Advance Data mining - Assignment 1

Ankith Dasu

10/27/2022

R Markdown

QA1. What is the main purpose of regularization when training predictive models? 10 points

Answer:

Regularization is a method in machine learning that prevents overfitting by simplifying the model. Complex models are prone to picking up random noise from training data, which can obscure the data's patterns. Regularization reduces the impact of noise on the predictive performance of the model. Regularization is classified into two types. l1 (lasso, sum of absolute value) and l2 (Ridge, proportional to the sum of squared residuals) have different effects on model complexity.

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? 10 points

Answer:

Role of Loss Function: It is a method of determining how well a particular algorithm models the given data. The loss function would be high if predictions deviated too much from actual results. Gradually, with the assistance of some optimization function, the loss function learns to reduce prediction error. The machine learning model's overall goal is to reduce the loss function so that the model can generalize and predict better.

Loss function for regression models:

1) Mean Absolute Error(MAE) - L1 loss
2) Mean Square Error (MSE) - L2 loss

There are other loss function in regression like MBE,MSLE and Huber Loss.

Loss function for Classification models:

1)Binary Cross Entropy Loss 2)Sparse_categorical_crossentropy

There are other Loss functions for Classification Model like Hingle Loss, Squared Hinge Loss etc

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.

Solution: Overfitting occurs when models are trained on smaller datasets. By providing a large number of hyperparameters, the model will be perfectly trained on the dataset, capturing all of the information in the dataset; in other words, this is a perfect model for one-time use.

Overfitting causes the model to fail to generalize when the 'test data' or a previously 'unseen data set' is used for evaluation, resulting in a high test error. We cannot trust this model in any way, unless the data is static and the model is designed for a specific task that does not rely on future data. It is preferable to have a large data set with a simple model complexity.

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

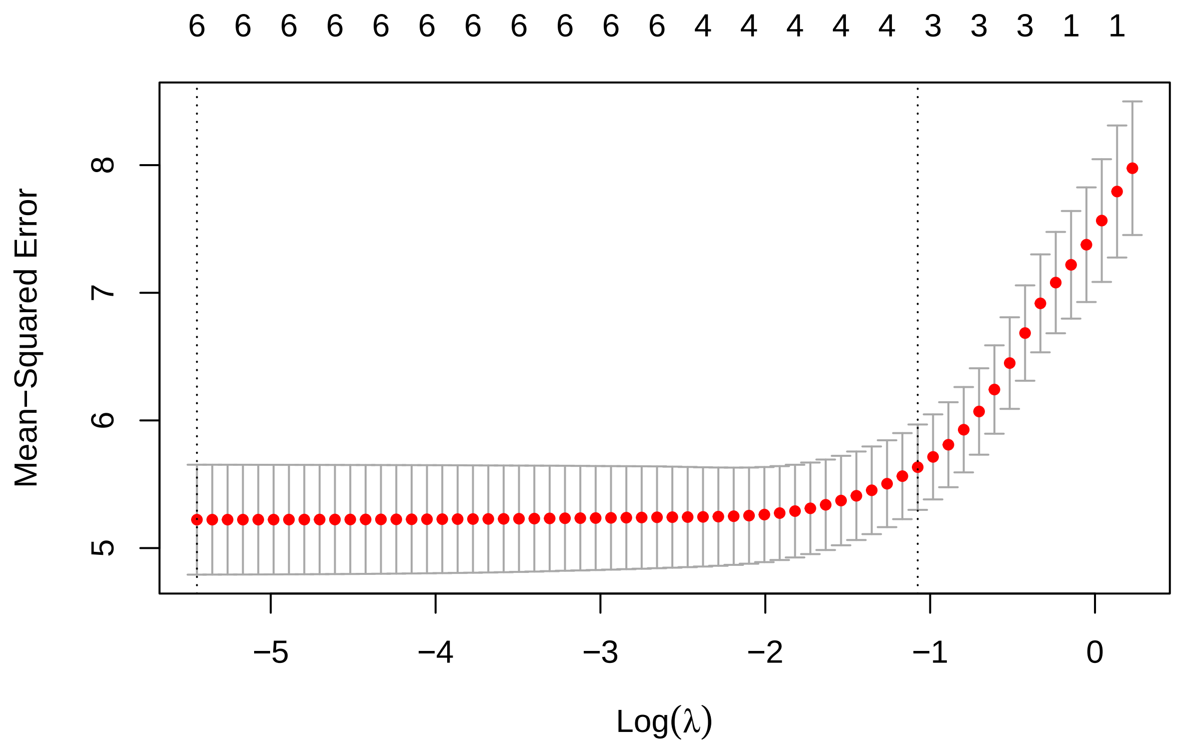
A lambda parameter balances between minimizing the sum squares of the error terms on the training set and reducing the model's coefficients. Higher lambda gives more weight to reducing model coefficients, or higher regularization.

Cross validation is used to select the tuning parameter lambda in these models. When lambda is small, the resulting estimates are essentially least squares. As lambda increases, shrinkage occurs, allowing zero-valued variables to be removed.

`__PART B__`

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)

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


Carseats_data <- Carseats %>% select("Sales", "Price", "Advertising","Population","Age","Income","Educa

#preprocessing the data with center and scale method
Scaled_Carseats_Filtered <- predict(preProcess(Carseats_data[,-1],method=c("scale","center")),Carseats_

# Here we assign the target variable "sales" to y
y<- Scaled_Carseats_Filtered$Sales
# We assign the remaining columns as matrix form to x
x<- as.matrix(Scaled_Carseats_Filtered[,-1])

#Lambda is the Tuning Parameter that controls the bias-variance tradeoff and we estimate its best value
# lambda sequence
lambdas<- 10^seq(2,-3,by=-0.1)

#Building the lasso regression model with k fold cross validation(Considering k=10 here)
Lasso<- cv.glmnet(x,y,alpha=1,standardize=TRUE,nfolds = 10)

#Here, We select the min lambda as the best lambda
best.lambda <- Lasso$lambda.min

plot(Lasso)
```



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

```
# Coefficient for the price with optimal lambda is defined as:
coef(Lasso,s=best.lambda)
```

```
## 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
```

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?

```
# checking the non-zero coefficients when lambda is 0.01
Lambda_Increment.1 <- glmnet(x,y,alpha=1,lambda = 0.01)

#lambda value is 0.1
Lambda_Increment1 <- glmnet(x,y, alpha =1,lambda=0.1)
```

```
coef(Lambda_Increment.1)
```

```
## 7 x 1 sparse Matrix of class "dgCMatrix"
##                      s0
## (Intercept)  7.49632500
## Price       -1.34733223
## Advertising  0.82026088
## Population  -0.12187685
## Age         -0.78190633
## Income       0.28488631
## Education   -0.08502707
```

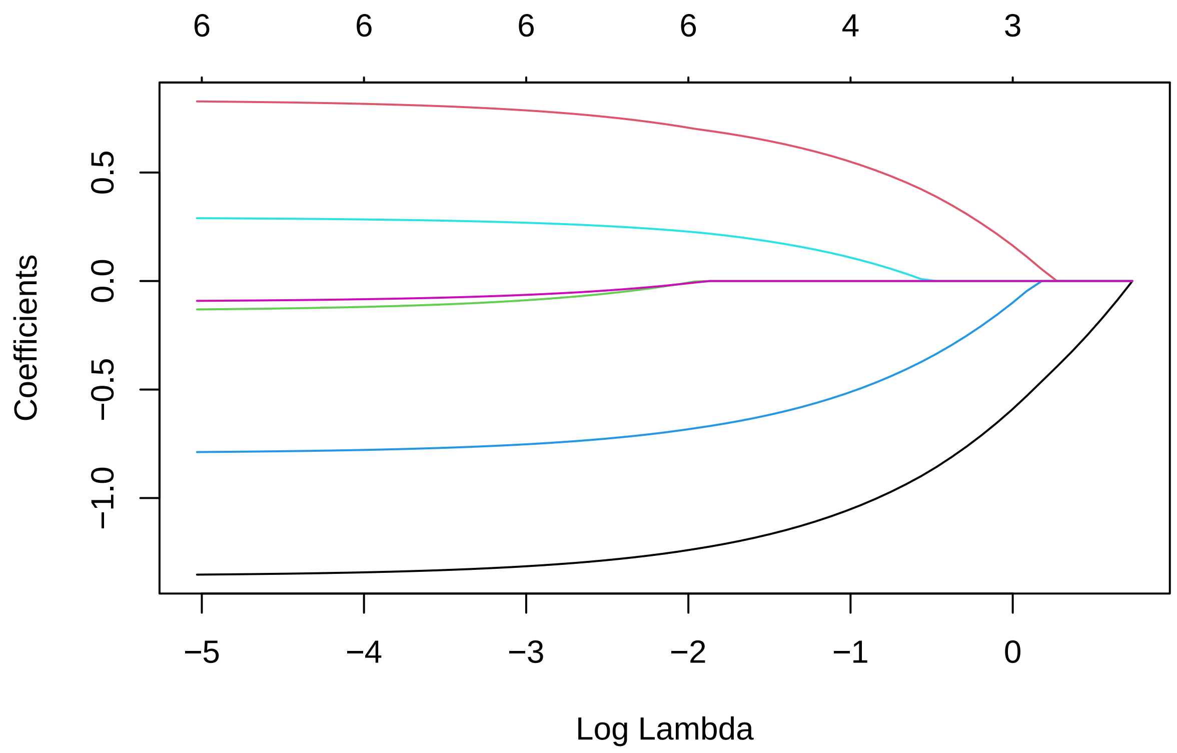
```
# From the above we observe that all the attributes remain unchanged.

coef(Lambda_Increment1)
```

```
## 7 x 1 sparse Matrix of class "dgCMatrix"
##                     s0
## (Intercept)  7.4963250
## Price       -1.2447745
## Advertising  0.7007230
## Population   .
## Age         -0.6775428
## Income       0.2139222
## Education    .
```

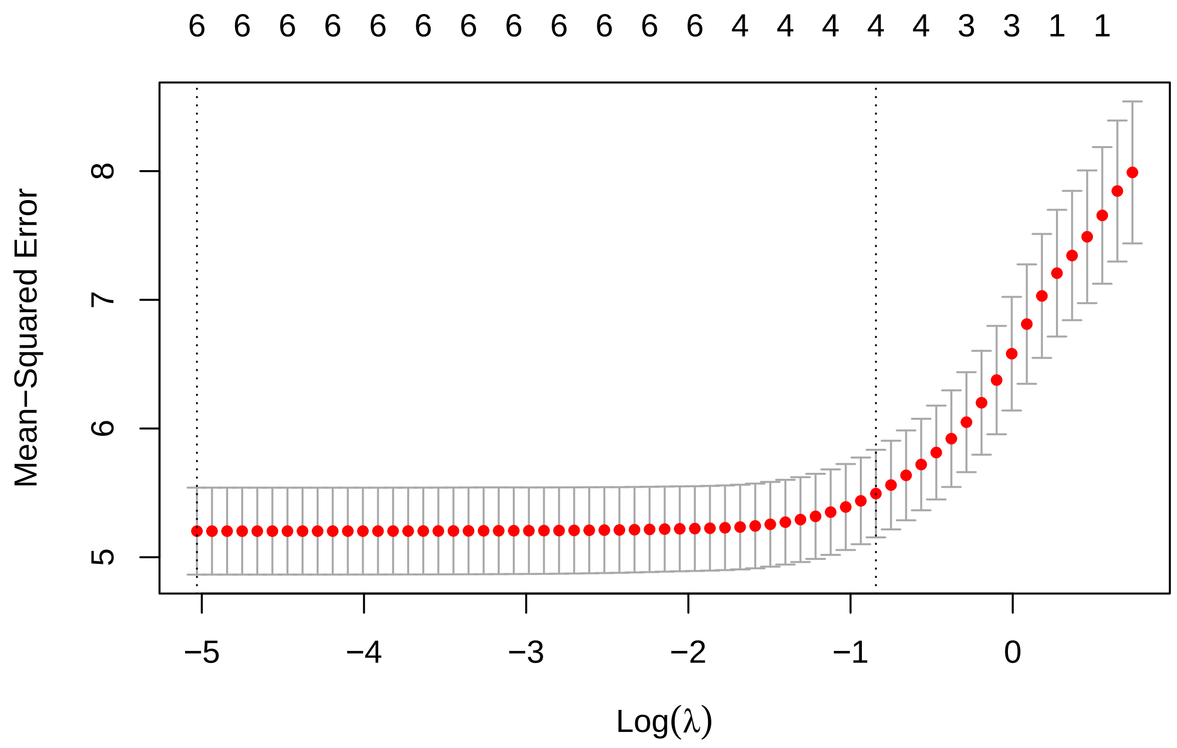
```
# As we have taken the lambda as 0.1, there are two attributes removed because their co-efficeint is ze
```

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

```
elasticnet <- glmnet(x,y,alpha = 0.6)
plot(elasticnet, xvar = "lambda")
```



```
plot(cv.glmnet(x,y,alpha=0.6))
```



```r
lambda_enet <- cv.glmnet(x,y,alpha=0.6)
# Identifying the best lambda
bestlambdaEN <- lambda_enet$lambda.min
print(paste0("Best lambda :",bestlambdaEN))
```

```
## [1] "Best lambda :0.00653806152801921"
```