Q2\_machine\_learning.R

admin

Thu Feb 12 20:03:59 2015

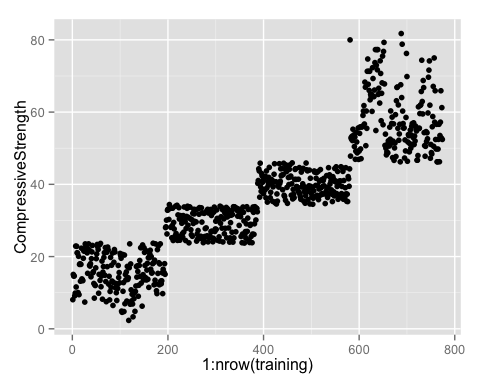
# Question 2   
library(AppliedPredictiveModeling)  
data(concrete)  
library(caret)

## Loading required package: lattice  
## Loading required package: ggplot2

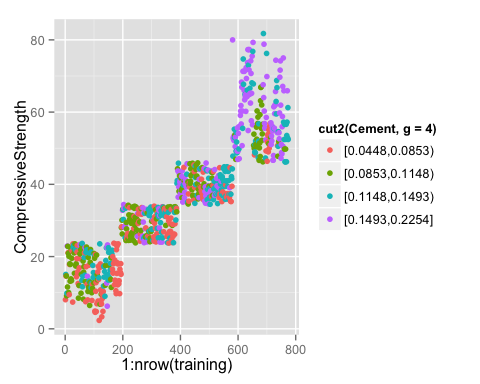
library(Hmisc)

## Loading required package: grid  
## Loading required package: survival  
## Loading required package: splines  
##   
## Attaching package: 'survival'  
##   
## The following object is masked from 'package:caret':  
##   
## cluster  
##   
## Loading required package: Formula  
##   
## Attaching package: 'Hmisc'  
##   
## The following objects are masked from 'package:base':  
##   
## format.pval, round.POSIXt, trunc.POSIXt, units

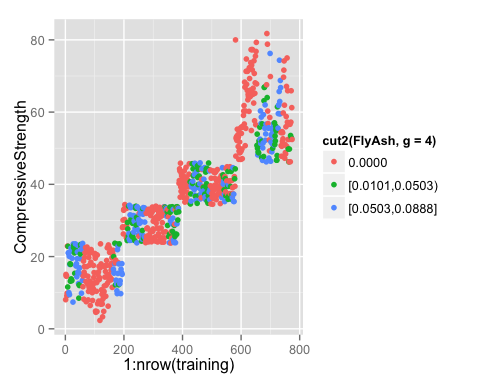
set.seed(975)  
inTrain = createDataPartition(mixtures$CompressiveStrength, p = 3/4)[[1]]  
training = mixtures[ inTrain,]  
testing = mixtures[-inTrain,]  
qplot(1:nrow(training), CompressiveStrength, data=training)



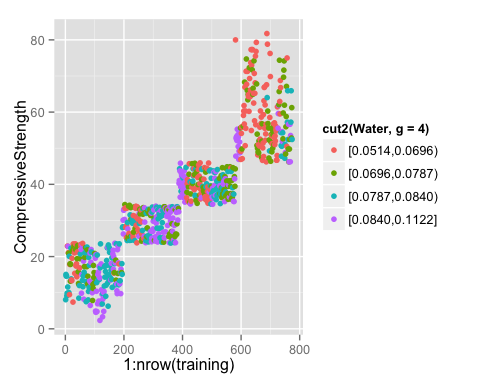
qplot(1:nrow(training), CompressiveStrength, colour=cut2(Cement, g=4), data=training)



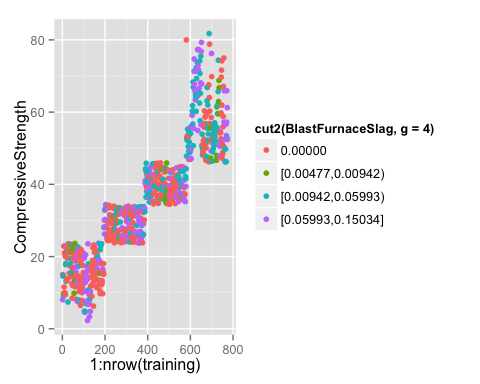
qplot(1:nrow(training), CompressiveStrength, colour=cut2(FlyAsh, g=4), data=training)



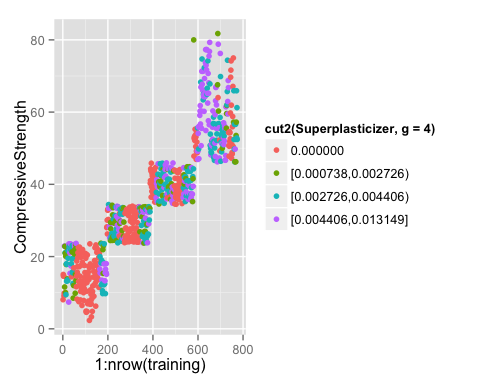
qplot(1:nrow(training), CompressiveStrength, colour=cut2(Water, g=4), data=training)



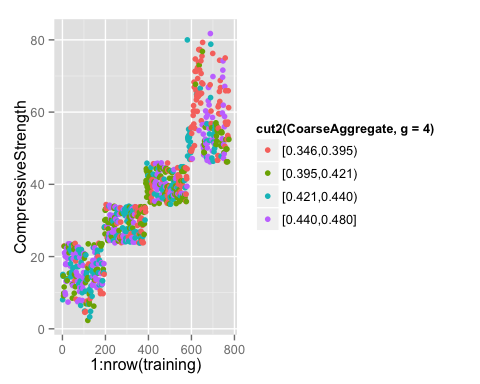
qplot(1:nrow(training), CompressiveStrength, colour=cut2(BlastFurnaceSlag, g=4), data=training)



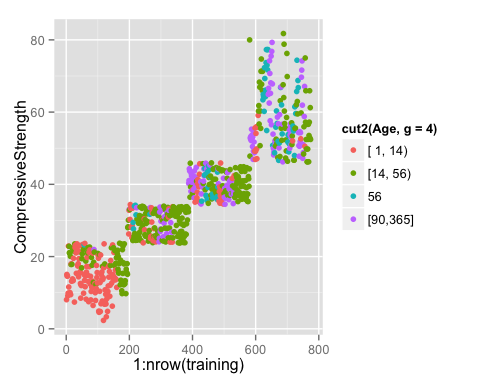
qplot(1:nrow(training), CompressiveStrength, colour=cut2(Superplasticizer, g=4), data=training)



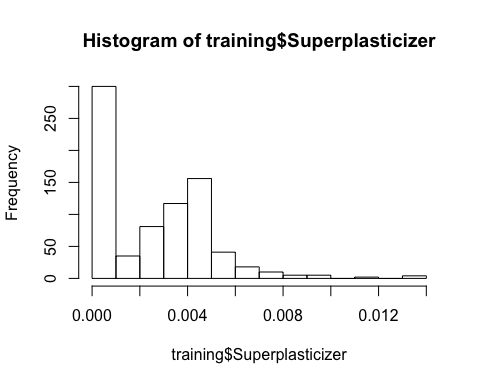
qplot(1:nrow(training), CompressiveStrength, colour=cut2(CoarseAggregate, g=4), data=training)



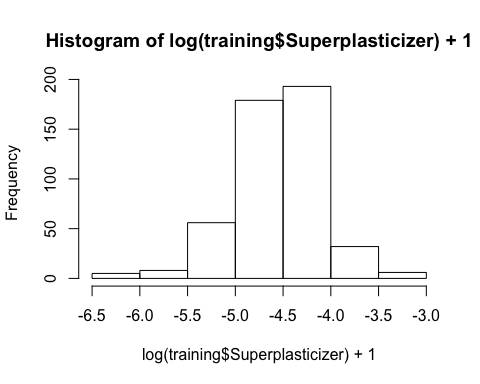
qplot(1:nrow(training), CompressiveStrength, colour=cut2(Age, g=4), data=training)



# Question 3  
library(AppliedPredictiveModeling)  
data(concrete)  
library(caret)  
set.seed(975)  
inTrain = createDataPartition(mixtures$CompressiveStrength, p = 3/4)[[1]]  
training = mixtures[ inTrain,]  
testing = mixtures[-inTrain,]  
hist(training$Superplasticizer)

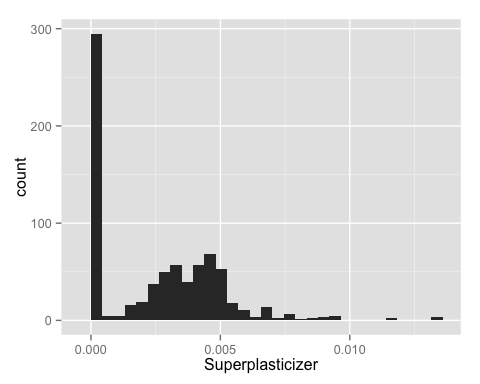


hist(log(training$Superplasticizer)+1)



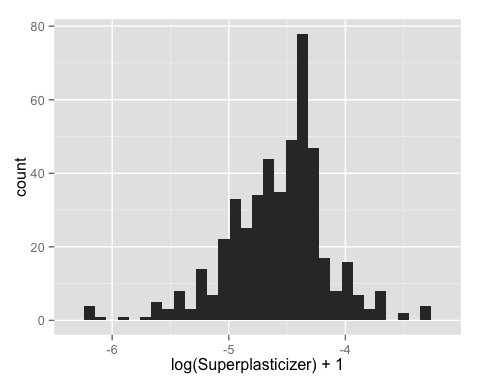
qplot(Superplasticizer, data=training)

## stat\_bin: binwidth defaulted to range/30. Use 'binwidth = x' to adjust this.



qplot(log(Superplasticizer)+1, data=training)

## stat\_bin: binwidth defaulted to range/30. Use 'binwidth = x' to adjust this.



# Question 5 answer  
library(caret)  
library(AppliedPredictiveModeling)  
install.packages("e1071")

##   
## The downloaded binary packages are in  
## /var/folders/5t/0086c30h8xjdvr001s6zsv780000gr/T//RtmpztwXH8/downloaded\_packages

set.seed(3433)  
data(AlzheimerDisease)  
adData = data.frame(diagnosis,predictors)  
inTrain = createDataPartition(adData$diagnosis, p = 3/4)[[1]]  
training = adData[ inTrain,]  
testing = adData[-inTrain,]  
trainData = training[, c(1, 58:69)]  
testData = testing[, c(1, 58:69)]  
  
preProc <- preProcess(trainData[,-1], method="pca", thresh=0.8)  
preProc # answer to question 4

##   
## Call:  
## preProcess.default(x = trainData[, -1], method = "pca", thresh = 0.8)  
##   
## Created from 251 samples and 12 variables  
## Pre-processing: principal component signal extraction, scaled, centered   
##   
## PCA needed 7 components to capture 80 percent of the variance

trainPC <- predict(preProc, trainData[,-1])  
modelFitPCA <- train(trainData$diagnosis ~., method="glm", data=trainPC)  
testPC <- predict(preProc, testData[,-1])  
confusionMatrix(testData$diagnosis, predict(modelFitPCA, testPC))

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction Impaired Control  
## Impaired 3 19  
## Control 4 56  
##   
## Accuracy : 0.7195   
## 95% CI : (0.6094, 0.8132)  
## No Information Rate : 0.9146   
## P-Value [Acc > NIR] : 1.000000   
##   
## Kappa : 0.0889   
## Mcnemar's Test P-Value : 0.003509   
##   
## Sensitivity : 0.42857   
## Specificity : 0.74667   
## Pos Pred Value : 0.13636   
## Neg Pred Value : 0.93333   
## Prevalence : 0.08537   
## Detection Rate : 0.03659   
## Detection Prevalence : 0.26829   
## Balanced Accuracy : 0.58762   
##   
## 'Positive' Class : Impaired   
##

preProc <- preProcess(trainData[,-1], method="pca", pcaComp=12)  
trainPC <- predict(preProc, trainData[,-1])  
modelFitAll <- train(trainData$diagnosis ~., method="glm", data=trainPC)  
testPC <- predict(preProc, testData[,-1])  
confusionMatrix(testData$diagnosis, predict(modelFitAll, testPC))

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction Impaired Control  
## Impaired 2 20  
## Control 9 51  
##   
## Accuracy : 0.6463   
## 95% CI : (0.533, 0.7488)  
## No Information Rate : 0.8659   
## P-Value [Acc > NIR] : 1.00000   
##   
## Kappa : -0.0702   
## Mcnemar's Test P-Value : 0.06332   
##   
## Sensitivity : 0.18182   
## Specificity : 0.71831   
## Pos Pred Value : 0.09091   
## Neg Pred Value : 0.85000   
## Prevalence : 0.13415   
## Detection Rate : 0.02439   
## Detection Prevalence : 0.26829   
## Balanced Accuracy : 0.45006   
##   
## 'Positive' Class : Impaired   
##