# STAT 775: Machine Learning
# HW 08

Terence Henriod

April 21, 2015

**Abstract**

In this assignment, we use Support Vector Machines (SVM) to classify hand-written digits that are typically hard to discern from one another for classifiers.

# 1 Exercise 01

## 1.1 Problem Statement

Using the zip.data data set from the ESL website and perform classification using Support Vector Machines. Perform binary classification on 7s vs. 9s, 3s vs. 5s, and 0s vs. 8s. You may use a library since implementing the SVM training algorithm is complex.

## 1.2 Results

The SVM method performed pretty well. Without fiddling with which kernel type was used too much or any parameter values, classification performance was comparable to the better methods we have explored this semester. It appeared that the linear kernel was the best kernel.

### 1.2.1 The Tables

The confusion matrices for classification are shown below.

7s vs. 9s:

| Actual/Prediction | 7 | 9 | % Correct |
|---|---|---|---|
| 7 | 136 | 11 | 92.5 |
| 9 | 2 | 175 | 98.9 |
| Overall | | | 96.0 |

3s vs. 5s:

| Actual/Prediction | 3 | 5 | % Correct |
|---|---|---|---|
| 3 | 152 | 14 | 91.6 |
| 5 | 13 | 147 | 91.9 |
| Overall | | | 91.7 |

0s vs. 8s:

| Actual/Prediction | 0 | 8 | % Correct |
|---|---|---|---|
| 0 | 351 | 8 | 97.8 |
| 8 | 15 | 151 | 91.0 |
| Overall | | | 95.6 |

### 1.2.2 The Code

The following R code was used to perform classification:

```
###############
# STAT775: Machine Learning
#
# HW08
# Exercise 01
#
# Using the zip.data data set from the ESL website, use Support Vector Machines
# (SVM) to perfrom classification on traditionally hard to differentiate digits:
# 7s vs. 9s, 3s vs. 5s, and 0s vs. 8s.
###############
```

```
#
# Initial Setup
#
setwd("C:/Users/Terence/Documents/GitHub/STAT775/HW07")

#
# Data Cleaning
#
DATA.PATH <- "../DataSets/zip.data/"
ZIP.TRAIN.FILE.NAME <- paste0(DATA.PATH, "zip.train")
ZIP.TEST.FILE.NAME <- paste0(DATA.PATH, "zip.test")

construct.classification.frame <- function(data, targets) {
  return(
    data.frame(
      'target' = targets,
      'prediction' = rep(0, length(targets)),
      data
    )
  )
}

get.data <- function(
  train.file, test.file, class.1, class.2, num.principle.components = 50) {
  # Args:
  #   train.file:
  #   test.file:
  #   class.1:
  #   class.2
  #   num.principle.components:
  #

  train <- read.table(train.file)
  train <- subset(train, V1 == class.1 | V1 == class.2)
  for (i in 1:nrow(train)) {
    train[i, 1] <- if (train[i, 1] == class.1) {1} else {2}
  }

  test <- read.table(test.file)
  test <- subset(test, V1 == class.1 | V1 == class.2)
  for (i in 1:nrow(test)) {
    test[i, 1] <- if (test[i, 1] == class.1) {1} else {2}
  }

  pca.model <- prcomp(train[, -1])

  train[, -1] <- predict(pca.model, train[, -1])
  train <- train[, 1:(num.principle.components + 1)]
  train <- construct.classification.frame(
    targets = train[, 1],
    data = train[, -1]
  )
```

```
  test[, -1] <- predict(pca.model, test[, -1])
  test <- test[, 1:(num.principle.components + 1)]
  test <- construct.classification.frame(
    targets = test[, 1],
    data = test[, -1]
  )

  return(list(
    train = train,
    test = test
  ))
}

#
# Data Presentation and Evaluation
#

compute.confusion.matrix <- function(predictions, targets, num.classes = 2) {
  confusion.matrix <- matrix(0, nrow = num.classes + 1, ncol = num.classes + 1)
  for (i in 1:length(predictions)) {
    confusion.matrix[targets[[i]], predictions[[i]]] <-
      confusion.matrix[targets[[i]], predictions[[i]]] + 1
  }
  confusion.matrix[num.classes + 1, num.classes + 1] <-
    sum(diag(confusion.matrix)) / sum(confusion.matrix)
  for (i in 1:num.classes) {
    confusion.matrix[i, num.classes + 1] <-
      confusion.matrix[i, i] / sum(confusion.matrix[i, 1:num.classes])
    confusion.matrix[num.classes + 1, i] <-
      confusion.matrix[i, i] / sum(confusion.matrix[1:num.classes, i])
  }
  return(confusion.matrix)
}

#
# Main
#
library('e1071')

tests <- list(
  list(7, 9),
  list(3, 5),
  list(0, 8)
)

test <- tests[[1]]

for (test in tests) {
  data <- get.data(
    train.file = ZIP.TRAIN.FILE.NAME,
    test.file = ZIP.TEST.FILE.NAME,
    class.1 = test[[1]],
    class.2 = test[[2]],
    num.principle.components = 50
```

```
  )

  svm.model <- svm(
    x = data$train[, -(1:2)],
    y = data$train[, 1],
    type = 'nu-classification',
    kernel = 'linear'
  )
  data$test$prediction <- (predict(svm.model, data$test[, -(1:2)]))

  print(compute.confusion.matrix(
    predictions = data$test$prediction,
    targets = data$test$target,
    num.classes = 2
  ))
}
```