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.

```
library("tidyverse")
```

```
## Warning: package 'tidyverse' was built under R version 3.6.2
```

```
## -- Attaching packages ---------------------------------------------------
-------------------------------------------------------------------------
---- tidyverse 1.3.0 --
```

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

```
## Warning: package 'ggplot2' was built under R version 3.6.1
```

```
## Warning: package 'tibble' was built under R version 3.6.2
```

```
## Warning: package 'tidyr' was built under R version 3.6.2
```

```
## Warning: package 'readr' was built under R version 3.6.2
```

```
## Warning: package 'purrr' was built under R version 3.6.2
```

```
## Warning: package 'dplyr' was built under R version 3.6.1
```

```
## Warning: package 'forcats' was built under R version 3.6.2
```

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

```
library("tidymodels")
```

```
## Warning: package 'tidymodels' was built under R version 3.6.2
```

```
## -- Attaching packages ---------------------------------------------------
-------------------------------------------------------------------------
--- tidymodels 0.0.3 --

## v broom     0.5.4     v recipes   0.1.9
## v dials     0.0.4     v rsample   0.0.5
## v infer     0.5.1     v yardstick 0.0.4
## v parsnip   0.0.5

## Warning: package 'dials' was built under R version 3.6.2

## Warning: package 'infer' was built under R version 3.6.2

## Warning: package 'parsnip' was built under R version 3.6.2

## Warning: package 'recipes' was built under R version 3.6.2

## Warning: package 'rsample' was built under R version 3.6.2

## Warning: package 'yardstick' was built under R version 3.6.2

## -- Conflicts ------------------------------------------------------------
-------------------------------------------------------------------------
tidymodels_conflicts() --
## x scales::discard() masks purrr::discard()
## x dplyr::filter()   masks stats::filter()
## x recipes::fixed()  masks stringr::fixed()
## x dplyr::lag()      masks stats::lag()
## x dials::margin()   masks ggplot2::margin()
## x yardstick::spec() masks readr::spec()
## x recipes::step()   masks stats::step()

library("plotly")

## Warning: package 'plotly' was built under R version 3.6.2

##
## 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

library("skimr")
```

```
## Warning: package 'skimr' was built under R version 3.6.2
```

```
library("lubridate")
```

```
## Warning: package 'lubridate' was built under R version 3.6.2
```

```
##
## Attaching package: 'lubridate'
```

```
## The following object is masked from 'package:base':
##
##     date
```

```
library("caret")
```

```
## Warning: package 'caret' was built under R version 3.6.2
```

```
## 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
```

```
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()
## )
```

```
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

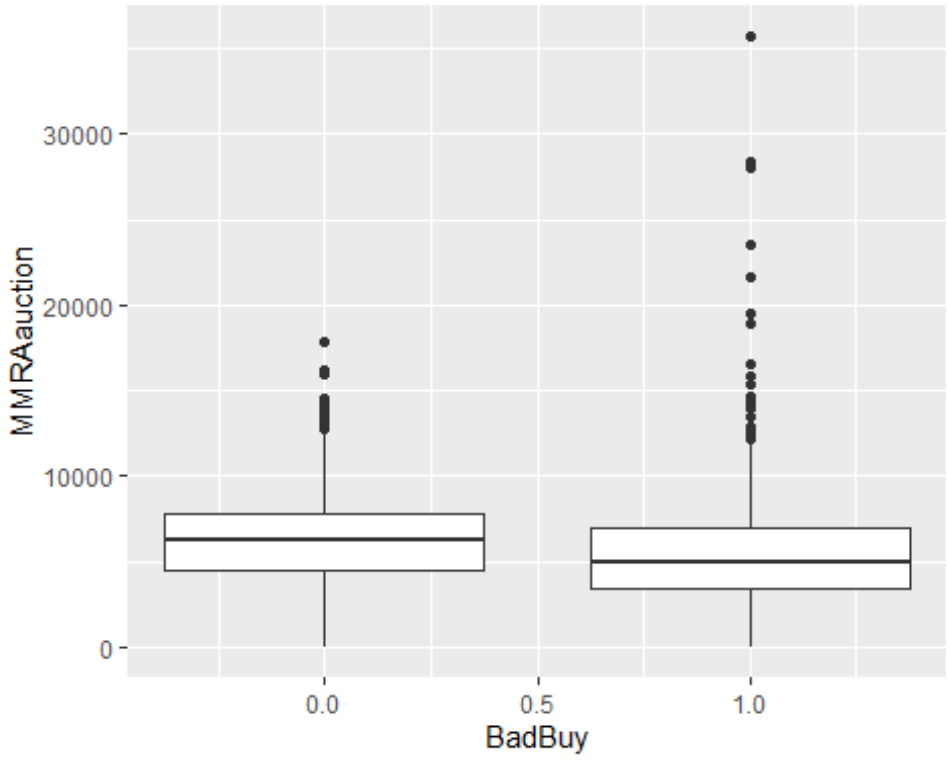
skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
Age	0	1	4.50	1.77	1	3	4	6	9	▃▇▅▂
Odo	0	1	72903.87	14498.87	9446	63488	74942	83663	115717	▁▃▇▂
MMRAauction	0	1	5812.38	2578.85	0	3877	5588	7450	35722	▇▃▁▁
MMRAretail	0	1	8171.51	3257.19	0	5872	8052	10315	39080	▇▇▁▁
BadBuy	0	1	0.50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

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

str(dfcTrain)

## Classes 'spec_tbl_df', 'tbl_df', 'tbl' and 'data.frame': 6540 obs. of  10
variables:
##  $ Auction    : chr  "MANHEIM" "MANHEIM" "MANHEIM" "ADESA" ...
##  $ Age        : num  4 5 2 4 5 4 2 7 6 8 ...
```

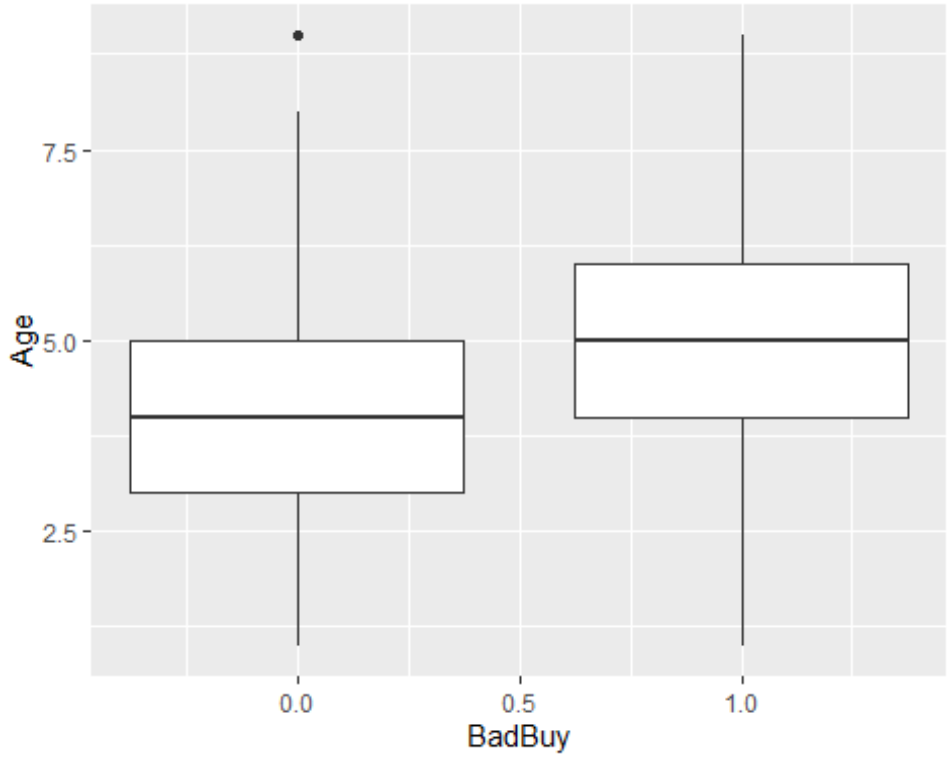
```
##  $ Make        : chr  "FORD" "MINI" "CHEVROLET" "NISSAN" ...
##  $ Color       : chr  "SILVER" "BLUE" "SILVER" "BLUE" ...
##  $ WheelType   : chr  "NULL" "Alloy" "Covers" "NULL" ...
##  $ Odo         : num  77591 80013 75493 84827 57388 ...
##  $ Size        : chr  "LARGETRUCK" "COMPACT" "LARGE" "MEDIUM" ...
##  $ MMRAauction: num  9774 11040 9707 6073 5574 ...
##  $ MMRAretail : num  14506 12423 13975 9791 8984 ...
##  $ BadBuy      : num  1 1 1 1 1 1 0 1 0 0 ...
##  - attr(*, "spec")=
##   .. 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()
##   .. )
```

Question 2) (a)

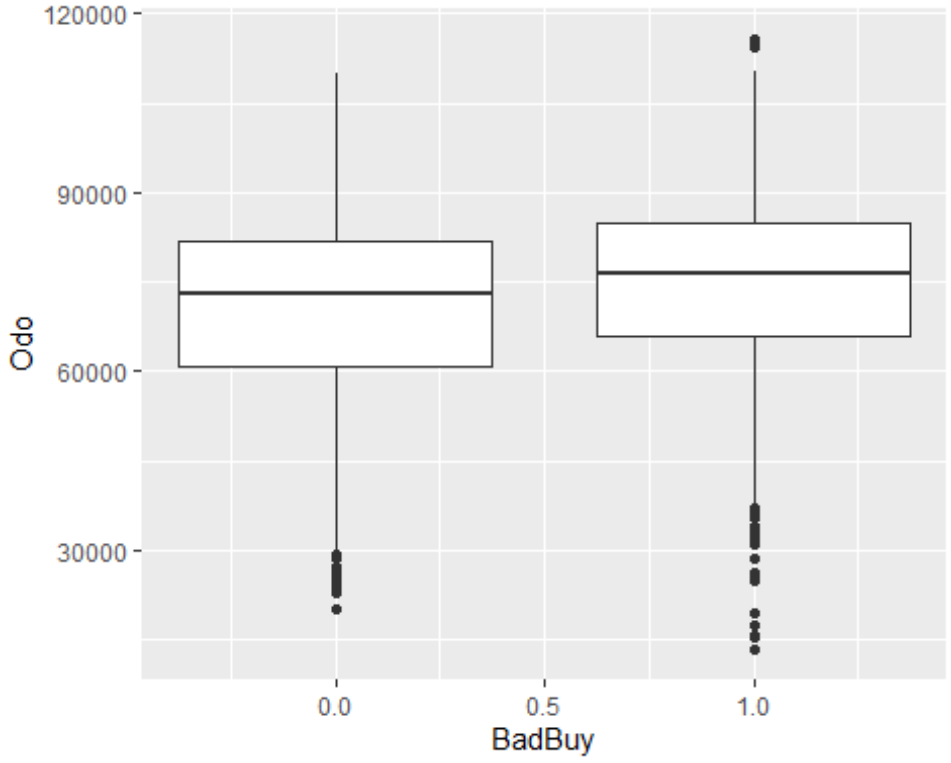
```
plot1 <- dfcTrain %>% ggplot(aes(x= BadBuy, y=MMRAauction, group= BadBuy)) +
geom_boxplot()
plot1
```



```
plot2 <- dfcTrain %>% ggplot(aes(x= BadBuy, y=Age, group= BadBuy)) +
geom_boxplot()
plot2
```




```
plot3 <- dfcTrain %>% ggplot(aes(x= BadBuy, y=Odo, group= BadBuy)) +
geom_boxplot()
plot3
```



Question 2) (b)

```
dfcTrain %>%
  group_by(Size) %>%
  summarise("Lemons" = sum(BadBuy), "GoodCars" = length(BadBuy)- sum(BadBuy),
"PercentLemon" = Lemons/length(BadBuy)*100, "PercentGoodCars" =
GoodCars/length(BadBuy)*100) %>%
  arrange(desc(PercentLemon),desc(PercentGoodCars))
```

```
## # A tibble: 12 x 5
##    Size       Lemons GoodCars PercentLemon PercentGoodCars
##    <chr>       <dbl>    <dbl>        <dbl>           <dbl>
##  1 COMPACT       448      309         59.2            40.8
##  2 LARGESUV       76       53         58.9            41.1
##  3 SPORTS         58       46         55.8            44.2
##  4 MEDIUMSUV     412      348         54.2            45.8
##  5 SMALLSUV      112       97         53.6            46.4
##  6 SMALLTRUCK     47       43         52.2            47.8
##  7 VAN           269      250         51.8            48.2
##  8 MEDIUM       1298     1384         48.4            51.6
##  9 SPECIALTY      68       79         46.3            53.7
## 10 LARGETRUCK    126      156         44.7            55.3
## 11 CROSSOVER      66       88         42.9            57.1
## 12 LARGE         284      423         40.2            59.8
```

Question 3) (a)

```
fitLmTrain <-
  lm(BadBuy~.,data= dfcTrain) %>%
  predict(.,dfcTrain) %>%
  bind_cols(dfcTrain, predictedBadBuy = .)

fitLmTrain
```

```
## # A tibble: 6,540 x 11
##    Auction   Age Make  Color WheelType   Odo Size  MMRAauction MMRAretail
BadBuy
##    <chr>   <dbl> <chr> <chr> <chr>     <dbl> <chr>       <dbl>      <dbl>
<dbl>
##  1 MANHEIM     4 FORD  SILV~ NULL      77591 LARG~        9774      14506
1
##  2 MANHEIM     5 MINI  BLUE  Alloy     80013 COMP~       11040      12423
1
##  3 MANHEIM     2 CHEV~ SILV~ Covers    75493 LARGE        9707      13975
1
##  4 ADESA       4 NISS~ BLUE  NULL      84827 MEDI~        6073       9791
1
##  5 MANHEIM     5 FORD  GREY  Alloy     57388 SPOR~        5574       8984
1
##  6 ADESA       4 SUZU~ BLACK NULL      75822 MEDI~        4033       6979
1
##  7 MANHEIM     2 KIA   BLACK Covers    51059 MEDI~        4839       5726
0
##  8 OTHER       7 FORD  GREY  NULL      74595 LARG~        7649      11059
1
##  9 MANHEIM     6 FORD  BLUE  Alloy     80328 LARG~        6172       7166
0
## 10 MANHEIM     8 PONT~ WHITE Alloy     97173 LARGE        3242       6225
0
## # ... with 6,530 more rows, and 1 more variable: predictedBadBuy <dbl>
```

```
fitLmTest <-
  lm(BadBuy~.,data= dfcTrain) %>%
  predict(.,dfcTest) %>%
  bind_cols(dfcTest, predictedBadBuy = .)

fitLmTest
```

```
## # A tibble: 3,521 x 11
##    Auction   Age Make  Color WheelType   Odo Size  MMRAauction MMRAretail
BadBuy
##    <chr>   <dbl> <chr> <chr> <chr>     <dbl> <chr>       <dbl>      <dbl>
<dbl>
##  1 MANHEIM     6 SATU~ WHITE Covers    81116 MEDI~        2667       3380
0
##  2 OTHER       5 CHEV~ RED   Alloy     54718 MEDI~        6921       7975
```

```
1
##  3 OTHER       5 CHEV~ GOLD  Covers    89365 VAN          6131       9793
1
##  4 ADESA       3 CHEV~ WHITE Covers    71794 VAN          6394       7406
0
##  5 OTHER       3 CHEV~ WHITE NULL      67229 COMP~        5785       9834
1
##  6 MANHEIM     3 DODGE GOLD  Covers    71079 MEDI~        4297       5141
1
##  7 MANHEIM     6 OLDS~ SILV~ Alloy     71235 MEDI~        3325       4091
1
##  8 MANHEIM     8 PONT~ SILV~ Alloy     90325 MEDI~        2150       4937
1
##  9 MANHEIM     6 PONT~ GREEN Alloy     96893 MEDI~        4059       4884
1
## 10 OTHER       2 DODGE BLUE  Covers    45151 MEDI~        7982       9121
1
## # ... with 3,511 more rows, and 1 more variable: predictedBadBuy <dbl>
```

```
perfTrain <- metric_set(rmse, mae)
perfTrain(fitLmTrain, truth= BadBuy, estimate= predictedBadBuy )
```

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

```
perfTest <- metric_set(rmse, mae)
perfTest(fitLmTest, truth= BadBuy, estimate= predictedBadBuy )
```

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

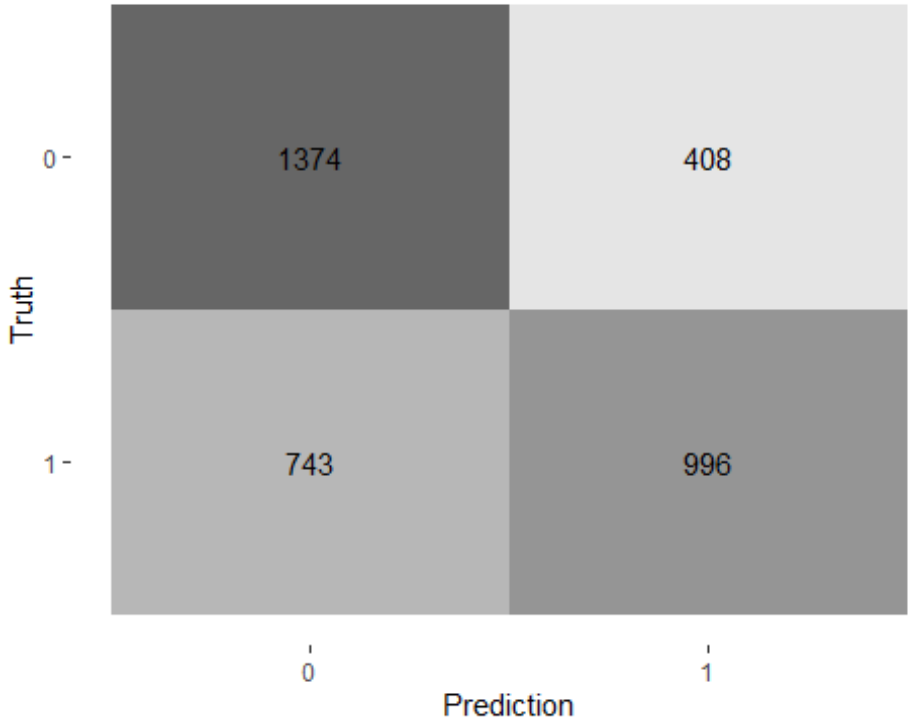
Question 3) (c)

```
resultslm <-
  lm(BadBuy ~., data= dfcTrain) %>%
  predict(dfcTest, type= "response") %>%
  bind_cols(dfcTest,predictedBadBuy=.) %>%
  mutate(predictedBadBuy = as.factor(ifelse(predictedBadBuy > 0.5,1,0)))


resultslm
```

```
## # A tibble: 3,521 x 11
##    Auction   Age Make  Color WheelType   Odo Size  MMRAauction MMRAretail
BadBuy
```

```
##    <chr>   <dbl> <chr> <chr> <chr>     <dbl> <chr>        <dbl>       <dbl>
<dbl>
##  1 MANHEIM     6 SATU~ WHITE Covers    81116 MEDI~         2667        3380
0
##  2 OTHER       5 CHEV~ RED   Alloy     54718 MEDI~         6921        7975
1
##  3 OTHER       5 CHEV~ GOLD  Covers    89365 VAN           6131        9793
1
##  4 ADESA       3 CHEV~ WHITE Covers    71794 VAN           6394        7406
0
##  5 OTHER       3 CHEV~ WHITE NULL      67229 COMP~         5785        9834
1
##  6 MANHEIM     3 DODGE GOLD  Covers    71079 MEDI~         4297        5141
1
##  7 MANHEIM     6 OLDS~ SILV~ Alloy     71235 MEDI~         3325        4091
1
##  8 MANHEIM     8 PONT~ SILV~ Alloy     90325 MEDI~         2150        4937
1
##  9 MANHEIM     6 PONT~ GREEN Alloy     96893 MEDI~         4059        4884
1
## 10 OTHER       2 DODGE BLUE  Covers    45151 MEDI~         7982        9121
1
## # ... with 3,511 more rows, and 1 more variable: predictedBadBuy <fct>
```

```r
resultslm$BadBuy <- as.factor(resultslm$BadBuy)

resultslm %>%
  conf_mat(truth= BadBuy, estimate= predictedBadBuy) %>%
  autoplot(type= "heatmap")
```



Question 3) (d)

```
accuracyConfMatrix <-
  resultslm %>%
  xtabs(~predictedBadBuy + BadBuy, .) %>%
  confusionMatrix(positive = '1')

accuracyConfMatrix
```

```
## Confusion Matrix and Statistics
##
##                BadBuy
## predictedBadBuy    0    1
##               0 1374  743
##               1  408  996
##
##                Accuracy : 0.6731
##                  95% CI : (0.6573, 0.6886)
##     No Information Rate : 0.5061
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.3446
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.5727
##             Specificity : 0.7710
##          Pos Pred Value : 0.7094
```

```
##          Neg Pred Value : 0.6490
##              Prevalence : 0.4939
##          Detection Rate : 0.2829
##    Detection Prevalence : 0.3988
##       Balanced Accuracy : 0.6719
##
##        'Positive' Class : 1
##
```

Question 4) (a)

```
dfcTrain$BadBuy <- as.factor(dfcTrain$BadBuy)
dfcTest$BadBuy <- as.factor(dfcTest$BadBuy)

modelglm <-
  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
```

```
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 object is masked from 'package:lubridate':
##
##     here

## 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

unique(dfc$Color)

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

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

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

unique(dfc$Make)

##  [1] "CHRYSLER"   "PONTIAC"    "SATURN"     "CHEVROLET"  "FORD"
##  [6] "DODGE"      "OLDSMOBILE" "MAZDA"      "JEEP"       "KIA"
## [11] "MITSUBISHI" "GMC"        "BUICK"      "SUZUKI"     "HYUNDAI"
## [16] "NISSAN"     "ISUZU"      "HONDA"      "MERCURY"    "TOYOTA"
## [21] "SCION"      "VOLKSWAGEN" "LINCOLN"    "VOLVO"      "MINI"
## [26] "LEXUS"      "CADILLAC"   "INFINITI"   "ACURA"      "SUBARU"

dfc %>%
  group_by(Make) %>%
  tally(name = 'Count')

## # A tibble: 30 x 2
##    Make      Count
##    <chr>     <int>
##  1 ACURA         9
```

```
##  2 BUICK       103
##  3 CADILLAC      3
##  4 CHEVROLET  2121
##  5 CHRYSLER   1217
##  6 DODGE      1653
##  7 FORD       1764
##  8 GMC          85
##  9 HONDA        77
## 10 HYUNDAI     239
## # ... with 20 more rows
```

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

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

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

resultsglm2 <-
modelGLM %>%
  predict(dfcTest2, type= "raw") %>%
  bind_cols(dfcTest2,predictedBadBuy=.)

resultsglm2
```

```
## # A tibble: 3,553 x 11
##    Auction   Age Make  Color WheelType   Odo Size  MMRAauction MMRAretail
BadBuy
##    <chr>   <dbl> <chr> <chr> <chr>     <dbl> <chr>       <dbl>      <dbl>
<fct>
##  1 MANHEIM     6 SATU~ WHITE Covers    81116 MEDI~        2667       3380
0
##  2 OTHER       5 CHEV~ RED   Alloy     54718 MEDI~        6921       7975
1
##  3 OTHER       5 CHEV~ GOLD  Covers    89365 VAN          6131       9793
1
##  4 ADESA       3 CHEV~ WHITE Covers    71794 VAN          6394       7406
0
##  5 OTHER       3 CHEV~ WHITE NULL      67229 COMP~        5785       9834
1
##  6 MANHEIM     3 DODGE GOLD  Covers    71079 MEDI~        4297       5141
1
##  7 MANHEIM     6 OLDS~ SILV~ Alloy     71235 MEDI~        3325       4091
1
##  8 MANHEIM     8 PONT~ SILV~ Alloy     90325 MEDI~        2150       4937
1
```

```
##  9 MANHEIM     6 PONT~ GREEN Alloy     96893 MEDI~        4059       4884
1
## 10 OTHER       2 DODGE BLUE  Covers    45151 MEDI~        7982       9121
1
## # ... with 3,543 more rows, and 1 more variable: predictedBadBuy <fct>
```

```
summary(resultsglm2)
```

```
##    Auction              Age            Make              Color
##  Length:3553        Min.   :1.000   Length:3553        Length:3553
##  Class :character   1st Qu.:3.000   Class :character   Class :character
##  Mode  :character   Median :4.000   Mode  :character   Mode  :character
##                     Mean   :4.545
##                     3rd Qu.:6.000
##                     Max.   :9.000
##   WheelType             Odo             Size            MMRAauction
##  Length:3553        Min.   :  9446   Length:3553        Min.   :    0
##  Class :character   1st Qu.: 63938   Class :character   1st Qu.: 3885
##  Mode  :character   Median : 75523   Mode  :character   Median : 5638
##                     Mean   : 73128                      Mean   : 5853
##                     3rd Qu.: 84057                      3rd Qu.: 7460
##                     Max.   :109348                      Max.   :32250
##    MMRAretail    BadBuy   predictedBadBuy
##  Min.   :    0   0:1794   0:2073
##  1st Qu.: 5923   1:1759   1:1480
##  Median : 8090
##  Mean   : 8246
##  3rd Qu.:10389
##  Max.   :35330
```

Question 4) (d)

```
confglm2 <- resultsglm2 %>%
  xtabs(~predictedBadBuy + BadBuy, .) %>%
  confusionMatrix(positive = '1')
```

```
confglm2
```

```
## Confusion Matrix and Statistics
##
##                BadBuy
## predictedBadBuy    0    1
##               0 1346  727
##               1  448 1032
##
##                Accuracy : 0.6693
##                  95% CI : (0.6535, 0.6848)
##     No Information Rate : 0.5049
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.3375
```

```
##
##  Mcnemar's Test P-Value : 5.058e-16
##
##             Sensitivity : 0.5867
##             Specificity : 0.7503
##          Pos Pred Value : 0.6973
##          Neg Pred Value : 0.6493
##              Prevalence : 0.4951
##          Detection Rate : 0.2905
##    Detection Prevalence : 0.4165
##       Balanced Accuracy : 0.6685
##
##        'Positive' Class : 1
##
```

Question 4) (e)

```
confglm2 %>%
  tidy()
```

```
## # A tibble: 13 x 6
##    term                class estimate conf.low conf.high   p.value
##    <chr>               <chr>    <dbl>    <dbl>     <dbl>     <dbl>
##  1 accuracy            <NA>     0.669    0.654     0.685  2.03e-87
##  2 kappa               <NA>     0.337   NA        NA      5.06e-16
##  3 sensitivity         1        0.587   NA        NA     NA
##  4 specificity         1        0.750   NA        NA     NA
##  5 pos_pred_value      1        0.697   NA        NA     NA
##  6 neg_pred_value      1        0.649   NA        NA     NA
##  7 precision           1        0.697   NA        NA     NA
##  8 recall              1        0.587   NA        NA     NA
##  9 f1                  1        0.637   NA        NA     NA
## 10 prevalence          1        0.495   NA        NA     NA
## 11 detection_rate      1        0.290   NA        NA     NA
## 12 detection_prevalence 1       0.417   NA        NA     NA
## 13 balanced_accuracy   1        0.668   NA        NA     NA
```

Question 5)

```
set.seed(123)
modelLDA  <-
    train(BadBuy ~ ., data= dfcTrain2, method=
'lda',trControl=trainControl(method='cv', number=10))
summary(modelLDA)
```

```
##             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 <-
    modelLDA %>%
    predict(dfcTest2, type= 'raw') %>%
    bind_cols(dfcTest2, predictedBadBuy=.)
resultsLda
```

```
## # A tibble: 3,553 x 11
##    Auction   Age Make  Color WheelType   Odo Size  MMRAauction MMRAretail
BadBuy
##    <chr>   <dbl> <chr> <chr> <chr>     <dbl> <chr>       <dbl>      <dbl>
<fct>
##  1 MANHEIM     6 SATU~ WHITE Covers    81116 MEDI~        2667       3380
0
##  2 OTHER       5 CHEV~ RED   Alloy     54718 MEDI~        6921       7975
1
##  3 OTHER       5 CHEV~ GOLD  Covers    89365 VAN          6131       9793
1
##  4 ADESA       3 CHEV~ WHITE Covers    71794 VAN          6394       7406
0
##  5 OTHER       3 CHEV~ WHITE NULL      67229 COMP~        5785       9834
1
##  6 MANHEIM     3 DODGE GOLD  Covers    71079 MEDI~        4297       5141
1
##  7 MANHEIM     6 OLDS~ SILV~ Alloy     71235 MEDI~        3325       4091
1
##  8 MANHEIM     8 PONT~ SILV~ Alloy     90325 MEDI~        2150       4937
1
##  9 MANHEIM     6 PONT~ GREEN Alloy     96893 MEDI~        4059       4884
1
## 10 OTHER       2 DODGE BLUE  Covers    45151 MEDI~        7982       9121
1
## # ... with 3,543 more rows, and 1 more variable: predictedBadBuy <fct>
```

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

```
## Confusion Matrix and Statistics
##
##                BadBuy
## predictedBadBuy    0    1
##               0 1382  755
```

```
##                1  412 1004
##
##                Accuracy : 0.6715
##                  95% CI : (0.6558, 0.687)
##     No Information Rate : 0.5049
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.3418
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.5708
##             Specificity : 0.7703
##          Pos Pred Value : 0.7090
##          Neg Pred Value : 0.6467
##              Prevalence : 0.4951
##          Detection Rate : 0.2826
##    Detection Prevalence : 0.3985
##       Balanced Accuracy : 0.6706
##
##        'Positive' Class : 1
##
```

```
#LDA Model

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



#summary(modelLDA)

#resultsLDA <-
 # modelLDA %>%
  #predict(dfcTest2, type= "raw") %>%
  #bind_cols(dfcTest2,predictedBadBuy=.)

#ldaConfMatrix <-
 # resultsLDA %>%
  #xtabs(~predictedBadBuy + BadBuy, .) %>%
  #confusionMatrix(positive = '1')

#ldaConfMatrix
```

Question 5) (b)

```
set.seed(123)
modelknn  <-
    train(BadBuy ~ ., data= dfcTrain2, method= 'knn',
trControl=trainControl(method='cv', number=10), tuneLength=20,
```

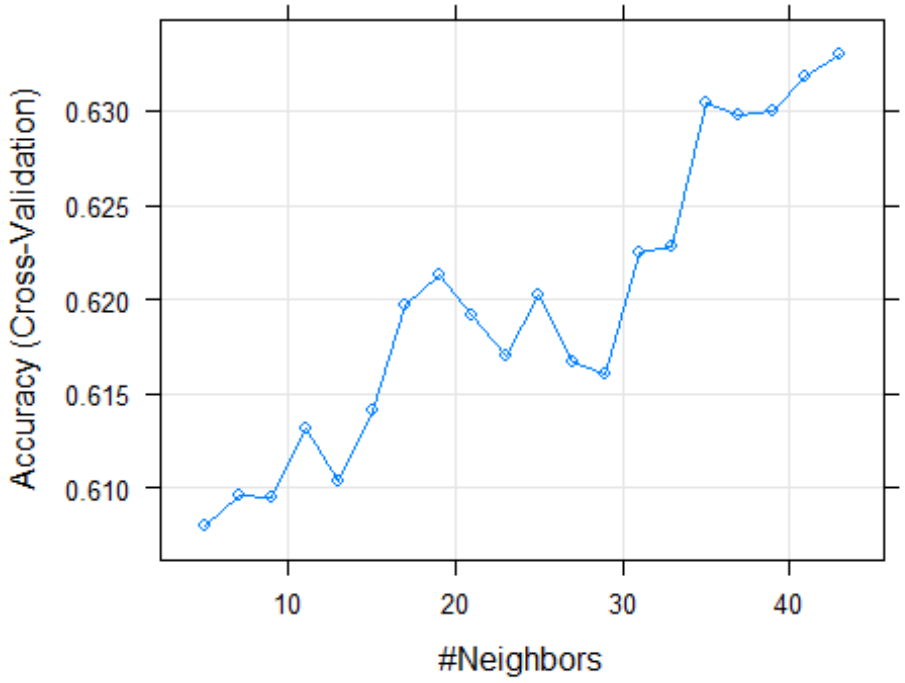
```
preProcess=c("center","scale"))
summary(modelknn)
```

```
##             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
```

```
resultsKNN <-
  modelknn %>%
  predict(dfcTest2, type= "raw") %>%
  bind_cols(dfcTest2,predictedBadBuy=.)

knnConfMatrix <-
  resultsKNN %>%
  xtabs(~predictedBadBuy + BadBuy, .) %>%
  confusionMatrix(positive = '1')

knnConfMatrix
```

```
## Confusion Matrix and Statistics
##
##                BadBuy
## predictedBadBuy    0    1
##               0 1329  818
##               1  465  941
##
##                Accuracy : 0.6389
##                  95% CI : (0.6229, 0.6547)
##     No Information Rate : 0.5049
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.2763
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.5350
##             Specificity : 0.7408
##          Pos Pred Value : 0.6693
##          Neg Pred Value : 0.6190
##              Prevalence : 0.4951
##          Detection Rate : 0.2648
##    Detection Prevalence : 0.3957
##       Balanced Accuracy : 0.6379
##
```

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

plot(modelknn)
```




```
modelknn$bestTune

##     k
## 20 43
```

Question 5) (c)

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

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

summary(modelLasso)

##             Length Class      Mode
## a0             75   -none-     numeric
## beta         4425   dgCMatrix  S4
## df             75   -none-     numeric
## dim             2   -none-     numeric
## lambda         75   -none-     numeric
```

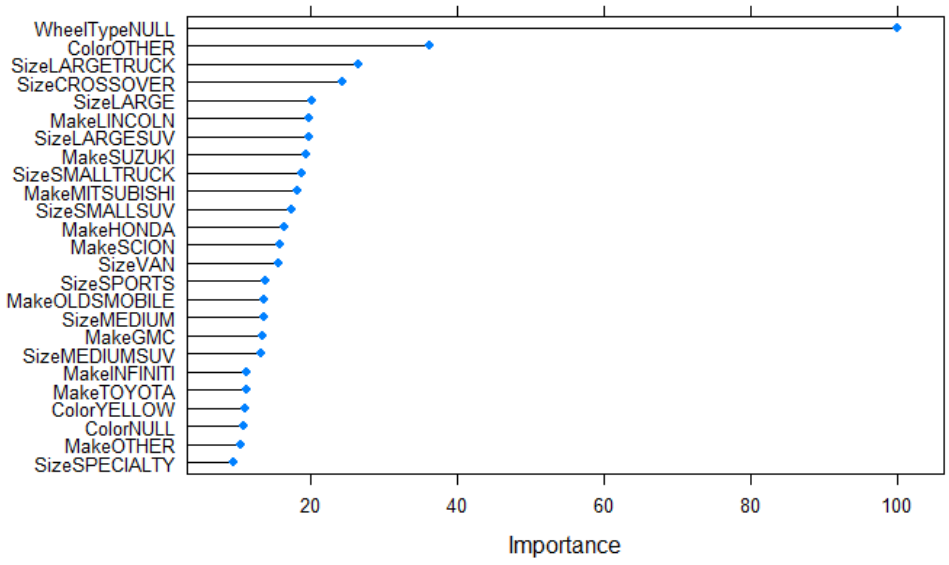
```
## dev.ratio     75   -none-     numeric
## nulldev        1   -none-     numeric
## npasses        1   -none-     numeric
## jerr           1   -none-     numeric
## offset         1   -none-     logical
## classnames     2   -none-     character
## call           5   -none-     call
## nobs           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 <-
  predict(modelLasso,dfcTest2, type= 'raw') %>%
  bind_cols(dfcTest2, predictedBadBuy=.)
```

```
varImp(modelLasso)$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
```

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




```
#Optimum lambda selected by the algorithm
modelLasso$bestTune$lambda   # You can also run fitLasso$finalModel$lambdaOpt
```

```
## [1] 0.0003053856
```

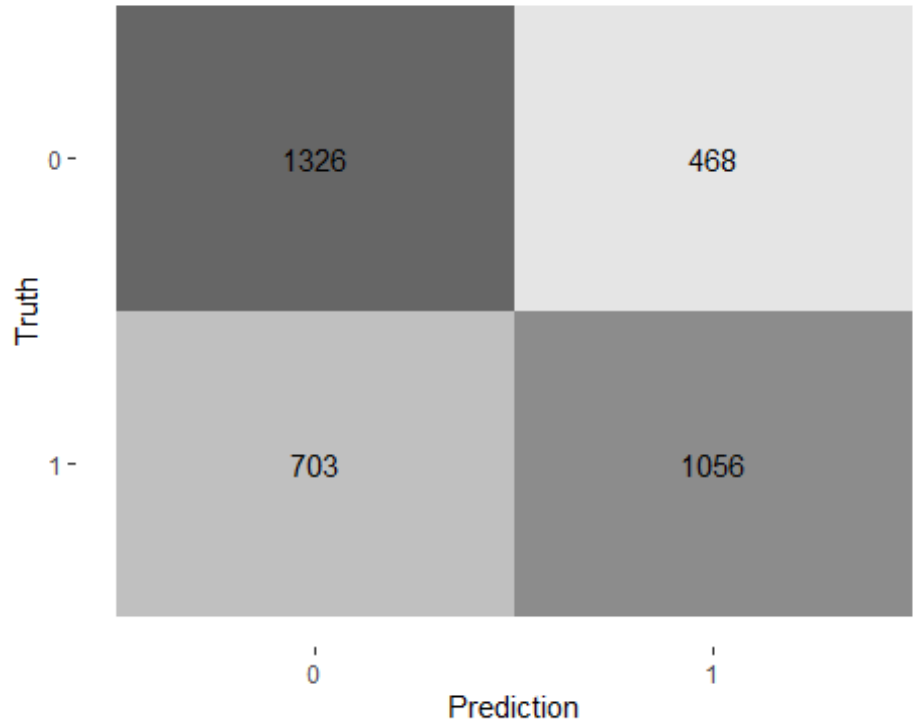
Question 5) (d)

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

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

summary(modelRidge)
```

```
##             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 <-
  predict(modelRidge,dfcTest2, type= 'raw') %>%
  bind_cols(dfcTest2, predictedBadBuy=.)

resultsRidge %>%
  conf_mat(truth = BadBuy , estimate = predictedBadBuy) %>%
  autoplot(type = 'heatmap')
```




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

## Confusion Matrix and Statistics
##
##                BadBuy
## predictedBadBuy    0    1
##               0 1326  703
##               1  468 1056
##
##                Accuracy : 0.6704
##                  95% CI : (0.6547, 0.6859)
```

```
##     No Information Rate : 0.5049
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.3399
##
##  Mcnemar's Test P-Value : 8.023e-12
##
##             Sensitivity : 0.6003
##             Specificity : 0.7391
##          Pos Pred Value : 0.6929
##          Neg Pred Value : 0.6535
##              Prevalence : 0.4951
##          Detection Rate : 0.2972
##    Detection Prevalence : 0.4289
##       Balanced Accuracy : 0.6697
##
##        'Positive' Class : 1
##
```

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

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

summary(modelElasticNet)
```

```
##             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
```

```
resultsElasticNet <-
  predict(modelElasticNet,dfcTest2, type= 'raw') %>%
  bind_cols(dfcTest2, predictedBadBuy=.)

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

```
## Confusion Matrix and Statistics
##
##                BadBuy
## predictedBadBuy    0    1
##               0 1343  729
##               1  451 1030
##
##                Accuracy : 0.6679
##                  95% CI : (0.6521, 0.6834)
##     No Information Rate : 0.5049
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.3347
##
##  Mcnemar's Test P-Value : 7.397e-16
##
##             Sensitivity : 0.5856
##             Specificity : 0.7486
##          Pos Pred Value : 0.6955
##          Neg Pred Value : 0.6482
##              Prevalence : 0.4951
##          Detection Rate : 0.2899
##    Detection Prevalence : 0.4168
##       Balanced Accuracy : 0.6671
##
##        'Positive' Class : 1
##
```

Question 5) (e)

```
set.seed(123)
modelQDA  <-
    train(BadBuy ~ ., data= dfcTrain2, method=
'qda',trControl=trainControl(method='cv', number=10))
```

```
## Warning: model fit failed for Fold03: 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(modelQDA)
```

```
##             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
```

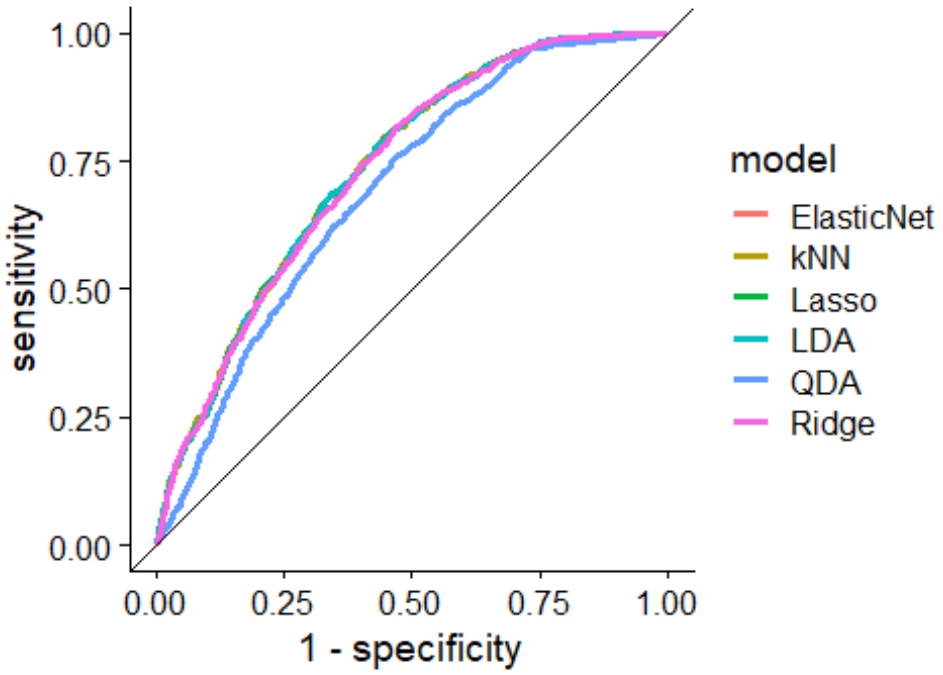
```
resultsQDA <-
  predict(modelQDA,dfcTest2, type= 'raw') %>%
  bind_cols(dfcTest2, predictedBadBuy=.)
```

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

```
## Confusion Matrix and Statistics
##
##                BadBuy
## predictedBadBuy    0    1
##               0 1483  973
##               1  311  786
##
##                Accuracy : 0.6386
##                  95% CI : (0.6226, 0.6544)
##     No Information Rate : 0.5049
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.2745
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.4468
##             Specificity : 0.8266
##          Pos Pred Value : 0.7165
##          Neg Pred Value : 0.6038
##              Prevalence : 0.4951
##          Detection Rate : 0.2212
##    Detection Prevalence : 0.3088
##       Balanced Accuracy : 0.6367
##
##        'Positive' Class : 1
##
```

Question 5) (f)

```
resultsLDAProb <- bind_cols(dfcTest2,modelGLM %>%  predict(dfcTest2,
type='prob') )%>%mutate(model="LDA")
resultsKNNProb <- bind_cols(dfcTest2,modelLDA %>%  predict(dfcTest2,
type='prob') )%>%mutate(model="kNN")
resultsLassoProb <- bind_cols(dfcTest2,modelLasso %>%  predict(dfcTest2,
type='prob') )%>%mutate(model="Lasso")
resultsRidgeProb <- bind_cols(dfcTest2,modelRidge %>%  predict(dfcTest2,
type='prob') )%>%mutate(model="Ridge")
resultsElasticNetProb <- bind_cols(dfcTest2,modelElasticNet %>%
predict(dfcTest2, type='prob') )%>%mutate(model="ElasticNet")
resultsQDAProb <- bind_cols(dfcTest2,modelQDA %>%  predict(dfcTest2,
type='prob') )%>%mutate(model="QDA")

library(cowplot)

## Warning: package 'cowplot' was built under R version 3.6.3

##
## ************************************************************

## Note: As of version 1.0.0, cowplot does not change the

##   default ggplot2 theme anymore. To recover the previous

##   behavior, execute:
##   theme_set(theme_cowplot())

## ************************************************************

##
## Attaching package: 'cowplot'

## The following object is masked from 'package:lubridate':
##
##     stamp

glmOutAll <- bind_rows(resultsLDAProb, resultsKNNProb, resultsLassoProb,
resultsRidgeProb, resultsElasticNetProb, resultsQDAProb)



glmOutAll %>%
  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() +
  theme_cowplot()
```



Bonus Question:

```
library(grplasso)

dfcTrainGroup <-
  dfcTrain2 %>%
  mutate(BadBuy = as.numeric(BadBuy)) %>%
  mutate(BadBuy = ifelse(BadBuy == 2, 1, 0))

set.seed(123)

modelGroupedLasso <- grplasso(BadBuy ~ ., data=dfcTrainGroup, model=LogReg(),
lambda=50)

## Lambda: 50  nr.var: 42

modelGroupedLasso$coefficients

##                              50
## (Intercept)      -1.732604e+00
## AuctionMANHEIM    0.000000e+00
## AuctionOTHER      0.000000e+00
## Age               2.272898e-01
## MakeCHEVROLET    -2.793690e-02
## MakeCHRYSLER      1.200698e-02
## MakeDODGE        -8.985755e-03
```

```
## MakeFORD          6.286615e-03
## MakeGMC          -5.337319e-02
## MakeHONDA        -4.819393e-02
## MakeHYUNDAI      -1.549876e-02
## MakeINFINITI      6.727813e-02
## MakeISUZU        -3.584038e-02
## MakeJEEP         -9.067129e-04
## MakeKIA          -1.930114e-02
## MakeLINCOLN       6.482665e-02
## MakeMAZDA         1.932910e-03
## MakeMERCURY       2.500740e-02
## MakeMITSUBISHI   -5.797448e-02
## MakeNISSAN       -7.640744e-05
## MakeOLDSMOBILE    4.177004e-02
## MakeOTHER         4.418157e-02
## MakePONTIAC      -1.669154e-02
## MakeSATURN        1.442574e-02
## MakeSCION        -6.077640e-02
## MakeSUZUKI        5.205267e-02
## MakeTOYOTA       -3.277233e-02
## MakeVOLKSWAGEN    2.505957e-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
## 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  -1.147940e-01
## WheelTypeNULL     2.715202e+00
## WheelTypeSpecial  1.092006e-02
## Odo               1.012407e-05
## SizeCROSSOVER    -1.973973e-01
## SizeLARGE        -2.388554e-01
## SizeLARGESUV     -1.600920e-01
## SizeLARGETRUCK   -2.665217e-01
## SizeMEDIUM       -1.229139e-01
## SizeMEDIUMSUV    -1.267654e-01
## SizeSMALLSUV     -1.507574e-01
## SizeSMALLTRUCK   -1.831945e-01
## SizeSPECIALTY    -3.367548e-02
## SizeSPORTS       -1.303393e-01
```

```
## SizeVAN          -1.484230e-01
## MMRAauction      -1.790527e-05
## MMRAretail        0.000000e+00
```

```
set.seed(123)
modelGroupedLasso1 <- grplasso(BadBuy ~ ., data=dfcTrainGroup,
model=LogReg(), lambda=100)
```

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

```
modelGroupedLasso1$coefficients
```

```
##                            100
## (Intercept)      -1.571244e+00
## AuctionMANHEIM    0.000000e+00
## AuctionOTHER      0.000000e+00
## Age               2.103677e-01
## 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
## MakeLINCOLN       0.000000e+00
## MakeMAZDA         0.000000e+00
## MakeMERCURY       0.000000e+00
## MakeMITSUBISHI    0.000000e+00
## MakeNISSAN        0.000000e+00
## MakeOLDSMOBILE    0.000000e+00
## MakeOTHER         0.000000e+00
## MakePONTIAC       0.000000e+00
## MakeSATURN        0.000000e+00
## MakeSCION         0.000000e+00
## MakeSUZUKI        0.000000e+00
## MakeTOYOTA        0.000000e+00
## MakeVOLKSWAGEN    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
## 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
```