# ADDRESSING ANOMALY DETECTION IN HIGH VOLUME SETTING

MUHAMMAD ARSLAN

MS (DATA SCIENCE)

*A thesis submitted in partial fulfillment of the requirements for the degree of Master of Science*

DEPARTMENT OF COMPUTER SCIENCE

NATIONAL UNIVERSITY OF COMPUTER AND EMERGING SCIENCES

2020

.

# ADDRESSING ANOMALY DETECTION IN HIGH VOLUME SETTING

.

Submitted by: Muhammad Arslan (17k-3004)

Colloquium: 01.08.2017 - Fall 2017

Supervisor: Dr. Tahir Q. Syed

Signature: _____

Co-Supervisor: Behroz Mirza

Signature: _____

Department of Computer Science
National University of Computer and Emerging Sciences
Karachi Campus

Date: _____

# Declaration of Authorship

I, Muhammad Arslan, declare that this thesis titled, “Addressing anomaly detection in high volume setting” and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.
- This is a true copy of my report, including any final revisions, as approved by my supervisor and the thesis committee, and that this report has not been submitted for a higher degree to any other University or Institution.

Signed:

Date:

# Plagiarism Undertaking

I solemnly declare that research work presented in the thesis titled "Addressing anomaly detection in high volume setting" is solely my research work with no significant contribution from any other person. Small contribution/help wherever taken has been duly acknowledged and that complete thesis has been written by me.

I understand the zero tolerance policy of the HEC and University "National University of Computer and Emerging Sciences, Karachi" towards plagiarism. Therefore, I as an Author of the above titled thesis declare that no portion of my thesis has been plagiarized and any material used as reference is properly referred/cited.

I undertake that if I am found guilty of any formal plagiarism in the above titled thesis even after award of Masters degree, the University reserves the rights to withdraw/revoke my Masters degree and that HEC and the University has the right to publish my name on the HEC/University Website on which names of students are placed who submitted plagiarized thesis.

Author Signature:

Name:

# Dedication

This study is wholeheartedly dedicated to our beloved parents, who have been our source of inspiration and gave us strength when we thought of giving up, who continually provide their moral, spiritual, emotional, and financial support. To our brothers, sisters, relatives, mentor, friends, and classmates who shared their words of advice and encouragement to finish this study. And lastly, we dedicated this study to the Almighty Allah, thank you for the guidance, strength, power of mind, protection and skills and for giving us a healthy life. All of these, we offer to You.

# *Abstract*

In Anti-money laundering systems, suspicious transactions are the ones that have a probability of being related with money laundering. Pakistan's AML Act 2010 requires the financial institutions to detect and report suspicious transactions at the earliest. While working with AML detection, two biggest data related challenges faced are: (1). class imbalance because the normal population is too high as compared to the few rare events of fraud/laundering thus leading to highly skewed data distribution; and of (2). high volume and high velocity as huge amount of data produced by the customers' transactions and other activities with the bank. To overcome the first challenge of class imbalance, a two-step technique is formulated namely deep balancer model comprising of deep non-linear generative models which generates samples by learning joint distribution. A balanced training set is achieved by generating and reducing the minority and majority class samples respectively.To combat the second challenge of high volume data, this work optimizes in-memory utilization for high volume datasets by proposing flexible weight ties namely Cramer Rao Lower Bound Optimization. The proposed model has the capacity to retain information under high volume constraint using optimum memory consumption. This is achieved by ensuring presence of the previous and current batch in memory for complete parameter update cycle. This is in distinction to other techniques which either requires complete dataset to be loaded into the memory. The technique was validated on four different anomaly datasets and also compared with the existing prevailing techniques. The proposed technique improves accuracy as compared to the existing prevailing techniques on an average of 2% in a 5 batch split, and in case of 20 batches where the distribution across batches change drastically, the proposed technique improved accuracy on an average on 30% which makes this technique novel that in strict memory constraints, this technique can be scaled out to large number of batches which will result in improved accuracy and information retention. The main contribution of this work is that it combats huge volume in on premise settings as in financial industry regulators do not allow data of any individual to be placed on the cloud as cloud security has been a major concern. Secondly, it retains the information in high volume data via memory optimization which makes it cost effective. Thirdly, in case of high changing distributions in either volume or velocity; the information retention and classification accuracy is better than the existing prevailing techniques.

**Keywords:** *data governance, anomaly detection, high volume, information retention, cramer rao bound*

# *Acknowledgements*

It is with immense gratitude that I acknowledge the support and help of my thesis supervisor Dr. Tahir Qasim Syed and Co-Supervisor Sir Behroz Mirza who believed in me and were continuously supporting me in every problem that occurred during this research. They consistently allowed this thesis to be my own work, but steered me in the right direction whenever they thought I needed it. Our cooperation was truly an inspiring experience. Very special thanks to Sir Ali Muhammad Khan Dehlavi for his eagerness and great assistance; his comments were of critical importance.

I would also like to acknowledge my colleagues; Mr. Karim Jivani, Muhammad Ahmad, Ahmed Rao, Furqan Hashim, and Mr. Behram, Without their passionate participation and input, the tasks have not been successfully conducted, and I am gratefully indebted to them for their valuable comments on this thesis.

Finally, I must express my very profound gratitude to my parents for providing me with unfailing support and continuous encouragement throughout my years of study and through the process of researching and writing this thesis. This accomplishment would not have been possible without them, especially my dad Mr. Muhammad Idrees, who's always been supportive, patient and encouraging....

# Contents

# List of Figures

# List of Tables

# Key terms and Abbreviations

Table 1: Key terms and Abbreviations.

| № | Term / Abbreviation | Description |
|---|---|---|
| 1 | TMS | Transaction monitoring system. |
| 2 | FCCM | Transactions monitoring system used by Bank X. The full name of this Oracle product is Financial Crime and Compliance Management. |
| 3 | ML&TF | Money Laundering and Terrorism Financing. |
| 4 | Effective Risk | Group of risks aggregated in FCCM corresponding to the riskiness of the alerting entity. |
| 5 | Activity Risk | Group of risks aggregated in FCCM corresponding to the risks related to the transactions, e.g. product, channel or transaction party risk. |
| 6 | KYC | Know Your Customer. |
| 7 | Customer Risk Rating | Result of profiling performed in separate FCCM KYC module stored in Misys. |
| 8 | False positive alerts | An alert that did not exhibit any suspicion from an investigatory perspective. The number of such alerts should be minimised in the optimized efficient transaction monitoring system. |
| 9 | Worthy alert (WA) | An alert that is considered so suspicious that it requires either a substantial investigative effort, or additional enquiries. In general, it could be considered as alert for which a case was opened, if a case manager is used. |
| 10 | SARs | Suspicious Activity Reports. |

# Chapter 1

# Introduction

Suspicious transactions are the ones that have a probability of being related with money laundering. Pakistan's AML Act 2010 requires the financial institutions to detect and report suspicious transactions at the earliest. In AML detection, the biggest challenges are of the class imbalance, it's just same as like searching for a needle in the haystack; and of high volume and high velocity transactional data logging into the systems on daily basis.

Organizations are devoting a large budget in big data projects but success proportion is low. One of the major reason being the high volume pressing challenge of big data and its mitigation has been a up hill task. Currently distributed cluster computing variants are used to counter high volume using data parallelism technique. These include:

- Cloud-based Mechanism: These offer compute facilities on cloud with data set to be hosted on the provider's cloud. The pitfall of this approach is a number of regulators do not allow any transactions data hosting on the cloud. Second issue is the security of the cloud based model has been a concern
- On-premises Mechanism: These implementations provide machine learning frameworks with distributed computing capabilities. The pitfalls include: firstly high native dependency where the learning is restricted to the algorithm API exposed by the framework. Secondly, these have high memory requirements as the speed is offset by the large amount of memory required to load complete dataset in one interval.

This works proposes an incremental learning suitable for high volume datasets.The technique stems from the idea of knowledge retention by using posterior probabilities. Requiring the presence of only the current and previous batch for an effective computation cycle the model has two fold effect. First it eliminates data storage requirement which is mandatory for techniques working with large data volume.Second it is memory efficient as the presence of only two batches in the memory makes the process light weight.

## 1.1 AML Background

Pakistan's AML Act 2010 requires the financial institutions to detect and report suspicious transactions at the earliest. Fraud and money laundering sequence lead to granting validity on illegal assets mostly accomplished by a series of transactions executed via a valid financial system. Thus to curb this menace a timely detection of these doubtful transactions is mandatory. However most of the current implementations use a rule based approach to identify transactions which results in increasingly huge number of false alerts thereby increasing the number of man hours for investigating the alerts per scenario.

Rule based systems are not able to detect dynamic patterns in the data and are not capturing the changing behavior of the money launderers. As rules and threshold values are adjusted based on the human directives, a possible leak leads to compromising the complete process and also will result in producing high false positive rates. This happens as rule sets work with fixed values and cannot take into account nonlinear patterns. Manual configuration and updating of rule based systems slow leading to time loss and are also unable to combat adaptation capability of the offenders.

On the contrary, data driven systems are able to identify the non-linear patterns and detect the dynamic patterns in the data which is not possible using rule based approach and are therefore able to reduce the false positive rates and precise and timely identification of suspicious transactions and thus to enforcement of an effective AML policy

In AML detection, transactional activity, [15] peaks and troughs do not help much in detecting the true positive instance of money laundering. This is just an attempt to use such data that differ greatly from the norm. However, as the head of ML analysts always maintained: "good money launderers try to look like the norm". This complicates matters considerably as 'outlier detection' techniques are not as useful as they are in fraud; profiling is conducted under conditions of considerable ambiguity.

In today's era, banks need to have anti-money laundering systems, controls and practices to mitigate the risk of money laundering and to save themselves from huge fines imposed by the central bank.

This research is carried out using the bank x data. The said bank x has already implemented a rule based Transaction Monitoring System offered by oracle called Financial crime and compliance module (FCCM) also known as Mantas, which detects the money laundering patterns using the hardcoded SQL queries in them and then certain thresholds were applied to generate the suspicious alerts which are further investigated by the AML analyst and are reported to the Financial Monitoring Unit (FMU). The current state of the art TMS produce a huge number of alerts and false positives that it always remains a challenge for the AML team to work as a result much of the human effort and time is wasted.

We analyzed the alerts and SARs statistics for each scenario and selected a particular scenario which was generating the largest volume of alerts for the certain year and has the biggest scenario share among all. The prime objective of this scenario is to apply enhanced scrutiny to transactions involving high-risk entities, as such activity that may subject the institution to a greater risk of money laundering or fraud. Any account, customer, correspondent bank, or external entity found on a watch list is considered to be a high-risk entity. This scenario monitors transactions to and from high-risk entities during a specified Look-back Period. Customers are segmented based on their transaction behaviors and different thresholds are applied for each of these rules based on each segment of customers.

The rest of the paper is organized as follows. Section 2 describes the detailed literature review for different focuses targeted in this paper. Section 3 entails the most important part of AML pre-processing phase where complete dataset preparation is also discussed. Sect. 4 focuses on methodology for handling the first challenge of class imbalance and in Sect. 5, the second challenge of huge volume based model training is encountered and we conclude our paper with a discussion in Sect. 6 followed by Conclusion in Sect. 7.

# Chapter 2

# Literature Review

The literature is sub-divided into Six major parts i.e. Anti-money Laundering Systems, Sampling techniques for handling class imbalance, Online Learning, Frameworks for Data Reduction, Big Data Frameworks and Open source libraries focused on overcoming the most important challenge of Big Data i.e. high volume and velocity.

## 2.1 Anti-Money Laundering Systems

It is essential to develop the detection techniques for capturing anomalous transactions as money laundering has become a serious crime. This problem is still not properly addressed after numerous researches have been carried out. Many money laundering detection techniques are introduced nowadays which includes [13] AML typologies using fuzzy rules, frequent rules, SVM, RBF, clustering and auto encoders which has the ability to detect and restrain the previously occurred similar money laundering cases as a result the number of false positives will automatically be reduced by restraining the duplicate alerts. Another way of detecting such transactions is link analysis which involves clustering and (FP Growth, FP Close  FP Max) and also social network analysis so as to identify the integrator through the graph analysis. Link analysis helps us identify the relationship between the bank accounts and the links constructed among transactions to expose the particular group as a whole. Behavioral modeling identifies the anomaly on the basis of the transactional behavior of the bank customers. There are various behavioral modeling techniques to extract information and group similar data including Expectation-maximization algorithm (EM) clustering and empirical mode decomposition. Another technique for reducing the false positives is risk scoring of alerts meaning assigning risk to all the transactions or customers based on statistical models and business rules. Decision tree is one them which splits rules used in the determining the transaction as a suspicious or normal. In [30], the author discusses the framework for reducing the efforts and time spent on the investigation of the alerts generated by the rule based system. They propose a novel AML Framework follows a distributed architecture to integrate different deep learning-based NLP modules to facilitate investigations of suspicious ML transactions. The proposed distributed framework performs news and tweet sentiment analysis, entity recognition, relation extraction, entity linking and link analysis on different data sources (e.g. news articles and tweets) to provide additional evidence to human investigators for final decision making. Each NLP module is evaluated on a task-specific data set, and the overall experiments are performed on synthetic

and real-world datasets. Feedback from AML practitioners suggests that our system can reduce approximately 30% time and cost compared to their previous manual approaches of AML investigation.

Another similar study [62] on Reserve Bank of India (RBI) for reducing the false positives introduces data mining based Anti-Money Laundering technique which can be able to detect the Rapid movement of funds using hash based association technique and also the integrator and agent who attempt to hide the origin in the layering stage of the money laundering. Similarly, to detect the malicious Online Social Network accounts that engage in laundering virtual currency [68], they devise multi-faceted features that characterize accounts from three aspects: account viability, transaction sequences, and spatial correlation among accounts. Finally, we propose a detection method by integrating these features using a statistical classifier. [33] Neuro-Fuzzy, ANFIS - (A fuzzy inference system is created using subtractive clustering method) to detect the riskiness of the user behavior in terms of anti-money laundering (Suspicious accounts) and [3] Combined Benford's Law and machine learning algorithms (logistic regression, decision trees, neural networks, and random forests) to find the patterns of money laundering. In [54], a hybrid model of distance based clustering and dynamic Bayesian networks, called SARDBN (Suspicious Activity Reporting using Dynamic Bayesian Network) is used to capture patterns in a customer's monthly transactional sequences to identify anomalies in context of money laundering, [69, 64], classification using decision tree and tools for outlier analysis were proposed in finding suspicious activity. In [45, 22, 23], i.e. tools for Cluster analysis, Bayesian classification, Genetic algorithms, support vector machine (SVM), intelligent agents and neural network were discussed. Among others, data mining methods are also used to detect suspicious activity. They focus on statistical analyses, discovering customer behavior and patterns to detect crimes [53]. Underlying methods for rule based learning mentioned in [19] can help identify suspicious activities from huge database of customer transactions in terms of money laundering.

## 2.2 Sampling Techniques

The Sampling techniques for handling class imbalance are categorized in sections as follows: Generative, synthetic and cluster based techniques for oversampling are discussed and followed by resampling, under sampling and hybrid techniques.

### 2.2.1 Generative oversampling

Generative models learn the data distribution and are then able to generate the minority class instances close to the actual minority instances. A well-known approach in [16] uses conditional Generative Adversarial Networks (cGAN) for generative oversampling. The approach is evaluated on 12 datasets available publically on ML Repository by generating minority class instances. In [67], Variational auto encoders are used for generating minority class images and are evaluated on high dimensional datasets along with performance comparison with the traditional approaches. Another approach [70] learns intermediate representation using Restricted Boltzmann machine which generates samples very near to the true distribution by transforming synthetically generated instances.

### 2.2.2 Cluster based oversampling

Cluster based oversampling tries to split the input space into similar groups and then uses sampling techniques to for re-adjustment of the cluster sizes. In [49], Adaptive semi supervised weighted oversampling technique is used which generates synthetic samples on the basis of weighting mechanism after determining the minority cluster sizes using cross validation. Another self-organizing maps oversampling technique [17] is used which preserves the underlying manifold structure of data and splits the input space into two dimensional representation and then apply SMOTE for generating inter and intra cluster synthetic data. In [8], the author uses Density-based SMOTE, which generates synthetic samples from each minority class shortest path instance to a pseudo-centroid of the clusters discovered from a DB-SCAN algorithm.

### 2.2.3 Synthetic oversampling

A very well-known and widely used technique for oversampling introduced in [10] is SMOTE - Synthetic minority over sampling which has novelty of generating synthetic instances by identifying nearest neighbors of the feature vectors through the Euclidean distance. Contrary to the random resampling techniques, SMOTE generated minority class samples make decision boundaries flexible and suitable for low dimensional data due to its linear nature hence lack variance in the generated samples which leads to poor generalization. Variations of smote includes borderline-1 and borderline-2 which generates the minority instances close to the borderline. In [29] Borderline-1 generates near the positive samples whereas borderline-2 generates samples close to the negative nearest neighbors. In [11], the accuracy for minority class was improved by extending the SMOTE using a boosting technique and Cat-Smote in [50] distinguish using the variances through discriminating the continuous and categorical variables. Another technique Adasyn introduced in [32] generates synthetic instances close to the misclassified samples using knn and creates decision boundaries. This is not adopted by the earlier techniques of SMOTE and its variances.

### 2.2.4 Undersampling

In [46], majority class is under sampled using N nearest neighbor concept based on NearMiss and its variations. In NearMiss1, positive samples are selected in case of smallest distance to N negative samples whereas vice versa in NearMiss-2. NearMiss-3 undergoes a step wise approach. In [60], introduced Instance Hardness Threshold which selects high probability samples after training a classifier on the input data and eliminates the samples with low probabilities. In [37], Tomek links eliminates the majority samples where nearest neighbor link between majority and minority instances is small however this setting is configurable. In [41], eliminated redundant samples and noise from the majority class. Edited Nearest Neighbour removes the samples which does not meet the selection criteria of nearest neighbor on under sampled instances.

### 2.2.5 Resampling

Resampling has been the traditional approach to handle class imbalance. The technique uses randomly eliminating the majority class samples and duplicating the minority class samples to balance the dataset discussed in [35]. In [9], the impact is discussed of adopting

the above approach. Demerits discussed are over fitting due to simple oversampling and loss of meaningful information in case of under sampling. In [18], the author has reported improved accuracy on several base classifiers on balanced datasets.

### 2.2.6 Hybrids

In [5, 4], SMOTETomek and SMOTEENN techniques are introduced to solve the problem with generated synthetic samples of being noisy i.e., either the samples lie in outliers or inliers and hence are cleaned using Edited nearest neighbors and Tomeks link. This technique performs oversampling prior undersampling and hence termed as hybrid.

## 2.3 Online Learning

This section of literature review will study and examine the focuses on approaches and techniques specifically for online learning so that the model can be trained by overcoming the catastrophic forgetting. "Catastrophic forgetting occurs when a neural network loses the information learned in a previous task after training on subsequent tasks" [59]. This is still a limitation for artificial intelligence systems where tasks are learned sequentially.
One widely-adopted and successful approach has been to design a connection between neuroscience and machine learning in which intelligent agents must own two individual learning systems, the first obtains structured knowledge (representations) step by step whereas the other quickly learns particular of individual experiences [42]. The weighing of individual experiences through recurrent activation can help in achieving generalization.
In [40], the proposed approach remembers previous tasks by selectively slowing down learning on the weights important for those tasks and validated that the technique is scalable and effective by demonstrating on a classification task of well-known hand-written digit dataset and also on multiple Atari 2600 games in a sequential learning manner. Similar to this, for learning a probability distribution on the weights of a neural network, a backpropagation-compatible algorithm, called Bayes by Backprop was presented in [7]. Specifically, deep neural networks has not made any remarkable progress in achieving continual learning however [61] neural networks with competing linear units tend to outperform those with non-competing nonlinear units and overcomes catastrophic forgetting as in biological neural networks, there is a local competition among nearest neurons, thus the choice of activation functions should always be cross-validated [27]. When developing flexible, intelligent and general purpose artificial intelligence solutions, lifelong learning becomes so important to deal with the limitations and challenges of learning many sequential tasks. In [57], a continual lifelong learning algorithm is proposed which keeps a sparsely shared basis for all the tasks and improves over time to achieve the maximum performance on all tasks. The knowledge learned in multiple tasks is transferred by altering the basis in a multi-task setting. Many variants of lifelong learning have been introduced in the literature [47, 51] which design meta-algorithms to generalize current stationary learning algorithms to multiple non-stationary ones to make the model capable of identifying the class and make it memory bounded in a piece-wise repeating sources. Another Bayesian approach to split the stream data into postulated task segments and developing the model for each task which is then under the logarithmic loss is validated empirically in multi-task setting for task identification. Similarly, to identify the sub-task that the sample most likely belongs to, a clustering

algorithm was used at the time of testing was proposed in [25], in which an individual read-out layer was trained for each sub-task.

## 2.4 Frameworks for Data Reduction

This section of literature review will study and examine the focuses on approaches and techniques specifically for data reduction so that training could be made possible where we have large datasets that does not fit in memory. In [56] a data reduction framework was proposed which would create value on enterprise-level as three data reduction layers were added which will discard data points as per enterprise rules.
There are a lot of challenges tied to the volume of data and also challenges tied to reduction of data in the form of increased error. Continuing on the topic of data reduction, many variants are proposed in literature which includes sample reduction [44], feature selection [31, 14], and clustering algorithms to extract representative data instances [65, 43]. A simulated annealing based feature selection approach was applied for detecting denial of service attacks which may become impractical if there are thousands of features [38].

## 2.5 Big Data Framework

Technology-centric elements are frame of reference in this section which enables the organization to perform predictive analytics in all big data 97 characteristics, which are described in terms of volume, veracity, velocity, variety and value [6].
To develop any in-house capability to analyze huge volume of data or use it to train a machine learning model, many commercial off the shelf technologies to collect, store, analyze [63] and extract data at different levels (real-time, cloud, offline, edge etc.) are available such as Map Reduce, Hadoop, Spark etc. to kick-start the capability of an organization for predictive analytics.
Spark can only process data that fits in the memory of its worker node (machine memory part of which is assigned to spark) however, using distributed environment we cannot train the model using gradient descent because each worker node should try to retain the required part of the RDD in memory at maximum instead of flushing data to disk [58]. As this RDD will be repeatedly iterated over in gradient descent optimization, this has to be stored in memory. As a workaround, Least Recently Used caching is used in which items are removed from the ram based on the last recently used data.
The other issue is that spark has some synchronization barriers for SGD because it requires more iteration. The proposed solution "Hog-wild" cannot be implemented on Spark as it requires wait to finish the broadcast, and then map and reduce and, then next broadcast. We can't tell Spark "not to wait". Spark computations are meant to be fault-tolerant, So, Spark is not well suited for SGD Hog-Wild. To deal with synchronization barriers, we have to program a parameter server to store weights as a single copy to share across the workers to compute gradient.
Apache Mahout is a distributed linear algebra framework which provides scalable ML libraries on Hadoop and has multiple distributed engine bindings like spark, flink H2O [21] but it also came across the same limitations as spark of synchronization barriers. Other limitations include availability of customized algorithms as there are limited algorithms available in this library which does not include neural networks.

Among many other big data technologies such as Hadoop, MapReduce, and Spark [20], parallel computing also plays a vital role for processing enormous volume of datasets. To achieve high performance computing, algorithms are designed to parallelize the tasks and computations [24, 36]. High-Performance Computing (HPC) combines the available computing power in order to do extreme amount of work in least possible time and due to its performance capabilities, it converged towards big data fields [55].
These performance capabilities are supported through the combination of 97 diverse hardware and developing parallelized algorithms and software. However, HPC languages and cluster architectures differ from the architectures of big data platforms as they are written in high-level programming languages and are designed as a distributed architecture [66]. On the contrary, enormous volume of big data may become bottleneck for parallel programming models such as Open Multi-Processing (OpenMP), accelerator models (CUDA, OpenCL, OpenACC) and MPI from sustaining high levels of parallelism [12]. This misalignment results in synchronization barriers which has failed to fully utilize the capabilities and resources of HPC and other parallel programming models for big data problems [1].

## 2.6 Open Source Libraries

This section of literature review will focus on the libraries from different programming platforms used to train the model that does not fit in memory. Following are some of the libraries.

1. Crème, a python library for online learning is recently published on google colab which is used to train the machine learning model on very huge datasets that does not fit in memory. This technique train the model from the stream of data just like stochastic gradient descent which process 1 observation at a time [28]. Currently, this work is done for two machine learning models e.g. Decision Trees and Gradient boosting and is underway for Bayesian linear models but not for any deep learning model.

2. Dask, a python library works under its ecosystem for parallel computing scalable to distributed clusters and multi-core machines by efficiently streaming data from disk [2]. To overcome the memory issues, dask executes computations in less memory by fetching data in chunks from disk. Dask also extends the numpy, pandas and scikit learn in python and is in development phase for implementing the hog-wild solution and setting up a parameter server for enabling parallel computing in neural networks in distributed environment.

3. Apart from the above, many progressive loading libraries are also available that loads data into memory as-needed for training and requires algorithms like stochastic gradient descent that can learn in an iterative manner, instead of algorithms which holds the whole data in memory to perform complex matrix operations [52].
   Although the above techniques solve the memory related issues but on contrary increases the training time heavily and also have implementation for limited algorithms – however the technique developed in this paper not only solves the memory related problems but also reduces the training time as compared to online learning and process the data in one go (batch-wise learning) to fit in memory.

# Chapter 3

# AML Pre-Processing Phase

In this section, we will be discussing the most important and time consuming activity in any data science project i.e. preparing dataset and it's pre-processing. The bank x TMS system is based on certain rules related to some scenarios which are detecting the well-known money laundering patterns and generates alerts if specific threshold is breached. The alert then follows investigation by the AML analyst and is decided as suspicious or normal. The monitoring of a particular account or customer starts in two major phases. One is when the customer is on-boarded, complete profile of the customer is assessed and assigned a score based on the profile, also known as KYC (Know your customer). The second phase is after on-boarding of the customer, the bank monitors the account transactions and behavior of the customer. Major data related challenges faced in this domain are class imbalance, volume, complexity of the data and class mislabeling. Class imbalance is the most common problem in the fraud and AML domain because the normal population is too high as compared to the few fraudulent cases. Volume refers to the amount of data produced by the customers' transaction and other activities with the bank. Class mislabeling is the crucial of all because machine learning purely relies on the data and this problem will affect the data quality in a sense that it will be very difficult to predict correctly from any model. Other data related limitations are discussed in detail in a separate section. Following are some steps discussed in pre-processing phases.

## 3.1 Rules and Thresholds Compilation

Bank x has implemented Rule based TMS system and acquiring the rules set of the scenarios and compiling will lead to identification of scenarios leading to flagging of suspicious transactions. The understanding of the FCCM was built with the discussions with business and IT Teams and complete data dictionary was prepared including some other artifacts.

## 3.2 Data Acquisition and Labelling

Acquisition of financial data will comprise of customer's transactions history over a specific time period, customer's KYC profile, customer's channel and product usage, customer demographics, customer segmentation details and scenario specific parameters. A labelling utility is to be developed to label the data as a binary-class (SAR/NON-SAR) problem dataset using the Worthy Alerts data briefed in the following section. The SQL utility will use the rules and thresholds to exhibit labelling.

## 3.3 Worthy Alerts / SARs

The AML team delivered their assessment of worthy alerts - these are genuinely suspicious alerts that needed to be investigated, which also include SARs. They have various levels of suspiciousness, but in general these alerts should be generated by TM system, because it indicates abnormal behavior, even if enough evidence was eventually found for SAR/STR submission.

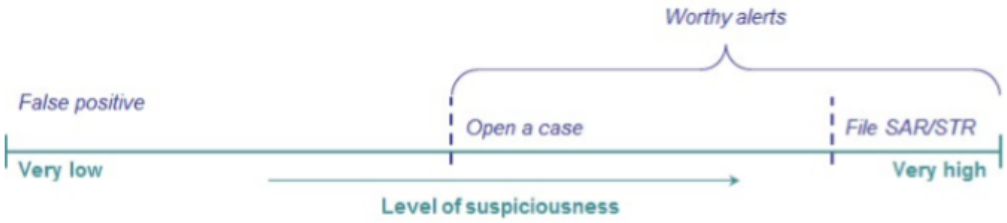


Figure 3.1: Alert level of Suspiciousness

## 3.4 Data-Set Preparation

### 3.4.1 Prepare Data Structure

From the literature review, we assessed the data requirements for the AML detection and acquired the data in the csv formats or in the form of tables in the database. The data was assembled at the local computer and the database relationships and indexes were created. Data form multiple sources includes around 93 million transactions of X million active customers for the period of 12 months, demographics, FCCM scenario related data aggregated on daily basis over the account, average balances, customer to account mapping tables, segmentation related transaction features and aggregated features of the customers. List of worthy alerts and SARs marked by the AML team was also provided from the FCCM for the above said period.

### 3.4.2 Consolidate Raw Transactions

Raw transactions were aggregated and joined with average balances, transaction behavior features and customer and activity risk features from FCCM. After consolidating the raw transactions, these were labelled with the class variable as a SAR or Non-SAR with the help of the SQL Utility developed. There was huge class imbalance observed in the data, out of X Million customers, only 3,647 are marked as suspicious accounts. The whole consolidation was done partially on SQL server and python notebooks.

### 3.4.3 Feature Engineering

Raw features does not provide enough granular view to set appropriate thresholds, thus decreases detection accuracy. The current (legal status based) data does not provide sufficient granularity to monitor different Money Laundering patterns and terrorist financing detection behavior that are detected based on transactional activities. E.g. The corporate group

contains all companies, from small, medium to large. When comparing annual turnover in 2017, customers in the same group vary from 1M PKR to close to 1TN PKR, yet the thresholds applied to transactions for all the companies are the same. So the additional features were extracted based on transactional patterns within last 12 months, e.g. the total turnover across all customer accounts, income stability, wire transaction ratio, account balance, cash ratios, etc by uncovering the FCCM core tables and their different mappings at account and customer level.

The features chosen were carefully hand-picked by domain experts and were extracted by converting the existing attributes, taking their mean, standard deviation on monthly and weekly basis, sub-dividing the transaction amounts into more granular units so as to capture the overall view and represent the large set of data. Hence the features are now more granular, yet reasonably sized for proper threshold determination and more homogeneous from the transaction perspective, thus more easily to set threshold and capture different behavior. The aim is to create data-driven features that would differentiate customers in FCCM from the perspective of Transaction Monitoring, and would fit conceptually with FCCM logics. Features should be technology agnostic (i.e. can be implemented under different technological changes and circumstances). Data is aggregated per customer across all accounts and measured on a monthly basis for the past 12 months. Only accounts that had at least one transaction during the period were included in the analysis.

Four groups of features are created, and from each of the groups the most differentiating criteria are selected. The table below lists criteria within each group. We found the data quality sustainable for these features and transformed data to be usable for machine learning. This approach ensures the quality both from analytical and human perspective so that no major detecting factors are omitted.

### 3.4.4 Create Customer Profile

From the above collected features defining customer behavior in terms of their financial activities, joining operations over multiple tables was performed in SQL including demographics etc. was performed which are highly time and resource consuming due to huge volume of data. The individual customer profile was created based on their transaction behavior, demographics and other details of the customer.

### 3.4.5 Handling Missing Values

When dealing with the transactional banking data, missing values is the most common issue faced while extracting data because various fields in the data are intentionally ommitted due to irrelevance in different context. However, these missing values can adversely affect the model performance so as a remedy we handled missing values by removing all the transactions.

### 3.4.6 Data Extract Validation

The extracted data set was validated in terms of its completeness and cleanness. Recommended tests for completeness were to:

- Verify the number of distinct Customer-IDs vs total number of records

| | Criteria | Description |
|---|---|---|
| Volume | Total turnover* | Total amount of incoming and outgoing transactions |
| | Total amt in* | Total received amount |
| | Total amt out* | Total sent amount |
| | Total trn no in-out* | Total number of incoming and outgoing transactions |
| | Total trn no in* | Number of incoming transactions |
| | Total trn no out* | Number of outgoing transactions |
| Behavioral | Savings amt | Difference between received and sent transaction amount |
| | Savings ratio | Ratio of received amount to sent transaction amount |
| Irregularity | Stdev of income | Standard deviation of monthly income |
| | Stable income months | Number of months with income as fraction of a year |
| | Dormancy months | Inactive months since last activity as of 1st Jan 2018, across all accounts |
| Digital profiling | Cash in ratio | Ratio of incoming cash transaction amount to total incoming transactions |
| | Cash out ratio | Ratio of outgoing cash transaction amount to total outgoing transactions |
| | Wire in ratio | Ratio of incoming wire transfers amount to total incoming transactions |
| | Wire out ratio | Ratio of outgoing wire transfers amount to total outgoing transactions |

* Monthly average

Figure 3.2: High level view of feature groups

- Check the active number of clients/accounts against official statistics published in the Bank. An active client is defined as a client who had at least one transaction in the given period
- F or a cleanness check, functions like summary() and str() were used to verify the format numbers of “NA”s.

### 3.4.7 Sampling

As feature calculation and model training on 93M transactions require substantial compute in terms of hardware which was not available at hand so as a remedy the majority class was then down sampled with the ratio of 1/1000 and the final dataset was prepared with 128 features and 3,647,000 observations. After one hot encoding of categorical features, there were 329 features as a whole with 3,296,087 transactions for which the features were

available. The actual imbalance in the data was 0.00389% whereas after sampling it was reduced to 3.89%.

### 3.4.8 Excluding the Effective Risk Variable

As the results of the initial experiments were approaching to 99% AUC, we revisited the features with the business and their impact on the class variable (SAR/WA) and analysis revealed that Effective risk has inverse effect on alert riskiness, i.e. the higher is Effective risk, the less risky is alert. Effective Risk Rate is an example of a categorical alert risk driver which has 10 levels, with 1 being the lowest risk and 10 - the highest.

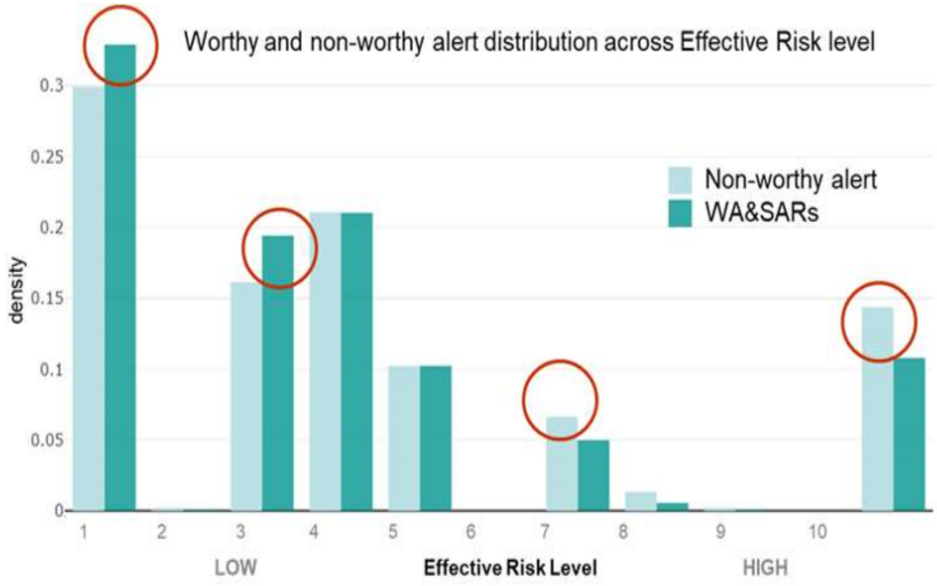


Figure 3.3: Effective risk rate calculated in FCCM 6 has opposite effect on alert riskiness

### 3.4.9 Normalization

Normalization of features is an essential step for certain algorithms and metrics. In this case, it is the generative models. The most widely used normalization techniques include MIN-MAX normalization and Z-score. Both of these techniques were tested. After data exploration, it was determined that MIN-MAX normalization contributes biases to model due to the large number of outliers present in the data. In such a situation, Z-score proved to be a more suitable normalization approach and was consistently applied for all variables. Data was demeaned and normalized by standard deviation to bring the distribution close to the standard normal N(0,1).

### 3.4.10 Limitations

- Data quality issues, that would reduce misclassification cases are outside of the scope and are not remediated.

- Effective and Activity risk calculations were corrupted/incomplete at some point in FCCM 6 driving errors in the data used to construct the model
- CRR score is not properly integrated in FCCM 6 (it flows as KYC risk element in customer's Effective risk, yet currently it is used to construct Business risk). CRR integration requires KYC module output adoption for TMS purpose (scaling, bucketing).
- Historic data limitation – historic data earlier from 2017 was not utilized for modeling because of data server space limitation
- Latest available WA and SARs earlier from 2017 were not utilized for model design because of historic data limitation on alert generation (previous point)
- The data for modeling was taken as is from FCCM without performing detailed reconciliation with original sources, or validation of its correctness in FCCM after FCCM processing. After the integration of additional data sources into FCCM, the model should be reviewed, and a change in the transactional patterns due to the new products should be considered.
- Quality of model training labels (Worthy Alert or SAR) relies on the investigators and precise selection of the closing action (e.g. duplicity, closed, proceeded to investigation). Worthy Alert  SAR quality was not addressed:
  - If WA and SAR suspiciousness and filling reasons correspond scenario that generated alert (e.g. in separate meeting it was found that sample of SARs based on alerts from scenario "Structuring: avoidance reporting threshold" does not indicate structuring or avoidance patterns)
- Initial training of the model was therefore done on limited set of variables which are currently known in FCCM and are used for alert generation. Additional criteria were explored by establishing the detailed understanding with business in multiple meetings and understanding their flow into FCCM from core banking are discussed in detail in the Feature Engineering section.

## 3.5 Exploratory data analysis

An extensive EDA pipeline was executed comprising of the following:

### 3.5.1 Continuous and Categorical Features identification

As the initial part of the EDA, we identified continuous and categorical variables.

| | v1 | v2 | v3 | v4 | v5 | v6 | v7 | v8 | v9 | v10 | ... | v119 | v120 | v121 | v122 | v123 | v124 | v125 | v126 | v127 | v128 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 3460322014591 | 21855691138 | IND | SIND | | 201705 | 5 | 0 | 0 | 0 | ... | CA | 0 | 0 | 0 | 1 | 0 | 0 | 23 | 0.00 | 0 |
| 1 | 3460322014591 | 21855691138 | IND | SIND | | 201706 | 4 | 0 | 0 | 0 | ... | CA | 0 | 0 | 0 | 1 | 0 | 0 | 23 | 147121.43 | 0 |
| 2 | 3460270758543 | 21855530593 | IND | SIND | | 201706 | 6 | 0 | 0 | 0 | ... | CA | 0 | 0 | 0 | 1 | 0 | 0 | 23 | 199368.66 | 0 |
| 3 | 3460328832255 | 21855765242 | IND | SIND | | 201707 | 1 | 0 | 0 | 0 | ... | CA | 0 | 0 | 0 | 1 | 0 | 0 | 23 | 1475.39 | 0 |
| 4 | 3430102640787 | 21854691861 | IND | SIND | | 201710 | 12 | 0 | 0 | 0 | ... | CA | 0 | 0 | 1 | 1 | 0 | 0 | 23 | 6572.10 | 0 |

Figure 3.4: All Data Snapshot.

| | v1 | v2 | v3 | v4 | v5 | v6 | v7 | v8 | v9 | v10 | ... | v108 | v112 | v113 | v114 | v115 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| count | 3296087 | 3296087 | 3296087 | 3296087 | 3296087 | 3296087 | 3296087 | 3296087 | 3296087 | 3296087 | ... | 3296087 | 3296087 | 3296087 | 3296087 | 3296087 |
| unique | 1719786 | 1719786 | 3 | 11 | 1 | 8 | 203 | 1 | 1 | 1 | ... | 35 | 3 | 6 | 32 | 10 |
| top | MA8JB8 | 21845859997 | IND | INDV | | 201710 | 2 | 0 | 0 | 0 | ... | EA | M | M | 23 | 1 |
| freq | 227 | 227 | 3131919 | 1561387 | 3296087 | 813794 | 451854 | 3296087 | 3296087 | 3296087 | ... | 1465724 | 2575000 | 1985655 | 971576 | 1195396 |

Figure 3.5: Statistics for Categorical Features.

| | count | mean | std | min | 25% | 50% | 75% | max |
|---|---|---|---|---|---|---|---|---|
| v101 | 3296087.0 | 1.573741 | 1.674853e+01 | 1.000000 | 1.000000 | 1.000000 | 1.000000 | 1.033500e+04 |
| v102 | 3296087.0 | 2.841050 | 2.110160e+00 | 1.000000 | 1.000000 | 3.000000 | 5.000000 | 1.000000e+01 |
| v104 | 3296087.0 | 168478.171487 | 1.308584e+07 | 0.010000 | 5000.000000 | 15000.000000 | 38700.000000 | 1.294746e+10 |
| v109 | 3296087.0 | 0.033680 | 1.804040e-01 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 1.000000e+00 |
| v110 | 3296087.0 | 7.258413 | 7.134722e+00 | 0.000000 | 2.861054 | 4.832306 | 8.955509 | 6.890623e+01 |
| v111 | 3296087.0 | 34.635507 | 1.777774e+01 | -31.638603 | 26.691307 | 34.770704 | 46.360027 | 6.835866e+01 |
| v120 | 3296087.0 | 0.070686 | 2.562995e-01 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 1.000000e+00 |
| v121 | 3296087.0 | 0.004326 | 6.562793e-02 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 1.000000e+00 |
| v122 | 3296087.0 | 0.195993 | 3.969630e-01 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 1.000000e+00 |
| v123 | 3296087.0 | 0.866215 | 3.404209e-01 | 0.000000 | 1.000000 | 1.000000 | 1.000000 | 1.000000e+00 |
| v124 | 3296087.0 | 0.061666 | 2.405471e-01 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 1.000000e+00 |
| v125 | 3296087.0 | 0.477044 | 1.540803e+00 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 1.400000e+01 |
| v127 | 3223933.0 | 237190.843189 | 1.552038e+06 | -403883.480000 | 3081.190000 | 17828.330000 | 103515.610000 | 4.715412e+08 |
| v128 | 3296087.0 | 0.001364 | 3.690364e-02 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 1.000000e+00 |

Figure 3.6: Statistics for Continuous Features.

### 3.5.2 Univariate Analysis

Continuous feature histograms are generated to observe variance in the said feature set.

### 3.5.3 Correlation Analysis

The example below shows that some highly inter-correlated features were still used. It is visually observable from the chart below that TOTAL_TURNOVER AND TOTAL_IN features have a high correlation. Both of features are elements, which describe volume. By including both of them, it is possible to weight those features as more important from the analysis perspective. This leads to more importance in modeling from volume perspective than from others, motivated by the fact that all following analysis in modeling, threshold settings will largely depend on volumes (size of transactions, number of transactions), and it is essential to isolate based on transaction behavior

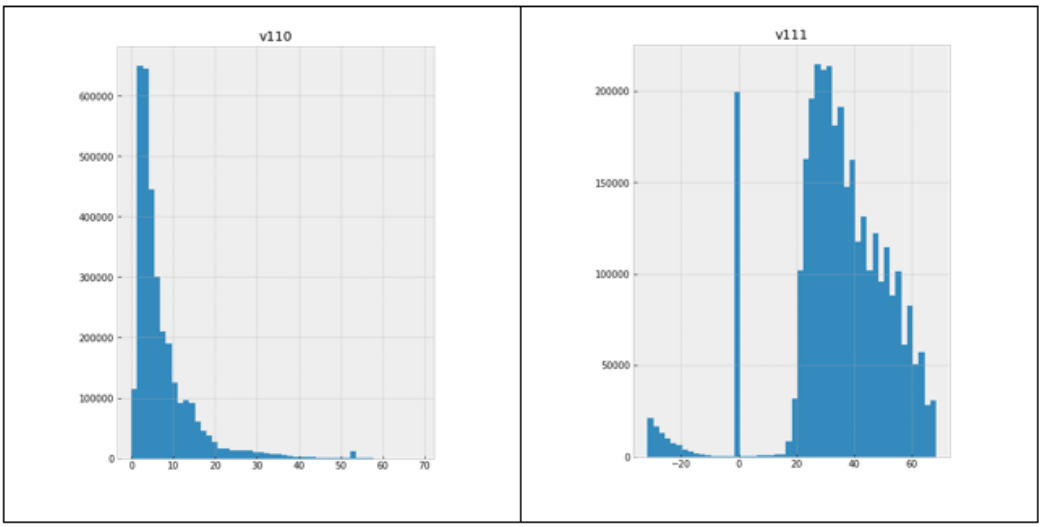


Figure 3.7: Continuous Features Histograms.

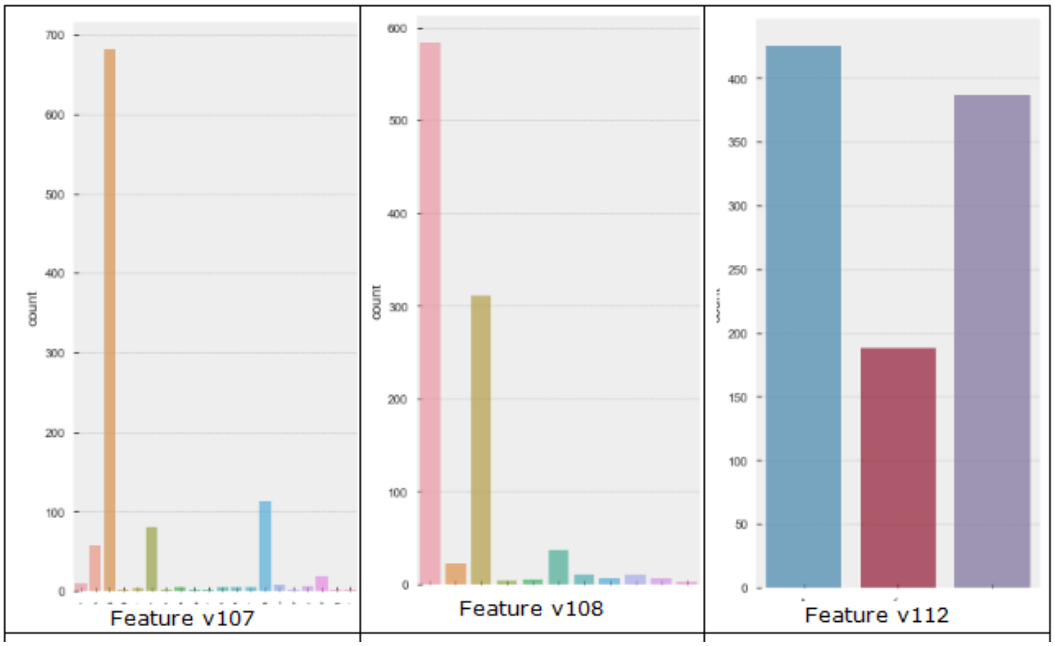


Figure 3.8: Categorical Features Bar Charts (1)

The final set of features was accepted despite the observed correlation:

- The volume of activities plays a significant role in AML detection over other features;
- In 2017, notable outgoing flows compared to incoming flows were observed (total savings negatively correlated to volume criteria).

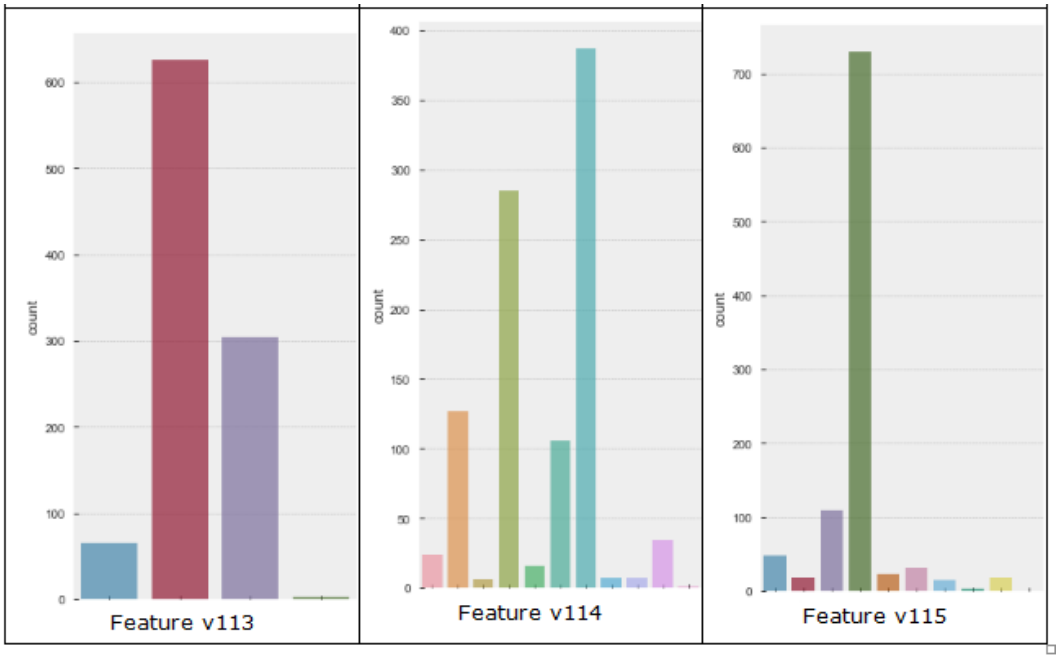


Figure 3.9: Categorical Features Bar Charts (2)

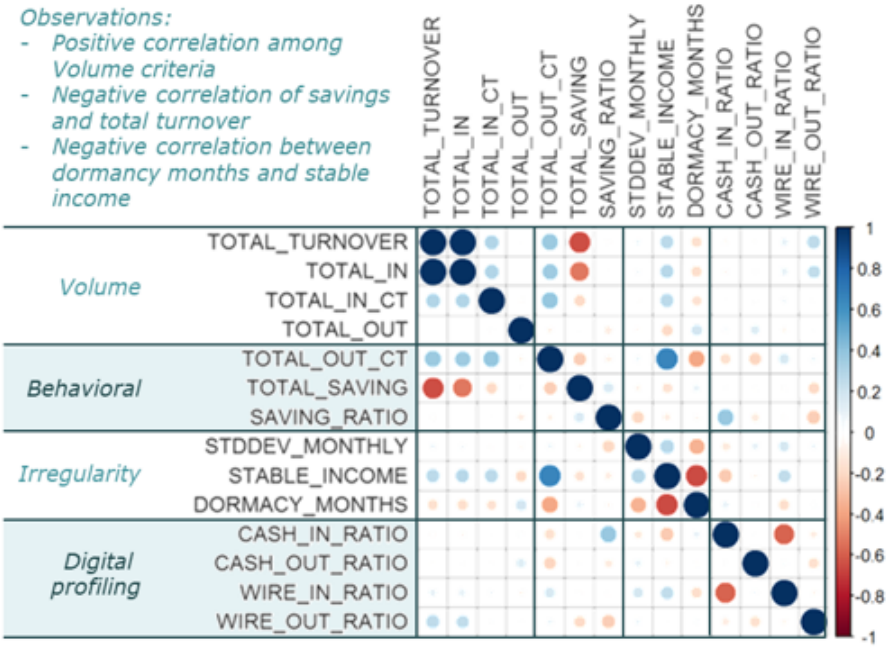


Figure 3.10: Correlation Analysis

### 3.5.4 Random Forest Permutation Importance Scores

Feature Importance was calculated using random forest algorithm. Following are the scores for top 5 features.

Table 3.1: Top 5 Permutation Importance Scores

| № | Learning Method | Feature | Weight |
|---|---|---|---|
| 1 | Random Forest | v102 | 0.0039 |
| 2 | Random Forest | v104 | 0.0018 |
| 3 | Random Forest | v96 | 0.0015 |
| 4 | Random Forest | v103 | 0.0006 |
| 5 | Random Forest | v17 | 0.0005 |

# Chapter 4

# Developing constraint imbalance counter technique

One of the biggest data related challenge faced in the domain of fraud and AML is class imbalance because the normal population is too high as compared to the few rare events of fraud/laundering thus leading to highly skewed data distribution. Money Launderers try to look like the norm which complicates the matter and coin the problem to anomaly detection. This high class imbalance is usually present in financial datasets in cases of fraud and laundering where minority class representation may be below 0.05% of the total samples in a dataset.

## 4.1 Materials & Methods

To counter this imbalance problem, two different approaches are proposed which includes cost sensitive learning and over and under sampling. The former approach modifies the cost function to optimize during training time. The latter is the prevailing sampling technique in which samples are duplicated/generated or eliminated from the dataset resulting in a balanced distribution as an effect to change in data volume.
On the contrary, density estimation based non-linear models could learn the distribution and representation of the data manifold of majority and minority classes and are then can be used for generative sampling. Such models may be classified under the umbrella of highly non-linear and expressive class of deep neural networks which are discussed in section below:

### 4.1.1 Deep, non-linear generative models

Deep non-linear generative models are capable of density approximation and thereby generating samples which are very close to the true distribution of the data having likelihood guarantees for all the parameters of the training data distribution.
Generative models when enriched with collection of high dimensional training samples are able to approximate the probability density function so well that describe the true data distribution. The joint distribution or probability density function over all variables is retained and used for generating new samples through a random process.

Recent advances in generative models have led to the development of powerful artifacts. In an effort to overcome the class imbalance challenge, we use these generative models to generate the minority class samples. In contrast to earlier traditional approaches which

were able to generate monotonous samples only, these non-linear generative models have outperformed all the prevailing linear techniques. Following variants of generative models and an undersampling technique is discussed below.

- Generative adversarial networks - GAN
- Variational autoencoders - VAE
- Restricted Boltzmann machines - RBM
- Instance hardness threshold - IHT

### Generative adversarial networks

GANs [26] depicts the notion of adversarial training where two networks compete each other in a game theoretic approach, thus the term "adversarial networks". The two networks, a generative model and a discriminative model are configured in a minmax setting to achieve the equilibrium state in which the generator model generates similar distribution samples whereas the discriminator model tries to discriminate the actual and generated samples. This simultaneous joint training of the two networks with opposite goals is defined and illustrated as:

$$L(D_dis, G_gen) = -E_{x\sim p_{X|Y=0}}[\log(1 - C(G(x, y = 0)))] - E_{x\sim p_{X|Y=1}}[\log(C(G(x, y = 1)))] \tag{4.1}$$

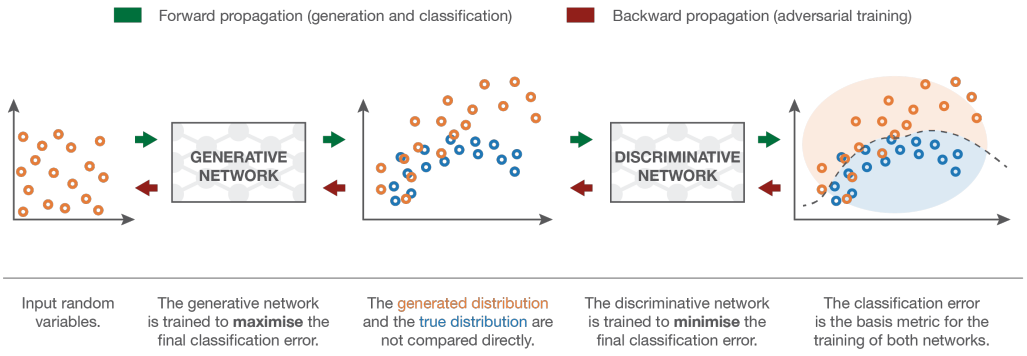


Figure 4.1: Generative Adversarial Networks [1]

Whereas the objectives of the two networks, generator and discriminator are defined as:

- *Training the generator:* The objective of the generator model is to maximize the overall classification error between generated and true distribution of the data so as to deceive the discriminator. The objective function is expressed in eq. below.

$$L(G_gen) = E_{x,y\sim p_{X,Y}}[\log(D(G(x, y)))] + L(D_dis, G_gen) \tag{4.2}$$

[1] https://towardsdatascience.com/understanding-generative-adversarial-networks-gans-cd6e4651a29

- *Training the discriminator:* The objective of the discriminator model is to minimize the overall classification error so as to identify the generated samples of the data. The objective function is summarized in eq. below.

$$L(D_d is) = -E_{u\sim p_U}[\log(1-D(u))] - E_{x,y\sim p_{X,Y}}[\log(D(G(x,y)))] \tag{4.3}$$

**Variational auto encoders**

The recent success of variational auto-encoders [39] is learning generative models in the original and latent data space with Gaussian priors. VAEs are capable of estimating the probability density (encoding) close enough to the true initial density of the data which was earlier intractable to estimate using any candidate density like Gaussian with minimization of Kullback-Leibler divergence. As these generative models induce probability in a random setting, hence are separated at a paradigmatic level from the traditional deterministic auto-encoders.

$$p\beta(x|z) = \frac{p\beta(x|z)p(z)\beta}{\beta(x)} \tag{4.4}$$

Estimating the posterior probability p(x | z)] has been intractable in graphical models which researchers has countered using two different approaches i.e, Variational inference and Gibbs sampling. VAE uses the former approach by maximizing the lower bound of the model.

$$L = E_{q(z)}[logp\beta(x|z)] - KL[q\phi(z|x)||p\beta(z))] \tag{4.5}$$

The VAE model has two terms in the loss function i.e., reconstruction error and KL divergence making the lower bound, hence the overall loss function is termed as regularized reconstruction loss. The parameter $\phi$ is used by encoder to generate the distribution parameters $\sigma$ and $\mu$ whereas decoder uses parameter $\beta$. These $\sigma$ and $\mu$ are then used by encoder to generate the representation of latent factor (x | z) and reconstruction (z | x) is also computed by decoder. An assumption by encoder is a tractable Gaussian distribution q $\phi$ (z | x) which tries to make p(z | x) distribution as close using KL divergence meaning prior near to the posterior which constitutes the second term of the loss function however the first term of the equation is an expectation maximization of the q $\phi$ ( z | x) conditional probability distribution w.r.t q(z) which minimize the reconstruction error of the decoder.

*VAE training and generation:* As both the terms in the loss function are differentiable hence VAE uses backpropagation for training. The reconstruction error uses reparameterization whereas the regularizer constitutes it as a closed form solution. A mini batch input is forwarded to the encoder where it learns the distribution parameter $\sigma$ and $\mu$ by using the parameter $\phi$. These $\sigma$ and $\mu$ are used to generate the latent factor z which is then forwarded to the decoder to generate the input (x) from the learnt latent representation. The gradients in the backward pass learn the encoder and decoder parameters $\phi$ and theta to maximize the likelihood of data. After model training, the encoder is pulled off and the samples are generated using learnt latent representation from the decoder. The latent representation z is now sampled from the Gaussian prior and new sample is generated.

#### Restricted Boltzmann machines

In [34], the authors proposed an unsupervised generative models also coined as non-linear generalization of existing dimensionality reduction technique, i.e., PCA. The models are designed in a symmetrical fashion of stochastic neurons (binary) in which a bipartite graph is formed between two layers. Although these nonlinear generative models have achieved better generalization yet a challenge of parameter optimization and model training is observed. The challenge stems from initialization of the weights which if large enough leads to local minima problem and if small enough leads to very small gradients however processed weights can learn the parameters well which require learning one layer at a time of features as every layer has high correlations with their layers unit activities.

The RBM model training is done in two steps which are:

- (i) Step 1 – Pre-training in Unsupervised settings: In unsupervised pre training, a stacked RBM is learned which consists of each having one layer of feature activations hence termed as layer wise pre training. The feature activations learned of a single RBM are processed weights which are input for the next RBM to train in the stack.
- (ii) Step 2 – Fine tuning ins supervised setting: To yield a deep auto-encoder, RBM are unfolded after pre-training which then uses back propagation for fine tuning.

#### An under-sampling technique - Instance Hardness Threshold

In this model the instances are removed on the basis of hardness computation which follows to step approach. First it learns which samples are misclassified frequently by learning multiple algorithms and second at any instance level, the reason of misclassification is identified. This is the aggregated approach which works in contrast to the existing techniques.

IHT Training: In the training, 9 classification algorithms are learned with more than 190,000 samples in 64 datasets. Initially, 20 algorithms are learned and difference between two algorithms is computed via predicted probability termed as score. This score forms a set of hardness on each sample which together are used as feedback for the lowering the majority class (under-sampling).

### 4.1.2 Proposed Model

To counter the class imbalance using data driven technique, the proposed model termed as "deep balancer" works for anomaly detection and maintaining the equilibrium in the dataset using the generative models along with an under-sampling technique. The models works in two steps:

- Step I – Oversampling using generative models
- Step II – Combining generative and reductive models

Step I – In this step, samples are generated using deep generative models which learns the join distributions and are more close to the true distribution as compared to the samples generated using synthetic or re-sampling techniques. The balance in the dataset is attained leading to improved performance metrics. This step encompasses of three deep generative models which GANs, VAEs and RBMs. VAEs and RBMs approximate the density function which are intractable hence the RBMs use gibbs sampling whereas VAEs use variational

inference for density estimation. GANs on the other hand use game theory approach to learn the representation.

Step II – Combining generative models with counterpart reductive IHT under sampling technique. IHT removes the low probability samples from the dataset using a complex decision boundary. The main goal of the this step is to bring equilibrium in the dataset by generating the minority class samples and removing the majority class in a systematic manner. This two-fold effect of generative samples of minority class from the learned joint distribution and removing majority class noisy samples improves the performance metrics which are discussed in the experimental setup.

## 4.2 Experimental Setup

In this section, details of dataset, model configuration and evaluation metrics used in the experimental setup. Table 1 shows three datasets used in the experiments

### 4.2.1 Datasets

| Name | Access | Dataset | | | |
|---|---|---|---|---|---|
| | | Non-Anomalous (N) | Anomalous (A) | Total (A+ N) | Imbalance |
| AML Dataset | Proprietary | 9,995 ,500 | 4500 | 10M | 0.04% |
| Credit Default | Public | 247 ,344 | 235, 71 | 270 ,915 | 8.7% |
| Credit Fraud | Public | 255 ,856 | 470 | 256 ,326 | 0.18% |

Table 4.1: Datasets

- *AML dataset:* Anti-money laundering dataset is a proprietary comprising of two classes namely SAR (suspicious activity report) and Non-SAR transactions of the individuals in a financial institution. These transactions were accumulated for the period of 1 year. The dataset was highly imbalance (0.04%) with the minority class of SAR (only 4500 transactions, i.e. 0.05) compared to the majority class (10 million transactions).

- *Credit default dataset:* Home credit group made this dataset publically available which contain the transactional and repayment history of the customers which are provided loan with no credit history. The company is a loan provider with the portfolio of over 100M customers and is willing to assess that the customer will default on provided loan or not based on the provided dataset which has an imbalance of 8.7%.

- *Credit card fraud dataset:* This dataset was made public by ULB, Worldline and The Machine Learning Group in a research. It contain the fraudulent and non-fraudulent credit card transactions of all the customers for a period of one month which are European credit card holders with the high imbalance of 0.18% of the minority class.

### 4.2.2 Model Configurations

Model wise configurations used in the experiments are listed in Table 4.2.

| | **GAN** | **RBM** | **VAE** |
|---|---|---|---|
| *Modeltype* | WGAN | BernoulliGuassian | Kingma |
| *Optimizer* | Adam | | |
| *Batchsize* | 64 | | |
| *Momentum* | 0.95 | | |
| *Learningrate* | 0.0001 | | |
| *Epochs* | 5000 | | |
| *Lossfunction* | CrossEntropy | MSE | KLDivergence(NLL) |

Table 4.2: Model Configurations

For classification with generative models, XGBoost was used with the parameter setting of weak learners = 100, depth = 5 and learning rate = 0.1.

### 4.2.3 Evaluation metrics

As in money laundering, it is important to reduce the false positives but is more important to take care of false negatives as the model should not miss any suspicious activity while minimizing the false positives so the measure F1 score which is the weighted average of Precision and Recall takes both false positives and false negatives into account. Precision is defined as division of correctly positives predicted over total positives predicted whereas recall is the division of correctly positives predicted over total positives actual. Intuitively it is not as easy to understand as accuracy, but F1 is usually more useful than accuracy, especially if you have an uneven class distribution. In the experiments, the evaluation metrics used along with F1 Score are AUC-ROC and balanced accuracy. Balanced accuracy is a measure of true positive and true negative whereas AUC-ROC provides summary of all the metrics in the confusion matrix.

### 4.2.4 Experimental Nomenclature

For each dataset, the experiments are performed which are further categorized into 3 based on the sampling techniques i.e., synthetic, hybrid and generative sampling. The proposed model of this study is generative oversampling which uses GANs, VAEs and RBMs to generate the minority class however comparison with the synthetic oversampling is also done. Hybrid approaches include SMOTEENN and the proposed combining generative with reductive models technique.

Due to spatial constraints, the result tables are paired by breaking up into two with the naming convention of I and II.

## 4.3 Results

| Sampling | Model | AUC | TP% | TN% | FP% | FN% |
|---|---|---|---|---|---|---|
| Original | Default | 0.78 | 0.09 | 99.84 | 0.02 | 0.05 |
| Hybrid | VAE+IHT | **0.95** | 0.13 | 95.94 | 3.92 | 0.0021 |
| | SMOTEEN | 0.82 | 0.09 | 99.54 | 0.33 | 0.04 |
| Synthetic | SMOTE | 0.8 | 0.12 | 99.59 | 0.23 | 0.06 |
| | ADASYN | 0.8 | 0.12 | 99.59 | 0.23 | 0.06 |
| Generative | GAN | **0.95** | 0.1 | 99.85 | 0.01 | 0.03 |
| | VAE | 0.84 | 0.09 | 99.85 | 0.01 | 0.04 |
| | RBM | 0.84 | 0.09 | 99.85 | 0.02 | 0.04 |

Table 4.3: AML Dataset - I

| Sampling | Model | Precision | Recall | Specificity | Balanced Accuracy | F1Score |
|---|---|---|---|---|---|---|
| Original | Default | 0.84 | 0.64 | 0.99 | 0.82 | 0.73 |
| Hybrid | VAE+IHT | 0.03 | **0.98** | 0.96 | **0.97** | 0.06 |
| | SMOTEEN | 0.21 | 0.71 | 0.98 | 0.86 | 0.34 |
| Synthetic | SMOTE | 0.34 | 0.67 | 0.99 | 0.83 | 0.46 |
| | ADASYN | 0.34 | 0.66 | 0.99 | 0.83 | 0.45 |
| Generative | GAN | **0.9** | **0.76** | 0.99 | **0.88** | **0.83** |
| | VAE | **0.86** | 0.69 | 0.99 | 0.85 | **0.77** |
| | RBM | 0.84 | 0.68 | 0.99 | 0.84 | 0.76 |

Table 4.4: AML Dataset - II

Results of AML dataset: GANs and VAE + IHT shares the best AUC i.e., 0.95. Summary of the results is: GAN achieved highest precision i.e., 0.9 whereas 0.86 is achieved by VAE with lowest false positives (0.01%). VAE+IHT also achieved highest recall (0.98) whereas GAN achieved (0.76) with false negatives (0.0021% ) and (0.03%) which are remarkably low. GANs also achieved highest F1 score (0.83) whereas VAE (0.77). VAE+IHT achieved 0.97 balance accuracy whereas GAN (0.88).

| Sampling | Model | AUC | TP% | TN% | FP% | FN% |
|---|---|---|---|---|---|---|
| Original | Default | 0.52 | 0.47 | 91.43 | 0.41 | 7.69 |
| Hybrid | VAE+IHT | **0.76** | 5.63 | 75.47 | 16.4 | **2.52** |
| | SMOTEEN | 0.59 | 1.73 | 89.05 | 2.8 | **6.43** |
| Synthetic | SMOTE | 0.52 | 0.52 | 91.34 | 0.5 | 7.64 |
| | ADASYN | 0.52 | 0.47 | 91.38 | 0.46 | 7.68 |
| Generative | GAN | **0.78** | 0.53 | 91.49 | 0.44 | 7.44 |
| | VAE | 0.58 | 1.42 | 91.72 | **0.12** | 6.74 |
| | RBM | 0.58 | 1.36 | 91.71 | **0.14** | 6.8 |

Table 4.5: Credit Default Dataset - I

| Sampling | Model | Precision | Recall | Specificity | Balanced Accuracy | F1Score |
|---|---|---|---|---|---|---|
| Original | Default | 0.53 | 0.05 | 0.99 | 0.53 | 0.1 |
| Hybrid | VAE+IHT | 0.25 | **0.69** | 0.82 | **0.76** | **0.37** |
| | SMOTEEN | 0.38 | **0.21** | 0.96 | 0.59 | 0.27 |
| Synthetic | SMOTE | 0.5 | 0.06 | 0.99 | 0.53 | 0.11 |
| | ADASYN | 0.5 | 0.05 | 0.99 | 0.53 | 0.1 |
| Generative | GAN | 0.55 | 0.06 | 0.99 | 0.53 | 0.12 |
| | VAE | **0.91** | 0.17 | 0.99 | **0.59** | **0.29** |
| | RBM | **0.9** | 0.16 | 0.99 | 0.58 | 0.28 |

Table 4.6: Credit Default Dataset - II

Results of Credit Default Dataset: GAN shares the best AUC (0.78) whereas VAE+IHT (0.76), RBM and VAE (0.58 each). Prevailing synthetic techniques perform too low with AUC (0.52). Summary of the results is: VAE reported highest precision (0.91) whereas RBM (0.90) with false positives (0.12%) and (0.14%) remarkably low. VAE+IHT achieved highest recall (0.69) whereas SMOTEENN (0.21) with false negatives (2.52%) and (6.43%) remarkably low. VAE+IHT achieved F1 score of 0.37 whereas VAE (0.29). Balanced accuracy is best shared by VAE+IHT (0.76) whereas VAE (0.59).

| Sampling | Model | AUC | TP% | TN% | FP% | FN% |
|---|---|---|---|---|---|---|
| Original | Default | 0.84 | 0.04 | 99.82 | 0.02 | 0.12 |
| Hybrid | VAE+IHT | 0.85 | 0.16 | 98.66 | 1.17 | **0.02** |
| | SMOTEEN | 0.86 | 0.13 | 99.77 | 0.06 | 0.05 |
| Synthetic | SMOTE | 0.88 | 0.08 | 99.71 | 0.11 | 0.09 |
| | ADASYN | 0.88 | 0.05 | 99.69 | 0.17 | 0.09 |
| Generative | GAN | **0.92** | 0.14 | 98.9 | 0.92 | **0.03** |
| | VAE | **0.88** | 0.14 | 99.76 | **0.06** | 0.04 |
| | RBM | 0.84 | 0.12 | 99.77 | **0.05** | 0.06 |

Table 4.7: Credit Card Fraud Dataset - I

| Sampling | Model | Precision | Recall | Specificity | Balanced Accuracy | F1 Score |
|---|---|---|---|---|---|---|
| Original | Default | 0.58 | 0.22 | 0.99 | 0.61 | 0.32 |
| Hybrid | VAE+IHT | 0.11 | **0.9** | 0.99 | **0.94** | 0.21 |
| | SMOTEEN | 0.69 | 0.72 | 0.98 | 0.86 | 0.68 |
| Synthetic | SMOTE | 0.41 | 0.46 | 0.99 | 0.73 | 0.44 |
| | ADASYN | 0.24 | 0.38 | 0.99 | 0.69 | 0.29 |
| Generative | GAN | 0.13 | **0.82** | 0.99 | **0.91** | 0.23 |
| | VAE | **0.69** | 0.78 | 0.99 | 0.89 | **0.74** |
| | RBM | **0.69** | 0.68 | 0.99 | 0.84 | **0.69** |

Table 4.8: Credit Card Fraud Dataset - II

Results of Credit Card Fraud Dataset: GAN achieved best AUC (0.92) whereas VAE (0.88). Summary of the results is: RBM and VAE achieved highest precision (0.69 each) and false positives (0.05%) and (0.06%) remarkably low. VAE+IHT achieved highest recall (0.9) whereas GAN (0.82) with false negatives (0.02%) and (0.03%) remarkably low. VAE shares the best F1 score (0.74) whereas RBM (0.69). Balanced accuracy is best shared by VAE+IHT (0.94) whereas GAN (0.91).

# Chapter 5

# Countering high volume using Cramer Rao Lower Bound

The chapter focuses on the details of overcoming the second biggest data related challenge faced in the domain of fraud and AML is high volume and velocity meaning the amount of data produced by the customers' transactions and other activities with the bank as data from multiple sources includes around 93 million transactions of X million active customers for the period of 12 months. To combat this, the technique developed using Cramer Rao Lower Bound Optimization is discussed in below section.

## 5.1 Materials & Methods

This work optimizes in-memory utilization for high volume datasets by proposing flexible weight ties. The proposed model has the capacity to retain information under high volume constraint using optimum memory consumption. This is achieved by ensuring presence of the previous and current batch in memory for complete parameter update cycle. This is in distinction to other techniques which either require complete dataset to be loaded into the memory. This information preservation releases the system from complete data retention required by traditional machine learning techniques for similar procedure.

Equation below elaborates the mathematical expression of the model. Posterior probability $P(\theta|D)$ is computed from $P(DCurr|\theta)$ and $P(\theta|DPrev)$ where the former being the current and later being the previous batch respectively. The bayes rule shows the significance of two batches and the requirement of absorption of previous knowledge via computing it as a posterior approximation.

$$logP_{total}(\theta|D) = logP(D_{Curr}|\theta) + logP(\theta|D_{Prev}) - logP(D_{Curr}) \tag{5.1}$$

The Algorithm in detail is stated as follows:

**Algorithm 1:** Weights preservation using Cramer Rao Lower Bound Optimization

**Initialization:**

Split dataset into $\beta$ batches where $\beta \in \{5, 20, 40\}$

batch = b $\in \beta$

$t \leftarrow 1$ = current batch

$t-1$ = previous batch

**Training:**

**case** *t=1* **do**

1. Initialize classifier with random weights;
2. Weights optimization - for $b_t$ are learned using CrossEntropyLoss;

//Distribution learnt using cross entropy which constitutes the term 1 of the equation 5.1;

**end**

**case** *t>1* **do**

**while** $t \leq \beta$ **do**

1. Initialize classifier with weights of $b_{t-1}$;
2. Weight update - for $b_t$ are learned using CrossEntropyLoss;
3. Weight update - for $b_{t-1}$ are adjusted using Cramer Rao bound;
   3.1 Mean $\leftarrow \theta_t - \theta_{t-1}$;
   3.2 Fisher Variance $\leftarrow E\left[\frac{\partial L(\theta|X_{t-1})}{\partial \theta}\right]$;
   3.3 Loss $L(\theta) \leftarrow L_B(\theta) + \sum_i E\left[\frac{\partial L(\theta|X_{t_i-1})}{\partial \theta}\right] \cdot (\theta_{t_i} - \theta_{t_i-1})^2$;

//Weights adjusted using Cramer Rao Lower bound which constitutes the term 2 of the equation 5.1;

t++;

**end**

**end**

## 5.2 Experimental Setup

In this section details of dataset, model configurations, evaluation metrics, batch split strategy and results of similarity experiments carried out under this experimental setup. The results of the experiments of the technique are discussed in Results and discussion chapter. Table 5.1 shows the datasets used later in the experiments.

### 5.2.1 Datasets

- *AML dataset:* Anti-money laundering dataset is a proprietary comprising of two classes namely SAR (suspicious activity report) and Non-SAR transactions of the individuals in a financial institution. These transactions were accumulated for the period of 1 year. The dataset was highly imbalance (0.04%) with the minority class of

| Name | Access | Dataset | | | |
|---|---|---|---|---|---|
| | | Non-Anomalous (N) | Anomalous (A) | Total (A+ N) | Imbalance |
| AML Dataset | Proprietary | 9,995 ,500 | 4500 | 10M | 0.04% |
| Credit Default | Public | 247 ,344 | 235, 71 | 270 ,915 | 8.7% |
| Credit Fraud | Public | 255 ,856 | 470 | 256 ,326 | 0.18% |
| Paysim | Public | 6,344,995 | 8312 | 6,353,307 | 0.13% |

Table 5.1: Datasets

SAR (only 4500 transactions, i.e. 0.05) compared to the majority class (10 million transactions).

- *Credit default dataset:* Home credit group made this dataset publically available which contain the transactional and repayment history of the customers which are provided loan with no credit history. The company is a loan provider with the portfolio of over 100M customers and is willing to assess that the customer will default on provided loan or not based on the provided dataset which has an imbalance of 8.7%.
- *Credit card fraud dataset:* This dataset was made public by ULB, Worldline and The Machine Learning Group in a research. It contain the fraudulent and non-fraudulent credit card transactions of all the customers for a period of one month which are European credit card holders with the high imbalance of 0.18% of the minority class.
- *Paysim:* This dataset has been made publically available by a multinational company, who is the provider of the mobile financial service. The dataset comprises of simulated mobile money transactions based real transactions extracted from one month of financial logs which consists of around 6 million transactions out of which 8312 transactions are labeled as fraud. It is highly imbalanced with 0.13% fraud transactions as shown in Table 1.

### 5.2.2 Model Configurations

Model wise configurations used in the experiments are listed in Table 5.2.

| **Model Configuration** | |
|---|---|
| Optimizer | SGD |
| Epochs | 20 |
| Batch size | 100 |
| Learning rate | 0.001 |
| Momentum | 0.1 |
| Activation | Relu |
| No. of hidden layers | 2 |
| *Fisher multiplier | 400 |
| Loss function | CrossEntropyLoss |
| *Fisher multiplier: The Fisher is scaled by this number to form the Cramer bound. | |

Table 5.2: Model Configurations

### 5.2.3 Batch split strategy

For this study, each dataset was split into 5 and 20 batches which in total comprising the complete dataset. The proportion of data in each batch was kept equal and also equal number of anomalies were present across batches. For model training, 80% of the stratified sampled data from each batch was used and the remaining 20% was used for testing purposes. The training done on current batch is tested against all the previous batches test sets as shown in Figure 6.1, so as to gauge the information retention across batches of a dataset.

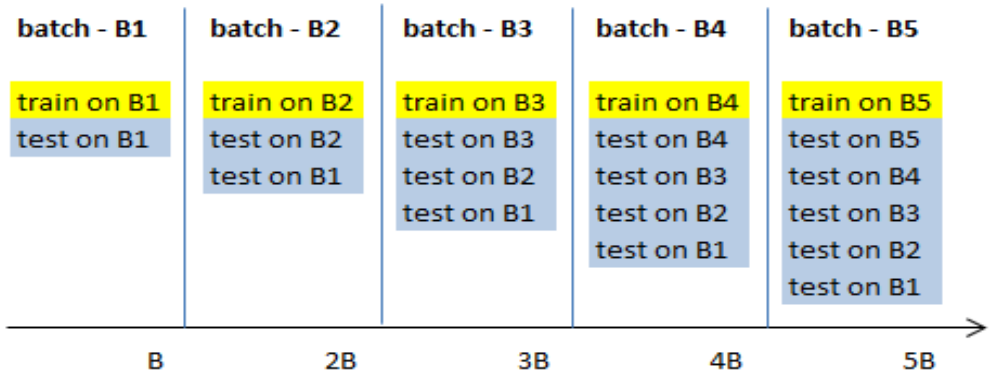


Figure 5.1: Batch testing as shown for 5 batches

### 5.2.4 Evaluation metrics

The evaluation metrics used in this experimental setup is accuracy which is measured as correctly predicted to the total predictions in the dataset.

### 5.2.5 Experimental Nomenclature

The experiments on each of the four datasets are categorized into 3 types: Cramer Rao Lower Bound, No Penalty and L2 penalty based on multiple batch settings e.g., 5 and 20 batches for each data-set. Cramer Rao Lower Bound Optimization is one of artifacts of the proposed model in this study which ensures the information retention of the previous batches when training on the current batch, however the competing techniques of optimization from the literature were also compared namely SGD in which gradient steps according to the current batch only and will minimize the loss of the current batch but destroy information of the previous batch and L2 penalty which is too severe that it tries to retain the information of previous batch only at the expense of not learning current batch data distribution. The experiments are carried out using multiple batch sizes as to validate the technique on different variances among the batches. Also in case of online setting, the technique is validated in comparison of the prevailing techniques.

### 5.2.6 Investigating for Similarity using T-Test statistic

To further investigate the results of the technique, a two sample mean t-test statistic was performed for finding the similarity among batches of a dataset. The hypothesis formulated for the same is as follows:

**Hypothesis:**

$H_0$: There is no change between the two means or distributions
($H_0 : \mu_d = 0$)

$H_1$: There is an average non-zero change between the two means or distributions
($H_1 : \mu_d \neq 0$)

In our case, the two-sample t statistic is defined as in [48]:

$$t = \frac{(\bar{x}_1 - \bar{x}_2) - (\mu_1 - \mu_2)}{\sqrt{\left(\frac{s_1^2}{n_1}\right) + \left(\frac{s_2^2}{n_2}\right)}} \tag{5.2}$$

Using the significance level of 0.05 and t from the above equation, a conservative P-value is obtained and we reject null hypothesis if critical value is less than |t| from a t-distribution with degrees of freedom = n-1.

However, the similarity between the batches was also assessed by analysing the percentage difference in the two means. The mean of a batch was calculated after averaging the mean of each column in the batch set. Following are the mean percentage difference between batches which will be referred in the results section below.

| Mean | **Batch 01** | **Batch 02** | **Batch 03** | **Batch 04** | **Batch 05** |
|---|---|---|---|---|---|
| | 34074.73 | 30468.13 | 30912.33 | 34100.34 | 31548.2 |
| **Batch 01** | 0.00% | | | | |
| **Batch 02** | -11.84% | 0.00% | | | |
| **Batch 03** | -10.23% | 1.44% | 0.00% | | |
| **Batch 04** | 0.08% | 10.65% | 9.35% | 0.00% | |
| **Batch 05** | -8.01% | 3.42% | 2.02% | -8.09% | 0.00% |
| **Net Effect** | | | -0.75% | | |

Table 5.3: Inter-batch similarity for AML dataset

| Mean | **Batch 01** | **Batch 02** | **Batch 03** | **Batch 04** | **Batch 05** |
|---|---|---|---|---|---|
| | 0.000042 | -0.000006 | -0.000004 | 0.000044 | -0.000032 |
| **Batch 01** | 0.00% | | | | |
| **Batch 02** | 800.00% | 0.00% | | | |
| **Batch 03** | 1150.00% | -50.00% | 0.00% | | |
| **Batch 04** | 4.55% | 113.64% | 109.09% | 0.00% | |
| **Batch 05** | 231.25% | 81.25% | 87.50% | 237.50% | 0.00% |
| **Net Effect** | 184.32% | | | | |

Table 5.4: Inter-batch similarity for Home Credit Default Dataset

| Mean | **Batch 01** | **Batch 02** | **Batch 03** | **Batch 04** | **Batch 05** |
|---|---|---|---|---|---|
| | 1036.999 | 3056.732 | 4432.433 | 5265.749 | 2025.034 |
| **Batch 01** | 0.00% | | | | |
| **Batch 02** | 66.07% | 0.00% | | | |
| **Batch 03** | 76.60% | 31.04% | 0.00% | | |
| **Batch 04** | 80.31% | 41.95% | 15.83% | 0.00% | |
| **Batch 05** | 48.79% | -50.95% | -118.88% | -160.03% | 0.00% |
| **Net Effect** | 2.05% | | | | |

Table 5.5: Inter-batch similarity for Credit Card Default Dataset

| Mean | **Batch 01** | **Batch 02** | **Batch 03** | **Batch 04** | **Batch 05** |
|---|---|---|---|---|---|
| | 367015.8 | 357982.5 | 351224.6 | 435812.8 | 394690.9 |
| **Batch 01** | 0.00% | | | | |
| **Batch 02** | -2.52% | 0.00% | | | |
| **Batch 03** | -4.50% | -1.92% | 0.00% | | |
| **Batch 04** | 15.79% | 17.86% | 19.41% | 0.00% | |
| **Batch 05** | 7.01% | 9.30% | 11.01% | -10.42% | 0.00% |
| **Net Effect** | 4.07% | | | | |

Table 5.6: Inter-batch similarity for Paysim Dataset

# Chapter 6

# Results and Discussion

This section is organized in further sub-sections based on the batch size of the dataset as the experiments were carried out under different settings discussed in experimental setup. Following are the results of our technique with competing state of the art techniques on 5 batches of each data-set which are formulated as:
*Results on 5 batch split:*

| L2 Regularizer | | | | | |
|---|---|---|---|---|---|
| Model Training | Testing Batch 1 | Testing Batch 2 | Testing Batch 3 | Testing Batch 4 | Testing Batch 5 |
| Batch 1 | 0.9985 | | | | |
| Batch 2 | 0.9983 | 0.9982 | | | |
| Batch 3 | 0.9977 | 0.9975 | 0.9975 | | |
| Batch 4 | 0.9981 | 0.9981 | 0.9981 | 0.9981 | |
| Batch 5 | 0.9986 | 0.9986 | 0.9986 | 0.9986 | 0.9987 |
| SGD | | | | | |
| Model Training | Testing Batch 1 | Testing Batch 2 | Testing Batch 3 | Testing Batch 4 | Testing Batch 5 |
| Batch 1 | 0.9983 | | | | |
| Batch 2 | 0.9984 | 0.9984 | | | |
| Batch 3 | 0.9981 | 0.9979 | 0.9979 | | |
| Batch 4 | 0.9973 | 0.9974 | 0.9973 | 0.9974 | |
| Batch 5 | 0.9986 | 0.9986 | 0.9986 | 0.9986 | 0.9987 |
| Cramer Rao Bound Regularizer | | | | | |
| Model Training | Testing Batch 1 | Testing Batch 2 | Testing Batch 3 | Testing Batch 4 | Testing Batch 5 |
| Batch 1 | 0.9980 | | | | |
| Batch 2 | 0.9982 | 0.9981 | | | |
| Batch 3 | 0.9983 | 0.9982 | 0.9984 | | |
| Batch 4 | 0.9980 | 0.9977 | 0.9980 | 0.9979 | |
| Batch 5 | 0.9980 | 0.9979 | 0.9981 | 0.9980 | 0.9981 |

Table 6.1: AML Dataset Results

Anti-money laundering detection dataset as shown in Table 6.1, the competing state of the art techniques and the cramer bound optimization technique shares the same accuracy with very negligible difference however information retention across batches is observed for all the techniques which was further investigated via similarity test between the batches, findings of which shows that the batches appears quite similar in distribution and inter-batch variance across batches is quite low with the net difference of only 0.75% in all the batches (see Table 5.3).

| L2 Regularizer | | | | | |
|---|---|---|---|---|---|
| Model Training | Testing Batch 1 | Testing Batch 2 | Testing Batch 3 | Testing Batch 4 | Testing Batch 5 |
| Batch 1 | 0.9131 | | | | |
| Batch 2 | 0.9155 | 0.9155 | | | |
| Batch 3 | 0.9108 | 0.9085 | 0.9098 | | |
| Batch 4 | 0.9180 | 0.9177 | 0.9185 | 0.9182 | |
| Batch 5 | 0.8996 | 0.9026 | 0.8993 | 0.9017 | 0.8980 |
| SGD | | | | | |
| Model Training | Testing Batch 1 | Testing Batch 2 | Testing Batch 3 | Testing Batch 4 | Testing Batch 5 |
| Batch 1 | 0.9103 | | | | |
| Batch 2 | 0.9071 | 0.9078 | | | |
| Batch 3 | 0.9193 | 0.9193 | 0.9193 | | |
| Batch 4 | 0.9192 | 0.9191 | 0.9192 | 0.9192 | |
| Batch 5 | 0.9113 | 0.9100 | 0.9096 | 0.9114 | 0.9093 |
| Cramer Rao Bound Regularizer | | | | | |
| Model Training | Testing Batch 1 | Testing Batch 2 | Testing Batch 3 | Testing Batch 4 | Testing Batch 5 |
| Batch 1 | 0.9172 | | | | |
| Batch 2 | 0.9180 | 0.9178 | | | |
| Batch 3 | 0.9180 | 0.9177 | 0.9159 | | |
| Batch 4 | 0.9176 | 0.9164 | 0.9148 | 0.9178 | |
| Batch 5 | 0.9173 | 0.9167 | 0.9152 | 0.9176 | 0.9216 |

Table 6.2: Home Credit Default Dataset Results

Home Credit Default dataset as shown in Table 6.2, the competing state of the art techniques and the cramer bound optimization technique stands far from each other and the best accuracy was shared by Cramer bound optimization with 0.9216 however the best information retention across batches is also observed in Cramer bound optimization technique which was further investigated via similarity test between the batches.

Findings of the inter-batch two sample mean test showed that the batches appears quite different in distribution and inter-batch variance across batches was observed quite high with the net difference of 184.32% in all the batches (see Table 5.4).

The inference obtained from this investigation is that in high volume settings, if the distribution of the data is similar then any state of the art technique could achieve comparable generalization however in case of varying distribution and high variance across batches, Cramer Rao bound optimization achieves best generalization by adjusting the weights in a way that it also retains the information of the previous batches which makes this technique work better over state of the art prevailing techniques.

| L2 Regularizer | | | | | |
|---|---|---|---|---|---|
| Model Training | Testing Batch 1 | Testing Batch 2 | Testing Batch 3 | Testing Batch 4 | Testing Batch 5 |
| Batch 1 | 0.9982 | | | | |
| Batch 2 | 0.9982 | 0.9982 | | | |
| Batch 3 | 0.9982 | 0.9982 | 0.9982 | | |
| Batch 4 | 0.0018 | 0.0018 | 0.1225 | 0.9992 | |
| Batch 5 | 0.0029 | 0.0018 | 0.8330 | 0.9982 | 0.9982 |
| SGD | | | | | |
| Model Training | Testing Batch 1 | Testing Batch 2 | Testing Batch 3 | Testing Batch 4 | Testing Batch 5 |
| Batch 1 | 0.9981 | | | | |
| Batch 2 | 0.2996 | 0.9975 | | | |
| Batch 3 | 0.9961 | 0.9982 | 0.9982 | | |
| Batch 4 | 0.0025 | 0.0018 | 0.2750 | 0.9993 | |
| Batch 5 | 0.7744 | 0.9982 | 0.9982 | 0.9982 | 0.9982 |
| Cramer Rao Bound Regularizer | | | | | |
| Model Training | Testing Batch 1 | Testing Batch 2 | Testing Batch 3 | Testing Batch 4 | Testing Batch 5 |
| Batch 1 | 0.9982 | | | | |
| Batch 2 | 0.9982 | 0.9982 | | | |
| Batch 3 | 0.9982 | 0.9982 | 0.9982 | | |
| Batch 4 | 0.9982 | 0.9982 | 0.9982 | 0.9982 | |
| Batch 5 | 0.9982 | 0.9982 | 0.9982 | 0.9982 | 0.9982 |

Table 6.3: Credit Card Fraud Dataset Results

Credit Card Fraud dataset as shown in Table 6.3, the competing state of the art techniques and the cramer bound optimization technique shares the same accuracy however information retention across batches is observed best for Cramer Rao Bound optimization technique. Further investigation via similarity test between the batches was carried out in which batches appears quite similar in distribution and inter-batch variance across batches is quite low with the net difference of only 2.05% in all the batches (see Table 5.5) hence the weights are quite stable for this dataset.

| L2 Regularizer | | | | | |
|---|---|---|---|---|---|
| Model Training | Testing Batch 1 | Testing Batch 2 | Testing Batch 3 | Testing Batch 4 | Testing Batch 5 |
| Batch 1 | 0.9987 | | | | |
| Batch 2 | 0.9987 | 0.9987 | | | |
| Batch 3 | 0.9987 | 0.9987 | 0.9987 | | |
| Batch 4 | 0.9987 | 0.9987 | 0.9987 | 0.9987 | |
| Batch 5 | 0.9987 | 0.9987 | 0.9987 | 0.9987 | 0.9987 |
| SGD | | | | | |
| Model Training | Testing Batch 1 | Testing Batch 2 | Testing Batch 3 | Testing Batch 4 | Testing Batch 5 |
| Batch 1 | 0.9990 | | | | |
| Batch 2 | 0.9917 | 0.9979 | | | |
| Batch 3 | 0.9990 | 0.9992 | 0.9992 | | |
| Batch 4 | 0.9990 | 0.9990 | 0.9990 | 0.9985 | |
| Batch 5 | 0.9988 | 0.9988 | 0.9988 | 0.9988 | 0.9988 |
| Cramer Rao Bound Regularizer | | | | | |
| Model Training | Testing Batch 1 | Testing Batch 2 | Testing Batch 3 | Testing Batch 4 | Testing Batch 5 |
| Batch 1 | 0.9990 | | | | |
| Batch 2 | 0.9992 | 0.9992 | | | |
| Batch 3 | 0.9992 | 0.9992 | 0.9992 | | |
| Batch 4 | 0.9990 | 0.9990 | 0.9991 | 0.9989 | |
| Batch 5 | 0.9990 | 0.9990 | 0.9991 | 0.9988 | 0.9991 |

Table 6.4: Paysim Dataset Results

Paysim dataset as shown in Table 6.4, the competing state of the art techniques and the cramer bound optimization technique results are close enough, still the best accuracy was shared by Cramer bound optimization with 0.9991 however information retention across batches is also observed in all the techniques which was further investigated via similarity test between the batches.

Findings of the inter-batch two sample mean test showed that the batches appears quite similar in distribution and inter-batch variance across batches was observed quite low with the net difference of 4.07% in all the batches (see Table 5.6).

| | | |
|---|---|---|
| AML Dataset | L2 Regularizer | 0.9986 |
| | Cramer Rao Bound Regularizer | 0.9985 |
| Home Credit Default Dataset | L2 Regularizer | 0.9192 |
| | Cramer Rao Bound Regularizer | 0.9194 |
| Credit Card Fraud Dataset | L2 Regularizer | 0.9983 |
| | Cramer Rao Bound Regularizer | 0.9982 |
| Paysim Dataset | L2 Regularizer | 0.9987 |
| | Cramer Rao Bound Regularizer | 0.9986 |

Table 6.5: Results for each dataset (Online Learning)

In online settings where the data with high velocity and high variance is exposed to the model for training, the proposed technique also stand out from the state of the art prevailing

techniques but in case of low variance and high similarity between the batches, the distribution to learn is not as challenging hence similar results for all the techniques. A comparison of the results is shown in the table 6.5.

*Results on 20 batch split:* The results of the Cramer Rao Lower Bound Optimization technique on 20 batch splits and comparison with state of the art techniques is stated in [Appendix A - Results], due to the spatial constraint as the result tables are large enough to fit in a single page. The results of 20 batches when compared to 5 batches were slightly improved as the inter-batch variance is increased.

# Chapter 7

# Conclusion

Organizations are devoting a large budget in big data projects but success proportion is low. One of the major reason being the high volume pressing challenge of big data and its mitigation has been a up hill task which we encountered while working with Anti-money laundering detection. Two biggest challenge of class imbalance and high volume were faced which are tackled by our research work. The study proposes the deep balancer model comprising of deep non-linear generative models which generates samples by learning joint distribution. A balanced training set is achieved by generating and reducing the minority and majority class samples respectively.

To combat the second challenge of high volume data, this work optimizes in-memory utilization for high volume datasets by proposing flexible weight ties namely Cramer Rao Lower Bound Optimization. The proposed model has the capacity to retain information under high volume constraint using optimum memory consumption. This is achieved by ensuring presence of the previous and current batch in memory for complete parameter update cycle. The technique was validated on four different anomaly datasets and also compared with the existing prevailing techniques.

From the results, it is evident that in high volume and velocity settings where the distribution is changing over time, the proposed technique improves accuracy as compared to the existing prevailing techniques (i.e., SGD and L2 penalty) on an average of 2% in a 5 batch split, however in case of 20 batches where the distribution across batches change drastically, the proposed technique improved accuracy on an average on 30% which makes this technique novel that in strict memory constraints, this technique can be scaled out to large number of batches which will result in improved accuracy and information retention.

The main contribution of this work is that it combats huge volume in on premise settings as in financial industry regulators do not allow data of any individual to be placed on the cloud as cloud security has been a major concern. Secondly, it retains the information in high volume data via memory optimization which makes it cost effective. Thirdly, in case of high inter batch variance in either volume or velocity; the information retention and classification accuracy is better than state of the art techniques.

# Appendix A

# Additional Results

Due to spatial constraint, tables are breakup into pairs so Table A:1  A:2 is read as one.

***L2Regularizer***

| Training | Batch1 | Batch2 | Batch3 | Batch4 | Batch5 | Batch6 | Batch7 | Batch8 | Batch9 | Batch10 |
|---|---|---|---|---|---|---|---|---|---|---|
| | TestingAccuracy | | | | | | | | | |
| Batch1 | 0.9985 | | | | | | | | | |
| Batch2 | 0.9986 | 0.9986 | | | | | | | | |
| Batch3 | 0.9980 | 0.9980 | 0.9983 | | | | | | | |
| Batch4 | 0.9975 | 0.9975 | 0.9975 | 0.9976 | | | | | | |
| Batch5 | 0.9981 | 0.9982 | 0.9983 | 0.9983 | 0.9981 | | | | | |
| Batch6 | 0.9986 | 0.9986 | 0.9986 | 0.9986 | 0.9986 | 0.9985 | | | | |
| Batch7 | 0.9985 | 0.9986 | 0.9986 | 0.9985 | 0.9985 | 0.9985 | 0.9986 | | | |
| Batch8 | 0.9985 | 0.9984 | 0.9985 | 0.9986 | 0.9986 | 0.9985 | 0.9986 | 0.9986 | | |
| Batch9 | 0.9968 | 0.9967 | 0.9973 | 0.9966 | 0.9967 | 0.9970 | 0.9964 | 0.9968 | 0.9964 | |
| Batch10 | 0.9985 | 0.9986 | 0.9986 | 0.9986 | 0.9986 | 0.9986 | 0.9986 | 0.9986 | 0.9986 | 0.9986 |
| Batch11 | 0.9980 | 0.9979 | 0.9978 | 0.9979 | 0.9979 | 0.9981 | 0.9979 | 0.9980 | 0.9979 | 0.9979 |
| Batch12 | 0.9968 | 0.9967 | 0.9966 | 0.9967 | 0.9964 | 0.9967 | 0.9966 | 0.9969 | 0.9969 | 0.9967 |
| Batch13 | 0.9985 | 0.9985 | 0.9986 | 0.9985 | 0.9985 | 0.9985 | 0.9985 | 0.9984 | 0.9984 | 0.9985 |
| Batch14 | 0.9975 | 0.9971 | 0.9976 | 0.9969 | 0.9972 | 0.9973 | 0.9972 | 0.9969 | 0.9969 | 0.9968 |
| Batch15 | 0.9975 | 0.9975 | 0.9975 | 0.9973 | 0.9975 | 0.9975 | 0.9973 | 0.9975 | 0.9973 | 0.9975 |
| Batch16 | 0.9981 | 0.9978 | 0.9981 | 0.9978 | 0.9974 | 0.9978 | 0.9979 | 0.9980 | 0.9978 | 0.9980 |
| Batch17 | 0.9983 | 0.9979 | 0.9981 | 0.9979 | 0.9980 | 0.9981 | 0.9980 | 0.9983 | 0.9980 | 0.9982 |
| Batch18 | 0.9985 | 0.9985 | 0.9985 | 0.9984 | 0.9985 | 0.9985 | 0.9985 | 0.9985 | 0.9985 | 0.9985 |
| Batch19 | 0.9984 | 0.9984 | 0.9983 | 0.9983 | 0.9985 | 0.9983 | 0.9984 | 0.9982 | 0.9983 | 0.9983 |
| Batch20 | 0.9982 | 0.9981 | 0.9984 | 0.9983 | 0.9981 | 0.9983 | 0.9980 | 0.9982 | 0.9982 | 0.9981 |

Table A.1: AMLDatasetResults(L2)-I

***L2Regularizer***

| Batch11 | Batch12 | Batch13 | Batch14 | Batch15 | Batch16 | Batch17 | Batch18 | Batch19 | Batch20 |
|---|---|---|---|---|---|---|---|---|---|
| TestingAccuracy | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| 0.9978 | | | | | | | | | |
| 0.9965 | 0.9968 | | | | | | | | |
| 0.9985 | 0.9986 | 0.9984 | | | | | | | |
| 0.9966 | 0.9971 | 0.9973 | 0.9973 | | | | | | |
| 0.9973 | 0.9975 | 0.9972 | 0.9975 | 0.9972 | | | | | |
| 0.9976 | 0.9978 | 0.9976 | 0.9980 | 0.9979 | 0.9978 | | | | |
| 0.9980 | 0.9982 | 0.9979 | 0.9981 | 0.9980 | 0.9980 | 0.9980 | | | |
| 0.9985 | 0.9986 | 0.9984 | 0.9985 | 0.9984 | 0.9985 | 0.9985 | 0.9986 | | |
| 0.9983 | 0.9984 | 0.9982 | 0.9984 | 0.9982 | 0.9982 | 0.9983 | 0.9984 | 0.9982 | |
| 0.9979 | 0.9982 | 0.9982 | 0.9984 | 0.9980 | 0.9979 | 0.9981 | 0.9983 | 0.9983 | 0.9981 |

Table A.2: AMLDatasetResults(L2)-II

| ***SGD*** | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Training | Batch1 | Batch2 | Batch3 | Batch4 | Batch5 | Batch6 | Batch7 | Batch8 | Batch9 | Batch10 |
| | TestingAccuracy | | | | | | | | | |
| Batch1 | 0.9980 | | | | | | | | | |
| Batch2 | 0.9975 | 0.9978 | | | | | | | | |
| Batch3 | 0.9986 | 0.9986 | 0.9986 | | | | | | | |
| Batch4 | 0.9985 | 0.9984 | 0.9985 | 0.9983 | | | | | | |
| Batch5 | 0.9980 | 0.9980 | 0.9982 | 0.9980 | 0.9979 | | | | | |
| Batch6 | 0.9985 | 0.9986 | 0.9986 | 0.9986 | 0.9985 | 0.9986 | | | | |
| Batch7 | 0.9971 | 0.9967 | 0.9969 | 0.9967 | 0.9967 | 0.9968 | 0.9968 | | | |
| Batch8 | 0.9966 | 0.9965 | 0.9965 | 0.9966 | 0.9965 | 0.9967 | 0.9964 | 0.9968 | | |
| Batch9 | 0.9979 | 0.9982 | 0.9982 | 0.9979 | 0.9981 | 0.9983 | 0.9981 | 0.9981 | 0.9981 | |
| Batch10 | 0.9986 | 0.9986 | 0.9986 | 0.9986 | 0.9986 | 0.9986 | 0.9986 | 0.9986 | 0.9986 | 0.9986 |
| Batch11 | 0.9984 | 0.9983 | 0.9984 | 0.9983 | 0.9984 | 0.9984 | 0.9984 | 0.9984 | 0.9984 | 0.9985 |
| Batch12 | 0.9981 | 0.9980 | 0.9982 | 0.9978 | 0.9978 | 0.9980 | 0.9979 | 0.9976 | 0.9977 | 0.9977 |
| Batch13 | 0.9975 | 0.9975 | 0.9976 | 0.9974 | 0.9972 | 0.9978 | 0.9974 | 0.9976 | 0.9971 | 0.9976 |
| Batch14 | 0.9986 | 0.9986 | 0.9986 | 0.9986 | 0.9986 | 0.9986 | 0.9986 | 0.9986 | 0.9986 | 0.9986 |
| Batch15 | 0.9985 | 0.9985 | 0.9985 | 0.9985 | 0.9985 | 0.9986 | 0.9985 | 0.9986 | 0.9986 | 0.9985 |
| Batch16 | 0.9985 | 0.9985 | 0.9985 | 0.9985 | 0.9985 | 0.9984 | 0.9985 | 0.9986 | 0.9985 | 0.9985 |
| Batch17 | 0.9983 | 0.9982 | 0.9980 | 0.9978 | 0.9978 | 0.9982 | 0.9982 | 0.9980 | 0.9980 | 0.9980 |
| Batch18 | 0.9959 | 0.9958 | 0.9961 | 0.9957 | 0.9956 | 0.9958 | 0.9957 | 0.9957 | 0.9960 | 0.9955 |
| Batch19 | 0.9965 | 0.9961 | 0.9957 | 0.9956 | 0.9962 | 0.9965 | 0.9959 | 0.9964 | 0.9961 | 0.9964 |
| Batch20 | 0.9986 | 0.9986 | 0.9986 | 0.9986 | 0.9986 | 0.9986 | 0.9986 | 0.9986 | 0.9986 | 0.9986 |

Table A.3: AMLDatasetResults(SGD)-I

| ***SGD*** | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| Batch11 | Batch12 | Batch13 | Batch14 | Batch15 | Batch16 | Batch17 | Batch18 | Batch19 | Batch20 |
| TestingAccuracy | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| 0.9983 | | | | | | | | | |
| 0.9974 | 0.9979 | | | | | | | | |
| 0.9972 | 0.9976 | 0.9975 | | | | | | | |
| 0.9986 | 0.9988 | 0.9985 | 0.9987 | | | | | | |
| 0.9986 | 0.9986 | 0.9983 | 0.9985 | 0.9985 | | | | | |
| 0.9984 | 0.9986 | 0.9984 | 0.9986 | 0.9984 | 0.9985 | | | | |
| 0.9979 | 0.9980 | 0.9981 | 0.9981 | 0.9978 | 0.9978 | 0.9980 | | | |
| 0.9958 | 0.9957 | 0.9958 | 0.9962 | 0.9960 | 0.9955 | 0.9961 | 0.9963 | | |
| 0.9961 | 0.9963 | 0.9959 | 0.9963 | 0.9959 | 0.9962 | 0.9961 | 0.9963 | 0.9968 | |
| 0.9986 | 0.9987 | 0.9985 | 0.9987 | 0.9985 | 0.9986 | 0.9986 | 0.9986 | 0.9986 | 0.9986 |

Table A.4: AMLDatasetResults(SGD)-II

CramerRaoLowerBoundOptimization

| Training | Batch1 | Batch2 | Batch3 | Batch4 | Batch5 | Batch6 | Batch7 | Batch8 | Batch9 | Batch |
|---|---|---|---|---|---|---|---|---|---|---|
| | TestingAccuracy | | | | | | | | | |
| Batch1 | 0.9983 | | | | | | | | | |
| Batch2 | 0.9985 | 0.9983 | | | | | | | | |
| Batch3 | 0.9984 | 0.9984 | 0.9985 | | | | | | | |
| Batch4 | 0.9986 | 0.9985 | 0.9985 | 0.9985 | | | | | | |
| Batch5 | 0.9982 | 0.9979 | 0.9982 | 0.9980 | 0.9980 | | | | | |
| Batch6 | 0.9985 | 0.9985 | 0.9985 | 0.9985 | 0.9986 | 0.9985 | | | | |
| Batch7 | 0.9985 | 0.9985 | 0.9984 | 0.9984 | 0.9984 | 0.9983 | 0.9983 | | | |
| Batch8 | 0.9984 | 0.9984 | 0.9984 | 0.9985 | 0.9985 | 0.9985 | 0.9985 | 0.9984 | | |
| Batch9 | 0.9981 | 0.9979 | 0.9982 | 0.9982 | 0.9981 | 0.9981 | 0.9982 | 0.9983 | 0.9983 | |
| Batch10 | 0.9984 | 0.9985 | 0.9985 | 0.9986 | 0.9985 | 0.9984 | 0.9985 | 0.9984 | 0.9985 | 0.9984 |
| Batch11 | 0.9986 | 0.9986 | 0.9985 | 0.9986 | 0.9987 | 0.9986 | 0.9986 | 0.9986 | 0.9985 | 0.9985 |
| Batch12 | 0.9986 | 0.9986 | 0.9986 | 0.9986 | 0.9986 | 0.9986 | 0.9986 | 0.9986 | 0.9985 | 0.9986 |
| Batch13 | 0.9985 | 0.9985 | 0.9985 | 0.9986 | 0.9985 | 0.9984 | 0.9985 | 0.9985 | 0.9986 | 0.9985 |
| Batch14 | 0.9984 | 0.9985 | 0.9985 | 0.9985 | 0.9985 | 0.9984 | 0.9985 | 0.9985 | 0.9986 | 0.9985 |
| Batch15 | 0.9982 | 0.9981 | 0.9982 | 0.9982 | 0.9982 | 0.9978 | 0.9982 | 0.9982 | 0.9982 | 0.9979 |
| Batch16 | 0.9984 | 0.9985 | 0.9984 | 0.9984 | 0.9984 | 0.9984 | 0.9985 | 0.9985 | 0.9985 | 0.9984 |
| Batch17 | 0.9983 | 0.9978 | 0.9983 | 0.9983 | 0.9982 | 0.9981 | 0.9982 | 0.9982 | 0.9984 | 0.9981 |
| Batch18 | 0.9983 | 0.9982 | 0.9983 | 0.9983 | 0.9983 | 0.9981 | 0.9982 | 0.9983 | 0.9984 | 0.9982 |
| Batch19 | 0.9981 | 0.9977 | 0.9980 | 0.9979 | 0.9979 | 0.9979 | 0.9978 | 0.9980 | 0.9981 | 0.9978 |
| Batch20 | 0.9985 | 0.9984 | 0.9985 | 0.9985 | 0.9985 | 0.9983 | 0.9985 | 0.9985 | 0.9985 | 0.9984 |

Table A.5: AMLDatasetResults(CRLBO)-I

| CramerRaoLowerBoundOptimization | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| Batch11 | Batch12 | Batch13 | Batch14 | Batch15 | Batch16 | Batch17 | Batch18 | Batch19 | Batch20 |
| TestingAccuracy | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| 0.9985 | | | | | | | | | |
| 0.9985 | 0.9987 | | | | | | | | |
| 0.9985 | 0.9986 | 0.9982 | | | | | | | |
| 0.9985 | 0.9985 | 0.9982 | 0.9985 | | | | | | |
| 0.9982 | 0.9982 | 0.9977 | 0.9982 | 0.9978 | | | | | |
| 0.9985 | 0.9985 | 0.9982 | 0.9985 | 0.9985 | 0.9983 | | | | |
| 0.9981 | 0.9982 | 0.9978 | 0.9984 | 0.9981 | 0.9978 | 0.9982 | | | |
| 0.9983 | 0.9983 | 0.9979 | 0.9984 | 0.9981 | 0.9979 | 0.9982 | 0.9981 | | |
| 0.9979 | 0.9980 | 0.9977 | 0.9981 | 0.9980 | 0.9976 | 0.9979 | 0.9978 | 0.9979 | |
| 0.9984 | 0.9985 | 0.9983 | 0.9985 | 0.9984 | 0.9981 | 0.9985 | 0.9984 | 0.9985 | 0.9985 |

Table A.6: AMLDatasetResults(CRLBO)-II

***L2Regularizer***

| Training | Batch1 | Batch2 | Batch3 | Batch4 | Batch5 | Batch6 | Batch7 | Batch8 | Batch9 | Batch10 |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | TestingAccuracy | | | | | |
| Batch1 | 0.7043 | | | | | | | | | |
| Batch2 | 0.8695 | 0.8777 | | | | | | | | |
| Batch3 | 0.5981 | 0.6051 | 0.6105 | | | | | | | |
| Batch4 | 0.8330 | 0.8384 | 0.8382 | 0.8362 | | | | | | |
| Batch5 | 0.6516 | 0.6532 | 0.6403 | 0.6519 | 0.6387 | | | | | |
| Batch6 | 0.8575 | 0.8653 | 0.8694 | 0.8644 | 0.8656 | 0.8795 | | | | |
| Batch7 | 0.7580 | 0.7576 | 0.7740 | 0.7662 | 0.7585 | 0.7694 | 0.7629 | | | |
| Batch8 | 0.8125 | 0.8163 | 0.8053 | 0.8197 | 0.8120 | 0.8195 | 0.8231 | 0.8127 | | |
| Batch9 | 0.9156 | 0.9169 | 0.9162 | 0.9145 | 0.9156 | 0.9182 | 0.9166 | 0.9172 | 0.9166 | |
| Batch10 | 0.8199 | 0.8104 | 0.8158 | 0.8135 | 0.8128 | 0.8063 | 0.8191 | 0.8076 | 0.8103 | 0.8098 |
| Batch11 | 0.5025 | 0.5315 | 0.5240 | 0.5153 | 0.5096 | 0.5152 | 0.5411 | 0.5220 | 0.5294 | 0.5265 |
| Batch12 | 0.7581 | 0.7689 | 0.7599 | 0.7548 | 0.7782 | 0.7625 | 0.7675 | 0.7791 | 0.7666 | 0.7786 |
| Batch13 | 0.7794 | 0.7787 | 0.7689 | 0.7813 | 0.7833 | 0.7739 | 0.7780 | 0.7774 | 0.7789 | 0.7754 |
| Batch14 | 0.7955 | 0.8130 | 0.8112 | 0.8033 | 0.8051 | 0.8094 | 0.8230 | 0.8071 | 0.8095 | 0.8100 |
| Batch15 | 0.8047 | 0.8090 | 0.7965 | 0.8006 | 0.8072 | 0.8001 | 0.8167 | 0.8081 | 0.8020 | 0.8206 |
| Batch16 | 0.7218 | 0.7435 | 0.7513 | 0.7337 | 0.7253 | 0.7396 | 0.7526 | 0.7289 | 0.7363 | 0.7424 |
| Batch17 | 0.9102 | 0.9108 | 0.9108 | 0.9104 | 0.9148 | 0.9103 | 0.9087 | 0.9088 | 0.9103 | 0.9108 |
| Batch18 | 0.9133 | 0.9139 | 0.9149 | 0.9132 | 0.9125 | 0.9146 | 0.9132 | 0.9130 | 0.9139 | 0.9155 |
| Batch19 | 0.8228 | 0.8313 | 0.8376 | 0.8430 | 0.8357 | 0.8332 | 0.8437 | 0.8422 | 0.8373 | 0.8437 |
| Batch20 | 0.7413 | 0.7370 | 0.7456 | 0.7287 | 0.7243 | 0.7513 | 0.7369 | 0.7454 | 0.7409 | 0.7502 |

Table A.7: CreditDefualtDatasetResults(L2)-I

***L2Regularizer***

| Batch11 | Batch12 | Batch13 | Batch14 | Batch15 | Batch16 | Batch17 | Batch18 | Batch19 | Batch20 |
|---|---|---|---|---|---|---|---|---|---|
| TestingAccuracy | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| 0.5238 | | | | | | | | | |
| 0.7709 | 0.7752 | | | | | | | | |
| 0.7747 | 0.7681 | 0.7898 | | | | | | | |
| 0.8045 | 0.7957 | 0.8069 | 0.8090 | | | | | | |
| 0.8070 | 0.8061 | 0.8102 | 0.8000 | 0.7943 | | | | | |
| 0.7319 | 0.7260 | 0.7320 | 0.7499 | 0.7243 | 0.7409 | | | | |
| 0.9166 | 0.9103 | 0.9105 | 0.9071 | 0.9108 | 0.9124 | 0.9090 | | | |
| 0.9159 | 0.9113 | 0.9114 | 0.9143 | 0.9135 | 0.9146 | 0.9169 | 0.9139 | | |
| 0.8374 | 0.8431 | 0.8486 | 0.8341 | 0.8363 | 0.8367 | 0.8316 | 0.8366 | 0.8384 | |
| 0.7391 | 0.7427 | 0.7513 | 0.7430 | 0.7468 | 0.7427 | 0.7502 | 0.7443 | 0.7249 | 0.7370 |

Table A.8: CreditDefualtDatasetResults(L2)-II

***SGD***

| Training | Batch1 | Batch2 | Batch3 | Batch4 | Batch5 | Batch6 | Batch7 | Batch8 | Batch9 | Batch10 |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | TestingAccuracy | | | | | |
| Batch1 | 0.7755 | | | | | | | | | |
| Batch2 | 0.8422 | 0.8366 | | | | | | | | |
| Batch3 | 0.8488 | 0.8639 | 0.8499 | | | | | | | |
| Batch4 | 0.6437 | 0.6433 | 0.6473 | 0.6516 | | | | | | |
| Batch5 | 0.9008 | 0.9024 | 0.8951 | 0.8971 | 0.9015 | | | | | |
| Batch6 | 0.6254 | 0.6310 | 0.6268 | 0.6339 | 0.6268 | 0.6295 | | | | |
| Batch7 | 0.6824 | 0.6857 | 0.6694 | 0.6803 | 0.6903 | 0.6859 | 0.6806 | | | |
| Batch8 | 0.5867 | 0.5725 | 0.5816 | 0.5922 | 0.5903 | 0.6002 | 0.5826 | 0.5898 | | |
| Batch9 | 0.8717 | 0.8733 | 0.8754 | 0.8774 | 0.8741 | 0.8854 | 0.8780 | 0.8754 | 0.8735 | |
| Batch10 | 0.8728 | 0.8822 | 0.8796 | 0.8840 | 0.8759 | 0.8862 | 0.8839 | 0.8841 | 0.8831 | 0.8896 |
| Batch11 | 0.8090 | 0.8046 | 0.8087 | 0.8081 | 0.8111 | 0.8135 | 0.8044 | 0.8066 | 0.8102 | 0.8106 |
| Batch12 | 0.8488 | 0.8560 | 0.8559 | 0.8626 | 0.8541 | 0.8610 | 0.8680 | 0.8617 | 0.8552 | 0.8623 |
| Batch13 | 0.3981 | 0.3990 | 0.3867 | 0.4071 | 0.3922 | 0.4014 | 0.4110 | 0.4099 | 0.3957 | 0.4149 |
| Batch14 | 0.5260 | 0.5038 | 0.5468 | 0.5199 | 0.5369 | 0.5176 | 0.5088 | 0.5309 | 0.5414 | 0.5170 |
| Batch15 | 0.2823 | 0.2872 | 0.2748 | 0.2770 | 0.2788 | 0.2805 | 0.2783 | 0.2745 | 0.2809 | 0.2737 |
| Batch16 | 0.5431 | 0.5205 | 0.5465 | 0.5354 | 0.5365 | 0.5451 | 0.5327 | 0.5347 | 0.5487 | 0.5471 |
| Batch17 | 0.4887 | 0.5011 | 0.5012 | 0.5049 | 0.4912 | 0.4916 | 0.5202 | 0.5111 | 0.5095 | 0.5041 |
| Batch18 | 0.8957 | 0.8924 | 0.8935 | 0.8940 | 0.8868 | 0.8991 | 0.8954 | 0.8943 | 0.8920 | 0.8955 |
| Batch19 | 0.8967 | 0.8935 | 0.8968 | 0.8981 | 0.8985 | 0.8974 | 0.8935 | 0.8998 | 0.9003 | 0.8940 |
| Batch20 | 0.6766 | 0.6935 | 0.6713 | 0.6700 | 0.6844 | 0.6745 | 0.6822 | 0.6676 | 0.6673 | 0.6874 |

Table A.9: CreditDefualtDatasetResults(SGD)-I

***SGD***

| Batch11 | Batch12 | Batch13 | Batch14 | Batch15 | Batch16 | Batch17 | Batch18 | Batch19 | Batch20 |
|---|---|---|---|---|---|---|---|---|---|
| TestingAccuracy | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| 0.8015 | | | | | | | | | |
| 0.8518 | 0.8546 | | | | | | | | |
| 0.3977 | 0.4068 | 0.3962 | | | | | | | |
| 0.5225 | 0.5441 | 0.5245 | 0.5255 | | | | | | |
| 0.2794 | 0.2802 | 0.2800 | 0.2723 | 0.2822 | | | | | |
| 0.5277 | 0.5503 | 0.5395 | 0.5415 | 0.5425 | 0.5451 | | | | |
| 0.5095 | 0.4951 | 0.5039 | 0.5092 | 0.4972 | 0.4916 | 0.5067 | | | |
| 0.8925 | 0.8944 | 0.8988 | 0.8927 | 0.8920 | 0.8896 | 0.8922 | 0.8995 | | |
| 0.8967 | 0.8998 | 0.8960 | 0.8990 | 0.9002 | 0.8981 | 0.9040 | 0.8945 | 0.9010 | |
| 0.6811 | 0.6739 | 0.6734 | 0.6845 | 0.6706 | 0.6827 | 0.6758 | 0.6525 | 0.6684 | 0.6966 |

Table A.10: CreditDefualtDatasetResults(SGD)-II

***CramerRaoLowerBoundOptimization***

| Training | Batch1 | Batch2 | Batch3 | Batch4 | Batch5 | Batch6 | Batch7 | Batch8 | Batch9 | Batch10 |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | TestingAccuracy | | | | | |
| Batch1 | 0.9067 | | | | | | | | | |
| Batch2 | 0.9083 | 0.9060 | | | | | | | | |
| Batch3 | 0.9051 | 0.9035 | 0.9079 | | | | | | | |
| Batch4 | 0.9070 | 0.9057 | 0.9090 | 0.8975 | | | | | | |
| Batch5 | 0.9041 | 0.9040 | 0.9056 | 0.8948 | 0.9006 | | | | | |
| Batch6 | 0.9080 | 0.9063 | 0.9096 | 0.9017 | 0.9049 | 0.9106 | | | | |
| Batch7 | 0.9064 | 0.9057 | 0.9093 | 0.8990 | 0.9043 | 0.9099 | 0.9120 | | | |
| Batch8 | 0.9038 | 0.9040 | 0.9060 | 0.8945 | 0.9009 | 0.9064 | 0.9097 | 0.9010 | | |
| Batch9 | 0.9041 | 0.9047 | 0.9063 | 0.8961 | 0.9017 | 0.9063 | 0.9097 | 0.9017 | 0.9094 | |
| Batch10 | 0.9032 | 0.9047 | 0.9063 | 0.8964 | 0.9017 | 0.9083 | 0.9097 | 0.9030 | 0.9094 | 0.9114 |
| Batch11 | 0.9045 | 0.9038 | 0.9073 | 0.8961 | 0.9019 | 0.9079 | 0.9107 | 0.9017 | 0.9107 | 0.9101 |
| Batch12 | 0.9045 | 0.9053 | 0.9084 | 0.8972 | 0.9017 | 0.9090 | 0.9101 | 0.9026 | 0.9094 | 0.9097 |
| Batch13 | 0.9035 | 0.9044 | 0.9066 | 0.8961 | 0.9017 | 0.9064 | 0.9107 | 0.9020 | 0.9104 | 0.9099 |
| Batch14 | 0.9074 | 0.9069 | 0.9104 | 0.9004 | 0.9062 | 0.9106 | 0.9123 | 0.9062 | 0.9114 | 0.9147 |
| Batch15 | 0.9035 | 0.9024 | 0.9044 | 0.8932 | 0.9006 | 0.9056 | 0.9083 | 0.8999 | 0.9084 | 0.9076 |
| Batch16 | 0.9016 | 0.9037 | 0.9050 | 0.8932 | 0.9006 | 0.9069 | 0.9081 | 0.9004 | 0.9081 | 0.9084 |
| Batch17 | 0.9051 | 0.9045 | 0.9069 | 0.8954 | 0.9022 | 0.9085 | 0.9117 | 0.9020 | 0.9107 | 0.9107 |
| Batch18 | 0.9035 | 0.9034 | 0.9048 | 0.8961 | 0.9010 | 0.9045 | 0.9088 | 0.8997 | 0.9094 | 0.9089 |
| Batch19 | 0.9051 | 0.9047 | 0.9083 | 0.8974 | 0.9026 | 0.9064 | 0.9101 | 0.9017 | 0.9094 | 0.9114 |
| Batch20 | 0.9074 | 0.9050 | 0.9082 | 0.8974 | 0.9023 | 0.9077 | 0.9110 | 0.9036 | 0.9094 | 0.9114 |

Table A.11: CreditDefualtDatasetResults(CRLBO)-I

***CramerRaoLowerBoundOptimization***

| Batch11 | Batch12 | Batch13 | Batch14 | Batch15 | Batch16 | Batch17 | Batch18 | Batch19 | Batch20 |
|---|---|---|---|---|---|---|---|---|---|
| TestingAccuracy | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| 0.9367 | | | | | | | | | |
| 0.9361 | 0.9089 | | | | | | | | |
| 0.9364 | 0.9096 | 0.9041 | | | | | | | |
| 0.9361 | 0.9125 | 0.9064 | 0.9066 | | | | | | |
| 0.9371 | 0.9074 | 0.9018 | 0.9023 | 0.9078 | | | | | |
| 0.9371 | 0.9089 | 0.9016 | 0.9034 | 0.9078 | 0.9063 | | | | |
| 0.9364 | 0.9106 | 0.9035 | 0.9063 | 0.9102 | 0.9092 | 0.9023 | | | |
| 0.9374 | 0.9074 | 0.9029 | 0.9033 | 0.9088 | 0.9076 | 0.8988 | 0.9024 | | |
| 0.9364 | 0.9100 | 0.9045 | 0.9050 | 0.9103 | 0.9086 | 0.9017 | 0.9031 | 0.9083 | |
| 0.9361 | 0.9100 | 0.9058 | 0.9057 | 0.9109 | 0.9086 | 0.9033 | 0.9037 | 0.9093 | 0.9119 |

Table A.12: CreditDefualtDatasetResults(CRLBO)-II

***L2Regularizer***

| Training | Batch1 | Batch2 | Batch3 | Batch4 | Batch5 | Batch6 | Batch7 | Batch8 | Batch9 | Batch10 |
|---|---|---|---|---|---|---|---|---|---|---|
| | TestingAccuracy | | | | | | | | | |
| Batch1 | 0.8217 | | | | | | | | | |
| Batch2 | 0.0086 | 1.0000 | | | | | | | | |
| Batch3 | 0.9831 | 0.9983 | 0.9983 | | | | | | | |
| Batch4 | 0.9983 | 0.9983 | 0.9983 | 0.8431 | | | | | | |
| Batch5 | 0.9789 | 0.0017 | 0.0017 | 0.0017 | 0.9582 | | | | | |
| Batch6 | 0.0055 | 0.0017 | 0.0017 | 0.1700 | 0.0017 | 0.6430 | | | | |
| Batch7 | 0.0028 | 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.8329 | 0.9997 | | | |
| Batch8 | 0.9855 | 0.9983 | 0.9983 | 0.9983 | 0.9983 | 0.9983 | 0.5141 | 0.3903 | | |
| Batch9 | 0.9907 | 0.9983 | 0.9983 | 0.9983 | 0.9983 | 0.9983 | 0.9983 | 0.9983 | 0.9983 | |
| Batch10 | 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.0031 | 0.0045 | 0.0163 | 0.2884 | 0.8973 |
| Batch11 | 0.9668 | 0.9983 | 0.9983 | 0.9983 | 0.9983 | 0.9983 | 0.9983 | 0.9983 | 0.9983 | 0.9983 |
| Batch12 | 0.9969 | 0.9983 | 0.9983 | 0.9983 | 0.9983 | 0.9979 | 0.9983 | 0.9976 | 0.9979 | 0.9972 |
| Batch13 | 0.0211 | 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.0017 |
| Batch14 | 0.0197 | 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.0017 |
| Batch15 | 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.0017 |
| Batch16 | 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.0017 |
| Batch17 | 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.0017 |
| Batch18 | 0.9983 | 0.0017 | 0.0017 | 0.0017 | 0.9983 | 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.0017 |
| Batch19 | 0.0017 | 0.9993 | 0.9986 | 0.9983 | 0.0017 | 0.9983 | 0.9983 | 0.9983 | 0.9983 | 0.9983 |
| Batch20 | 0.0066 | 1.0000 | 0.9983 | 0.9983 | 0.0017 | 0.9983 | 0.9983 | 0.9983 | 0.9983 | 0.9983 |

Table A.13: CreditCardFraudDatasetResults(L2)-I

***L2Regularizer***

| Batch11 | Batch12 | Batch13 | Batch14 | Batch15 | Batch16 | Batch17 | Batch18 | Batch19 | Batch20 |
|---|---|---|---|---|---|---|---|---|---|
| TestingAccuracy | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| 0.7469 | | | | | | | | | |
| 0.9948 | 0.9578 | | | | | | | | |
| 0.0017 | 0.3430 | 1.0000 | | | | | | | |
| 0.0017 | 0.0017 | 0.1447 | 1.0000 | | | | | | |
| 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.2072 | | | | | |
| 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.0031 | 0.9289 | | | | |
| 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.2922 | 0.9527 | | | |
| 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.9983 | 0.9983 | 0.9983 | 0.7637 | | |
| 0.9983 | 0.9983 | 0.9983 | 0.9983 | 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.6682 | |
| 0.9983 | 0.9983 | 0.9983 | 0.9983 | 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.2984 | 0.9993 |

Table A.14: CreditCardFraudDatasetResults(L2)-II

***SGD***

| Training | Batch1 | Batch2 | Batch3 | Batch4 | Batch5 | Batch6 | Batch7 | Batch8 | Batch9 | Batch10 |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | TestingAccuracy | | | | | |
| Batch1 | 0.9983 | | | | | | | | | |
| Batch2 | 0.9983 | 0.6561 | | | | | | | | |
| Batch3 | 0.9945 | 0.9983 | 0.9917 | | | | | | | |
| Batch4 | 0.9848 | 0.9983 | 0.9983 | 0.7376 | | | | | | |
| Batch5 | 0.9872 | 0.0017 | 0.0017 | 0.0017 | 0.3059 | | | | | |
| Batch6 | 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.5166 | | | | |
| Batch7 | 0.9983 | 0.9983 | 0.9983 | 0.9983 | 0.9983 | 0.8321 | 0.8667 | | | |
| Batch8 | 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.0028 | 0.4438 | 0.9281 | | |
| Batch9 | 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.7821 | |
| Batch10 | 0.9976 | 0.9983 | 0.9983 | 0.9983 | 0.9983 | 0.9983 | 0.9983 | 0.9983 | 0.9979 | 0.1748 |
| Batch11 | 0.9800 | 0.9983 | 0.9983 | 0.9983 | 0.9983 | 0.9983 | 0.9983 | 0.9983 | 0.9983 | 0.9983 |
| Batch12 | 0.0024 | 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.0017 |
| Batch13 | 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.0021 | 0.0017 | 0.508 | 0.9489 | 0.9779 |
| Batch14 | 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.0017 |
| Batch15 | 0.9914 | 0.0017 | 0.0017 | 0.0017 | 0.9983 | 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.0017 |
| Batch16 | 0.0221 | 1.0000 | 0.9983 | 0.9983 | 0.0086 | 0.9983 | 0.9983 | 0.9983 | 0.9983 | 0.9983 |
| Batch17 | 0.9972 | 0.0017 | 0.0017 | 0.0017 | 0.9979 | 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.0017 |
| Batch18 | 0.0052 | 0.0014 | 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.0017 |
| Batch19 | 0.0017 | 0.0918 | 0.0000 | 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.0017 |
| Batch20 | 0.0076 | 0.1678 | 0.0038 | 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.0017 |

Table A.15: CreditCardFraudDatasetResults(SGD)-I

***SGD***

| Batch11 | Batch12 | Batch13 | Batch14 | Batch15 | Batch16 | Batch17 | Batch18 | Batch19 | Batch20 |
|---|---|---|---|---|---|---|---|---|---|
| TestingAccuracy | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| 0.9189 | | | | | | | | | |
| 0.0017 | 0.8819 | | | | | | | | |
| 0.9897 | 0.9983 | 0.9983 | | | | | | | |
| 0.0017 | 0.0017 | 0.3243 | 0.9965 | | | | | | |
| 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.9731 | | | | | |
| 0.9983 | 0.9983 | 0.9983 | 0.9983 | 0.9372 | 0.999 | | | | |
| 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.9972 | 0.9979 | 0.9983 | | | |
| 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.781 | | |
| 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.4041 | 1 | |
| 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.0017 | 0.7414 |

Table A.16: CreditCardFraudDatasetResults(SGD)-II

***L2Regularizer***

| Training | Batch1 | Batch2 | Batch3 | Batch4 | Batch5 | Batch6 | Batch7 | Batch8 | Batch9 | Batch10 |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | TestingAccuracy | | | | | |
| Batch1 | 0.9978 | | | | | | | | | |
| Batch2 | 0.9967 | 0.9987 | | | | | | | | |
| Batch3 | 0.9987 | 0.9987 | 0.9987 | | | | | | | |
| Batch4 | 0.9987 | 0.9987 | 0.9987 | 0.9987 | | | | | | |
| Batch5 | 0.9987 | 0.9987 | 0.9987 | 0.9987 | 0.9987 | | | | | |
| Batch6 | 0.9977 | 0.9989 | 0.9989 | 0.9989 | 0.9987 | 0.9988 | | | | |
| Batch7 | 0.9988 | 0.9983 | 0.9983 | 0.9981 | 0.9989 | 0.998 | 0.998 | | | |
| Batch8 | 0.9987 | 0.9987 | 0.9987 | 0.9987 | 0.9987 | 0.9987 | 0.9987 | 0.9987 | | |
| Batch9 | 0.9987 | 0.9987 | 0.9987 | 0.9987 | 0.9987 | 0.9986 | 0.9981 | 0.9983 | 0.9985 | |
| Batch10 | 0.9930 | 0.9986 | 0.9985 | 0.9986 | 0.9972 | 0.9986 | 0.9981 | 0.9951 | 0.9975 | 0.9985 |
| Batch11 | 0.9987 | 0.9987 | 0.9987 | 0.9987 | 0.9987 | 0.9987 | 0.9987 | 0.9987 | 0.9987 | 0.9987 |
| Batch12 | 0.9976 | 0.9988 | 0.9988 | 0.9988 | 0.9985 | 0.9983 | 0.9963 | 0.9919 | 0.9964 | 0.9987 |
| Batch13 | 0.9972 | 0.9988 | 0.9989 | 0.9989 | 0.9984 | 0.9983 | 0.9956 | 0.9887 | 0.9947 | 0.9984 |
| Batch14 | 0.9988 | 0.9988 | 0.9989 | 0.9989 | 0.9988 | 0.9989 | 0.9988 | 0.9988 | 0.9988 | 0.9988 |
| Batch15 | 0.9987 | 0.9989 | 0.9988 | 0.9989 | 0.9987 | 0.9989 | 0.9986 | 0.9958 | 0.9975 | 0.9984 |
| Batch16 | 0.9985 | 0.9987 | 0.9987 | 0.9987 | 0.9986 | 0.998 | 0.9956 | 0.9901 | 0.9957 | 0.9981 |
| Batch17 | 0.9887 | 0.9983 | 0.9983 | 0.9984 | 0.9953 | 0.9977 | 0.995 | 0.9872 | 0.9937 | 0.9979 |
| Batch18 | 0.9984 | 0.9990 | 0.9991 | 0.9991 | 0.9989 | 0.9988 | 0.9975 | 0.9954 | 0.9983 | 0.999 |
| Batch19 | 0.9987 | 0.9987 | 0.9987 | 0.9987 | 0.9987 | 0.9987 | 0.9987 | 0.9987 | 0.9987 | 0.9987 |
| Batch20 | 0.9893 | 0.9985 | 0.9984 | 0.9984 | 0.9958 | 0.9986 | 0.9984 | 0.9985 | 0.9984 | 0.9984 |

Table A.17: PaysimDatasetResults(L2)-I

***L2Regularizer***

| Batch11 | Batch12 | Batch13 | Batch14 | Batch15 | Batch16 | Batch17 | Batch18 | Batch19 | Batch20 |
|---|---|---|---|---|---|---|---|---|---|
| TestingAccuracy | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| 0.9987 | | | | | | | | | |
| 0.9986 | 0.9987 | | | | | | | | |
| 0.9983 | 0.9986 | 0.9988 | | | | | | | |
| 0.9988 | 0.9989 | 0.9988 | 0.9988 | | | | | | |
| 0.9984 | 0.9987 | 0.9989 | 0.9986 | 0.9988 | | | | | |
| 0.9981 | 0.9983 | 0.9984 | 0.9983 | 0.9987 | 0.9987 | | | | |
| 0.9978 | 0.9982 | 0.9983 | 0.9984 | 0.997 | 0.9975 | 0.9978 | | | |
| 0.999 | 0.9991 | 0.999 | 0.9989 | 0.999 | 0.9991 | 0.999 | 0.9991 | | |
| 0.9987 | 0.9987 | 0.9987 | 0.9987 | 0.9987 | 0.9987 | 0.9987 | 0.9987 | 0.9987 | |
| 0.9986 | 0.9985 | 0.9985 | 0.9987 | 0.9972 | 0.9978 | 0.9978 | 0.9981 | 0.9981 | 0.9985 |

Table A.18: PaysimDatasetResults(L2)-II

***SGD***

| Training | Batch1 | Batch2 | Batch3 | Batch4 | Batch5 | Batch6 | Batch7 | Batch8 | Batch9 | Batch10 |
|---|---|---|---|---|---|---|---|---|---|---|
| | TestingAccuracy | | | | | | | | | |
| Batch1 | 0.9987 | | | | | | | | | |
| Batch2 | 0.9987 | 0.9987 | | | | | | | | |
| Batch3 | 0.9877 | 0.9982 | 0.9982 | | | | | | | |
| Batch4 | 0.9972 | 0.9989 | 0.9989 | 0.9990 | | | | | | |
| Batch5 | 0.9987 | 0.9987 | 0.9987 | 0.9987 | 0.9987 | | | | | |
| Batch6 | 0.9987 | 0.9987 | 0.9987 | 0.9987 | 0.9987 | 0.9988 | | | | |
| Batch7 | 0.9987 | 0.9987 | 0.9987 | 0.9987 | 0.9987 | 0.9987 | 0.9987 | | | |
| Batch8 | 0.9987 | 0.9981 | 0.9982 | 0.9981 | 0.9987 | 0.998 | 0.9982 | 0.9981 | | |
| Batch9 | 0.9983 | 0.9987 | 0.9987 | 0.9987 | 0.9987 | 0.9986 | 0.9985 | 0.9987 | 0.9987 | |
| Batch10 | 0.9987 | 0.9987 | 0.9987 | 0.9987 | 0.9987 | 0.9987 | 0.9987 | 0.9987 | 0.9987 | 0.9987 |
| Batch11 | 0.9988 | 0.9988 | 0.9989 | 0.9989 | 0.9988 | 0.9989 | 0.9988 | 0.9987 | 0.9987 | 0.9986 |
| Batch12 | 0.9989 | 0.9990 | 0.9991 | 0.9991 | 0.9989 | 0.9991 | 0.999 | 0.999 | 0.9991 | 0.999 |
| Batch13 | 0.9987 | 0.9984 | 0.9984 | 0.9984 | 0.9987 | 0.9983 | 0.998 | 0.9975 | 0.998 | 0.9976 |
| Batch14 | 0.9987 | 0.9987 | 0.9988 | 0.9988 | 0.9987 | 0.9981 | 0.9955 | 0.9926 | 0.9965 | 0.9986 |
| Batch15 | 0.9969 | 0.9973 | 0.9981 | 0.9976 | 0.9983 | 0.9981 | 0.9919 | 0.9869 | 0.9886 | 0.9919 |
| Batch16 | 0.9982 | 0.9987 | 0.9987 | 0.9987 | 0.9987 | 0.9987 | 0.9987 | 0.9987 | 0.9987 | 0.9987 |
| Batch17 | 0.4887 | 0.5011 | 0.5012 | 0.5049 | 0.4912 | 0.4916 | 0.5202 | 0.5111 | 0.5095 | 0.5041 |
| Batch18 | 0.4887 | 0.5011 | 0.5012 | 0.5049 | 0.4912 | 0.4916 | 0.5202 | 0.5111 | 0.5095 | 0.5041 |
| Batch19 | 0.9844 | 0.9981 | 0.9980 | 0.9980 | 0.9924 | 0.9982 | 0.9976 | 0.9974 | 0.9975 | 0.9975 |
| Batch20 | 0.9987 | 0.9987 | 0.9987 | 0.9987 | 0.9987 | 0.9987 | 0.9987 | 0.9986 | 0.9987 | 0.9987 |

Table A.19: PaysimDatasetResults(SGD)-I

***SGD***

| Batch11 | Batch12 | Batch13 | Batch14 | Batch15 | Batch16 | Batch17 | Batch18 | Batch19 | Batch20 |
|---|---|---|---|---|---|---|---|---|---|
| TestingAccuracy | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| 0.9985 | | | | | | | | | |
| 0.999 | 0.999 | | | | | | | | |
| 0.9976 | 0.9979 | 0.998 | | | | | | | |
| 0.9984 | 0.9987 | 0.9987 | 0.9987 | | | | | | |
| 0.9932 | 0.9957 | 0.9973 | 0.9959 | 0.9984 | | | | | |
| 0.9987 | 0.9987 | 0.9987 | 0.9987 | 0.9987 | 0.9987 | | | | |
| 0.5095 | 0.4951 | 0.5039 | 0.5092 | 0.4972 | 0.4916 | 0.5067 | | | |
| 0.5095 | 0.4951 | 0.5039 | 0.5092 | 0.4972 | 0.4916 | 0.5067 | 0.9972 | | |
| 0.9977 | 0.9978 | 0.998 | 0.9982 | 0.9953 | 0.9969 | 0.9966 | 0.9987 | 0.9987 | |
| 0.9987 | 0.9987 | 0.9987 | 0.9987 | 0.9987 | 0.9987 | 0.9987 | 0.9987 | 0.9987 | 0.9987 |

Table A.20: PaysimDatasetResults(SGD)-II

***CramerRaoLowerBoundOptimization***

| Training | Batch1 | Batch2 | Batch3 | Batch4 | Batch5 | Batch6 | Batch7 | Batch8 | Batch9 | Batch10 |
|---|---|---|---|---|---|---|---|---|---|---|
| | TestingAccuracy | | | | | | | | | |
| Batch1 | 0.9989 | | | | | | | | | |
| Batch2 | 0.9619 | 0.9982 | | | | | | | | |
| Batch3 | 0.9679 | 0.9982 | 0.9982 | | | | | | | |
| Batch4 | 0.9670 | 0.9985 | 0.9983 | 0.9986 | | | | | | |
| Batch5 | 0.9986 | 0.9991 | 0.9991 | 0.9992 | 0.999 | | | | | |
| Batch6 | 0.9711 | 0.9986 | 0.9984 | 0.9987 | 0.9936 | 0.999 | | | | |
| Batch7 | 0.9571 | 0.9977 | 0.9977 | 0.9980 | 0.9875 | 0.9985 | 0.998 | | | |
| Batch8 | 0.9908 | 0.9968 | 0.9971 | 0.9965 | 0.9944 | 0.9972 | 0.996 | 0.9965 | | |
| Batch9 | 0.9897 | 0.9988 | 0.9989 | 0.9990 | 0.9972 | 0.999 | 0.9988 | 0.9961 | 0.9981 | |
| Batch10 | 0.9802 | 0.9987 | 0.9988 | 0.9989 | 0.996 | 0.999 | 0.9986 | 0.9952 | 0.9977 | 0.9982 |
| Batch11 | 0.9664 | 0.9981 | 0.9981 | 0.9984 | 0.9906 | 0.9988 | 0.9983 | 0.9945 | 0.9969 | 0.9978 |
| Batch12 | 0.9746 | 0.9983 | 0.9985 | 0.9986 | 0.993 | 0.9988 | 0.9984 | 0.9951 | 0.9974 | 0.998 |
| Batch13 | 0.9798 | 0.9981 | 0.9985 | 0.9986 | 0.9936 | 0.9988 | 0.9984 | 0.9956 | 0.9976 | 0.9981 |
| Batch14 | 0.9377 | 0.9960 | 0.9962 | 0.9963 | 0.9757 | 0.9975 | 0.9967 | 0.9929 | 0.995 | 0.9962 |
| Batch15 | 0.9897 | 0.9990 | 0.9991 | 0.9992 | 0.998 | 0.9992 | 0.9989 | 0.9956 | 0.9977 | 0.9981 |
| Batch16 | 0.9924 | 0.9991 | 0.9992 | 0.9992 | 0.9983 | 0.9992 | 0.999 | 0.9958 | 0.9977 | 0.998 |
| Batch17 | 0.9970 | 0.9991 | 0.9991 | 0.9992 | 0.9988 | 0.9992 | 0.9991 | 0.9979 | 0.9982 | 0.9982 |
| Batch18 | 0.9619 | 0.9981 | 0.9980 | 0.9983 | 0.9895 | 0.9987 | 0.9983 | 0.9939 | 0.9964 | 0.9972 |
| Batch19 | 0.9840 | 0.9989 | 0.9989 | 0.9991 | 0.997 | 0.9991 | 0.9987 | 0.9949 | 0.9971 | 0.9976 |
| Batch20 | 0.9923 | 0.9991 | 0.9992 | 0.9993 | 0.9984 | 0.9992 | 0.9991 | 0.996 | 0.998 | 0.9982 |

Table A.21: PaysimDatasetResults(CRLBO)-I

***CramerRaoLowerBoundOptimization***

| Batch11 | Batch12 | Batch13 | Batch14 | Batch15 | Batch16 | Batch17 | Batch18 | Batch19 | Batch20 |
|---|---|---|---|---|---|---|---|---|---|
| TestingAccuracy | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| 0.998 | | | | | | | | | |
| 0.9982 | 0.9983 | | | | | | | | |
| 0.9981 | 0.9983 | 0.9982 | | | | | | | |
| 0.9968 | 0.9967 | 0.997 | 0.9981 | | | | | | |
| 0.9983 | 0.9984 | 0.9984 | 0.9986 | 0.9985 | | | | | |
| 0.9982 | 0.9983 | 0.9983 | 0.9985 | 0.9987 | 0.9989 | | | | |
| 0.9984 | 0.9984 | 0.9984 | 0.9986 | 0.9989 | 0.9991 | 0.9991 | | | |
| 0.9975 | 0.9977 | 0.9978 | 0.9984 | 0.9943 | 0.9962 | 0.9965 | 0.9968 | | |
| 0.9979 | 0.9981 | 0.9981 | 0.9984 | 0.9979 | 0.9983 | 0.9986 | 0.9987 | 0.9987 | |
| 0.9984 | 0.9985 | 0.9985 | 0.9986 | 0.9988 | 0.999 | 0.9991 | 0.9991 | 0.9991 | 0.999 |

Table A.22: PaysimDatasetResults(CRLBO)-II

Table A.23: AMLAugmented40Batches(CRLBO)-I

| CramerRaoLowerBoundOptimization | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | Batch1 | Batch2 | Batch3 | Batch4 | Batch5 | Batch6 | Batch7 | Batch8 | Batch9 | Batch10 |
| Batch1 | 0.9983 | | | | | | | | | |
| Batch2 | 0.9985 | 0.9983 | | | | | | | | |
| Batch3 | 0.9984 | 0.9984 | 0.9985 | | | | | | | |
| Batch4 | 0.9986 | 0.9985 | 0.9985 | 0.9985 | | | | | | |
| Batch5 | 0.9982 | 0.9979 | 0.9982 | 0.9980 | 0.9980 | | | | | |
| Batch6 | 0.9985 | 0.9985 | 0.9985 | 0.9985 | 0.9986 | 0.9985 | | | | |
| Batch7 | 0.9985 | 0.9985 | 0.9984 | 0.9984 | 0.9984 | 0.9983 | 0.9983 | | | |
| Batch8 | 0.9984 | 0.9984 | 0.9984 | 0.9985 | 0.9985 | 0.9985 | 0.9985 | 0.9984 | | |
| Batch9 | 0.9981 | 0.9979 | 0.9982 | 0.9982 | 0.9981 | 0.9981 | 0.9982 | 0.9983 | 0.9983 | |
| Batch10 | 0.9984 | 0.9985 | 0.9985 | 0.9986 | 0.9985 | 0.9984 | 0.9985 | 0.9984 | 0.9985 | 0.9984 |
| Batch11 | 0.9986 | 0.9986 | 0.9985 | 0.9986 | 0.9987 | 0.9986 | 0.9986 | 0.9986 | 0.9985 | 0.9985 |
| Batch12 | 0.9986 | 0.9986 | 0.9986 | 0.9986 | 0.9986 | 0.9986 | 0.9986 | 0.9986 | 0.9985 | 0.9986 |
| Batch13 | 0.9985 | 0.9985 | 0.9985 | 0.9986 | 0.9985 | 0.9984 | 0.9985 | 0.9985 | 0.9986 | 0.9985 |
| Batch14 | 0.9984 | 0.9985 | 0.9985 | 0.9985 | 0.9985 | 0.9984 | 0.9985 | 0.9985 | 0.9986 | 0.9985 |
| Batch15 | 0.9982 | 0.9981 | 0.9982 | 0.9982 | 0.9982 | 0.9978 | 0.9982 | 0.9982 | 0.9982 | 0.9979 |
| Batch16 | 0.9984 | 0.9985 | 0.9984 | 0.9984 | 0.9984 | 0.9984 | 0.9985 | 0.9985 | 0.9985 | 0.9984 |
| Batch17 | 0.9983 | 0.9978 | 0.9983 | 0.9983 | 0.9982 | 0.9981 | 0.9982 | 0.9982 | 0.9984 | 0.9981 |
| Batch18 | 0.9983 | 0.9982 | 0.9983 | 0.9983 | 0.9983 | 0.9981 | 0.9982 | 0.9983 | 0.9984 | 0.9982 |
| Batch19 | 0.9981 | 0.9977 | 0.9980 | 0.9985 | 0.9985 | 0.9985 | 0.9985 | 0.9980 | 0.9981 | 0.9978 |
| Batch20 | 0.9985 | 0.9984 | 0.9982 | 0.9982 | 0.9981 | 0.9981 | 0.9982 | 0.9982 | 0.9982 | 0.9981 |
| Batch21 | 0.9983 | 0.9985 | 0.9985 | 0.9986 | 0.9985 | 0.9984 | 0.9985 | 0.9985 | 0.9986 | 0.9985 |
| Batch22 | 0.9985 | 0.9985 | 0.9985 | 0.9986 | 0.9987 | 0.9986 | 0.9986 | 0.9985 | 0.9986 | 0.9987 |
| Batch23 | 0.9984 | 0.9984 | 0.9986 | 0.9986 | 0.9986 | 0.9986 | 0.9986 | 0.9986 | 0.9986 | 0.9986 |
| Batch24 | 0.9986 | 0.9979 | 0.9985 | 0.9986 | 0.9985 | 0.9984 | 0.9985 | 0.9985 | 0.9986 | 0.9985 |
| Batch25 | 0.9982 | 0.9985 | 0.9985 | 0.9985 | 0.9985 | 0.9984 | 0.9985 | 0.9985 | 0.9985 | 0.9985 |
| Batch26 | 0.9985 | 0.9986 | 0.9982 | 0.9982 | 0.9985 | 0.9985 | 0.9985 | 0.9982 | 0.9982 | 0.9985 |
| Batch27 | 0.9985 | 0.9986 | 0.9984 | 0.9984 | 0.9982 | 0.9981 | 0.9981 | 0.9984 | 0.9984 | 0.9982 |
| Batch28 | 0.9984 | 0.9985 | 0.9983 | 0.9983 | 0.9986 | 0.9985 | 0.9984 | 0.9983 | 0.9983 | 0.9986 |
| Batch29 | 0.9981 | 0.9986 | 0.9986 | 0.9982 | 0.9986 | 0.9987 | 0.9986 | 0.9986 | 0.9982 | 0.9986 |
| Batch30 | 0.9984 | 0.9986 | 0.9985 | 0.9986 | 0.9986 | 0.9986 | 0.9986 | 0.9985 | 0.9986 | 0.9986 |
| Batch31 | 0.9986 | 0.9986 | 0.9985 | 0.9986 | 0.9986 | 0.9985 | 0.9984 | 0.9985 | 0.9986 | 0.9986 |
| Batch32 | 0.9986 | 0.9985 | 0.9982 | 0.9986 | 0.9985 | 0.9985 | 0.9984 | 0.9982 | 0.9986 | 0.9985 |
| Batch33 | 0.9985 | 0.9982 | 0.9984 | 0.9986 | 0.9982 | 0.9982 | 0.9978 | 0.9982 | 0.9986 | 0.9982 |
| Batch34 | 0.9984 | 0.9984 | 0.9982 | 0.9985 | 0.9984 | 0.9984 | 0.9984 | 0.9985 | 0.9986 | 0.9984 |
| Batch35 | 0.9982 | 0.9983 | 0.9986 | 0.9982 | 0.9983 | 0.9982 | 0.9981 | 0.9982 | 0.9982 | 0.9983 |
| Batch36 | 0.9984 | 0.9983 | 0.9985 | 0.9984 | 0.9984 | 0.9984 | 0.9985 | 0.9985 | 0.9985 | 0.9984 |
| Batch37 | 0.9983 | 0.9979 | 0.9985 | 0.9983 | 0.9982 | 0.9981 | 0.9982 | 0.9982 | 0.9984 | 0.9981 |
| Batch38 | 0.9983 | 0.9985 | 0.9982 | 0.9983 | 0.9983 | 0.9981 | 0.9982 | 0.9983 | 0.9984 | 0.9982 |
| Batch39 | 0.9981 | 0.9977 | 0.9984 | 0.9979 | 0.9979 | 0.9979 | 0.9978 | 0.9980 | 0.9981 | 0.9978 |
| Batch40 | 0.9985 | 0.9984 | 0.9982 | 0.9985 | 0.9985 | 0.9983 | 0.9985 | 0.9985 | 0.9985 | 0.9984 |

Table A.24: AMLAugmented40Batches(CRLBO)-II

| CramerRaoLowerBoundOptimization | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| Batch11 | Batch12 | Batch13 | Batch14 | Batch15 | Batch16 | Batch17 | Batch18 | Batch19 | Batch20 |
| TestingAccuracy | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| 0.9985 | | | | | | | | | |
| 0.9985 | 0.9987 | | | | | | | | |
| 0.9985 | 0.9986 | 0.9982 | | | | | | | |
| 0.9985 | 0.9985 | 0.9982 | 0.9985 | | | | | | |
| 0.9982 | 0.9982 | 0.9977 | 0.9982 | 0.9978 | | | | | |
| 0.9985 | 0.9985 | 0.9982 | 0.9985 | 0.9985 | 0.9983 | | | | |
| 0.9981 | 0.9982 | 0.9978 | 0.9984 | 0.9981 | 0.9978 | 0.9982 | | | |
| 0.9983 | 0.9983 | 0.9979 | 0.9984 | 0.9981 | 0.9979 | 0.9982 | 0.9981 | | |
| 0.9979 | 0.9980 | 0.9982 | 0.9982 | 0.9981 | 0.9981 | 0.9982 | 0.9978 | 0.9979 | |
| 0.9981 | 0.9982 | 0.9985 | 0.9986 | 0.9985 | 0.9984 | 0.9985 | 0.9985 | 0.9984 | 0.9985 |
| 0.9984 | 0.9985 | 0.9985 | 0.9986 | 0.9987 | 0.9986 | 0.9986 | 0.9987 | 0.9985 | 0.9985 |
| 0.9986 | 0.9986 | 0.9986 | 0.9986 | 0.9986 | 0.9986 | 0.9986 | 0.9986 | 0.9987 | 0.9987 |
| 0.9986 | 0.9986 | 0.9985 | 0.9986 | 0.9985 | 0.9984 | 0.9985 | 0.9985 | 0.9986 | 0.9986 |
| 0.9984 | 0.9985 | 0.9985 | 0.9985 | 0.9985 | 0.9984 | 0.9985 | 0.9985 | 0.9985 | 0.9985 |
| 0.9984 | 0.9985 | 0.9982 | 0.9982 | 0.9985 | 0.9985 | 0.9985 | 0.9982 | 0.9985 | 0.9985 |
| 0.9985 | 0.9985 | 0.9984 | 0.9984 | 0.9982 | 0.9981 | 0.9981 | 0.9985 | 0.9986 | 0.9987 |
| 0.9981 | 0.9981 | 0.9983 | 0.9983 | 0.9982 | 0.9982 | 0.9981 | 0.9981 | 0.9982 | 0.9982 |
| 0.9985 | 0.9984 | 0.9986 | 0.9982 | 0.9985 | 0.9986 | 0.9985 | 0.9984 | 0.9985 | 0.9985 |
| 0.9987 | 0.9986 | 0.9985 | 0.9985 | 0.9985 | 0.9986 | 0.9987 | 0.9986 | 0.9986 | 0.9985 |
| 0.9986 | 0.9986 | 0.9985 | 0.9985 | 0.9986 | 0.9986 | 0.9986 | 0.9986 | 0.9986 | 0.9986 |
| 0.9985 | 0.9984 | 0.9982 | 0.9986 | 0.9985 | 0.9986 | 0.9985 | 0.9984 | 0.9985 | 0.9985 |
| 0.9985 | 0.9984 | 0.9985 | 0.9985 | 0.9985 | 0.9985 | 0.9985 | 0.9984 | 0.9985 | 0.9985 |
| 0.9982 | 0.9978 | 0.9982 | 0.9985 | 0.9982 | 0.9982 | 0.9985 | 0.9985 | 0.9985 | 0.9982 |
| 0.9984 | 0.9984 | 0.9985 | 0.9982 | 0.9984 | 0.9984 | 0.9982 | 0.9981 | 0.9981 | 0.9984 |
| 0.9982 | 0.9981 | 0.9982 | 0.9984 | 0.9983 | 0.9983 | 0.9986 | 0.9985 | 0.9984 | 0.9983 |
| 0.9985 | 0.9985 | 0.9982 | 0.9983 | 0.9986 | 0.9982 | 0.9986 | 0.9987 | 0.9986 | 0.9986 |
| 0.9981 | 0.9982 | 0.9978 | 0.9986 | 0.9985 | 0.9986 | 0.9986 | 0.9986 | 0.9986 | 0.9985 |
| 0.9983 | 0.9983 | 0.9979 | 0.9985 | 0.9985 | 0.9986 | 0.9986 | 0.9985 | 0.9984 | 0.9985 |
| 0.9979 | 0.9980 | 0.9977 | 0.9985 | 0.9982 | 0.9986 | 0.9985 | 0.9985 | 0.9984 | 0.9982 |
| 0.9984 | 0.9985 | 0.9983 | 0.9982 | 0.9986 | 0.9985 | 0.9985 | 0.9984 | 0.9985 | 0.9985 |

Table A.25: AMLAugmented40Batches(CRLBO)-III

| Cramer Rao Lower Bound Optimization | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| Batch21 | Batch22 | Batch23 | Batch24 | Batch25 | Batch26 | Batch27 | Batch28 | Batch29 | Batch30 |
| TestingAccuracy | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| 0.9984 | | | | | | | | | |
| 0.9986 | 0.9982 | | | | | | | | |
| 0.9986 | 0.9985 | 0.9986 | | | | | | | |
| 0.9984 | 0.9985 | 0.9986 | 0.9987 | | | | | | |
| 0.9984 | 0.9986 | 0.9986 | 0.9986 | 0.9986 | | | | | |
| 0.9986 | 0.9985 | 0.9986 | 0.9985 | 0.9984 | 0.9985 | | | | |
| 0.9982 | 0.9981 | 0.9981 | 0.9982 | 0.9987 | 0.9986 | 0.9986 | | | |
| 0.9986 | 0.9985 | 0.9984 | 0.9985 | 0.9986 | 0.9986 | 0.9987 | 0.9986 | | |
| 0.9986 | 0.9987 | 0.9986 | 0.9986 | 0.9985 | 0.9984 | 0.9986 | 0.9986 | 0.9986 | |
| 0.9986 | 0.9986 | 0.9986 | 0.9986 | 0.9987 | 0.9986 | 0.9985 | 0.9984 | 0.9985 | 0.9986 |
| 0.9986 | 0.9985 | 0.9984 | 0.9985 | 0.9986 | 0.9986 | 0.9987 | 0.9986 | 0.9986 | 0.9986 |
| 0.9986 | 0.9987 | 0.9986 | 0.9986 | 0.9986 | 0.9987 | 0.9986 | 0.9986 | 0.9987 | 0.9984 |
| 0.9986 | 0.9986 | 0.9986 | 0.9986 | 0.9986 | 0.9986 | 0.9986 | 0.9986 | 0.9986 | 0.9986 |
| 0.9986 | 0.9985 | 0.9984 | 0.9985 | 0.9985 | 0.9985 | 0.9986 | 0.9986 | 0.9985 | 0.9984 |
| 0.9986 | 0.9985 | 0.9984 | 0.9985 | 0.9986 | 0.9987 | 0.9986 | 0.9986 | 0.9987 | 0.9986 |
| 0.9986 | 0.9987 | 0.9986 | 0.9986 | 0.9986 | 0.9986 | 0.9986 | 0.9986 | 0.9986 | 0.9986 |
| 0.9986 | 0.9986 | 0.9986 | 0.9986 | 0.9986 | 0.9986 | 0.9986 | 0.9986 | 0.9986 | 0.9986 |
| 0.9986 | 0.9985 | 0.9984 | 0.9985 | 0.9986 | 0.9985 | 0.9984 | 0.9985 | 0.9985 | 0.9985 |
| 0.9985 | 0.9985 | 0.9984 | 0.9985 | 0.9985 | 0.9985 | 0.9984 | 0.9985 | 0.9985 | 0.9985 |
| 0.9982 | 0.9985 | 0.9985 | 0.9985 | 0.9982 | 0.9985 | 0.9985 | 0.9985 | 0.9982 | 0.9985 |

Table A.26: AMLAugmented40Batches(CRLBO)-IV

| CramerRaoLowerBoundOptimization | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| Batch31 | Batch32 | Batch33 | Batch34 | Batch35 | Batch36 | Batch37 | Batch38 | Batch39 | Batch40 |
| TestingAccuracy | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| 0.9986 | | | | | | | | | |
| 0.9987 | 0.9985 | | | | | | | | |
| 0.9986 | 0.9986 | 0.9986 | | | | | | | |
| 0.9985 | 0.9986 | 0.9986 | 0.9986 | | | | | | |
| 0.9986 | 0.9985 | 0.9985 | 0.9986 | 0.9985 | | | | | |
| 0.9986 | 0.9986 | 0.9986 | 0.9985 | 0.9986 | 0.9985 | | | | |
| 0.9986 | 0.9986 | 0.9986 | 0.9986 | 0.9986 | 0.9986 | 0.9985 | | | |
| 0.9984 | 0.9985 | 0.9985 | 0.9986 | 0.9985 | 0.9985 | 0.9985 | 0.9986 | | |
| 0.9984 | 0.9985 | 0.9985 | 0.9984 | 0.9985 | 0.9985 | 0.9986 | 0.9986 | 0.9986 | |
| 0.9985 | 0.9985 | 0.9982 | 0.9985 | 0.9985 | 0.9982 | 0.9985 | 0.9986 | 0.9985 | 0.9986 |

# References


[1] Waleed Al Shehri et al. “EVALUATION OF HIGH-PERFORMANCE COMPUTING TECHNIQUES FOR BIG DATA APPLICATIONS”. In: (2019).

[2] Inc Anaconda. “Dask - Library for Parallel Computing”. In: *https://docs.dask.org /en/latest/why.html : https://docs.google.com/presentation/d/e/2PACX-1vSTH2kAR0 DCR0nw8pFBe5kuYbOk3inZ9cQfZbzOIRjyzQoVaOoMfI2JONGBz-* (2018).

[3] Elena Badal-Valero, Jose A Alvarez-Jareno, and Jose M Pavía. “Combining Benford’s Law and machine learning to detect money laundering. An actual Spanish court case”. In: *Forensic science international* 282 (2018), pp. 24–34.

[4] Gustavo EAPA Batista, Ana LC Bazzan, and Maria Carolina Monard. “Balancing Training Data for Automated Annotation of Keywords: a Case Study.” In: *WOB*. 2003, pp. 10–18.

[5] Gustavo EAPA Batista, Ronaldo C Prati, and Maria Carolina Monard. “A study of the behavior of several methods for balancing machine learning training data”. In: *ACM SIGKDD explorations newsletter* 6.1 (2004), pp. 20–29.

[6] Sonia Bergamaschi et al. “New Opportunities in High Performance Data Analytics (HPDA) and High Performance Computing (HPC)”. In: (2014).

[7] Charles Blundell et al. “Weight uncertainty in neural networks”. In: *arXiv preprint arXiv:1505.05424* (2015).

[8] Chumphol Bunkhumpornpat, Krung Sinapiromsaran, and Chidchanok Lursinsap. “DBSMOTE: density-based synthetic minority over-sampling technique”. In: *Applied Intelligence* 36.3 (2012), pp. 664–684.

[9] Nitesh V Chawla. “Data mining for imbalanced datasets: An overview”. In: *Data mining and knowledge discovery handbook*. Springer, 2009, pp. 875–886.

[10] Nitesh V Chawla et al. “SMOTE: synthetic minority over-sampling technique”. In: *Journal of artificial intelligence research* 16 (2002), pp. 321–357.

[11] Nitesh V Chawla et al. “SMOTEBoost: Improving prediction of the minority class in boosting”. In: *European conference on principles of data mining and knowledge discovery*. Springer. 2003, pp. 107–119.

[12] Min Chen, Shiwen Mao, and Yunhao Liu. “Big data: A survey”. In: *Mobile networks and applications* 19.2 (2014), pp. 171–209.

[13] Zhiyuan Chen et al. “Machine learning techniques for anti-money laundering (AML) solutions in suspicious transaction detection: a review”. In: *Knowledge and Information Systems* 57.2 (2018), pp. 245–285.

[14] Maximilian Christ, Andreas W Kempa-Liehr, and Michael Feindt. “Distributed and parallel time series feature extraction for industrial big data applications”. In: *arXiv preprint arXiv:1610.07717* (2016).

[15] Dionysios S Demetis. "Fighting money laundering with technology: A case study of Bank X in the UK". In: *Decision Support Systems* 105 (2018), pp. 96–107.

[16] Georgios Douzas and Fernando Bacao. "Effective data generation for imbalanced learning using conditional generative adversarial networks". In: *Expert Systems with applications* 91 (2018), pp. 464–471.

[17] Georgios Douzas and Fernando Bacao. "Self-Organizing Map Oversampling (SOMO) for imbalanced data set learning". In: *Expert Systems with Applications* 82 (2017), pp. 40–52.

[18] Andrew Estabrooks, Taeho Jo, and Nathalie Japkowicz. "A multiple resampling method for learning from imbalanced data sets". In: *Computational intelligence* 20.1 (2004), pp. 18–36.

[19] Liu Fang et al. "Ontology-based fraud detection". In: *International Conference on Computational Science*. Springer. 2007, pp. 1048–1055.

[20] Sandro Fiore, Mohamed Bakhouya, and Waleed W Smari. *On the road to exascale: Advances in High Performance Computing and Simulations—An overview and editorial*. 2018.

[21] Apache Software Foundation. "Mahout 0.12.0 Features by Engine". In: *https:// mahout.apache.org/users/basics/algorithms.html* (2019).

[22] Shijia Gao and Dongming Xu. "Conceptual modeling and development of an intelligent agent-assisted decision support system for anti-money laundering". In: *Expert Systems with Applications* 36.2 (2009), pp. 1493–1504.

[23] Shijia Gao et al. "Intelligent anti-money laundering system". In: *Service Operations and Logistics, and Informatics, 2006. SOLI'06. IEEE International Conference on*. IEEE. 2006, pp. 851–856.

[24] Yiannis Georgiou and Matthieu Hautreux. "Evaluating scalability and efficiency of the resource and job management system on large HPC clusters". In: *Workshop on Job Scheduling Strategies for Parallel Processing*. Springer. 2012, pp. 134–156.

[25] Alexander Gepperth and Saad Abdullah Gondal. "Incremental learning with deep neural networks using a test-time oracle." In: *ESANN*. 2018.

[26] Ian Goodfellow et al. "Generative Adversarial Nets". In: *Advances in Neural Information Processing Systems 27*. Ed. by Z. Ghahramani et al. Curran Associates, Inc., 2014, pp. 2672–2680. URL: http://papers.nips.cc/paper/5423-generative-adversarial-nets.pdf.

[27] Ian J Goodfellow et al. "An empirical investigation of catastrophic forgetting in gradient-based neural networks". In: *arXiv preprint arXiv:1312.6211* (2013).

[28] Max Halford. "Crème - Library by Google Colab". In: *https://maxhalford.github.io /slides/creme-tds/1 : https://creme-ml.github.io* (2019).

[29] Hui Han, Wen-Yuan Wang, and Bing-Huan Mao. "Borderline-SMOTE: a new over-sampling method in imbalanced data sets learning". In: *International conference on intelligent computing*. Springer. 2005, pp. 878–887.

[30] Jingguang Han et al. "Nextgen aml: Distributed deep learning based language technologies to augment anti money laundering investigation". In: *Proceedings of ACL 2018, System Demonstrations*. 2018, pp. 37–42.

[31] Asad A Ul Haq, Keren Wang, and Dragan Djurdjanovic. "Feature construction for dense inline data in semiconductor manufacturing processes". In: *IFAC-PapersOnLine* 49.28 (2016), pp. 274–279.

[32] Haibo He et al. "ADASYN: Adaptive synthetic sampling approach for imbalanced learning". In: *2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)*. IEEE. 2008, pp. 1322–1328.

[33] Neda Heidarinia, Ali Harounabadi, and Mehdi Sadeghzadeh. "An Intelligent Anti-Money Laundering Method for Detecting Risky Users in the Banking Systems". In: *International Journal of Computer Applications* 97.22 (2014).

[34] Geoffrey E Hinton and Ruslan R Salakhutdinov. "Reducing the dimensionality of data with neural networks". In: *science* 313.5786 (2006), pp. 504–507.

[35] Nathalie Japkowicz and Shaju Stephen. "The class imbalance problem: A systematic study". In: *Intelligent data analysis* 6.5 (2002), pp. 429–449.

[36] Emmanuel Jeannot et al. "Communication and topology-aware load balancing in Charm++ with TreeMatch". In: *2013 IEEE International Conference on Cluster Computing (CLUSTER)*. IEEE. 2013, pp. 1–8.

[37] Piyasak Jeatrakul, Kok Wai Wong, and Chun Che Fung. "Classification of imbalanced data by combining the complementary neural network and SMOTE algorithm". In: *International Conference on Neural Information Processing*. Springer. 2010, pp. 152–159.

[38] In-Seon Jeong et al. "A Feature Selection Approach Based on Simulated Annealing for Detecting Various Denial of Service Attacks". In: *Software Networking* 2018.1 (2018), pp. 173–190.

[39] Diederik P Kingma and Max Welling. "Auto-encoding variational bayes". In: *arXiv preprint arXiv:1312.6114* (2013).

[40] James Kirkpatrick et al. "Overcoming catastrophic forgetting in neural networks". In: *Proceedings of the national academy of sciences* 114.13 (2017), pp. 3521–3526.

[41] Miroslav Kubat, Stan Matwin, et al. "Addressing the curse of imbalanced training sets: one-sided selection". In: *Icml*. Vol. 97. Nashville, USA. 1997, pp. 179–186.

[42] Dharshan Kumaran, Demis Hassabis, and James L McClelland. "What learning systems do intelligent agents need? Complementary learning systems theory updated". In: *Trends in cognitive sciences* 20.7 (2016), pp. 512–534.

[43] Phillip M LaCasse, Wilkistar Otieno, and Francisco P Maturana. "A Survey of Feature Set Reduction Approaches for Predictive Analytics Models in the Connected Manufacturing Enterprise". In: *Applied Sciences* 9.5 (2019), p. 843.

[44] Amirali Lalehpour, Cody Berry, and Ahmad Barari. "Adaptive data reduction with neighbourhood search approach in coordinate measurement of planar surfaces". In: *Journal of Manufacturing Systems* 45 (2017), pp. 28–47.

[45] Xuan Liu and Pengzhu Zhang. "An agent based anti-money laundering system architecture for financial supervision". In: *Wireless Communications, Networking and Mobile Computing, 2007. WiCom 2007. International Conference on*. IEEE. 2007, pp. 5472–5475.

[46] Inderjeet Mani and I Zhang. "kNN approach to unbalanced data distributions: a case study involving information extraction". In: *Proceedings of workshop on learning from imbalanced datasets*. Vol. 126. 2003.

[47] Kieran Milan et al. "The forget-me-not process". In: *Advances in Neural Information Processing Systems*. 2016, pp. 3702–3710.

[48] David S Moore and Stephane Kirkland. *The basic practice of statistics*. Vol. 2. WH Freeman New York, 2007.

[49] Iman Nekooeimehr and Susana K Lai-Yuen. "Adaptive semi-unsupervised weighted oversampling (A-SUWO) for imbalanced datasets". In: *Expert Systems with Applications* 46 (2016), pp. 405–416.

[50] Hien M Nguyen, Eric W Cooper, and Katsuari Kamei. "Borderline over-sampling for imbalanced data classification". In: *Proceedings: Fifth International Workshop on Computational Intelligence & Applications*. Vol. 2009. 1. IEEE SMC Hiroshima Chapter. 2009, pp. 24–29.

[51] German I Parisi et al. "Continual lifelong learning with neural networks: A review". In: *Neural Networks* (2019).

[52] Jason Brownlee PhD. "Progressive Loading". In: *https://machinelearningmastery .com/large-data-files-machine-learning/* (2017).

[53] Jon TS Quah and M Sriganesh. "Real-time credit card fraud detection using computational intelligence". In: *Expert systems with applications* 35.4 (2008), pp. 1721–1732.

[54] Saleha Raza and Sajjad Haider. "Suspicious activity reporting using dynamic bayesian networks". In: *Procedia Computer Science* 3 (2011), pp. 987–991.

[55] Daniel A Reed and Jack Dongarra. "Exascale computing and big data". In: *Communications of the ACM* 58.7 (2015), pp. 56–68.

[56] Muhammad Habib ur Rehman et al. "Big data reduction framework for value creation in sustainable enterprises". In: *International Journal of Information Management* 36.6 (2016), pp. 917–928.

[57] Paul Ruvolo and Eric Eaton. "ELLA: An efficient lifelong learning algorithm". In: *International Conference on Machine Learning*. 2013, pp. 507–515.

[58] Andreas Santucci. "Distributed Algorithms and Optimization". In: *https://stanf ord.edu/ rezab/classes/cme32* (2017).

[59] Joan Serrà et al. "Overcoming catastrophic forgetting with hard attention to the task". In: *arXiv preprint arXiv:1801.01423* (2018).

[60] Michael R Smith, Tony Martinez, and Christophe Giraud-Carrier. "An instance level analysis of data complexity". In: *Machine learning* 95.2 (2014), pp. 225–256.

[61] Rupesh K Srivastava et al. "Compete to compute". In: *Advances in neural information processing systems*. 2013, pp. 2310–2318.

[62] Ch Suresh, K Thammi Reddy, and N Sweta. "A hybrid approach for detecting suspicious accounts in money laundering using data mining techniques". In: *International Journal of Information Technology and Computer Science (IJITCS)* 8.5 (2016), p. 37.

[63] Kristin M Tolle, D Stewart W Tansley, and Anthony JG Hey. "The fourth paradigm: Data-intensive scientific discovery [point of view]". In: *Proceedings of the IEEE* 99.8 (2011), pp. 1334–1337.

[64] Su-Nan Wang and Jian-Gang Yang. "A money laundering risk evaluation method based on decision tree". In: *Machine Learning and Cybernetics, 2007 International Conference on*. Vol. 1. IEEE. 2007, pp. 283–286.

[65] Wei Wang et al. "Abstracting massive data for lightweight intrusion detection in computer networks". In: *Information Sciences* 433 (2018), pp. 417–430.

[66] Pengfei Xuan et al. "Big data analytics on traditional HPC infrastructure using two-level storage". In: *Proceedings of the 2015 International Workshop on Data-Intensive Scalable Computing Systems*. ACM. 2015, p. 4.

[67] Yazhou Zhang. "Deep generative model for multi-class imbalanced learning". In: (2018).

[68] Yadong Zhou et al. "Analyzing and Detecting Money-Laundering Accounts in Online Social Networks". In: *IEEE Network* 32.3 (2017), pp. 115–121.

[69] Tianqing Zhu. "An outlier detection model based on cross datasets comparison for financial surveillance". In: *Services Computing, 2006. APSCC'06. IEEE Asia-Pacific Conference on*. IEEE. 2006, pp. 601–604.

[70] Maciej Zięba, Jakub M Tomczak, and Adam Gonczarek. "RBM-SMOTE: restricted Boltzmann machines for synthetic minority oversampling technique". In: *Asian Conference on Intelligent Information and Database Systems*. Springer. 2015, pp. 377–386.