svm_hw.R

alexaubrey

Mon May 28 10:36:13 2018

```
library(e1071)
library(caret)
```

```
## Warning: package 'caret' was built under R version 3.4.4
```

```
## Loading required package: lattice
```

```
## Loading required package: ggplot2
```

```
## Warning in as.POSIXlt.POSIXct(Sys.time()): unknown timezone 'zone/tz/2018c.
## 1.0/zoneinfo/America/Chicago'
```

```
data <- read.csv('http://archive.ics.uci.edu/ml/machine-learning-databases/tic-tac-toe/tic-tac-toe.data',
                 col.names = c('top-left-square', 'top-middle-square', 'top-right-square', 'middle-left-square',
                               'middle-middle-square','middle-right-square','bottom-left-square','bottom-middle-square',
                               'bottom-right-square', 'Class'))

### Change each char to a num
change_to_nums <- function(dp) {
  if (!is.na(dp)) {
    if (dp == 'x') {
      return(-1)
    }
    else if (dp == 'o') {
      return(0)
    }
    else if (dp == 'b') {
      return(1)
    }
    else {
      return(dp)
    }
  }
}

#apply the function to each item...
data <- as.data.frame(apply(data, c(1,2), change_to_nums))

#Ensure the transformed columns are numeric
data[,c(1:9)] <- sapply(data[,c(1:9)], as.numeric)

n <- dim(data)[1]

t1 = sample(1:957, n*.8)
t2 = setdiff(1:957, t1)
train = subset(data[t1,])
test = subset(data[t2,], select =-Class)

cl = data[t2,]$Class

y = train$Class
x = subset(train, select=-Class)

P_model <- train(x,y, method="svmPoly", tuneLength=5,
                 trControl=trainControl(method='repeatedcv', number=10, repeats=10))

L_model <- train(x,y, method="svmLinear", tuneLength=5,
                 trControl=trainControl(method='repeatedcv', number=10, repeats=10))

R_model <- train(x,y, method="svmRadial", tuneLength=5,
                 trControl=trainControl(method='repeatedcv', number=10, repeats=10))
```

```
## svmPoly gave us best results
max(P_model$results[4])
```

```
## [1] 0.990053
```

```
max(L_model$results[2])
```

```
## [1] 0.6601162
```

```
max(R_model$results[3])
```

```
## [1] 0.9465362
```

```
pred <- predict(P_model, test)
confusionMatrix(pred, cl)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction negative positive
##   negative       72        0
##   positive        0      120
##
##                Accuracy : 1
##                  95% CI : (0.981, 1)
##     No Information Rate : 0.625
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 1
##  Mcnemar's Test P-Value : NA
##
##             Sensitivity : 1.000
##             Specificity : 1.000
##          Pos Pred Value : 1.000
##          Neg Pred Value : 1.000
##              Prevalence : 0.375
##          Detection Rate : 0.375
##    Detection Prevalence : 0.375
##       Balanced Accuracy : 1.000
##
##        'Positive' Class : negative
##
```