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.

1.a.

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

library("tidyverse")
```

```
## -- 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
```

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

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

```
## Registered S3 method overwritten by 'xts':
##   method     from
##   as.zoo.xts zoo
```

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

```
## v broom     0.5.3     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
```

```
## -- 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()
## x 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
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

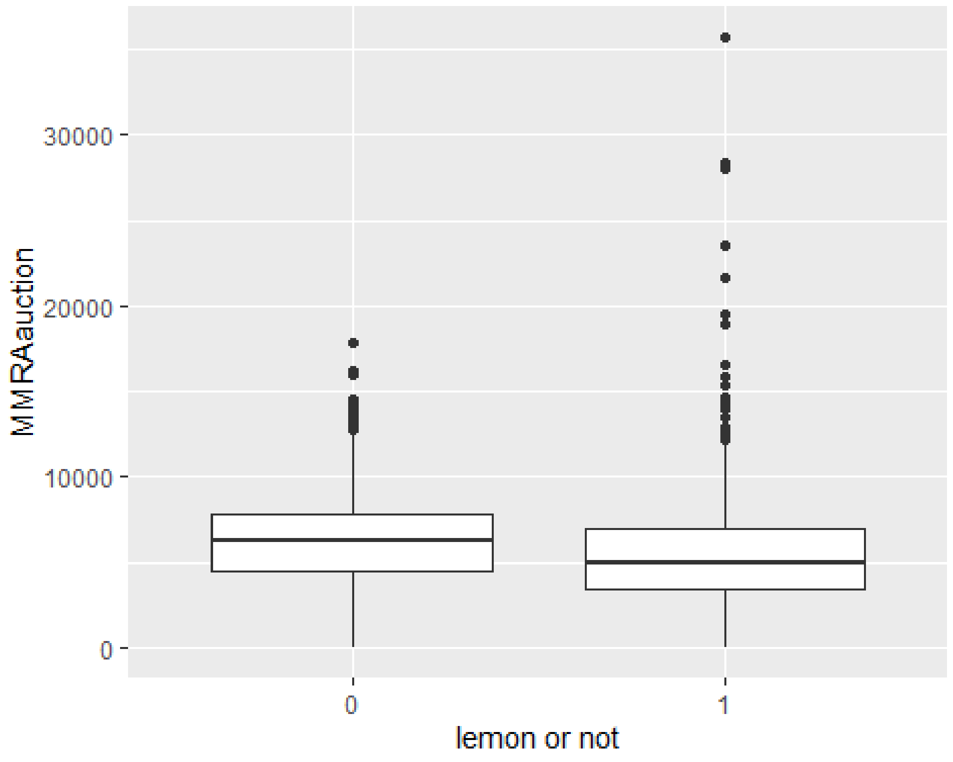
skim_variable	n_missing	complete_rate	min	max	empty	n_unique	whitespace
Auction	0	1	5	7	0	3	0
Make	0	1	3	10	0	30	0
Color	0	1	3	8	0	17	0
WheelType	0	1	4	7	0	4	0
Size	0	1	3	10	0	12	0

Variable type: numeric

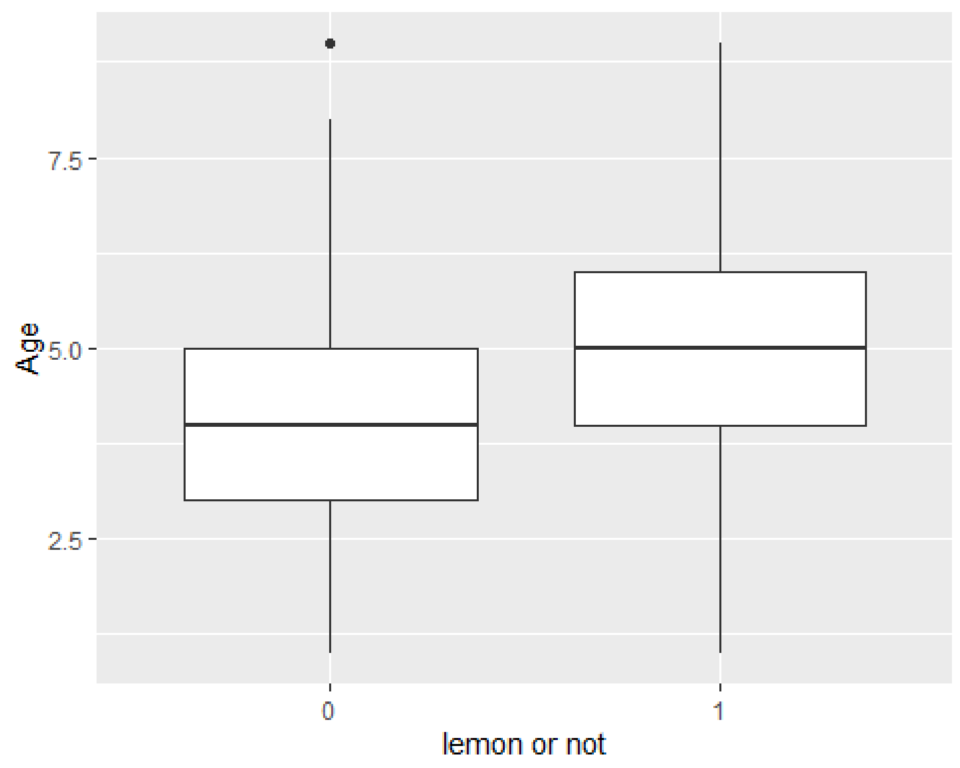
skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
Age	0	1	4.50	1.77	1	3	4	6	9	▂▇▃▅▁
Odo	0	1	72903.87	14498.87	9446	63488	74942	83663	115717	▁▂▆▇▁
MMRAauction	0	1	5812.38	2578.85	0	3877	5588	7450	35722	▇▃▁▁▁
MMRAretail	0	1	8171.51	3257.19	0	5872	8052	10315	39080	▇▇▁▁▁
BadBuy	0	1	0.50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

1.b.

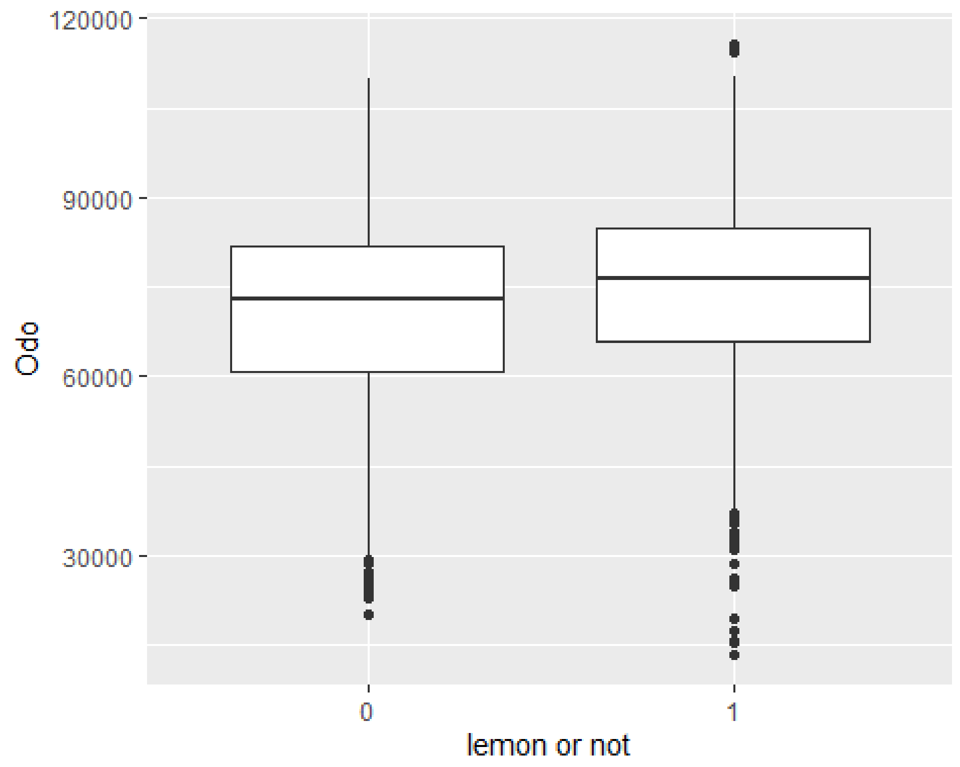
```
set.seed(52156)
dfcTrain1<- dfc %>% sample_frac(0.65)
dfcTest1<- dplyr::setdiff(dfc, dfcTrain1)

plot1<- dfcTrain1%>%
  ggplot()+geom_boxplot(aes(x=as.factor(BadBuy), y=MMRAauction))+xlab("lemon
or not")
plot1
```



```
plot2<- dfcTrain1%>%
  ggplot()+geom_boxplot(aes(x=as.factor(BadBuy), y=Age))+xlab("lemon or not")
plot2
```



```
plot3<- dfcTrain1%>%
  ggplot()+geom_boxplot(aes(x=as.factor(BadBuy), y=Odo))+xlab("lemon or not")
plot3
```



2.b.

```
dfcTrain1 %>%
  group_by(BadBuy, Size) %>%
  tally() %>%
  mutate(pct = 100*n/sum(n)) %>%
  arrange(desc(BadBuy), desc(pct))
```

```
## # A tibble: 24 x 4
## # Groups:   BadBuy [2]
##    BadBuy Size            n   pct
##     <dbl> <chr>       <int> <dbl>
##  1      1 MEDIUM       1298 39.8
##  2      1 COMPACT       448 13.7
##  3      1 MEDIUMSUV     412 12.6
##  4      1 LARGE         284  8.70
##  5      1 VAN           269  8.24
##  6      1 LARGETRUCK    126  3.86
##  7      1 SMALLSUV      112  3.43
##  8      1 LARGESUV       76  2.33
##  9      1 SPECIALTY      68  2.08
## 10      1 CROSSOVER      66  2.02
## # ... with 14 more rows
```

```
fitlem<-
  lm(formula = BadBuy~ . , data=dfcTrain1)

summary(fitlem)
```

```
##
## Call:
## lm(formula = BadBuy ~ ., data = dfcTrain1)
##
## 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
```

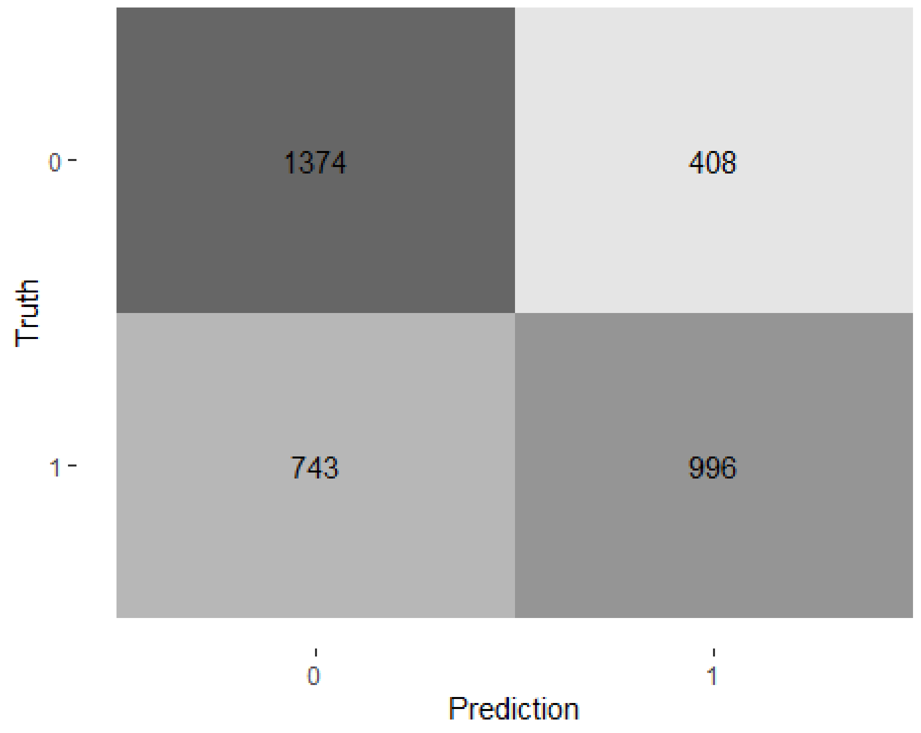
3.a

```
testresult <- dfcTest1 %>%
            mutate(predictedtest = predict(fitlem, dfcTest1))
testresult

## # 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: predictedtest <dbl>

performance <-
   metric_set(rmse, mae)
performance(testresult, truth= BadBuy, estimate = predictedtest)

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

trainresult <- dfcTrain1 %>%
            mutate(predictedtest = predict(fitlem, dfcTrain1))
trainresult

## # 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: predictedtest <dbl>

performance <-
   metric_set(rmse, mae)
performance(trainresult, truth= BadBuy, estimate = predictedtest)

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

3.c.

```
lemresults <-
    lm(formula = BadBuy ~ . , data = dfcTrain1) %>%
  predict(dfcTest1, type  = 'response') %>%
  bind_cols(dfcTest1, predictedProb=.) %>%
  mutate(predictedclass = as.factor(ifelse(predictedProb>0.5,1,0)))
```

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

lemresults %>%
  conf_mat(truth = (BadBuy) ,estimate = predictedclass) %>%
  autoplot(type = 'heatmap')
```



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

predict(fitlem, result, type="response")

##          1
## -0.1410712
```

4.a.

```r
dfcTrain1<- dfcTrain1%>%
  mutate(BadBuy=as.factor(BadBuy))
dfcTest1<- dfcTest1%>%
  mutate(BadBuy=as.factor(BadBuy))

fitlemon <-
    train(BadBuy~ ., data= dfcTrain1, family='binomial', method='glm') %>%
    predict(dfcTest1 , type='raw')%>%
    bind_cols(dfcTest1, predictedProb=.)
```

```
## 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
```

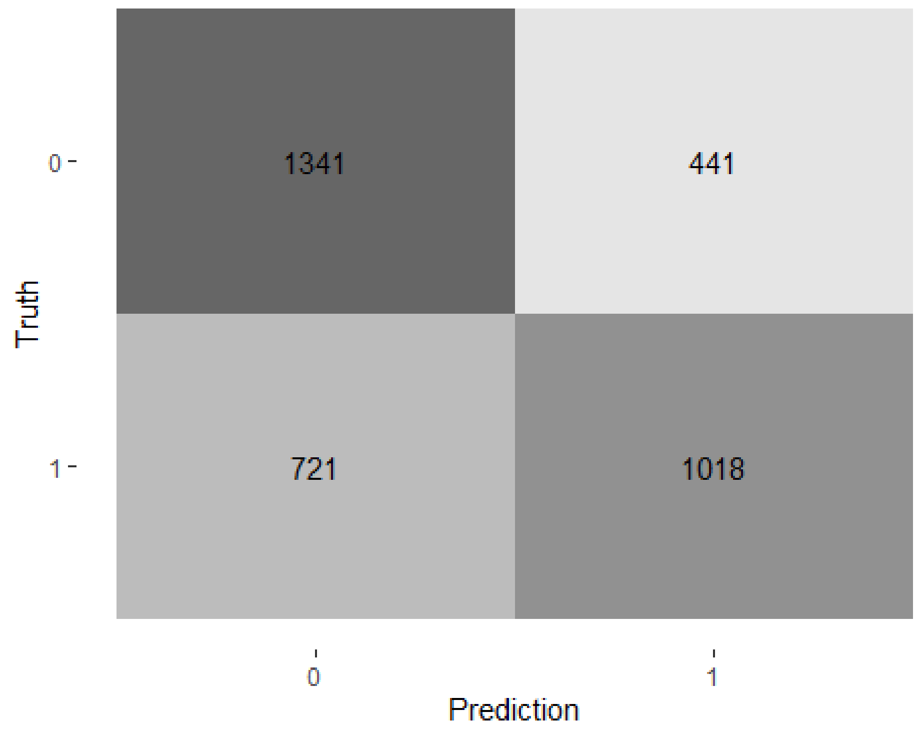
4.a.i

```
dfc <- dfc%>%
  mutate(Color = if_else(Color == "NULL", "NOTAVAIL", Color))

dfc<- dfc%>%
  mutate(Make = if_else(Make %in% c("ACURA", "CADILLAC", "VOLVO",
"SUBARU","MINI", "LEXUS", NA), "OTHER", Make))

dfc$BadBuy <- as.factor(dfc$BadBuy)

set.seed(52156)
dfcTrain1<- dfc %>% sample_frac(0.65)
dfcTest1<- dplyr::setdiff(dfc, dfcTrain1)

dfcTrain1<- dfcTrain1%>%
  mutate(BadBuy=as.factor(BadBuy))
dfcTest1<- dfcTest1%>%
  mutate(BadBuy=as.factor(BadBuy))

fitlemon <-
    glm(BadBuy~ ., data= dfcTrain1, family='binomial')

summary(fitlemon)
```

```
##
## Call:
## glm(formula = BadBuy ~ ., family = "binomial", data = dfcTrain1)
##
## 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
## ColorNOTAVAIL    -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
```

4.d.

```
fitlemonglm <-
    glm(formula = BadBuy ~ . ,family= 'binomial', data = dfcTrain1) %>%
  predict(dfcTest1, type  = 'response') %>%
  bind_cols(dfcTest1, predictedProb=.) %>%
  mutate(predictedclass = as.factor(ifelse(predictedProb>0.5,1,0)))

fitlemonglm %>%
  conf_mat(truth = BadBuy ,estimate = predictedclass) %>%
  autoplot(type = 'heatmap')
```



4.e.

```
result1 = data.frame(Auction="ADESA", Age=1, Make="HONDA",Color="SILVER",
WheelType="Covers",Odo=10000, Size="LARGE",MMRAauction=8000,
MMRAretail=10000)

predict(fitlemon, result1, type="response")

##          1
## 0.04152115
```

5.a.

```
set.seed(123)

ctrl <- trainControl(method = "repeatedcv", repeats = 10)


caretLDAresults <-
    train(BadBuy~., family = 'binomial', method = 'lda', data = dfcTrain1,
trControl = ctrl) %>%
    predict(dfcTest1, type = 'raw') %>%
    bind_cols(dfcTest1, predictedProb=.)

caretLDAresults
```

```
## # 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>
<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 3,511 more rows, and 1 more variable: predictedProb <fct>
```

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

```
## Confusion Matrix and Statistics
##
##               BadBuy
## predictedProb    0    1
##              0 1377  749
##              1  405  990
##
##                Accuracy : 0.6723
##                  95% CI : (0.6565, 0.6878)
##     No Information Rate : 0.5061
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.3428
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.5693
##             Specificity : 0.7727
##          Pos Pred Value : 0.7097
##          Neg Pred Value : 0.6477
##              Prevalence : 0.4939
##          Detection Rate : 0.2812
##    Detection Prevalence : 0.3962
##       Balanced Accuracy : 0.6710
##
##        'Positive' Class : 1
##
```

```
summary(caretLDAresults)
```

```
##    Auction                Age             Make              Color
##  Length:3521        Min.   :1.000   Length:3521        Length:3521
##  Class :character   1st Qu.:3.000   Class :character   Class :character
##  Mode  :character   Median :4.000   Mode  :character   Mode  :character
##                     Mean   :4.542
##                     3rd Qu.:6.000
##                     Max.   :9.000
##   WheelType              Odo             Size             MMRAauction
##  Length:3521        Min.   :  9446   Length:3521        Min.   :    0
##  Class :character   1st Qu.: 63971   Class :character   1st Qu.: 3883
##  Mode  :character   Median : 75523   Mode  :character   Median : 5626
##                     Mean   : 73158                      Mean   : 5827
##                     3rd Qu.: 84057                      3rd Qu.: 7434
##                     Max.   :109348                      Max.   :32250
##    MMRAretail    BadBuy   predictedProb
##  Min.   :    0   0:1782   0:2126
##  1st Qu.: 5916   1:1739   1:1395
##  Median : 8072
##  Mean   : 8219
##  3rd Qu.:10382
##  Max.   :35330
```

5.b.

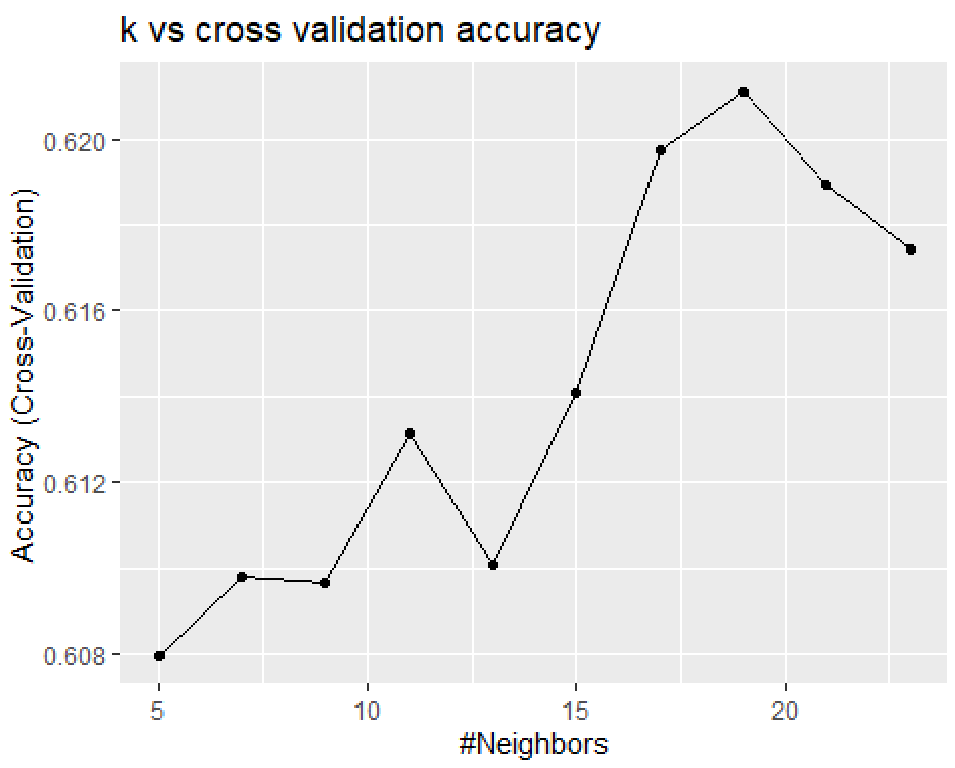
```
set.seed(123)

ctrl <- trainControl(method = "cv", number=10)
caretknnresults <- train(BadBuy~., data = dfcTrain1, method = "knn",
trControl=ctrl, preProcess = c("center", "scale"), tuneLength = 10)




caretknnresults
```

```
## k-Nearest Neighbors
##
## 6540 samples
##    9 predictor
##    2 classes: '0', '1'
##
```

```
## Pre-processing: centered (59), scaled (59)
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 5885, 5886, 5887, 5887, 5885, 5886, ...
## Resampling results across tuning parameters:
##
##   k   Accuracy   Kappa
##    5  0.6079622  0.2157892
##    7  0.6098006  0.2194493
##    9  0.6096444  0.2191307
##   11  0.6131531  0.2261576
##   13  0.6100987  0.2200507
##   15  0.6140707  0.2279765
##   17  0.6197345  0.2392840
##   19  0.6211079  0.2420234
##   21  0.6189621  0.2377276
##   23  0.6174393  0.2346678
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was k = 19.
```

5.b.i&ii

```
caretknnresults %>%
  ggplot(aes(x=k, y=Accuracy))+ggtitle("k vs cross validation accuracy")
```



5.b.iii

```
knnresults<-caretknnresults%>%
  predict(dfcTest1, type = 'raw') %>%
    bind_cols(dfcTest1, predictedProb=.)

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

## Confusion Matrix and Statistics
##
##              BadBuy
## predictedProb    0    1
##             0 1249  774
##             1  533  965
##
##                Accuracy : 0.6288
##                  95% CI : (0.6126, 0.6448)
##     No Information Rate : 0.5061
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.2562
##
##  Mcnemar's Test P-Value : 3.168e-11
##
##             Sensitivity : 0.5549
##             Specificity : 0.7009
##          Pos Pred Value : 0.6442
##          Neg Pred Value : 0.6174
##              Prevalence : 0.4939
##          Detection Rate : 0.2741
##    Detection Prevalence : 0.4254
##       Balanced Accuracy : 0.6279
##
##        'Positive' Class : 1
##
```

5.c.

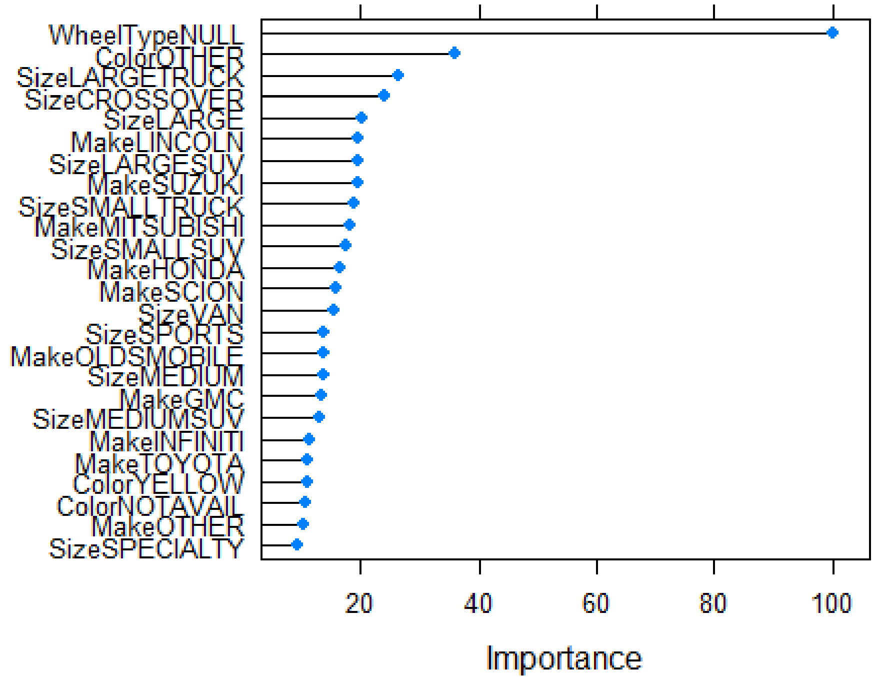
```
library("glmnet")

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

## Loading required package: Matrix

##
## Attaching package: 'Matrix'

## The following objects are masked from 'package:tidyr':
##
##     expand, pack, unpack
## Loaded glmnet 3.0-2
```

```
lambdaValues <- 10^seq(-5, 2, length = 100)

set.seed(123)

fitlemonlasso <- train(BadBuy ~ ., family='binomial', data=dfcTrain1,
method='glmnet', trControl=trainControl(method='cv', number=10), tuneGrid =
expand.grid(alpha=1, lambda=lambdaValues))


varImp(fitlemonlasso)$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
```

5.c.ii

```
plot(varImp(fitlemonlasso), top = 25)
```



5.c.iii

```
fitlemonlasso$bestTune$lambda

## [1] 0.0003053856
```

5.c.iv.

```
Lassoresults <-
  fitlemonlasso %>%
  predict(dfcTest1, type='raw') %>%
  bind_cols(dfcTest1, predictedClass=.)

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

## Confusion Matrix and Statistics
##
##               BadBuy
## predictedClass    0    1
##              0 1339  721
##              1  443 1018
##
##                Accuracy : 0.6694
##                  95% CI : (0.6536, 0.6849)
##     No Information Rate : 0.5061
##     P-Value [Acc > NIR] : < 2e-16
```

```
##
##                   Kappa : 0.3374
##
##  Mcnemar's Test P-Value : 4.7e-16
##
##             Sensitivity : 0.5854
##             Specificity : 0.7514
##          Pos Pred Value : 0.6968
##          Neg Pred Value : 0.6500
##              Prevalence : 0.4939
##          Detection Rate : 0.2891
##    Detection Prevalence : 0.4149
##       Balanced Accuracy : 0.6684
##
##        'Positive' Class : 1
##
```

5.d.i

```
set.seed(123)

fitlemonridge <- train(BadBuy ~ ., family='binomial', data=dfcTrain1,
method='glmnet', trControl=trainControl(method='cv', number=10), tuneGrid =
expand.grid(alpha=0, lambda=lambdaValues))

ridgeresults <-
  fitlemonridge %>%
  predict(dfcTest1, type='raw') %>%
  bind_cols(dfcTest1, predictedClass=.)

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

## Confusion Matrix and Statistics
##
##               BadBuy
## predictedClass    0    1
##              0 1323  699
##              1  459 1040
##
##                Accuracy : 0.6711
##                  95% CI : (0.6553, 0.6866)
##     No Information Rate : 0.5061
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.341
##
##  Mcnemar's Test P-Value : 2.166e-12
##
##             Sensitivity : 0.5980
##             Specificity : 0.7424
##          Pos Pred Value : 0.6938
##          Neg Pred Value : 0.6543
##              Prevalence : 0.4939
##          Detection Rate : 0.2954
##    Detection Prevalence : 0.4257
##       Balanced Accuracy : 0.6702
##
##        'Positive' Class : 1
##
```

```
fitlemonridge$bestTune$lambda
```

```
## [1] 0.0559081
```

5.d.ii

```
#elastic net
set.seed(123)
fitLemonElastic <- train(BadBuy ~ ., family='binomial', data=dfcTrain1,
method='glmnet', trControl=trainControl(method='cv', number=10),
tuneLength=10)

elasticResults <-
  fitLemonElastic %>%
  predict(dfcTest1, type='raw') %>%
  bind_cols(dfcTest1, predictedClass=.)

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

```
## Confusion Matrix and Statistics
##
##               BadBuy
## predictedClass    0    1
##              0 1338  721
##              1  444 1018
##
##                Accuracy : 0.6691
##                  95% CI : (0.6533, 0.6847)
##     No Information Rate : 0.5061
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.3369
##
##  Mcnemar's Test P-Value : 6.154e-16
##
##             Sensitivity : 0.5854
##             Specificity : 0.7508
##          Pos Pred Value : 0.6963
##          Neg Pred Value : 0.6498
##              Prevalence : 0.4939
##          Detection Rate : 0.2891
##    Detection Prevalence : 0.4152
##       Balanced Accuracy : 0.6681
##
##        'Positive' Class : 1
##
```

5.e

```
set.seed(123)

fitlemonqda <- train(BadBuy ~ ., family= "binomial", data=dfcTrain1,
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.

qdaresults <- fitlemonqda %>%
  predict(dfcTest1, type = 'raw') %>%
  bind_cols(dfcTest1, predictedProb=.)

qdaresults

## # 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>
<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 3,511 more rows, and 1 more variable: predictedProb <fct>

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

## Confusion Matrix and Statistics
##
##                BadBuy
## predictedProb    0    1
##              0 1483  973
##              1  299  766
##
##                Accuracy : 0.6387
##                  95% CI : (0.6226, 0.6546)
##     No Information Rate : 0.5061
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.274
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.4405
##             Specificity : 0.8322
##          Pos Pred Value : 0.7192
##          Neg Pred Value : 0.6038
##              Prevalence : 0.4939
##          Detection Rate : 0.2176
##    Detection Prevalence : 0.3025
##       Balanced Accuracy : 0.6363
##
##        'Positive' Class : 1
##
```