AMS 580 Quiz 10 SVM

Srinivasa Phani Madhav

30/04/2024

Load packages

```
if (!requireNamespace("tidyverse")) install.packages('tidyverse')
```

```
## Loading required namespace: tidyverse
```

```
if (!requireNamespace("caret")) install.packages('caret')
```

```
## Loading required namespace: caret
```

```
if (!requireNamespace("kernlab")) install.packages('kernlab')
```

```
## Loading required namespace: kernlab
```

```
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.5.0     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.1
## v purrr     1.0.2
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
library(caret)
```

```
## Loading required package: lattice
##
## Attaching package: 'caret'
##
## The following object is masked from 'package:purrr':
##
##     lift
```

```
library(kernlab)
```

```
##
## Attaching package: 'kernlab'
##
## The following object is masked from 'package:purrr':
##
##     cross
```

```
##
## The following object is masked from 'package:ggplot2':
##
##     alpha
```

Question 1

```
banknote <- read.csv("C:/Users/MSP/Downloads/banknote.csv")
banknote <- na.omit(banknote)
str(banknote)
```

```
## 'data.frame':    1372 obs. of  5 variables:
##  $ variance: num  3.622 4.546 3.866 3.457 0.329 ...
##  $ skewness: num  8.67 8.17 -2.64 9.52 -4.46 ...
##  $ curtosis: num  -2.81 -2.46 1.92 -4.01 4.57 ...
##  $ entropy : num  -0.447 -1.462 0.106 -3.594 -0.989 ...
##  $ class   : int  1 1 1 1 1 1 1 1 1 1 ...
```

```
banknote$class = as.factor(banknote$class)

# (a) split the data into training and testing
set.seed(123)
training.samples <- banknote$class %>%
  createDataPartition(p = 0.75, list = FALSE)
train.data  <- banknote[training.samples, ]
test.data <- banknote[-training.samples, ]

banknote$class <- as.factor(banknote$class)
```

Question 2

```
# Fit the model on the training set
set.seed(123)
model <- train(
  class ~., data = train.data, method = "svmLinear",
  trControl = trainControl("cv", number = 10),
  metric = "Kappa", measure.type = "class",
  preProcess = c("center","scale")
  # This will scale all the predictors in the model
  )

# Make predictions on the test data
probabilities <- model %>% predict(test.data)
head(probabilities)
```

```
## [1] 1 1 1 1 1 1
## Levels: 0 1
```

```
# Confusion matrix
confusionMatrix(as.factor(probabilities), as.factor(test.data$class), positive = '1')
```

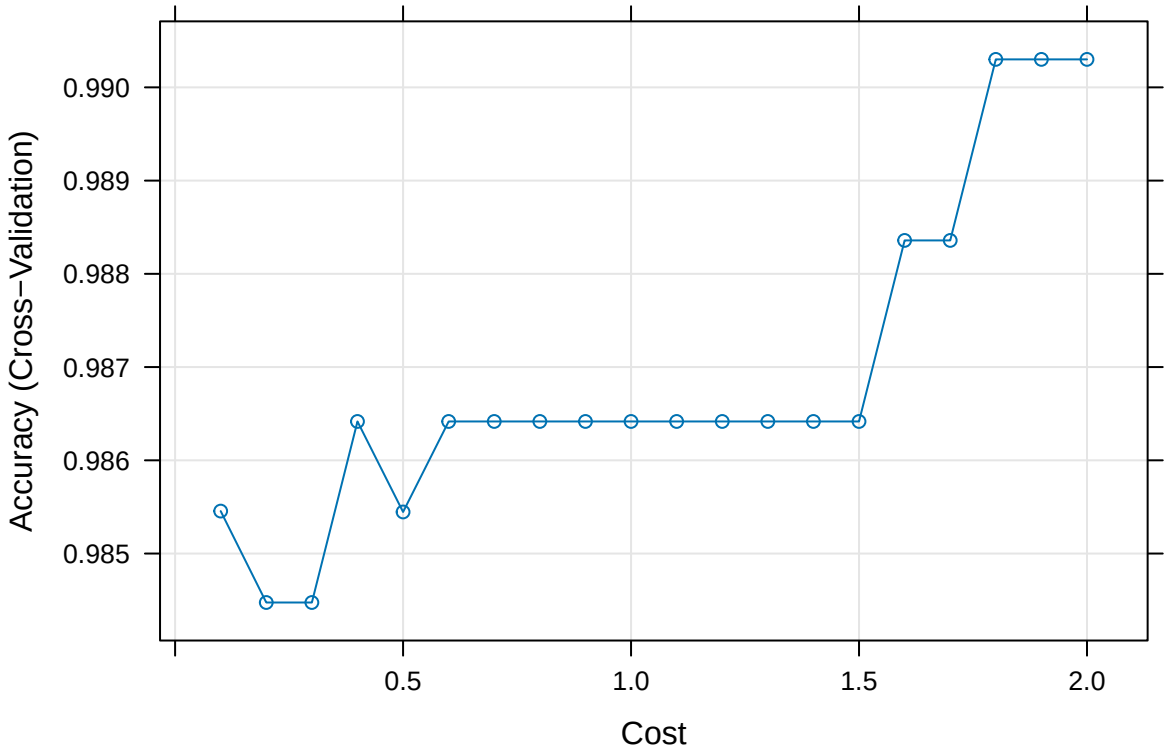
```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 152   7
##          1   0 183
##
```

```
##                Accuracy : 0.9795
##                  95% CI : (0.9583, 0.9917)
##     No Information Rate : 0.5556
##     P-Value [Acc > NIR] : < 2e-16
##
##                   Kappa : 0.9587
##
##  Mcnemar's Test P-Value : 0.02334
##
##             Sensitivity : 0.9632
##             Specificity : 1.0000
##          Pos Pred Value : 1.0000
##          Neg Pred Value : 0.9560
##              Prevalence : 0.5556
##          Detection Rate : 0.5351
##    Detection Prevalence : 0.5351
##       Balanced Accuracy : 0.9816
##
##        'Positive' Class : 1
##
```

Question 3

```r
# Fit the model on the training set
set.seed(123)
model <- train(
  class ~., data = train.data, method = "svmLinear",
  trControl = trainControl("cv", number = 10),
  tuneGrid = expand.grid(C = seq(0.1, 2, length = 20))
  )

# Plot model accuracy vs different values of Cost
plot(model)
```




```
# Print the best tuning parameter C that
# maximizes model accuracy
model$bestTune
```

```
##      C
## 18 1.8
```

```
# Make predictions on the test data
predictions <- model %>% predict(test.data)
# Make predictions on the test data
test.data$linear <- predict(model, newdata = test.data)
# Confusion matrix
# predicted.classes <- ifelse(predictions > 0.5, 1, 0)
head(predictions)
```

```
## [1] 1 1 1 1 1 1
## Levels: 0 1
```

```
confusionMatrix(factor(predictions), factor(test.data$class), positive = '1')
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 151   6
##          1   1 184
##
##                Accuracy : 0.9795
```

```
##                  95% CI : (0.9583, 0.9917)
##     No Information Rate : 0.5556
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.9587
##
##  Mcnemar's Test P-Value : 0.1306
##
##             Sensitivity : 0.9684
##             Specificity : 0.9934
##          Pos Pred Value : 0.9946
##          Neg Pred Value : 0.9618
##              Prevalence : 0.5556
##          Detection Rate : 0.5380
##    Detection Prevalence : 0.5409
##       Balanced Accuracy : 0.9809
##
##        'Positive' Class : 1
##
```

Question 4

```
# Fit the model on the training set
set.seed(123)
model <- train(
  class ~., data = train.data, method = "svmRadial",
  trControl = trainControl("cv", number = 10),
  tuneLength = 10
  )
# Print the best tuning parameter sigma and C that
# maximizes model accuracy
model$bestTune
```

```
##       sigma   C
## 2 0.4006328 0.5
```

```
# Make predictions on the test data
predicted.classes <- model %>% predict(test.data)
test.data$radial <- predict(model, newdata = test.data)
# Confusion matrix
# predicted.classes <- ifelse(probabilities > 0.5, 1, 0)
head(predicted.classes)
```

```
## [1] 1 1 1 1 1 1
## Levels: 0 1
```

```
confusionMatrix(factor(predicted.classes), factor(test.data$class), positive = '1')
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 152   0
##          1   0 190
##
##                Accuracy : 1
##                  95% CI : (0.9893, 1)
```

```
##     No Information Rate : 0.5556
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 1
##
##  Mcnemar's Test P-Value : NA
##
##             Sensitivity : 1.0000
##             Specificity : 1.0000
##          Pos Pred Value : 1.0000
##          Neg Pred Value : 1.0000
##              Prevalence : 0.5556
##          Detection Rate : 0.5556
##    Detection Prevalence : 0.5556
##       Balanced Accuracy : 1.0000
##
##        'Positive' Class : 1
##
```

Question 5

```
# Fit the model on the training set
set.seed(123)
model <- train(
  class ~., data = train.data, method = "svmPoly",
  trControl = trainControl("cv", number = 10),
  tuneLength = 4
  )
# Print the best tuning parameter sigma and C that
# maximizes model accuracy
model$bestTune
```

```
##    degree scale    C
## 29      2     1 0.25
```

```
# Make predictions on the test data
predicted.classes <- model %>% predict(test.data)
test.data$poly <- predict(model, newdata = test.data)
# Confusion matrix
confusionMatrix(factor(predicted.classes), factor(test.data$class), positive = '1')
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 152   0
##          1   0 190
##
##                Accuracy : 1
##                  95% CI : (0.9893, 1)
##     No Information Rate : 0.5556
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 1
##
##  Mcnemar's Test P-Value : NA
```

```
##
##             Sensitivity : 1.0000
##             Specificity : 1.0000
##          Pos Pred Value : 1.0000
##          Neg Pred Value : 1.0000
##              Prevalence : 0.5556
##          Detection Rate : 0.5556
##    Detection Prevalence : 0.5556
##       Balanced Accuracy : 1.0000
##
##        'Positive' Class : 1
##
```

6

Overall accuracy rate:

Linear: 0.9795

Radial basis kernel: 1

Polynomial kernel: 1

Here we can observe that the radial basis as well as the polynomial kernel both tend to give the most accuracy (1) followed by the linear kernel with an accuracy of 0.9795.

7

```
ensemble_preds <- apply(test.data[, c("linear", "radial", "poly")], 1, function(x) {
  unlist(names(sort(table(x), decreasing = TRUE)))[1]
})
confusionMatrix(factor(ensemble_preds), factor(test.data$class), positive = '1')
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 152   0
##          1   0 190
##
##                Accuracy : 1
##                  95% CI : (0.9893, 1)
##     No Information Rate : 0.5556
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 1
##
##  Mcnemar's Test P-Value : NA
##
##             Sensitivity : 1.0000
##             Specificity : 1.0000
##          Pos Pred Value : 1.0000
##          Neg Pred Value : 1.0000
##              Prevalence : 0.5556
##          Detection Rate : 0.5556
```

```
##    Detection Prevalence : 0.5556
##       Balanced Accuracy : 1.0000
##
##        'Positive' Class : 1
##
```