Homework 5 Solution

Brian Lois

3/5/2017

```
knitr::opts_chunk$set(echo = TRUE)
library(caret)
library(AppliedPredictiveModeling)
library(class)
library(kernlab)
data(hepatic)
set.seed(111)
```

1.

a.

Stratified sampling should be used because of the small number of 'Severe' samples.

```
train_indices <- createDataPartition(injury, p = .8, list = FALSE)
combined_data <- cbind(bio, injury)
train_data <- combined_data[train_indices, ]
test_data <- combined_data[-train_indices, ]
```

b & c.

```
train_data$none <- train_data$injury == 'None'
train_data$mild <- train_data$injury == 'Mild'
train_data$severe <- train_data$injury == 'Severe'

make_formula <- function(target){
  input_features <- colnames(train_data)[1:184]
  input_features_string<- paste(input_features, collapse = ' + ')
  formula_string <- paste(target, ' ~ ', input_features_string)
  formula <- as.formula(formula_string)
  return(formula)
}

models <- list()
preds <- as.data.frame(test_data$injury)
for (outcome in c('none', 'mild', 'severe')){
  models[[outcome]] <- glm(make_formula(outcome), data = train_data, family = binomial())
  preds[[outcome]] <- predict(models[[outcome]], test_data)
}

preds$pc_num <- max.col(preds[ ,2:4])
preds$predicted_class[preds$pc_num == 1] <- 'None'
preds$predicted_class[preds$pc_num == 2] <- 'Mild'
preds$predicted_class[preds$pc_num == 3] <- 'Severe'
```

```
head(preds)
```

```
##   test_data$injury       none      mild      severe pc_num predicted_class
## 1             None -113.67586 -293.3932  -83.953439      3          Severe
## 2           Severe -605.03561  330.3093 -127.774070      2            Mild
## 3           Severe -652.67042  405.5730    7.284565      2            Mild
## 4             None  -77.67313 -239.7453  258.027800      3          Severe
## 5             Mild  -26.20327  127.6525  -44.609875      2            Mild
## 6             None  -50.25891  619.3967 -361.736579      2            Mild
```

d.

Print the confusion matrix

```
(conf_mat <- table(preds$`test_data$injury`, preds$predicted_class))
```

```
##
##          Mild None Severe
##   Mild     16    8      5
##   None     11    6      4
##   Severe    6    0      0
```

Accuracy:

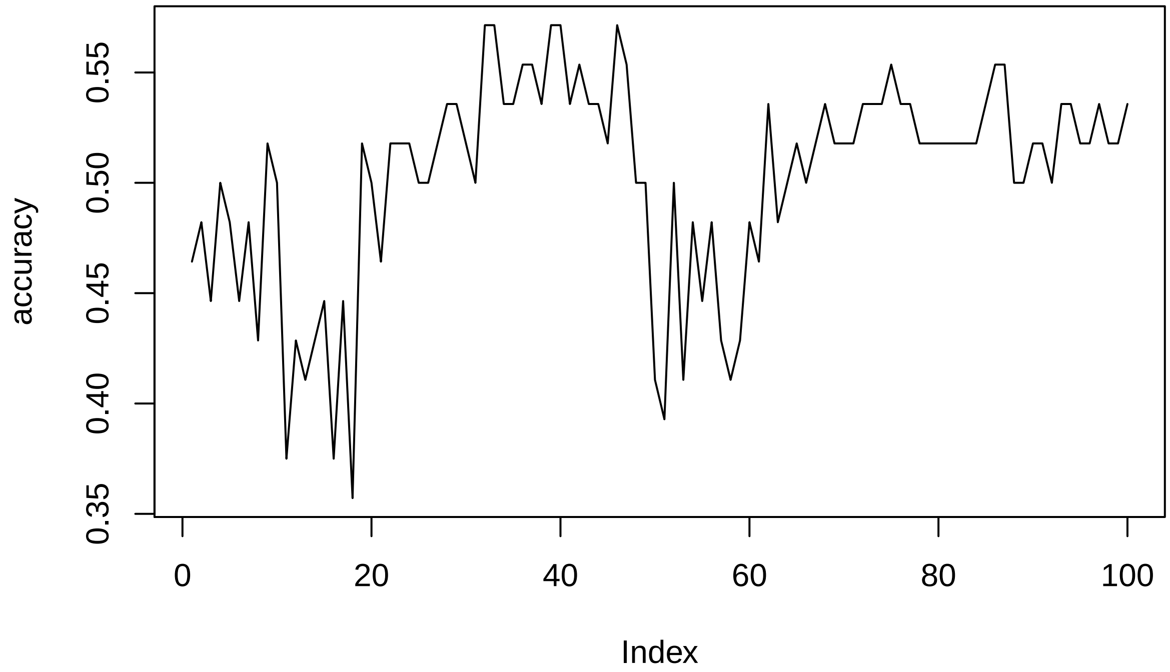
```
(conf_mat[1,1] + conf_mat[2,2] + conf_mat[3,3]) / nrow(test_data)
```

```
## [1] 0.3928571
```

Since 51% of the data are ‘Mild’ we could do better by predicting ‘Mild’ for everything.

2.

```
accuracy <- c()
for (k in seq(1, 100, 1)){
  model <- knn(train_data[,1:184], test_data[,1:184], train_data$injury, k = k)
  conf_mat <- table(model, test_data$injury)
  acc <- (conf_mat[1,1] + conf_mat[2,2] + conf_mat[3,3]) / nrow(test_data)
  accuracy <- c(accuracy, acc)
}
plot(accuracy, type = 'l')
```



From this plot, I would choose k = 40.

3.

```
train_data <- train_data[train_data$injury != 'Severe', ]
test_data <- test_data[test_data$injury != 'Severe', ]

train_data$injury <- factor(train_data$injury)
test_data$injury <- factor(test_data$injury)

for (ker in c('vanilladot', 'rbfdot')){
  for (C in seq(.1, 5.1, 1)){
    svm_model <- ksvm(x = as.matrix(train_data[ ,1:184]),
                      y = train_data$injury,
                      kernel = ker,
                      C = C)
    preds <- predict(svm_model, as.matrix(test_data[,1:184]))
    conf_mat <- table(preds, test_data$injury)
    acc <- (conf_mat[1,1] + conf_mat[2,2]) / nrow(test_data)
    print(c(ker, C, acc))
  }
}
```

```
##  Setting default kernel parameters
## [1] "vanilladot" "0.1"        "0.66"
##  Setting default kernel parameters
## [1] "vanilladot" "1.1"        "0.62"
##  Setting default kernel parameters
## [1] "vanilladot" "2.1"        "0.64"
##  Setting default kernel parameters
## [1] "vanilladot" "3.1"        "0.6"
##  Setting default kernel parameters
## [1] "vanilladot" "4.1"        "0.62"
##  Setting default kernel parameters
## [1] "vanilladot" "5.1"        "0.58"
## [1] "rbfdot" "0.1"    "0.58"
## [1] "rbfdot" "1.1"    "0.62"
## [1] "rbfdot" "2.1"    "0.62"
## [1] "rbfdot" "3.1"    "0.64"
## [1] "rbfdot" "4.1"    "0.6"
## [1] "rbfdot" "5.1"    "0.68"
```