Support Vector Machine

Using the Breast Cancer Wisconsin (Diagnostic) Data Set.

```
my_link <- 'http://archive.ics.uci.edu/ml/machine-learning-databases/breast-cancer-wisconsin/breast-can
data <- read.table(url(my_link), stringsAsFactors = FALSE, header = FALSE, sep = ',')
names(data) <- c('id','ct','ucsize','ucshape','ma','secs','bn','bc','nn','miti','class')
head(data)
```

```
##        id ct ucsize ucshape ma secs bn bc nn miti class
## 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
```

Any missing data?

```
dim(data)[1] * dim(data)[2]
```

```
## [1] 7689
```

```
table(is.na.data.frame(data))
```

```
##
## FALSE
##  7689
```

What's the structure?

```
str(data)
```

```
## 'data.frame':    699 obs. of  11 variables:
##  $ id     : int  1000025 1002945 1015425 1016277 1017023 1017122 1018099 1018561 1033078 1033078 ...
##  $ ct     : int  5 5 3 6 4 8 1 2 2 4 ...
##  $ ucsize : int  1 4 1 8 1 10 1 1 1 2 ...
##  $ ucshape: int  1 4 1 8 1 10 1 2 1 1 ...
##  $ ma     : int  1 5 1 1 3 8 1 1 1 1 ...
##  $ secs   : int  2 7 2 3 2 7 2 2 2 2 ...
##  $ bn     : chr  "1" "10" "2" "4" ...
##  $ bc     : int  3 3 3 3 3 9 3 3 1 2 ...
##  $ nn     : int  1 2 1 7 1 7 1 1 1 1 ...
##  $ miti   : int  1 1 1 1 1 1 1 1 5 1 ...
##  $ class  : int  2 2 2 2 2 4 2 2 2 2 ...
```

Why is the bare nuclei (bn) stored as characters instead of integers?

```
table(data$bn)
```

```
##
##   ?   1  10   2   3   4   5   6   7   8   9
##  16 402 132  30  28  19  30   4   8  21   9
```

Change the question marks into NA's and then into median values.

```
data$bn <- gsub(pattern = '\\?', replacement = NA, x = data$bn)
data$bn <- as.integer(data$bn)
my_median <- median(data$bn, na.rm = TRUE)
```

```
data$bn[is.na(data$bn)] <- my_median
str(data)
```

```
## 'data.frame':    699 obs. of  11 variables:
##  $ id     : int  1000025 1002945 1015425 1016277 1017023 1017122 1018099 1018561 1033078 1033078 ...
##  $ ct     : int  5 5 3 6 4 8 1 2 2 4 ...
##  $ ucsize : int  1 4 1 8 1 10 1 1 1 2 ...
##  $ ucshape: int  1 4 1 8 1 10 1 2 1 1 ...
##  $ ma     : int  1 5 1 1 3 8 1 1 1 1 ...
##  $ secs   : int  2 7 2 3 2 7 2 2 2 2 ...
##  $ bn     : int  1 10 2 4 1 10 10 1 1 1 ...
##  $ bc     : int  3 3 3 3 3 9 3 3 1 2 ...
##  $ nn     : int  1 2 1 7 1 7 1 1 1 1 ...
##  $ miti   : int  1 1 1 1 1 1 1 1 5 1 ...
##  $ class  : int  2 2 2 2 2 4 2 2 2 2 ...
```

The class should be a factor; 2 is benign and 4 is malignant.

```
data$class <- factor(data$class)
```

Finally remove id the row name, which was not unique anyway.

```
data <- data[,-1]
```

Separate into training (80%) and testing (20%).

```
set.seed(31)
my_decider <- rbinom(n=nrow(data),size=1,p=0.8)
table(my_decider)
```

```
## my_decider
##   0   1
## 151 548
```

```
train <- data[as.logical(my_decider),]
test <- data[!as.logical(my_decider),]
```

Using the `e1071` package.

```
library(e1071)

tuned <- tune.svm(class ~ ., data = train, gamma = 10^(-6:-1), cost = 10^(-1:1))
summary(tuned)
```

```
##
## Parameter tuning of 'svm':
##
## - sampling method: 10-fold cross validation
##
## - best parameters:
##  gamma cost
##   0.01    1
##
## - best performance: 0.02548822
##
## - Detailed performance results:
##    gamma cost      error dispersion
## 1  1e-06  0.1 0.34306397 0.07307879
```

```
## 2  1e-05  0.1 0.34306397 0.07307879
## 3  1e-04  0.1 0.34306397 0.07307879
## 4  1e-03  0.1 0.31740741 0.07132964
## 5  1e-02  0.1 0.03286195 0.02073629
## 6  1e-01  0.1 0.03282828 0.01432483
## 7  1e-06  1.0 0.34306397 0.07307879
## 8  1e-05  1.0 0.34306397 0.07307879
## 9  1e-04  1.0 0.31555556 0.07164412
## 10 1e-03  1.0 0.03649832 0.02430065
## 11 1e-02  1.0 0.02548822 0.01755013
## 12 1e-01  1.0 0.02734007 0.01763158
## 13 1e-06 10.0 0.34306397 0.07307879
## 14 1e-05 10.0 0.31555556 0.07164412
## 15 1e-04 10.0 0.03649832 0.02430065
## 16 1e-03 10.0 0.02548822 0.01755013
## 17 1e-02 10.0 0.03646465 0.01480125
## 18 1e-01 10.0 0.04016835 0.02241415
```

Train model using the best values for gamma and cost.

```
svm_model <- svm(class ~ ., data = train, kernel="radial", gamma=0.01, cost=1)
summary(svm_model)
```

```
##
## Call:
## svm(formula = class ~ ., data = train, kernel = "radial", gamma = 0.01,
##     cost = 1)
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  radial
##        cost:  1
##       gamma:  0.01
##
## Number of Support Vectors:  74
##
##  ( 37 37 )
##
##
## Number of Classes:  2
##
## Levels:
##  2 4
```

Predict test cases.

```
svm_predict <- predict(svm_model, test)
table(svm_predict, test$class)
```

```
##
## svm_predict  2  4
##           2 95  5
##           4  3 48
```