Regression Kernel

Isabelle Villegas and Jack Asaad

2022-10-23

Classification Assignment

This data given by an airline organization. The actual name of the company is not given due to various purposes which is why the name Invistico airlines.

This dataset consists of the details of customers who have already flown with them. The feedback of the customers on various context and their flight data has been consolidated.

The main purpose of this dataset is to predict whether a future customer would be satisfied with their service given the details of the other parameters values.

Also the airlines need to know on which aspect of the services offered by them have to be emphasized more to generate more satisfied customers.

The link for the data set can be found here: https://www.kaggle.com/datasets/sjleshrac/airlines-customer-satisfaction

```
data <- read.csv("airline_data_1.csv")
data <- data[1:10050, ]
str(data)
```

```
## 'data.frame':    10050 obs. of  23 variables:
##  $ Class                            : chr  "Eco" "Business" "Eco" "Eco" ...
##  $ satisfaction                     : chr  "satisfied" "satisfied" "satisfied" "satisfied" ...
##  $ Gender                           : chr  "Female" "Male" "Female" "Female" ...
##  $ Customer.Type                    : chr  "Loyal Customer" "Loyal Customer" "Loyal Customer" "Loyal
##  $ Age                              : int  65 47 15 60 70 30 66 10 56 22 ...
##  $ Type.of.Travel                   : chr  "Personal Travel" "Personal Travel" "Personal Travel" "Pe
##  $ Flight.Distance                  : int  265 2464 2138 623 354 1894 227 1812 73 1556 ...
##  $ Seat.comfort                     : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ Departure.Arrival.time.convenient: int  0 0 0 0 0 0 0 0 0 0 ...
##  $ Food.and.drink                   : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ Gate.location                    : int  2 3 3 3 3 3 3 3 3 3 ...
##  $ Inflight.wifi.service            : int  2 0 2 3 4 2 2 2 5 2 ...
##  $ Inflight.entertainment           : int  4 2 0 4 3 0 5 0 3 0 ...
##  $ Online.support                   : int  2 2 2 3 4 2 5 2 5 2 ...
##  $ Ease.of.Online.booking           : int  3 3 2 1 2 2 5 2 4 2 ...
##  $ On.board.service                 : int  3 4 3 1 2 5 5 3 4 2 ...
##  $ Leg.room.service                 : int  0 4 3 0 0 4 0 3 0 4 ...
##  $ Baggage.handling                 : int  3 4 4 1 2 5 5 4 1 5 ...
##  $ Checkin.service                  : int  5 2 4 4 4 5 5 5 5 3 ...
##  $ Cleanliness                      : int  3 3 4 1 2 4 5 4 4 4 ...
##  $ Online.boarding                  : int  2 2 2 3 5 2 3 2 4 2 ...
##  $ Departure.Delay.in.Minutes       : int  0 310 0 0 0 0 17 0 0 30 ...
##  $ Arrival.Delay.in.Minutes         : int  0 305 0 0 0 0 15 0 0 26 ...
```

Cleaning Up The Data Set

Cleaning up data set for logistic regression, by converting qualitative columns into factors.

```r
# Factor columns
data$satisfaction<- factor(data$satisfaction)
data$Gender <- factor(data$Gender)

# removing columns that would only have 2 levels
data <- subset (data, select = -4)
data <- subset (data, select = -5)

data$Class <- factor(data$Class)

# Create new cleaned CustomerData data frame for full factoring (linear regression)
CustomerData_factored <- data

# Continue factoring numeric finite columns
#for(i in 4:21) {
#CustomerData_factored[,i] <- factor(CustomerData_factored[,i], levels=c(0,1,2,3,4,5))
#}

# Remove na rows
data_complete <- data[complete.cases(data),]
data <- CustomerData_factored[complete.cases(CustomerData_factored),]
str(data)
```

```
## 'data.frame':    10000 obs. of  21 variables:
##  $ Class                            : Factor w/ 3 levels "Business","Eco",..: 2 1 2 2 2 2 2 2 1 2 ..
##  $ satisfaction                     : Factor w/ 2 levels "dissatisfied",..: 2 2 2 2 2 2 2 2 2 2 ...
##  $ Gender                           : Factor w/ 2 levels "Female","Male": 1 2 1 1 1 2 1 2 1 2 ...
##  $ Age                              : int  65 47 15 60 70 30 66 10 56 22 ...
##  $ Flight.Distance                  : int  265 2464 2138 623 354 1894 227 1812 73 1556 ...
##  $ Seat.comfort                     : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ Departure.Arrival.time.convenient: int  0 0 0 0 0 0 0 0 0 0 ...
##  $ Food.and.drink                   : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ Gate.location                    : int  2 3 3 3 3 3 3 3 3 3 ...
##  $ Inflight.wifi.service            : int  2 0 2 3 4 2 2 2 5 2 ...
##  $ Inflight.entertainment           : int  4 2 0 4 3 0 5 0 3 0 ...
##  $ Online.support                   : int  2 2 2 3 4 2 5 2 5 2 ...
##  $ Ease.of.Online.booking           : int  3 3 2 1 2 2 5 2 4 2 ...
##  $ On.board.service                 : int  3 4 3 1 2 5 5 3 4 2 ...
##  $ Leg.room.service                 : int  0 4 3 0 0 4 0 3 0 4 ...
##  $ Baggage.handling                 : int  3 4 4 1 2 5 5 4 1 5 ...
##  $ Checkin.service                  : int  5 2 4 4 4 5 5 5 5 3 ...
##  $ Cleanliness                      : int  3 3 4 1 2 4 5 4 4 4 ...
##  $ Online.boarding                  : int  2 2 2 3 5 2 3 2 4 2 ...
##  $ Departure.Delay.in.Minutes       : int  0 310 0 0 0 0 17 0 0 30 ...
##  $ Arrival.Delay.in.Minutes         : int  0 305 0 0 0 0 15 0 0 26 ...
```

```r
str(data)
```

```
## 'data.frame':    10000 obs. of  21 variables:
##  $ Class                            : Factor w/ 3 levels "Business","Eco",..: 2 1 2 2 2 2 2 2 1 2 ..
##  $ satisfaction                     : Factor w/ 2 levels "dissatisfied",..: 2 2 2 2 2 2 2 2 2 2 ...
##  $ Gender                           : Factor w/ 2 levels "Female","Male": 1 2 1 1 1 2 1 2 1 2 ...
```

```
##  $ Age                              : int  65 47 15 60 70 30 66 10 56 22 ...
##  $ Flight.Distance                  : int  265 2464 2138 623 354 1894 227 1812 73 1556 ...
##  $ Seat.comfort                     : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ Departure.Arrival.time.convenient: int  0 0 0 0 0 0 0 0 0 0 ...
##  $ Food.and.drink                   : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ Gate.location                    : int  2 3 3 3 3 3 3 3 3 3 ...
##  $ Inflight.wifi.service            : int  2 0 2 3 4 2 2 2 5 2 ...
##  $ Inflight.entertainment           : int  4 2 0 4 3 0 5 0 3 0 ...
##  $ Online.support                   : int  2 2 2 3 4 2 5 2 5 2 ...
##  $ Ease.of.Online.booking           : int  3 3 2 1 2 2 5 2 4 2 ...
##  $ On.board.service                 : int  3 4 3 1 2 5 5 3 4 2 ...
##  $ Leg.room.service                 : int  0 4 3 0 0 4 0 3 0 4 ...
##  $ Baggage.handling                 : int  3 4 4 1 2 5 5 4 1 5 ...
##  $ Checkin.service                  : int  5 2 4 4 4 5 5 5 5 3 ...
##  $ Cleanliness                      : int  3 3 4 1 2 4 5 4 4 4 ...
##  $ Online.boarding                  : int  2 2 2 3 5 2 3 2 4 2 ...
##  $ Departure.Delay.in.Minutes       : int  0 310 0 0 0 0 17 0 0 30 ...
##  $ Arrival.Delay.in.Minutes         : int  0 305 0 0 0 0 15 0 0 26 ...
```

a. Divide into 80/20 train/test

Calculating where in the data set it needs to be split for an 80/20 training and test set and then creating the training set from the first element to the split-th element

```
set.seed(1234)
split <- round(nrow(data)*0.8)
train <- data[1:split, ]
```

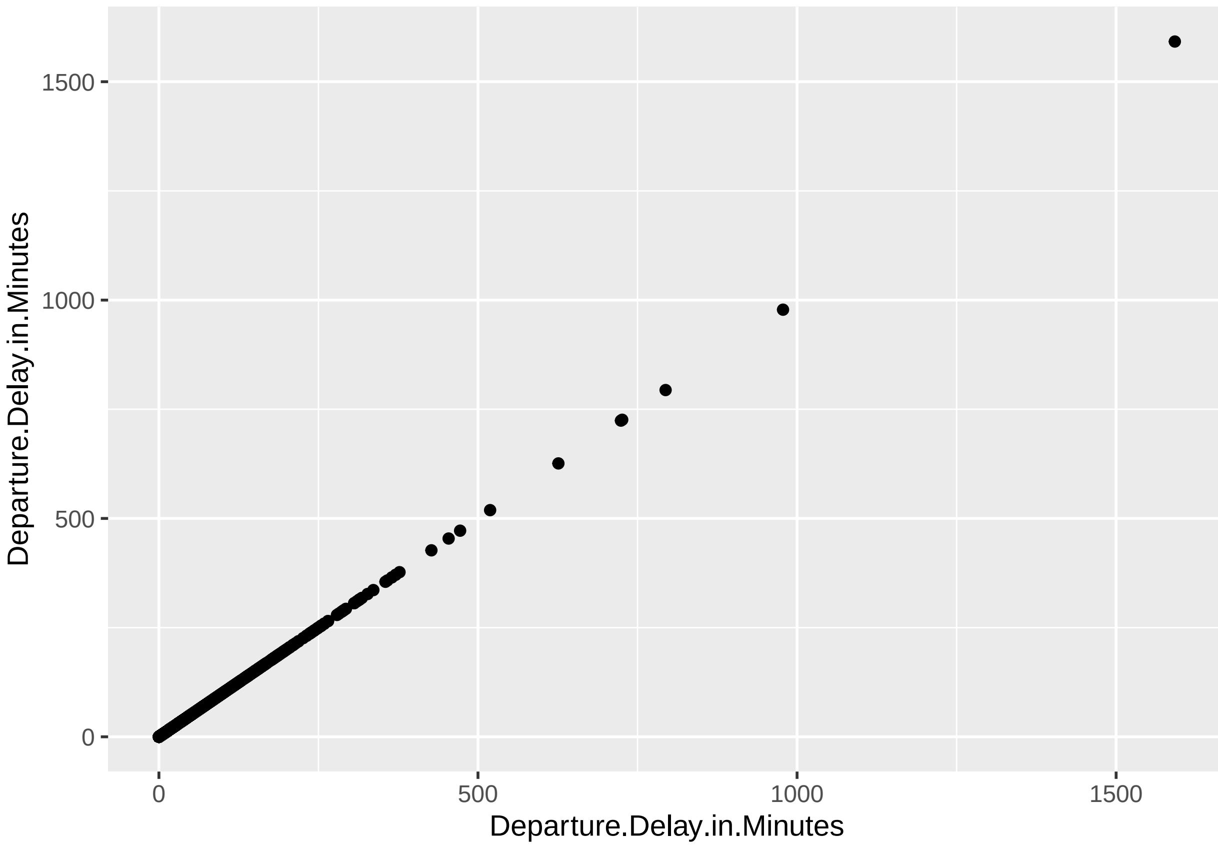
Creating the test data set going from the split point + 1 all the way to the end of the data set

```
test <- data[(split+1):nrow(data),]
```

Explore the training set statistically and graphically

Here we can see that the Departure Delay and how they are spread out. There are more points for a low departure delay and less with a longer one.

```
ggplot(data, aes(x = Departure.Delay.in.Minutes, y = Departure.Delay.in.Minutes)) + geom_point()
```



Perform SVM regression, trying linear, polynomial, and radial kernels with various C and gamma hyperparameters.

```
svm1 <- svm(Departure.Delay.in.Minutes~., data=train, kernel="linear", cost=10, scale=TRUE)
summary(svm1)
```

```
##
## Call:
## svm(formula = Departure.Delay.in.Minutes ~ ., data = train, kernel = "linear",
##     cost = 10, scale = TRUE)
##
##
## Parameters:
##    SVM-Type:  eps-regression
##  SVM-Kernel:  linear
##        cost:  10
##       gamma:  0.04545455
##     epsilon:  0.1
##
##
## Number of Support Vectors:  2904
```

```
pred <- predict(svm1, newdata=test)
cor_svm1 <- cor(pred, test$Departure.Delay.in.Minutes)
mse_svm1 <- mean((pred - test$Departure.Delay.in.Minutes)^2)
```

Trying a polynomial kernal

```r
svm2 <- svm(Departure.Delay.in.Minutes~., data=train, kernel="polynomial", cost=10, scale=TRUE)
summary(svm2)
```

```
##
## Call:
## svm(formula = Departure.Delay.in.Minutes ~ ., data = train, kernel = "polynomial",
##     cost = 10, scale = TRUE)
##
##
## Parameters:
##    SVM-Type:  eps-regression
##  SVM-Kernel:  polynomial
##        cost:  10
##      degree:  3
##       gamma:  0.04545455
##      coef.0:  0
##     epsilon:  0.1
##
##
## Number of Support Vectors:  3903
```

```r
pred <- predict(svm2, newdata=test)
cor_svm2 <- cor(pred, test$Departure.Delay.in.Minutes)
mse_svm2 <- mean((pred - test$Departure.Delay.in.Minutes)^2)
```

Trying a radial kernal

```r
svm3 <- svm(Departure.Delay.in.Minutes~., data=train, kernel="radial", cost=10, gamma=1, scale=TRUE)
summary(svm3)
```

```
##
## Call:
## svm(formula = Departure.Delay.in.Minutes ~ ., data = train, kernel = "radial",
##     cost = 10, gamma = 1, scale = TRUE)
##
##
## Parameters:
##    SVM-Type:  eps-regression
##  SVM-Kernel:  radial
##        cost:  10
##       gamma:  1
##     epsilon:  0.1
##
##
## Number of Support Vectors:  7469
```

```r
pred <- predict(svm3, newdata=test)
cor_svm3 <- cor(pred, test$Departure.Delay.in.Minutes)
mse_svm3 <- mean((pred - test$Departure.Delay.in.Minutes)^2)
```

Tuning the Hyperparameters

```r
#set.seed(1234)
#une.out <- tune(svm, Departure.Delay.in.Minutes~., data=data, kernel="radial",
 #                ranges=list(cost=c(0.1,1,10,100,1000),
  #                           gamma=c(0.5,1,2,3,4)))
```

```
#summary(tune.out)
#svm4 <- svm(Departure.Delay.in.Minutes~., data=train, kernel="radial", cost=100, gamma=0.5, scale=TRUE
#pred <- predict(svm4, newdata=test)
#cor_svm4 <- cor(pred, test$Departure.Delay.in.Minutes)
#mse_svm4 <- mean((pred - test$Departure.Delay.in.Minutes)^2)
```