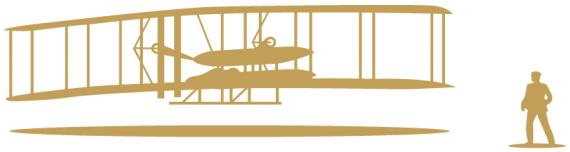

WRIGHT STATE
UNIVERSITY

Credit Card Fraud Detection Modeling:

A Machine Learned Analysis On Real Data

Spring 2024

by

Reza Amini
amini.4@wright.edu

Under the supervision of

Dr. Bin Li

Raj Soin College of Business
Wright State University

Contents

1 Introduction **1**
1.1 Fraud Detection Approaches 1
1.2 Data Insight and Fraud Detection 2

2 Dataset **3**

3 Analysis **4**
3.1 Data Preprocessing 4
3.2 Modeling and Metrics 7

4 Future work **8**

1 Introduction

Credit card fraud detection is paramount in safeguarding financial assets and maintaining consumer trust in the digital age. With the continuous rise in electronic transaction volumes, the likelihood of fraudulent activities exploiting payment system vulnerabilities also escalates. A report by the Nilson Report in 2021 revealed that global card-fraud losses amounted to US$28.65 billion, underscoring the urgent need for businesses to adopt comprehensive fraud detection and prevention strategies. The evolution towards online and card-not-present transactions has further complicated verification processes, significantly increasing the risk of fraud. This shift has resulted in a marked rise in fraud cases, with the cost to businesses for every dollar of fraud in US retail and e-commerce now reaching $3.75, a 20% increase from 2019. These figures highlight the critical role of effective fraud detection mechanisms in protecting revenue and ensuring a trustworthy customer experience [1].

Furthermore, the complexity and varieties of fraud have necessitated the development of advanced fraud detection methodologies. Modern credit card fraud detection systems heavily rely on digitization and automation, utilizing artificial intelligence (AI) and machine learning (ML) to analyze data and identify fraudulent transactions. These systems are designed to monitor anomalies and predict fraudulent patterns, thereby enabling real-time detection and prevention of unauthorized transactions. The adoption of such technologies is crucial in combating the increasingly sophisticated methods employed by fraudsters, particularly in the context of card-not-present fraud, which has seen a dramatic rise due to the proliferation of online shopping and the availability of stolen credit card information on the dark web. As such, credit card companies and financial institutions are investing in advanced technologies and security measures to mitigate the risks associated with credit card fraud, highlighting the importance of fraud detection in maintaining the integrity of payment systems and protecting consumers from financial harm[2].

1.1 Fraud Detection Approaches

The **Visa** credit card network implements a system known as Visa Advanced Authorization to combat fraudulent activities where individuals attempt to make purchases under someone else's identity. This system leverages artificial intelligence (AI) and machine learning techniques to sift through vast amounts of transaction data to evaluate risk levels.

Key risk indicators that Visa Advanced Authorization aims to pinpoint within a fraction of a second include:

- The nature of the transaction, distinguishing whether it's conducted online, through contactless means, within apps, or using chip or magnetic stripe technology.
- The compatibility of the transaction with the historical spending patterns of the cardholder.
- Any anomalies related to the transaction, such as it being executed at an unusual time or involving an unusually large sum of money.

  Upon the completion of data analysis, the system assigns a risk score, where a score of one signifies minimal risk and a score of 99 represents a high risk of fraud. This score is then communicated to the cardholder's bank for a final decision on whether the transaction should proceed.

Visa reports that this authorization mechanism has significantly enhanced the detection of suspicious transactions, contributing to a global fraud rate of under 0.1 percent—a figure that has decreased by two-thirds compared to twenty years ago, despite a more than tenfold increase in transaction volume over the same period [3].

[1] https://stripe.com/in/resources/more/credit-card-fraud-detection-and-prevention
[2] https://www.inscribe.ai/fraud-detection/credit-fraud-detection
[3] https://www.bankrate.com/finance/credit-cards/major-credit-card-networks-protect-against-fraud

Mastercard employs the Mastercard Identity Check, incorporating EMV 3-D Secure technology, to verify identities and combat credit card fraud, particularly in card-not-present transactions. This system enhances real-time transaction authentication by evaluating numerous factors through AI and machine learning, allowing for informed decisions on transaction approvals. It even considers user behavior, like screen brightness and gestures, alongside transaction history and insights from merchants and issuers. For added security, it may use biometrics or single-use passwords when necessary [4].

American Express operates as both a credit card network and issuer, offering payment services directly to cardholders and merchants without the intermediary role of third-party banks, unlike many other card issuers. This direct model grants American Express greater access to vital transaction data, enabling it to more effectively identify and prevent potentially fraudulent activities. The company's network oversees about $1.2 trillion in transactions annually, employing advanced fraud protection technologies, including machine learning, to analyze and assess risk factors in real-time for each transaction.

In its fight against fraud, American Express adopts a comprehensive, multi-layered strategy. This includes the utilization of EMV chips in its chip and PIN cards, which generate a unique, encrypted code for each transaction to instantly authenticate account information. This technology significantly reduces the risk of counterfeit fraud, as it ensures that every transaction generates a distinct code, making it virtually impossible for thieves to create fraudulent copies of the card [5].

1.2 Data Insight and Fraud Detection

Data is crucial for accurately pinpointing fraudulent activities. However, without advanced analytics such as machine learning, deciphering fraud patterns from vast datasets can be challenging. Credit card companies employ fraud detection models powered by machine learning to conduct real-time monitoring and swiftly generate fraud assessments in milliseconds with each card use globally.

Machine learning is significantly enhancing the way credit card fraud is detected and prevented by efficiently analyzing massive datasets to uncover patterns and irregularities indicative of fraudulent activities. This subset of artificial intelligence (AI) empowers computer systems to learn from data, make informed decisions, and identify patterns that humans might overlook. It operates on various models, including supervised, unsupervised, and reinforcement learning, each catering to different aspects of fraud detection [6]

In the practical application of combating payment fraud, machine learning algorithms are pivotal. They undertake real-time analysis of transaction data to discern legitimate activities from suspicious ones, employing techniques like anomaly detection, risk scoring, and network analysis. For example, by evaluating transaction variables such as time, location, and frequency, machine learning can flag transactions that deviate from a customer's normal behavior, signaling potential fraud. This advanced technology also adapts and evolves, continuously improving its detection capabilities to stay ahead of fraudsters' changing tactics [7]

[4] https://www.bankrate.com/finance/credit-cards/major-credit-card-networks-protect-against-fraud
[5] https://www.americanexpress.com/en-us/credit-cards/credit-intel/fraud-alerts/
[6] https://stripe.com/resources/more/how-machine-learning-works-for-payment-fraud-detection-and-prevention
[7] https://www.paypal.com/us/brc/article/payment-fraud-detection-machine-learning

2 Dataset

In our study, we utilize a dataset thatpublic on Kaggle. This dataset collected credit card transactions from European cardholders during September 2013. [1][2][3][6][5][4][7][8][9][10]
The dataset comes from a research collaboration of Worldline and the Machine Learning Group (http://mlg.ulb.ac.be) of ULB (Université Libre de Bruxelles) related to detecting fraud by big data mining.

It includes data from transactions over a period of two days, with 492 of the 284,807 transactions being identified as fraud. The dataset is significantly skewed, with fraudulent transactions (the positive class) making up only 0.172% of the total. [8] On the other hand, with this small proportion of the fraudulent transaction, a total amount of 60,127.97 dollars can be saved depending on the performance of our model.

The data features are numerical, derived from a Principal Component Analysis (PCA) transformation. Due to privacy concerns, the original data attributes and detailed background information are removed and data are anonymized to preserve the privacy concerns. The principal components from the PCA are labeled V1 through V28. The exceptions to PCA transformation are the 'Time' and 'Amount' features. 'Time' represents the seconds that have elapsed between each transaction and the dataset's initial transaction. The 'Amount' indicates the value of the transaction, which is useful for cost-sensitive learning that varies by example. The 'Class' feature serves as the dependent variable, indicating a fraud with a value of 1 and non-fraudulent transactions with a value of 0.

[8] https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud

3 Analysis

In this section, we will delve into all the phases I undertook to accomplish this project, which include data processing, pattern modeling, and model evaluation. Let's explore these steps in detail.

3.1 Data Preprocessing

Every data analysis project begins with extensive efforts in data preparation and comprehension. Similarly, for this project, I devoted approximately 40% of my time to refining the data, ensuring it was clean and structured in a way that would be useful for the model.

First, we start to read the dataset and gain initial information about it into a Python data frame. The features V1 through V28 are anonymized using PCA. However, 'Time' and 'Amount' are the only features not subjected to PCA.

- 'Time' represents the time between transactions.
- 'Amount' represents the transaction amount.
- 'Class' represents the class or the label for our data assuming a value of 1 for fraud and 0 otherwise.

```
In [13]: df = pandas.read_csv('creditcard.csv', sep=',')
         df.head()
```

Out[13]:

	Time	V1	V2	V3	V4	V5	V6	V7	V8	V9	...	V21	V22	V23	V24	
0	0.0	-1.359807	-0.072781	2.536347	1.378155	-0.338321	0.462388	0.239599	0.098698	0.363787	...	-0.018307	0.277838	-0.110474	0.066928	0.12
1	0.0	1.191857	0.266151	0.166480	0.448154	0.060018	-0.082361	-0.078803	0.085102	-0.255425	...	-0.225775	-0.638672	0.101288	-0.339846	0.16
2	1.0	-1.358354	-1.340163	1.773209	0.379780	-0.503198	1.800499	0.791461	0.247676	-1.514654	...	0.247998	0.771679	0.909412	-0.689281	-0.32
3	1.0	-0.966272	-0.185226	1.792993	-0.863291	-0.010309	1.247203	0.237609	0.377436	-1.387024	...	-0.108300	0.005274	-0.190321	-1.175575	0.64
4	2.0	-1.158233	0.877737	1.548718	0.403034	-0.407193	0.095921	0.592941	-0.270533	0.817739	...	-0.009431	0.798278	-0.137458	0.141267	-0.20

5 rows × 31 columns

Figure 1: Data frame general information.

Figure 2 reflects more insight from the dataset and shows the type value of each column. As it is shown, the dataset is all numeric data, and for modeling, we don't need to do any data transformation and type change.

```
In [10]: df.info()
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 284807 entries, 0 to 284806
Data columns (total 31 columns):
Time      284807 non-null float64
V1        284807 non-null float64
V2        284807 non-null float64
V3        284807 non-null float64
V4        284807 non-null float64
V5        284807 non-null float64
V6        284807 non-null float64
V7        284807 non-null float64
V8        284807 non-null float64
V9        284807 non-null float64
V10       284807 non-null float64
V11       284807 non-null float64
V12       284807 non-null float64
V13       284807 non-null float64
V14       284807 non-null float64
V15       284807 non-null float64
V16       284807 non-null float64
V17       284807 non-null float64
V18       284807 non-null float64
V19       284807 non-null float64
V20       284807 non-null float64
V21       284807 non-null float64
V22       284807 non-null float64
V23       284807 non-null float64
V24       284807 non-null float64
V25       284807 non-null float64
V26       284807 non-null float64
V27       284807 non-null float64
V28       284807 non-null float64
Amount    284807 non-null float64
Class     284807 non-null int64
dtypes: float64(30), int64(1)
memory usage: 67.4 MB
```

Figure 2: Caption for the centered image.

```
In [14]: df['Class'].value_counts()

Out[14]: 0    284315
         1       492
         Name: Class, dtype: int64

In [30]: plt.figure(figsize=(8, 8))
         plt.pie(class_counts, labels=class_counts.index, autopct='%1.1f%%', startangle=140)
         plt.title('Class Representation with Counts')
         plt.legend(class_counts.index, title="Classes", loc="center left", bbox_to_anchor=(1, 0, 0.5, 1))

         # Display the plot
         plt.show()
```

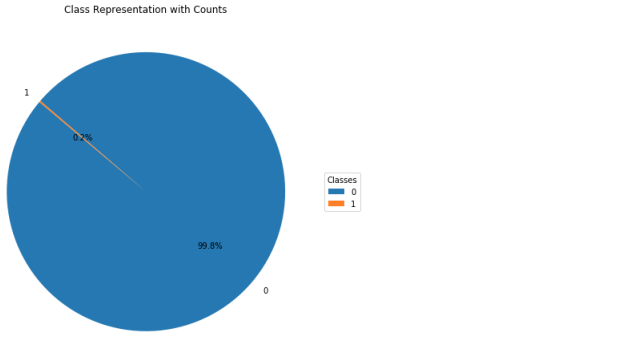

Figure 3: Class distribution representation

Class imbalance is indeed a significant concern among data analysts and data scientists, especially when dealing with classification problems. It occurs when the classes in the dataset are not represented equally, meaning one or more classes have significantly fewer instances than the others. This imbalance can lead to biased or inaccurate models, as the algorithms tend to favor the majority class over the minority ones. As we also see in Figure 3, we are encountering a class imbalanced issue as the fraud class (class:1) is only taking 0.17% of the whole dataset while this class is the main target of this research for detecting the fraudulent transaction by this class.

Battling class-imbalanced data is one of the usual tasks in data analysis and modeling world. For this challenge, there are different known approaches introduced from which we can name a few:

- Resampling: In this approach, the minority class would be oversampled or the majority class will be under-sampled. This depends on the purposed of the work, the importance of each class for analysis, the count of each sample and many other factors.
- Synthetic data generation: In this approach the undersampled class of data will be reproduced in a synthetic way using available libraries.
- Weight assigned in modeling: This approach takes the dataset as is and the modeling algorithm will give more weight to the under-sampled class.

In this analysis, I decided to go with the resampling approach since I had previously successful experience in taking the same approach in mitigating the class imbalanced data.

```
In [42]: df_0 = df[df['Class'] == 0].sample(n=492, random_state=42)
         df_1= df[df['Class'] == 1].sample(n=492, random_state=42)
         df_concat=pandas.concat([df_0,df_1], ignore_index=True).sample(frac=1)
         df_concat
```

Out[42]:

	Time	V1	V2	V3	V4	V5	V6	V7	V8	V9	...	V21	V22	V23	V24	V25	V26	V27	V28	Amount	Class
182	141,893.00	1.98	0.67	-1.42	3.67	1.09	-0.04	0.65	-0.24	-1.14	...	-0.06	-0.06	-0.00	-0.79	0.31	0.07	-0.06	-0.07	15.72	0
207	100,876.00	-3.74	-5.83	1.41	1.24	4.80	-4.44	-4.19	0.51	2.76	...	0.64	0.48	1.64	0.32	-2.36	0.06	0.13	0.49	15.95	0
171	25,899.00	1.04	-0.92	0.15	-0.19	-1.29	-1.30	0.05	-0.41	-0.94	...	-0.31	-0.63	-0.09	0.99	0.32	1.04	-0.10	0.03	174.98	0
115	33,277.00	0.49	-1.77	0.96	0.98	-1.52	0.89	-0.65	0.24	1.93	...	0.06	-0.15	-0.48	-0.34	0.33	0.73	-0.03	0.08	374.00	0
469	59,640.00	-1.39	1.57	1.23	-0.30	-0.39	-0.80	0.47	0.18	0.28	...	-0.30	-0.63	0.05	0.47	-0.08	0.06	0.66	0.38	3.56	0
506	150,139.00	-6.68	-2.71	-5.77	1.45	-0.66	-1.15	0.85	0.43	-1.32	...	0.22	1.19	0.34	0.22	0.80	0.04	-0.05	0.08	237.26	1
93	111,137.00	2.10	0.03	-1.44	0.33	0.35	-0.73	0.02	-0.32	2.16	...	0.09	0.62	-0.00	0.42	0.34	-0.47	-0.03	-0.06	1.00	0
129	114,098.00	-0.96	0.65	2.35	-0.97	1.46	0.69	1.68	-1.48	1.29	...	-0.43	-0.39	-0.45	0.03	0.26	-0.67	-1.92	-1.11	4.99	0
381	137,175.00	1.75	-0.09	-2.47	0.41	1.43	1.10	0.01	0.34	0.55	...	-0.02	0.15	0.07	-0.88	-0.02	-0.23	0.05	-0.02	73.27	0
413	75,286.00	1.26	-0.12	-1.73	0.33	0.66	-1.18	1.18	-0.73	-2.03	...	-0.04	0.22	-0.65	-0.38	1.57	0.19	-0.06	0.00	138.43	0

Figure 4: Oversampling in the minority class

```
In [34]: x=df.drop(columns='Class', axis=1)
         y=df['Class']
```

Figure 5: Data being divided into X and Y for classification

In order to work with any classification model, data needs to be chunked into X and Y. In this setup, X are the features or columns that are being used to be trained in a model to finally predict Y. Since we are also going to do classification modeling in this study, I will divide the data into X and Y as below:

```
In [38]: scaler=StandardScaler()
         x=scaler.fit_transform(x_notNormed)
```

Figure 6: Applying standard scaler to data

```
In [53]: x_train, x_test, y_train, y_test=train_test_split(x,y, test_size=0.10, random_state=123)
         print(f'x_train{x_train.shape}\n, x_test{x_test.shape}\n, y_train{y_train.shape}\n, y_test{y_test.shape}')

         x_train(885, 30)
         , x_test(99, 30)
         , y_train(885,)
         , y_test(99,)
```

Figure 7: Slicing the data into train and test

Another step for preparing the dataset for modeling is to scale the dataset and values in each column. The approach I used here is a standard scaler that is included in scikit-learn package. [9] We need to standardize the dataset to increase the efficiency of the classification model.

3.2 Modeling and Metrics

Our goal is to correctly identify the class:1 rows in this dataset in order to build a reliable model for fraud detection problems. After exploring and working with various models, I decided to start with the less complex model and popular baseline mode since the dataset is small after resampling and the features are limited. I selected the linear regression model which is the fundamental model in statistics and machine learning. This model is also less prone to overfitting which is a common issue with smaller datasets.

First step in every modeling pipeline is to slide the dataset into train and test set. It is a common approach to take 80% of the data fro training and 20% for testing. However, because of the limitation in our datapoint which is due to the undersampling of the imbalanced data, I decided to consider more data for training which is 90% of my dataset and 10% for evaluating the model. Fig 7.

I used the logistic regression model implemented by the Scikit Learn Python library: This model has the base implementation of the logistic regression algorithm and I fit the train and test data into it as Fig 8. First I create an instance of the LogisticRegression class defined in scikit learn and then I fit the training data to it. Model takes the features from x_train and it takes the classes or labels from y_train and starts training. Then we ask the model to give us its predicted labels with the .predict() function. By taking the predicted labels with our model we can now evaluate how accurate is the predicted classes to real labels. As we also kept 10% of our data for final evaluation, now we use it to get the overall report of the model.

```
y_train_pred=lr.predict(x_train)
y_train_cl_report=classification_report(y_train, y_train_pred, target_names = ['No
    ␣Fraud', 'Fraud'])
```

For evaluation purposes, there are metrics that we should check:

- Precision: Percentage of correct detection of positive class all predicted positive class. In this case, we can consider how many times the model detects fraud transactions correctly among all the fraud classes that it detects.
- Recall: Percentage of correct detection of positive class all real positive class the exists. In this case, we can consider how many times the model detects fraud transactions correctly among all fraud activities that exist.
- F1score: it is a harmonic mean of precision and recall.

For this dataset and goal, it is important to consider both precision and recall as important factors. Precision is important to avoid labeling normal transactions as fraud and recall is crucial because the more

[9] https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html

```
In [54]: def logic_regression(x_train, y_train, x_test):
             lr=LogisticRegression()
             lr.fit(x_train, y_train)
             y_train_pred=lr.predict(x_train)
             y_train_cl_report=classification_report(y_train, y_train_pred, target_names = ['No Fraud', 'Fraud'])
             print("_"*100)
             print("TRAIN MODEL CLASSIFICATION REPORT")
             print("_"*100)
             print(y_train_cl_report)
             y_test_pred=lr.predict(x_test)
             y_test_cl_report=classification_report(y_test, y_test_pred, target_names = ['No Fraud', 'Fraud'])
             print("_"*100)
             print("TEST MODEL CLASSIFICATION REPORT")
             print("_"*100)
             print(y_test_cl_report)
             print("_"*100)
             return y_test_pred, lr

         y_test_pred, lr= logic_regression(x_train, y_train, x_test)
```

```
____________________________________________________________________________________________________
TRAIN MODEL CLASSIFICATION REPORT
____________________________________________________________________________________________________
              precision    recall  f1-score   support

    No Fraud       0.92      0.99      0.95       443
       Fraud       0.99      0.91      0.95       442

   micro avg       0.95      0.95      0.95       885
   macro avg       0.95      0.95      0.95       885
weighted avg       0.95      0.95      0.95       885

____________________________________________________________________________________________________
TEST MODEL CLASSIFICATION REPORT
____________________________________________________________________________________________________
              precision    recall  f1-score   support

    No Fraud       0.92      0.94      0.93        49
       Fraud       0.94      0.92      0.93        50

   micro avg       0.93      0.93      0.93        99
   macro avg       0.93      0.93      0.93        99
weighted avg       0.93      0.93      0.93        99

____________________________________________________________________________________________________
```

Figure 8: Logistic Regression implementation in Python

the model detects the suspisous transaction the more money it can save and more trust from the customers it can get.

For evaluating this model, we can look into the report generated by code, (check Fig 8). On the training part, the model is showing 98% f1-Score on both classes of fraudulent activities and non-fraudulent activities. By looking into the precision and recall column for class:Fraud, the model is 99% correct to identify which transaction is Fraud; however, the recall of 91% shows that the model can only discover 91% of the Fradu activities among all activities. On our small test dataset, the model is 94% precise in detecting the Fraud transaction correctly while among all Fraud transactions, only 92% of them are discovered by the model. Other metrics such as micro, micro and weighted average are useful when we want to feed the imbalanced data into the model. However, in this case, we balanced the data by using the undersampling method before doing any training, and because of that these metrics are the same among both classes.

4 Future work

For future learning purposes, I like to try these few experiments with this project:

- Trying other resampling approaches such as over-sampling of the minority class in order to have a larger dataset.
- Managing the imbalanced data issue with weighting the classes in the modeling phase and using the original dataset.
- Implementing other models like Neural Networks and Random Forests with imbalanced data.

References

[1] Fabrizio Carcillo, Andrea Dal Pozzolo, Yann-Aël Le Borgne, Olivier Caelen, Yannis Mazzer, and Gianluca Bontempi. Scarff: a scalable framework for streaming credit card fraud detection with spark. *Information fusion*, 41:182–194, 2018.

[2] Fabrizio Carcillo, Yann-Aël Le Borgne, Olivier Caelen, and Gianluca Bontempi. Streaming active learning strategies for real-life credit card fraud detection: assessment and visualization. *International Journal of Data Science and Analytics*, 5:285–300, 2018.

[3] Fabrizio Carcillo, Yann-Aël Le Borgne, Olivier Caelen, Yacine Kessaci, Frédéric Oblé, and Gianluca Bontempi. Combining unsupervised and supervised learning in credit card fraud detection. *Information sciences*, 557:317–331, 2021.

[4] Andrea Dal Pozzolo, Giacomo Boracchi, Olivier Caelen, Cesare Alippi, and Gianluca Bontempi. Credit card fraud detection: a realistic modeling and a novel learning strategy. *IEEE transactions on neural networks and learning systems*, 29(8):3784–3797, 2017.

[5] Andrea Dal Pozzolo, Olivier Caelen, Reid A Johnson, and Gianluca Bontempi. Calibrating probability with undersampling for unbalanced classification. In *2015 IEEE symposium series on computational intelligence*, pages 159–166. IEEE, 2015.

[6] Andrea Dal Pozzolo, Olivier Caelen, Yann-Ael Le Borgne, Serge Waterschoot, and Gianluca Bontempi. Learned lessons in credit card fraud detection from a practitioner perspective. *Expert systems with applications*, 41(10):4915–4928, 2014.

[7] Yann-A Le Borgne and Gianluca Bontempi. Machine learning for credit card fraud detection-practical handbook. *ACM SIGKDD explorations newsletter*, 6(1):1–6, 2004.

[8] Bertrand Lebichot, Yann-Aël Le Borgne, Liyun He-Guelton, Frederic Oblé, and Gianluca Bontempi. Deep-learning domain adaptation techniques for credit cards fraud detection. In *Recent Advances in Big Data and Deep Learning: Proceedings of the INNS Big Data and Deep Learning Conference INNS-BDDL2019, held at Sestri Levante, Genova, Italy 16-18 April 2019*, pages 78–88. Springer, 2020.

[9] Bertrand Lebichot, Gian Marco Paldino, Wissam Siblini, Liyun He-Guelton, Frédéric Oblé, and Gianluca Bontempi. Incremental learning strategies for credit cards fraud detection. *International Journal of Data Science and Analytics*, 12(2):165–174, 2021.

[10] Aaron Rosenbaum. Detecting credit card fraud with machine learning.