Coursera - Practical Machine Learning – Quiz 3

Question 1

For this quiz we will be using several R packages. R package versions change over time, the right answers have been checked using the following versions of the packages.

AppliedPredictiveModeling: v1.1.6

caret: v6.0.47

ElemStatLearn: v2012.04-0

pgmm: v1.1

rpart: v4.1.8

If you aren't using these versions of the packages, your answers may not exactly match the right answer, but hopefully should be close.

Load the cell segmentation data from the AppliedPredictiveModeling package using the commands:

```
library(AppliedPredictiveModeling)
## Warning: package 'AppliedPredictiveModeling' was built under R version
## 3.3.2
data(segmentationOriginal)
library(caret)
## Warning: package 'caret' was built under R version 3.3.2
## Loading required package: lattice
## Warning: package 'lattice' was built under R version 3.3.2
## Loading required package: ggplot2
## Warning: package 'ggplot2' was built under R version 3.3.2
```

1. Subset the data to a training set and testing set based on the Case variable in the data set.
2. Set the seed to 125 and fit a CART model with the rpart method using all predictor variables and default caret settings.
3. In the final model what would be the final model prediction for cases with the following variable values:

a. TotalIntench2 = 23,000; FiberWidthCh1 = 10; PerimStatusCh1=2

b. TotalIntench2 = 50,000; FiberWidthCh1 = 10;VarIntenCh4 = 100

c. TotalIntench2 = 57,000; FiberWidthCh1 = 8;VarIntenCh4 = 100

d. FiberWidthCh1 = 8;VarIntenCh4 = 100; PerimStatusCh1=2

-
    a. PS
    b. Not possible to predict

c. PS
d. Not possible to predict

- a. Not possible to predict
  b. WS
  c. PS
  d. PS
- a. PS
  b. WS
  c. PS
  d. WS
- a. PS
  b. WS
  c. PS
  d. Not possible to predict

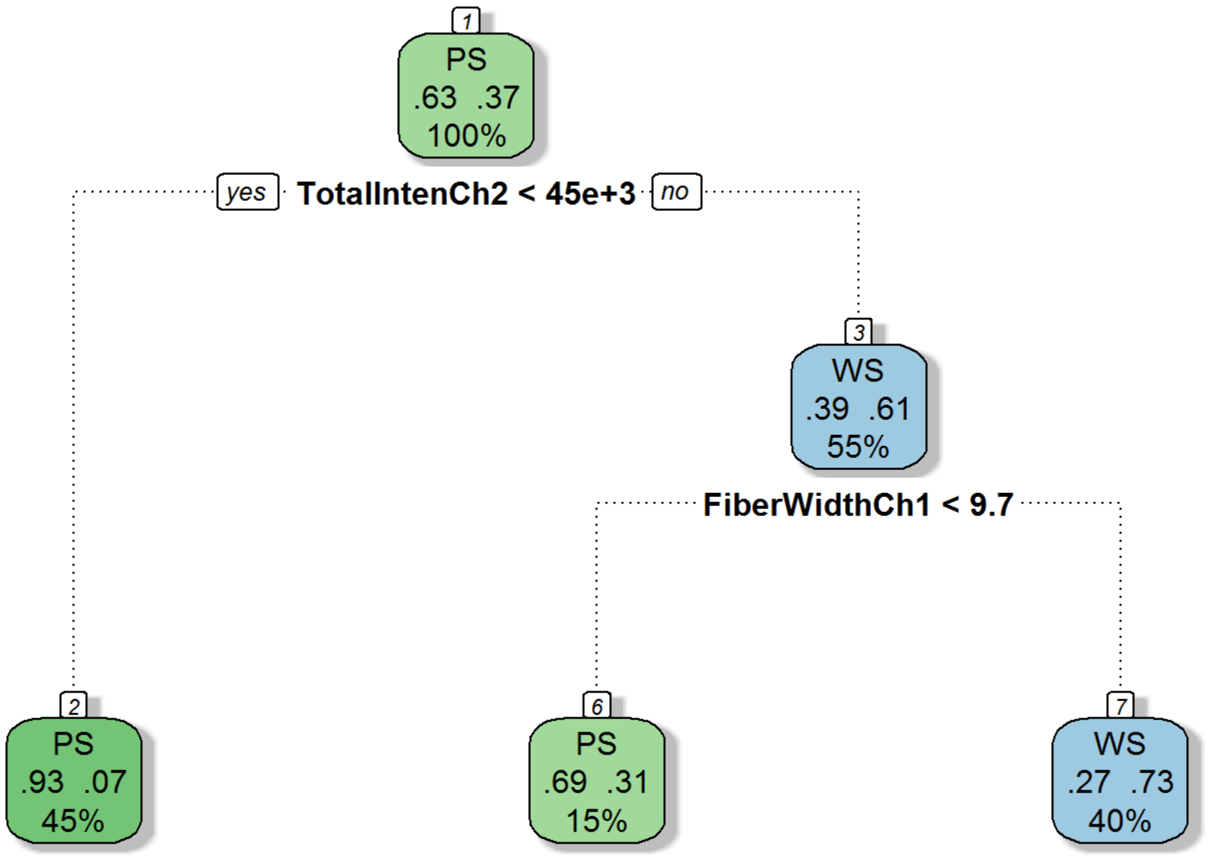
```
library(rattle)
## Warning: package 'rattle' was built under R version 3.3.2
## Rattle : une interface graphique gratuite pour l'exploration de données avec R.
## Version 4.1.0 Copyright (c) 2006-2015 Togaware Pty Ltd.
## Entrez 'rattle()' pour secouer, faire vibrer, et faire défiler vos données.
str(segmentationOriginal)
## 'data.frame':    2019 obs. of  119 variables:
##  $ Cell                         : int  207827637 207932307 207932463 207932470 2079324
55 207827656 207827659 207827661 207932479 207932480 ...
##  $ Case                         : Factor w/ 2 levels "Test","Train": 1 2 2 2 1 1 1 1 1
1 ...
##  $ Class                        : Factor w/ 2 levels "PS","WS": 1 1 2 1 1 2 2 1 2 2 ..
.
##  $ AngleCh1                     : num  143.25 133.75 106.65 69.15 2.89 ...
##  $ AngleStatusCh1               : int  1 0 0 0 2 2 1 1 2 1 ...
##  $ AreaCh1                      : int  185 819 431 298 285 172 177 251 495 384 ...
##  $ AreaStatusCh1                : int  0 1 0 0 0 0 0 0 0 0 ...
##  $ AvgIntenCh1                  : num  15.7 31.9 28 19.5 24.3 ...
##  $ AvgIntenCh2                  : num  3.95 205.88 115.32 101.29 111.42 ...
##  $ AvgIntenCh3                  : num  9.55 69.92 63.94 28.22 20.47 ...
##  $ AvgIntenCh4                  : num  2.21 164.15 106.7 31.03 40.58 ...
##  $ AvgIntenStatusCh1            : int  0 0 0 0 0 1 1 0 0 0 ...
##  $ AvgIntenStatusCh2            : int  2 0 0 0 0 1 1 2 0 0 ...
##  $ AvgIntenStatusCh3            : int  2 0 0 0 0 1 1 0 0 0 ...
##  $ AvgIntenStatusCh4            : int  2 0 0 2 0 1 0 2 0 0 ...
##  $ ConvexHullAreaRatioCh1       : num  1.12 1.26 1.05 1.2 1.11 ...
##  $ ConvexHullAreaRatioStatusCh1 : int  0 1 0 0 0 0 0 0 0 0 ...
##  $ ConvexHullPerimRatioCh1      : num  0.92 0.797 0.935 0.866 0.957 ...
##  $ ConvexHullPerimRatioStatusCh1: int  0 2 0 2 0 1 1 2 2 2 ...
##  $ DiffIntenDensityCh1          : num  29.5 31.9 32.5 26.7 31.6 ...
##  $ DiffIntenDensityCh3          : num  13.8 43.1 36 22.9 21.7 ...
##  $ DiffIntenDensityCh4          : num  6.83 79.31 51.36 26.39 25.03 ...
##  $ DiffIntenDensityStatusCh1    : int  2 0 0 2 0 1 1 2 2 2 ...
##  $ DiffIntenDensityStatusCh3    : int  2 0 0 0 0 1 0 0 2 0 ...
##  $ DiffIntenDensityStatusCh4    : int  2 0 0 2 2 1 1 2 0 0 ...
##  $ EntropyIntenCh1              : num  4.97 6.09 5.88 5.42 5.66 ...
##  $ EntropyIntenCh3              : num  4.37 6.64 6.68 5.44 5.29 ...
##  $ EntropyIntenCh4              : num  2.72 7.88 7.14 5.78 5.24 ...
##  $ EntropyIntenStatusCh1        : int  2 0 0 2 2 0 0 2 2 2 ...
##  $ EntropyIntenStatusCh3        : int  0 1 1 0 0 1 1 0 0 0 ...
##  $ EntropyIntenStatusCh4        : int  2 1 0 0 0 0 0 2 0 0 ...
##  $ EqCircDiamCh1                : num  15.4 32.3 23.4 19.5 19.1 ...
##  $ EqCircDiamStatusCh1          : int  0 1 0 0 0 2 2 0 0 0 ...
##  $ EqEllipseLWRCh1              : num  3.06 1.56 1.38 3.39 2.74 ...
##  $ EqEllipseLWRStatusCh1        : int  1 0 0 1 0 0 0 0 0 0 ...
##  $ EqEllipseOblateVolCh1        : num  337 2233 802 725 608 ...
##  $ EqEllipseOblateVolStatusCh1  : int  0 1 0 0 0 0 0 0 0 0 ...
##  $ EqEllipseProlateVolCh1       : num  110 1433 583 214 222 ...
##  $ EqEllipseProlateVolStatusCh1 : int  0 1 0 0 0 0 0 0 0 0 ...
##  $ EqSphereAreaCh1              : num  742 3279 1727 1195 1140 ...
##  $ EqSphereAreaStatusCh1        : int  0 1 0 0 0 0 0 0 0 0 ...
##  $ EqSphereVolCh1               : num  1901 17654 6751 3884 3621 ...
##  $ EqSphereVolStatusCh1         : int  0 1 0 0 0 0 0 0 0 0 ...
##  $ FiberAlign2Ch3               : num  0 0.488 0.301 0.22 0.491 ...
##  $ FiberAlign2Ch4               : num  0 0.352 0.522 0.733 0.385 ...
##  $ FiberAlign2StatusCh3         : int  2 0 0 0 0 0 0 2 0 1 ...
##  $ FiberAlign2StatusCh4         : int  2 0 0 1 0 0 0 2 0 1 ...
##  $ FiberLengthCh1               : num  27 64.3 21.1 43.1 34.7 ...
##  $ FiberLengthStatusCh1         : int  0 1 0 0 0 0 0 0 0 0 ...
##  $ FiberWidthCh1                : num  7.41 13.17 21.14 7.4 8.48 ...
##  $ FiberWidthStatusCh1          : int  2 0 1 2 2 0 0 2 0 0 ...
##  $ IntenCoocASMCh3              : num  0.01118 0.02805 0.00686 0.03096 0.02277 ...
##  $ IntenCoocASMCh4              : num  0.05045 0.01259 0.00614 0.01103 0.07969 ...
##  $ IntenCoocASMStatusCh3        : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ IntenCoocASMStatusCh4        : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ IntenCoocContrastCh3          : num  40.75 8.23 14.45 7.3 15.85 ...
##  $ IntenCoocContrastCh4          : num  13.9 6.98 16.7 13.39 3.54 ...
##  $ IntenCoocContrastStatusCh3    : int  1 0 0 0 0 0 0 1 0 1 ...
##  $ IntenCoocContrastStatusCh4    : int  1 0 1 1 2 0 1 0 0 1 ...
##  $ IntenCoocEntropyCh3           : num  7.2 6.82 7.58 6.31 6.78 ...
##  $ IntenCoocEntropyCh4           : num  5.25 7.1 7.67 7.2 5.5 ...
##  $ IntenCoocEntropyStatusCh3     : int  0 0 1 0 0 0 0 1 0 1 ...
##  $ IntenCoocEntropyStatusCh4     : int  0 0 1 0 0 0 0 2 0 1 ...
##  $ IntenCoocMaxCh3               : num  0.0774 0.1532 0.0284 0.1628 0.1274 ...
##  $ IntenCoocMaxCh4               : num  0.172 0.0739 0.0232 0.0775 0.2785 ...
##  $ IntenCoocMaxStatusCh3         : int  0 0 2 0 0 2 2 2 0 0 ...
##  $ IntenCoocMaxStatusCh4         : int  0 0 2 0 0 2 2 1 0 0 ...
##  $ KurtIntenCh1                  : num  -0.6567 -0.2488 -0.2935 0.6259 0.0421 ...
##  $ KurtIntenCh3                  : num  -0.608 -0.331 1.051 0.128 0.952 ...
##  $ KurtIntenCh4                  : num  0.726 -0.265 0.151 -0.347 -0.195 ...
##  $ KurtIntenStatusCh1            : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ KurtIntenStatusCh3            : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ KurtIntenStatusCh4            : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ LengthCh1                     : num  26.2 47.2 28.1 37.9 36 ...
##  $ LengthStatusCh1               : int  0 1 0 0 0 2 2 0 0 0 ...
##  $ MemberAvgAvgIntenStatusCh2    : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ MemberAvgTotalIntenStatusCh2 : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ NeighborAvgDistCh1            : num  370 174 158 206 205 ...
##  $ NeighborAvgDistStatusCh1      : int  1 2 2 0 0 0 0 0 0 0 ...
##  $ NeighborMinDistCh1            : num  99.1 30.1 34.9 33.1 27 ...
##  $ NeighborMinDistStatusCh1      : int  1 0 0 0 0 0 0 0 0 0 ...
##  $ NeighborVarDistCh1            : num  128 81.4 90.4 116.9 111 ...
##  $ NeighborVarDistStatusCh1      : int  0 2 2 0 0 0 0 0 2 2 ...
##  $ PerimCh1                      : num  68.8 154.9 84.6 101.1 86.5 ...
##  $ PerimStatusCh1                : int  0 1 0 0 0 0 0 0 0 0 ...
##  $ ShapeBFRCh1                   : num  0.665 0.54 0.724 0.589 0.6 ...
##  $ ShapeBFRStatusCh1             : int  0 2 1 0 0 0 1 0 0 0 ...
##  $ ShapeLWRCh1                   : num  2.46 1.47 1.33 2.83 2.73 ...
##  $ ShapeLWRStatusCh1             : int  0 0 0 1 1 0 0 0 0 0 ...
##  $ ShapeP2ACh1                   : num  1.88 2.26 1.27 2.55 2.02 ...
##  $ ShapeP2AStatusCh1             : int  0 0 0 1 0 0 0 1 0 0 ...
##  $ SkewIntenCh1                  : num  0.455 0.399 0.472 0.882 0.517 ...
##  $ SkewIntenCh3                  : num  0.46 0.62 0.971 1 1.177 ...
##  $ SkewIntenCh4                    : num  1.233 0.527 0.325 0.604 0.926 ...
##  $ SkewIntenStatusCh1              : int  0 0 0 1 0 2 2 0 0 0 ...
##  $ SkewIntenStatusCh3              : int  0 0 0 0 0 2 2 2 0 0 ...
##  $ SkewIntenStatusCh4              : int  0 0 0 0 0 0 0 1 0 0 ...
##  $ SpotFiberCountCh3               : int  1 4 2 4 1 1 0 2 1 1 ...
##  $ SpotFiberCountCh4               : int  4 11 6 7 7 4 4 7 11 7 ...
##   [list output truncated]
```

```
trainSet <- segmentationOriginal[segmentationOriginal$Case =="Train",]
testSet <- segmentationOriginal[segmentationOriginal$Case =="Test",]

set.seed(125)
model_rpart <- train(Class~.,data=trainSet,method="rpart")
```

```
## Loading required package: rpart
```

```
fancyRpartPlot(model_rpart$finalModel)
```



Rattle 2017-janv.-17 18:22:03 Jean-Luc

Answer : : the correct answer is : ***a: PS;b: WS;c: PS;d: Not possible to predict***. The choices a to c lead to a specific leaf or the tree. n the CART algorithm, the majority class is then considered as the prediction.

Question 2

If K is small in a K-fold cross validation is the bias in the estimate of out-of-sample (test set) accuracy smaller or bigger? If K is small is the variance in the estimate of out-of-sample (test set) accuracy smaller or bigger? Is K large or small in leave one out cross validation ?

- The bias is larger and the variance is smaller. Under leave one out cross validation K is equal to the sample size.
- The bias is larger and the variance is smaller. Under leave one out cross validation K is equal to one.
- The bias is smaller and the variance is bigger. Under leave one out cross validation K is equal to one.
- The bias is smaller and the variance is smaller. Under leave one out cross validation K is equal to the sample size.

Answer : ***The bias is larger and the variance is smaller. Under leave one out cross validation K is equal to the sample size***.

Question 3

Load the olive oil data using the commands:

```
library(pgmm)
## Warning: package 'pgmm' was built under R version 3.3.2
data(olive)
olive = olive[,-1]
```

(NOTE: If you have trouble installing the pgmm package, you can download the -code-olive-/code- dataset here: olive_data.zip. After unzipping the archive, you can load the file using the -code-load()-/code- function in R.)

These data contain information on 572 different Italian olive oils from multiple regions in Italy. Fit a classification tree where Area is the outcome variable. Then predict the value of area for the following data frame using the tree command with all defaults.

```
newdata = as.data.frame(t(colMeans(olive)))
```

What is the resulting prediction? Is the resulting prediction strange? Why or why not?

- 0.005291005 0 0.994709 0 0 0 0 0 0. The result is strange because Area is a numeric variable and we should get the average within each leaf.
- 4.59965. There is no reason why the result is strange.
- 2.783. It is strange because Area should be a qualitative variable - but tree is reporting the average value of Area as a numeric variable in the leaf predicted for newdata
- 2.783. There is no reason why this result is strange.

```
str(olive)
## 'data.frame':    572 obs. of  9 variables:
##  $ Area       : num  1 1 1 1 1 1 1 1 1 1 ...
##  $ Palmitic   : num  1075 1088 911 966 1051 ...
##  $ Palmitoleic: num  75 73 54 57 67 49 66 61 60 55 ...
##  $ Stearic    : num  226 224 246 240 259 268 264 235 239 213 ...
##  $ Oleic      : num  7823 7709 8113 7952 7771 ...
##  $ Linoleic   : num  672 781 549 619 672 678 618 734 709 633 ...
##  $ Linolenic  : num  36 31 31 50 50 51 49 39 46 26 ...
##  $ Arachidic  : num  60 61 63 78 80 70 56 64 83 52 ...
##  $ Eicosenoic : num  29 29 29 35 46 44 29 35 33 30 ...
```

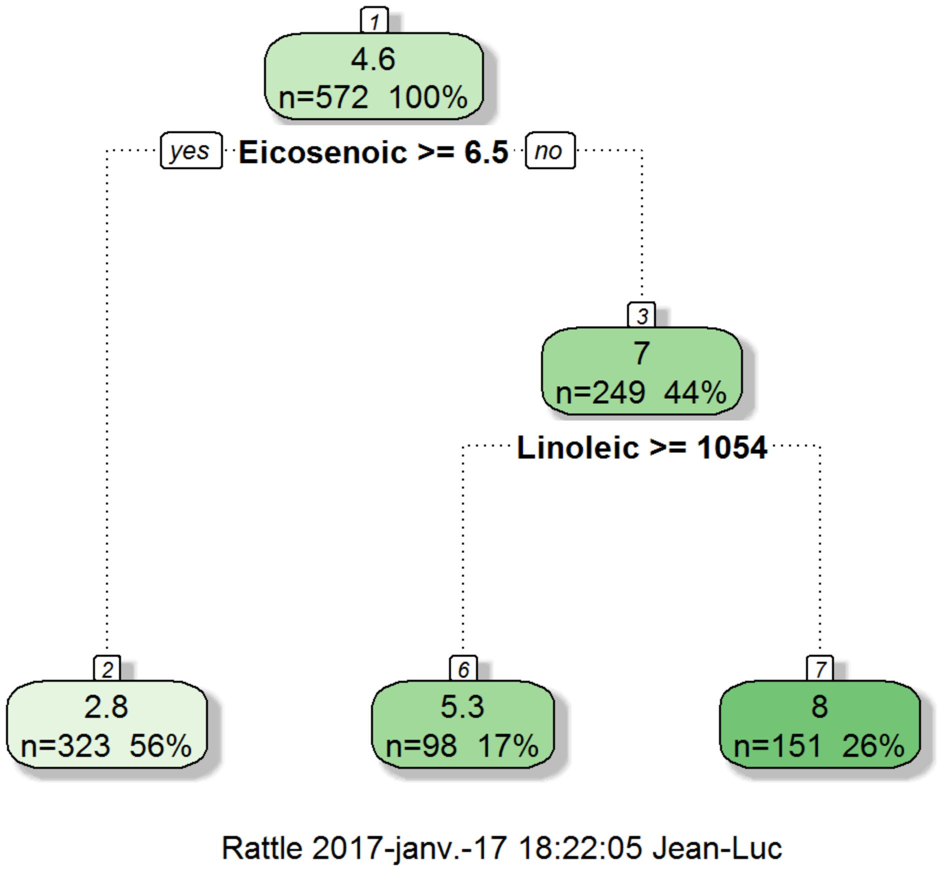
```
table(olive$Area)
```

```
## 
##   1   2   3   4   5   6   7   8   9 
##  25  56 206  36  65  33  50  50  51
```

```
olive_rpart <- train(Area~.,data=olive,method="rpart")
```

```
## Warning in nominalTrainWorkflow(x = x, y = y, wts = weights, info =
## trainInfo, : There were missing values in resampled performance measures.
```

```
fancyRpartPlot(olive_rpart$finalModel)
```



```
predict(olive_rpart,newdata=newdata)
```

```
##        1 
## 2.783282
```

Answer : : the correct answer is : ***2.783. It is strange because Area should be a qualitative variable - but tree is reporting the average value of Area as a numeric variable in the leaf predicted for newdata***.

Question 4

Load the South Africa Heart Disease Data and create training and test sets with the following code:

```
library(ElemStatLearn)
## Warning: package 'ElemStatLearn' was built under R version 3.3.2
data(SAheart)
set.seed(8484)
train = sample(1:dim(SAheart)[1],size=dim(SAheart)[1]/2,replace=F)
trainSA = SAheart[train,]
testSA = SAheart[-train,]
```

Then set the seed to 13234 and fit a logistic regression model (method="glm", be sure to specify family="binomial") with Coronary Heart Disease (chd) as the outcome and age at onset, current alcohol consumption, obesity levels, cumulative tabacco, type-A behavior, and low density lipoprotein cholesterol as predictors. Calculate the misclassification rate for your model using this function and a prediction on the "response" scale:

```
missClass = function(values,prediction){sum(((prediction > 0.5)*1) != values)/length(valu
es)}
```

What is the misclassification rate on the training set? What is the misclassification rate on the test set?

- Test Set Misclassification: 0.27
  Training Set: 0.31
- Test Set Misclassification: 0.43
  Training Set: 0.31
- Test Set Misclassification: 0.31
  Training Set: 0.27
- Test Set Misclassification: 0.35
  Training Set: 0.31

```
set.seed(13234)


# definition of the training model
regModel <- train(chd~age+alcohol+obesity+tobacco+typea+ldl,data=trainSA,method="glm",fam
ily="binomial")
## Warning in train.default(x, y, weights = w, ...): You are trying to do
## regression and your outcome only has two possible values Are you trying to
## do classification? If so, use a 2 level factor as your outcome column.
# computation of the misclasssification on the training set and test set
missClassTrain <- missClass(trainSA$chd,predict(regModel,newdata=trainSA))
missClassTest <- missClass(testSA$chd,predict(regModel,newdata=testSA))
missClassTrain
## [1] 0.2727273
missClassTest
## [1] 0.3116883
```

Answer : : the correct answer is : ***Test Set Misclassification: 0.31***
Training Set: 0.27.

Question 5

Load the vowel.train and vowel.test data sets:

```
library(ElemStatLearn)

data(vowel.train)

data(vowel.test)
```

Set the variable y to be a factor variable in both the training and test set. Then set the seed to 33833. Fit a random forest predictor relating the factor variable y to the remaining variables. Read about variable importance in random forests here: http://www.stat.berkeley.edu/~breiman/RandomForests/cc_home.htm#ooberr.

The caret package uses by default the Gini importance.

Calculate the variable importance using the varImp function in the caret package. What is the order of variable importance?

[NOTE: Use randomForest() specifically, not caret, as there's been some issues reported with that approach. 11/6/2016]

- The order of the variables is:
  x.1, x.2, x.3, x.8, x.6, x.4, x.5, x.9, x.7,x.10
- The order of the variables is:
  x.10, x.7, x.9, x.5, x.8, x.4, x.6, x.3, x.1,x.2
- The order of the variables is:
  x.2, x.1, x.5, x.6, x.8, x.4, x.9, x.3, x.7,x.10
- The order of the variables is:
  x.10, x.7, x.5, x.6, x.8, x.4, x.9, x.3, x.1,x.2

```
set.seed(33833)

str(vowel.train)
## 'data.frame':    528 obs. of  11 variables:

##  $ y    : int  1 2 3 4 5 6 7 8 9 10 ...

##  $ x.1 : num  -3.64 -3.33 -2.12 -2.29 -2.6 ...

##  $ x.2 : num  0.418 0.496 0.894 1.809 1.938 ...

##  $ x.3 : num  -0.67 -0.694 -1.576 -1.498 -0.846 ...

##  $ x.4 : num  1.779 1.365 0.147 1.012 1.062 ...

##  $ x.5 : num  -0.168 -0.265 -0.707 -1.053 -1.633 ...

##  $ x.6 : num  1.627 1.933 1.559 1.06 0.764 ...

##  $ x.7 : num  -0.388 -0.363 -0.579 -0.567 0.394 0.217 0.322 -0.435 -0.512 -0.466 ...

##  $ x.8 : num  0.529 0.51 0.676 0.235 -0.15 -0.246 0.45 0.992 0.928 0.702 ...

##  $ x.9 : num  -0.874 -0.621 -0.809 -0.091 0.277 0.238 0.377 0.575 -0.167 0.06 ...

##  $ x.10: num  -0.814 -0.488 -0.049 -0.795 -0.396 -0.365 -0.366 -0.301 -0.434 -0.836 ..
.
library(randomForest)
```

```
## Warning: package 'randomForest' was built under R version 3.3.2
## randomForest 4.6-12
## Type rfNews() to see new features/changes/bug fixes.
##
## Attaching package: 'randomForest'
## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
vowel.train$y <- as.factor(vowel.train$y)
vowel.test$y <- as.factor(vowel.test$y)
modelRF <- randomForest(y~.,data=vowel.train)
order(varImp(modelRF),decreasing=TRUE)
```

```
##  [1]  2  1  5  6  8  4  9  3  7 10
```

Answer : : the correct answer is : ***The order of the variables is: x.2, x.1, x.5, x.6, x.8, x.4, x.9, x.3, x.7,x.10***.