Analysis of Anti-Money Laundering (AML) with Spark ML

Katherine Belknap, Snehil Sarkar, Sapan Shah
Department of Information Systems, California State University Los Angeles
CIS 5560-01 Intro to Big Data Science
kbelkna2@calstatela.edu, ssarkar4@calstatela.edu, sshah82@calstatela.edu

Abstract: The findings of this study hold the potential to significantly impact the financial sector, corporations, and governments. Money Laundering is a serious issue that costs upwards of two trillion dollars annually, which accounts for roughly 2-5% the global Gross Domestic Product (GDP). By predicting future occurrences of money laundering activities and visualizing potential trends, it could lead to safer and more secure financial transactions. Due to the vast amount of financial data created daily, the best way to predict these occurrences is through Machine Learning (ML). Various algorithms can be utilized to uncover illicit transactions, this project seeks to find the best amongst them. For this project we used Community Databricks to develop our models and Pyspark CLI through the Hadoop File System to run the final models. We utilized an IBM simulated dataset found through Kaggle and used the following algorithms: Logistic Regression (LR), Linear Support Vector Machine (SVC), Random Forest (RF), and Gradient Boosted Tree (GBT). Each algorithm was used with both training and cross validation splits.

1. Introduction

[1] According to Chen with investopedia, Money Laundering is the blanket term used to describe money that originates from illegal activity, but is carefully processed to appear legitimate. [1] "The money from the criminal activity is considered dirty, and the process "launders" it to look clean." [2] "The United Nations estimated the cost of global money laundering annually to be between USD 800 billion – USD 2 trillion, which makes up 2-5% of global GDP"

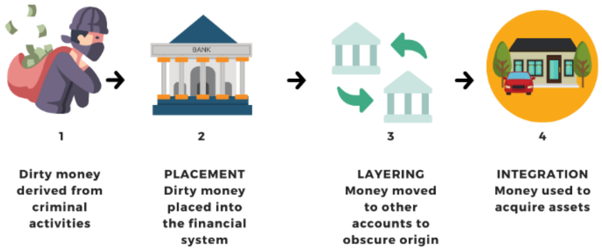
BANK
1
Dirty money derived from criminal activities
2
PLACEMENT
Dirty money placed into the financial system
3
LAYERING
Money moved to other accounts to obscure origin
4
INTEGRATION
Money used to acquire assets

Figure 1 – Money Laundering Process

Money laundering begins with the acquisition of funds through less than legal means. The actual prcoess of money laundering starts with placement of the "dirty" money into the finacial system. This can be done through cash purchases or cash deposits. [3] "In cases of cash deposits, the money launderers deposit cash that falls below the threshold of suspicious transactions into various bank accounts." The next step is called layering, through which the initial source of the money is concealed. There are various methods, but the end result typically finds the money in an offshore account. The final step is integration, in which the now "clean" money is withdrawn and used.

There are many types of transactions that money launderers use to obfuscate the origin of "dirty" money. [4] Smurfing, a term that "appears to be borrowed from illegal drug manufacturers", involves splitting what would be a large transaction into multiple smaller ones to fall below the level of scrutiny of financial insitutions. Cash smugglers, on the other hand move the physical cash acrros borders initially then make the deposit into those foreign accounts. [1] Investment in commodities, or otherwise "using gems and gold that can be moved easily to other jurisdictions." [1] Either buying or selling, "using cash for quick turnaround investment in assets" is also popular. [5] Art has been a popular asset since "art dealers are not obligated to disclose the identities of their buyers," especially in the United States, in which "galleries or auction houses are not required to report high-value transactions to any government body." Finally, Gambling and shell companies are some ways in which money launderers can attempt to "clean" their money.

2. Related Work

[3] "Anti-money laundering which is commonly abbreviated as "AML" refers to a set of rules, laws, regulations, policies, procedures, and controls adopted to combat money laundering." [1] In 1970 the Bank Secrecy Act was passed "requiring financial institutions to report cash transactions above $10,000 or unusual activity on a suspicious activity report (SAR) to the Department of the Treasury." By 1986 the Money Laundering Control Act was passed, and after the events of September 11th 2001 efforts to prevent money laundering were expanded.

[3] According to Van Rompaye, only 1% of laundering cases are actually detected and acted upon by the proper authorities. [3] Therefore, as a way to increase detection "regulatory bodies are pushing the banks harder each year by updating anti-money laundering (AML) compliance requirements and tightening the controls." However, most banks are still operating in a somewhat traditional way making the increased requirements a burden to operations causing a drain on efficiency. Transaction monitoring and rule-based systems have been the norm, but the structured nature of these systems leaves them vulnerable to loopholes. For example, smurfing will go unnoticed when only looking for transaction over a certain size rather than multiple small transactions between the same accounts.

This is where machine learning (ML) comes into play. [3] "ML removes the need to manually define rules to sift through data." [3] According to Van Rompaye, "identification of suspicious activity can be improved by up to 40% by replacing rule-based and scenario-based tools with ML models." With higher scrutiny and fines for any deficiencies, as well as advancements in ML technologies over the years, banks are looking toward ML solutions more than ever in the anti-money laundering effort.

3. Specifications

For this project we used Community Databricks to test the fraud detection models during development, as it offers a scalable computing environment. Pyspark CLI through the Hadoop File System was then used to run the final models.

Table 1. Hardware Specifications

Community Databricks	
Version	12.2 LTS
Memory	15.3 GB
Cores	2
Nodes	1
Hadoop	
Version	Hadoop 3.3.3 Pyspark 3.2.1
CPU Speed	1995.312 GHz
CPU cores	8
Nodes	5 (3 Master, 2 worker)
Memory	806.40GB

Table 2. Data Set Specifications

IBM Transactions for Anti-Money Laundering	
https://www.kaggle.com/datasets/ealtman2019/ibm-transactions-for-anti-money-laundering-aml	
File used	HI_Medium_Trans.csv
Data size	2.82 GB
Total transactions	31,898,238
Laundering/Non-Laundering Ratio	1 : 904

4. Working with the Data

The dataset used in this project is simulated data by IBM, therefore not based on actual transactions from real accounts. The dataset is divided amongst six separate files and segregated into two categories based on the prevalence of money laundering, either high or low ratios. Additioanlly, each file is labeled for the size of the set. For training a small file was used, whereas for the final model, a medium file proved sufficient.

4.1 Project Workflow

The dataset used in this project was made available to us via Kaggle. After download, the small high illicit transaction file was used for training with Community Databricks. Once the Cluster was ready and the file was uploaded it was time to develop the models. For usability of the data, indexers were used to transform string data into numeric values. To build the models, vector assemblers and a pipeline were used. The models additionally utilized tuning parameters, as well as, both train and cross validations in order to build the best models possible. Finally, models were evaluated using binary evaluation, as all models were of the binary variety. Feature importance was also considered at this step.

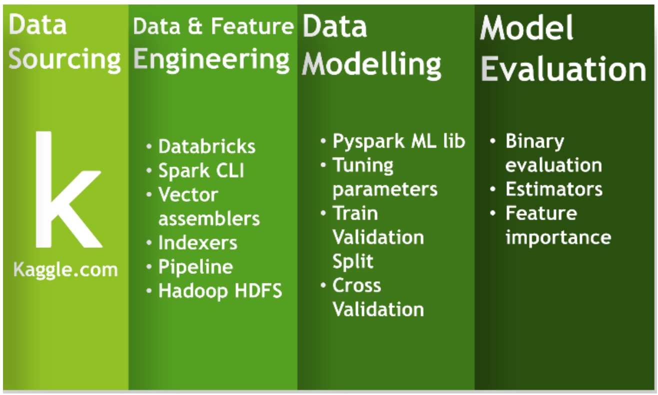

Figure 2 – Project Architecture

In order, the process begins with the data sourcing, moving into data and feature engineering. From there the data is split into the training and testing portions, with a 7 :3 split, .the assembler is created and the training begins. Four algorithms were chosen for this project and after the appropriate pipelines were created the testing phase begins. Lastly, the previously mentioned training validators were used and we could compare measures for accracy against the results. Once the small file was running smoothly with the models devleoped in Databricks, we moved on to applying the models to the medium file utilizing Pyspark in the HDFS environment.

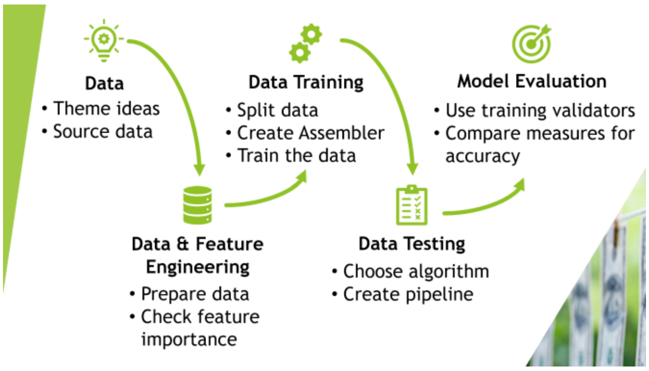

Figure 3 - Implementation Flowchart

4.2 Data and Feature Engineering

Balancing the dataset was important, as even with the files that contained more instances of illicit transactions the laundering to non-laundering ratio was dramatic as can be seen in the following table. With a ratio of one laundering case per over 900 cases of non-laundering in the dataset used for this project, over sampling would not work well. The count for laundering cases was far too low to raise it up to the level of non-laundering cases. Therefore, rather than raising the laundering cases to match, we used under sampling to bring down the number of non-laundering cases to match the much lower volume of cases of laundering.

Doing so ensures that any model developed with the data is not biased. Using the data as it was, that is to say, non-balanced data, could result in a model that favors labeling a transaction as not laundering, due to the overwhelming non-laundering cases in the dataset.

Table 3. Ratio of Laundering to Non-Laundering Cases in Medium_HI Dataset

Total Transactions	31,898,238
Total Laundering	35,230
Total Non-Laundering	31,863,008
Ratio	1 : 904

The following table describes the features that were used in the models developed for this project. The label for the models is the 'Is Laundering' binary column. Other columns in the dataset were not included in the vector assembler, such as timestamp and account information.

Table 4. Features and Label for Models

Feature	**Description of data**
From Bank	Numeric code for bank where transaction originates
To Bank	Numeric code for bank where transaction ends
Amount Received	Monetary amount received
Receiving Currency	Currency (dollars, euros, etc.) of account
Amount Paid	Monetary amount received
Payment Currency	Currency (dollars, euros, etc.) of account
Payment Format	How transaction was conducted, e.g. cheque, ACH, wire, credit cards, etc.
Is Laundering	**Binary code for if the transaction is flagged as illicit**

4.3 Data Modeling

The following machine learning algorithms were used in this project: Logistic Regression (LR), Linear Support Vector Machine (SVC), Random Forest (RF), and Gradient Boosted Tree (GBT). The tuning parameters for each algorithm are shown in the following table. Each algorithm employees the same number of parameters making the models fairer in comparison to each other, in terms of robustness and time spent computing.

Table 5. Hyperparameter Tuning by Algorithm

#Define paramGrid
paramGrid = ParamGridBuilder() \	

LR	.addGrid(lr.regParam, [0.01, 0.1, 1.0]) \ .addGrid(lr.elasticNetParam, [0.0, 0.5, 1.0]) \
SVC	.addGrid(lsvc.regParam, [0.01, 0.1, 1.0]) \ .addGrid(lsvc.maxIter, [10, 20, 30]) \
RF	.addGrid(rf.numTrees, [10, 20, 30]) \ .addGrid(rf.maxDepth, [3, 5, 8]) \
GBT	.addGrid(gbt.maxDepth, [3, 5, 8]) \ .addGrid(gbt.maxIter, [10, 20, 30]) \
.build()	

5. Model Evaluation

For Anti-Money Laundering the most important measure is recall. A false positive is not much of an issue, as accidentally labeling a transaction as illicit when it was not could be moved on from after investigation and an apology. Meanwhile, letting actually illicit transactions fall through the cracks, with a false negative, can cost millions if not billions of dollars. Not to mention the sustained harm caused by letting criminal organizations get away with such a large amount of money, furthering their funding for even more illegal activity. Area Under the Curve (AUC) is also an important measure for our models as it represents how accurate the model is overall.

Table 6. Model Comparison

Algorithm	**Precision**	**Recall**	**AUC**	**Time**
Train Validation Split				
LR	0.7462	0.8370	0.7082	2.26m
SVC	0.6878	0.8635	0.7046	3.20m
RF	0.8663	0.9381	0.9648	3.54m
GBT	0.8695	0.9631	0.9703	7.14m
Cross Validation Split				
LR	0.7074	0.8195	0.7058	5.18m
SVC	0.6876	0.8711	0.7121	8.05m
RF	0.8671	0.9365	0.9655	6.47m
GBT	0.8654	0.9525	0.9684	17.17m

Based on the comparison provided (table 6.), the best model for anti-money laundering is the Gradient Boosted Tree model, particularly with the Train Validation Split. This model has not only the highest recall but also the highest AUC. The second-best model is also GBT but with the Cross Validation Split. However, GBT with CVS takes far longer to run, therefore, if time is an issue, Random Forest with Cross Validation Split is the next best option.

Table 7. Feature Importance

Feature	**Importance**
Payment Format	**0.618812**
From Bank	0.137167
Amount Received	0.062771
Receiving Currency	0.060838
Amount Paid	0.048353
Payment Currency	0.036077
To Bank	0.035981

Feature importance was explored for the best model, which as previously stated is the GBT-CV model. Payment Format turned out to have the most effect on the model. We decided to explore this feature more closely and found that for the dataset used the Automated Clearing House (ACH) type was the format most used for illicit transactions. An ACH transaction is a is a type of electronic transfer between two financial institutes. On the other hand, reinvestment and wire transfers had no counts of laundering in the dataset. The low count for cash was surprising, but the use of cash has been on the decline in recent years. [6] Cash use dropped from 20% to 18% usage between 2021 and 2022, "making it the third-most used payment method in the U.S."

Table 7. Payment Formats Laundering Count

Payment Format	Laundering Count	Non-Laundering Count
ACH	**30,746**	**3,837,664**
Cheque	2,220	12,277,838
Credit Card	1,354	8,776,462
Cash	666	1,119,774
Bitcoin	244	688794
Reinvestment	0	1,945,611
Wire	0	3,216,865

6. Conclusions

Gradient Boosted Tree with Train Validation Split is the best model for this dataset due to the higher Recall and Area Under the Curve values. Recall is more important than precision in our case, as the ramifications for false negatives are far worse than that of false positives. This model also uses a fraction of the time that Cross Validation Split needs for the GBT algorithm.

For this model, 'Payment Format' was shown to be the most important feature of money laundering instances. The originating bank was a far second in terms of importance. Looking further into the formats available, the Automated Clearing House type was most used for illicit transactions. Meanwhile, reinvestment and wire transfer had no cases of money laundering within the dataset.

Table 8. Review of Top Three Models

Algorithm	Recall	AUC	Time
GBT - TVS	0.9631	0.9703	7.14m
GBT - CV	0.9525	0.9684	17.17m
RF - TVS	0.9381	0.9648	3.54m

References1

[1] Chen, J. (2024, April 11). *What is money laundering?*. Investopedia. https://www.investopedia.com/terms/m/moneylaundering.asp

[2] Van Rompaye, B. (2023, August 29). *Machine learning for anti-money laundering (ML for AML)*. KPMG. https://kpmg.com/be/en/home/insights/2023/08/lh-machine-learning-for-anti-money-laundering.html#:~:text=By%20streamlining%20the%20process%20of,and%20help%20prevent%20money%20laundering.

[3] Sabao, K. (2022, August 25). *The AML compliance series: Part 1 - basics of anti-money laundering*. LinkedIn. https://www.linkedin.com/pulse/aml-compliance-series-basics-anti-money-laundering-part-kelvin-sabao

[4] Chen, J. (2024, April 11). *What is money laundering?*. Investopedia. https://www.investopedia.com/terms/m/moneylaundering.asp

[5] Amparo, M. (2022, February 8). *Money laundering in the art market: How and why it happens*. Bolder. https://boldergroup.com/insights/blogs/money-laundering-in-the-art-market/#:~:text=The%20incognito%20culture%20in%20the,venue%20to%20store%20dirty%20cash.

[6] Lindsay, J. (2023, November 2). *A fatal cash crash? conditions were ripe for it after the pandemic hit, but it didn't happen*. Federal Reserve Bank of Boston. https://www.bostonfed.org/news-and-events/news/2023/11/cash-crash-pandemic-increasing-credit-card-use-diary-of-consumer-payment-choice.aspx#:~:text=In%202021%2C%20cash%20use%20accounted,fall%20below%20for%20a%20while.

[1] Github Link: https://github.com/ssarkar4/AntiMoneyLaundering_BigData