homework 6

library(nnet)  
library(MASS)  
library(kknn)  
cancer<- read.table("http://archive.ics.uci.edu/ml/machine-learning-databases/breast-cancer-wisconsin/breast-cancer-wisconsin.data",sep = ",", stringsAsFactors = FALSE, header=F)  
head(cancer)

## V1 V2 V3 V4 V5 V6 V7 V8 V9 V10 V11  
## 1 1000025 5 1 1 1 2 1 3 1 1 2  
## 2 1002945 5 4 4 5 7 10 3 2 1 2  
## 3 1015425 3 1 1 1 2 2 3 1 1 2  
## 4 1016277 6 8 8 1 3 4 3 7 1 2  
## 5 1017023 4 1 1 3 2 1 3 1 1 2  
## 6 1017122 8 10 10 8 7 10 9 7 1 4

# check for the missings

str(cancer)

## 'data.frame': 699 obs. of 11 variables:  
## $ V1 : int 1000025 1002945 1015425 1016277 1017023 1017122 1018099 1018561 1033078 1033078 ...  
## $ V2 : int 5 5 3 6 4 8 1 2 2 4 ...  
## $ V3 : int 1 4 1 8 1 10 1 1 1 2 ...  
## $ V4 : int 1 4 1 8 1 10 1 2 1 1 ...  
## $ V5 : int 1 5 1 1 3 8 1 1 1 1 ...  
## $ V6 : int 2 7 2 3 2 7 2 2 2 2 ...  
## $ V7 : chr "1" "10" "2" "4" ...  
## $ V8 : int 3 3 3 3 3 9 3 3 1 2 ...  
## $ V9 : int 1 2 1 7 1 7 1 1 1 1 ...  
## $ V10: int 1 1 1 1 1 1 1 1 5 1 ...  
## $ V11: int 2 2 2 2 2 4 2 2 2 2 ...

# We noticed that V1-V11 are all integer values, except V7 has missing data, marked as “?”

# check how many data are missing in V7

mis<-subset(cancer,cancer$V7=="?")  
nrow(mis)

## [1] 16

# 16 obs were missing, which account for 16/699=2.29% of the total data. We can go ahead and impute values for the missings.

# 1. Use the mean/mode imputation method to impute values for the missing data.

# Find the mode value

v1<-nrow(subset(cancer,cancer$V7==1))  
v2<-nrow(subset(cancer,cancer$V7==2))  
v3<-nrow(subset(cancer,cancer$V7==3))  
v4<-nrow(subset(cancer,cancer$V7==4))  
v5<-nrow(subset(cancer,cancer$V7==5))  
v6<-nrow(subset(cancer,cancer$V7==6))  
v7<-nrow(subset(cancer,cancer$V7==7))  
v8<-nrow(subset(cancer,cancer$V7==8))  
v9<-nrow(subset(cancer,cancer$V7==9))  
v10<-nrow(subset(cancer,cancer$V7==10))  
v<-c( v1 , v2 , v3 , v4 , v5 , v6 , v7 , v8 , v9 , v10 )  
  
mode<-which.max(v)  
mode

## [1] 1

# 1 is the mode

# Assign mode value to the missings

cancer1<-cancer  
cancer1$V7[cancer1$V7=="?"]<-mode  
sum(cancer1$V7=="?")#We have sucessfully changed "?" to 1

## [1] 0

cancer1$V7<-as.integer(cancer1$V7)  
str(cancer1$V7)

## int [1:699] 1 10 2 4 1 10 10 1 1 1 ...

# 2. Use regression to impute values for the missing data.

# Leave out the response variables and V1 which is ID, and use stepwise method to predict the V7 with all the other variables

cancer2<-cancer[cancer$V7!="?",2:10]  
cancer2$V7 <- as.integer(cancer2$V7)

# 70% for training

mask\_train<-sample(nrow(cancer2), size = floor(nrow(cancer2) \* 0.7))

# training data set

train<-cancer2[mask\_train,]

# Using the remaining data for test

test<-cancer2[-mask\_train, ] # all rows except training

# Fit the model

reg<- multinom(V7 ~ ., data = train)

## # weights: 100 (81 variable)  
## initial value 1100.635674   
## iter 10 value 738.506697  
## iter 20 value 546.160554  
## iter 30 value 479.243308  
## iter 40 value 432.340956  
## iter 50 value 424.368253  
## iter 60 value 423.160896  
## iter 70 value 422.578859  
## iter 80 value 422.074623  
## iter 90 value 421.298496  
## iter 100 value 420.270150  
## final value 420.270150   
## stopped after 100 iterations

summary(reg)

## Call:  
## multinom(formula = V7 ~ ., data = train)  
##   
## Coefficients:  
## (Intercept) V2 V3 V4 V5 V6  
## 2 -4.717949 0.15570995 0.12979546 -0.30938051 -0.04399826 0.05644216  
## 3 -3.800999 -0.03723532 0.21116166 0.05390461 0.21550949 -0.05180473  
## 4 -5.753214 0.21279724 0.37648965 0.02305562 -0.05610244 -0.12010129  
## 5 -4.944981 0.18300893 -0.03782338 0.27806962 0.26608868 0.05526566  
## 6 -24.646647 -1.64196772 1.02195343 3.75165874 -1.81272892 -4.18796189  
## 7 -8.215568 0.63027705 -0.75256975 0.72682523 0.32657153 -0.92634166  
## 8 -7.309866 0.05760286 0.06672619 0.32941415 0.30333228 0.34938933  
## 9 -7.060588 -0.10358529 0.12739754 0.27452567 -0.02637578 -0.33538970  
## 10 -6.454512 0.31794210 -0.09753105 0.31544033 0.40304005 0.11732450  
## V8 V9 V10  
## 2 0.41234919 0.048187886 0.40044583  
## 3 -0.01566152 0.077776199 0.27493870  
## 4 0.11368783 0.149248626 0.12758427  
## 5 0.14886235 0.136433546 -0.62391550  
## 6 -7.58742416 3.527511651 3.83488941  
## 7 0.15229462 0.370226768 0.22533082  
## 8 0.25222710 -0.007207503 -0.37493148  
## 9 0.44430271 0.211937012 0.53460981  
## 10 0.30522100 0.012094298 0.07435297  
##   
## Std. Errors:  
## (Intercept) V2 V3 V4 V5 V6  
## 2 0.6055716 0.11705757 0.2051448 0.2145342 0.16917444 0.1931160  
## 3 0.5790774 0.11598304 0.1920415 0.1910212 0.12829261 0.1973117  
## 4 0.9164978 0.15165487 0.2078790 0.2036061 0.18925469 0.2208576  
## 5 0.7776761 0.11516890 0.1957008 0.2069221 0.12151056 0.1704099  
## 6 16.5881518 11.53426761 9.8912871 21.1457943 32.39631601 13.2292612  
## 7 2.2080246 0.27591292 0.4714759 0.4266245 0.21225069 0.6549271  
## 8 1.0864427 0.13921935 0.2154042 0.2247189 0.13152735 0.1712012  
## 9 1.4563364 0.23652054 0.3367735 0.3410405 0.24003063 0.3755600  
## 10 0.6161176 0.08704563 0.1469826 0.1519413 0.09454802 0.1287049  
## V8 V9 V10  
## 2 0.1596219 0.13066645 0.1631515  
## 3 0.1710745 0.12445152 0.1682904  
## 4 0.1904857 0.12947967 0.2313830  
## 5 0.1566109 0.11040643 0.4188275  
## 6 28.5317700 11.00520741 7.5821159  
## 7 0.3002198 0.23097065 0.2869629  
## 8 0.1718295 0.11825923 0.3041753  
## 9 0.2768113 0.21178338 0.2254439  
## 10 0.1175911 0.08551794 0.1546946  
##   
## Residual Deviance: 840.5403   
## AIC: 1002.54

# Use stepwise method to re-fit the model with all the predictors

stp<-stepAIC(reg, direction="both")

## Start: AIC=1002.54  
## V7 ~ V2 + V3 + V4 + V5 + V6 + V8 + V9 + V10  
##   
## # weights: 90 (72 variable)  
## initial value 1100.635674   
## iter 10 value 711.133497  
## iter 20 value 577.660674  
## iter 30 value 472.754989  
## iter 40 value 440.705768  
## iter 50 value 437.603072  
## iter 60 value 436.461167  
## iter 70 value 435.632302  
## iter 80 value 433.797799  
## iter 90 value 432.742701  
## iter 100 value 432.707697  
## final value 432.707697   
## stopped after 100 iterations  
## # weights: 90 (72 variable)  
## initial value 1100.635674   
## iter 10 value 754.723632  
## iter 20 value 555.837734  
## iter 30 value 485.210782  
## iter 40 value 439.381985  
## iter 50 value 431.504242  
## iter 60 value 430.220665  
## iter 70 value 429.533102  
## iter 80 value 427.836019  
## iter 90 value 426.715800  
## iter 100 value 426.652167  
## final value 426.652167   
## stopped after 100 iterations  
## # weights: 90 (72 variable)  
## initial value 1100.635674   
## iter 10 value 767.215576  
## iter 20 value 571.430176  
## iter 30 value 483.392606  
## iter 40 value 442.443155  
## iter 50 value 434.721973  
## iter 60 value 433.141720  
## iter 70 value 432.069675  
## iter 80 value 429.716580  
## iter 90 value 428.683587  
## iter 100 value 428.631787  
## final value 428.631787   
## stopped after 100 iterations  
## # weights: 90 (72 variable)  
## initial value 1100.635674   
## iter 10 value 770.113546  
## iter 20 value 555.735675  
## iter 30 value 479.787499  
## iter 40 value 448.096723  
## iter 50 value 442.682360  
## iter 60 value 440.974714  
## iter 70 value 439.943898  
## iter 80 value 438.745417  
## iter 90 value 436.863681  
## iter 100 value 436.493098  
## final value 436.493098   
## stopped after 100 iterations  
## # weights: 90 (72 variable)  
## initial value 1100.635674   
## iter 10 value 768.501946  
## iter 20 value 566.709912  
## iter 30 value 463.407626  
## iter 40 value 438.650980  
## iter 50 value 432.046967  
## iter 60 value 430.197857  
## iter 70 value 429.446042  
## iter 80 value 428.589506  
## iter 90 value 426.826719  
## iter 100 value 426.544463  
## final value 426.544463   
## stopped after 100 iterations  
## # weights: 90 (72 variable)  
## initial value 1100.635674   
## iter 10 value 772.658742  
## iter 20 value 558.872371  
## iter 30 value 480.875893  
## iter 40 value 441.445124  
## iter 50 value 434.000155  
## iter 60 value 432.006565  
## iter 70 value 430.988913  
## iter 80 value 429.336369  
## iter 90 value 427.192383  
## iter 100 value 426.964997  
## final value 426.964997   
## stopped after 100 iterations  
## # weights: 90 (72 variable)  
## initial value 1100.635674   
## iter 10 value 766.084586  
## iter 20 value 551.491595  
## iter 30 value 463.947794  
## iter 40 value 433.369779  
## iter 50 value 429.142605  
## iter 60 value 427.648176  
## iter 70 value 426.646912  
## iter 80 value 425.637509  
## iter 90 value 423.705907  
## iter 100 value 423.551213  
## final value 423.551213   
## stopped after 100 iterations  
## # weights: 90 (72 variable)  
## initial value 1100.635674   
## iter 10 value 757.843432  
## iter 20 value 556.558436  
## iter 30 value 461.930058  
## iter 40 value 443.168883  
## iter 50 value 438.423208  
## iter 60 value 436.822757  
## iter 70 value 436.172147  
## iter 80 value 435.186890  
## iter 90 value 433.464828  
## iter 100 value 432.791943  
## final value 432.791943   
## stopped after 100 iterations  
## Df AIC  
## - V9 9 991.10  
## - V6 9 997.09  
## - V3 9 997.30  
## - V8 9 997.93  
## - V4 9 1001.26  
## <none> 1002.54  
## - V2 9 1009.42  
## - V10 9 1009.58  
## - V5 9 1016.99  
## # weights: 90 (72 variable)  
## initial value 1100.635674   
## iter 10 value 766.084586  
## iter 20 value 551.491595  
## iter 30 value 463.947794  
## iter 40 value 433.369779  
## iter 50 value 429.142605  
## iter 60 value 427.648176  
## iter 70 value 426.646912  
## iter 80 value 425.637509  
## iter 90 value 423.705907  
## iter 100 value 423.551213  
## final value 423.551213   
## stopped after 100 iterations  
##   
## Step: AIC=991.1  
## V7 ~ V2 + V3 + V4 + V5 + V6 + V8 + V10  
##   
## # weights: 80 (63 variable)  
## initial value 1100.635674   
## iter 10 value 645.393352  
## iter 20 value 503.722363  
## iter 30 value 452.873649  
## iter 40 value 442.076596  
## iter 50 value 439.822123  
## iter 60 value 439.106112  
## iter 70 value 438.667499  
## iter 80 value 437.582767  
## iter 90 value 435.973337  
## iter 100 value 435.941920  
## final value 435.941920   
## stopped after 100 iterations  
## # weights: 80 (63 variable)  
## initial value 1100.635674   
## iter 10 value 779.391272  
## iter 20 value 598.633317  
## iter 30 value 474.555967  
## iter 40 value 442.636396  
## iter 50 value 437.863560  
## iter 60 value 435.790308  
## iter 70 value 434.125422  
## iter 80 value 431.327159  
## iter 90 value 430.787708  
## iter 100 value 430.767366  
## final value 430.767366   
## stopped after 100 iterations  
## # weights: 80 (63 variable)  
## initial value 1100.635674   
## iter 10 value 768.286874  
## iter 20 value 594.300728  
## iter 30 value 471.775100  
## iter 40 value 445.483611  
## iter 50 value 440.926453  
## iter 60 value 438.933196  
## iter 70 value 436.291649  
## iter 80 value 432.938418  
## iter 90 value 432.846256  
## iter 100 value 432.844519  
## final value 432.844519   
## stopped after 100 iterations  
## # weights: 80 (63 variable)  
## initial value 1100.635674   
## iter 10 value 800.313023  
## iter 20 value 607.431893  
## iter 30 value 488.989526  
## iter 40 value 453.649611  
## iter 50 value 447.349172  
## iter 60 value 444.964993  
## iter 70 value 443.301212  
## iter 80 value 442.011230  
## iter 90 value 439.727743  
## iter 100 value 439.542452  
## final value 439.542452   
## stopped after 100 iterations  
## # weights: 80 (63 variable)  
## initial value 1100.635674   
## iter 10 value 801.227472  
## iter 20 value 549.097135  
## iter 30 value 454.978416  
## iter 40 value 437.190679  
## iter 50 value 434.178350  
## iter 60 value 432.457385  
## iter 70 value 431.341647  
## iter 80 value 429.094330  
## iter 90 value 428.684600  
## iter 100 value 428.681730  
## final value 428.681730   
## stopped after 100 iterations  
## # weights: 80 (63 variable)  
## initial value 1100.635674   
## iter 10 value 777.559014  
## iter 20 value 588.354765  
## iter 30 value 482.609825  
## iter 40 value 447.294777  
## iter 50 value 440.866629  
## iter 60 value 437.827430  
## iter 70 value 435.221972  
## iter 80 value 431.806711  
## iter 90 value 430.462819  
## iter 100 value 430.426656  
## final value 430.426656   
## stopped after 100 iterations  
## # weights: 80 (63 variable)  
## initial value 1100.635674   
## iter 10 value 772.734337  
## iter 20 value 545.227677  
## iter 30 value 472.708864  
## iter 40 value 444.473667  
## iter 50 value 441.221084  
## iter 60 value 440.500276  
## iter 70 value 439.874980  
## iter 80 value 438.169581  
## iter 90 value 436.082159  
## iter 100 value 435.981859  
## final value 435.981859   
## stopped after 100 iterations  
## # weights: 100 (81 variable)  
## initial value 1100.635674   
## iter 10 value 738.506697  
## iter 20 value 546.160554  
## iter 30 value 479.243308  
## iter 40 value 432.340956  
## iter 50 value 424.368253  
## iter 60 value 423.160896  
## iter 70 value 422.578859  
## iter 80 value 422.074623  
## iter 90 value 421.298496  
## iter 100 value 420.270150  
## final value 420.270150   
## stopped after 100 iterations  
## Df AIC  
## - V6 9 983.36  
## - V8 9 986.85  
## - V3 9 987.53  
## <none> 991.10  
## - V4 9 991.69  
## - V2 9 997.88  
## - V10 9 997.96  
## + V9 9 1002.54  
## - V5 9 1005.08  
## # weights: 80 (63 variable)  
## initial value 1100.635674   
## iter 10 value 801.227472  
## iter 20 value 549.097135  
## iter 30 value 454.978416  
## iter 40 value 437.190679  
## iter 50 value 434.178350  
## iter 60 value 432.457385  
## iter 70 value 431.341647  
## iter 80 value 429.094330  
## iter 90 value 428.684600  
## iter 100 value 428.681730  
## final value 428.681730   
## stopped after 100 iterations  
##   
## Step: AIC=983.36  
## V7 ~ V2 + V3 + V4 + V5 + V8 + V10  
##   
## # weights: 70 (54 variable)  
## initial value 1100.635674   
## iter 10 value 745.394110  
## iter 20 value 547.172360  
## iter 30 value 469.015006  
## iter 40 value 449.117503  
## iter 50 value 445.785775  
## iter 60 value 444.446731  
## iter 70 value 443.750818  
## iter 80 value 443.063326  
## iter 90 value 442.479799  
## iter 100 value 442.474599  
## final value 442.474599   
## stopped after 100 iterations  
## # weights: 70 (54 variable)  
## initial value 1100.635674   
## iter 10 value 666.723188  
## iter 20 value 514.901259  
## iter 30 value 459.238686  
## iter 40 value 443.370432  
## iter 50 value 440.129634  
## iter 60 value 439.102266  
## iter 70 value 437.226686  
## iter 80 value 436.151645  
## iter 90 value 436.143872  
## final value 436.143847   
## converged  
## # weights: 70 (54 variable)  
## initial value 1100.635674   
## iter 10 value 659.010539  
## iter 20 value 522.421538  
## iter 30 value 467.001661  
## iter 40 value 447.479713  
## iter 50 value 443.170541  
## iter 60 value 441.438564  
## iter 70 value 440.722126  
## iter 80 value 438.433673  
## iter 90 value 438.386889  
## final value 438.386396   
## converged  
## # weights: 70 (54 variable)  
## initial value 1100.635674   
## iter 10 value 657.569618  
## iter 20 value 540.338804  
## iter 30 value 463.572497  
## iter 40 value 453.609238  
## iter 50 value 450.794923  
## iter 60 value 449.957825  
## iter 70 value 449.598706  
## iter 80 value 449.450392  
## final value 449.449948   
## converged  
## # weights: 70 (54 variable)  
## initial value 1100.635674   
## iter 10 value 671.869937  
## iter 20 value 517.477864  
## iter 30 value 456.590869  
## iter 40 value 443.388162  
## iter 50 value 440.678146  
## iter 60 value 439.654394  
## iter 70 value 436.299906  
## iter 80 value 435.721397  
## iter 90 value 435.713822  
## final value 435.713805   
## converged  
## # weights: 70 (54 variable)  
## initial value 1100.635674   
## iter 10 value 790.806488  
## iter 20 value 567.383311  
## iter 30 value 471.211250  
## iter 40 value 453.229406  
## iter 50 value 447.425116  
## iter 60 value 444.578827  
## iter 70 value 442.462015  
## iter 80 value 440.103835  
## iter 90 value 439.829599  
## iter 100 value 439.827919  
## final value 439.827919   
## stopped after 100 iterations  
## # weights: 90 (72 variable)  
## initial value 1100.635674   
## iter 10 value 766.084586  
## iter 20 value 551.491595  
## iter 30 value 463.947794  
## iter 40 value 433.369779  
## iter 50 value 429.142605  
## iter 60 value 427.648176  
## iter 70 value 426.646912  
## iter 80 value 425.637509  
## iter 90 value 423.705907  
## iter 100 value 423.551213  
## final value 423.551213   
## stopped after 100 iterations  
## # weights: 90 (72 variable)  
## initial value 1100.635674   
## iter 10 value 768.501946  
## iter 20 value 566.709912  
## iter 30 value 463.407626  
## iter 40 value 438.650980  
## iter 50 value 432.046967  
## iter 60 value 430.197857  
## iter 70 value 429.446042  
## iter 80 value 428.589506  
## iter 90 value 426.826719  
## iter 100 value 426.544463  
## final value 426.544463   
## stopped after 100 iterations  
## Df AIC  
## - V8 9 979.43  
## - V3 9 980.29  
## <none> 983.36  
## - V4 9 984.77  
## - V10 9 987.66  
## + V6 9 991.10  
## - V2 9 992.95  
## + V9 9 997.09  
## - V5 9 1006.90  
## # weights: 70 (54 variable)  
## initial value 1100.635674   
## iter 10 value 671.869937  
## iter 20 value 517.477864  
## iter 30 value 456.590869  
## iter 40 value 443.388162  
## iter 50 value 440.678146  
## iter 60 value 439.654394  
## iter 70 value 436.299906  
## iter 80 value 435.721397  
## iter 90 value 435.713822  
## final value 435.713805   
## converged  
##   
## Step: AIC=979.43  
## V7 ~ V2 + V3 + V4 + V5 + V10  
##   
## # weights: 60 (45 variable)  
## initial value 1100.635674   
## iter 10 value 731.504396  
## iter 20 value 552.630066  
## iter 30 value 470.847206  
## iter 40 value 457.733917  
## iter 50 value 454.741519  
## iter 60 value 452.151398  
## iter 70 value 451.794922  
## iter 80 value 451.789699  
## final value 451.789688   
## converged  
## # weights: 60 (45 variable)  
## initial value 1100.635674   
## iter 10 value 660.129658  
## iter 20 value 507.630633  
## iter 30 value 461.520173  
## iter 40 value 450.939142  
## iter 50 value 448.214228  
## iter 60 value 445.901867  
## iter 70 value 443.573158  
## iter 80 value 443.542887  
## final value 443.542663   
## converged  
## # weights: 60 (45 variable)  
## initial value 1100.635674   
## iter 10 value 655.597300  
## iter 20 value 520.650290  
## iter 30 value 466.033752  
## iter 40 value 454.666241  
## iter 50 value 452.045817  
## iter 60 value 450.948409  
## iter 70 value 450.347114  
## iter 80 value 450.267082  
## iter 90 value 450.183407  
## final value 450.178632   
## converged  
## # weights: 60 (45 variable)  
## initial value 1100.635674   
## iter 10 value 666.644282  
## iter 20 value 550.742238  
## iter 30 value 487.954596  
## iter 40 value 468.063992  
## iter 50 value 464.994016  
## iter 60 value 463.881252  
## iter 70 value 463.753061  
## final value 463.752379   
## converged  
## # weights: 60 (45 variable)  
## initial value 1100.635674   
## iter 10 value 697.913823  
## iter 20 value 505.796360  
## iter 30 value 462.446364  
## iter 40 value 454.013114  
## iter 50 value 452.495566  
## iter 60 value 451.941305  
## iter 70 value 451.888402  
## final value 451.888372   
## converged  
## # weights: 80 (63 variable)  
## initial value 1100.635674   
## iter 10 value 777.559014  
## iter 20 value 588.354765  
## iter 30 value 482.609825  
## iter 40 value 447.294777  
## iter 50 value 440.866629  
## iter 60 value 437.827430  
## iter 70 value 435.221972  
## iter 80 value 431.806711  
## iter 90 value 430.462819  
## iter 100 value 430.426656  
## final value 430.426656   
## stopped after 100 iterations  
## # weights: 80 (63 variable)  
## initial value 1100.635674   
## iter 10 value 801.227472  
## iter 20 value 549.097135  
## iter 30 value 454.978416  
## iter 40 value 437.190679  
## iter 50 value 434.178350  
## iter 60 value 432.457385  
## iter 70 value 431.341647  
## iter 80 value 429.094330  
## iter 90 value 428.684600  
## iter 100 value 428.681730  
## final value 428.681730   
## stopped after 100 iterations  
## # weights: 80 (63 variable)  
## initial value 1100.635674   
## iter 10 value 798.636917  
## iter 20 value 581.741305  
## iter 30 value 476.224362  
## iter 40 value 447.168601  
## iter 50 value 441.876529  
## iter 60 value 439.026279  
## iter 70 value 436.709791  
## iter 80 value 433.936640  
## iter 90 value 433.316445  
## iter 100 value 433.311037  
## final value 433.311037   
## stopped after 100 iterations  
## Df AIC  
## - V3 9 977.09  
## <none> 979.43  
## + V8 9 983.36  
## + V6 9 986.85  
## - V4 9 990.36  
## + V9 9 992.62  
## - V2 9 993.58  
## - V10 9 993.78  
## - V5 9 1017.50  
## # weights: 60 (45 variable)  
## initial value 1100.635674   
## iter 10 value 660.129658  
## iter 20 value 507.630633  
## iter 30 value 461.520173  
## iter 40 value 450.939142  
## iter 50 value 448.214228  
## iter 60 value 445.901867  
## iter 70 value 443.573158  
## iter 80 value 443.542887  
## final value 443.542663   
## converged  
##   
## Step: AIC=977.09  
## V7 ~ V2 + V4 + V5 + V10  
##   
## # weights: 50 (36 variable)  
## initial value 1100.635674   
## iter 10 value 736.340060  
## iter 20 value 518.323893  
## iter 30 value 470.603145  
## iter 40 value 463.509312  
## iter 50 value 461.385626  
## iter 60 value 460.302541  
## iter 70 value 460.271839  
## final value 460.271699   
## converged  
## # weights: 50 (36 variable)  
## initial value 1100.635674   
## iter 10 value 674.157506  
## iter 20 value 526.278855  
## iter 30 value 484.607598  
## iter 40 value 472.238428  
## iter 50 value 471.618382  
## iter 60 value 471.543343  
## final value 471.543138   
## converged  
## # weights: 50 (36 variable)  
## initial value 1100.635674   
## iter 10 value 680.188938  
## iter 20 value 542.272152  
## iter 30 value 487.781308  
## iter 40 value 474.562506  
## iter 50 value 472.692989  
## final value 472.647231   
## converged  
## # weights: 50 (36 variable)  
## initial value 1100.635674   
## iter 10 value 689.879923  
## iter 20 value 530.789592  
## iter 30 value 477.365215  
## iter 40 value 462.862152  
## iter 50 value 462.412202  
## iter 60 value 462.393521  
## final value 462.393509   
## converged  
## # weights: 70 (54 variable)  
## initial value 1100.635674   
## iter 10 value 671.869937  
## iter 20 value 517.477864  
## iter 30 value 456.590869  
## iter 40 value 443.388162  
## iter 50 value 440.678146  
## iter 60 value 439.654394  
## iter 70 value 436.299906  
## iter 80 value 435.721397  
## iter 90 value 435.713822  
## final value 435.713805   
## converged  
## # weights: 70 (54 variable)  
## initial value 1100.635674   
## iter 10 value 653.629398  
## iter 20 value 521.489975  
## iter 30 value 457.007711  
## iter 40 value 444.851167  
## iter 50 value 442.599835  
## iter 60 value 441.687108  
## iter 70 value 438.629557  
## iter 80 value 437.688954  
## iter 90 value 437.679958  
## final value 437.679906   
## converged  
## # weights: 70 (54 variable)  
## initial value 1100.635674   
## iter 10 value 666.723188  
## iter 20 value 514.901259  
## iter 30 value 459.238686  
## iter 40 value 443.370432  
## iter 50 value 440.129634  
## iter 60 value 439.102266  
## iter 70 value 437.226686  
## iter 80 value 436.151645  
## iter 90 value 436.143872  
## final value 436.143847   
## converged  
## # weights: 70 (54 variable)  
## initial value 1100.635674   
## iter 10 value 663.053645  
## iter 20 value 521.462420  
## iter 30 value 469.670065  
## iter 40 value 447.157436  
## iter 50 value 444.293073  
## iter 60 value 443.002012  
## iter 70 value 440.410109  
## iter 80 value 440.080794  
## iter 90 value 440.076336  
## final value 440.076307   
## converged  
## Df AIC  
## <none> 977.09  
## + V3 9 979.43  
## + V8 9 980.29  
## + V6 9 983.36  
## + V9 9 988.15  
## - V2 9 992.54  
## - V10 9 996.79  
## - V4 9 1015.09  
## - V5 9 1017.29

stp$anova

## Stepwise Model Path   
## Analysis of Deviance Table  
##   
## Initial Model:  
## V7 ~ V2 + V3 + V4 + V5 + V6 + V8 + V9 + V10  
##   
## Final Model:  
## V7 ~ V2 + V4 + V5 + V10  
##   
##   
## Step Df Deviance Resid. Df Resid. Dev AIC  
## 1 397 840.5403 1002.5403  
## 2 - V9 9 6.562125 406 847.1024 991.1024  
## 3 - V6 9 10.261035 415 857.3635 983.3635  
## 4 - V8 9 14.064150 424 871.4276 979.4276  
## 5 - V3 9 15.657716 433 887.0853 977.0853

summary(stp)

## Call:  
## multinom(formula = V7 ~ V2 + V4 + V5 + V10, data = train)  
##   
## Coefficients:  
## (Intercept) V2 V4 V5 V10  
## 2 -4.244079 0.233314972 -0.01831701 0.09842886 0.4461068  
## 3 -3.977251 -0.002954217 0.23971446 0.26597426 0.2680801  
## 4 -5.984342 0.311416141 0.39004871 0.07554026 0.1362541  
## 5 -4.784467 0.200656304 0.40964542 0.32030620 -0.5156343  
## 6 -174.618945 -26.217078440 51.51823310 -138.74236122 24.7517131  
## 7 -8.738836 0.568360901 0.22925155 0.21899372 0.2849337  
## 8 -6.659441 0.142579032 0.63234144 0.36065952 -0.2551220  
## 9 -6.861861 0.006896686 0.52630341 0.09042632 0.5020227  
## 10 -6.004218 0.346808200 0.42398836 0.46102001 0.1049665  
##   
## Std. Errors:  
## (Intercept) V2 V4 V5 V10  
## 2 0.4985172 0.11167151 0.15751771 0.14376253 0.1480218  
## 3 0.4758144 0.11000987 0.12846841 0.11967032 0.1655908  
## 4 0.7961750 0.13979062 0.14234663 0.16011529 0.2147479  
## 5 0.6899172 0.10958731 0.12103548 0.11680544 0.3917847  
## 6 2.3055036 20.49769251 22.95502299 2.30818043 2.5236261  
## 7 1.6504876 0.22498564 0.20329670 0.18762045 0.2174785  
## 8 0.9550593 0.12773761 0.13886534 0.12713083 0.2792393  
## 9 1.1871724 0.20899273 0.20195468 0.21076758 0.2147189  
## 10 0.5434100 0.08321057 0.09448628 0.09089512 0.1463541  
##   
## Residual Deviance: 887.0853   
## AIC: 977.0853

# Generate the model from stepwise method

model<- lm(V7~V2+V4+V5+V8, cancer2)  
summary(model)

##   
## Call:  
## lm(formula = V7 ~ V2 + V4 + V5 + V8, data = cancer2)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -9.8115 -0.9531 -0.3111 0.6678 8.6889   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) -0.53601 0.17514 -3.060 0.0023 \*\*   
## V2 0.22617 0.04121 5.488 5.75e-08 \*\*\*  
## V4 0.31729 0.05086 6.239 7.76e-10 \*\*\*  
## V5 0.33227 0.04431 7.499 2.03e-13 \*\*\*  
## V8 0.32378 0.05606 5.775 1.17e-08 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 2.274 on 678 degrees of freedom  
## Multiple R-squared: 0.6129, Adjusted R-squared: 0.6107   
## F-statistic: 268.4 on 4 and 678 DF, p-value: < 2.2e-16

# Use test dataset to validate

pred<-round(predict(model,test))  
acc<-sum(pred == test$V7) / nrow(test)  
acc

## [1] 0.3268293

# 0.356 accuracy rate is not good.But this is the only model we got, so I will go ahead and use this model to impute the missings

# Get the subset of the data with the missings, and the subset with all the valid data points

mis2<-subset(cancer,V7=="?")  
ok<-subset(cancer,V7!="?")

# Assign the imputed values to V7

mis2$V7<-round(predict(model,mis2))

# Put these data back to the cancer dataset

cancer2final<-rbind(ok,mis2)  
cancer2final$V7<-as.integer(cancer2final$V7)

# make the values outside of the orignal range back to [1,10]

cancer2final$V7[cancer2final$V7 > 10] <- 10  
cancer2final$V7[cancer2final$V7 < 1] <- 1

# 3. Use regression with perturbation to impute values for the missing data.

set.seed(123)  
  
v7<-round(predict(model,mis2))  
  
mis3<-subset(cancer,V7=="?")  
  
v7new<-round(rnorm(nrow(mis3),v7,sd(v7)))

# make the values outside of the orignal range back to [1,10]

mis3$V7<-v7new  
  
mis3$V7[mis3$V7 > 10] <- 10  
mis3$V7[mis3$V7 < 1] <- 1  
  
cancer3<-rbind(ok,mis3)  
cancer3$V7<-as.integer(cancer3$V7)

# 4.Compare the results and quality of classification models (e.g., SVM, KNN) build using

# (1) the data sets from questions 1,2,3;

# 70% for training

mask\_train1<-sample(nrow(cancer1), size = floor(nrow(cancer1) \* 0.7))

# 4.1.1, with mode imputation

acc<-rep(0,25)

# training data set

train1<-cancer1[mask\_train1,]

# Using the remaining data for test

test1<-cancer1[-mask\_train1, ]   
  
for (k in 1:5){  
knn4.1.1 <- kknn(V11~V2+V3+ V4+ V5+ V6+ V7+ V8+ V9+ V10,train1,test1,k=k)  
pred1 <- as.integer(fitted(knn4.1.1)+0.5)  
acc[k]<-sum(pred1 == test1$V11) / nrow(test1)  
}

# 4.1.2, with regression imputation

train2<-cancer2final[mask\_train1,]   
test2<-cancer2final[-mask\_train1, ]   
  
for (k in 1:5){  
 knn4.1.2 <- kknn(V11~V2+V3+ V4+ V5+ V6+ V7+ V8+ V9+ V10,train2,test2,k=k)  
 pred2 <- as.integer(fitted(knn4.1.2)+0.5)  
 acc[k+5]<-sum(pred2 == test2$V11) / nrow(test2)  
}

# 4.1.3, with regression imputation

train3<-cancer3[mask\_train1,]   
test3<-cancer3[-mask\_train1, ]   
  
for (k in 1:5){  
 knn4.1.3 <- kknn(V11~V2+V3+ V4+ V5+ V6+ V7+ V8+ V9+ V10,train3,test3,k=k)  
 pred3<- as.integer(fitted(knn4.1.3)+0.5)  
 acc[k+10]<-sum(pred3 == test3$V11) / nrow(test3)  
}

# 4.2, the data that remains after data points with missing values are removed;

cancer4<-subset(cancer,V7!="?")  
cancer4$V7<-as.integer(cancer4$V7)  
train4<-cancer4[mask\_train1,]   
test4<-cancer4[-mask\_train1, ]   
  
for (k in 1:5){  
 knn4.2 <- kknn(V11~V2+V3+ V4+ V5+ V6+ V7+ V8+ V9+ V10,train4,test4,k=k)  
 pred4<- as.integer(fitted(knn4.2)+0.5)  
 acc[k+15]<-sum(pred4 == test4$V11) / nrow(test4)  
}

# 4.3, the data set when a binary variable is introduced to indicate missing values

# Add a binary variable to the original data to indicate if V7 is missing or not. 0=missing,1= not missing

cancer5 <- cancer  
cancer5$V12[cancer5$V7 == "?"] <- 0  
cancer5$V12[cancer5$V7 != "?"] <- 1

# Create interaction factor for V7 and V12.

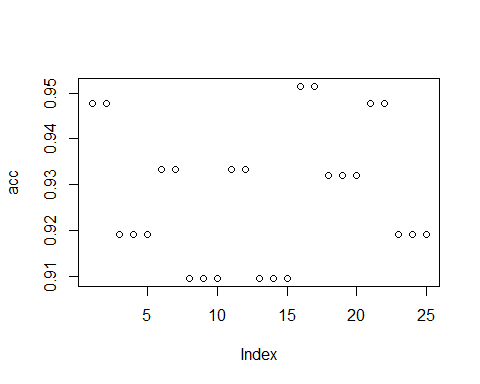
cancer5$V13[cancer5$V7 == "?"] <- 0  
cancer5$V13[cancer5$V7 != "?"] <- as.integer(ok$V7)  
  
train5<-cancer5[mask\_train1,]   
test5<-cancer5[-mask\_train1, ]

# Use the interaction factor in the modeling.

for (k in 1:5){  
 knn4.3 <- kknn(V11~V2+V3+ V4+ V5+ V6+ V8+ V9+ V10+V13,train5,test5,k=k)  
 pred5<- as.integer(fitted(knn4.3)+0.5)  
 acc[k+20]<-sum(pred5 == test5$V11) / nrow(test5)  
}  
  
acc

## [1] 0.9476190 0.9476190 0.9190476 0.9190476 0.9190476 0.9333333 0.9333333  
## [8] 0.9095238 0.9095238 0.9095238 0.9333333 0.9333333 0.9095238 0.9095238  
## [15] 0.9095238 0.9514563 0.9514563 0.9320388 0.9320388 0.9320388 0.9476190  
## [22] 0.9476190 0.9190476 0.9190476 0.9190476

plot(acc)



which.max(acc)

## [1] 16

# There isn’t much differences between the differenct methods to deal with the missing data (the accuracy rate are all withn 90%-95%).

# However, removing the missing values, generated a slightly higher predictive accuracy at k=1, for the knn model.

# Question 15.1

# Describe a situation or problem from your job, everyday life, current events, etc., for which optimization

# would be appropriate. What data would you need?

# Graduate students may want to decide which courses to choose in each semester, in order to maximize the GPA when graduating.

# Data needed:

# Workload of each courses and the time needed per week

# Personal schedules and estimated time that can be used for study

# Study plan that indicates which courses must be taken (based on school requirment, personal interest, and career goal)

# The order of the coursers (take introduction courses before the ones that require deeper understanding )

# Total credits taken each semester should meet school requirement

# The amount paid should within the education budget.