Natan Alper 4/1/2020

Business Intelligence & Consumer Insights- Professor Kovtun

HW #6

**1 (a)**

> # Natan Alper - Hw 6

> # 1

> ci <- data.frame(cps\_income)

> # 1a

**2 qual vars:**

race and gender

**3 quant vars:**

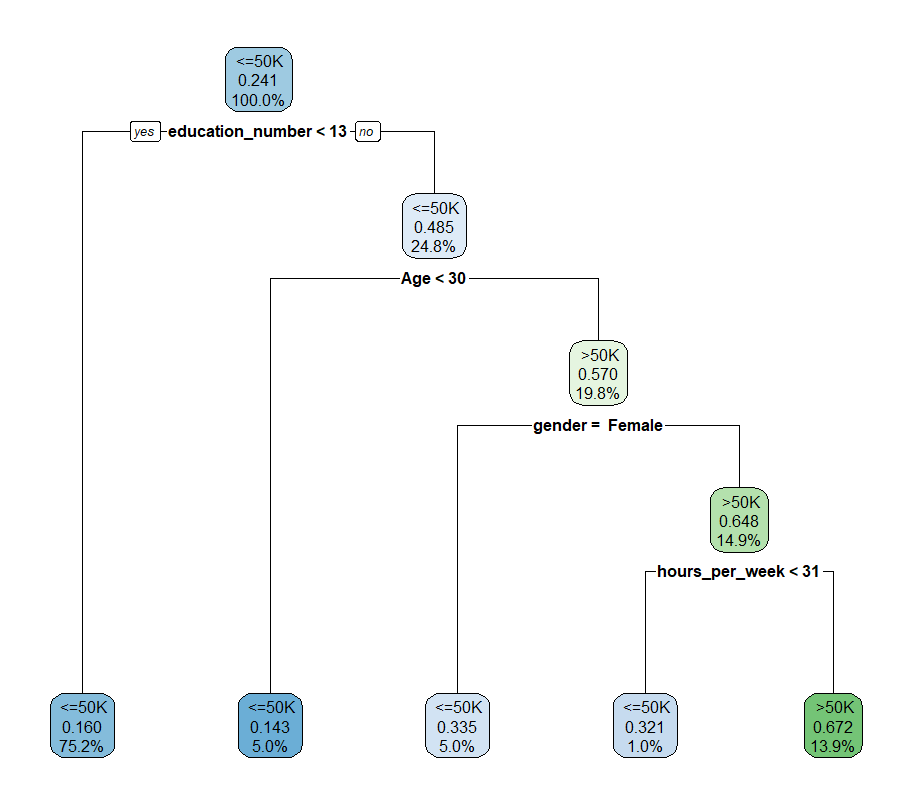
Age, education\_number, hours\_per\_week

> library(rpart)

> library(rpart.plot)

> ciTree <- rpart(Income~race+gender+Age+education\_number+hours\_per\_week, data=ci)

> rpart.plot(ciTree,digits=-3)



**1 (b)**

> # 1b

> # Find better cp

> ciTree <- rpart(Income~race+gender+Age+education\_number+hours\_per\_week, data=ci, control=rpart.control(cp=.0001))

> printcp(ciTree,digits=5)

Classification tree:

rpart(formula = Income ~ race + gender + Age + education\_number +

hours\_per\_week, data = ci, control = rpart.control(cp = 1e-04))

Variables actually used in tree construction:

[1] Age education\_number gender

[4] hours\_per\_week race

Root node error: 7840/32560 = 0.24079

n= 32560

CP nsplit rel error xerror xstd

1 0.05765306 0 1.00000 1.00000 0.0098407

2 0.01492347 3 0.81684 0.81862 0.0091561

3 0.00263605 4 0.80191 0.80344 0.0090915

4 0.00197704 7 0.79401 0.80064 0.0090794

5 0.00140306 9 0.79005 0.79630 0.0090606

6 0.00114796 10 0.78865 0.79770 0.0090667

7 0.00094752 11 0.78750 0.79681 0.0090628

8 0.00076531 19 0.77972 0.79847 0.0090700

9 0.00063776 20 0.77895 0.79847 0.0090700

10 0.00055272 23 0.77704 0.80038 0.0090783

11 0.00051020 29 0.77372 0.80383 0.0090931

12 0.00045918 30 0.77321 0.80383 0.0090931

13 0.00038265 36 0.77028 0.80383 0.0090931

14 0.00034014 40 0.76875 0.80293 0.0090893

15 0.00031888 46 0.76633 0.80281 0.0090887

16 0.00030612 66 0.75804 0.80370 0.0090925

17 0.00025510 75 0.75472 0.80702 0.0091068

18 0.00022959 96 0.74911 0.80778 0.0091101

19 0.00022321 118 0.74120 0.80982 0.0091188

20 0.00021259 123 0.74005 0.80982 0.0091188

21 0.00019133 132 0.73801 0.80982 0.0091188

22 0.00017007 150 0.73431 0.81263 0.0091307

23 0.00015944 174 0.72946 0.81607 0.0091453

24 0.00014349 183 0.72793 0.81658 0.0091475

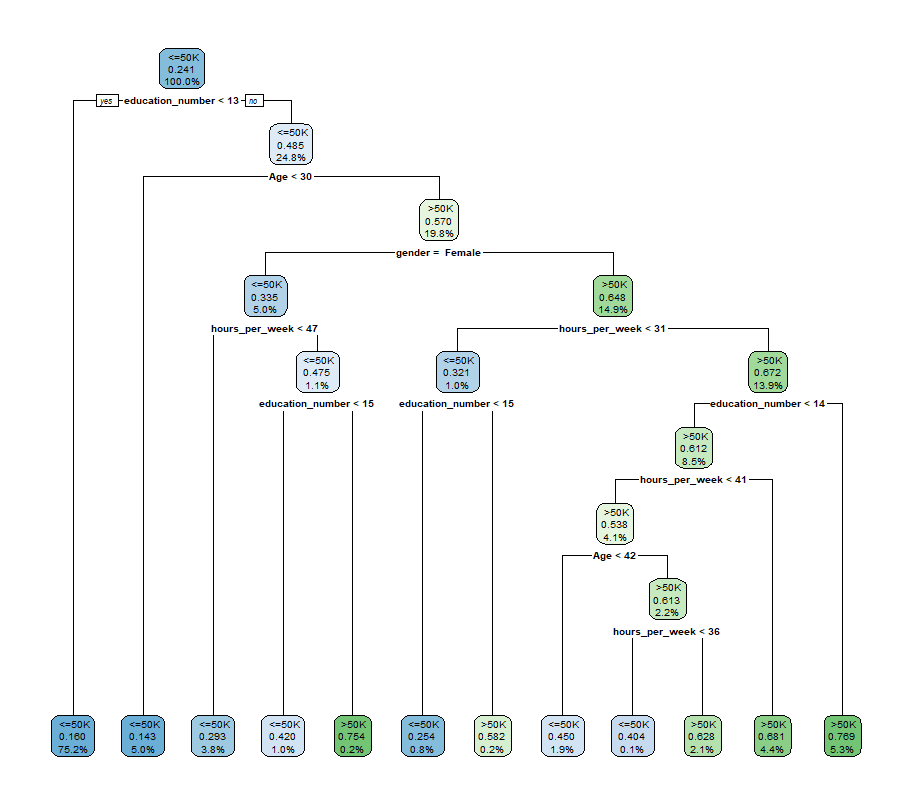
25 0.00012755 191 0.72679 0.82028 0.0091631

26 0.00010629 258 0.71811 0.82232 0.0091717

27 0.00010000 264 0.71747 0.82526 0.0091840

> ciTree <- rpart(Income~race+gender+Age+education\_number+hours\_per\_week, data=ci, control=rpart.control(cp=.0012))

> rpart.plot(ciTree,digits=-3)



**1 (c)**

> # 1c

> predict(ciTree,type="prob")[1,]

<=50K >50K

0.5501618 0.4498382

> # Approximately a 45% chance that the first individual earns >50K

**1 (d)**

> # 1d

> predictions <- predict(ciTree,type="class")

> predictions[1]

1

<=50K

Levels: <=50K >50K

**1 (e)**

> # 1e

> # 32560 total entries

> ### Only run “randSamp” once

> randSamp <- sample((1:nrow(ci)), size = 30000)

> training <- ci[randSamp, ]

> testing <- ci[-randSamp, ]

> trainTree <- rpart(Income~race+gender+Age+education\_number+hours\_per\_week, data=training)

> # Pred using training data tree

> predictions <- predict(trainTree,type="class", newdata = testing)

> # Misclassification rate of when comparing to testing data

> sum(testing$Income!=predictions)/nrow(testing)

[1] 0.203125

**1 (f)**

> # 1f

> tenthOfdata <- nrow(ci)/10

> g1 <- 1:tenthOfdata

> g2 <- ((tenthOfdata)+1):(2\*tenthOfdata)

> g3 <- ((2\*tenthOfdata)+1):(3\*tenthOfdata)

> g4 <- ((3\*tenthOfdata)+1):(4\*tenthOfdata)

> g5 <- ((4\*tenthOfdata)+1):(5\*tenthOfdata)

> g6 <- ((5\*tenthOfdata)+1):(6\*tenthOfdata)

> g7 <- ((6\*tenthOfdata)+1):(7\*tenthOfdata)

> g8 <- ((7\*tenthOfdata)+1):(8\*tenthOfdata)

> g9 <- ((8\*tenthOfdata)+1):(9\*tenthOfdata)

> g10 <- ((9\*tenthOfdata)+1):(10\*tenthOfdata)

> Groups <- data.frame(g1,g2,g3,g4,g5,g6,g7,g8,g9,g10)

> predictions <- c()

> for(i in 1:10){

+ ciTree <- rpart(Income~., data=ci[-Groups[,i],],control=rpart.control(cp=.001))

+ predictions\_per\_fold <- predict(ciTree,type="class",newdata=ci[Groups[,i],])

+ predictions <- c(predictions,as.character(predictions\_per\_fold))

+ }

> sum(ci$Income!=predictions)/length(predictions)

[1] 0.139312

**2**

> # 2

> # Select only quant columns

> ciQuant <- ci[, c(1,3,5,11,12,13,15)]

> # Standardize data

> for(i in 1:(ncol(ciQuant)-1)){ ## Note that response variable is assumed to be the last column here

+ ciQuant[,i] <- (ciQuant[,i]-mean(ciQuant[,i]))/sd(ciQuant[,i])

+ }

> train1 <- ciQuant[randSamp, -7]

> test1 <- ciQuant[-randSamp, -7]

> ##Pred contains predictions

> Pred <- knn(train=train1,test=test1,cl=ciQuant[randSamp,7],k=20) ##Note that we don't include the INCOME column in test or train

> Pred

[1] <=50K <=50K >50K <=50K <=50K <=50K <=50K <=50K <=50K

[10] <=50K <=50K >50K >50K <=50K <=50K >50K <=50K <=50K

[19] <=50K <=50K <=50K >50K >50K <=50K <=50K <=50K <=50K

[28] <=50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K

[37] <=50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K

[46] <=50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K

[55] <=50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K

[64] <=50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K

[73] <=50K <=50K >50K <=50K <=50K <=50K <=50K >50K <=50K

[82] <=50K <=50K >50K >50K <=50K >50K >50K <=50K <=50K

[91] >50K <=50K <=50K <=50K <=50K <=50K >50K <=50K <=50K

[100] <=50K <=50K <=50K <=50K >50K <=50K <=50K <=50K <=50K

[109] <=50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K >50K

[118] <=50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K

[127] <=50K <=50K >50K <=50K <=50K <=50K <=50K <=50K <=50K

[136] <=50K <=50K <=50K >50K >50K <=50K <=50K <=50K >50K

[145] <=50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K

[154] <=50K <=50K <=50K <=50K <=50K >50K <=50K <=50K >50K

[163] <=50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K

[172] <=50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K >50K

[181] <=50K <=50K >50K <=50K <=50K <=50K <=50K <=50K <=50K

[190] <=50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K

[199] <=50K <=50K >50K >50K <=50K <=50K <=50K <=50K <=50K

[208] <=50K >50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K

[217] <=50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K >50K

[226] <=50K >50K <=50K >50K <=50K <=50K <=50K <=50K <=50K

[235] <=50K <=50K <=50K <=50K <=50K >50K <=50K >50K <=50K

[244] <=50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K

[253] <=50K <=50K <=50K >50K <=50K >50K <=50K >50K <=50K

[262] <=50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K >50K

[271] <=50K <=50K <=50K <=50K >50K <=50K <=50K >50K <=50K

[280] <=50K <=50K <=50K >50K <=50K <=50K <=50K <=50K >50K

[289] <=50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K

[298] <=50K <=50K <=50K >50K <=50K <=50K <=50K <=50K <=50K

[307] <=50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K

[316] <=50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K >50K

[325] <=50K <=50K >50K <=50K <=50K <=50K <=50K <=50K <=50K

[334] >50K <=50K <=50K <=50K <=50K <=50K <=50K >50K <=50K

[343] >50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K

[352] <=50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K

[361] <=50K >50K <=50K >50K >50K <=50K >50K <=50K <=50K

[370] <=50K <=50K <=50K >50K <=50K >50K <=50K >50K <=50K

[379] <=50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K >50K

[388] <=50K <=50K >50K <=50K <=50K >50K <=50K <=50K <=50K

[397] <=50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K

[406] <=50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K >50K

[415] <=50K <=50K <=50K <=50K <=50K <=50K <=50K >50K <=50K

[424] >50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K

[433] >50K <=50K <=50K <=50K <=50K <=50K <=50K >50K <=50K

[442] <=50K <=50K <=50K >50K <=50K <=50K <=50K >50K <=50K

[451] <=50K >50K >50K <=50K <=50K <=50K >50K <=50K <=50K

[460] <=50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K

[469] <=50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K

[478] <=50K <=50K <=50K <=50K >50K >50K <=50K <=50K <=50K

[487] <=50K >50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K

[496] <=50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K

[505] >50K >50K <=50K <=50K <=50K >50K <=50K <=50K <=50K

[514] <=50K <=50K <=50K >50K >50K <=50K <=50K <=50K <=50K

[523] <=50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K

[532] >50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K

[541] <=50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K

[550] <=50K <=50K <=50K <=50K >50K <=50K <=50K <=50K <=50K

[559] <=50K >50K >50K <=50K <=50K <=50K <=50K >50K <=50K

[568] <=50K <=50K <=50K >50K >50K <=50K <=50K <=50K <=50K

[577] >50K <=50K <=50K <=50K <=50K >50K <=50K <=50K >50K

[586] <=50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K >50K

[595] <=50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K

[604] >50K <=50K <=50K <=50K <=50K <=50K <=50K >50K <=50K

[613] <=50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K

[622] <=50K <=50K <=50K <=50K <=50K >50K <=50K <=50K <=50K

[631] <=50K <=50K <=50K <=50K <=50K <=50K >50K <=50K <=50K

[640] <=50K <=50K >50K >50K <=50K <=50K <=50K <=50K <=50K

[649] <=50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K

[658] <=50K >50K <=50K <=50K <=50K >50K <=50K <=50K >50K

[667] <=50K >50K <=50K <=50K >50K <=50K <=50K <=50K <=50K

[676] <=50K <=50K <=50K >50K <=50K <=50K <=50K <=50K <=50K

[685] <=50K <=50K >50K <=50K <=50K <=50K <=50K <=50K <=50K

[694] <=50K >50K <=50K <=50K <=50K >50K >50K <=50K <=50K

[703] <=50K <=50K >50K <=50K <=50K <=50K <=50K <=50K <=50K

[712] <=50K >50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K

[721] <=50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K

[730] <=50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K

[739] <=50K <=50K <=50K <=50K <=50K <=50K <=50K >50K <=50K

[748] <=50K >50K <=50K <=50K >50K >50K <=50K <=50K >50K

[757] <=50K <=50K <=50K >50K <=50K <=50K <=50K <=50K <=50K

[766] >50K <=50K >50K <=50K <=50K <=50K <=50K <=50K >50K

[775] <=50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K

[784] <=50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K

[793] >50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K

[802] <=50K >50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K

[811] >50K <=50K <=50K >50K <=50K <=50K >50K <=50K >50K

[820] >50K >50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K

[829] <=50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K

[838] <=50K >50K <=50K <=50K <=50K <=50K <=50K <=50K >50K

[847] <=50K <=50K <=50K >50K <=50K <=50K <=50K <=50K <=50K

[856] <=50K <=50K <=50K <=50K <=50K >50K <=50K <=50K <=50K

[865] <=50K <=50K <=50K <=50K <=50K <=50K <=50K >50K <=50K

[874] <=50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K >50K

[883] <=50K <=50K <=50K <=50K <=50K <=50K >50K <=50K <=50K

[892] >50K <=50K >50K <=50K <=50K <=50K <=50K >50K <=50K

[901] <=50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K

[910] <=50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K

[919] <=50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K

[928] <=50K <=50K <=50K <=50K <=50K <=50K <=50K >50K <=50K

[937] <=50K <=50K <=50K <=50K >50K <=50K <=50K <=50K <=50K

[946] <=50K >50K <=50K <=50K <=50K <=50K <=50K >50K <=50K

[955] <=50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K

[964] <=50K <=50K <=50K <=50K <=50K <=50K <=50K >50K <=50K

[973] <=50K >50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K

[982] <=50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K

[991] <=50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K <=50K

[1000] >50K

[ reached getOption("max.print") -- omitted 1560 entries ]

Levels: <=50K >50K

> ##Let's compare to the true values to compute misclassification rate:

> misclass <- sum(ciQuant[-randSamp, 7]!=Pred)/length(Pred)

> misclass

[1] 0.1824219

**3**

> # 3

> misclass <- c()

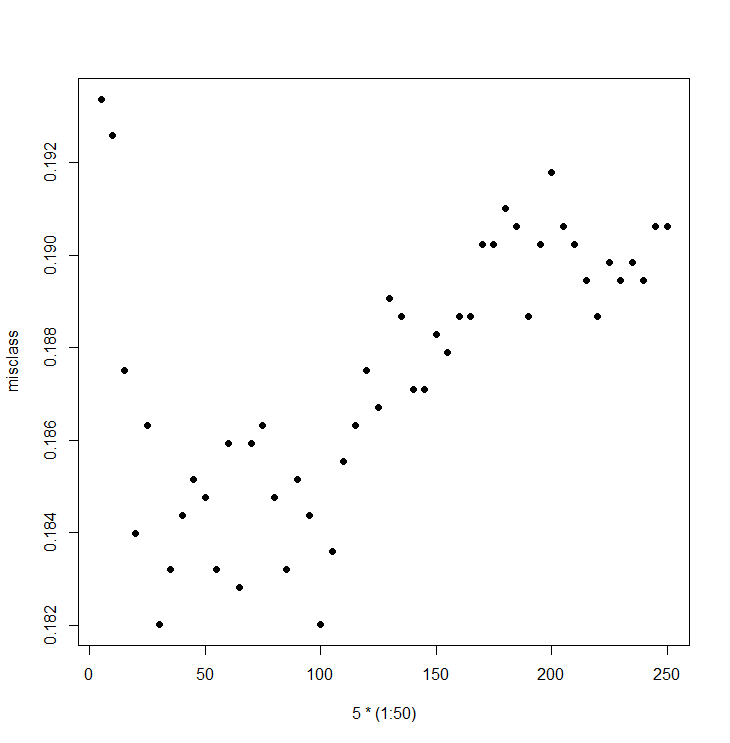
> for(k in 5\*(1:50)){

+ Pred <- knn(train=train1,test=test1,cl=ciQuant[randSamp,7],k=k)

+ misclass <- c(misclass, sum(ciQuant[-randSamp, 7]!=Pred)/length(Pred))

+ }

> plot(5\*(1:50), misclass,pch=16) ## We are trying to find a choice of "k" which would lead to the lowest misclassification rate.



According to the plot, k=35 is when misclassification rate seems lowest.