Activity – Logistic Regression

Learning outcomes: Logistic Regression

After completing this exercise, you should be able to understand and perform below tasks.

1. Building classification model using logistic regression technique.
2. Validating the model results.
3. Handling multicollinearity and dimensionality reduction.
4. Evaluation of error metrics.
5. Applying the models on un-seen data
    a. Splitting data into train and test data sets
    b. Comparing the error metrics
6. Interpretation of the results.
7. ROC curve.

Problem Statement:

The “Bank.txt” file consists of the data related to direct marketing campaigns of a Portuguese banking institution. The marketing campaigns were based on phone calls. Often, more than one contact to the same client was required, to access if the product (bank term deposit) would be (or not) subscribed. The data and attribute description are in the folder.

1. Import the data into R
2. Understand the data and preform required preprocessing steps. Explain the reason for each step.
    - Structure and summary of the data
    - Dealing with missing values
    - Changing and recoding the factor levels for the following attributes
3. Create a new variable "outcome" based on "y" if y="yes" then outcome=1 else outcome=0 and convert it to appropriate data type.

```
##Recode the levels for marital,month and poutcome

Bank$outcome<-ifelse(Bank$y=="yes",1,0)

# Converting the “outcome” column into factor

Bank$outcome <- as.factor(as.character(Bank$outcome))
```

4. Drop the attribute “y” from the data frame.
5. Split the data into train and test datasets

```
###############Data for model building#################

#Split the data into train and test data sets

rows=seq(1,nrow(Bank),1)
```

```
set.seed(123)

trainRows=sample(rows,(70*nrow(Bank))/100)

train = Bank[trainRows,]

test = Bank[-trainRows,]
```

6. Implement the logistic regression model using all attributes and predict the results

```
################Logistic regression model################

LogReg <- glm(outcome ~ ., data=train, family=binomial)

# Predicting on the train data

prob<-predict(LogReg, type="response")

# Considering the threshold as 0.5
pred_class <- ifelse(prob >  0.5, 1, 0)

table(train$outcome,pred_class)
```

7. Identify appropriate error metric for this problem, and compute the values for these metrics on both train and test data.
    - Precision
    - Recall
    - Accuracy

```
# Generating the confusion metric on train  data

conf.mat1 = table(train$outcome,pred_class)

# Calculating the accuracy of the model on train data

accuracy1 = sum(diag(conf.mat1))/sum(conf.mat1)

# Calculating the precision of the model

precision1 = conf.mat1[2,2]/sum(conf.mat1[,2])

# Calculating the recall of the model

recall1 = conf.mat1[2,2]/sum(conf.mat1[2,])

# Predicting on test data
```

```
fitted.results <- predict(LogReg,test,type='response')

fitted.class <- ifelse(fitted.results > 0.5,1,0)

table(test$outcome,fitted.class)

# Generating the confusion metric on test  data

conf.mat2 = table(test$outcome,fitted.class)

# Calculating the accuracy of the model on test data

accuracy2 = sum(diag(conf.mat2))/sum(conf.mat2)

# Calculating the precision of the model

precision2 = conf.mat2[2,2]/sum(conf.mat2[,2])

# Calculating the recall of the model

recall2 = conf.mat2[2,2]/sum(conf.mat2[2,])
```

8. Identify the important attributes using VIF and stepAIC
    - Implement the logistic regression model using these attributes (Hint: Follow steps 6 and 7 on the identified features)

```
## Variable selection

library(car)

vif(LogReg)

library(MASS)

stepAIC(LogReg)

# Building the logistic regression on important features

LogReg_updated <- glm(formula = outcome ~ marital + housing + loan + contact

                           + day + month + duration + campaign + poutcome, family
                           = binomial, data = train)

# Predicting on the train data

prob1 <- predict(LogReg_updated, type="response")

# Considering the threshold as 0.5
```

```
pred_class1 <- ifelse(prob1> 0.5, 1, 0)

table(train$outcome,pred_class1)

# Predicting on the test data

fitted.results1 <- predict(LogReg_updated,test,type='response')

# Considering the threshold as 0.5

fitted.class1 <- ifelse(fitted.results1 > 0.5,1,0)

table(test$outcome,fitted.class1)
```

9. Use ROCR curve to obtain, reasonable cutoff for probabilities and using that probability as a threshold to obtain best set of predictions

```
##ROCR curves..Loading the required libraries

library(ROCR)

library(ggplot2)

# Predicting on the train data

predicted <- predict(LogReg_updated,type="response")

prob <- prediction(predicted, train$outcome)

# Getting the true positive rate and false negative rate

tprfpr <- performance(prob, "tpr", "fpr")

# Plotting the true positive rate and false negative rate based on the threshold value

plot(tprfpr)

str(tprfpr)

# For different threshold values identifying the tpr and fpr

cutoffs <- data.frame(cut=tprfpr@alpha.values[[1]], fpr=tprfpr@x.values[[1]],

                    tpr=tprfpr@y.values[[1]])

# Sorting the data frame in the decreasing order based on tpr

cutoffs <- cutoffs[order(cutoffs$tpr, decreasing=TRUE),]
```

```
head(subset(cutoffs, fpr < 0.2))

# Plotting the true positive rate and false negative rate based based on the cutoff
# increasing from 0.1-1

plot(tprfpr, colorize = TRUE, print.cutoffs.at=seq(0,1,by=0.1), text.adj=c(-0.2,1.7))

tpr <- unlist(slot(tprfpr, "y.values"))

fpr <- unlist(slot(tprfpr, "x.values"))

# creating the data frame with tpr and fpr

roc <- data.frame(tpr, fpr)

# Plotting the graph

ggplot(roc) + geom_line(aes(x = fpr, y = tpr)) +

  geom_abline(intercept=0,slope=1,colour="gray") +

  ylab("Sensitivity") +    xlab("1 - Specificity")
```

10. Identifying the threshold obtained by the ROC curve, Based on the probability threshold predict on the train and test data and calculate the error metrics.

```
#Using the ROC curve, obtain the appropriate threshold of probalility for

## for calculating the error metric

# Consuidering the threshold as 0.11 based on ROC curve

pred_class1 <- ifelse(prob1> 0.11, 1, 0)

table(train$outcome,pred_class1)

# Generate the confusion metrics on train data

conf.mat1 = table(train$outcome,pred_class1)

# Calculating the accuracy, precision and recall on train data

accuracy1 = sum(diag(conf.mat1))/sum(conf.mat1)

precision1 = conf.mat1[2,2]/sum(conf.mat1[,2])

recall1 = conf.mat1[2,2]/sum(conf.mat1[2,])
```

```
# Test results

fitted.results1 <- predict(LogReg_updated,test,type='response')

fitted.class1 <- ifelse(fitted.results1 > 0.11,1,0)

# Generate the confusion metrics on test data

conf.mat2 = table(test$outcome,fitted.class1)

# Calculating the accuracy, precision and recall on test data

accuracy2 = sum(diag(conf.mat1))/sum(conf.mat1)

precision2 = conf.mat1[2,2]/sum(conf.mat1[,2])

recall2 = conf.mat1[2,2]/sum(conf.mat1[2,])
```

Exercise:

Problem Statement:

A large child education toy company which sells edutainment tablets and gaming systems both online and in retail stores wanted to analyze the customer data. They are operating from last few years and maintaining all transactional information data. The given data 'CustomerData.csv' is a sample of customer level data extracted and processed for the analysis from various set of transactional files.

The objectives of today's activity are

- Building a classification model to predict whether the given customer will churn or not churn based on other known factors

Logistic Regression Model:

1. Read the data sets 'CustomerData_Classification.csv' into R.
2. Understand the structure of the data and pre-process the data
    a. Drop the attribute 'CustomerID'
    b. Convert 'City' as factor variable
3. Target attribute is: Churned
4. Convert the attributes to appropriate data type.
5. Split the data into train and test data sets

6. Build logistic regression and interpret the results
7. Generate the error metrics on train and test data
    a. Precision
    b. Recall
    c. Accuracy
8. Evaluation on train & test data
9. Identify the important features and build the logistic regression on these features
10. Study the ROC curve and identify the best threshold values.
11. Generate the evaluation of error metrics on train and test data based on the threshold
    a. Precision
    b. Recall
    c. Accuracy