn <-
    dfcknn %>%
    predict(dfcTest2, type= 'raw') %>%
    bind_cols(dfcTest2, predictedClass=.)
#resultsknn
```

```r
resultsknn %>%
  xtabs(~BadBuy+predictedClass, .) %>%
  confusionMatrix(positive = '1')
```

```
## Confusion Matrix and Statistics
##
##       predictedClass
## BadBuy    0    1
##      0 1349  433
##      1  820  919
##
##                Accuracy : 0.6441
##                  95% CI : (0.6281, 0.66)
##     No Information Rate : 0.616
##     P-Value [Acc > NIR] : 0.0003035
##
##                   Kappa : 0.2862
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.6797
##             Specificity : 0.6219
##          Pos Pred Value : 0.5285
##          Neg Pred Value : 0.7570
##              Prevalence : 0.3840
##          Detection Rate : 0.2610
##    Detection Prevalence : 0.4939
##       Balanced Accuracy : 0.6508
##
##        'Positive' Class : 1
##
```

```r
plot(dfcknn)
```




```
dfcknn$bestTune

##     k
## 23 49

#Q5 c)
lambdaValues <- 10^seq(-5, 2, length = 100)
set.seed(123)

fitLasso <- train(BadBuy ~ ., family='binomial', data=dfcTrain2,
method='glmnet', trControl=trainControl(method='cv', number=10), tuneGrid =
expand.grid(alpha=1, lambda=lambdaValues))

summary(fitLasso)

##             Length Class      Mode
## a0            75   -none-     numeric
## beta        4425   dgCMatrix  S4
## df            75   -none-     numeric
## dim            2   -none-     numeric
## lambda        75   -none-     numeric
## dev.ratio     75   -none-     numeric
## nulldev        1   -none-     numeric
## npasses        1   -none-     numeric
```

```
## jerr           1   -none-     numeric
## offset         1   -none-     logical
## classnames     2   -none-     character
## call           5   -none-     call
## nobs           1   -none-     numeric
## lambdaOpt      1   -none-     numeric
## xNames        59   -none-     character
## problemType    1   -none-     character
## tuneValue      2   data.frame list
## obsLevels      2   -none-     character
## param          1   -none-     list
```

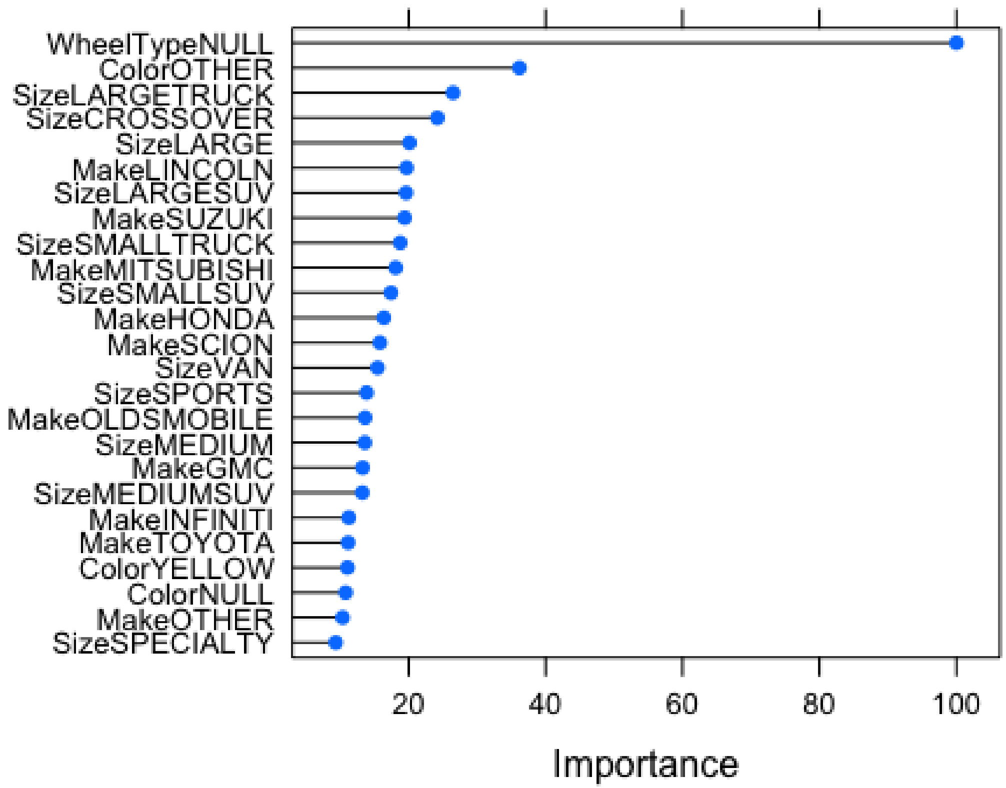
```
resultsLasso <-
  fitLasso %>%
    predict(dfcTest2, type= 'raw') %>%
    bind_cols(dfcTest2, predictedClass=.)
#resultsLasso

resultsLasso %>%
  xtabs(~BadBuy+predictedClass, .) %>%
  confusionMatrix(positive = '1')
```

```
## Confusion Matrix and Statistics
##
##       predictedClass
## BadBuy    0    1
##      0 1339  443
##      1  721 1018
##
##                Accuracy : 0.6694
##                  95% CI : (0.6536, 0.6849)
##     No Information Rate : 0.5851
##     P-Value [Acc > NIR] : < 2e-16
##
##                   Kappa : 0.3374
##
##  Mcnemar's Test P-Value : 4.7e-16
##
##             Sensitivity : 0.6968
##             Specificity : 0.6500
##          Pos Pred Value : 0.5854
##          Neg Pred Value : 0.7514
##              Prevalence : 0.4149
##          Detection Rate : 0.2891
##    Detection Prevalence : 0.4939
##       Balanced Accuracy : 0.6734
##
##        'Positive' Class : 1
##
```

```r
varImp(fitLasso)$importance %>%    # Add scale=FALSE inside VarImp if you don't want to scale
  rownames_to_column(var = "Variable") %>%
  mutate(Importance = scales::percent(Overall/100)) %>%
  arrange(desc(Overall)) %>%
  as_tibble()
```

```
## # A tibble: 59 x 3
##    Variable       Overall Importance
##    <chr>            <dbl> <chr>
##  1 WheelTypeNULL    100   100%
##  2 ColorOTHER        36.2 36%
##  3 SizeLARGETRUCK    26.5 26%
##  4 SizeCROSSOVER     24.2 24%
##  5 SizeLARGE         20.1 20%
##  6 MakeLINCOLN       19.7 20%
##  7 SizeLARGESUV      19.6 20%
##  8 MakeSUZUKI        19.4 19%
##  9 SizeSMALLTRUCK    18.8 19%
## 10 MakeMITSUBISHI    18.1 18%
## # … with 49 more rows
```

```r
#Variable importance plot with the most important variables
plot(varImp(fitLasso),top=25)    # Add top = XX to change the number of visible variables
```




```
#Optimum lambda selected by the algorithm
fitLasso$bestTune$lambda

## [1] 0.0003053856

#Q5 d i)

lambdaValues <- 10^seq(-5, 2, length = 100)
set.seed(123)

fitRidge <- train(BadBuy ~ ., family='binomial', data=dfcTrain2,
method='glmnet', trControl=trainControl(method='cv', number=10), tuneGrid =
expand.grid(alpha=0, lambda=lambdaValues))

summary(fitRidge)

##             Length Class      Mode
## a0           100   -none-     numeric
## beta        5900   dgCMatrix  S4
## df           100   -none-     numeric
## dim            2   -none-     numeric
## lambda       100   -none-     numeric
## dev.ratio    100   -none-     numeric
```

```
## nulldev        1   -none-     numeric
## npasses        1   -none-     numeric
## jerr           1   -none-     numeric
## offset         1   -none-     logical
## classnames     2   -none-     character
## call           5   -none-     call
## nobs           1   -none-     numeric
## lambdaOpt      1   -none-     numeric
## xNames        59   -none-     character
## problemType    1   -none-     character
## tuneValue      2   data.frame list
## obsLevels      2   -none-     character
## param          1   -none-     list
```

```
resultsRidge <-
  fitRidge %>%
    predict(dfcTest2, type= 'raw') %>%
    bind_cols(dfcTest2, predictedClass=.)
#resultsRidge
```

```
resultsRidge %>%
  xtabs(~BadBuy+predictedClass, .) %>%
  confusionMatrix(positive = '1')
```

```
## Confusion Matrix and Statistics
##
##       predictedClass
## BadBuy    0    1
##      0 1323  459
##      1  699 1040
##
##                Accuracy : 0.6711
##                  95% CI : (0.6553, 0.6866)
##     No Information Rate : 0.5743
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.341
##
##  Mcnemar's Test P-Value : 2.166e-12
##
##             Sensitivity : 0.6938
##             Specificity : 0.6543
##          Pos Pred Value : 0.5980
##          Neg Pred Value : 0.7424
##              Prevalence : 0.4257
##          Detection Rate : 0.2954
##    Detection Prevalence : 0.4939
##       Balanced Accuracy : 0.6740
##
```

```
##        'Positive' Class : 1
##

# Q5 d ii)

lambdaValues <- 10^seq(-5, 2, length = 100)
set.seed(123)

fitNet <- train(BadBuy ~ ., family='binomial', data=dfcTrain2,
method='glmnet', trControl=trainControl(method='cv', number=10), tuneGrid =
expand.grid(alpha=0.5, lambda=lambdaValues))

summary(fitNet)

##             Length Class      Mode
## a0            76   -none-     numeric
## beta        4484   dgCMatrix  S4
## df            76   -none-     numeric
## dim            2   -none-     numeric
## lambda        76   -none-     numeric
## dev.ratio     76   -none-     numeric
## nulldev        1   -none-     numeric
## npasses        1   -none-     numeric
## jerr           1   -none-     numeric
## offset         1   -none-     logical
## classnames     2   -none-     character
## call           5   -none-     call
## nobs           1   -none-     numeric
## lambdaOpt      1   -none-     numeric
## xNames        59   -none-     character
## problemType    1   -none-     character
## tuneValue      2   data.frame list
## obsLevels      2   -none-     character
## param          1   -none-     list

resultsNet <-
  fitNet %>%
    predict(dfcTest2, type= 'raw') %>%
    bind_cols(dfcTest2, predictedClass=.)
#resultsNet

resultsNet %>%
  xtabs(~BadBuy+predictedClass, .) %>%
  confusionMatrix(positive = '1')

## Confusion Matrix and Statistics
##
##       predictedClass
## BadBuy    0    1
##      0 1339  443
##      1  723 1016
```

```
##
##                Accuracy : 0.6688
##                  95% CI : (0.653, 0.6844)
##     No Information Rate : 0.5856
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.3363
##
##  Mcnemar's Test P-Value : 3.068e-16
##
##             Sensitivity : 0.6964
##             Specificity : 0.6494
##          Pos Pred Value : 0.5842
##          Neg Pred Value : 0.7514
##              Prevalence : 0.4144
##          Detection Rate : 0.2886
##    Detection Prevalence : 0.4939
##       Balanced Accuracy : 0.6729
##
##        'Positive' Class : 1
##
```

```
# Q5 e)
fitQda  <-
    train(BadBuy ~ ., data= dfcTrain2, method=
'qda',trControl=trainControl(method='cv', number=10))
```

```
## Warning: model fit failed for Fold06: parameter=none Error in
qda.default(x, grouping, ...) : rank deficiency in group 0
```

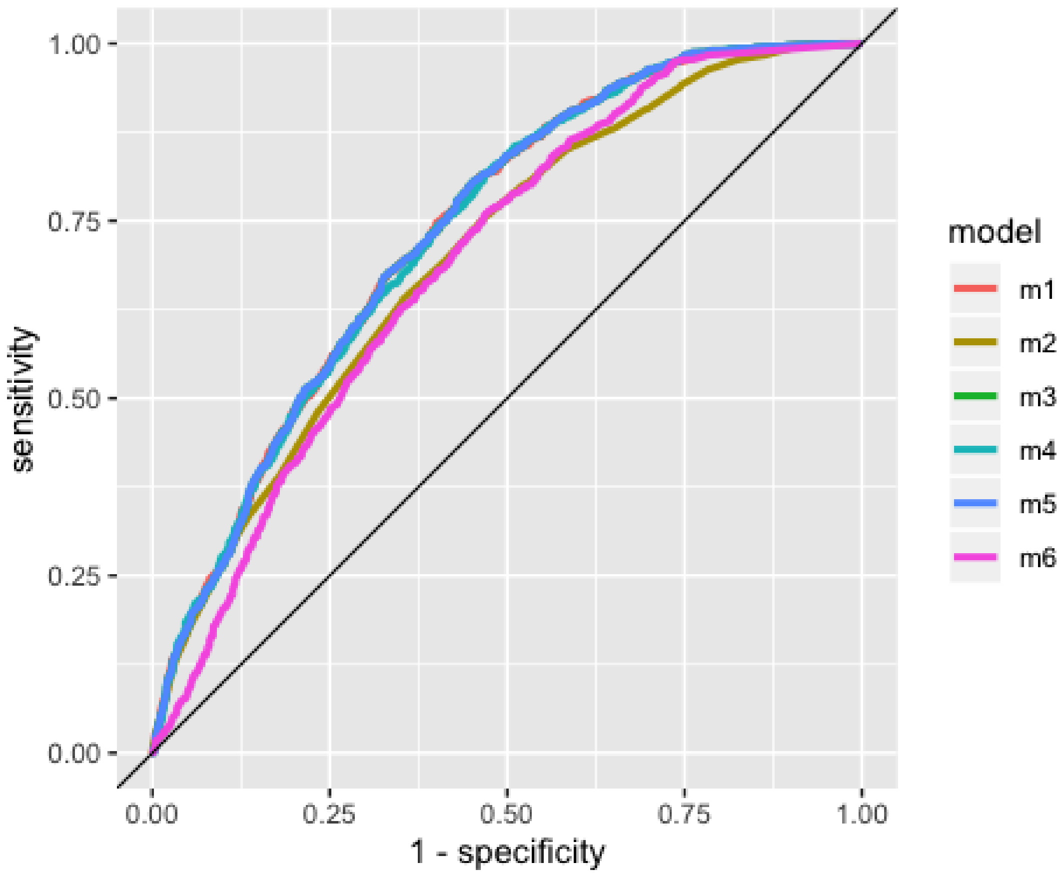
```
## Warning in nominalTrainWorkflow(x = x, y = y, wts = weights, info =
trainInfo, :
## There were missing values in resampled performance measures.
```

```
summary(fitQda)
```

```
##             Length Class      Mode
## prior          2   -none-     numeric
## counts         2   -none-     numeric
## means        118   -none-     numeric
## scaling     6962   -none-     numeric
## ldet           2   -none-     numeric
## lev            2   -none-     character
## N              1   -none-     numeric
## call           3   -none-     call
## xNames        59   -none-     character
## problemType    1   -none-     character
## tuneValue      1   data.frame list
## obsLevels      2   -none-     character
## param          0   -none-     list
```

```r
resultsQda <-
  fitQda %>%
  predict(dfcTest2, type='raw') %>%
  bind_cols(dfcTest2, predictedClass=.)
#resultsQda

resultsQda %>%
  xtabs(~BadBuy+predictedClass, .) %>%
  confusionMatrix(positive = '1')

## Confusion Matrix and Statistics
##
##       predictedClass
## BadBuy    0    1
##      0 1483  299
##      1  973  766
##
##                Accuracy : 0.6387
##                  95% CI : (0.6226, 0.6546)
##     No Information Rate : 0.6975
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.274
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.7192
##             Specificity : 0.6038
##          Pos Pred Value : 0.4405
##          Neg Pred Value : 0.8322
##              Prevalence : 0.3025
##          Detection Rate : 0.2176
##    Detection Prevalence : 0.4939
##       Balanced Accuracy : 0.6615
##
##        'Positive' Class : 1
##

#Q5 f)

resultsLdaProb <- bind_cols(dfcTest2,dfcLda %>%  predict(dfcTest2,
type='prob')) %>% mutate(model="m1")
resultsknnProb <- bind_cols(dfcTest2,dfcknn %>%  predict(dfcTest2,
type='prob')) %>% mutate(model="m2")
resultsLassoProb <- bind_cols(dfcTest2,fitLasso %>%  predict(dfcTest2,
type='prob')) %>% mutate(model="m3")
resultsRidgeProb <- bind_cols(dfcTest2,fitRidge %>%  predict(dfcTest2,
type='prob')) %>% mutate(model="m4")
resultsNetProb <- bind_cols(dfcTest2,fitNet %>%  predict(dfcTest2,
type='prob')) %>% mutate(model="m5")
```

```r
resultsQdaProb <- bind_cols(dfcTest2,fitQda %>%  predict(dfcTest2,
type='prob')) %>% mutate(model="m6")

fitAll <-
bind_rows(resultsLdaProb,resultsknnProb,resultsLassoProb,resultsRidgeProb,res
ultsNetProb,resultsQdaProb)

fitAll %>%
  group_by(model) %>% # group to get individual AUC value for each model
  roc_auc(truth = BadBuy, '1')

## # A tibble: 6 x 4
##   model .metric .estimator .estimate
##   <chr> <chr>   <chr>          <dbl>
## 1 m1    roc_auc binary         0.736
## 2 m2    roc_auc binary         0.700
## 3 m3    roc_auc binary         0.736
## 4 m4    roc_auc binary         0.733
## 5 m5    roc_auc binary         0.736
## 6 m6    roc_auc binary         0.692

fitAll %>%
  group_by(model) %>% # group to get individual ROC curve for each model
  roc_curve(truth = BadBuy, "1") %>% # get values to plot an ROC curve
  ggplot(aes(x = 1 - specificity, y = sensitivity, color = model)) + # plota
ROC curve for each model
  geom_line(size = 1.1) +
  geom_abline(slope = 1, intercept = 0, size = 0.4) +
  coord_fixed()
```




```
# Bonus Question
library("grplasso")

dfTrainGroup <-
  dfcTrain %>%
  mutate(BadBuy = as.numeric(BadBuy)) %>%
  mutate(BadBuy = ifelse(BadBuy == 2, 1, 0))

set.seed(123)

fitGroupLasso <- grplasso(BadBuy ~ ., data=dfTrainGroup, model=LogReg(),
lambda=50)

## Lambda: 50  nr.var: 47

fitGroupLasso$coefficients

##                          50
## (Intercept)      -1.813941e+00
## AuctionMANHEIM    0.000000e+00
## AuctionOTHER      0.000000e+00
## Age               2.268497e-01
## MakeBUICK         8.456070e-02
```

```
## MakeCADILLAC       1.094347e-01
## MakeCHEVROLET      5.835279e-02
## MakeCHRYSLER       9.574693e-02
## MakeDODGE          7.610346e-02
## MakeFORD           9.040031e-02
## MakeGMC            3.456621e-02
## MakeHONDA          3.947392e-02
## MakeHYUNDAI        6.996982e-02
## MakeINFINITI       1.478825e-01
## MakeISUZU          5.094803e-02
## MakeJEEP           8.373619e-02
## MakeKIA            6.639583e-02
## MakeLEXUS          3.860784e-01
## MakeLINCOLN        1.453315e-01
## MakeMAZDA          8.634450e-02
## MakeMERCURY        1.079654e-01
## MakeMINI           1.324372e-01
## MakeMITSUBISHI     3.020255e-02
## MakeNISSAN         8.452620e-02
## MakeOLDSMOBILE     1.235771e-01
## MakePONTIAC        6.886209e-02
## MakeSATURN         9.801410e-02
## MakeSCION          2.764329e-02
## MakeSUBARU         7.977499e-02
## MakeSUZUKI         1.332292e-01
## MakeTOYOTA         5.392668e-02
## MakeVOLKSWAGEN     1.081392e-01
## MakeVOLVO         -9.932604e-02
## ColorBLACK         0.000000e+00
## ColorBLUE          0.000000e+00
## ColorBROWN         0.000000e+00
## ColorGOLD          0.000000e+00
## ColorGREEN         0.000000e+00
## ColorGREY          0.000000e+00
## ColorMAROON        0.000000e+00
## ColorNOTAVAIL      0.000000e+00
## ColorNULL          0.000000e+00
## ColorORANGE        0.000000e+00
## ColorOTHER         0.000000e+00
## ColorPURPLE        0.000000e+00
## ColorRED           0.000000e+00
## ColorSILVER        0.000000e+00
## ColorWHITE         0.000000e+00
## ColorYELLOW        0.000000e+00
## WheelTypeCovers   -1.155588e-01
## WheelTypeNULL      2.715004e+00
## WheelTypeSpecial   1.104936e-02
## Odo                1.014138e-05
## SizeCROSSOVER     -1.962560e-01
## SizeLARGE         -2.384180e-01
```

```
## SizeLARGESUV     -1.589875e-01
## SizeLARGETRUCK   -2.655394e-01
## SizeMEDIUM       -1.223653e-01
## SizeMEDIUMSUV    -1.262012e-01
## SizeSMALLSUV     -1.498841e-01
## SizeSMALLTRUCK   -1.826592e-01
## SizeSPECIALTY    -3.197423e-02
## SizeSPORTS       -1.295130e-01
## SizeVAN          -1.479581e-01
## MMRAauction      -1.844662e-05
## MMRAretail        0.000000e+00
```

```
dfTrainGroup <-
  dfcTrain %>%
  mutate(BadBuy = as.numeric(BadBuy)) %>%
  mutate(BadBuy = ifelse(BadBuy == 2, 1, 0))

set.seed(123)

fitGroupedLasso2 <- grplasso(BadBuy ~ ., data=dfTrainGroup, model=LogReg(),
lambda=100)
```

```
## Lambda: 100  nr.var: 7
```

```
fitGroupedLasso2$coefficients
```

```
##                            100
## (Intercept)      -1.571244e+00
## AuctionMANHEIM    0.000000e+00
## AuctionOTHER      0.000000e+00
## Age               2.103677e-01
## MakeBUICK         0.000000e+00
## MakeCADILLAC      0.000000e+00
## MakeCHEVROLET     0.000000e+00
## MakeCHRYSLER      0.000000e+00
## MakeDODGE         0.000000e+00
## MakeFORD          0.000000e+00
## MakeGMC           0.000000e+00
## MakeHONDA         0.000000e+00
## MakeHYUNDAI       0.000000e+00
## MakeINFINITI      0.000000e+00
## MakeISUZU         0.000000e+00
## MakeJEEP          0.000000e+00
## MakeKIA           0.000000e+00
## MakeLEXUS         0.000000e+00
## MakeLINCOLN       0.000000e+00
## MakeMAZDA         0.000000e+00
## MakeMERCURY       0.000000e+00
## MakeMINI          0.000000e+00
## MakeMITSUBISHI    0.000000e+00
## MakeNISSAN        0.000000e+00
```

```
## MakeOLDSMOBILE     0.000000e+00
## MakePONTIAC        0.000000e+00
## MakeSATURN         0.000000e+00
## MakeSCION          0.000000e+00
## MakeSUBARU         0.000000e+00
## MakeSUZUKI         0.000000e+00
## MakeTOYOTA         0.000000e+00
## MakeVOLKSWAGEN     0.000000e+00
## MakeVOLVO          0.000000e+00
## ColorBLACK         0.000000e+00
## ColorBLUE          0.000000e+00
## ColorBROWN         0.000000e+00
## ColorGOLD          0.000000e+00
## ColorGREEN         0.000000e+00
## ColorGREY          0.000000e+00
## ColorMAROON        0.000000e+00
## ColorNOTAVAIL      0.000000e+00
## ColorNULL          0.000000e+00
## ColorORANGE        0.000000e+00
## ColorOTHER         0.000000e+00
## ColorPURPLE        0.000000e+00
## ColorRED           0.000000e+00
## ColorSILVER        0.000000e+00
## ColorWHITE         0.000000e+00
## ColorYELLOW        0.000000e+00
## WheelTypeCovers   -1.096563e-01
## WheelTypeNULL      2.285604e+00
## WheelTypeSpecial   2.736726e-02
## Odo                7.164414e-06
## SizeCROSSOVER      0.000000e+00
## SizeLARGE          0.000000e+00
## SizeLARGESUV       0.000000e+00
## SizeLARGETRUCK     0.000000e+00
## SizeMEDIUM         0.000000e+00
## SizeMEDIUMSUV      0.000000e+00
## SizeSMALLSUV       0.000000e+00
## SizeSMALLTRUCK     0.000000e+00
## SizeSPECIALTY      0.000000e+00
## SizeSPORTS         0.000000e+00
## SizeVAN            0.000000e+00
## MMRAauction       -1.587228e-05
## MMRAretail         0.000000e+00
```