Assignment 5

James Tran

4/13/2018

Load the libraries.

library(tidyverse)

## -- Attaching packages ------------------------------------------------------------------------------------------------------------------------------------- tidyverse 1.2.1 --

## v ggplot2 2.2.1 v purrr 0.2.4  
## v tibble 1.4.2 v dplyr 0.7.4  
## v tidyr 0.7.2 v stringr 1.2.0  
## v readr 1.1.1 v forcats 0.2.0

## -- Conflicts ---------------------------------------------------------------------------------------------------------------------------------------- tidyverse\_conflicts() --  
## x dplyr::filter() masks stats::filter()  
## x dplyr::lag() masks stats::lag()

library(broom)  
library(pROC)

## Warning: package 'pROC' was built under R version 3.4.4

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

##   
## Attaching package: 'pROC'

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

str(diamonds)

## Classes 'tbl\_df', 'tbl' and 'data.frame': 53940 obs. of 10 variables:  
## $ carat : num 0.23 0.21 0.23 0.29 0.31 0.24 0.24 0.26 0.22 0.23 ...  
## $ cut : Ord.factor w/ 5 levels "Fair"<"Good"<..: 5 4 2 4 2 3 3 3 1 3 ...  
## $ color : Ord.factor w/ 7 levels "D"<"E"<"F"<"G"<..: 2 2 2 6 7 7 6 5 2 5 ...  
## $ clarity: Ord.factor w/ 8 levels "I1"<"SI2"<"SI1"<..: 2 3 5 4 2 6 7 3 4 5 ...  
## $ depth : num 61.5 59.8 56.9 62.4 63.3 62.8 62.3 61.9 65.1 59.4 ...  
## $ table : num 55 61 65 58 58 57 57 55 61 61 ...  
## $ price : int 326 326 327 334 335 336 336 337 337 338 ...  
## $ x : num 3.95 3.89 4.05 4.2 4.34 3.94 3.95 4.07 3.87 4 ...  
## $ y : num 3.98 3.84 4.07 4.23 4.35 3.96 3.98 4.11 3.78 4.05 ...  
## $ z : num 2.43 2.31 2.31 2.63 2.75 2.48 2.47 2.53 2.49 2.39 ...

## Problem 1

In the diamonds dataframe create the variable price per carat (ppc) and then creae a boolean variable HiValue which is true when the price per carat of a diamond is greater than the 90th percentile of this variable. Produce a table of the variable HiValue to verify that your code worked.

diamonds %>%  
 mutate(ppc = price/carat) %>%  
 mutate(HiValue = ifelse(ppc > quantile(ppc, .90), TRUE, FALSE)) -> k1  
  
str(k1)

## Classes 'tbl\_df', 'tbl' and 'data.frame': 53940 obs. of 12 variables:  
## $ carat : num 0.23 0.21 0.23 0.29 0.31 0.24 0.24 0.26 0.22 0.23 ...  
## $ cut : Ord.factor w/ 5 levels "Fair"<"Good"<..: 5 4 2 4 2 3 3 3 1 3 ...  
## $ color : Ord.factor w/ 7 levels "D"<"E"<"F"<"G"<..: 2 2 2 6 7 7 6 5 2 5 ...  
## $ clarity: Ord.factor w/ 8 levels "I1"<"SI2"<"SI1"<..: 2 3 5 4 2 6 7 3 4 5 ...  
## $ depth : num 61.5 59.8 56.9 62.4 63.3 62.8 62.3 61.9 65.1 59.4 ...  
## $ table : num 55 61 65 58 58 57 57 55 61 61 ...  
## $ price : int 326 326 327 334 335 336 336 337 337 338 ...  
## $ x : num 3.95 3.89 4.05 4.2 4.34 3.94 3.95 4.07 3.87 4 ...  
## $ y : num 3.98 3.84 4.07 4.23 4.35 3.96 3.98 4.11 3.78 4.05 ...  
## $ z : num 2.43 2.31 2.31 2.63 2.75 2.48 2.47 2.53 2.49 2.39 ...  
## $ ppc : num 1417 1552 1422 1152 1081 ...  
## $ HiValue: logi FALSE FALSE FALSE FALSE FALSE FALSE ...

table(k1$HiValue)

##   
## FALSE TRUE   
## 48546 5394

## Problem 2

Create a logistic regression model, ppc1, with HiValue as the dependent variable. Display a summary of this model.

ppc1 = glm(HiValue ~ color + clarity + depth, data = k1, family="binomial")  
summary(ppc1)

##   
## Call:  
## glm(formula = HiValue ~ color + clarity + depth, family = "binomial",   
## data = k1)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -0.9199 -0.5229 -0.4168 -0.3289 2.8416   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) 0.09803 10.91646 0.009 0.992835   
## color.L -0.62344 0.07141 -8.730 < 2e-16 \*\*\*  
## color.Q -0.89242 0.06771 -13.180 < 2e-16 \*\*\*  
## color.C -0.34424 0.05794 -5.941 2.83e-09 \*\*\*  
## color^4 -0.01182 0.04654 -0.254 0.799536   
## color^5 -0.41706 0.04035 -10.337 < 2e-16 \*\*\*  
## color^6 -0.12489 0.03348 -3.730 0.000191 \*\*\*  
## clarity.L 8.35911 47.07808 0.178 0.859070   
## clarity.Q -7.84906 47.07807 -0.167 0.867587   
## clarity.C 5.56875 37.55532 0.148 0.882121   
## clarity^4 -3.53371 24.58575 -0.144 0.885714   
## clarity^5 2.38539 13.05717 0.183 0.855043   
## clarity^6 -0.75843 5.36518 -0.141 0.887585   
## clarity^7 0.33627 1.48836 0.226 0.821251   
## depth -0.06826 0.01070 -6.377 1.81e-10 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 35070 on 53939 degrees of freedom  
## Residual deviance: 33567 on 53925 degrees of freedom  
## AIC: 33597  
##   
## Number of Fisher Scoring iterations: 15

## Problem 3

Create and display the accuracy rate and confusion matrix based on a threshold of .5.

Probppc1 = predict(ppc1, type= "response")  
 Predppc1 = Probppc1 > .5  
   
 table(Predppc1, k1$HiValue)

##   
## Predppc1 FALSE TRUE  
## FALSE 48546 5394

#Accuracy Rate  
 AccRate = mean(Predppc1 == k1$HiValue)  
 AccRate

## [1] 0.9

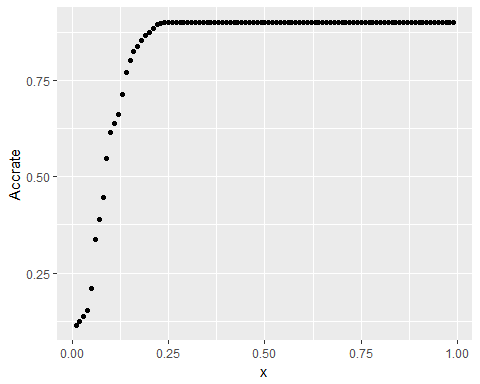
## Problem 4

Create the ROC curve for your model

fAccrate = function(th){  
 Predppc1 = Probppc1 > th  
 return(mean(Predppc1 == k1$HiValue))  
}  
x = seq(from = .01,to=.99,by = .01)  
Accrate = sapply(x,fAccrate)  
Accrate

## [1] 0.1137375 0.1257879 0.1376344 0.1523545 0.2106415 0.3364294 0.3888024  
## [8] 0.4450871 0.5469596 0.6149425 0.6370782 0.6621431 0.7129403 0.7701149  
## [15] 0.8009455 0.8239340 0.8383574 0.8528550 0.8673155 0.8752503 0.8848535  
## [22] 0.8951057 0.8984798 0.8994067 0.8995921 0.8998146 0.8999815 0.8999815  
## [29] 0.8999815 0.8999815 0.8999815 0.8999815 0.8999815 0.8999815 0.9000000  
## [36] 0.9000000 0.9000000 0.9000000 0.9000000 0.9000000 0.9000000 0.9000000  
## [43] 0.9000000 0.9000000 0.9000000 0.9000000 0.9000000 0.9000000 0.9000000  
## [50] 0.9000000 0.9000000 0.9000000 0.9000000 0.9000000 0.9000000 0.9000000  
## [57] 0.9000000 0.9000000 0.9000000 0.9000000 0.9000000 0.9000000 0.9000000  
## [64] 0.9000000 0.9000000 0.9000000 0.9000000 0.9000000 0.9000000 0.9000000  
## [71] 0.9000000 0.9000000 0.9000000 0.9000000 0.9000000 0.9000000 0.9000000  
## [78] 0.9000000 0.9000000 0.9000000 0.9000000 0.9000000 0.9000000 0.9000000  
## [85] 0.9000000 0.9000000 0.9000000 0.9000000 0.9000000 0.9000000 0.9000000  
## [92] 0.9000000 0.9000000 0.9000000 0.9000000 0.9000000 0.9000000 0.9000000  
## [99] 0.9000000

result=data.frame(x,Accrate)  
result %>% ggplot(aes(x, Accrate)) + geom\_point()



ppc1 = glm(HiValue ~ ppc, data = k1, family="binomial")

## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

summary(ppc1)

##   
## Call:  
## glm(formula = HiValue ~ ppc, family = "binomial", data = k1)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -0.2658 0.0000 0.0000 0.0000 0.4092   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -64123.159 34049.299 -1.883 0.0597 .  
## ppc 9.182 4.875 1.883 0.0597 .  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 3.5070e+04 on 53939 degrees of freedom  
## Residual deviance: 3.0686e-01 on 53938 degrees of freedom  
## AIC: 4.3069  
##   
## Number of Fisher Scoring iterations: 25

ppc1\_prob <- predict(ppc1, data = k1, type = "response")  
ROC <- roc(k1$HiValue, ppc1\_prob)  
plot(ROC, col= "Red")

