ADMPA- Assignment 3

Steven Pavliga

2025-04-27

QA1. What is the difference between SVM with hard margin and soft margin? (10% of total points)

Hard margins do not allow for any incorrect classifications whereas soft margins allow for them, usually to maintain a more realistic decision boundary. Hard margins can only be applied to linearly separable data, whereas soft margins can also be applied to linearly non-separable data. Additionally, there are some instances (such as outliers) where even though hard margins are technically possible, it wouldn't make the most sense given the general trend of the data. Soft margins also tend to generalize better, however hard margins may be better suited in some contexts.

QA2. What is the role of the cost parameter, C, in SVM (with soft margin) classifiers? (10% of total points)

The cost parameter C is a regularization parameter that balances training error and margin. A smaller value of C will allow for more incorrect classifications in order to maintain larger margins. A smaller value of C better allows constraints to be ignored, increasing the sizes of the margins. The opposite is also true, where increasing C will decrease the size of the margins. A lower C will typically result in less overfitting but underfitting may start to become a concern after a point.

QA3. Will the following perceptron (not pictured) be activated (2.8 is the activation threshold) (10% of total points)

(.1 x 0.8) + (11.1 x -0.2)

0.08 + -2.22 = -2.14

2.8 (activation threshold) > -2.14, perceptron will not be activated

QA4. What is the role of alpha, the learning rate in the delta rule? (10% of total points)

The value of alpha determines how the weights are adjusted with each update, increasing the learning rate (higher alpha) or decreasing it (lower alpha). A higher value of alpha will result in a higher learning rate (the model learning faster) but can risk overfitting. A lower alpha will result in a smoother, more stable learning rate but a slower process overall, often requiring more updates.

QB1. Build a linear SVM regression model to predict Sales based on all other attributes ("Price", "Advertising", "Population", "Age", "Income" and "Education"). Hint: use caret train() with method set to "svmLinear". What is the R-squared of the model? (15 % of total points)

```r
library(ISLR)
library(dplyr)

##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union

library(glmnet)

## Warning: package 'glmnet' was built under R version 4.4.2

## Loading required package: Matrix

## Loaded glmnet 4.1-8

library(caret)

## Loading required package: ggplot2

## Loading required package: lattice

library(rpart)
library(kernlab)

##
## Attaching package: 'kernlab'

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

library(neuralnet)

## Warning: package 'neuralnet' was built under R version 4.4.3

##
## Attaching package: 'neuralnet'

## The following object is masked from 'package:dplyr':
##
##     compute

set.seed(123)

Carseats_Filtered <- Carseats %>% select("Sales", "Price",
"Advertising","Population","Age","Income","Education")
```

```
SVMInTrain <- createDataPartition(y = Carseats_Filtered$Sales, p = 0.7, list
= FALSE)
SVMTrain <- Carseats_Filtered[SVMInTrain,]
SVMTest <- Carseats_Filtered[-SVMInTrain,]

TControl <- trainControl(method = "repeatedcv", number = 10, repeats = 3)

SVMLinear <- train(Sales ~ Price + Advertising + Population + Age + Income +
Education,
                   data = SVMTrain,
                   method = "svmLinear",
                   trControl = TControl,
                   preProcess = c("center", "scale"),
                   tuneLength = 10)

print(SVMLinear)
```

```
## Support Vector Machines with Linear Kernel
##
## 281 samples
##   6 predictor
##
## Pre-processing: centered (6), scaled (6)
## Resampling: Cross-Validated (10 fold, repeated 3 times)
## Summary of sample sizes: 253, 253, 252, 253, 253, 253, ...
## Resampling results:
##
##   RMSE      Rsquared   MAE
##   2.282738  0.3454715  1.824389
##
## Tuning parameter 'C' was held constant at a value of 1
```

The R-squared is 0.3454715.

QB2. Customize the search grid by checking the model's performance for C parameter of 0.1,.5,1 and 10 using 2 repeats of 5-fold cross validation. (15% of total points)

```
set.seed(123)

TControl <- trainControl(method = "repeatedcv", number = 5, repeats = 2)

GridExpand <- expand.grid(C = c(0.1, 0.5, 1, 10))

SVMLinearGrid <- train(Sales ~ Price + Advertising + Population + Age +
Income + Education,
                       data = SVMTrain,
                       method = "svmLinear",
                       trControl = TControl,
                       preProcess = c("center", "scale"),
                       tuneGrid = GridExpand)
```

```
print(SVMLinearGrid)
```

```
## Support Vector Machines with Linear Kernel
##
## 281 samples
##   6 predictor
##
## Pre-processing: centered (6), scaled (6)
## Resampling: Cross-Validated (5 fold, repeated 2 times)
## Summary of sample sizes: 225, 225, 224, 225, 225, 224, ...
## Resampling results across tuning parameters:
##
##   C     RMSE      Rsquared   MAE
##    0.1  2.309559  0.3241794  1.835261
##    0.5  2.310987  0.3232364  1.837227
##    1.0  2.310416  0.3239224  1.836032
##   10.0  2.309694  0.3243030  1.834689
##
## RMSE was used to select the optimal model using the smallest value.
## The final value used for the model was C = 0.1.
```

The lowest RMSE came from a cost parameter of 0.1. The highest R-squared value came from a cost parameter of 10. The lowest MAE also came from 10. While the cost parameter of 10 appeared to be the best performing, all four values were comparable to each other.

QB3. Train a neural network model to predict Sales based on all other attributes (“Price”, “Advertising”, “Population”, “Age”, “Income” and “Education”). Hint: use caret train() with method set to “nnet”. What is the R-square of the model with the best hyper parameters (using default caret search grid) – hint: don’t forget to scale the data. (15% of total points)

```
set.seed(123)

TControl <- trainControl(method = "repeatedcv", number = 10, repeats = 3)

NeuralNetwork <- train(Sales ~ Price + Advertising + Population + Age +
Income + Education,
                       data = SVMTrain,
                       method = "nnet",
                       trControl = TControl,
                       preProcess = c("center", "scale"),
                       trace = FALSE)

## Warning in nominalTrainWorkflow(x = x, y = y, wts = weights, info =
trainInfo,
## : There were missing values in resampled performance measures.

print(NeuralNetwork)
```

```
## Neural Network
##
## 281 samples
##   6 predictor
##
## Pre-processing: centered (6), scaled (6)
## Resampling: Cross-Validated (10 fold, repeated 3 times)
## Summary of sample sizes: 253, 253, 253, 253, 253, 253, ...
## Resampling results across tuning parameters:
##
##   size  decay  RMSE      Rsquared   MAE
##   1     0e+00  7.056239        NaN  6.509619
##   1     1e-04  7.056239        NaN  6.509619
##   1     1e-01  7.056498  0.3281850  6.509897
##   3     0e+00  7.056239        NaN  6.509619
##   3     1e-04  7.056240  0.1685868  6.509619
##   3     1e-01  7.056411  0.3276987  6.509805
##   5     0e+00  7.056239        NaN  6.509619
##   5     1e-04  7.056240  0.1220451  6.509619
##   5     1e-01  7.056379  0.3285734  6.509771
##
## RMSE was used to select the optimal model using the smallest value.
## The final values used for the model were size = 1 and decay = 1e-04.
```

The RMSE and MAE values are too close to each other to definitively evaluate. The R-squared value for 5 neurons and a decay of 1e-01 appears to be the highest at 0.3285734.

QB4. Consider the following input: • Sales=9 • Price=6.54 • Population=124 • Advertising=0 • Age=76 • Income= 110 • Education=10

What will be the estimated Sales for this record using the above neuralnet model? (15% of total points)

```r
set.seed(123)

QB4NewData <- data.frame(
  Price = 6.54,
  Advertising = 0,
  Population = 124,
  Age = 76,
  Income = 110,
  Education = 10
)

QB4Prediction <- predict(NeuralNetwork, newdata = QB4NewData)

print(QB4Prediction)
```

```
## 1
## 1
```