IMBALANCED CLASSIFICATION: HANDLING IMBALANCED DATA IN CREDIT CARD FRAUD DETECTION

Report submitted to the SASTRA Deemed to be University
as the requirement for the course

CSE300 - MINI PROJECT

Submitted by

ANUSHA R
(Reg. No.: 122003031, B.Tech. Computer Science & Engineering)
BATTULA TEJASWINI
(Reg. No.: 122003047, B.Tech. Computer Science & Engineering)
S ANJANA PRIYA SWATHI
(Reg. No.: 122003222, B.Tech. Computer Science & Engineering)

January 2022

SCHOOL OF COMPUTING

THANJAVUR, TAMIL NADU, INDIA – 613 401

SASTRA

ENGINEERING · MANAGEMENT · LAW · SCIENCES · HUMANITIES · EDUCATION

DEEMED TO BE UNIVERSITY

(U/S 3 OF THE UGC ACT, 1956)

THINK MERIT | THINK TRANSPARENCY | THINK SASTRA

SCHOOL OF COMPUTING
THANJAVUR – 613 401

Bonafide Certificate

This is to certify that the report titled " **Imbalanced Classification: Handling Imbalanced Classification in Credit Card Fraud Detection**" submitted as a requirement for the course CSE300 - MINI PROJECT for B.Tech. is a bonafide record of the work done by **Ms. Anusha R(Reg No.122003031, B.Tech. Computer Science and Engineering), Ms. Battula Tejaswini (Reg. No.122003047, B.Tech. Computer Science and Engineering), Ms. S Anjana Priya Swathi(Reg. No.122003222, B.Tech. Computer Science and Engineering)** during the academic year 2021-22, in the School of Computing, under my supervision.

Signature of Project Supervisor:

Name with Affiliation: Dr. SARAVANAN P, Asst. Professor III

Date: 09/01/2022

Project Based Work Viva voce held on 18-01-2021

Examiner 1 Examiner 2

ACKNOWLEDGEMENTS

We would like to thank our Honorable Chancellor **Prof. R. Sethuraman** for providing us with an opportunity and the necessary infrastructure for carrying out this project as a part of our curriculum.

We would like to thank our Honorable Vice-Chancellor **Dr. S. Vaidhyasubramaniam** and **Dr. S. Swaminathan**, Dean, Planning & Development, for the encouragement and strategic support at every step of our college life.

We extend our sincere thanks to **Dr. R. Chandramouli**, Registrar, SASTRA Deemed to be University for providing us the opportunity to pursue this project. We extend our heartfelt thanks to **Dr. A. Umamakeswari**, Dean, School of Computing, **Dr. V. S. Shankar Sriram**, Associate Dean, Department of Computer Science and Engineering, **Dr. R. Muthaiah**, Associate Dean, Department of Information Technology and Information & Communication Technology and **Dr. B. Santhi**, Associate Dean, Department of Computer Application.

Our guide **Dr. P. Saravanan**, AP-III, School of Computing was the driving force behind this whole idea from the start. His deep insight in the field and invaluable suggestions helped us in making progress throughout our project work. We also thank the project review panel members for their valuable comments and insights which made this project better. We would like to extend our gratitude to all the teaching and non-teaching faculties of the School of Computing who have either directly or indirectly helped us in the completion of the project. We gratefully acknowledge all the contributions and encouragement from our family and friends resulting in the successful completion of this project.

We thank you all for providing me an opportunity to showcase our skills through project.

LIST OF FIGURES

LIST OF TABLES

ABBREVIATIONS

SVM	Support Vector Machines
ANN	Artificial Neural Network
KNN	K-Nearest Neighbors
LR	Logistic Regression
NB	Naive Bayes
AANN	Auto Associative Neural Network
RO	Random Oversampling
OCC	One Class Classification
CS	Cost Sensitive

ABSTRACT

Credit cards are powerful means of electronic payment by which purchases involve transactions without liquid cash. Off late, since such financial transactions occur online, credit card fraudulences have started reaching new heights. Credit frauds are the criminal use of a victim's personal credentials and their credit standing to gain possession and use their savings without repayment. Though staying skeptical and usage of secure browsers can help prevent this crime, there is still a need for some economic mechanisms to detect and prevent such activities. Dataset having user's credentials will be used for running on various classification algorithms to check the accurate one for fraud detection. Skewedness of one classification results in a severe imbalance and will end in a poor prediction of the minority class. Class imbalance assists a big challenge in detecting the features of such activities and deriving their patterns. Hence, in this paper a comparative study will be done on the detection for handling the imbalanced dataset. Most efficient algorithms will be used for classification.

Keywords: F*raud detection, Exploratory data analysis, Classification algorithms, Imbalanced classification, Performance metrics*

TABLE OF CONTENTS

CHAPTER 1
SUMMARY OF THE BASE PAPER

Table 1.1 Base Paper Details

Title	Imbalanced Classification: Handling Imbalanced Data in Credit Card Fraud Detection
Authors	SARA MAKKI , ZAINAB ASSAGHIR , YEHIA TAHER , RAFIQUL HAQUE, MOHAND-SAÏD HACID1 , AND HASSAN ZEINEDDINE
Journal Name	Open Access Journal
Year of Publishing	2019
Publisher	IEEE
Link	https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8756130
Electronic ISSN	2169 - 3536
DOI	10.1109/ACCESS.2019.2927266

1.1 Work plan

Dataset was first downloaded from the link given in base paper. It was then imported in the R directory. Exploratory data analysis was done to explore more about the features and their statistical details. Since the dataset was huge, it was brought down to 2L and was trained. The same was used to run on various algorithms. After training, a separate set called the test set was used to make predictions and evaluation of results was done based on the outcome after passing test set as input to the algorithms. Fig 1.1.1 depicts the work plan carried out.

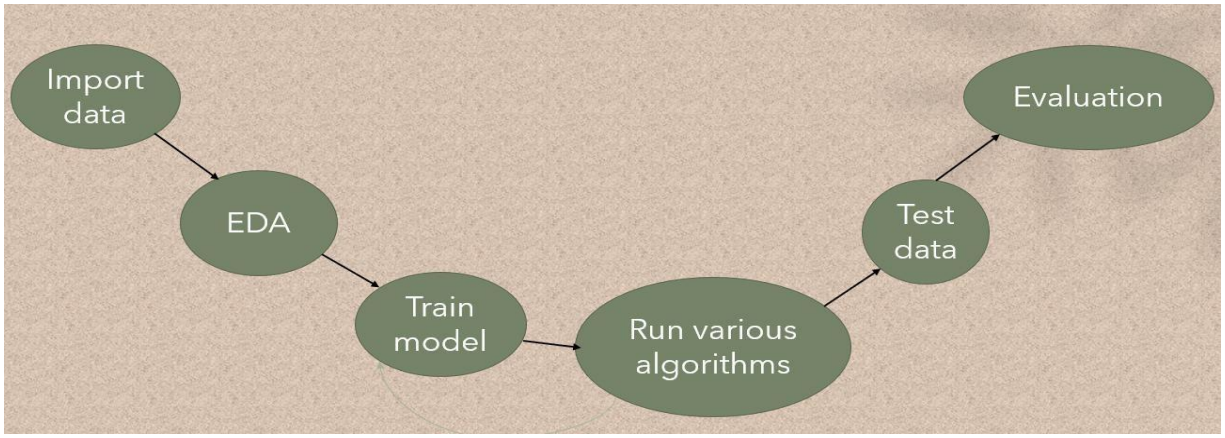

Fig. 1.1.1 Work Plan

1.2 ALOGIRTHMS

C5.0

C5.0 is a type of decision tree algorithm. This algorithm makes use of cross entropy for the calculation of values of the split nodes.

Support Vector Machine

SVM is one of the most versatile algorithms. An optimal hyperplane is found by manipulating the margin that maximizes the distance between the support vectors.

Fig 1.2.1 shows how SVM makes use of hyperplane to classify 2 classes.

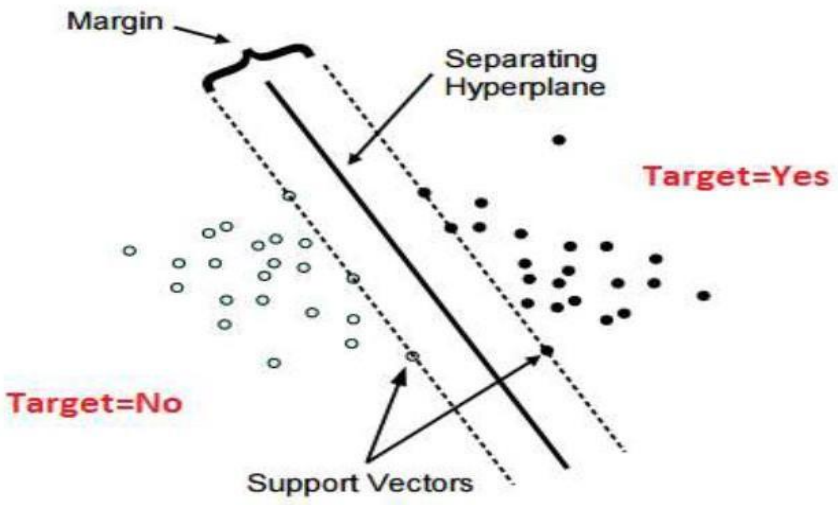

Fig. 1.2.1 SVM

Artificial Neural Network

Artificial neural network built here composes of an input layer with 7 nodes, one hidden layer with 3 nodes and one output layer with a node. Sigmoid/logistic is used as activation function. Rprop is used as the backpropagation algorithm while training the dataset. The weights are adjusted such that classification is accurate. Fig 1.3 shows the network generated using ANN.

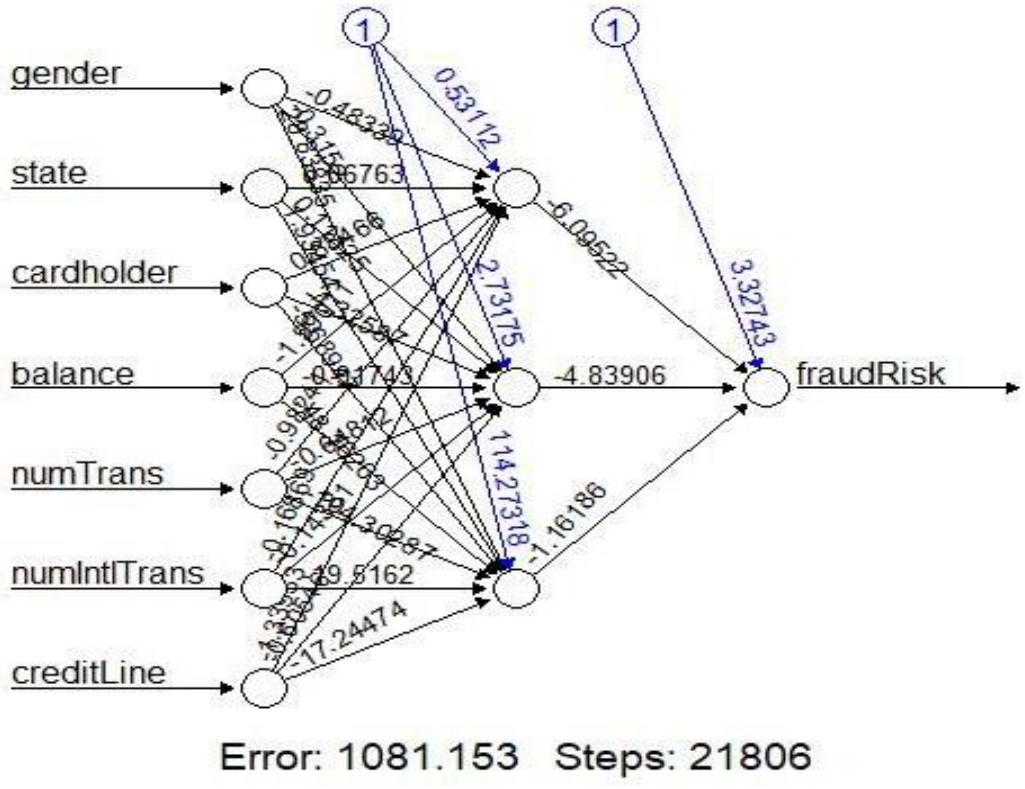

Fig. 1.2.2. ANN

Naive Bayes

NB makes use of the Bayes conditional probability rule for Classification. A class that maximizes the output value is chosen for the given input. Hence values are found such that value of Y is maximized $\mathbf{P(Y|X_1,X_2,X_3…X_n)}$.

Logistic Regression

Logistic regression is used here for binary classification problem. The outcome canhence be either a categorical value or discrete. The output will be a probabilistic value ranging between 0 and 1.

K-Nearest Neighbor

Knn is again one such algorithm that can be used for both classification and regression where k points are first chosen and Euclidean distance is calculated with respect to the remaining points. New assignment is made again based on the maximum value amongst the central node values.

Imbalanced classification is a supervised machine learning problem where one class exceeds other classes by a large difference.

The balancing techniques that were used are:

- Random oversampling
- Cost sensitive models
- One class classification
- Auto associative neural networks

8 machine learning methods are presented in the literature for fraud detection and classification. Most efficient methods according to three performance measures were chosen. Then, comparison was made on those algorithms with the same algorithms but with balancing techniques being applied on them. Comparative study was done with the added value of these methods by comparing the imbalanced classification approaches to the classic methodologies. This comparison helped understand the limitations of these approaches. Results are reported in detail and the limitations of the solutions are discussed and experimented in this paper.

CHAPTER 2
MERITS AND DEMERITS OF THE BASE PAPER

2.1 LITERATURE SURVEY

Credit Fraud Detection in this paper [4], had synthetic minority oversampling technique and adaptive synthetic techniques were proposed to balance the imbalanced dataset and train the same using machine learning algorithms.

In this paper [1], an experimental study was done in detecting credit card frauds with imbalanced dataset. Results have shown that balancing the imbalanced dataset is not needed to measure the performance instead performance can be measured using Accuracy, Sensitivity and Area under Precision/recall curve.

Predicting Credit Card Fraud on an Imbalanced Data: In this Paper, the dataset was balanced with oversampling technique and trained using Random Forest, KNN, Decision Tree, and Logistic Regression.

2.2 MERITS

1)Efficient pre-processing
2) Pattern recognition
3) Accurate predictions
4) The extensions of algorithms in the paper helps in finding the best algorithm in detecting frauds.
5) KNN algorithm in the paper can be used to determine anomalies inthe target instance and is easy to implement
6) Neural networks have learned the previous behavior so, can detect the frauds

2.3 DEMERITS

1) Neural networks take longer training period.

2) SVM takes long time to train the model.

3) Size of hidden layers in ANN was difficult.

4) SVM is hard for auditors to process outcomes due to transformation of input set andsometimes it fails to detect fraud cases, expensive.

CHAPTER 3
SOURCE CODE

SVM.R , C5.0 .R, ANN.R , C5.0CS.R , C5.0RO.R

```
dataset = read.csv('data.csv')

#EDA
View(dataset)
is.null(dataset$custID)
is.null(dataset$gender)
is.null(dataset$state)
is.null(dataset$cardholder)
is.null(dataset$balance)
is.null(dataset$numTrans)
is.null(dataset$numIntlTrans)
is.null(dataset$creditLine)
is.null(dataset$fraudRisk)
head(dataset)
str(dataset)
summary(dataset)
#imbalance
table(dataset$gender)
table(dataset$fraudRisk)
prop.table(table(dataset$fraudRisk))


#Dataset reduction
df=dataset[1:(0.02*(9403986+596014)),]

#imbalance in new dataset
```

```
str(df)
table(df$fraudRisk)
prop.table(table(df$fraudRisk))
barplot(prop.table(table(df$fraudRisk)),col=rainbow(2),ylim=c(0,1),main = "Imbalance in
fraudRisk")
#frauds in actual dataset (10M)
(596014/(9403986+596014))*100        #5.96%
#frauds in 2% of dataset (first 200,000 records)
(12145 /(12145+187855))*100          #6.07%


#Visualization
df$fraudRisk<-as.factor(df$fraudRisk)
ggplot(data = df, aes(x = creditLine, y = state)) + geom_point(aes(col=fraudRisk))


#SPLITTING DATASET(df)
set.seed(123)
split = sample.split(df$fraudRisk, SplitRatio = 0.7)
training_set = subset(df, split == TRUE)
test_set = subset(df, split == FALSE)
training_set
test_set

#FEATURE SCALING
training_set[, 1:8] = scale(training_set[, 1:8])
test_set[, 1:8] = scale(test_set[, 1:8])
training_set
test_set
```

#MACHINE LEARNING

 # SVM

```
vars <- c("gender","state","cardholder","balance", "numTrans", "numIntlTrans","creditLine")
classifier = svm(formula = fraudRisk~ .,
                 data =  training_set,
                 type = 'C-classification',
                 kernel = 'linear')
classifier
y_pred = predict(classifier,test_set[, vars])
y_pred
summary(y_pred)
plot(y_pred)
cm = table(y_pred,test_set$fraudRisk)
confusionMatrix(cm)
sensitivity(cm)
pr<-pr.curve(scores.class0 = y_pred, scores.class1 = test_set$fraudRisk,curve = TRUE)
plot(pr)
```

 # C5.0

```
vars <- c("gender","state","cardholder","balance", "numTrans", "numIntlTrans","creditLine")
training_set$fraudRisk<-as.factor(training_set$fraudRisk)
str(training_set$fraudRisk)
tree_mod=C5.0(x = training_set[, vars], y = training_set$fraudRisk)
tree_mod
summary(tree_mod)
plot(tree_mod)
y_predd=predict(tree_mod,test_set[, vars])
y_predd
```

```
summary(y_predd)
plot(y_predd)
auc_score=auc(test_set$fraudRisk,y_predd)
auc_score
pr<-pr.curve(scores.class0 = y_predd, scores.class1 = test_set$fraudRisk,curve = TRUE)
plot(pr)
x=table(y_predd,test_set$fraudRisk)
confusionMatrix(x)
sensitivity(x)
```

 # ANN

```
set.seed(333)
#training_set$fraudRisk=ifelse(training_set$fraudRisk == 1, TRUE, FALSE)
var=c("gender","state","cardholder","balance", "numTrans",
"numIntlTrans","creditLine","fraudRisk")
t=training_set[1:140000,var]
n <- neuralnet(fraudRisk ~ gender + state + cardholder + balance + numTrans +
numIntlTrans + creditLine,
        data = t,hidden = 3,linear.output = FALSE, err.fct = 'ce',likelihood = TRUE,
        act.fct = "logistic",algorithm = "rprop+")
plot(n)
y_pann=predict(n,test_set[, vars])
y_pann
summary(y_pann)
plot(y_pann)
xann=table(y_pann,test_set$fraudRisk)
xann=confusionMatrix(xann)
sensitivity(xann)
```

```
h2o.init(nthreads = -1)
classifier1=h2o.deeplearning(y='fraudRisk',training_frame = as.h2o(training_set),activation =
'Rectifier',
                    hidden = c(7,3,1),epochs = 100,train_samples_per_iteration = -2)
vars <- c("gender","state","cardholder","balance", "numTrans", "numIntlTrans","creditLine")
prob_pred=h2o.predict(classifier1,newdata = as.h2o(test_set[,vars]))
y_preddd=(prob_pred > 0.5)
y_preddd=as.vector(y_preddd)
y_preddd
summary(y_preddd)
plot(y_preddd)
x2=table(y_preddd,test_set$fraudRisk)
confusionMatrix(x2)
sensitivity(x2)
pr<-pr.curve(scores.class0 = y_preddd, scores.class1 = test_set$fraudRisk,curve = TRUE)
plot(pr)
```

 # C5.0 - CS

```
cost_mat <- matrix(c(0, 1, 3, 0), nrow = 2)
rownames(cost_mat) <- colnames(cost_mat) <- c("0", "1")
cost_mat
training_set$fraudRisk<-as.factor(training_set$fraudRisk)
str(training_set)
cost_mod <- C5.0(x = training_set[, vars], y = training_set$fraudRisk,costs = cost_mat)
summary(cost_mod)
y_predd1=predict(cost_mod,test_set[, vars])
y_predd1
pr1<-pr.curve(scores.class0 = y_predd1, scores.class1 = test_set$fraudRisk,curve = TRUE)
plot(pr1)
summary(y_predd1)
plot(y_predd1)
```

```
x1=table(y_predd1,test_set$fraudRisk)
confusionMatrix(x1)
sensitivity(x1)
```

 # C5.0 - RO

```
df_balanced <- ovun.sample(fraudRisk ~ ., data = training_set, method =
"over",p=0.15,seed=1)$data
table(df_balanced$fraudRisk)
table(training_set$fraudRisk)

vars <- c("gender","state","cardholder","balance", "numTrans", "numIntlTrans","creditLine")
df_balanced$fraudRisk<-as.factor(df_balanced$fraudRisk)
str(df_balanced$fraudRisk)
tree_modd=C5.0(x = df_balanced[, vars], y = df_balanced$fraudRisk)
tree_modd
summary(tree_modd)
plot(tree_modd)
y_predd2=predict(tree_modd,test_set[, vars])
y_predd2
auc_score=auc(test_set$fraudRisk,y_predd2)
auc_score
pr<-pr.curve(scores.class0 = y_predd2, scores.class1 = test_set$fraudRisk,curve = TRUE)
plot(pr)
summary(y_predd2)
plot(y_predd2)
x3=table(y_predd2,test_set$fraudRisk)
confusionMatrix(x3)
sensitivity(x3)
```

KNN.R

```
# Importing the dataset
dataset = read.csv("C:/Users/Dell/Downloads/main.csv")
dataset = dataset[2:9]



# Splitting the dataset into the Training set and Test set
 #install.packages('caTools')
library(caTools)
set.seed(123)
split = sample.split(dataset$fraudRisk, SplitRatio = 0.70)
training_set = subset(dataset, split == TRUE)
test_set = subset(dataset, split == FALSE)

# Feature Scaling
training_set[-8] = scale(training_set[-8])
test_set[-8] = scale(test_set[-8])

# Fitting K-NN to the Training set and Predicting the Test set results
library(class)
y_pred = knn(train = training_set[, -8],
             test = test_set[, -8],
             cl = training_set[, 8],
             k = 5,
             prob = TRUE)

# Making the Confusion Matrix
cm = table(test_set[, 8], y_pred)
```

```
#install.packages('caret')
library(caret)
#library(klaR)
xt=table(test_set$fraudRisk,y_pred)
confusionMatrix(xt)
n = sum(cm) # number of instances
nc = nrow(cm) # number of classes
diag = diag(cm) # number of correctly classified instances per class
rowsums = apply(cm, 1, sum) # number of instances per class
colsums = apply(cm, 2, sum) # number of predictions per class
recall = diag / rowsums
print(recall)

library(PRROC)
pr=pr.curve(scores.class0 = y_pred,weights.class0 = test_set$fraudRisk,curve=TRUE)
plot(pr)
```

LR.R

```
# Logistic Regression
# Importing the dataset
dataset = read.csv('main.csv')
dataset = dataset[2:9]

# Splitting the dataset into the Training set and Test set
#install.packages('caTools')
library(caTools)
set.seed(123)
```

```
split = sample.split(dataset$fraudRisk, SplitRatio = 0.75)
training_set = subset(dataset, split == TRUE)
test_set = subset(dataset, split == FALSE)
classifier = glm(formula = fraudRisk ~ .,family = binomial,data = training_set)

# Predicting the Test set results
prob_pred = predict(classifier, type = 'response', newdata = test_set[-8])
y_pred = ifelse(prob_pred > 0.5, 1, 0)

# Making the Confusion Matrix
cm = table(test_set[, 8], y_pred > 0.5)
#install.packages('caret')
library(caret)
xt=table(test_set$fraudRisk,y_pred)
confusionMatrix(xt)
sensitivity(xt)

n = sum(xt)# number of instances
nc = nrow(xt) # number of classes
diag = diag(xt) # number of correctly classified instances per class
rowsums = apply(xt, 1, sum)# number of instances per class
colsums = apply(xt, 2, sum) # number of predictions per class
Sensitivity = diag / rowsums
print(Sensitivity)

#install.packages("PRROC")

library(PRROC)
PRROC_obj <- pr.curve(scores.class0 = prob_pred, weights.class0=test_set$fraudRisk,
  curve=TRUE)
plot(PRROC_obj,color=2)
```

PRROC_obj

NB.R

```
# Logistic Regression
# Importing the dataset
dataset = read.csv('main.csv')
dataset = dataset[2:9]

# Splitting the dataset into the Training set and Test set
#install.packages('caTools')
library(caTools)
set.seed(123)
split = sample.split(dataset$fraudRisk, SplitRatio = 0.70)
training_set = subset(dataset, split == TRUE)
test_set = subset(dataset, split == FALSE)

#install.packages('e1071')
library(e1071)
classifier = naiveBayes(x = training_set[-8], y = training_set$fraudRisk)
#classifier=naiveBayes(fraudRisk ~ ., data=training_set)

# Predicting the Test set results
y_pred = predict(classifier, newdata = test_set[-8])
# Making the Confusion Matrix
#install.packages('caret')
library(caret)
xt=table(test_set$fraudRisk,y_pred)
confusionMatrix(xt)
```

```
sensitivity(xt)
n = sum(xt)# number of instances
nc = nrow(xt) # number of classes
diag = diag(xt) # number of correctly classified instances per class
rowsums = apply(xt, 1, sum)# number of instances per class
colsums = apply(xt, 2, sum) # number of predictions per class
Sensitivity = diag / rowsums
print(Sensitivity)

#install.packages("PRROC")

library(PRROC)

PRROC_obj <- roc.curve(scores.class0 = prob_pred, weights.class0=dataset$fraudRisk,
          curve=TRUE)
plot(PRROC_obj)
```

SVM - OCC

```
library(caTools)
library(caret)
library(caTools)
library(e1071)

dataset = read.csv("C:/Users/Dell/Downloads/main.csv")
dataset = dataset[2:9]


# Splitting the dataset into the Training set and Test set
```

```
#install.packages('caTools')

set.seed(123)
split = sample.split(dataset$fraudRisk, SplitRatio = 0.75)
training_set = subset(dataset, split == TRUE)
test_set = subset(dataset, split == FALSE)

train=subset(training_set,fraudRisk==1)

x=subset(train,select=-fraudRisk)
y=as.factor(train$fraudRisk)
cs=svm(x,y,type='one-classification')
pred2=predict(cs,subset(test_set,select=-fraudRisk))
pred3=format(round(pred2))
cm1=table(pred3,test_set$fraudRisk)

n = sum(cm1) # number of instances
nc = nrow(cm1) # number of classes
diag = diag(cm1) # number of correctly classified instances per class
rowsums = apply(cm1, 1, sum) # number of instances per class
colsums = apply(cm1, 2, sum) # number of predictions per class
sensitivity = diag / rowsums
print(sensitivity)

library(PRROC)
pr=pr.curve(scores.class0 = pred2,weights.class0 = test_set$fraudRisk,curve=TRUE)
plot(pr)
```

SVM - CS

```
library(caTools)
library(caret)
library(caTools)
library(e1071)

dataset = read.csv("C:/Users/Dell/Downloads/main.csv")
dataset = dataset[2:9]

# Encoding the target feature as factor
#dataset$Purchased = factor(dataset$Purchased, levels = c(0, 1))

# Splitting the dataset into the Training set and Test set
#install.packages('caTools')

set.seed(123)
split = sample.split(dataset$fraudRisk, SplitRatio = 0.75)
training_set = subset(dataset, split == TRUE)
test_set = subset(dataset, split == FALSE)
cost_mat<-matrix(c(0,1,3,0),nrow=2)
rownames(cost_mat)
colnames(cost_mat)<-c("0","1")
cost_mat
training_set$fraudRisk<-as.factor(training_set$fraudRisk)
str(training_set)
x=subset(training_set,select=-fraudRisk)
cost_mod<-svm(x,y=training_set$fraudRisk,costs=cost_mat)
pred2=predict(cost_mod,subset(test_set,select=-fraudRisk))
pred3=format(round(pred2))
cm=table(pred2,test_set$fraudRisk)
```

```
n = sum(cm) # number of instances
nc = nrow(cm) # number of classes
diag = diag(cm) # number of correctly classified instances per class
rowsums = apply(cm, 1, sum) # number of instances per class
colsums = apply(cm, 2, sum) # number of predictions per class
sensitivity = diag / rowsums
print(sensitivity)
```

AANN.R

```
# Auto Associative Nueral Network
# Importing the dataset
dataset = read.csv('main.csv')
dataset = dataset[2:9]

# Splitting the dataset into the Training set and Test set
#install.packages('caTools')
library(caTools)
set.seed(123)
split = sample.split(dataset$fraudRisk, SplitRatio = 0.75)
training_set = subset(dataset, split == TRUE)

test_set = subset(dataset, split == FALSE)

#one class classification
df<- subset(training_set ,  fraudRisk==1)#choose only one of the classes
df=df[1:8]

#test_set<-subset(test_set,fraudRisk==1)
```

```
#test_set=test_set[1:8]
#install.packages("neuralnet")
library(neuralnet)
class=neuralnet(fraudRisk ~ gender+state+cardholder+
                balance+numTrans+numIntlTrans+creditLine,data=df,hidden=6,
                linear.output=FALSE,likelihood=TRUE,act.fct="logistic",
                algorithm="rprop+")


# Predicting the Test set results
prob_pre = predict(class, newdata = test_set[-8])
plot(prob_pre)
y_pre = ifelse(prob_pre > 0.9998, 1, 0)
# Making the Confusion Matrix
library(caret)

kl=table(test_set$fraudRisk,y_pre)
kl
confusionMatrix(kl)
sensitivity(kl)
n = sum(kl)# number of instances
nc = nrow(kl) # number of classes
diag = diag(kl) # number of correctly classified instances per class
rowsums = apply(kl, 1, sum)# number of instances per class
colsums = apply(kl, 2, sum) # number of predictions per class
Sensitivity = diag / rowsums
print(Sensitivity)

#install.packages("PRROC")
library(PRROC)
PRROC_obj <- roc.curve(scores.class0 = prob_pre, weights.class0=dataset$fraudRisk,
```

```
                curve=TRUE)
plot(PRROC_obj)

prcurve <- pr.curve(scores.class0 = y_pre, weights.class0=test_set$fraudRisk,
                curve=TRUE)
plot(prcurve)
```

CHAPTER 4
SNAPSHOTS

Table 4.1 Performance Metrics

Algorithm	Accuracy	Sensitivity	AUPRC Value
SVM	95.85	42.68	0.9491
C5.0	95.66	45.59	0.9499
ANN	95.88	4741	0.4823
KNN	95.8	45.97	0.3308
LR	95.89	47.59	0.6847
NB	93.48	55.77	0.3090
C5.0 - RO	92.23	65.61	0.9567
C5.0 - CS	93.49	69.61	0.9559
SVM - OCC	81.58	16.02	0.3857
SVM - CS	95.8	80.41	Could not be plotted
AANN	66.93	47.95	0.122

AUPRC Curves of all algorithms

Fig 4.1 shows the AUPRC curve for support vector machine.

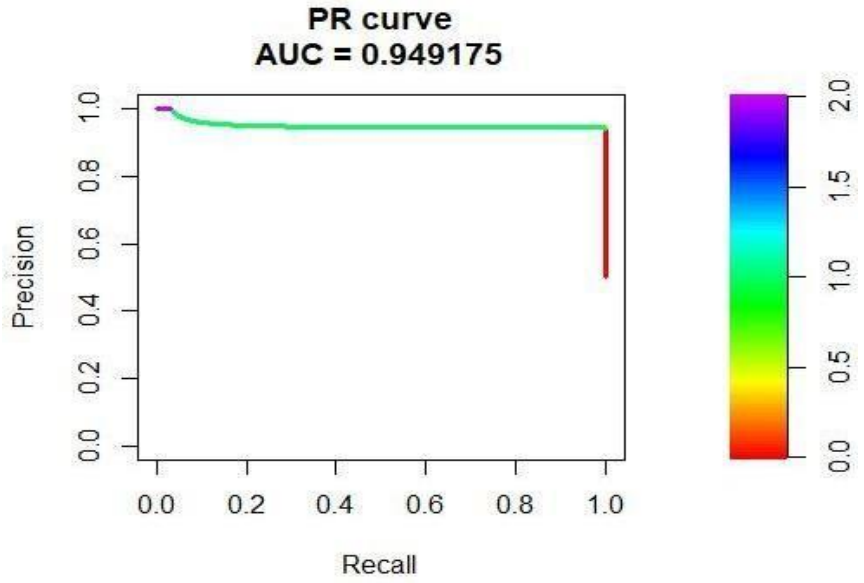

Fig. 4.1. SVM - AUPRC Curve

Fig 4.2 shows the AUPRC curve for C5.0.

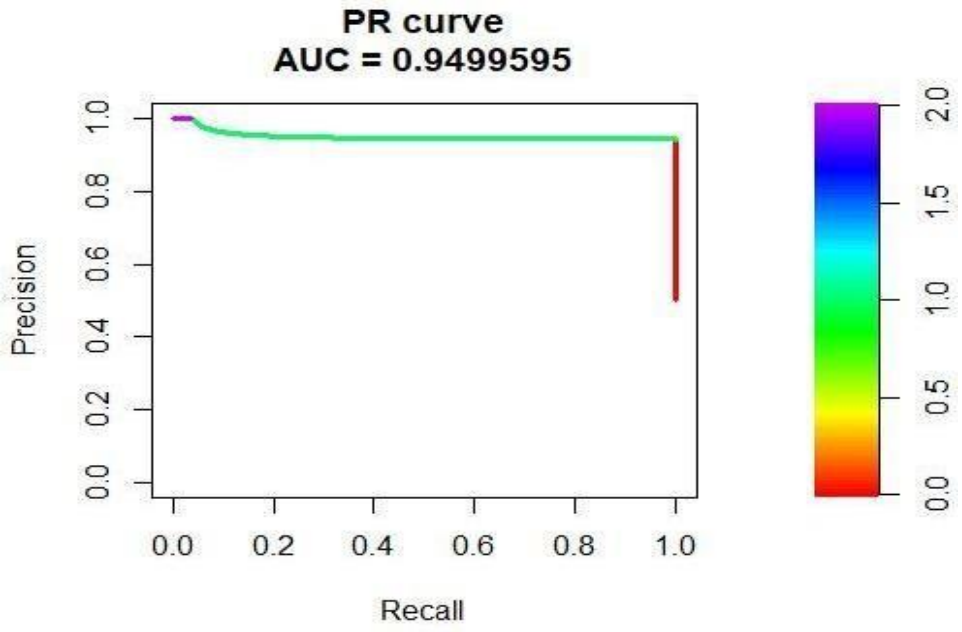

Fig. 4.2. C5.0-AUPRC Curve

Fig 4.3 shows the AUPRC curve for artificial neural networks.

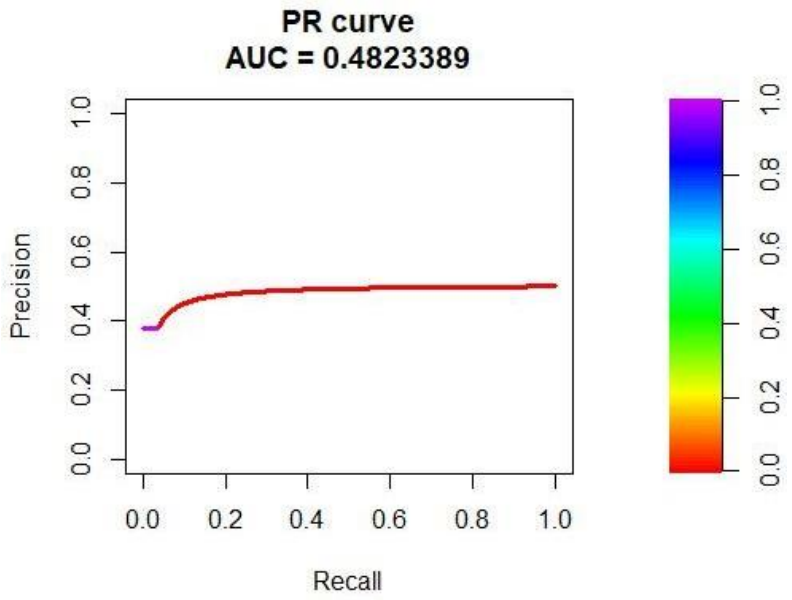

Fig. 4.3. ANN - AUPRC Curve

Fig 4.4 shows the AUPRC curve for K-nearest neighbors.

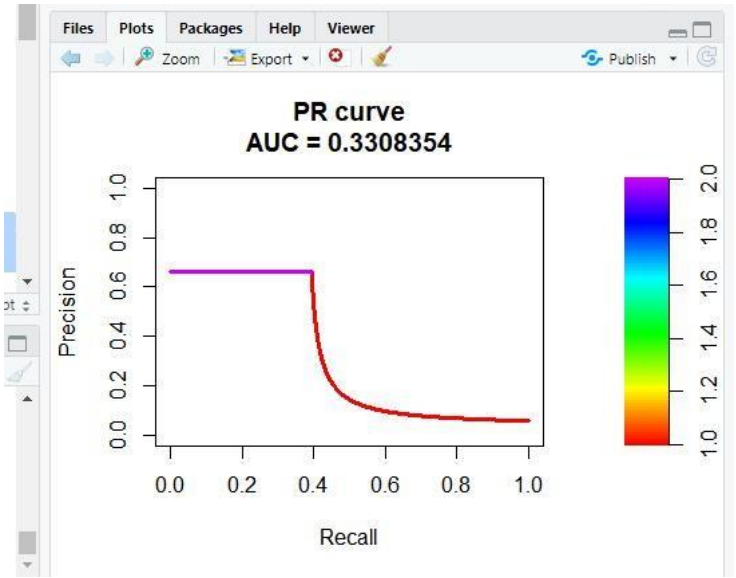

Fig. 4.4. KNN-AUPRC Curve

Fig 4.5 shows the AUPRC curve for logistic regression.

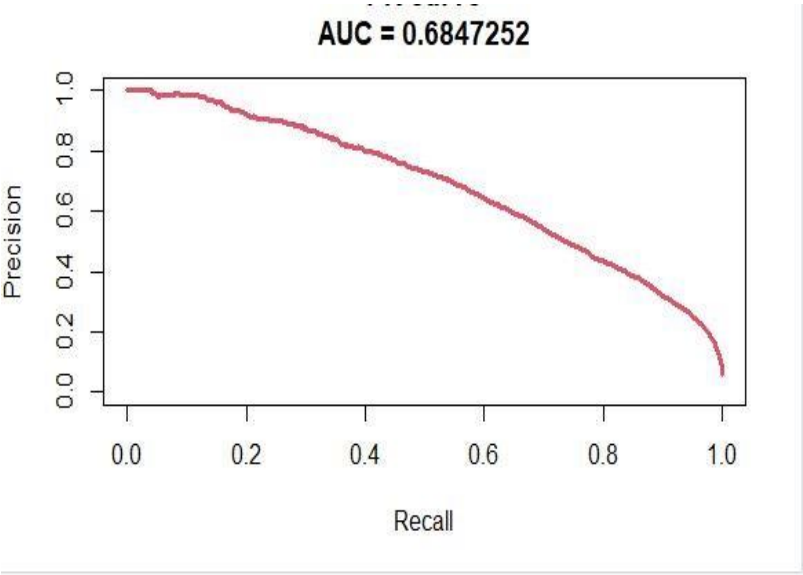

Fig. 4.5. LR-AUPRC Curve

Fig 4.6 shows the AUPRC curve for naïve bayes.

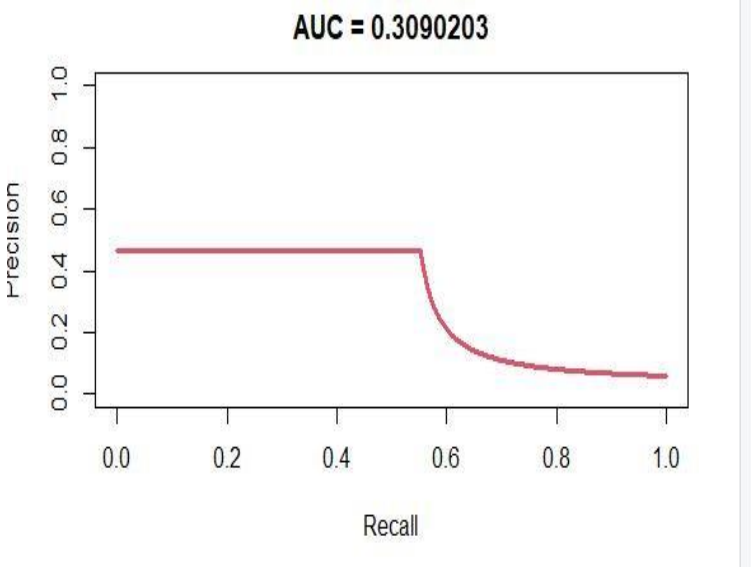

Fig. 4.6. NB-AUPRC Curve

Fig 4.7 shows the AUPRC curve for C5.0 random oversampling.

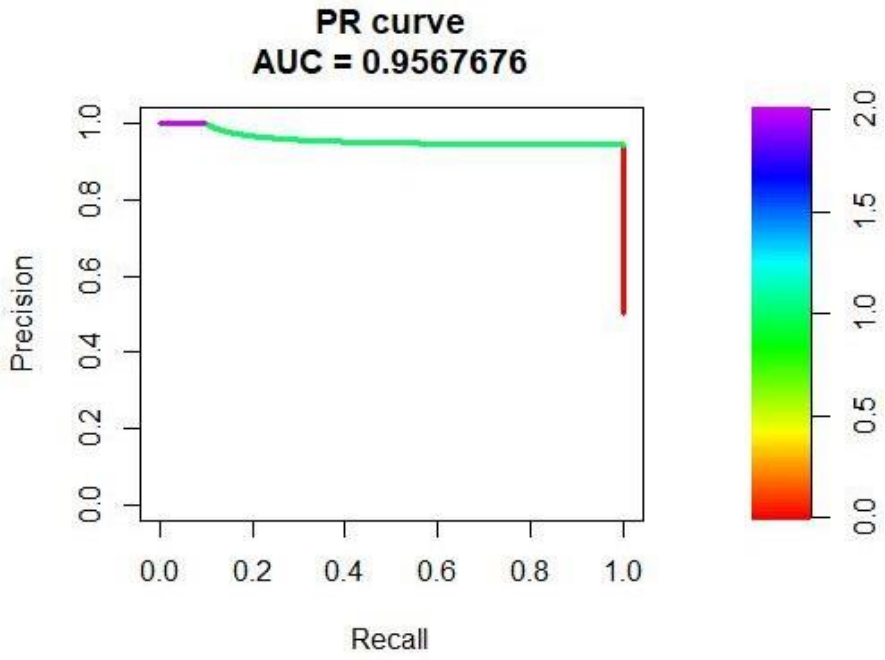

Fig. 4.7. C5.0 RO- AUPRC Curve

Fig 4.8 shows the AUPRC curve for C5.0 Cost sensitive models.

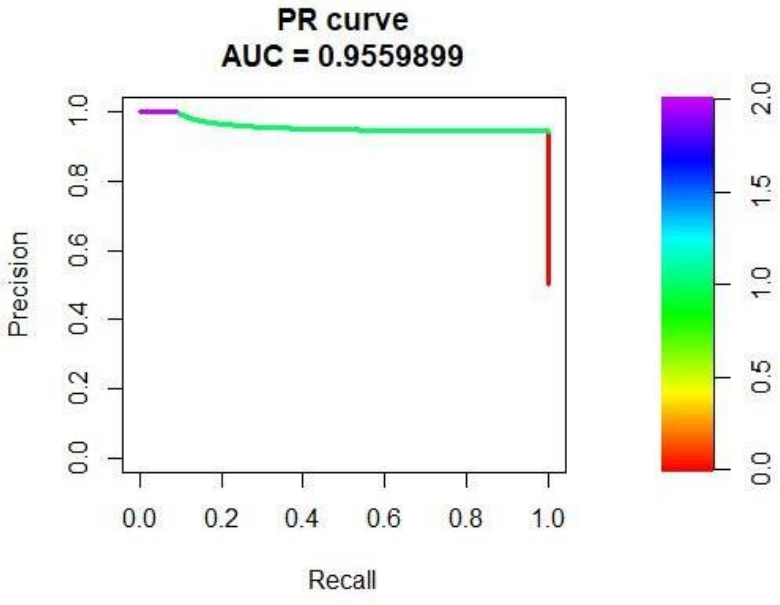

Fig. 4.8. C5.0(CS)- AUPRC Curve

Fig 4.9 shows the AUPRC curve for support vector machine one class classification.

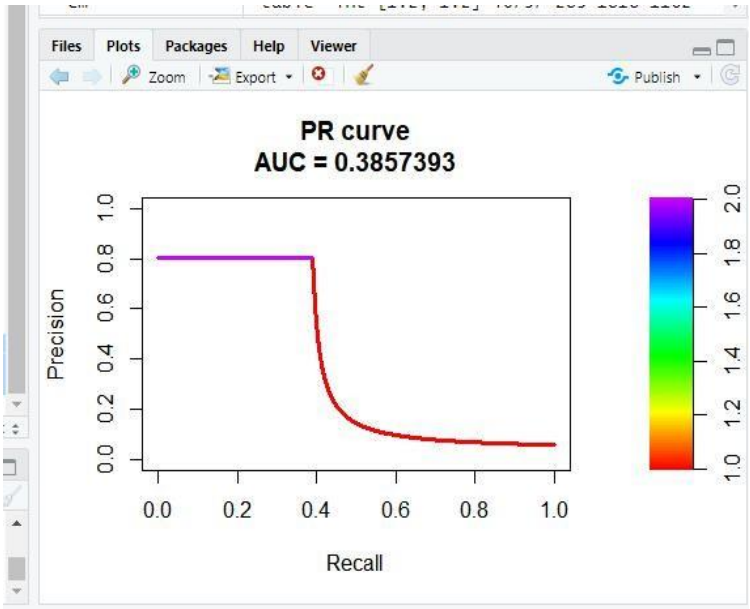

Fig. 4.9. SVM(OCC) - AUPRC Curve

Fig 4.10 shows the AUPRC curve for auto associative neural network

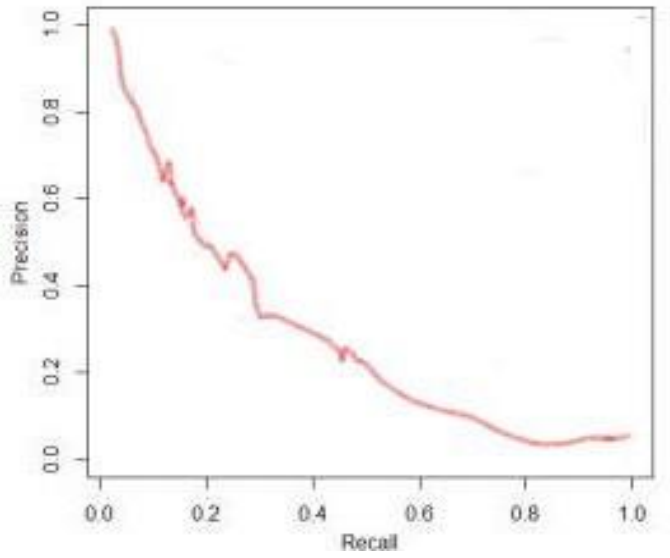

Fig. 4.10. AANN- AUPRC Curve

CHAPTER 5
CONCLUSION AND FUTURE PLANS

5.1 Conclusion

Fraud is a significant risk for both businesses and individuals. Every year, many forms of fraudulent activities cost billions of dollars. Among them, Credit Card fraud is the most common and expensive. Detecting Credit Card Fraud is quite difficult. Several issues make finding solutions difficult for fraud detection, of which the problem of class imbalance is one of the most serious problems now a days. Several solutions have been proposed in this paper. In this paper, imbalanced dataset is trained using six machine learning algorithms. We analyzed the results of six machine learning approaches for detecting credit card fraud and identified their flaws. More specifically, we examined the approaches to imbalanced classification and looked at how effective they are in extreme cases of imbalance. As per the value of Accuracy, Sensitivity and AUPRC value C5.0 decision tree algorithm, SVM, ANN gave the best results among the remaining algorithms. This paper showed that how extreme imbalance generates large number of false alarms which gives low sensitivity and it also showed that one performance measure is not sufficient in imbalanced classification for choosing the best algorithms. Techniques like Random Oversampling, one class classification and cost sensitive models are used to balance the dataset and the results obtained after balancing have been compared with the best algorithms obtained before balancing the dataset.

5.2 Future Plans

The actual dataset is a massive 10 million dataset that cannot be trained on computers, hence in this study we just used 2% of the dataset that is 2 lakh datasets. In order to get ideal results, we'll need to use 10 million datasets, which our laptops won't be able to handle. So, as a further study We will create a Big Data-driven ecosystem. Our present study in this paper was obviously limited by the size of the dataset, as we were unable to run it on significant volumes of data. As a result, we intend to use Big Data technology to create a scalable environment.

CHAPTER 6
REFERENCES

Baabdullah, T., Alzahrani, A., & Rawat, D. B. (2020, May). On the Comparative Study of Prediction Accuracy for Credit Card Fraud Detection wWith Imbalanced Classifications. In *2020 Spring Simulation Conference (SpringSim)* (pp. 1-12). IEEE.

Shamsudin, H., Yusof, U. K., Jayalakshmi, A., & Khalid, M. N. A. (2020, October). Combining oversampling and undersampling techniques for imbalanced classification: A comparative study using credit card fraudulent transaction dataset. In *2020 IEEE 16th International Conference on Control & Automation (ICCA)* (pp. 803-808). IEEE.

Benchaji, I., Douzi, S., & El Ouahidi, B. (2018, October). Using genetic algorithm to improve classification of imbalanced datasets for credit card fraud detection. In *International Conference on Advanced Information Technology, Services and Systems* (pp. 220-229). Springer, Cham.

Tran, T. C., & Dang, T. K. (2021, January). Machine Learning for Prediction of Imbalanced Data: Credit Fraud Detection. In *2021 15th International Conference on Ubiquitous Information Management and Communication (IMCOM)* (pp. 1-7). IEEE.

Soh, W. W., & Yusuf, R. M. (2019). Predicting credit card fraud on a imbalanced data. *International Journal of Data Science and Advanced Analytics (ISSN 2563-4429)*, *1*(1), 12-17.

Rai, A. K., & Dwivedi, R. K. (2020, July). Fraud detection in credit card data using unsupervised machine learning based scheme. In *2020 International Conference on Electronics and Sustainable Communication Systems (ICESC)* (pp.421-426).IEEE.

CHAPTER 7
APPENDIX – BASE PAPER

Received June 3, 2019, accepted June 27, 2019, date of publication July 8, 2019, date of current version July 29, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2927266

An Experimental Study With Imbalanced Classification Approaches for Credit Card Fraud Detection

SARA MAKKI[1,2], ZAINAB ASSAGHIR[2], YEHIA TAHER[3], RAFIQUL HAQUE[4], MOHAND-SAÏD HACID[1], AND HASSAN ZEINEDDINE[2]

[1]LIRIS, Computer Science Department, Claude Bernard University Lyon 1, 69100 Villeurbanne, France
[2]Applied Mathematics Department, Lebanese University, Beirut 1003, Lebanon
[3]David, Computer Science Department, Versailles Saint-Quentin-en-Yvelines, 78000 Versailles, France
[4]Centre for Big Data Science Research and Development, Cognitus, 75008 Paris, France Corresponding author: Sara Makki (sara.makki@etu.univ-lyon1.fr)

This work supported in part by the Lebanese National Council for Scientific Research (CNRS-L).

ABSTRACT Credit card fraud is a criminal offense. It causes severe damage to financial institutions and individuals. Therefore, the detection and prevention of fraudulent activities are critically important to financial institutions. Fraud detection and prevention are costly, time-consuming, and labor-intensive tasks. A number of significant research works have been dedicated to developing innovative solutions to detect different types of fraud. However, these solutions have been proved ineffective. According to Cifa, 33305 cases of credit card identity fraud were reported between January and June in 2018.[1] Various weaknesses of existing solutions have been reported in the literature. Among them all, the imbalance classification is the most critical and well-known problem. Imbalance classification consists of having a small number of observations of the minority class compared with the majority in the data set. In this problem, the ratio *fraud: legitimate* is very small, which makes it extremely difficult for the classification algorithm to detect fraud cases. In this paper, we will conduct a rigorous experimental study with the solutions that tackle the imbalance classification problem. We explored these solutions along with the machine learning algorithms used for fraud detection. We identified their weaknesses and summarized the results that we obtained using a credit card fraud labeled dataset. According to this paper, imbalanced classification approaches are ineffective, especially when the data are highly imbalanced. This paper reveals that the existing approaches result in a large number of false alarms, which are costly to financial institutions. This may lead to inaccurate detection as well as increasing the occurrence of fraud cases.

INDEX TERMS Fraud analysis and detection, fraud cybercrimes, imbalanced classification, secure society.

I. INTRODUCTION

Fraud detection concerns a large number of financial institutions and banks as this crime costs them around $ 67 billion per year. There are different types of fraud: insurance fraud, credit card fraud, statement fraud, securities fraud *etc.* Of all of them, credit card fraud is the most common type. It is defined as an unauthorized use of a credit card account. It occurs when the cardholder and the card issuer are not aware that the card is being used by a third party.

The associate editor coordinating the review of this manuscript and approving it for publication was Mahmoud Barhamgi.

[1]https://www.thesun.co.uk/money/7040706/33000-credit-card-fraudcases-reported-six-months/

The fraudsters can obtain goods without paying, or gain illegal access to funds from an account. Credit card fraud is classified into different types based on the nature of fraudulent activities. They are briefly introduced in the following:

- *Simple theft (offline fraud)*: a stolen card is the most straightforward type of credit card fraud. It is also the fastest to be detected.
- *Application fraud*: when individuals obtain new credit cards using false personal information.
- *Bankruptcy fraud*: this consists in using a credit card while being insolvent, and purchasing goods knowing that they are not able to pay. This type can be prevented with credit scoring techniques.
- *Internal fraud*: when bank employees steal the card details to use it remotely.
- *Counterfeit fraud / behavioral fraud / cardholder-notpresent fraud*: when transactions are made remotely (mobile sales, online, *etc.*), the cardholder does not need to be present, only the details of a legitimate credit card are needed. The card's details can be obtained by skimming or shoulder surfing. The detection of this type of credit card fraud can take time, and needs sophisticated methods that catch the transactions patterns.

In this paper, we focus on counterfeit fraud as it is far more challenging to detect, and the damage of this fraud is irrevocable.

A. PROBLEM DESCRIPTION

Several challenges of credit card fraud have been reported in the literature (*e.g.*, [1]–[3]). However, the class imbalance problem is one of the most critical challenges of all. This problem is defined as having an extremely imbalanced and highly skewed distribution of the data [4]. In other words, the ratio of fraudulent or criminal activities is considerably smaller than the legitimate and genuine ones. Figure 1 illustrates a class imbalance problem found in one of our experiments. In our experiment, the percentage of fraudulent transactions is 5.96%. Figure 1 shows an imbalance situation in our dataset according to two features. The red dots in the figure represent the fraud cases that have a much lower frequency than the other cases.

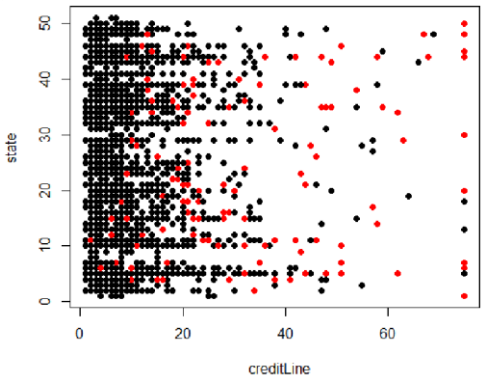

FIGURE 1. The class imbalance in our example.

Class imbalance promotes a huge challenge in detecting thecharacteristicsoffraudulentactivitiesandextractingfraud patterns. Due to the dominance of one class, most of the optimization steps (concerning accuracy) performed by the classification algorithm, aim to correctly classify the dominant class while ignoring the others. These minority observations like the fraud observations are the most critical to be classified correctly. If the classification algorithm is unable to detect fraud patterns, the illegal transactions are considered legal, thus causing severe financial damage to individuals and organizations.

Many research works have been dedicated to the imbalance classification problem. Several solutions have been proposed in a large body of literature (*e.g.*, [5]–[7]) which, to the best of our knowledge, are built on machine learning and data mining algorithms. However, class imbalance remained an unsolved issue [1], [8].

In this paper, we aim at identifying weaknesses of existing class imbalance approaches; more specifically, the machine learning methods. We conducted an experimental study with the state of the

art technologies currently used in applications to tackle the imbalance classification problem. Our goal is to detect the issues that must be solved to product a highly efficient solution for the class imbalance problem.

B. OUR CONTRIBUTION

In this paper, we report a rigorous experimentation and compare the performance of solutions that deal with the imbalance classification problem. Also, we identify their weaknesses to help researchers target the right issues for tackling the problem in the real world. Given below is the list of our specific contributions in this paper:

- We compared 8 machine learning methods presented in the literature that are used for fraud detection and classification.
- We selected the most efficient methods according to three performance measures. Then, we compared them with class imbalance approaches applied to these same algorithms.
- We studied the added value of these methods by comparing the imbalanced classification approaches to the classic methods. This comparison helps understand the limitations of these approaches.
- We reported the results in detail and discussed the limitations of the solutions that we experimented in this paper.

The remainder of the paper is organized as follows. In Section II, we reviewed literature related to credit card fraud and imbalanced classification. Section III provides a brief description of the eight machine learning methods used for credit card fraud detection and the imbalanced classification approaches. In Section IV, we provided a detailed description of our experimentation study, which include experimental design, results and discussion of the limitations of solutions that we found. Finally, we provided a conclusion and briefly explained future work.

II. RELATED WORK

In this section, we present works that revolve around credit card fraud detection. Also, we strongly focus on the research that addresses the class imbalance problem in fraud detection.

A. CREDIT CARD FRAUD DETECTION

Credit card datasets contain information about transactions like account number, type of card, kind of purchase, location and time of transaction, client's name, merchant code, size of transaction, *etc.* This information was used by several researchers as variables to determine whether the transaction is fraudulent or legitimate; or to detect outliers that merit investigation. Bolton and Hand [9] proposed two clustering techniques: peer group analysis and break-point analysis to detect behavioral fraud. Weston *et al.* [10] used peer group analysis on real credit card transaction data to find outliers and suspicious transactions. Duman and Ozcelik [11] used genetic algorithms combined with scatter search to minimize the number of wrongfully classified transactions. Ramakalyani and Umadevi [12] also applied genetic programming to detect fraudulent card transactions. Bentley *et al.* [13] presented a fuzzy darwinian detection model based on genetic programming to produce fuzzy logic rules.

Srivastava *et al.* [14] modeled the sequence of credit card transactions using a Hidden Markov Model (HMM) initially trained with the cardholder's normal behavior, and showed how the model can be used for the detection of frauds. Other studies were conducted by Esakkiraj and Chidambaram [15] and Mishra *et al.* [16] with the same purpose using HMM. Artificial immune systems were investigated by Brabazon *et al.* [17] and Wong *et al.* [18] for the detection of fraudulent transactions, imitating the immune system's ability to distinguish between *self* and *non-self*. Sánchez *et al.* [19] used association rules to extract knowledge about fraudulent and unlawful card transactions. Sahin *et al.* [20] proposed a cost-sensitive decision tree approach for fraud detection. Bahnsen *et al.* [21] proposed a cost sensitive approach based on Bayes minimum risk for credit card fraud detection. Pasarica [22] suggested the use of support vector machine classification with the Gaussian kernel function and found it to be the best approach for detecting the fraud patterns. In their study, Sahin and Duman [23] compared decision trees (using three algorithms CART, C5.0 and CHAID) and support vector machines (with linear, sigmoid, polynomial, and radial kernel functions) and showed that decision trees, especially CART algorithm outperforms support vector machine methods. Ganji and Mannem [24] proposed a data stream outlier detection algorithm based on reverse K-nearest neighbors for detecting fraud.

Several studies focused on neural network applications to credit card fraud detection, [25]–[27]. Other studies combined a neural network with other algorithms, Ogwueleka [28] used an artificial neural netwrok with a rule based component, while Patidar and Sharma [29] applied an artificial neural network

tuned by genetic algorithms. Syeda *et al.* [30] introduced a fuzzy neural network on parallel machines with the purpose of speeding up rule production for customer-specific credit card fraud detection. Maes *et al.* [31] compared artificial neural networks and Bayesian belief network on real world financial data and showed that Bayesian belief networks detects 8% more fraudulent transactions than artificial neural networks. Whitrow *et al.* [32] introduced transaction aggregation and showed that it is effective. Also, they proved that random forest performs better than other methods such as support vector machine, logistic regression and K-nearest neighbors.

Bhattacharyya *et al.* [2] conducted a comparative study between logistic regression, support vector machine and random forest, they showed that random forest outperformed the two other methods. Subashini and Chitra [33] compared five models: decision trees (CART and C5.0), support vector machines with polynomial kernels, logistic regression and bayesian belief networks, and concluded that the CART algorithm performs better than other methods. In a recent study, Mahmoudi and Duman [34] proposed a modified Fisher discriminant function, using simple linear discriminant analysis for credit card detection for the first time, and modified it to be more sensitive to false negatives.

Hormozi *et al.* [35] presented a credit card fraud detection system by executing the negative selection algorithm (one of the artificial immune system algorithms) using the Hadoop and MapReduce paradigm. Quah and Sriganesh [36] developed a real-time credit card fraud detection model using self-organizing maps to distinguish fraud activities from normal behavior patterns. A hybridization of BLAST and SSAHA algorithms was used by Kundu *et al.* [37] as a profile analyzer and a deviation analyzer for credit card fraud. Sherly and Nedunchezhian [38] developed an adaptive credit card fraud detection system, using the BootstrappedOptimisticAlgorithmforTreeconstruction(BOAT). Minegishi and Niimi [39] introduced an online type of decision tree called Very Fast Decision Tree (VFDT) for the detection of fraudulent credit card use. Carcillo *et al.* [40] presented active learning strategies to label credit card fraud transactions. The authors investigated these strategies and studied their performance and detection accuracies.

B. CLASS IMBALANCE FRAUD CLASSIFICATION

In classification problems, class imbalance is defined as having the majority of observations from one class, which makes it challenging for the classifier to detect the minority group. Many researchers studied class imbalance problem classification and presented solutions. According to Krawczyk [41], the skewed class distribution can be tackled in different ways. The straightforward way is on data-level, like over- or undersampling that is used to balance the classes before applying any classification algorithm. Another way is an algorithmlevel approach, like cost-sensitive models, that aims to give a higher cost to the minority class; or one-class classification methods, when the training is performed using only the minority class.

Kamaruddin and Ravi [42] proposed a one-class classification approach to solve the imbalance problem. They proposed a hybrid system of Particle Swarm Optimization and AutoAssociative Neural Network (PSOAANN), and implemented it in a Spark computational framework. Wei *et al.* [43] presented an online fraud detection system and proved its efficiency in large volumes of extremely imbalanced data. Their approach includes three algorithms, contrast pattern mining, neural network and decision forest, where their results are all combined at the end. Padmaja *et al.* [4] introduced a fraud detection method that combines backpropagation, naive Bayes and C4.5 tree algorithms. They applied these algorithms to data that are derived from oversamplingwithreplacement.Theauthorsprovedtheeff iciency of their approach using a healthcare insurance fraud data set.

Padmaja *et al.* [44] focused on the class imbalance issue in fraud detection, by eliminating extreme outliers from the minority class using K reverse nearest neighbor. Then, a combination of oversampling minority class (SMOTE) and undersampling majority class was carried out. The authors conducted experiments based on multiple classifiers like C4.5, naive Bayes, k-nearest neighbor and Radial Basis Function networks. Chan *et al.* [45] introduced a method for tackling several issues in fraud detection including skewed data distribution. They proposed combining multiple learned fraud detectors under a cost model.

III. CLASSIFICATION ALGORITHMS FOR CREDIT CARD FRAUD DETECTION

Various solutions have been used to solve the class imbalance problem. These solutions are briefly discussed in this section. However, before discussing class imbalance solutions,wesummarizetheclassificationalgorithmsthat aretypically used for credit card fraud detection. It is

worth noting that a classification algorithm is an important component and is commonly used in class imbalance solutions. To be more explicit, a class imbalance solution is implemented through a classification algorithm that handles the minority group.

A. MACHINE LEARNING CLASSIFICATION ALGORITHMS

In this section, we briefly describe the machine learning algorithms used for credit card fraud detection that we subsequently evaluated and compared.

1) C5.0 ALGORITHM

C5.0 is one of the most common decision tree algorithms. It is an advanced version of the C4.5 algorithm [46]. Both C4.5 and C5.0 use cross entropy (information statistics and information gain) instead of the Gini index when evaluating the splits. A split is a ''*variable* < (or >) *value*'' rule that is used to divide a certain node into two daughter nodes. These splits are evaluated to find the one that best discriminates the target variable. While using C5.0, this evaluation was carried out by calculating the information gain after each split. The greater the information gain, the better the result. The information gain is calculated as follows:

$$gain(\text{split}) = info(\text{prior to split}) - info(\text{after the split})$$

where

$$info(\text{prior to split}) = -\left[\frac{N_1}{N} \times \log \frac{N_1}{N}\right] - \left[\frac{N_0}{N} \times \log \frac{N_0}{N}\right]$$

where N_1, N_0 and N are, respectively, the frequency of fraud, the frequency of legitimate transactions, and the total number of observations in the parent node. *info*(after the split) is the sum of the *info* for the two resulting nodes: *greater than split* and *less than split*, each multiplied by $\frac{n_i}{N}$, where n_i is the number of observations in the node.

2) SUPPORT VECTOR MACHINE (SVM)

SVM is a classification tool that aims at finding the hyperplane that separates data points in two classes optimally [47]. Formally, a given training vector x_i in R^n, $i = 1,\ldots,l$, n is the number of exploratory variables, and l is the number of observations in the train set. $y \in \mathsf{R}^l$ taking the values of 1 and −1. The binary classification is done by solving the following optimization problem.

$$\text{minimize} \quad -\sum_{i=1}^{l} \zeta_i \geq 0, \quad i = 1$$

$$y_i \times (w^T \delta(x_i) + b) \geq 1 - \zeta_i \text{ subject to } ,\ldots,l$$

The hyperplane equation is defined as: $w^T \delta(x_i) + b$, where w is a vector of weights, $\delta(x_i)$ maps x_i into a higherdimensional space. Slack variables ζ_i are added, to allow for some errors or miscalculations, in case these points are not linearly separable. C is a cost parameter > 0 associated with these errors. The aim of minimizing $\frac{1}{2}||w||^2$ is to maximize the distance between the two margins that is equal to $||w^2||_2$ in order to find the hyperplane that best separates the two classes.

a: ARTIFICIAL NEURAL NETWORK (ANN)

ANN is a connection of multiple neurons or nodes [48]. The multilayer feed-forward perceptron is made up of several layers: an input, one or more hidden layers and an output layer. The first layer contains the input nodes representing the exploratory variables. These inputs are multiplied with a specific weights and transferred to each of the hidden layer's nodes where they are added together with a certain bias. An activation function is then applied to this summation to produce the output of the neuron, which will be transferred to the next layer. Finally, the output layer provides the algorithm's response. The weights used are first set randomly, then using the training set these weights are adjusted to minimize the error, using specific algorithms like backpropagation.

3) NAÏVE BAYES (NB)

Naïve Bayes uses the Bayes conditional probability rule for classification [48]. This method consists in finding a class to the new observation that maximizes its probability given the values of the variables. Our objective is to find the value of Y that maximizes $P(Y/X_1,X_2,\ldots,X_n)$. Using the Bayes theorem:

$$P(Y/X_1,X_2,\ldots,X_n) = \frac{P(X_1,X_2,\ldots,X_n/Y)P(Y)}{P(X_1,X_2,\ldots,X_n)}$$

Maximizing $P(Y/X_1,X_2,\ldots,X_n)$ is equivalent to maximizing $P(X_1,X_2,\ldots,X_n/Y)$. This can be easily estimated from the historical data, assuming class-

conditional independence among variables:

$$P(X_1,X_2,...,X_n/Y) = P(X_1/Y)P(X_2/Y)...P(X_n/Y)$$

This assumption is not always satisfied. Another limitation of this method is the democratization of continuous variables. This means that some information may be lost, or that these variables are assumed to be approximately normally distributed which may not be true.

4) BAYESIAN BELIEF NETWORK (BBN)

Bayesian Belief Networks are graphical models for probabilistic relationships between a set of variables. They were developed to relax the *independence assumption* in Naïve Bayes and allow for dependencies among variables [48]. Variables are represented as nodes, while conditional dependencies between variables are represented as arcs between nodes. Each node is linked to a conditional probability table that generates probabilities of the node's variable conditionally to the values of the parent's node. The computational workflow of BBN is as follows:

- The first step is to find a structure for the network: this may be constructed by human experts or inferred from the data using specific algorithms.
- Once this topology is found, training the network is straightforward using the historical data as in Naïve Bayes. Note that the continuous variables are either discretized or assumed normally distributed. Also, in BBN, it is assumed that each node is independent of its non descendants given its parents in the graph. This is known as the *Markov condition*.

5) LOGISTIC REGRESSION (LR)

LogisticRegressionisatypeofgeneralizedlinearmodel[46]. Simple linear regression is not suitable when the variable to be predicted is binary. The vector $\alpha = (\alpha_0,\alpha_1,\alpha_2,...,\alpha_n)$ represents the coefficients, $X = (1,X_1,X_2,...,X_n)$ the exploratory variables, and the model's error. The linear model is defined as follows:

$$Y = \alpha_0 + \alpha_1 X_1 + \alpha_2 X_2 + \cdots + \alpha_n X_n + = X\alpha +$$

In logistics regression, a logit link function g over $[0,1]$ in $\mathbb{R}$ is introduced, to force the linear combination of the variables to take values between 0 and 1: $g(p) = X\alpha$, where p is the probability of fraud risk that we are estimating. The logit function is defined as:

$$g(p) = \ln \frac{p}{1-p} \quad \text{with} \quad = \frac{e^{X\alpha} p}{1 + e^{X\alpha}}$$

a: K-NEAREST NEIGHBOR (KNN)

KNN consists of the K nearest points to the one that we aim to predict [48]. A specific norm is used to measure the distance between points. The new observation is assigned the class with the majority of the K nearest points. The norm usually used to measure the distance between two observations p and q (an observation is $\in \mathbb{R}^n$) is the euclidean distance given by:

$$d(p,q) = \sqrt{\sum_{i=1}^{n}(p_i - q_i)^2}$$

6) ARTIFICIAL IMMUNE SYSTEMS (AIS)

Artificial Immune Systems are concerned with extracting the role of the immune system to create computational and predictive systems. One of the most common algorithms is the Negative Selection Algorithm (NSA). The negative selection is carried out in two phase: *detector set generation* and *classification* [35].

In AIS, random observations are generated and compared to the observations in the *self* set (i.e. legitimate cases). If these random observations match a self observation, then they are rejected. Otherwise, they are considered to be fraud detectorsandformthedetectorset.However,intheclassification phase, if a new observation matches at least one detector, it is classified as a fraud. Matching is measured using the euclidean distance and a specific threshold. The huge amount of time it takes to find the detector set is a disadvantage of this method, particularly when using a high threshold or big data.

B. IMBALANCED CLASSIFICATION

Two techniques are employed in imbalanced classification approaches. The first technique is employed on data as a prepossessing step to balance classes, like oversampling, undersampling, *etc.* The second technique is used within the classification algorithm like Cost-Sensitive (CS) approaches or One-Class Classification (OCC).

1) RANDOM OVERSAMPLING (RO)

Random Oversampling is used to balance classes by

simply replicating observations as needed until the balance between classes is reached. Our aim is to modify the behavior of the classification model to concentrate on both minority class (fraud) and majority class (legitimate) equally.

2) ONE-CLASS CLASSIFICATION (OCC)

This approach uses only one class of the data (usually the minority class) and learns its characteristics. In our case, the classification takes place in the testing phase. After training the algorithm with one class, it should be able to determine whether a certain transaction belongs to the minority class or not.

- *One-Class Classification SVM*: The aim of this approach is to find a ''small'' region capturing most of the training data points. This is achieved by estimating a function f taking the value 1 if a point is in this region and −1 elsewhere [49].
  - First, points are mapped using a mapping function δ from the variables space to a higher dimensional space F.
  - Then, these points are separated from the origin with a maximum margin using a hyperplane of equation: $w^T \delta(x_i) = \rho$.

  Our objective is to maximize the margin that is equal to $\frac{\rho}{||w||}$. The optimization problem is then:

$$\begin{aligned} \text{minimize} \quad & \frac{1}{2}||w||^2 + \frac{1}{vl}\sum_{i=1}^{l}\zeta_i - \rho \\ \text{subject to} \quad & w^T \delta(x_i) \geq \rho - \zeta_i \\ & \zeta_i \geq 0, i = 1,\ldots,l \end{aligned}$$

  where w is a weight vector and ρ is an offset parameter for the hyperplane in space F.
- *Auto Associative Neural Network (AANN)*: The Auto Associative Neural Network is typically used as an approach for ANN, using unlabeled or one-class data [42]. It is an ANN with a specific architecture, with the same number of nodes in the input and output layers. The AANN is trained using only the exploratory variables of the fraud class. Then, the average error of the training phase is calculated and used as a threshold for classification. If the average error of the estimation of a new observation is greater than this threshold, then it belongs to the majority class.

3) COST-SENSITIVE MODELS (CS)

The basic idea behind Cost Sensitive models is to assign higher weight to the minority class. It is equivalent to specifying a higher cost to wrongly classify a fraud case.

- *Cost Sensitive C5.0*: Assigning costs to the decision tree is different for each algorithm. When using the CART algorithm, the costs are added to the Gini index at data split. For C5.0, costs are implemented in the decision boundaries,notinthetrainingalgorithm[46].Therefore, the revised decision boundary for classifying an observation in class 1 is:

$$p_1 C_{1/0}\pi_0 > p_0 C_{0/1}\pi_1$$

  where π_i is the prior probability of an observation to be in class i. p_i and $C_{j/i}$ are, respectively, the estimators of theprobabilityofanobservationtobeclassifiedinclass$i$ and the cost of wrongly classifying an observation of class i as j.
- *Cost Sensitive SVM*: Cost Sensitive SVM is computed by assigning weights to each class. In Cost Sensitive SVM, two parameters C^+ and C^- are added [47] instead of having just one cost parameter C like the one in SVM(sectionIII-A.2).Theoptimizationproblemisthen written as follows:

$$\begin{aligned} \text{minimize} \quad & \frac{1}{2}||w||^2 + C^+ \sum_{y_i=1}\zeta_i + C^- \sum_{y_i=-1}\zeta_i \\ \text{subject to} \quad & y_i \times (w^T \delta(x_i) + b) \geq 1 - \zeta_i \\ & \zeta_i \geq 0, i = 1,\ldots,l \end{aligned}$$

IV. EXPERIMENTAL STUDY

In this section, we report our experimental study that we performed with selected machine learning algorithms and imbalance classification approaches.

[1] Available at http://packages.revolutionanalytics.com/datasets/

First, we provide a detailed description of the design of experiments followed by the results and a discussion. Finally, we discuss some critical shortcomings we discovered in our experiments.

A. DESIGN OF EXPERIMENTS

This section briefly presents the workflow of our experiments, the dataset used, the selection of target variables and performance measure.

1) WORKFLOW OF EXPERIMENTS

Our experimental study is organized as follows. The experiments are presented and discussed in two phases. In the first phase, eight classification methods (discussed earlier in Section III-A) are compared. The comparison was carried out with respect to three parameters including the following: *accuracy*, *sensitivity*, and *the Area Under Precision-Recall Curve* (AUPRC). This comparison results in selecting the most suitable algorithms including the following: C5.0, SVM and ANN.

In the second phase, the selected algorithms are used in comparing selected imbalance classification approaches such as *Random Oversampling*, *One-Class Classification* and *Cost Sensitive*. The C5.0 decision tree algorithm is used in the credit card fraud dataset, and compared to C5.0 with Random Oversampling and C5.0 with Cost Sensitive tree. Then, the SVM is used as a binary classification tool, and compared to the One-Class Classification SVM and Cost Sensitive SVM. Also, the ANN is applied and compared to the Auto-Associative Neural Network.

2) DATASET AND VARIABLE SELECTION

The dataset used in our experiment contains credit card fraud
labeled data.[2] It contains ten million credit card transactions described by 8 variables listed here:

- *custID* is an auto increasing integer value that represents the customer ID. This variable is removed later as it has no relevance for detecting fraud.
- *gender* represents the customer's gender.
- *state* represents the state in which the customer lives in the United States.
- *cardholder* is the number of cards that the customer holds (maximum 2).
- *balance* indicates the balance on the credit card in USD.
- *numTrans* is a discrete variable that represents the number of transactions made to date.
- *numIntTrans* is a discrete variable representing the number of international transactions made to date.
- *creditLine*: denotes the customer's credit limit.
- *fraudRisk*: the binary target variable, taking the values 0 denoting legitimate transaction, and 1 denoting fraudulent transaction.

Statistical characteristics of the numerical variables are shown in Table 1. In the data set, gender's frequencies are, respectively, 6,178,231 male (61.7%) and 3,821,769 female (38.3%). Moreover, 596,014 (5.96%) are fraud cases and 9,403,986 (94.04%) are legitimate. These data demonstrate the extreme imbalance problem with 5.96% fraud cases. Data are divided into train set (70%) to create the models, and a test set (30%) to study their performance. Furthermore, we used a smaller dataset that is 2% of the original dataset. We maintained the same imbalance ratio and number of transactions (i.e., 199999 transactions). We processed the dataset and evaluate the performance of the selected methods. Also, a much smaller dataset containing only 1000 transactions is used exceptionally for the NSA algorithm. We chose a small number of transactions since the NSA algorithm is computationally expensive (we will discuss more in detail).

TABLE 1. Descriptive analysis of the numerical variables.

Variable	balance	numTrans	numIntTrans	creditLine
Minimum	0	0	0	1
1^{st} Quartile	0	10	0	4
Median	3706	19	0	6
Mean	4110	28.94	4.04	9.1
3^{rd} Quartile	6000	39	4	11
Maximum	41480	100	60	75

The methods mentioned in the Section III-A are compared using the same reduced data. All the variables mentioned earlier except the *fraudRisk* are used as exploratory variables. The variable *fraudRisk* is the output (response) variable.

3) SELECTION OF PERFORMANCE MEASURES

Typically, ''accuracy'' is the most common parameter for measuring the performance in a classification problem. However, in our case accuracy is not adequate because we are tackling imbalance classification. The *Accuracy* alone as a measure of performance may distort the result of fraudulent transactions (we provide more details later in this section). Therefore, we selected other performance measures and provide details of our selection process.

The fraud percentage in our experiment is 5.96% of fraud cases and 94.04% of legitimate cases, which means that a classification's accuracy rate of less than 94% is not acceptable. Fraud detection may not be achievable by gaining a high accuracy rate. To the best of our knowledge, *sensitivity*, which is the correctly classified observations of the minority class, is critically important for measuring the performance. Therefore, we select sensitivity to compute performance along with accuracy.

To calculate these measures, a confusion in matrix (Table 2) is evaluated using the test set.

TABLE 2. Form of confusion matrix.

Predicted	Actual	
	Legitimate (0)	Fraud (1)
Legitimate (0)	True Negative (TN)	False Negative (FN)
Fraud (1)	False Positive (FP)	True Positive (TP)

The accuracy rate represents the percentage of correctly classified samples:

$$Accuracy = \frac{TP + TN}{total\ number\ of\ observations\ in\ test} \times 100$$

The sensitivity also known as recall or True Positive Rate (TPR) is the proportion of positives correctly classified as positives:

$$Recall = \frac{TP}{number\ of\ actual\ fraud\ observations}total$$

The Receiver Operating Characteristic (ROC) curve is a well-known measure of performance and normally used for classifiers. It is essentially a graphical plot that presents the diagnostic ability of a binary classifier system as its discrimination threshold is varied.[2] It is created by plotting the TPR over the False Positive Rate (FPR). However, in imbalanced classification, this curve can mask poor performance. Thus, we singled out this measure. To the best of our understanding, the Precision-Recall (PR) curve is relatively a better measure because it is more sensitive to the class imbalance than the ROC curve [50]. This curve is defined by plotting precision rate over recall rate. The use of precision instead of FPR, used with ROC curve, allows to capture the effect of large negative samples on the algorithm's performance. The precision represents the fraction of examples classified as positive that are truly positive observations. These two measures are defined as:

$$Precision = \frac{TP}{TP + FP}$$

$$FPR = \frac{FP}{total\ number\ of\ actual\ legitimate\ cases}$$

In the PR plot, the model quality is determined by the proximity of the curve to the upper-right-hand corner. This can be measured with the *Area within the PR Curve (AUPRC)*.

It is worth noting that each of these three performance measures (accuracy, sensitivity and AUPRC) are interpreted differently. Also, none of these methods can be used alone to confirm the competitive quality of the algorithms that we experimented in this paper.

4) SYSTEM SPECIFICATION

We used a simple physical node for our experiment. Given below is the specification of our experiment environment.

- *Operating System*: Windows 10
- *System Type*: 64 bit
- *Processor*: Intel(R) Core (TM) i7 - 7500U
- *Processing Speed*: 2.70GHz 2.90 GHz
- *Memory*: 16 GB
- *Development Language*: R (version 3.5.0)

B. RESULTS AND DISCUSSION

In this section, we discuss the results of our experiments. We conducted experiments with all eight machine learning classification algorithms described in Section III-A. The result of this experiment is summarized in Table 3. According to the results we found, for all algorithms the accuracies are higher than 90% with different sensitivity and AUPRC values. However, based on our study with the results, we concluded that of all these algorithms, C5.0 algorithm, SVM and ANN are the most eligible ones to be used in evaluating the performance of the imbalance classification approaches described in Section III-B. The reason is two-fold: (i) comparing the results produced by the other algorithms, C5.0, SVM, and ANN produced balanced and the most reasonable outcomes in all three measures (Accuracy, Sensitivity, and AURPC); and (ii) these three algorithms are not constrained by mathematical or statistical assumptions.

TABLE 3. Performance of different methods.

Method	Accuracy	Sensitivity	AUPRC	Performance

[2] http://www.ashukumar27.io/roc-auc/

C5.0	96%	43%	0.6	Highest accuracy
SVM	96%	39%	0.63	Highest accuracy
ANN	96%	47%	0.62	Highest accuracy
LR	96%	49%	0.66	Highest AUPRC
NB	93%	56%	0.5	Highest sensitivity
BBN	94%	15%	0.64	High AUPRC
KNN	95%	45%	0.29	Lowest AUPRC
NSA	92%	51%	0.29	Lowest AUPRC

In the following, we provide more details about our experiments and selection of algorithms. NB and NSA have the highest sensitivities, yet the sensitivity is obtained with increasing a number of false alarms (accuracy less than 94%). Moreover, one of the major limitations of NB is the Conditional Independence (CI) assumption, which is violated in our case according to the CI test that uses mutual information. Table 4 shows p-values of this test for all variables. All p-values smaller than 0.05 are considered as significant, and the assumption is not satisfied, for example *balance* and *creditLine*.

The experiments with BBN produced the lowest sensitivity (15%) and the highest AUPRC (0.64) - which is promising. In this case we concluded that the different threshold of class probabilities may produce better results. However, BBN needs normality assumption for the continuous variables, which is not always satisfied. The BBN layout is shown in Figure 2. This structure was implemented based on expert knowledge. The algorithms commonly used to find the BBN layout yielded non-logical results, such as the node *fraud* being orphan, or a parent node, or being only dependent on variables such as *gender*, *state* or *cardholder*.

TABLE 4. p-values for the CI test for all variables pairs given the *fraudRisk* variable.

	gender	*state*	*cardholder*	*balance*	*numTrans*	*numIntTrans*	*creditLine*
gender	NA	0.711	0.138	0	0	0.302	0
state		NA	0.139	0	0	0	0
cardholder			NA	0.221	0.235	0	0.129
balance				NA	0	0	0
numTrans					NA	0	0
numIntTrans						NA	0
creditLine							NA

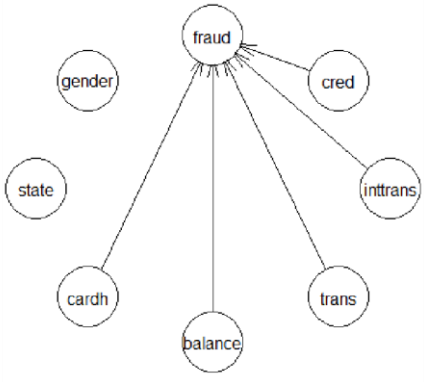

FIGURE 2. BBN layout.

In our experiment with KNN, data are first normalized using the min-max scale. The normalization is important especially when the variables are in diverse scales and ranges. Otherwise, the algorithm will be biased towards variables with larger scales [48]. For example in our case, the bias will be towards the variable ''balance'', whereas it may not be the most important variable. The euclidean distance is used, and five nearest neighbors are considered in the classification. The simplicity of this method does not allow for efficient results as shown in Table 3. We obtained the lowest AUPRC using this method.

We found that the results of the LR algorithm are the best of all. First, the multi-collinearity test was carried out using Variance Inflation Factors (VIF) to make sure that no variable can be written as a linear model in terms of the others. The highest AUPRC was achieved using this method, indicating the best sensitivity with a high accuracy.

The NSA algorithm is a variant of the AIS (Artificial Immune System). In our experiment with NSA, selecting thresholds for the euclidean distance was challenging. After investigating several thresholds, it was set to 100. This method generated a relatively good sensitivity rate but with the lowest accuracy and AUPRC. It is worth noting that, unlike all other methods, a smaller dataset was used to train this algorithm (1000 transactions), while keeping the same imbalance ratio. The reason behind using a small dataset is that the computation cost for large datasets is high; a high-performance system is required to conduct experimentswithNSAonlargerdataset.Thesystemweusedforour experiments was rather a simple system and hence not able to handle high computation costs. Besides, multiple parameters affect the accuracy of this model, such as *the threshold* and *the size of the detector set*. We found a set of 2000 detectors; yet a bigger set is needed to achieve better results that would increase training time and computation cost significantly.

Figure 3 presents different PR curves for the methods singled out for our experiments with imbalance classification approaches. According to the results of AUPRC shown in Table 3, LR is the best method. Yet, LR was not chosen because LR needs assumptions and conditions for variables which is not feasible in our case. The other methods, KNN and NSA, resulted in either low sensitivity, or low accuracy due to high sensitivity, which leads to the low AUPRC.

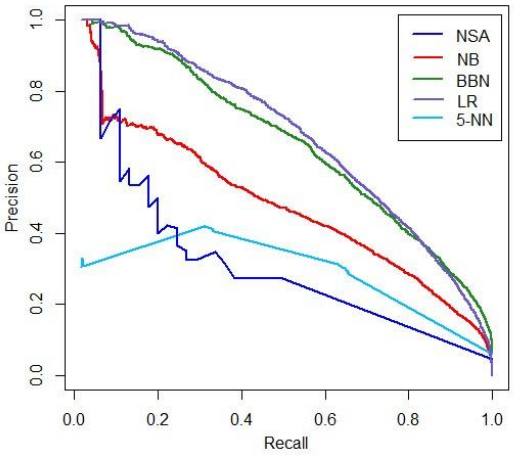

FIGURE 3. Comparing PR curves.

After selecting C5.0, SVM and ANN algorithms, we studied the performance of the imbalance approaches and their improvementstothesemethods.WebeginwiththeC5.0algorithm. The confusion matrices of the C5.0 methods are as follows:

C5.0

Predicted	Actual	
	0	1
0	55838	2053
1	546	1563

RO C5.0

Predicted	Actual	
	0	1
0	52988	1197
1	3396	2419

CS C5.0

Predicted	Actual	
	0	1
0	54200	1252
1	2184	2364

In the experiment with random oversampling, we replicated each observation on average 15 times due to the ratio of imbalance.

For the cost sensitive approach, the two new parameters that are added, C^+ and C^-, are equal to 3 and 1 respectively. In other words, the cost of wrongly classifying a fraud is considered three times the cost of wrongly classifying a legitimate transaction. The trees are too complex to be visualized (145 leaves for C5.0, 5020 leaves for RO C5.0, and 581 for CS C5.0). According to the tree's algorithm, the most important variables contributing to the discrimination are *balance*, *creditLine*, *numTrans* and *numIntTrans*.

SVM

Predicted	Actual	
	0	1
0	56049	2190
1	335	1426

CS SVM

Predicted	Actual	
	0	1
0	54723	1268
1	1661	2348

OCC SVM

Predicted	Actual	
	0	1
0	51349	2522
1	5035	1094

In the case of SVM, like the CS C5.0 the parameters are considered 1 denotes not fraud class and 3 denotes fraud. For OCC, in the training set only the fraud class is used. For the test set, the observations of both classes are used to evaluate this approach.

ANN

Predicted	Actual	
	0	1
0	55824	1893
1	560	1723

AANN

Predicted	Actual	
	0	1
0	44843	1924
1	11541	1692

In the experiment with ANN (shown in Figure 4) the network is composed of an input layer (7 nodes), one hidden layer (3 nodes) and an output layer. The activation function used is the sigmoid activation function. The algorithm used to adjust weights is the Resilient back propagation algorithm (Rprop). Figure 5 shows the network generated by AANN.
It is composed of an input layer and an output layer with 7 nodes, and of 3 hidden layers with 2, 1 and 2 nodes respectively.

In the experiment with AANN, only the fraud cases are used to train the network. The Mean Absolute Error (MAE) is calculated as an average error and used as a threshold in the testing phase. For each observation, having a MAE higher than the training's MAE is considered a fraud.

Table 5 shows the accuracy, the sensitivity and the AUPRC of the methods employed for imbalanced classification. These curves are shown in the following Figures 6, 7 and 8. Note that the PR curves could not be plotted for CS C5.0, as the final class prediction for a sample according to the equation presented in Section III-B.3, is a function of the

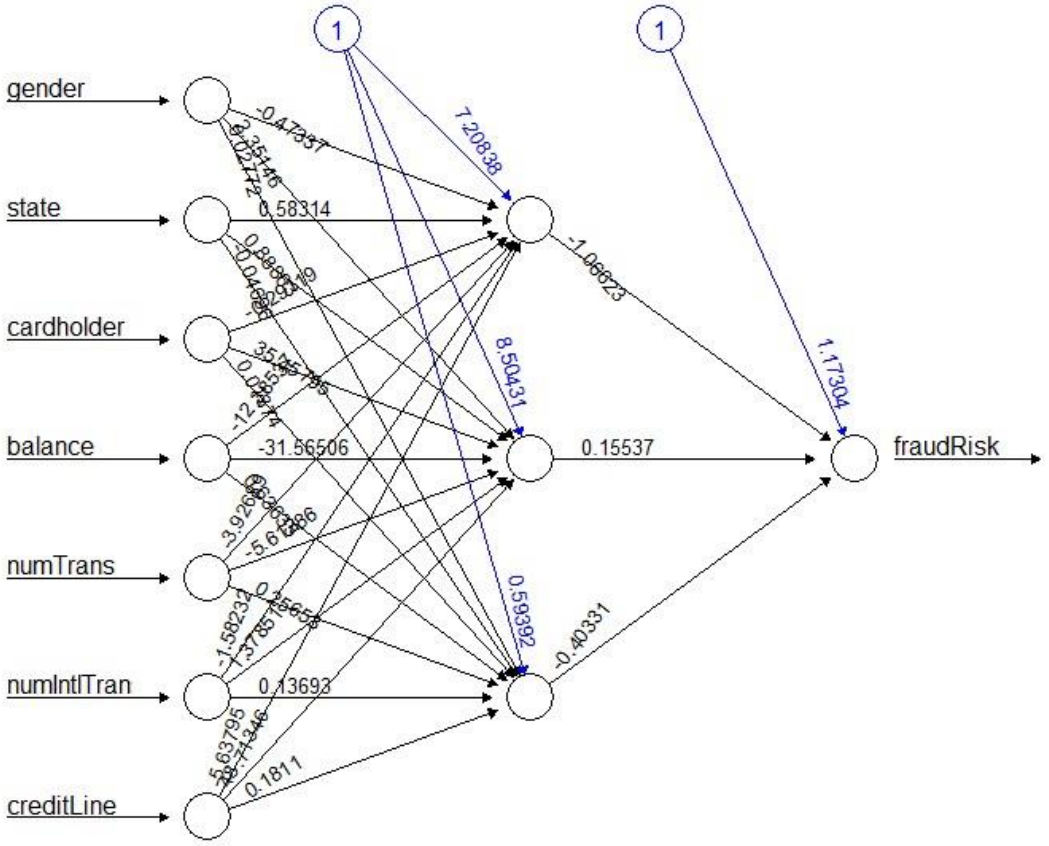

FIGURE 4. ANN plot.

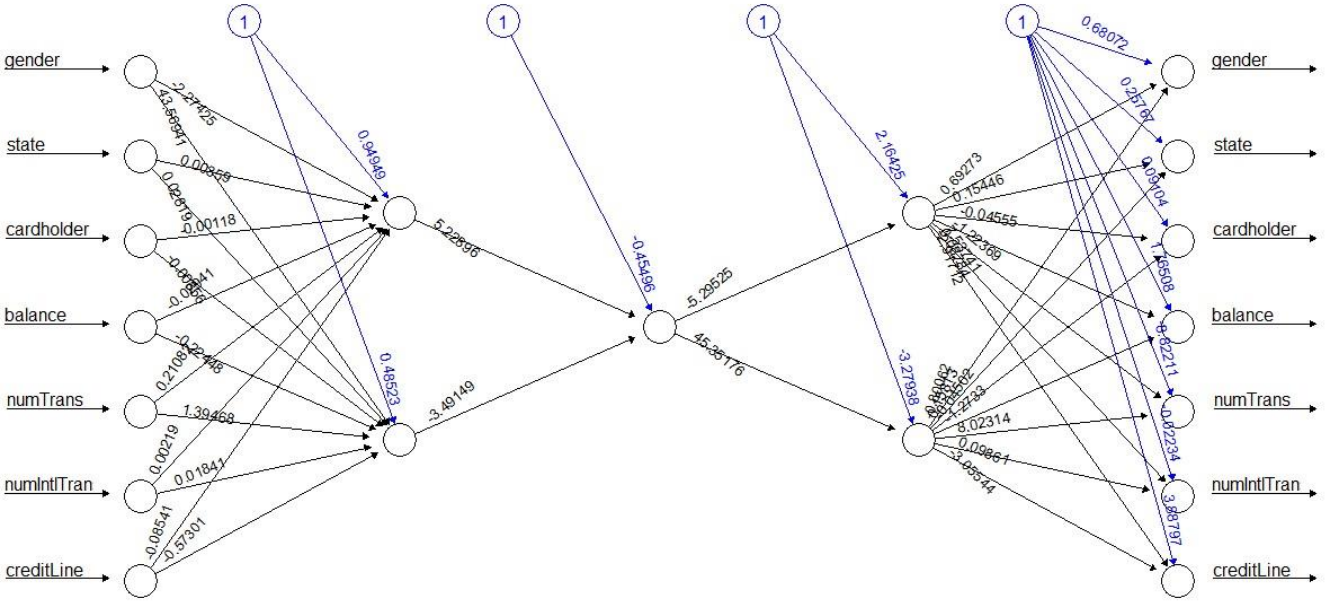

FIGURE 5. AANN plot.

class probability and the cost structure, and not just of Table 5 shows that the accuracy for all methods is higher

the class probability [46]. Also, the class probabilities for than 94% with different sensitivity levels, except for the OCC one-class classification SVM are not supported yet. approaches (for SVM and ANN). According to AUPRC,

TABLE 5. Table summarizing the performance measures of imbalance approaches.

Method	Accuracy	Sensitivity	AUPRC
C5.0	96%	43%	0.6
RO C5.0	92%	66%	0.52
CS C5.0	94%	65%	Could not be plotted
SVM	96%	39%	0.63
OCC SVM	87%	30%	Could not be plotted
CS SVM	95%	65%	0.62
ANN	96%	47%	0.62
AANN	77%	46%	0.11

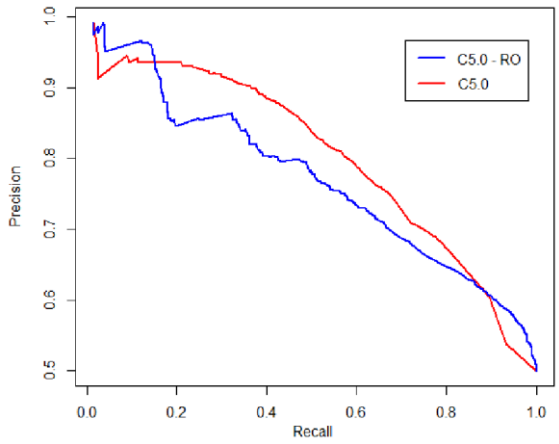

FIGURE 6. PR curves for C5.0 methods.

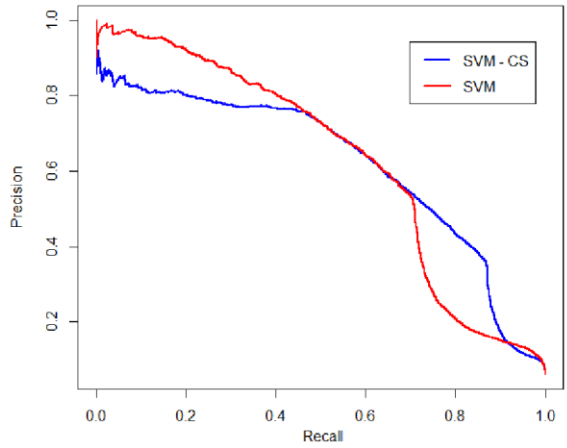

FIGURE 7. PR curves for SVM methods.

the original methods (C5.0, SVM and ANN) are better than those using imbalanced classification approaches.

According to AUPRC, even LR performs better. Table 5 shows that while the imbalanced classification approaches increase the sensitivity, accuracy decreases. It is important for all the models that the CS approach improves performance. However, the OCC approach severely affects the algorithm

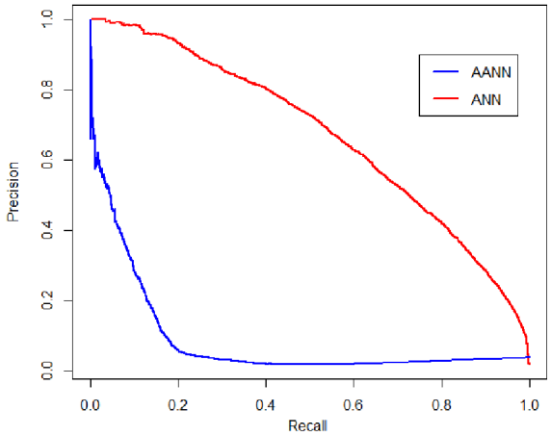

FIGURE 8. PR curves for ANN methods.

in terms of accuracy and sensitivity. We believe that the reason behind this decrease in performance is overfitting of data since the OCC approaches are built using the fraud observations only.

C. SHORTCOMINGS OF EXISTING METHODS

In this experimental study, we discovered several shortcomings of existing methods. We found that the approaches designed specifically to tackle the imbalance problem are not adequately effective. While these approaches improve sensitivity, this improvement leads to an increase in the number of false alarm and thus also to a drop in accuracy and AUPRC. Practically speaking, this can be costly for the financial institution just the same way a fraud event costs. Even a minimal deterioration of accuracy, say 1% hides a large misclassification rate of the majority group.

The problem is summarized as follows: using imbalanced classification approaches, the number of false alarms generated is higher than the number of frauds that are detected.

The results of this experimental study greatly motivated us to explore the other methods that focus on detecting the hidden patterns of fraud, with minimum misclassification.

V. CONCLUSION AND FURTHER WORK

Fraud is a critical concern for business organizations as well as individuals. Different types of fraudulent activities cost billions of dollars every year. Of all types, credit card fraud is the most common and costly one which has raised huge concern globally. Detecting credit card fraud is enormously challenging. Several problems make it hard to find solutions for fraud detection, among which the class imbalance problem is one of the major ones. Several solutions have been proposed to deal with this problem. In this paper, we studied these solutions.

We compared the performance of eight machine learning methods applied to credit card fraud detection and to identify their weaknesses. More precisely, we compared the imbalanced classification approaches and studied how effective they are in the case of extreme imbalance. We found that the LR, C5.0 decision tree algorithm, SVM and ANN are the best methods according to the 3 considered performance measures (Accuracy, Sensitivity andAUPRC). We used C5.0, SVMand ANN, taking into account that the other models assumptions and conditions are not easy to meet. Our experimental study revealed that the approaches normally used to solve imbalance problems may have unpleasant consequences when the imbalanceisextreme,suchasgeneratingasignificantnumber of false positives. Even though these techniques improve the classifier's performance, significant fraud cases continue to go undetected. Also, our study showed that considering just one performance measure for imbalanced learning is misleading.

We encountered a few problems during the study which need further investigation. The first is finding an optimized BBN structure. On the other hand, the time consumption of the NSA training phase is difficult to deal with. Moreover, finding optimized parameters such as the costs for CS approaches, k for the KNN, and thresholds used for AANN and NSA, is not straightforward.

Several works are lined up that will be carried out in the next step of this experimental study. We will develop a model for the class imbalance problem to find a trade-off between sensitivity and accuracy. We will develop a Big Data driven ecosystem and test our model with massive scale data. Size of the data was an obvious limitation of our study as we could not conduct our study on large amounts of data. Therefore, we plan to develop a scalable environment using Big Data technology.

REFERENCES

[1] P. Richhariya and P. K. Singh, ''Evaluating and emerging payment card fraud challenges and resolution,'' *Int. J. Comput. Appl.*, vol. 107, no. 14, pp. 5–10, Jan. 2014.

[2] S. Bhattacharyya, S. Jha, K. Tharakunnel, and J. C. Westland, ''Data mining for credit card fraud: A comparative study,'' *Decis. Support Syst.*, vol. 50, no. 3, pp. 602–613, 2011.

[3] A. Dal Pozzolo, O. Caelen, Y.-A. L. Borgne, S. Waterschoot, and G. Bontempi, ''Learned lessons in credit card fraud detection from a practitioner perspective,'' *Expert Syst. Appl.*, vol. 41, no. 10, pp. 4915–4928, 2014.

[4] C. Phua, D. Alahakoon, and V. Lee, ''Minority report in fraud detection: Classification of skewed data,'' *ACM SIGKDD Explorations Newslett.*, vol. 6, no. 1, pp. 50–59, 2004.

[5] Z.-H. Zhou and X.-Y. Liu, ''Training cost-sensitive neural networks with methods addressing the class imbalance problem,'' *IEEE Trans. Knowl. Data Eng.*, vol. 18, no. 1, pp. 63–77, Jan. 2006.

[6] S. Ertekin, J. Huang, and C. L. Giles, ''Active learning for class imbalance problem,'' in *Proc. 30th Annu. Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, 2007, pp. 823–824.

[7] M. Wasikowski and X. Chen, ''Combating the small sample class imbalance problem using feature selection,'' *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 10, pp. 1388–1400, Oct. 2010.

[8] S. Wang and X. Yao, ''Multiclass imbalance problems: Analysis and potential solutions,'' *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 42, no. 4, pp. 1119–1130, Aug. 2012.

[9] R. J. Bolton and D. J. Hand, ''Statistical fraud detection: A review,'' *Stat. Sci.*, vol. 17, no. 3, pp. 235–249, Aug. 2002.

[10] D. J. Weston, D. J. Hand, N. M. Adams, C. Whitrow, and P. Juszczak, ''Plastic card fraud detection using peer group analysis,'' *Adv. Data Anal. Classification*, vol. 2, no. 1, pp. 45–62, 2008.

[11] E. Duman and M. H. Ozcelik, ''Detecting credit card fraud by genetic algorithm and scatter search,'' *Expert Syst. Appl.*, vol. 38, no. 10, pp. 13057–13063, Sep. 2011.

[12] K. Ramakalyani and D. Umadevi, ''Fraud detection of credit card payment system by genetic algorithm,'' *Int. J. Sci. Eng. Res.*, vol. 3, no. 7, pp. 1–6, Jul. 2012.

[13] P. J. Bentley, J. Kim, G.-H. Jung, and J.-U. Choi, ''Fuzzy darwinian detection of credit card fraud,'' in *Proc. 14th Annu. Fall Symp. Korean Inf. Process. Soc.*, Oct. 2000, pp. 1–4.

[14] A. Srivastava, A. Kundu, S. Sural, and A. K. Majumdar, ''Credit card fraud detection using hidden Markov model,'' *IEEE Trans. Depend. Sec. Comput.*, vol. 5, no. 1, pp. 37–48, Jan./Mar. 2008.

[15] S. Esakkiraj and S. Chidambaram, ''A predictive approach for fraud detection using hidden Markov model,'' *Int. J. Eng. Res. Technol.*, vol. 2, no. 1, pp. 1–7, Jan. 2013.

[16] J. S. Mishra, S. Panda, and A. K. Mishra, ''A novel approach for credit card fraud detection targeting the Indian market,'' *Int. J. Comput. Sci.*, vol. 10, no. 3, pp. 172–179, May 2013.

[17] A. Brabazon, J. Cahill, P. Keenan, and D. Walsh, ''Identifying online credit card fraud using artificial immune systems,'' in *Proc. IEEE Congr. Evol. Comput.*, Jul. 2010, pp. 1–7.

[18] N. Wong, P. Ray, G. Stephens, and L. Lewis, ''Artificial immune systems for the detection of credit card fraud: An architecture, prototype and preliminary results,'' *Inf. Syst.*, vol. 22, no. 1, pp. 53–76, Jan. 2012.

[19] D. Sánchez, M. A. Vila, L. Cerda, and J. M. Serrano, ''Association rules applied to credit card fraud detection,'' *Expert Syst. Appl.*, vol. 36, no. 2, pp. 3630–3640, 2009.

[20] Y. Sahin, S. Bulkan, and E. Duman, ''A cost-sensitive decision tree approach for fraud detection,'' *Expert Syst. Appl.*, vol. 40, no. 15, pp. 5916–5918, Nov. 2013.

[21] A. C. Bahnsen, A. Stojanovic, D. Aouada, and B. Ottersten, ''Costsensitive credit card fraud detection using Bayes minimum risk,'' in *Proc. 12th Int. Conf. Mach. Learn. Appl.*, Dec. 2013, pp. 333–338.

[22] A. E. Pasarica, ''Card fraud detection using learning machines,'' Bull. Polytech. Inst. Jassy, Fac. Cybern., Statist. Econ. Inform., Bucharest, Romania,Tech. Rep., 2014, pp. 29–45.

[23] Y. Sahin and E. Duman, ''Detecting credit card fraud by decision trees and support vector machines,'' in *Proc. Int. Multiconf. Eng. Comput. Sci.*, vol. 1, Mar. 2011, pp. 442–447.

[24] V. R. Ganji and S. N. P. Mannem, ''Credit card fraud detection using antik nearest neighbor algorithm,'' *Int. J. Comput. Sci. Eng.*, vol. 4, no. 06, pp. 1035–1039, Jun. 2012.

[25] S. Ghosh and D. L. Reilly, ''Credit card fraud detection with a neural-network,'' in *Proc. 27th Hawaii Int. Conf. Syst. Sci.*, Jan. 1994, pp. 621–630.

[26] J. R. Dorronsoro, F. Ginel, C. Sgnchez, and C. S. Cruz, ''Neural fraud detection in credit card operations,'' *IEEE Trans. Neural Netw.*, vol. 8, no. 4, pp. 827–834, Jul. 1997.

[27] V. Zaslavsky and A. Strizhak, ''Credit card fraud detection using selforganizing maps,'' *Inf. Secur.*, vol. 18, no. 1, pp. 48–63, 2006.

[28] F. N. Ogwueleka, ''Data mining application in credit card fraud detection system,'' *J. Eng. Sci. Technol.*, vol. 6, no. 3, pp. 311–322, Jun. 2011.

[29] R. Patidar and L. Sharma, ''Credit card fraud detection using neural network,'' *Int. J. Soft Comput. Eng.*, vol. 1, pp. 32–38, Jun. 2011.

[30] M. Syeda, Y.-Q. Zhang, Y. Pan, and C. Science, ''Parallel granular neural networks for fast credit card fraud detection,'' in *Proc. IEEE Int. Conf. Fuzzy Syst.*, vol. 1, May 2002, pp. 572–577.

[31] S. Maes, K. Tuyls, B. Vanschoenwinkel, and B. Manderick, ''Credit card fraud detection using Bayesian and neural networks,'' in *Proc. 1st Int. Naiso Congr. Neuro Fuzzy Technol.*, Jan. 2002, pp. 261–270.

[32] C. Whitrow, D. Hand, J. Juszczak, D. Weston, and N. M. Adams, ''Transaction aggregation as a strategy for credit card fraud detection,'' *Data Mining Knowl. Discovery*, vol. 18, no. 1, pp. 30–55, Feb. 2009.

[33] B. Subashini and K. Chitra, ''Enhanced system for revealing fraudulence in credit card approval,'' *Int. J. Eng. Res. Technol.*, vol. 2, no. 8, pp. 936–949, Aug. 2013.
[34] N. Mahmoudi and E. Duman, ''Detecting credit card fraud by modified Fisher discriminant analysis,'' *Expert Syst. Appl.*, vol. 42, no. 5, pp. 2510–2516, Apr. 2014.
[35] H. Hormozi, M. K. Akbari, E. Hormozi, and M. S. Javan, ''Credit cards fraud detection by negative selection algorithm on Hadoop (To reduce the training time),'' in *Proc. 5th Conf. Inf. Knowl. Technol.*, May 2013, pp. 40–43.
[36] J. T. S. Quah and M. Sriganesh, ''Real-time credit card fraud detection using computational intelligence,'' *Expert Syst. Appl.*, vol. 35, no. 4, pp. 2007–2009, Nov. 2008.
[37] A. Kundu, S. Panigrahi, S. Sural, and A. K. Majumdar, ''BLAST-SSAHA hybridization for credit card fraud detection,'' *IEEE Trans. Dependable Secure Comput.*, vol. 6, no. 4, pp. 309–315, Oct./Dec. 2009.
[38] K. K. Sherly and R. Nedunchezhian, ''Boat adaptive credit card fraud detection system,'' in *Proc. IEEE Int. Conf. Comput. Intell. Comput. Res.*, Dec. 2010, pp. 1–7.
[39] T. Minegishi and A. Niimi, ''Detection of fraud use of credit card by extended VFDT,'' in *Proc. World Congr. Internet Secur. (WorldCIS)*, Feb. 2011, pp. 166–173.
[40] F. Carcillo, Y. L. Borgne, O. Caelen, and G. Bontempi, ''Streaming active learning strategies for real-life credit card fraud detection: Assessment and visualization,'' *Int. J. Data Sci. Anal.*, vol. 5, no. 4, pp. 285–300, 2018.
[41] B. Krawczyk, ''Learning from imbalanced data: Open challenges and future directions,'' *Prog. Artif. Intell.*, vol. 5, no. 4, pp. 221–232, 2016.
[42] S. Kamaruddin and V. Ravi, ''Credit card fraud detection using big data analytics : Use of PSOAANN based one-class classification,'' in *Proc. Int. Conf. Inform. Anal.*, Aug. 2016, pp. 1–8.
[43] W. Wei, J. Li, L. Cao, Y. Ou, and J. Chen, ''Effective detection of sophisticated Online banking fraud on extremely imbalanced data,'' *World Wide Web*, vol. 16, no. 4, pp. 449–475, 2013.
[44] T. M. Padmaja, N. Dhulipalla, R. S. Bapi, and P. R. Krishna, ''Unbalanced data classification using extreme outlier elimination and sampling techniquesforfrauddetection,''in*Proc.15thInt.Conf.Adv.Comput.Commun. (ADCOM)*, Dec. 2007, pp. 511–516.
[45] P. K. Chan, W. Fan, A. L. Prodromidis, and S. J. Stolfo, ''Distributed data mining in credit card fraud detection,'' *IEEE Intell. Syst. Appl.*, vol. 14, no. 6, pp. 67–74, Nov./Dec. 1999.
[46] M. Kuhn and K. Johnson, *Applied Predictive Modeling*. New York, NY, USA: Springer, 2013.
[47] C.-C. Chang and C.-J. Lin, ''LIBSVM: A library for support vector machines,''*ACMTrans.Intell.Syst.Technol.*,vol.2,no.3,p.27,Apr.2011. [48] J. Han, M. Kamber, and J. Pei, *Data Mining: Concepts and Techniques*. San Mateo, CA, USA: Morgan Kaufman, 2003.
[49] B. Schölkopf, J. C. Platt, J. C. Shawe-Taylor, A. J. Smola, and R. C. Williamson, ''Estimating the support of a high-dimensional distribution,'' *Neural Comput.*, vol. 13, no. 7, pp. 1443–1471, Jul. 2001.
[50] H. He and E. A. Garcia, ''Learning from imbalanced data,'' *IEEE Trans. Knowl. Data Eng.*, vol. 21, no. 9, pp. 1263–1284, Sep. 2009.

SARA MAKKI is currently pursuing the Ph.D. degree with the University of Claude Bernard University Lyon 1 and Lebanese University. She is currently involved in the solution for fraud detection in banks and financial institutions using big data analytics. Her research interests include the intersection of statistical analysis, machine learning, and financial risk management.

ZAINAB ASSAGHIR is currently an Associate Professor with the Applied Mathematics Department, Faculty of Science, Lebanese University. Her research interests include uncertainty estimation and modeling, risk mathematical estimation, and data analysis and classification to biology, finance, actuarial sciences, and agronomy.

YEHIA TAHER is currently an Associate Professor with Versailles Saint-Quentin-en-Yvelines University and also a member of the DAVID Research Laboratory. His research interests include a broad field of computer science, including real-time analysis of big data, complex event processing, cloud computing, business process management, and service-oriented computing.

RAFIQUL HAQUE received the Ph.D. degree in computer science and information systems. He is currently the Chief Technology Officer and the Co-Founder of Cognitus. He has dedicated most of his career to research. His research areas include big data engineering and analytics, scalable and distributed computing, service-oriented computing (SOC), and cloud computing. He led the development of several innovative methods within big data. He has involved in various projects funded by the European Union, Enterprise Ireland, SFI, and ANR. He has written several research papers in conferences and journals

published by the IEEE, ACM, and Springer.

MOHAND-SAÏD HACID is currently a Professor with Claude Bernard University Lyon 1 and the Director of the InfoRmatic Laboratory in Image and Information Systems. His research focuses on the design, implementation, and practical use of formalism and inference techniques that can improve the level of abstraction in the design of advanced information systems. His areas of research include the integration of representational knowledge and reasoning techniques for new applications, such as Web services, semantic web, data security, and massive data management.

HASSAN ZEINEDDINE is currently a Mathematics Professor with the Pure Mathematics Department, Faculty of Science, Lebanese University, and is the former Dean of the Faculty of Science. His areas of research include linear algebra, applied mathematics, and data analysis with applications in biology, medical sciences, and finance.