ตรวจจับการฉ้อโกงบัตรเครดิตด้วย Logistic Regression ใน Machine Learning

Domain

การฉ้อโกงบัตรเครดิต

มารีอา อูกนิช

Digital Business Transformation, College of Innovation, Thammasat University, Thailand

Abstract

สร้าง Logistic Regresssion Model ในการตรวจจับการฉ้อโกงบัตรเครดิตจากชุดข้อมูลที่ประกอบไปด้วยธุรกรรมที่เกิดขึ้นภายใน 2 วันในเดือนกันยายน ปี 2013 ของผู้ถือครองบัตรเครดิตสัญชาติยุโรป การทำธุรกรรมจำนวน 284,807 ครั้ง มีการฉ้อโกงเกิดขึ้นทั้งหมด 492 ครั้ง นับเป็น 0.17% ของการทำธุรกรรมทั้งหมดในชุดข้อมูล ทำให้ชุดข้อมูลมีความไม่สมดุลสูงมาก เราจัดการกับความไม่สมดุลนี้ด้วย SMOTE (synthetic minority oversampling technique) และสร้าง Logistic Regression Model ในการตรวจจับการฉ้อโกงบัตรเครดิต จากการตรวจสอบความถูกต้องของการประมวลผลของโมเดล โมเดลที่สร้างมีความถูกต้อง 92% โดยดูจากค่า accuracy, precision, recall, f1 score และ ROC Curve

Keyword

การตรวจจับการฉ้อโกงบัตรเครดิต, Logistic Regression, Supervised Machine Learning Technique

Introduction

บริษัทที่รับชำระเงินบัตรผ่านบัตรเครดิตต้องสามารถตรวจสอบได้ว่าธุรกรรมแต่ละอันนั้นเป็นธุรกรรมที่ฉ้อโกงหรือไม่ เพื่อที่ลูกค้าจะไม่ถูกเรียกเก็บเงินสำหรับสินค้าที่พวกเขาได้ซื้อ

อ้างอิงจาก Javelin Strategy & Research (https://www.javelinstrategy.com/sites/default/files/17-1001J-2017-LL-Identity-Fraud-Hits-Record-Highs-Javelin.pdf) สถาบันการเงินใช้เวลา กว่า 40 วัน ในการตรวจจับการฉ้อโกง และการฉ้อโกงยังส่งผลกระทบต่อธนาคารที่ให้บริการชำระเงินออนไลน์ด้วย ตัวอย่างผลกระทบเช่น ลูกค้า 20 เปอร์เซ็นต์เปลี่ยนธนาคารผู้ให้บริการหลังจากพบปัญหาการฉ้อโกง

เว็บไซต์ altexsoft.com ให้ข้อมูลว่าการตรวจจับการฉ้อโกงด้วย Machine Learning (ML) ได้รับความสนใจเป็นอย่างมากในช่วงไม่กี่ปีที่ผ่านมา และอุตสาหกรรมได้เปลี่ยนความสนใจจากระบบตรวจจับการฉ้อโกงแบบ rule-based fraud detection systems ไปเป็นการแก้ปัญหาแบบใช้ Machine Learning แทน

ยิ่งไปกว่านั้น Eric Knorr (https://www.infoworld.com/article/2907877/how-paypal-reduces-fraud-with-machine-learning.html) สัมภาษณ์ Dr. Hui Wang ผู้อำนวยการอาวุโสด้านวิทยาศาสตร์ความเสี่ยงของ PayPal โดย PayPal ใช้ machine learning algorithms ทั้งหมด 3 ประเภท สำหรับการจัดการ

ความเสี่ยง ได้แก่ linear, neural network และ deep learning ผลลัพธ์แสดงให้เห็นว่า การจัดการความเสี่ยงในหลายๆเคส มีประสิทธิภาพที่สุดเมื่อใช้ทั้งสามวิธีพร้อมกัน

Logistic regression เป็นวิธีการทางสถิติที่ได้รับการยอมรับอย่างดี เป็น Machine Learning technique แบบ supervised learning algorithm ซึ่งใช้สำหรับปัญหาแบบ classification โดย Algorithm นี้สามารถทำนายผลลัพธ์แบบ binomial หรือ multinomial ได้ ซึ่งหมายความว่าสามารถทำนายได้ว่าธุรกรรมนั้นเป็นการฉ้อโกงหรือไม่

Objective

Data Mining Technique ที่ใช้ในรายงานนี้คือ Logistic Regression ซึ่งเป็น algorithm แบบ classification โดยปัญหาที่กล่าวไปข้างต้นในบทนำ คือปัญหาประเภท Binary Classification problems และชุดข้อมูลมี output เป็น binary (0,1 - โดย 0 ไม่ใช่การฉ้อโกง และ 1 เป็นการฉ้อโกง) ดังนั้น Logistic Regression จึงใช้ในกรณีนี้ได้

Literature review

การตรวจจับธุรกรรมการฉ้อโกงมีความสำคัญอย่างยิ่งสำหรับบริษัทบัตรเครดิตใดๆ โดยปัญหาเกี่ยวกับการฉ้อโกงคือปัญหาที่สามารถตรวจจับได้โดยใช้ Logistic Regression ซึ่งเป็นเทคนิคทั่วไปที่รู้จักกันดีในการแก้ปัญหาแบบ Binary Classification problems

อย่างไรก็ตามแนวทางปฏิบัติที่ดีที่สุดคือการใช้ร่วมกับ Supervised Machine Learning methods อื่นๆ เช่น Random Forest, Support vector machine, K-Nearest Neighbours, Neural Networks และ Deep Neural Networks เป็นต้น เพื่อให้สามารถนำไปประยุกต์ใช้กับปัญหาที่ซับซ้อนได้

งานศึกษาหลายชิ้นแสดงให้เห็นว่าความแม่นยำของการตรวจจับการฉ้อโกงนั้นมีประสิทธิภาพมากยิ่งขึ้นหากใช้เทคนิคอย่างน้อย 2 เทคนิคร่วมกัน

- การตรวจจับการฉ้อโกงบัตรเครดิตโดยใช้ Logistic Regression และ Bayesian Network (Bala and Garg, 2019)
  ใช้ Logistic Regression ในการประเมินรูปแบบความเสี่ยงตามกฎที่กำหนดและส่งต่อข้อมูลที่เกี่ยวข้องไปยัง Bayesian network เพื่อจำแนกประเภทและตรวจจับระดับการฉ้อโกง จากนั้นประเมินประสิทธิภาพขั้นตอนการทำงานที่กล่าวของ model เพื่อดูว่าสามารถตรวจจับการฉ้อโกงบัตรเครดิตได้อย่างมีประสิทธิภาพหรือไม่

- การตรวจจับการฉ้อโกงบัตรเครดิตโดย ANN และ Logistic Regression (Sahin and Duman, 2011)
  ประสิทธิภาพของ classifier models ที่สร้างขึ้นโดยใช้วิธีการ ANN และ Logistic Regression แสดงความถูกต้องในการตรวจจับการฉ้อโกง โดยผลปรากฏว่าการใช้วิธีการทั้ง 2 วิธีร่วมกัน ช่วยเพิ่มประสิทธิภาพให้แก่โมเดลมากขึ้น แต่หากแยกกันแล้ว วิธี ANN มีประสิทธิภาพดีกว่า วิธีแบบ Logistic Regression

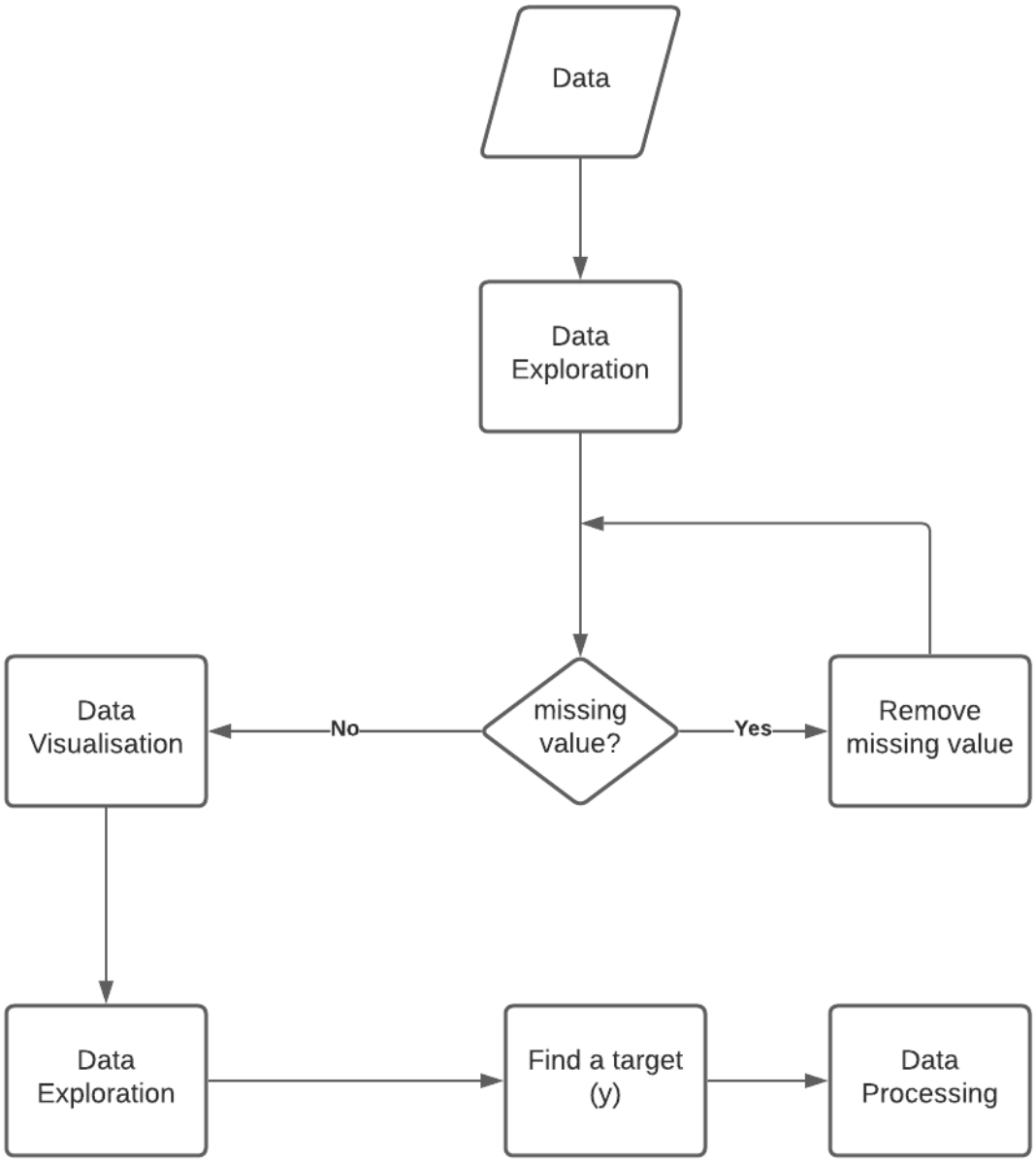
- การสร้างแบบจำลองเชิงคาดการณ์สำหรับการตรวจจับการฉ้อโกงบัตรเครดิตโดยใช้การวิเคราะห์ข้อมูล (Patil, Nemade และ Soni, 2018)
  การใช้แบบจำลองการวิเคราะห์ใช้เพื่อตรวจสอบว่าธุรกรรมขาเข้าเป็นธุรกรรมที่ถูกต้องตามกฎหมายหรือไม่ Logistic Regression และ Decision Tree ถูกนำมาใช้เพื่อตรวจจับการฉ้อโกง ผลที่ได้จากการตรวจจับการฉ้อโกง แบบ real time ปรากฏว่าช่วย ลดความเสี่ยงของการฉ้อโกงและเพิ่มความพึงพอใจให้ลูกลูกค้าของสถาบัน

อย่างไรก็ตามในชุดข้อมูลที่ไม่มีความซับซ้อน Logistic Regression เพียงอย่างเดียวก็สามารถช่วยทำนายและให้ผลลัพธ์ที่น่าพึงพอใจได้

Data Pre-Processing

ในขั้นตอน data pre-processing เราต้องนำเข้าข้อมูลและทำความเข้าใจข้อมูล เพื่อดูว่าเราจะนำข้อมูลนี้มาใช้อย่างไร และโมเดลใดที่เหมาะสมกับข้อมูลนี้

ขั้นตอนการทำ Data Pre-Processing แสดงขั้นตอนตาม flowchart ด้านล่างนี้

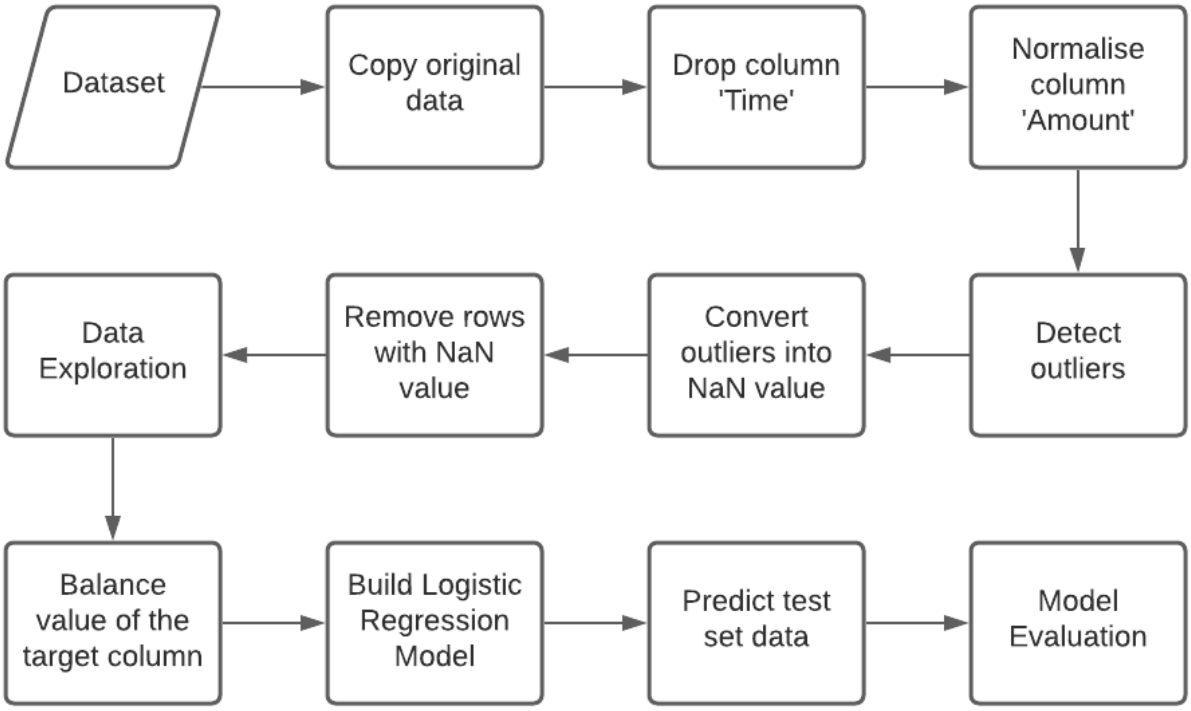


เนื่องจาก dataset ที่เรานำมาใช้ไม่มี missing value หมายความว่า dataset นี้ค่อนข้าง clean พอสมควร หลังจากนั้นเราได้ทำ data visualisation เพื่อดูความสัมพันธ์ของค่าในคอลัมน์ต่างๆ และดูว่า target ที่เราต้องหาคือตัวไหน
จากขั้นตอน data exploration เราพบว่า target ของเราอยู่ในคอลัมน์ชื่อ Class ซึ่งแสดงผลแบบ Binary (0,1) เมื่อ 0 คือ ธุรกรรมปกติ และ 1 คือ ธุรกรรมที่เป็นการฉ้อโกง
เมื่อทำความเข้าใจข้อมูลได้เพียงพอแล้ว เราจะข้ามไปขั้นตอนของ Data Processing เพื่อจัดเตรียมข้อมูลให้พร้อมต่อการทำโมเดล

Data Processing

ขั้นตอนนี้จะเป็นขั้นตอนการจัดการข้อมูลให้พร้อมต่อการนำมาทำโมเดล

ขั้นตอนการทำ Data Processing ในรายงานนี้ แสดงขั้นตอนตาม flowchart ด้านล่างนี้



เริ่มแรกเราจะ copy original dataset ไว้ก่อน เนื่องจากเราต้องการคำนวณหลังจาก processing เสร็จแล้วว่า ก่อนและหลัง processing เราเหลือข้อมูลที่จะนำมาทำเท่าไหร่

ขั้นตอนต่อมาคือดูว่าคอลัมน์ไหนที่ไม่จำเป็นต้องใช้ ให้ drop ออก

จากนั้นเราทำการ scale คอลัมน์ ‘Amount’ เพื่อให้อยู่ใน range 0-1 แทน เพราะ range เดิมนั้นสูงมาก

จากนั้นเราต้องทำการกำจัด outlier เนื่องจาก outlier จะส่งผลต่อการทำนายผลของโมเดลของเราทำให้โมเดลของเราไม่มีความแม่นยำ

ขั้นตอนสุดท้ายก่อนสร้างโมเดล คือ การทำให้ข้อมูลใน target คอลัมน์มีความสมดุลเสียก่อน เนื่องจาก row ข้อมูลมีความไม่สมดุลอย่างมาก (99.83% ต่อ 0.17%)

ประเภทของปัญหาที่เราต้องการ Predict เป็นปัญหาประเภท Binary Classification คือ การค้นหาคำตอบว่าคำตอบจะเป็น 0 หรือ 1 (ใช่ หรือ ไม่ใช่)

โมเดลที่ได้รับความนิยมในการแก้ปัญหาประเภทนี้ มีดังนี้

1. Logistic Regression
2. k-Nearest Neighbors
3. Decision Trees
4. Support Vector Machine
5. Naive Bayes

ในรายงานนี้ เราเลือกโมเดล Logistic Regression ในการแก้ปัญหา

หลังจากเลือกโมเดลที่เหมาะสมกับปัญหาแล้ว เราได้สร้างโมเดล และทำการทำนายผลบนชุดข้อมูล test set

---

Model Evaluation

ต่อมาเราต้องคำนวณว่าโมเดลที่เราสร้างมีความถูกต้องมากน้อยแค่ไหน เราจะใช้ classification_report จาก sklearn ค่าที่ได้จะมีค่า 0-1 โดย 1 แปลว่ามีความแม่นยำมาก

```
#CALCULATE THE ACCURACY USING CLASSIFICATION_REPORT SKLEARN
report = classification_report(y_test,y_prediction)
print(report)

Output:
              precision    recall  f1-score   support

           0       0.95      0.88      0.91     43968
           1       0.89      0.95      0.92     43824

    accuracy                           0.92     87792
   macro avg       0.92      0.92      0.92     87792
weighted avg       0.92      0.92      0.92     87792
```

ค่า Accuracy ที่ได้คือ 0.92 แปลว่าความแม่นยำที่ได้คือ 92%

ค่า precision, recall และ f1_score ก็ใช้ในการรายงานผลความแม่นยำด้วยเช่นกัน

ดูค่า f1_score

```
#see f1-score
f1 = f1_score(y_test, y_prediction)
print(f1)

Output:
0.9196786704825467
```

ดูค่า AUC

```
print("AUC score is: ", roc_auc_score(y_test, y_prediction))

Output:
AUC score is:  0.917255066315281
```

ตรวจสอบความแม่นยำของโมเดลด้วย confusion matrix

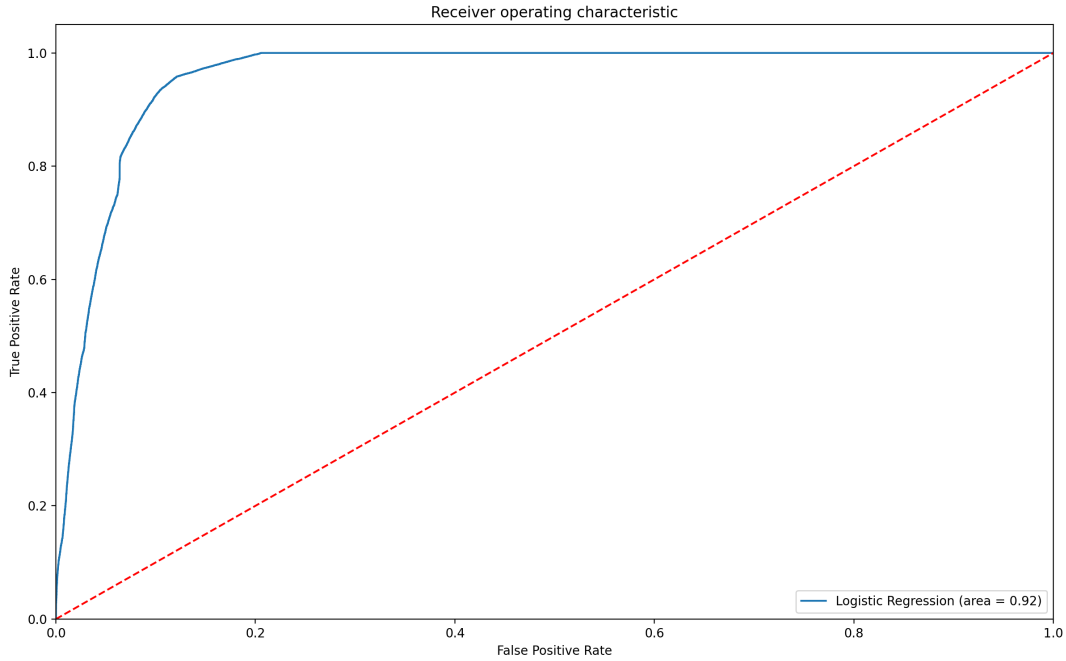
```
#CHECK CONFUSION MATRIX
confusion_matrix = confusion_matrix(y_test, y_prediction)
print(confusion_matrix)

Output:
[[38908  5060]
 [ 2209 41615]]
```

เราสามารถอ่านผลที่ได้ได้ดังนี้
TP = True Positive = 38908
FP = False Positive = 5060
FN =False Negative = 2209
TN = True Negative = 41615

หมายความว่าโมเดลของเราทำนายผลถูกต้อง 38908 + 41615 = 80523 ครั้ง
และทำนายผลผิดพลาด 5060 + 2209 = 7269 ครั้ง

สุดท้ายนี้เราจะ plot Receiver Operating Characteristic (ROC) curve



ROC Curve เป็นอีกเครื่องมือหนึ่งที่ใช้สำหรับตรวจสอบความแม่นยำของโมเดลที่มีข้อมูลแบบ binary classifiers โมเดลที่มีความแม่นยำสูง เส้น Logistic Regression จะอยู่ห่างจากเส้น ROC curve ให้มากที่สุด (อยู่มุมซ้ายบนของ chart)

Conclusion

เราได้ทำโมเดล Logistic Regression เพื่อทำนายว่าธุรกรรมนั้นเป็นธุรกรรมที่เป็นการฉ้อโกงหรือไม่ โมเดลที่สร้างมีความแม่นยำค่อนข้างสูง เราสามารถใช้โมเดลอื่นๆ ในการทำนายได้ เช่น k-Nearest Neighbors (KNN) , Decision Trees ,Support Vector Machine(SVM) และ Naive Bayes เป็นต้น ในอนาคต เราสนใจที่จะนำเอาโมเดลอื่นๆมาสร้างบนชุดข้อมูลนี้ เพื่อดูว่าโมเดลไหนให้ผลทำนายที่แม่นยำมากกว่ากัน
โมเดลนี้สามารถนำไปใช้แก้ปัญหาอื่นๆอีกได้ เช่น การทำนายว่าลูกค้าจะกลับมาซื้ออีกหรือไม่, การทำนายว่าธนาคารควรให้สินเชื่อกับผู้ขอสินเชื่อหรือไม่ เป็นต้น

Reference

Jordan, J. (2017). Evaluating a machine learning model. https://www.jeremyjordan.me/evaluating-a-machine-learning-model/

Brownlee, J. (2019). 14 Different Types of Learning in Machine Learning. https://machinelearningmastery.com/types-of-learning-in-machine-learning/

Lutkevich, B. (2020). association rules. https://searchbusinessanalytics.techtarget.com/definition/association-rules-in-data-mining

Patil, S., Nemade, V., Soni, P. (2018). Predictive Modelling For Credit Card Fraud Detection Using Data Analytics. https://pdf.sciencedirectassets.com/280203/1-s2.0-S1877050918X00088/1-s2.0-S1877050918309347/main.pdf

Sahin, Y., Duman, E. (2011). https://www.researchgate.net/publication/252016456_Detecting_credit_card_fraud_by_ANN_and_logistic_regression

Bala, R., Garg, D. (2019). Credit Card Fraud Detection using Logistic Regression and Bayesian Network https://www.ijirset.com/upload/2019/june/119_Credit.pdf

altexsoft (2020). Fraud Detection: How Machine Learning Systems Help Reveal Scams in Fintech, Healthcare, and eCommerce. https://www.altexsoft.com/whitepapers/fraud-detection-how-machine-learning-systems-help-reveal-scams-in-fintech-healthcare-and-ecommerce/

Knorr, E. (2015). How PayPal beats the bad guys with machine learning https://www.infoworld.com/article/2907877/how-paypal-reduces-fraud-with-machine-learning.html

Python Code

Source code ของรายงานนี้อยู่ที่ https://github.com/maria0911/thammasart-university

```
'''
Created on Feb 14, 2021

@author: maria
'''
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report, roc_auc_score,
roc_curve, f1_score
from sklearn.metrics import confusion_matrix
from imblearn.over_sampling import SMOTE


#set pandas to show all columns
pd.set_option("expand_frame_repr", False)

fraud = pd.read_csv('/Users/maria/Document/python/tuxsa/csv/
creditcard.csv')

origin_fraud = fraud.copy()
#DROP TIME COLUMN
fraud.drop('Time', axis=1, inplace=True)
print(fraud)
#INFO
info = fraud.info()

#CHECK TARGET COLUMN --> NOT BALANCE
sns.countplot(fraud['Class'])

#HISTAGRAM OF ALL COLUMN
for col in fraud.columns[:-1]:
    plt.title(f'Histogram of {col}')
    fraud[col].hist(bins=20)
    plt.show()


#CORRELATION USING
correlation_matrix = fraud.corr().round(2)
sns.heatmap(data=correlation_matrix, annot=True)
plt.show()

#CHECK FOR NULL VALUE
null = fraud.isnull().sum().sum()
print('Null value in dataset:', null)

#SCALE AMOUNT
```

```
scaler = StandardScaler()
fraud['NormalizedAmount'] =
scaler.fit_transform(fraud['Amount'].values.reshape(-1, 1))
print(fraud)

#for col in fraud.columns[:-1]:
#    plt.title(f'Boxplot of {col}')
#    plt.boxplot(fraud[col])
#    plt.show()

#SELECT OUTLIERS
outlier = fraud.columns.drop('Class')

#FUNCTION FOR REMOVING OUTLIER FROM BOXPLOT
def remove_outlier(data):
    Q1 = data.quantile(0.25)
    Q3 = data.quantile(0.75)
    IQR = Q3 - Q1

    clean = data[~((data < (Q1-1.5*IQR)) | (data >
(Q3+1.5*IQR))).any(axis=1)]

    return clean

#REMOVE ALL OUTLIERS USING FUNCTION, REPLACE WITH NAN VALUE
fraud[outlier] = remove_outlier(fraud[outlier])
print('\nReplace outliers with NaN value:\n',fraud)

fraud.dropna(inplace=True)
print('\nRemove row with NaN value:\n',fraud)

print('Shape of data with outliers: ', origin_fraud.shape)
print('Shape of data without outliers: ', fraud.shape)
print('Number of removed outliers: ', origin_fraud.shape[0] -
fraud.shape[0])

#BALANCE TARGET COLUMN
data = fraud.drop(['Amount'], axis = 1)
X = data.drop(['Class'], axis = 1)
y = data['Class']

#DATA VISAULIOSATION NOT BALANCE TARGET
#sns.countplot(y)
#plt.show()

smote = SMOTE(random_state=0)
X, y = smote.fit_resample(X, y)

#DISPLAY NEW TARGET COLUMN
#sns.countplot(y)
#plt.show()

#SPLIT DATA
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size =
0.3, random_state = 0)
```

```
#BUILD MODEL
model = LogisticRegression()
model.fit(X_train, y_train)

pred = model.predict(X_test)
print(pred)

report = classification_report(y_test,pred)
print(report)

f1 = f1_score(y_test, pred)
print(f1)

confusion_matrix = confusion_matrix(y_test, pred)
print(confusion_matrix)

logit_roc_auc = roc_auc_score(y_test, model.predict(X_test))
fpr, tpr, thresholds = roc_curve(y_test, model.predict_proba(X_test)
[:,1])
plt.figure()
plt.plot(fpr, tpr, label='Logistic Regression (area = %0.2f)' %
logit_roc_auc)
plt.plot([0, 1], [0, 1],'r--')
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Receiver operating characteristic')
plt.legend(loc="lower right")
plt.show()
```

ตัวอย่าง dataset

	V15	V16	V17	V18	V19	V20	V21	V22	V23	V24	V25	V26	V27	V28	Amount	Class
	1.468177	-0.470401	0.207971	0.025791	0.403993	0.251412	-0.018307	0.277838	-0.110474	0.066928	0.128539	-0.189115	0.133558	-0.021053	149.62	0
	0.635558	0.463917	-0.114805	-0.183361	-0.145783	-0.069083	-0.225775	-0.638672	0.101288	-0.339846	0.167170	0.125895	-0.008983	0.014724	2.69	0
	2.345865	-2.890083	1.109969	-0.121359	-2.261857	0.524980	0.247998	0.771679	0.909412	-0.689281	-0.327642	-0.139097	-0.055353	-0.059752	378.66	0
	0.631418	-1.059647	-0.684093	1.965775	-1.232622	-0.208038	-0.108300	0.005274	-0.190321	-1.175575	0.647376	-0.221929	0.062723	0.061458	123.50	0
	0.175121	-0.451449	-0.237033	-0.038195	0.803487	0.408542	-0.009431	0.798278	-0.137458	0.141267	-0.206010	0.502292	0.219422	0.215153	69.99	0

	V1	V2	V3	V4	V5	V6	V7	V8	V9	V10	V11	V12	V13	V14	V15	
0	-1.359807	-0.072781	2.536347	1.378155	-0.338321	0.462388	0.239599	0.098698	0.363787	0.090794	-0.551600	-0.617801	-0.991390	-0.311169	1.468177	-
1	1.191857	0.266151	0.166480	0.448154	0.060018	-0.082361	-0.078803	0.085102	-0.255425	-0.166974	1.612727	1.065235	0.489095	-0.143772	0.635558	
2	-1.358354	-1.340163	1.773209	0.379780	-0.503198	1.800499	0.791461	0.247676	-1.514654	0.207643	0.624501	0.066084	0.717293	-0.165946	2.345865	-
3	-0.966272	-0.185226	1.792993	-0.863291	-0.010309	1.247203	0.237609	0.377436	-1.387024	-0.054952	-0.226487	0.178228	0.507757	-0.287924	-0.631418	-
4	-1.158233	0.877737	1.548718	0.403034	-0.407193	0.095921	0.592941	-0.270533	0.817739	0.753074	-0.822843	0.538196	1.345852	-1.119670	0.175121	-