Assignment 1-ADM

Chathurani Ekanayake

2023-10-22

Part A

QA1. What is the main purpose of regularization when training predictive models?

The main purpose of regularization when training predictive models is to **improve generalization performance**. Generalization performance refers to how well a model performs on new data that it has not seen during training.

Regularization works by preventing the model from overfitting the training data. Overfitting occurs when a model learns the training data too well and is unable to generalize to new data.

Regularization can be implemented in a variety of ways, but two of the most common methods are L1 and L2 regularization. L1 regularization penalizes the absolute values of the model parameters, while L2 regularization penalizes the squared values of the model parameters.

Both L1 and L2 regularization can help to improve generalization performance by making the model more robust to noise in the training data and by preventing the model from learning complex patterns that are unlikely to generalize to new data.

QA2. What is the role of a loss function in a predictive model? And name two common loss functions for regression models and two common loss functions for classification models.

The role of a loss function in a predictive model is to measure how well the model's predictions match the actual values. The loss function is used to train the model by adjusting the model's parameters to minimize the loss function.

loss functions for regression models

Mean squared error (MSE): This loss function calculates the average squared difference between the predicted values and the actual values.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

where:

- y_i is the actual target value.
- $\hat{y}_i$ is the predicted value.
- n is the number of data points.

Mean absolute error (MAE): This loss function calculates the average absolute difference between the predicted values and the actual values.

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

loss functions for classification models

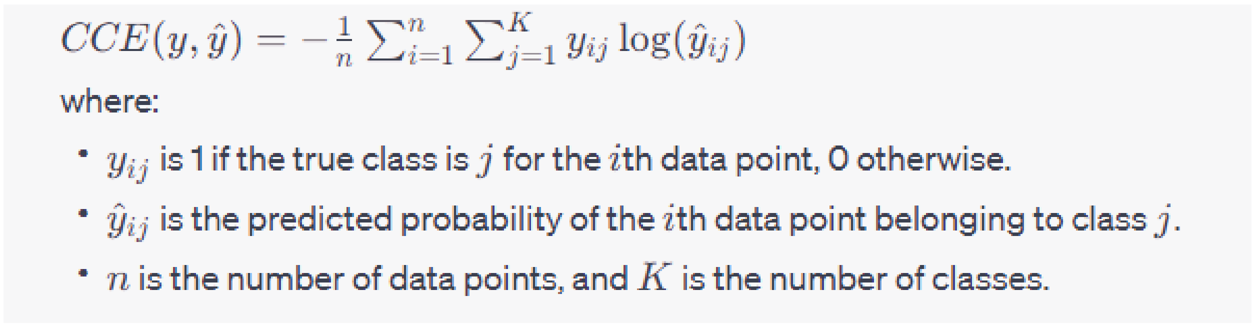
Binary Cross-Entropy Loss (Log Loss): This is the most common loss function for binary classification models. It measures the average cross-entropy between the predicted and actual probabilities.

$$BCE(y, \hat{y}) = -\frac{1}{n}\sum_{i=1}^{n}[y_i \log(\hat{y}_i) + (1 - y_i)\log(1 - \hat{y}_i)]$$

where:

- y_i is the true binary label.
- $\hat{y}_i$ is the predicted probability of belonging to class 1.
- n is the number of data points.

Categorical Cross-Entropy Loss: this is used for to multi-class classification models. It measures the average cross-entropy between the predicted and actual probability distributions.



$$CCE(y,\hat{y}) = -\frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{K} y_{ij}\log(\hat{y}_{ij})$$

where:

- y_{ij} is 1 if the true class is j for the ith data point, 0 otherwise.
- $\hat{y}_{ij}$ is the predicted probability of the ith data point belonging to class j.
- n is the number of data points, and K is the number of classes.

QA3. Consider the following scenario. You are building a classification model with many hyper parameters on a relatively small dataset. You will see that the training error is extremely small. Can you fully trust this model? Discuss the reason.

No. We cannot fully trust the classification model although it has extremely small training error on a relatively small dataset. This is because the model may be overfitting the training data. Overfitting is a phenomenon where a machine learning model learns the training data too well and is unable to generalize to new data. This is because the model has learned to memorize the training data instead of learning the underlying patterns in the data. This can happen when the model has too many hyperparameters or when the training dataset is too small.

The following actions can be taken to avoid the risk of overfitting

- Use a validation set to evaluate the model's performance on new data.
- Use regularization techniques, such as L1 or L2 regularization, to penalize the model for having too many weights.
- Use simpler models with fewer hyperparameters.
- Collect more training data.

QA4. What is the role of the lambda parameter in regularized linear models such as Lasso or Ridge regression models?

The lambda parameter controls the trade-off between bias and variance. Bias is the error that occurs when the model is too simple and cannot learn the underlying patterns in the data. Variance is the error that occurs when the model is too complex and overfits the training data. The lambda parameter controls the strength of the regularization. A higher lambda value will lead to a simpler model with more bias but less variance. A lower lambda value will lead to a more complex model with less bias but more variance.

The optimal lambda value depends on the specific dataset and the problem being solved. It is important to tune the lambda parameter to find the value that gives the best model performance.

- **Lasso regression**: Lasso regression uses L1 regularization, which penalizes the absolute values of the model coefficients. This can lead to some of the coefficients being shrunk to zero, which can help to select important features and reduce overfitting. The lambda parameter controls the strength of the L1 regularization.
- **Ridge regression**: Ridge regression uses L2 regularization, which penalizes the squared values of the model coefficients. This does not lead to coefficients being shrunk to zero, but it does shrink them all towards zero to some extent. This can help to reduce overfitting and improve the generalization performance of the model. The lambda parameter controls the strength of the L2 regularization.

Reference

- https://www.analyticsvidhya.com/blog/2022/06/understanding-loss-function-in-deep-learning/#:~:text=Q1.,cost%20function%20in%20deep%20learning%3F
- https://blogs.brain-mentors.com/loss-functions-in-deep-learning-explained-with-math/
- https://pub.towardsai.net/ridge-and-lasso-regression-51705b608fb9

Part B

This part of the assignment involves building generalized linear regression models to answer a number of questions. We will use the Carseats dataset that is part of the ISLR package (you need to install and load the library). We may also need the following packages: caret, dplyr and glmnet.

```
library(ISLR)
library(dplyr)
library(glmnet)
library(caret)
```

Load the carsetas datset

```
Data <- Carseats
```

Selecting the necessary attributes out of the Carseats dataset.

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

checking for null values

```
colMeans(is.na(Carseats_filtered))

##       Sales       Price Advertising  Population         Age      Income
##           0           0           0           0           0           0
##   Education
##           0
```

Checking the structure of the dataset

```
str(Carseats_filtered)

## 'data.frame':    400 obs. of  7 variables:
##  $ Sales      : num  9.5 11.22 10.06 7.4 4.15 ...
##  $ Price      : num  120 83 80 97 128 72 108 120 124 124 ...
##  $ Advertising: num  11 16 10 4 3 13 0 15 0 0 ...
##  $ Population : num  276 260 269 466 340 501 45 425 108 131 ...
##  $ Age        : num  42 65 59 55 38 78 71 67 76 76 ...
##  $ Income     : num  73 48 35 100 64 113 105 81 110 113 ...
##  $ Education  : num  17 10 12 14 13 16 15 10 10 17 ...
```

Checking the summary of the dataset

```
summary(Carseats)

##      Sales          CompPrice       Income        Advertising
##  Min.   : 0.000   Min.   : 77   Min.   : 21.00   Min.   : 0.000
##  1st Qu.: 5.390   1st Qu.:115   1st Qu.: 42.75   1st Qu.: 0.000
##  Median : 7.490   Median :125   Median : 69.00   Median : 5.000
##  Mean   : 7.496   Mean   :125   Mean   : 68.66   Mean   : 6.635
##  3rd Qu.: 9.320   3rd Qu.:135   3rd Qu.: 91.00   3rd Qu.:12.000
##  Max.   :16.270   Max.   :175   Max.   :120.00   Max.   :29.000
##    Population        Price        ShelveLoc        Age          Education
##  Min.   : 10.0   Min.   : 24.0   Bad   : 96   Min.   :25.00   Min.   :10.0
##  1st Qu.:139.0   1st Qu.:100.0   Good  : 85   1st Qu.:39.75   1st Qu.:12.0
##  Median :272.0   Median :117.0   Medium:219   Median :54.50   Median :14.0
##  Mean   :264.8   Mean   :115.8                Mean   :53.32   Mean   :13.9
##  3rd Qu.:398.5   3rd Qu.:131.0                3rd Qu.:66.00   3rd Qu.:16.0
##  Max.   :509.0   Max.   :191.0                Max.   :80.00   Max.   :18.0
##  Urban       US
##  No :118   No :142
##  Yes:282   Yes:258
##
##
##
##
```

QB1. Build a Lasso regression model to predict Sales based on all other attributes ("Price", "Advertising", "Population", "Age", "Income" and "Education"). What is the best value of lambda for such a lasso model? (Hint1: Do not forget to scale your input attributes – you can use the caret preprocess() function to scale and center the data. Hint 2: glment library expect the input attributes to be in the matrix format. You can use the as.matrix() function for converting)

Data Normalization

```
set.seed(456)
Normalized_Carsets<-preProcess(Carseats_filtered[,-1],method =
c("scale","center"))
Norm_car <- predict(Normalized_Carsets,Carseats_filtered)
```

Data Transformation

##Input attributes-Matrix format

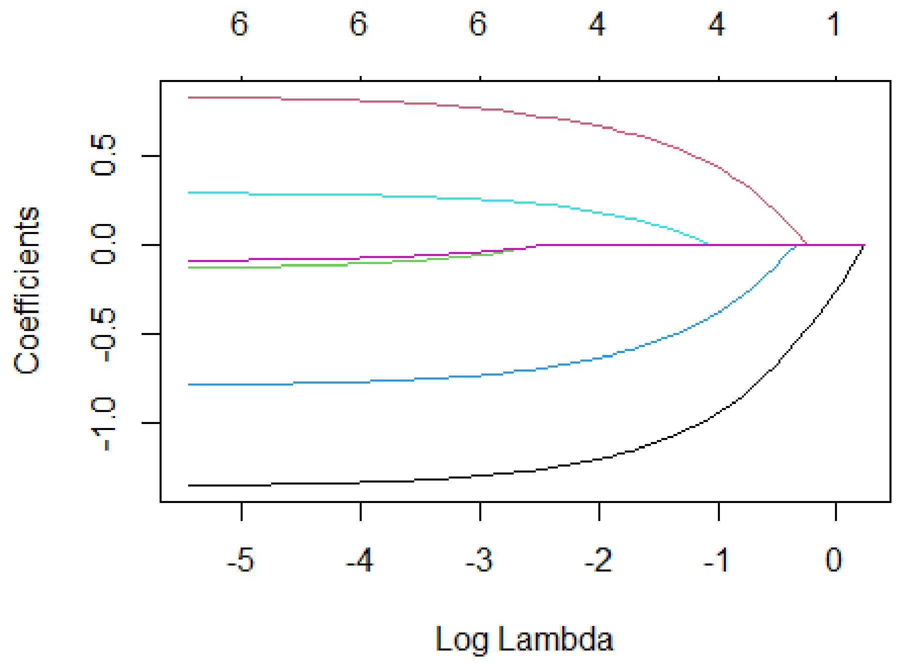
```
train_A <-as.matrix(Norm_car[2:7])
```

##Target variable

```
train_B <- as.numeric(Norm_car$Sales)
```

Building lasso regression model

```
Lasso <- glmnet(train_A,train_B,alpha = 1)
plot(Lasso,xvar="lambda")
```



```
set.seed(789)
LR <- cv.glmnet(train_A, train_B, data = Norm_car, nfolds = 5, alpha = 1)
LR

##
## Call:  cv.glmnet(x = train_A, y = train_B, nfolds = 5, data = Norm_car,
alpha = 1)
##
## Measure: Mean-Squared Error
##
##      Lambda Index Measure     SE Nonzero
## min 0.00431    62   5.181 0.3295       6
## 1se 0.28326    17   5.493 0.2863       4
```

Through lambda.min we can know the lambda value that is optimal and minimizes the cross validation mean square of the error. # The best value of lambda

```
LR$lambda.min

## [1] 0.004305309
```

According to the above value 0.004 can be considered as the best lambda value for the lasso model which we have built.

Lambda 1SE

```
LR$lambda.1se

## [1] 0.2832606
```

Checking the coefficients that were eliminated

```
coef(LR,s="lambda.min")

## 7 x 1 sparse Matrix of class "dgCMatrix"
##                       s1
## (Intercept)  7.49632500
## Price       -1.35383399
## Advertising  0.82805813
## Population  -0.13061347
## Age         -0.78854992
## Income       0.28931898
## Education   -0.09102484
```

When the lambda.min is 0.004305309 non of the attributes has been eliminated from the model.

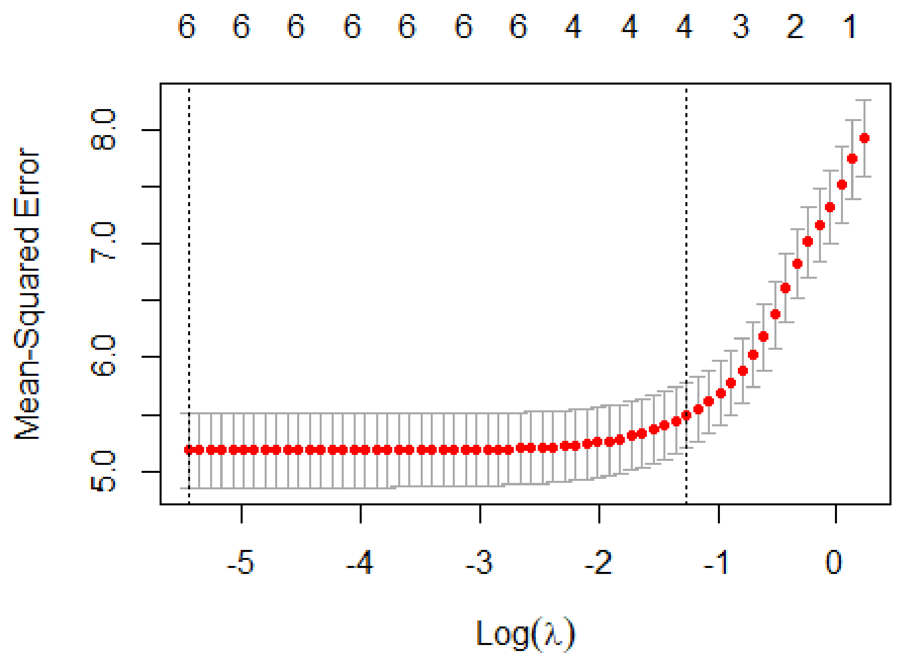
```
coef(LR,s="lambda.1se")

## 7 x 1 sparse Matrix of class "dgCMatrix"
##                       s1
## (Intercept)  7.49632500
## Price       -1.04155049
## Advertising  0.51856113
## Population   .
## Age         -0.47486663
## Income       0.05364767
## Education    .
```

When the lambda.1se is 0.2832606 two of the attributes have been eliminated from the model from the model.

Plotting the Model

```
plot(LR)
```



The Optimal (Min) and 1SE lambda values with respect to the above graph

```
lambda_min <- log(0.004305309)
lambda_min

## [1] -5.447906

lambda_1se <- log(0.2832606)
lambda_1se

## [1] -1.261388
```

QB2. What is the coefficient for the price (normalized) attribute in the best model (i.e. model with the optimal lambda)?

```
coef(LR,s="lambda.min")

## 7 x 1 sparse Matrix of class "dgCMatrix"
##                        s1
## (Intercept)  7.49632500
## Price       -1.35383399
## Advertising  0.82805813
## Population  -0.13061347
## Age         -0.78854992
## Income       0.28931898
## Education   -0.09102484
```

With the optimal lambda (lambda.min) the coefficient of the "price attribute" is -1.35383399.

In this context, a coefficient of -1.35383399 for the "price" attribute suggests that for every one-unit increase in the "price" attribute, the model predicts a decrease of approximately 1.35383399 units in the dependent variable, assuming all other variables remain the same.

QB3. How many attributes remain in the model if lambda is set to 0.01? How that number changes if lambda is increased to 0.1? Do you expect more variables to stay in the model (i.e., to have non-zero coefficients) as we increase lambda?

#lambda = 0.01

```
coef(LR, s="lambda.min")

## 7 x 1 sparse Matrix of class "dgCMatrix"
##                        s1
## (Intercept)  7.49632500
## Price       -1.35383399
## Advertising  0.82805813
## Population  -0.13061347
## Age         -0.78854992
## Income       0.28931898
## Education   -0.09102484
```

When lambda (s) = 0.01 all the input attributes are remaining.

#lambda = 0.1

```
coef(LR, s=0.1)

## 7 x 1 sparse Matrix of class "dgCMatrix"
##                       s1
## (Intercept)  7.4963250
## Price       -1.2447750
## Advertising  0.7007231
## Population   .
## Age         -0.6775428
## Income       0.2139222
## Education    .
```

When lambda (s) is increased to 0.1, the attributes "Education" and "population" have been removed from the model.

Increasing lambda in regularized regression models like Lasso typically leads to fewer variables (attributes) with non-zero coefficients in the model.

As lambda increases, the regularization penalty becomes stronger, encouraging the model to simplify by assigning zero coefficients to less important features. The goal of regularization is to prevent overfitting by reducing the model's complexity and emphasizing the most important features.

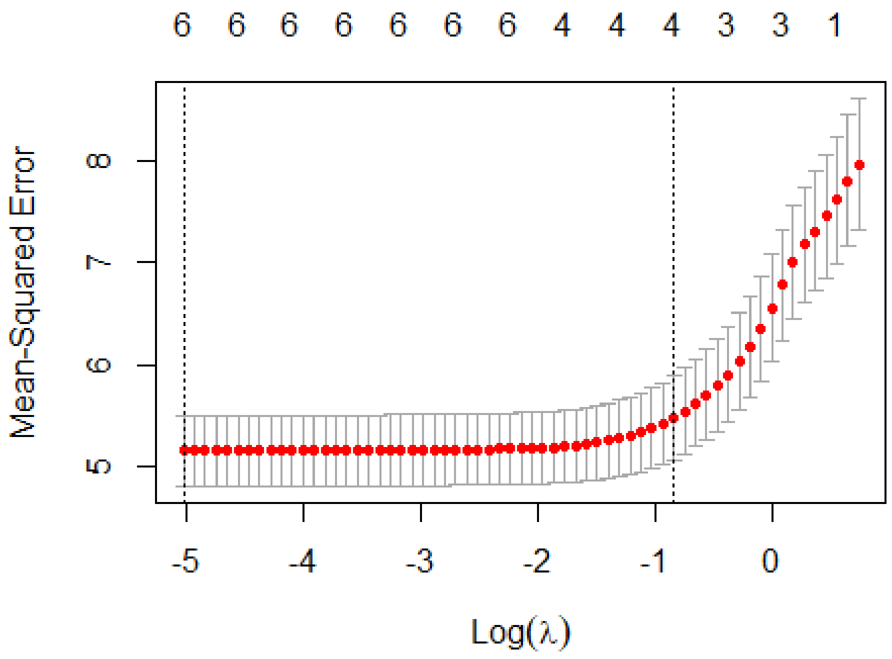
QB4. Build an elastic-net model with alpha set to 0.6. What is the best value of lambda for such a model?

```
LR2 <- cv.glmnet(train_A,train_B,data=Norm_car, nfolds = 10, alpha=0.6)
LR2

##
## Call:  cv.glmnet(x = train_A, y = train_B, nfolds = 10, data = Norm_car,
alpha = 0.6)
##
## Measure: Mean-Squared Error
##
##      Lambda Index Measure     SE Nonzero
## min 0.0065    63   5.156 0.3486       6
## 1se 0.4302    18   5.481 0.4083       4
```

cv.glmnet()plot

```
plot(LR2)
```



```
coef(LR2,s="lambda.min")

## 7 x 1 sparse Matrix of class "dgCMatrix"
##                      s1
## (Intercept)  7.49632500
## Price       -1.35288810
## Advertising  0.82765532
## Population  -0.13077204
## Age         -0.78811113
## Income       0.28949277
## Education   -0.09135756
```

According to the above answer, the none of the attributes are eliminated when alpha is 0.6

Finding the optimal lambda

```
LR2$lambda.min

## [1] 0.006538062
```

Optimal value of lambda when alpha is 0.6 is 0.006538062