Test Errors

Yamuna Dhungana

Let’s assume we have a dataset with ‘n’ observations and we’re planning to use ‘K’ folds for cross-validation. In this approach, the test set consists of ‘n/K’ observations, while the remaining ‘n - n/K’ observations make up the training set. Importantly, there is no overlap between the test sets in each fold.

For instance, if we have a total of 1000 observations and choose ‘K’ to be 5, each test set in a fold contains 200 observations without replacement. Subsequently, we calculate the model error for each fold using the training data, and then these errors are averaged. This process is known as K-fold cross-validation.

Some of the advantages of K-fold cross-validation are as follows:

K-fold cross-validation provides robust estimates of test error. It mitigates the influence of variations in the composition of training and validation sets, resulting in more reliable performance evaluation.

It helps us assess the model’s generalization performance, reducing the risk of overestimating the test error rate for the model fit on the entire dataset.

However, a drawback of this method is that it can be computationally intensive due to the need to fit and evaluate the model K times, where K represents the number of folds.

To illustrate this approach, we will now use it to estimate the test error of a logistic regression model that utilizes income and balance as predictors to forecast the likelihood of ‘default’.

```r
library(ISLR)
data("Default")
# head(Default)

model0 <- glm(default~balance+income, data = Default, family = binomial)
summary(model0)
```

```
##
## Call:
## glm(formula = default ~ balance + income, family = binomial,
##     data = Default)
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.154e+01  4.348e-01 -26.545  < 2e-16 ***
## balance      5.647e-03  2.274e-04  24.836  < 2e-16 ***
## income       2.081e-05  4.985e-06   4.174 2.99e-05 ***
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2920.6  on 9999  degrees of freedom
## Residual deviance: 1579.0  on 9997  degrees of freedom
## AIC: 1585
##
```

```
## Number of Fisher Scoring iterations: 8
```

Following model fitting, we determined that both the ‘balance’ and ‘income’ variables exhibit statistical significance.

```
# Splitting data in the ration of 3:2

n <- dim(Default)[1]
set.seed(16489)
num <- sample(1:n, size= round(n/1.5), replace = FALSE)
train.data <- Default[num,]
valid.data <- Default[-num,]
summary(train.data$default)
```

```
##   No  Yes
## 6446  221
```

```
summary(valid.data$default)
```

```
##   No  Yes
## 3221  112
```

b(i)

We utilized the `set.seed` function with a value of 16489 when partitioning the data into training and validation sets.

Subsequently, we fitted multiple logistic regression models exclusively using the training data.

```
model1 <- glm(default~balance+income, data = train.data, family = binomial)
summary(model1)
```

```
##
## Call:
## glm(formula = default ~ balance + income, family = binomial,
##     data = train.data)
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.140e+01  5.319e-01 -21.440  < 2e-16 ***
## balance      5.629e-03  2.786e-04  20.207  < 2e-16 ***
## income       1.699e-05  6.108e-06   2.783  0.00539 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1940.4  on 6666  degrees of freedom
## Residual deviance: 1051.4  on 6664  degrees of freedom
## AIC: 1057.4
##
## Number of Fisher Scoring iterations: 8
```

A logistic regression model for predicting default status was constructed based on income and balance using the training dataset.

In addition, we generated predictions for the default status of individuals in the validation set. This was achieved by computing the posterior probability of default for each individual and categorizing them as ‘default’ if the posterior probability exceeded 0.5.

```
actualdef <- as.numeric(valid.data$default)-1
probs <- round(predict(model1, newdata= valid.data, type = "response"))
preds <- rep(0, length(probs))
for(i in 1:length(preds)){
  if(probs[i]>0.5){
    preds[i]=1
  }
}
```

In this context, the prediction of default status is represented using dummy coding. A value of 1 signifies the default status, while a value of 0 signifies non-default status.

The validation set error is then computed as the fraction of observations in the validation set that have been misclassified.

```
TPR_FPR <- function(con){
  Accuracy=100*((con[1,1]+con[2,2])/sum(con))
  TPR=100*(con[2,2]/(con[2,2]+con[1,2]))
  FPR=100*(1-con[1,1]/(con[1,1]+con[2,1]))
  return(as.data.frame(rbind(Accuracy,TPR, FPR)))
}
missclassfied_table <-  TPR_FPR(table(preds, actualdef))
colnames(missclassfied_table) <- "Error Rate"

knitr::kable(missclassfied_table, digits = 3,
             caption = "Validation set Error")
```

Table 1: Validation set Error

	Error Rate
Accuracy	97.450
TPR	33.036
FPR	0.310

The test error was determined by assessing the model's performance on the validation set. The model exhibited an accuracy of 97.45%, resulting in an error rate of 2.55%. Additionally, the true positive rate of the model was 33.036%, while the false positive rate was 0.310%.

This process was repeated three times, each time with distinct random splits of the observations into training and validation sets.

```
# Split-1

attach(Default)
set.seed(16489)
# splitval <- c(1.2(0.83),1.6(0.625), 1.8(0.556))
num.1 <- sample(1:n, size= round(n/1.2), replace = FALSE)
train.data.1 <- Default[num.1,]
valid.data.1 <- Default[-num.1,]
summary(train.data.1$default)
```

```
##   No  Yes
## 8059  274
```

```r
summary(valid.data.1$default)
```

```
##   No  Yes
## 1608   59
```

```r
model1.a <- glm(default~balance+income, data = train.data.1, family = binomial)
summary(model1.a)
```

```
##
## Call:
## glm(formula = default ~ balance + income, family = binomial,
##     data = train.data.1)
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.148e+01  4.774e-01 -24.042  < 2e-16 ***
## balance      5.678e-03  2.522e-04  22.514  < 2e-16 ***
## income       1.655e-05  5.438e-06   3.043  0.00234 **
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2410.2  on 8332  degrees of freedom
## Residual deviance: 1300.8  on 8330  degrees of freedom
## AIC: 1306.8
##
## Number of Fisher Scoring iterations: 8
```

```r
actualdef.1 <- as.numeric(valid.data.1$default)-1
probs.1 <- round(predict(model1.a, newdata= valid.data.1, type = "response"))
preds.1 <- rep(0, length(probs.1))
for(i in 1:length(preds.1)){
  if(probs.1[i]>0.5){
    preds.1[i]=1
  }
}

missclassfied_table.1 <-  TPR_FPR(table(preds.1, actualdef.1))
colnames(missclassfied_table.1) <- "Error Rate with 1st split"


####################################################################
# Second split
set.seed(16489)
num.2 <- sample(1:n, size= round(n/1.6), replace = FALSE)
train.data.2 <- Default[num.2,]
valid.data.2 <- Default[-num.2,]
summary(train.data.2$default)
```

```
##   No  Yes
## 6044  206
```

```r
summary(valid.data.2$default)
```

```
##   No  Yes
## 3623  127
```

```r
model1.b <- glm(default~balance+income, data = train.data.2, family = binomial)
summary(model1.b)
```

```
##
## Call:
## glm(formula = default ~ balance + income, family = binomial,
##     data = train.data.2)
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.142e+01  5.505e-01 -20.753   <2e-16 ***
## balance      5.623e-03  2.879e-04  19.532   <2e-16 ***
## income       1.799e-05  6.334e-06   2.841   0.0045 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1811.07  on 6249  degrees of freedom
## Residual deviance:  982.76  on 6247  degrees of freedom
## AIC: 988.76
##
## Number of Fisher Scoring iterations: 8
```

```r
actualdef.2 <- as.numeric(valid.data.2$default)-1
probs.2 <- round(predict(model1.b, newdata= valid.data.2, type = "response"))
preds.2 <- rep(0, length(probs.2))
for(i in 1:length(preds.2)){
  if(probs.2[i]>0.5){
    preds.2[i]=1
  }
}

missclassfied_table.2 <-  TPR_FPR(table(preds.2, actualdef.2))
colnames(missclassfied_table.2) <- "Error Rate with 2nd split"

#####################################################
# Third split
set.seed(16489)
num.3 <- sample(1:n, size= round(n/1.8), replace = FALSE)
train.data.3 <- Default[num.3,]
valid.data.3 <- Default[-num.3,]
summary(train.data.3$default)
```

```
##   No  Yes
## 5379  177
```

```r
summary(valid.data.3$default)
```

```
##   No  Yes
## 4288  156
```

```r
model1.c <- glm(default~balance+income, data = train.data.3, family = binomial)
summary(model1.c)
```

```
##
## Call:
## glm(formula = default ~ balance + income, family = binomial,
##     data = train.data.3)
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.133e+01  5.840e-01 -19.399   <2e-16 ***
## balance      5.628e-03  3.081e-04  18.264   <2e-16 ***
## income       1.406e-05  6.757e-06   2.081   0.0375 *
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1568.36  on 5555  degrees of freedom
## Residual deviance:  850.99  on 5553  degrees of freedom
## AIC: 856.99
##
## Number of Fisher Scoring iterations: 8
```

```r
actualdef.3 <- as.numeric(valid.data.3$default)-1
probs.3 <- round(predict(model1.c, newdata= valid.data.3, type = "response"))
preds.3 <- rep(0, length(probs.3))
for(i in 1:length(preds.3)){
  if(probs.3[i]>0.5){
    preds.3[i]=1
  }
}

missclassfied_table.3 <-  TPR_FPR(table(preds.3, actualdef.3))
colnames(missclassfied_table.3) <- "Error Rate with 3rd split"


tablecombined <- as.data.frame(cbind(missclassfied_table.1, missclassfied_table.2, missclassfied_table.
knitr::kable(tablecombined, digits = 3,
             caption = "Validation set Error")
```

Table 2: Validation set Error

	Error Rate with 1st split	Error Rate with 2nd split	Error Rate with 3rd split
Accuracy	97.361	97.493	97.277
TPR	32.203	33.858	31.410
FPR	0.249	0.276	0.326

The data has been divided into three separate sets, each set with the same seed value. Across all three splits, there is a minimal difference of only 0.1 in accuracy. The true positive rate (TPR) and false positive rate (FPR) also show similar results in these splits.

Now, to a logistic regression model that forecasts the likelihood of default. This model incorporates income, balance, and a binary student status variable. We will estimate the test error for this model using the validation set approach.

```r
set.seed(16489)

num.d <- sample(1:n, size= round(n/2), replace = FALSE)
train.data.d <- Default[num.d,]
valid.data.d <- Default[-num.d,]
summary(train.data.d$default)
```

```
##   No  Yes
## 4838  162
```

```r
summary(valid.data.d$default)
```

```
##   No  Yes
## 4829  171
```

```r
model1.d <- glm(default~balance+income+student, data = train.data.d, family = binomial)
summary(model1.d)
```

```
##
## Call:
## glm(formula = default ~ balance + income + student, family = binomial,
##     data = train.data.d)
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.096e+01  7.113e-01 -15.406   <2e-16 ***
## balance      5.691e-03  3.286e-04  17.319   <2e-16 ***
## income       4.912e-06  1.196e-05   0.411    0.681
## studentYes  -3.520e-01  3.386e-01  -1.040    0.298
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1429.88  on 4999  degrees of freedom
## Residual deviance:  778.45  on 4996  degrees of freedom
## AIC: 786.45
##
## Number of Fisher Scoring iterations: 8
```

```r
actualdef.d <- as.numeric(valid.data.d$default)-1
probs.d <- round(predict(model1.d, newdata= valid.data.d, type = "response"))
preds.d <- rep(0, length(probs.d))
for(i in 1:length(preds.d)){
  if(probs.d[i]>0.5){
    preds.d[i]=1
  }
}

missclassfied_table.d <-  TPR_FPR(table(preds.d, actualdef.d))
colnames(missclassfied_table.d) <- "Error Rate"
knitr::kable(missclassfied_table.d, digits = 3,
             caption = "Validation set Error")
```

Table 3: Validation set Error

	Error Rate
Accuracy	97.380
TPR	33.918
FPR	0.373

```
############################################################

# For clearification
attach(Default)
set.seed(16489)
# splitval <- c(1.2(0.83),1.6(0.625), 1.8(0.556))
num.1d <- sample(1:n, size= round(n/1.2), replace = FALSE)
train.data.1d <- Default[num.1d,]
valid.data.1d <- Default[-num.1d,]
summary(train.data.1d$default)
```

```
##   No  Yes
## 8059  274
```

```
summary(valid.data.1d$default)
```

```
##   No  Yes
## 1608   59
```

```
model1.ad <- glm(default~balance+income+student, data = train.data.1d, family = binomial)
summary(model1.ad)
```

```
##
## Call:
## glm(formula = default ~ balance + income + student, family = binomial,
##     data = train.data.1d)
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.082e+01  5.415e-01 -19.986   <2e-16 ***
## balance      5.765e-03  2.571e-04  22.426   <2e-16 ***
## income      -8.136e-07  9.027e-06  -0.090   0.9282
## studentYes  -6.274e-01  2.594e-01  -2.419   0.0156 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2410.2  on 8332  degrees of freedom
## Residual deviance: 1295.1  on 8329  degrees of freedom
## AIC: 1303.1
##
## Number of Fisher Scoring iterations: 8
```

```
actualdef.1d <- as.numeric(valid.data.1d$default)-1
probs.1d <- round(predict(model1.ad, newdata= valid.data.1d, type = "response"))
preds.1d <- rep(0, length(probs.1d))
for(i in 1:length(preds.1d)){
  if(probs.1d[i]>0.5){
    preds.1d[i]=1
  }
}

missclassfied_table.1d <-  TPR_FPR(table(preds.1d, actualdef.1d))
colnames(missclassfied_table.1d) <- "Error Rate with 1st split"


###################################################################
# Second split
set.seed(16489)
num.2d <- sample(1:n, size= round(n/1.6), replace = FALSE)
train.data.2d <- Default[num.2d,]
valid.data.2d <- Default[-num.2d,]
summary(train.data.2d$default)
```

```
##   No  Yes
## 6044  206
```

```
summary(valid.data.2d$default)
```

```
##   No  Yes
## 3623  127
```

```
model1.bd <- glm(default~balance+income+student, data = train.data.2d, family = binomial)
summary(model1.bd)
```

```
##
## Call:
## glm(formula = default ~ balance + income + student, family = binomial,
##     data = train.data.2d)
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.089e+01  6.343e-01 -17.171   <2e-16 ***
## balance      5.679e-03  2.913e-04  19.494   <2e-16 ***
## income       4.346e-06  1.058e-05   0.411    0.681
## studentYes  -4.856e-01  3.010e-01  -1.614    0.107
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1811.07  on 6249  degrees of freedom
## Residual deviance:  980.18  on 6246  degrees of freedom
## AIC: 988.18
##
## Number of Fisher Scoring iterations: 8
```

```
actualdef.2d <- as.numeric(valid.data.2d$default)-1
probs.2d <- round(predict(model1.bd, newdata= valid.data.2d, type = "response"))
preds.2d <- rep(0, length(probs.2d))
for(i in 1:length(preds.2d)){
  if(probs.2d[i]>0.5){
    preds.2d[i]=1
  }
}

missclassfied_table.2d <-  TPR_FPR(table(preds.2d, actualdef.2d))
colnames(missclassfied_table.2d) <- "Error Rate with 2nd split"

#####################################################
# Third split
set.seed(16489)
num.3d <- sample(1:n, size= round(n/1.8), replace = FALSE)
train.data.3d <- Default[num.3d,]
valid.data.3d <- Default[-num.3d,]
summary(train.data.3d$default)
```

```
##   No  Yes
## 5379  177
```

```
summary(valid.data.3d$default)
```

```
##   No  Yes
## 4288  156
```

```
model1.cd <- glm(default~balance+income+student, data = train.data.3d, family = binomial)
summary(model1.cd)
```

```
##
## Call:
## glm(formula = default ~ balance + income + student, family = binomial,
##     data = train.data.3d)
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.089e+01  6.784e-01 -16.046   <2e-16 ***
## balance      5.676e-03  3.117e-04  18.211   <2e-16 ***
## income       2.667e-06  1.145e-05   0.233    0.816
## studentYes  -4.021e-01  3.255e-01  -1.235    0.217
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1568.36  on 5555  degrees of freedom
## Residual deviance:  849.48  on 5552  degrees of freedom
## AIC: 857.48
##
## Number of Fisher Scoring iterations: 8
```

```
actualdef.3d <- as.numeric(valid.data.3d$default)-1
probs.3d <- round(predict(model1.cd, newdata= valid.data.3, type = "response"))
preds.3d <- rep(0, length(probs.3d))
for(i in 1:length(preds.3d)){
  if(probs.3d[i]>0.5){
    preds.3d[i]=1
  }
}
```

```
missclassfied_table.3d <-  TPR_FPR(table(preds.3d, actualdef.3d))
colnames(missclassfied_table.3d) <- "Error Rate with 3rd split"


tablecombined_d <- as.data.frame(cbind(missclassfied_table.1d, missclassfied_table.2d, missclassfied_tal
knitr::kable(tablecombined_d, digits = 3,
             caption = "Validation set Error for clearification")
```

Table 4: Validation set Error for clearification

	Error Rate with 1st split	Error Rate with 2nd split	Error Rate with 3rd split
Accuracy	97.241	97.467	97.322
TPR	30.508	32.283	32.051
FPR	0.311	0.248	0.303

Data partitioning was executed following the previous question's procedure, employing a set seed value of 16489. The data analysis indicated statistical significance for both balance and income. The model achieved an accuracy of 97.400%, corresponding to a test error rate of 2.62%. The true positive rate was 33.9%, and the false positive rate was 0.373%. These results closely resemble the errors observed in the table with three splits from the prior discussion.

It's worth noting that the introduction of the student variable might account for the slight shift in accuracy, although it remains modest. To clarify this, I repeated the same process as in question C, leading to the accuracy mentioned in the table. The error exhibited a comparable difference to question C when contrasted with each other. However, when comparing the error with the same data split, the addition of the student variable resulted in a slight reduction in accuracy, even if only by 0.1. Consequently, it can be concluded that the inclusion of the student variable has a marginal adverse impact on the model's accuracy.

Fitting a logistic regression model that predicts Direction using Lag1 and Lag2.

```
library(ISLR)
data("Weekly")
dim(Weekly)
```

```
## [1] 1089    9
```

```
weeklymodel1=glm(Direction~Lag1+Lag2,data=Weekly, family=binomial)
summary(weeklymodel1)
```

```
##
## Call:
## glm(formula = Direction ~ Lag1 + Lag2, family = binomial, data = Weekly)
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.22122    0.06147   3.599 0.000319 ***
## Lag1        -0.03872    0.02622  -1.477 0.139672
## Lag2         0.06025    0.02655   2.270 0.023232 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1496.2  on 1088  degrees of freedom
## Residual deviance: 1488.2  on 1086  degrees of freedom
## AIC: 1494.2
##
## Number of Fisher Scoring iterations: 4
```

The model did not pick the Lag1 variable as statistically significant.

b.Fit a logistic regression model that predicts Direction using Lag1 and Lag2 using all but the first observation.

```
model.butfirst=glm(Direction~Lag1+Lag2,data=Weekly[-1,], family=binomial)
summary(model.butfirst)
```

```
##
## Call:
## glm(formula = Direction ~ Lag1 + Lag2, family = binomial, data = Weekly[-1,
##     ])
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.22324    0.06150   3.630 0.000283 ***
## Lag1        -0.03843    0.02622  -1.466 0.142683
## Lag2         0.06085    0.02656   2.291 0.021971 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1494.6  on 1087  degrees of freedom
## Residual deviance: 1486.5  on 1085  degrees of freedom
## AIC: 1492.5
##
## Number of Fisher Scoring iterations: 4
```

In this instance, the model no longer considers the 'Lag1' variable as statistically significant, unlike the previous analysis. Utilizing the previously established model, we aim to forecast the direction of the initial observation. This is achieved by predicting that the first observation will exhibit an 'up' direction if the conditional probability of 'Direction="Up"|Lag1, Lag2' exceeds 0.5.

```
firstpred=(predict(model.butfirst, newdata = Weekly[1,],type="response"))
firstpred=ifelse(firstpred > 0.5,"UP","Down")
firstpred
```

```
##    1
## "UP"
```

```
Weekly[1,]$Direction
```

```
## [1] Down
## Levels: Down Up
```

First observation was classified as Up. It was a misclassification.

For each observation in the dataset, where 'n' represents the total number of observations, a series of steps are performed. First, a logistic regression model is fitted using all observations except the current one (ith observation) to forecast the market direction ('Direction') based on the 'Lag1' and 'Lag2' variables. Next, the posterior probability of the market moving up for the ith observation is computed. This posterior probability is then used to make a prediction regarding whether the market will move up or not for the ith observation. Finally, the system checks whether there was an error in predicting the direction for the ith observation. If an error is detected, it is marked as '1'; otherwise, it is marked as '0'.

```
# qestion d and e
dataN <- dim(Weekly)[1]
 misclassifyYes=rep(NA,dataN)
for (i in 1:dataN){
  modL <- glm(Direction~Lag1 +Lag2, data = Weekly[-i,], family = "binomial")
  actualDirection=ifelse(Weekly$Direction[i]=="Up",1,0)
  predL=round(predict(modL,newdata=Weekly[i,],type="response"))
  misclassifyYes[i]=abs(actualDirection - predL)
}
sum(misclassifyYes)
```

```
## [1] 490
```

```
## 490
paste0("Percent Error: ",round(100*(sum(misclassifyYes)/dataN),3))
```

```
## [1] "Percent Error: 44.995"
```

```
## 44.995
```

The loop was executed 1089 times, corresponding to the total number of observations, which was 1089. Out of these, 490 observations were misclassified, resulting in an overall error rate of 4.995%.

```
library(ISLR)
data(Auto)
# head(Auto)
AutoN <- dim(Auto)[1]
# k= 30
valueofK <- function(k){
# folds = AutoN/k
stopsoffolds <- c(0,round(c(1:k)*(AutoN/k)))
for (i in 1:k){
  (stopsoffolds[i+1]-stopsoffolds[i])
}
set.seed(16489)
randomizedindex=sample(1:AutoN,AutoN)
testMSE=rep(NA,k)
for(i in 1:k){
  autotestindex=randomizedindex[((stopsoffolds[i]+1):stopsoffolds[i+1])]
  autotrain=Auto[-autotestindex,]
  autotest=Auto[autotestindex,]

  automodel=glm(mpg~horsepower+I(horsepower^2),data=autotrain)
  pred=predict(automodel,newdata=autotest)

  testMSE[i]=sum((pred-autotest$mpg)^2)/dim(autotest)[1]
}
(testMSE)
sum(testMSE)/k
}

MSE_when_k_5 <- valueofK(5)

MSE_when_k_30  <- valueofK(30)
```

```r
finaltab <- cbind(MSE_when_k_5, MSE_when_k_30)
knitr::kable(finaltab, digits = 3,
             caption = "K-fold cross validation with two different K")
```

Table 5: K-fold cross validation with two different K

MSE_when_k_5	MSE_when_k_30
19.108	19.203

In this analysis, two different cross-validation techniques were employed, specifically 5-fold and 30-fold cross-validation, with a random seed value of 16489. The dataset comprises 392 observations. In each run, one fold was reserved for validation, while the remaining 5 folds (or 30 folds) were used to train the model. A custom function was designed to calculate the Mean Squared Error (MSE) for each of these folds, and the results are presented in the table above.

Comparing the results, it's evident that the MSE for the 30-fold cross-validation appears slightly higher than that of the 5-fold cross-validation. The individual MSE values for the 5 folds are as follows: 17.07849, 17.11979, 19.11285, 22.27647, and 19.95182, with an overall 5-fold MSE of 19.108. In contrast, the 30-fold MSE is 19.203.

This comparison suggests that the 30-fold cross-validation method exhibits relatively lower bias compared to the 5-fold approach. However, it comes at the expense of slightly higher variability in the results.