Credit Card Fraud Detection

This project focuses on identifying fraudulent transactions using credit card data, employing machine learning algorithms to distinguish between legitimate and fraudulent activities. You will learn about data preprocessing, anomaly detection techniques, and the application of machine learning models in R.

Duration: 20 hours

Project Complexity: Medium

Learning Outcome: Understanding of data preprocessing, anomaly detection, and machine learning model implementation in R

Portfolio Worthiness: Yes

Required Pre-requisites:

- Intermediate R programming skills
- Basic understanding of machine learning concepts
- Familiarity with data preprocessing techniques

Resources Required:

- R and RStudio
- Credit card transaction dataset (available on platforms like Kaggle)
- caret, ROCR, e1071, and dplyr packages for machine learning and data manipulation

Real-World Application:

- Enhancing security measures for credit card transactions
- Developing algorithms to detect and prevent fraudulent activities in real-time

Credit-card-fraud-detection.R

```
# CREDIT CARD FRAUD DETECTION

#-----------------------------------------------------------------------------------------------------------------------
-

# The credit card dataset contains 20000 records and 31 columns.

#-----------------------------------------------------------------------------------------------------------------------
-

# Installing all the required packages

install.packages("dplyr")

install.packages("caret")

install.packages("e1071")

install.packages("ggplot2")

install.packages("caTools")

install.packages("ROSE")

install.packages("smotefamily")

install.packages("rpart")

install.packages("rpart.plot")


# Importing required libraries

library(dplyr)

library(caret)

library(ggplot2)
```

```
library(caTools)

library(ROSE)

library(smotefamily)

library(rpart)

library(rpart.plot)


#Loading the dataset

credit_card<-read.csv("D:/Shalaka(All Data)/Intellipaat/Data Science with R/Data Science with R Part-I/Project-4/creditcard.csv")


#Viewing dataset

View(credit_card)


# Glance at the structure of the dataset

str(credit_card)


# Number of rows & columns

nrow(credit_card)

ncol(credit_card)


# Convert class to a factor variable

credit_card$Class <- factor(credit_card$Class, levels =  c(0,1))


# Get the summary of the data

summary(credit_card)


# Count the missing values

sum(is.null(credit_card))
```

```
# Get the distribution of fraud and legit transactions in the dataset

table(credit_card$Class)


# Get the percentage of fraud and legit transactions in the datasets

prop.table(table(credit_card$Class))


#Pie Chart of credit card transactions

labels <- c("legit","fraud")

labels <- paste(labels, round(100*prop.table(table(credit_card$Class)),2))

labels <- paste(labels ,"%")


par("mar")

par(mar=c(1,1,1,1))

par("mar")


pie(table(credit_card$Class),labels, col = c("orange" , "red"),

   main = "Pie chart of Credit Card Transactions")


#-----------------------------------------------------------------------------------------------------------------------
-


# No model predictions

predictions <- rep.int(0,nrow(credit_card))

predictions <- factor(predictions, levels= c(0,1))


confusionMatrix(data = predictions, reference = credit_card$Class)
```

```
#-------------------------------------------------------------------------------------------------------------------------
-


set.seed(1)

credit_card <- credit_card %>% sample_frac(0.1)


table(credit_card$Class)


ggplot(data = credit_card , aes(x = V1,y = V2 ,col = Class))+

  geom_point()+

  theme_bw()+

  scale_color_manual(values =c('dodgerblue2','red'))


#-------------------------------------------------------------------------------------------------------------------------
----------


#Creating training and test sets for fraud detection model


set.seed(123)


data_sample = sample.split(credit_card$Class,SplitRatio= 0.80)


train_data = subset(credit_card,data_sample == TRUE)


test_data = subset(credit_card, data_sample == FALSE)


dim(train_data)

dim(test_data)
```

```
View(train_data)

View(test_data)

#-------------------------------------------------------------------------------------------------------------------------
---------------


# Random Over-Sample (ROS)


table(train_data$Class)


n_legit <- 22750

new_frac_legit <- 0.50

new_n_total <- n_legit/new_frac_legit


oversampling_result <- ovun.sample(Class ~ . ,

                                   data = train_data,

                                   method = "over",

                                   N = new_n_total,

                                   seed = 123)


oversampled_credit <- oversampling_result$data


table(oversampled_credit$Class)


ggplot(data = oversampled_credit, aes (x = V1, y = V2, col = Class))+

  geom_point(position = position_jitter(width = 0.2))+

  theme_bw()+
```

```
  scale_color_manual(values = c('dodgerblue2','red'))


#-------------------------------------------------------------------------------------------------------------------------------------------
-


# Random Under-Sampling (RUS)


table(train_data$Class)


n_fraud <- 35

new_frac_fraud <- 0.50

new_n_total <- n_fraud/new_frac_fraud


undersampling_result <- ovun.sample(Class ~ .,

                          data = train_data,

                          method = "under",

                          N = new_n_total,

                          seed =123)


undersampled_credit <- undersampling_result$data


table(undersampled_credit$Class)


ggplot(data = undersampled_credit, aes(x = V1, y = V2, col = Class))+

  geom_point()+

  theme_bw()+

  scale_color_manual(values = c('dodgerblue2','red'))
```

```
#-------------------------------------------------------------------------------------------------------------------------------
--------------

# ROS and RUS

n_new <- nrow(train_data)

fraction_fraud_new <-0.50

sampling_result <-ovun.sample(Class ~ .,
                              data = train_data,
                              method = "both",
                              N = n_new,
                              p = fraction_fraud_new,
                              seed =123)

sampled_credit <- sampling_result$data

table(sampled_credit$Class)

prop.table(table(sampled_credit$Class))

ggplot(data = sampled_credit , aes(x = V1, y = V2, col =Class))+
  geom_point(position = position_jitter(width = 0.2))+
  theme_bw()+
  scale_color_manual(values = c('dodgerblue2','red'))

#-------------------------------------------------------------------------------------------------------------------------------
--------
```

```
#Using SMOTE to balance the dataset

table(train_data$Class)

#Set the number of fraud and legitimate cases, and the desired percentage of legitmate cases

n0 <- 22750

n1 <- 35

r0 <- 0.6

#Calculate the values for the dup_size parameter of SMOTE

ntimes <- ((1 - r0)/ r0) *(n0 / n1) - 1

ntimes

smote_output = SMOTE(X = train_data[ , -c(1,31)],

                     target = train_data$Class,

                     K = 5,

                     dup_size = ntimes)

credit_smote <- smote_output$data

colnames(credit_smote)[30] <-"Class"

prop.table(table(credit_smote$Class))
```

```
#Class distribution for original dataset


ggplot(train_data, aes(x = V1, y = V2, color = Class))+
  geom_point()+
  scale_color_manual(values = c('dodgerblue2','red'))


# Class distribution for original dataset


ggplot(credit_smote, aes(x = V1, y = V2, color = Class))+
  geom_point()+
  scale_color_manual(values = c('dodgerblue2','red'))


#-------------------------------------------------------------------------------------------------------------------------
---------


CART_model <- rpart(Class ~ ., credit_smote)


rpart.plot(CART_model, extra = 0 , type = 5, tweak = 1.2)


#Predict fraud Classes
predicted_val <-predict(CART_model, test_data, type ='class')
predicted_val


#Build Confusion Matrix


confusionMatrix(predicted_val, test_data$Class)
```

```
#-------------------------------------------------------------------------------------------------------------------------------
-------------


# Decision tree without SMOTE


CART_model <-rpart(Class~ ., train_data[,-1])


rpart.plot(CART_model,extra = 0, type = 5, tweak = 1.2)


#Predict fraud classes

predicted_val <-predict(CART_model, test_data[-1], type = 'class')


confusionMatrix(predicted_val, test_data$Class)


#-------------------------------------------------------------------------------------------------------------------------------


predicted_val <-predict(CART_model , credit_card[,-1], type = 'class')

confusionMatrix(predicted_val, credit_card$Class)


#-------------------------------------------------------------------------------------------------------------------------------
-
```