University of Toronto
ECE496 Final Report

Credit Card Fraud Detection

Team Number: 2022524

Supervisor: Zeb Tate

Shadman Kaif, Abdurrafay Khan, Krutarth Patel, Shanthosh Sivayogalingam

ECE496Y

Voluntary Document Release Consent Form

To all ECE496 students:

To better help future students, we would like to provide examples that are drawn from excerpts of past student reports. The examples will be used to illustrate general communication principles as well as how the document guidelines can be applied to a variety of categories of design projects (e.g. electronics, computer, software, networking, research).

Any material chosen for the examples will be altered so that all names are removed. In addition, where possible, much of the technical details will also be removed so that the structure or presentation style are highlighted rather than the original technical content. These examples will be made available to students on the course website, and in general may be accessible by the public. The original reports will not be released but will be accessible only to the course instructors and administrative staff.

Participation is completely voluntary and students may refuse to participate or may withdraw their permission at any time. Reports will only be used with the signed consent of all team members. Participating will have no influence on the grading of your work and there is no penalty for not taking part.

If your group agrees to take part, please have all members sign the bottom of this form.

Consent Statement

We verify that we have read the above letter and are giving permission for the ECE496 course coordinator to use our reports as outlined above.

Team #: 2022524 Project Title: Credit Card Fraud Detection

Supervisor(s): Zeb Tate Administrator: Inci McGreal

Name	Shadman Kaif	**Signature**	Shadman Kaif	**Date:**	April 5, 2023
Name	Abdurrafay Khan	**Signature**	A Khan	**Date:**	April 5, 2023
Name	Krutarth Patel	**Signature**	Kpatel	**Date:**	April 5, 2023
Name	Shanthosh Sivayogalingam	**Signature**	S. Shanthosh	**Date:**	April 5, 2023

Executive Summary (author: K. Patel)

The motivation behind this project stemmed from the importance of credit card transaction processing standards keeping in line with the growth of e-commerce. With clear indications of a growing trend in consumer habits and credit card usage, it is imperative that banks are able to authenticate and approve genuine transactions and accurately reject those that could be fraudulent. Even with the current state-of-the-art machine learning (ML) models, $118 billion was lost due to false positives in 2014 and these models only show a 78% accuracy when distinguishing between fraud and legitimate transactions. That is why the goal of our project was to create a linearly classifiable ensemble ML algorithm that minimizes the number of false positives to at least 20-30% in a credit card dataset.

The objective of this project was to design a solution that also achieves an accuracy of 95% while being able to process at least 65,000 transactions/sec. Late additions to these objectives were meeting a recall score of at least 50% and a false negative rate below 30%. The team had to complete this design solution considering constraints such as being limited to simulated transaction data for privacy and security reasons as well as not being able to process real-time data without a history of transactions for the model to learn from. Ultimately, the team was able to find an optimal ML model using the AdaBoost ensemble learning method.

Through various iterations of tuning different hyperparameters, the team finalized the optimal design solution to be an AdaBoost model with a learning rate of 1.0, 10 estimators and a probability threshold of 50%. This final model produced the best results in minimizing false positives, keeping false negatives low while also not sacrificing a high accuracy. From the testing and verification the model only failed to meet the goal of processing at least 65,000 transactions/sec but speed was the team's least valued objective because it is strongly correlated to hardware.

Overall, the final model is ready and can be used in the backend of a bank system to better combat fraud. The AdaBoost model outperformed current state-of-the-art models as expected with respect to false positives. The team does understand that the testing and verification could have accounted for other variations of the AdaBoost model which use different estimators which could lead to better results. Future adaptations of this project can be to develop this model to assist fraud in other financial sectors such as insurance or loans.

Table of Contents

Individual Contributions - Shadman Kaif

Shadman's contributions during the Capstone project are outlined in the table below. Shadman was instrumental in the acquisition of the dataset, the development and testing of the LSTM model, hyperparameter tuning with the AdaBoost model, confusion matrix and processing speed tests, and the creation of the design fair poster. The team believes Shadman contributed 25% to the final project.

Task	Status	Date Started	Date Completed
IBM Dataset Acquisition	Done	August 26, 2022	September 16, 2022
Draft Proposal	Done	September 16, 2022	September 23, 2022
Final Proposal	Done	September 24, 2022	October 27, 2022
Interim Demo Review Slides	Done	October 3, 2022	October 14, 2022
Implementation Plan	Done	December 1, 2022	January 9, 2023
LSTM Model Implementation	Done	January 16, 2023	January 23, 2023
LSTM Model Test	Done	January 22, 2023	February 2, 2023
Test Document	Done	January 24, 2023	February 1, 2023
Oral Presentation Slides	Done	February 9, 2023	March 9, 2023
Final Report	Done	March 10, 2023	April 6, 2023
AdaBoost Model Test with LR = 1.00, Batches = 100, Num. Estimators = 2, 5, 10	Done	March 25, 2023	March 27, 2023
AdaBoost Model Test with LR = 1.00, Batches = 300, Num. Estimators = 2, 5, 10	Done	March 27, 2023	April 1, 2023
AdaBoost Model Test with LR = 1.00, Batches = 1000, Num. Estimators = 2, 5, 10	Done	March 29, 2023	April 3, 2023
Confusion Matrix Tests	Done	March 25, 2023	April 4, 2023
Processing Speed Tests	Done	April 3, 2023	April 4, 2023
Design Fair Poster	Done	April 1, 2023	April 5, 2023

Individual Contributions - Abdurrafay Khan

Abdurrafay's contributions during the Capstone project are outlined in the table below. Abdurrafay was instrumental in the initial data processing, the development of the AdaBoost model, the hyperparameter tuning data processing, graph creation, model selection, and the creation of the design fair poster. The team believes Abdurrafay contributed 25% to the final project.

Task	Status	Date Started	Date Completed
Draft Proposal	Done	September 16, 2022	September 23, 2022
Final Proposal	Done	September 24, 2022	October 27, 2022
Interim Demo Review Slides	Done	October 3, 2022	October 14, 2022
Implementation Plan	Done	December 1, 2022	January 9, 2023
AdaBoost Model Implementation	Done	January 23, 2023	January 30, 2023
Test Document	Done	January 24, 2023	February 1, 2023
Oral Presentation Slides	Done	February 9, 2023	March 9, 2023
Final Report	Done	March 10, 2023	April 6, 2023
AdaBoost Model Test with LR = 1.50, Batches = 100, Num. Estimators = 2, 5, 10	Done	March 25, 2023	March 27, 2023
AdaBoost Model Test with LR = 1.50, Batches = 300, Num. Estimators = 2, 5, 10	Done	March 27, 2023	April 1, 2023
AdaBoost Model Test with LR = 1.50, Batches = 1000, Num. Estimators = 2, 5, 10	Done	March 29, 2023	April 3, 2023
Analysis and Inferencing for Hyperparameter Tuning Tests	Done	March 27, 2023	April 4, 2023
Graph Creation	Done	March 27, 2023	April 4, 2023
Model Selection	Done	March 27, 2023	April 4, 2023
Design Fair Poster	Done	April 1, 2023	April 5, 2023

Individual Contributions - Krutarth Patel

Krutarth's contributions during the Capstone project are outlined in the table below. Krutarth was essential in the communication with both the administrator and supervisor, the development of the LSTM model, hyperparameter tuning with the AdaBoost model, and the creation and acquisition of the design fair poster. The team believes Krutarth contributed 25% to the final project.

Task	Status	Date Started	Date Completed
Communication with Administrator and Supervisor	Done	Start of Project	End of Project
Draft Proposal	Done	September 16, 2022	September 23, 2022
Final Proposal	Done	September 24, 2022	October 27, 2022
Interim Demo Review Slides	Done	October 3, 2022	October 14, 2022
Implementation Plan	Done	December 1, 2022	January 9, 2023
LSTM Model Implementation	Done	January 16, 2023	January 23, 2023
Test Document	Done	January 24, 2023	February 1, 2023
Oral Presentation Slides	Done	February 9, 2023	March 9, 2023
Final Report	Done	March 10, 2023	April 6, 2023
AdaBoost Model Test with LR = 0.50, Batches = 100, Num. Estimators = 2, 5, 10	Done	March 25, 2023	March 27, 2023
AdaBoost Model Test with LR = 0.50, Batches = 300, Num. Estimators = 2, 5, 10	Done	March 27, 2023	April 1, 2023
AdaBoost Model Test with LR = 0.50, Batches = 1000, Num. Estimators = 2, 5, 10	Done	March 29, 2023	April 3, 2023
Design Fair Poster	Done	April 1, 2023	April 5, 2023
Digital Poster Acquisition	Ongoing	April 5, 2023	TBD

Individual Contributions - Shanthosh Sivayogalingam

Shanthosh's contributions during the Capstone project are outlined in the table below. Shanthosh was instrumental in the research for the motivation, the initial data processing, the development of the AdaBoost model, hyperparameter tuning tests, and the creation of the design fair poster. The team believes Shanthosh contributed 25% to the final project.

Task	Status	Date Started	Date Completed
Research for Motivation and Design Solution	Done	September 16, 2022	October 27, 2022
Draft Proposal	Done	September 16, 2022	September 23, 2022
Final Proposal	Done	September 24, 2022	October 27, 2022
Interim Demo Review Slides	Done	October 3, 2022	October 14, 2022
Implementation Plan	Done	December 1, 2022	January 9, 2023
Data Processing	Done	January 9, 2023	January 16, 2023
AdaBoost Model Implementation	Done	January 23, 2023	January 30, 2023
Test Document	Done	January 24, 2023	February 1, 2023
Oral Presentation Slides	Done	February 9, 2023	March 9, 2023
Final Report	Done	March 10, 2023	April 6, 2023
AdaBoost Model Test with LR = 0.75, Batches = 100, Num. Estimators = 2, 5, 10	Done	March 25, 2023	March 27, 2023
AdaBoost Model Test with LR = 0.75, Batches = 300, Num. Estimators = 2, 5, 10	Done	March 27, 2023	April 1, 2023
AdaBoost Model Test with LR = 0.75, Batches = 1000, Num. Estimators = 2, 5, 10	Done	March 29, 2023	April 3, 2023
Design Fair Poster	Done	April 1, 2023	April 5, 2023

1.0 Introduction (author: A. Khan)

The following report outlines the entire design process surrounding a credit card fraud detection implementation using ensemble machine learning as part of the final year design project course EC496. Specifically, it highlights the motivation, factors contributing to the final design and also the testing/verification for the final model. The end of the document will summarize the team's final thoughts on the overall project including how well the final design solved the initial problem, how closely the project goals were met and final insights into how this project can be deployed in the credit card industry.

The motivation behind this project stemmed from the importance of credit card transaction processing standards keeping in line with the growth of e-commerce. From 2019 to 2020, the US Census Bureau Annual Retail Trade Survey reported that there was roughly a 43% increase in e-commerce sales [1]. During the same time, the credit card payment market has seen an increasing trend and is expected to grow roughly 7.4% annually over the course of the next 5 years [2]. With clear indications of a growing trend in consumer habits and credit card usage, it is imperative that banks are able to authenticate and approve genuine transactions and accurately reject those that could be fraudulent. Today, credit card companies use a mix of machine learning (ML) models and algorithms to evaluate and classify purchases that are likely to be fraudulent [3]. Even with such protocols in place, in 2014, $118 billion was lost in legitimate sales due to false positives [3]. The overwhelming money lost due to false positives makes it clear that a new way to minimize them would lead to better profit margins for credit card companies and merchants. Long short-term memory models which are the current state of the art models show only a 78% accuracy when distinguishing between fraudulent and legitimate transactions. A more recent development in the field of machine learning, ensemble learning, can be used to reduce these false positive rates. Ensemble learning showed promising results in other markets such as malware detection and detecting spam twitter posts [4][5]. The lack of testing with ensemble learning in the credit card industry leaves the question of how well can they work to solve the issue of false positives.

1.1 Project Goal (author: A. Khan)

The goal of this project is to create a linearly classifiable ensemble ML algorithm that minimizes the number of false positives to at least 20-30% [3] in a credit card dataset.

1.2 Requirements Specification (author: A. Khan)

Functions

The project's major task is to detect fraudulent credit card transactions using ensemble learning. Primary functions correspond to this functional basis while secondary functions were generated as a result.

Primary	Detect fraudulent credit card transactions using ensemble learning
Secondary	Use the AdaBoost ensemble algorithm

Table 1: Primary and Secondary Functions

Objectives

Table below outlines aspects considered when measuring the success of possible solutions.

No.	Objectives: Should	Metric	Goal
1	Minimize the amount of false positives	False Positives (%)	20-30% [3]
2	Be accurate	Model Accuracy (%)	95% [6]
3	Processing speed	Speed (transactions/s)	65,000 [7]
4	Recall	True Positive Rate (%)	50-100%
5	Minimize False Negatives	False Negative Rate (%)	0-30%

Table 2: List of Project Objectives

Constraints

Table below shows project limitations based on the gap and certain codes/regulations within financial institutions.

Constraint:	Type	Reasoning
Model cannot make a prediction for a real-time transaction when the transaction does not have historical data	Functional	There must be a sequence length for our model to base previous transactions made by a card in order to predict the next transaction.
No access to real data from credit card companies	Ethical	Financial institutions must adhere to "Protection of Personal Information" [8]. As a result, the design team will resort to a simulated dataset [9].

Table 3: Design Constraints

2.0 Final Design (author: K. Patel)

Having considered various design alternatives and conducting several tests, the final design solution is an AdaBoost algorithm with a Decision Tree Classifier having a learning rate of 1.0, 10 estimators and a probability threshold of 50%.

2.1 System-Level Overview (author: K. Patel)

After considering various alternatives such as support vector machines and bagging, we have made the decision to use the AI system that uses the decision tree AdaBoost technology. This was due to the fact that when each alternative was compared to the others and their capabilities were evaluated against our objectives, the AdaBoost model was the most effective and aligned more with our goals. AdaBoost is an ensemble learning technique that improves the performance and accuracy of machine learning algorithms. It lowers the bias and variance of a prediction model and is used in both regression and classification models to prevent overfitting of the data [10]. Figure 1 illustrates a system-level overview of how AdaBoost is utilized within our scope.

2.2 System Block Diagram (author: K. Patel)

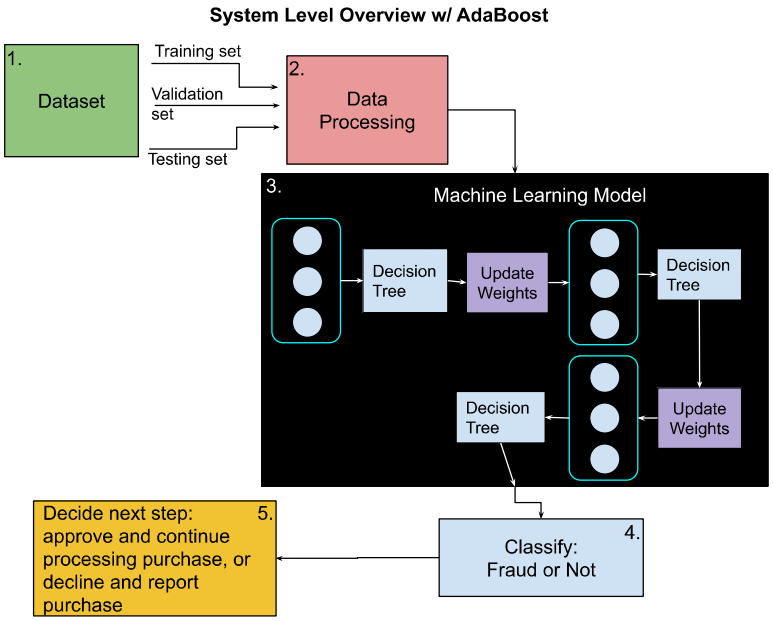

Figure 1: System Level Overview of Final Design

2.3 Module-level Descriptions and Designs (author: S. Sivayogalingam)

The final design can be divided into 3 main modules which are shown in the table below.

No	Module	Description
1	Dataset	The final model is using an IBM dataset consisting of 24 million transactions. Figure 2 shows a partial snapshot of the data being used. The data is split into training (50%), validation (30%), and testing (20%) sets.
2	Data Processing	The data from the dataset is processed and information that contains texts such as transaction type, merchant name, etc will be converted to numerical data using encoders to make it easier and more efficient for the computer to analyze it.
3	Machine Learning Model	The data from the data processing module will be used as the input to the model to create a weighted total that represents the boosted classifier's final results. Initially, all the training samples have the same weight. After each iteration, the weight distribution is updated such that misclassified samples have more weight. The 10 weak estimators in association with the AdaBoost algorithm are used to train and validate the final model.

Table 4: Module Level Descriptions

	User	Card	Year	Month	Day	Time	Amount	Use Chip
0	0	0	2002	9	1	06:21	$134.09	Swipe Transaction
1	0	0	2002	9	1	06:42	$38.48	Swipe Transaction
2	0	0	2002	9	2	06:22	$120.34	Swipe Transaction
3	0	0	2002	9	2	17:45	$128.95	Swipe Transaction
4	0	0	2002	9	3	06:23	$104.71	Swipe Transaction
...	...	...	...	...	...	...	...	...
16499995	1338	1	2004	8	21	16:55	$38.26	Online Transaction
16499996	1338	1	2004	8	22	10:49	$17.28	Swipe Transaction
16499997	1338	1	2004	8	22	11:55	$40.29	Online Transaction
16499998	1338	1	2004	8	23	11:50	$37.01	Online Transaction
16499999	1338	1	2004	8	23	16:40	$27.14	Online Transaction

Figure 2: Partial Component of Raw Data

2.4 Assessment of Final Design (author: S.Sivayogalingam)

To determine a final model with set parameters that provided optimal results for all objectives, the team conducted multiple training and testing iterations while tuning parameters to compare results. The main parameters that were altered through different iterations included the learning rate, batch size, number of estimators and various probability thresholds for classification. Specifically, the experimented values for these parameters were learning of 0.5,0.75, 1.0, 1.5 and 5, batch sizes varying from 100, 300 and 1000, number of estimators varying from 2, 5 and 10, and thresholds varying from 25-60% incrementing by 5%. (see Appendix A, B, C, D, E).

The main objective to choose the ideal model was to choose the one which showed the lowest false negative and false positive rates without sacrificing the total accuracy of the model. The team decided to value transaction speed the least because that can vary depending on the hardware used (computer) and because this would stay in line with the decision matrix used to eliminate alternative solutions to our problem. The graph [Figure 3] and table [Figure 4] below illustrate the results of one of the tests conducted to help decide on the chosen solution.

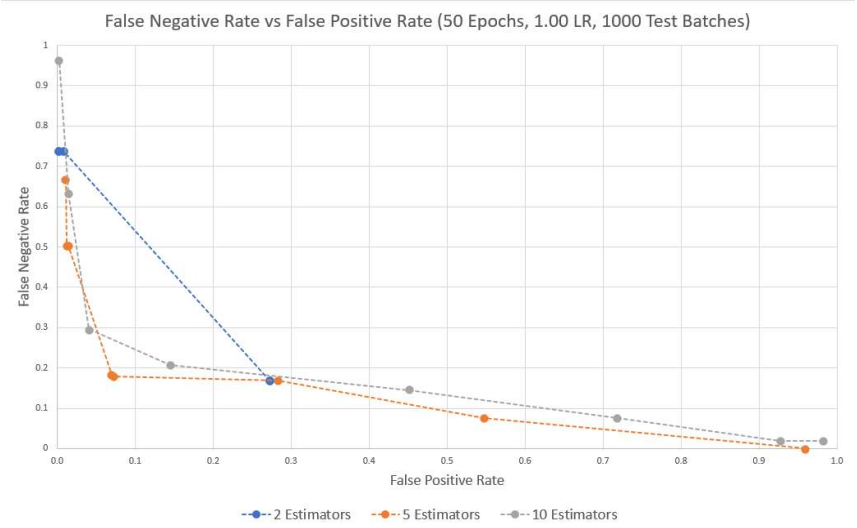

Figure 3: False Negatives vs False Positives for Final Design Solution

Training Parameters		Testing Parameters		Results		
Learning Rate	Estimator	Number of Test Batches	Threshold	FP Rate	FN Rate	Average Accuracy
1	10	1000	25%	0.982010430550958000	0.018779342720000000	0.0198
			30%	0.927138218218576000	0.018779342720000000	0.0746
			35%	0.718562981384239000	0.075117370890000000	0.2827
			40%	0.451716210292788000	0.145539906100000000	0.5489
			45%	0.145312066698274000	0.206572770000000000	0.8546
			50%	0.040926046856969000	0.295774647900000000	0.9586
			55%	0.014491846100172600	0.633802816900000000	0.9843
			60%	0.002477926771449300	0.962441314600000000	0.9957

Figure 4: Spreadsheet results from Final Design Solution

3.0 Testing and Verification (author: S. Kaif)

The table below outlines several requirements the team considered when creating the final design. Furthermore, it has a method of testing each requirement, and the verification result and proof. Each requirement and its proceedings will be expanded upon following the table.

No	Requirement	Verification Method	Verification Result and Proof
1	Detect fraudulent credit card transactions using ensemble learning	Evaluation of final design	PASS, Verified, functional model. See Testing Document.
2.1	Minimize the amount of false positives to 20-30% or lower	TEST: Confusion Matrix	PASS, 4.10%. See Section 3 for calculations and Testing Document.
2.2	Achieve an accuracy of 95% or higher	TEST: Confusion Matrix	PASS, 95.9%. See Section 3 for calculations and Testing Document.
2.3	Achieve a recall score of 50% or higher	TEST: Confusion Matrix	PASS, 70.4%. See Section 3 for calculations.
2.4	Achieve a false negative rate of 30% or lower	TEST: Confusion Matrix	PASS, 29.6%. See Section 3 for calculations.
3	Process at least 65,000 transactions per second	TEST: Time stamps upon model evaluation	FAIL, 46,257 transactions processed per second. See Section 3 and Testing Document.
4	Adherence to “Protection of Personal Information” [8]	Direct usage of a simulated credit card dataset [11]	Pass, Verified. See Section 3 for details.

Table 5: Verification for Design Requirements

In order to acquire confusion matrix metrics, the team compared the results of the model's prediction (y_pred) to the actual test data result (y_true), illustrated in Figure 5. The "1" and "0" represent binary true or false predictions and results, in this case, fraud or non-fraud. For example, for false positives, indicated as "fp", the model's prediction will be a fraud and the actual result is non-fraud. Based on these calculations for 1000 batches of test data, which summates to 112,000 rows (as it is a product of 1000 batches, 7 sequential length, and 16 batch), the confusion matrix with the frequency of true positives, false positives, true negatives, and false negatives is illustrated in Figure 6.

```
def evaluate_model(y_true, y_pred):
    tp = np.sum((y_true == 1) & (y_pred == 1))
    tn = np.sum((y_true == 0) & (y_pred == 0))
    fp = np.sum((y_true == 0) & (y_pred == 1))
    fn = np.sum((y_true == 1) & (y_pred == 0))
```

Figure 5: Code to Acquire Confusion Matrix Metrics

3.1 Verification Using Confusion Matrix (author: S. Kaif)

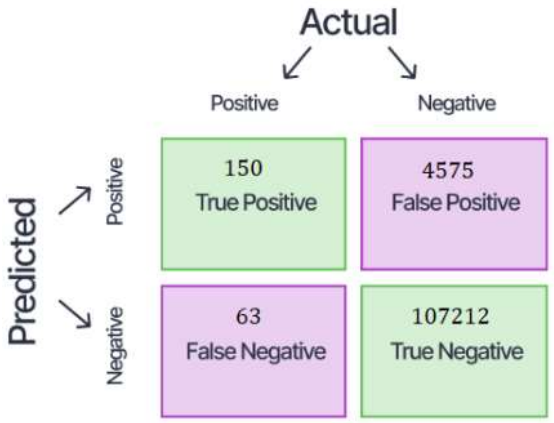

Figure 6: Confusion Matrix for Model's Test over 1000 Batches of Test Data

Based on the confusion matrix in Figure 6, the four metrics (accuracy, false positive rate, recall, and false negative rate) were calculated for evaluation purposes.

The calculation for accuracy, which was 95.9%, better than the 95.0% target objective is below:

$$\text{Accuracy} = \frac{True\ Positives + True\ Negatives}{True\ Positives + True\ Negatives + \ False\ Positives + False\ Negatives}$$
$$= \frac{150 + 107212}{150 + 107212 + 4575 + 63} = 95.9\%$$

The calculation for false positive rate, which was 4.10%, better than the 20-30% minimization objective is below:

$$\text{False Positive Rate} = \frac{False\ Positives}{False\ Positives + True\ Negatives}$$
$$= \frac{4575}{4575 + 107212} = 4.10\%$$

Recall is a new objective the team introduced in the latter stages of testing. The team realized that the model should be predicting more true positives compared to false negatives in order to quantify the model's fraud detection success rate. This recall objective also ties in with the functional goal to predict fraudulent transactions correctly in the first place. The calculation for recall, which was 70.4%, better than the 50% target objective is below:

$$\text{Recall} = \frac{True\ Positives}{True\ Positives + False\ Negatives}$$
$$= \frac{150}{150 + 63} = 70.4\%$$

The team also believed introducing a false negative rate objective would be beneficial for the merchant. The team did not want to minimize false positives at the expense of false negatives. Both rates should be below 30% in order to be better than industry standards and economical. The calculation for false negative rate, which was 29.6%, better than the 30% minimization objective is below:

$$\text{False Negative Rate} = \frac{False\ Negatives}{False\ Negatives + True\ Positives}$$
$$= \frac{63}{63 + 150} = 29.6\%$$

3.2 Verification Processing Speed (author: K. Patel)

The team set an objective to process 65,000 transactions per second to match Visa's current processing speed [12]. The team tested the model over 10 iterations and the processing speed ranged from 45,103 transactions per second to 47,339 transactions per second; see Figure 7. On average, the model processes 46,257 transactions per second, falling short of Visa's current standard by almost 20,000 transactions per second. However, this is an objective the team would rather fail over the confusion matrix objectives since the team operated entirely on Google Colab's free platform, having access to only 12.1 GB of RAM (Random Access Memory). Furthermore, the team valued processing speed lower when creating the weighted decision matrix between design alternatives in the implementation plan (see Appendix A). Even though the team could have passed this objective using an AdaBoost model with 2 or 5 estimators, the team sacrificed this objective by using an AdaBoost model with 10 estimators as it was able to pass the confusion matrix objectives thoroughly.

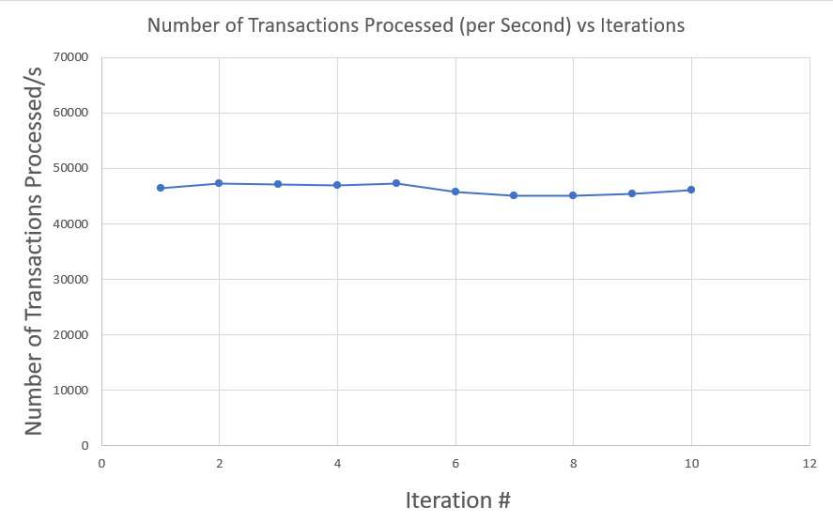

Figure 7: Processing Speed over 10 Iterations of Model on Test Data

3.3 Verification against Privacy Constraint (author: S. Sivayogalingam)

In order to adhere to "Protection of Personal Information" [8], the team used a simulated credit card dataset that was open source from IBM (https://github.com/IBM/TabFormer) [11]. The data protects user identities as it replaces the user name, card number and merchant name with integers and floats. An example of the privacy measures is illustrated in Figure 8.

User	Card	Merchant Name
0	0	3527213246127876953
0	0	-727612092139916043
0	0	-727612092139916043
0	0	3414527459579106770
0	0	5817218446178736267
...	...	...
1338	1	-2088492411650162548
1338	1	-6259030734646478313
1338	1	-2088492411650162548
1338	1	-2088492411650162548
1338	1	-2088492411650162548

Figure 8: Privacy Measures Taken for User, Card Number, and Merchant Name

4.0 Summary and Conclusion (author: S. Sivayogalingam)

The main goal of this project was to create an ensemble machine learning model to minimize the number of false positives detected in a credit card dataset to at least 20-30%. The final design solution has successfully achieved that metric while also passing the majority of the team's outlined objectives. The team is aware that the testing and validation could have accounted for more specific parameters to further assist in determining the most optimal model.

Specifically, different types of estimators could have been tested to see the changes in those results but with other parameters (learning rate and thresholds) the team is confident in their patterns and the finalized values for them.

The only idea that did not work out for the final model was implementing the AdaBoost learning algorithm with LSTMs as its base estimators. This would have been an ideal way for the team to build on top of the current state of the art, however LSTMs are not considered weak learners due to their ability to model complex relationships. As a result, the team had to change the design approach to use decision trees in order to complete the AdaBoost model.

Key takeaways from this project is that the AdaBoost ensemble learning algorithm showed that the promise it had based on other similar detection implementations translates well into the field of credit card fraud. It is able to outperform current state of the art machine learning models in classifying fraudulent and legitimate transactions. The final design solution can potentially be used in the backend of a bank system to combat fraud. Future continuations of this project could involve developing on top of this model for it to be able to detect fraud in other similar financial sectors such as with insurances or loans that also could be dealing with similar fraud related issues. A more simple near-future update could be building a graphical interface which bank employees who may not be experienced reading outputs from a coding standpoint would be able to understand and use.

5.0 References

[1] M. Brewster, “Annual Retail Trade Survey Shows Impact of Online Shopping on Retail Sales During COVID-19 Pandemic,” *Census.gov*, 26-Apr-2022. [Online]. Available: https://www.census.gov/library/stories/2022/04/ecommerce-sales-surged-during-pandemic.html.

[2] Vantage Market Research, “Global $72+ Bn Credit Card Payment Market is Expected to Grow at a CAGR of over 7.4% During 2022-2028,” *GlobeNewswire News Room*, 11-Apr-2022. [Online]. Available:https://www.globenewswire.com/en/news-release/2022/04/11/2419928/0/en/Global-72-Bn-Credit-Card-Payment-Market-is-Expected-to-Grow-at-a-CAGR-of-over-7-4-During-2022-2028-Vantage-Market-Research.html.

[3] F. Wallny, “False Positives in Credit Card Fraud Detection: Measurement and Mitigation,” 2022. [Online]. Available: https://scholarspace.manoa.hawaii.edu/.

[4] K. N. Khasawneh, M. Ozsoy, C. Donovick, N. Abu-Ghazaleh, and D. Ponomarev, “Ensemble learning for low-level hardware-supported malware detection,” *Research in Attacks, Intrusions, and Defenses*, pp. 3–25, 2015.

[5] S. Liu, Y. Wang, J. Zhang, C. Chen, and Y. Xiang, “Addressing the class imbalance problem in Twitter spam detection using Ensemble Learning,” *ScienceDirect*, 13-Dec-2016. [Online]. Available: https://reader.elsevier.com/reader/sd/pii/S0167404816301754?token=39F0203C8C0EBCE1DCDE4495D153A4678D27E055F52E742164DF963E5A26E2FE66C0291FB67B1909BA70F32E70EABD4D&originRegion=us-east-1&originCreation=20220923030108.

[6] R. Bin Sulaiman, V. Schetinin, and P. Sant, “Review of Machine Learning Approach on credit card fraud detection,” *Human-Centric Intelligent Systems*, vol. 2, no. 1-2, pp. 55–68, 2022.

[7] “Visanet: Global Electronic Payment Network,” *Visa*. [Online]. Available: https://www.visa.ca/en_CA/about-visa/visanet.html.

[8] “Protection of Personal Information,” *Bank of Canada*. [Online]. Available: https://www.bankofcanada.ca/privacy/protection-of-personal-information/.

[9] K. Shenoy, “Credit Card Transactions Fraud Detection Dataset,” *Kaggle*, 05-Aug-2020. [Online]. Available: https://www.kaggle.com/datasets/kartik2112/fraud-detection?select=fraudTrain.csv.

[10] A. Biswal, “What is bagging in machine learning and how to perform bagging,” *Simplilearn.com*, 30-Nov-2022. [Online]. Available: https://www.simplilearn.com/tutorials/machine-learning-tutorial/bagging-in-machine-learning#what_is_bagging_in_machine_learning. [Accessed: 09-Jan-2023].

[11] IBM, “IBM/Tabformer: Code & data for ‘Tabular transformers for modeling multivariate time series’ (ICASSP, 2021),” *GitHub*. [Online]. Available: https://github.com/IBM/TabFormer.

[12] “Visanet: Global Electronic Payment Network,” *Visa*. [Online]. Available: https://www.visa.ca/en_CA/about-visa/visanet.html.

6.0 Appendices

6.1 Appendix A: Weighted Decision Matrix in Implementation Plan (author: S. Kaif)

Objective	Weight	Design 1 - SVM	Design 2 - Bagging	Design 3 - AdaBoost
Minimize the amount of false positives	0.6	6	8	9
Be accurate	0.3	7	8	9
Processing speed	0.1	10	7	6
Total	1	6.7	7.9	8.7

Table 6: Weighted Decision Matrix of Three Design Alternatives

6.2 Appendix B: Graphical Results for 100 test batches (authors: A. Khan & K. Patel)

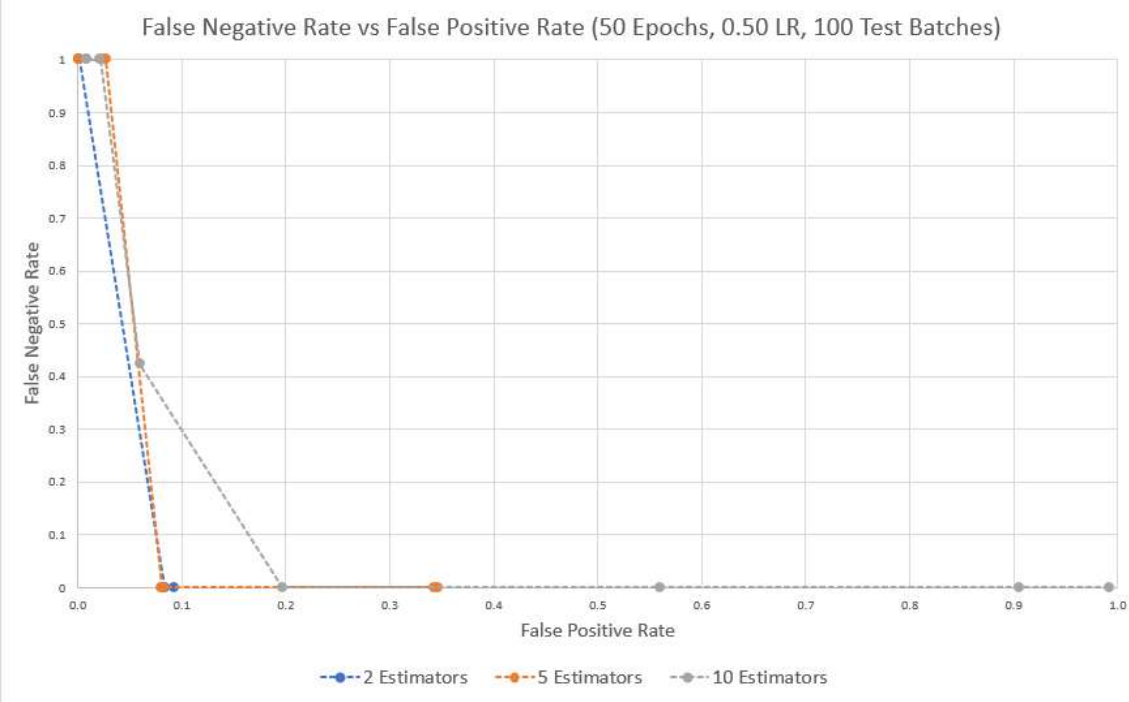

Figure 9: False Negatives vs False Positives (learning rate of 0.50 varying the thresholds)

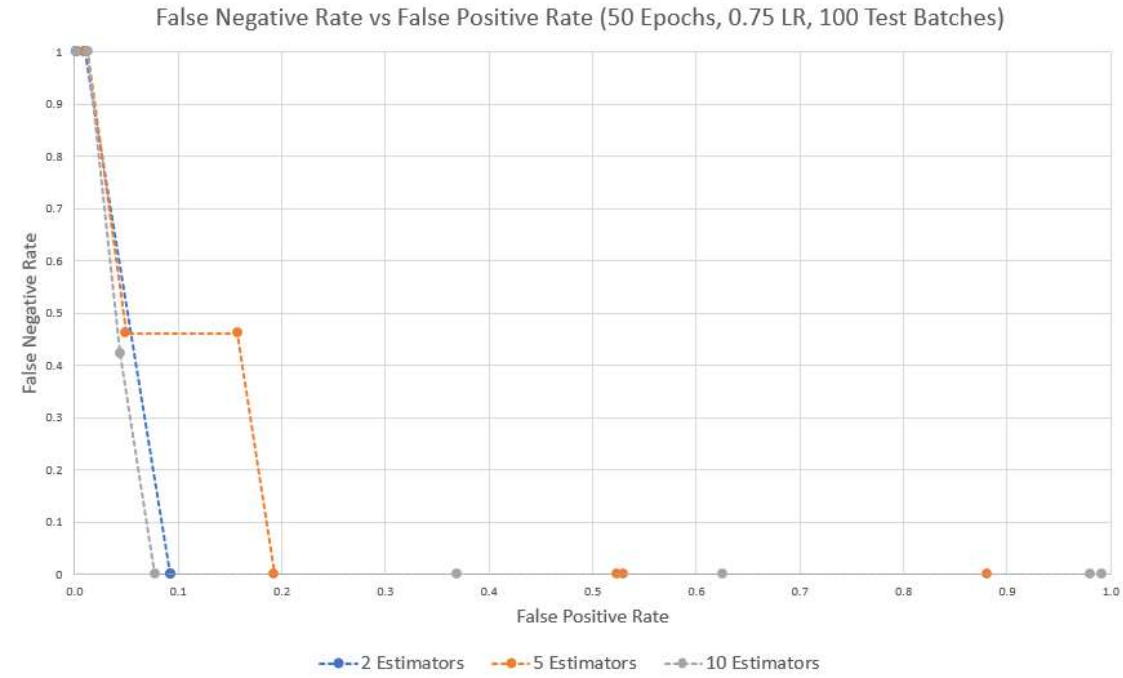

Figure 10: False Negatives vs False Positives (learning rate of 0.75 varying the thresholds)

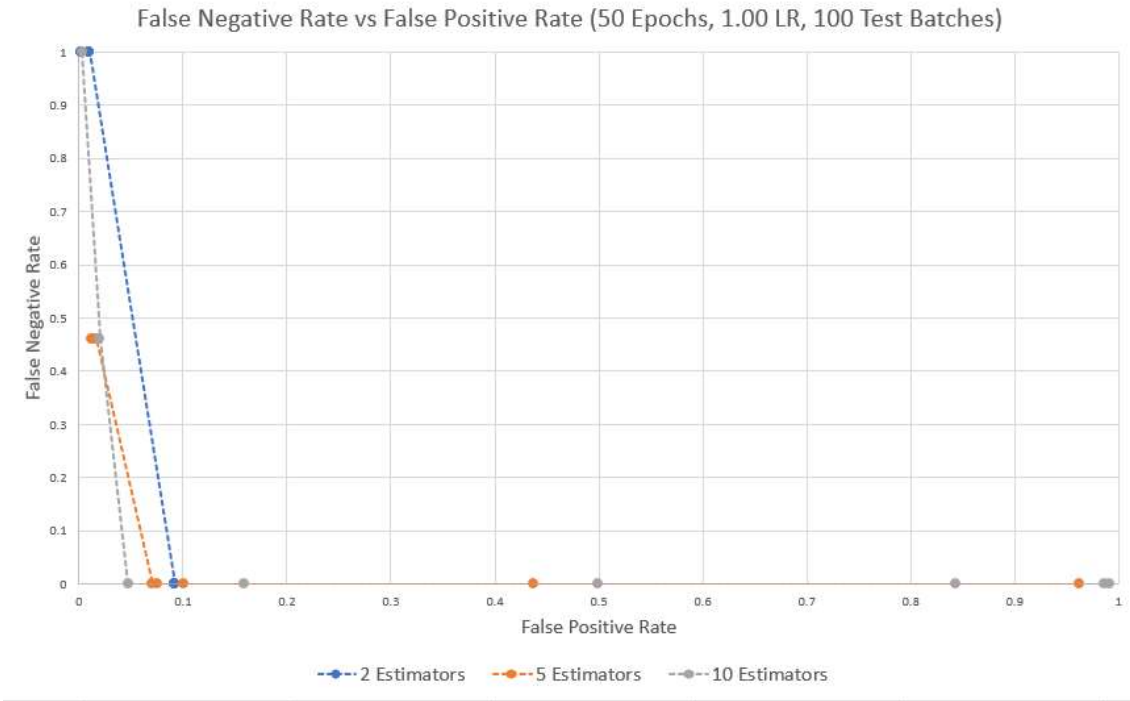

Figure 11: False Negatives vs False Positives (learning rate of 1.00 varying the thresholds)

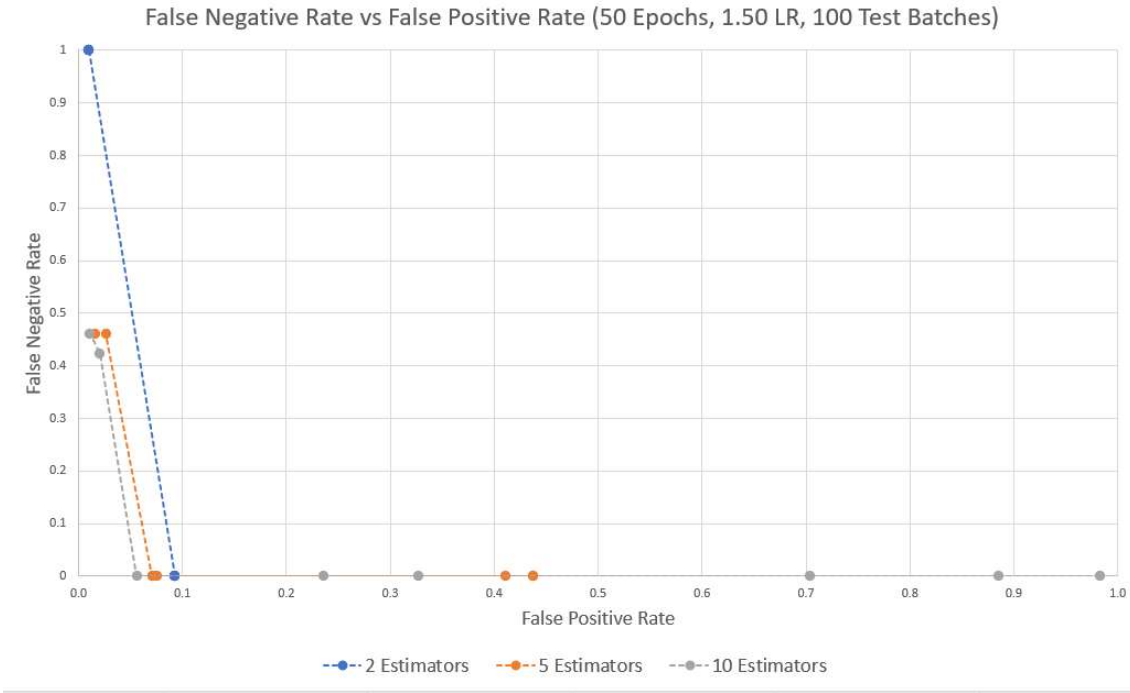

Figure 12: False Negatives vs False Positives (learning rate of 1.50 varying the thresholds)

Figure 13: False Negatives vs False Positives (learning rate of 5.00 varying the thresholds)

6.2 Appendix C: Graphical Results for 300 test batches (authors: A. Khan & K. Patel)

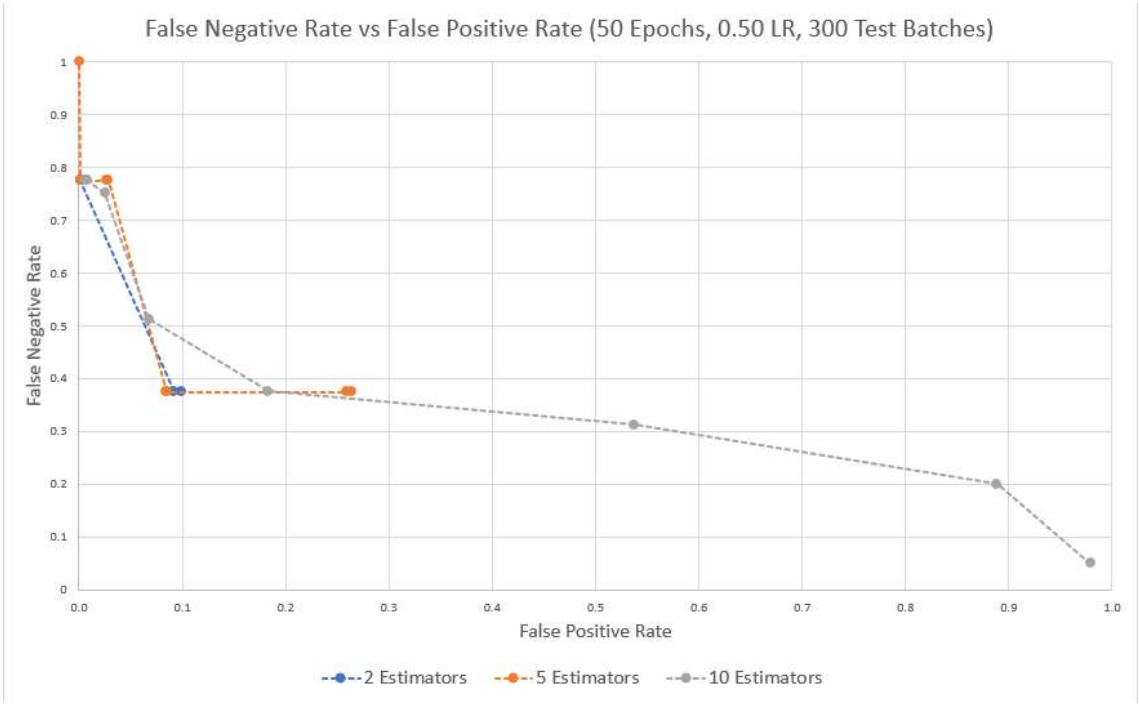

Figure 14: False Negatives vs False Positives (learning rate of 0.50 varying the thresholds)

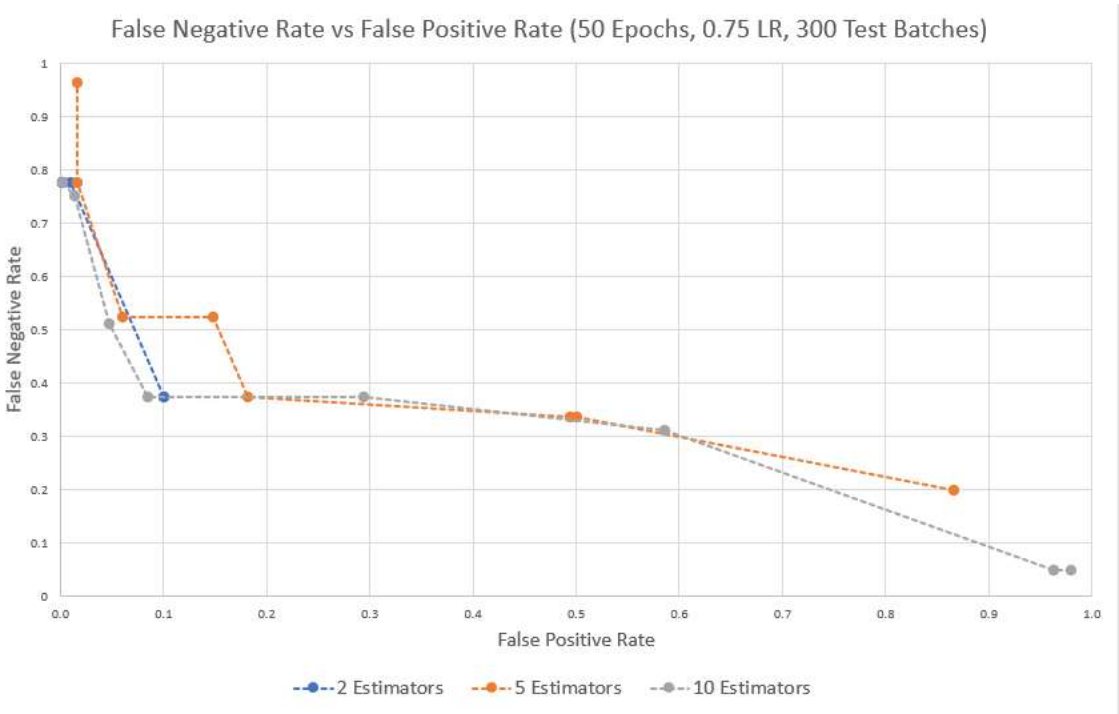

Figure 15: False Negatives vs False Positives (learning rate of 0.75 varying the thresholds)

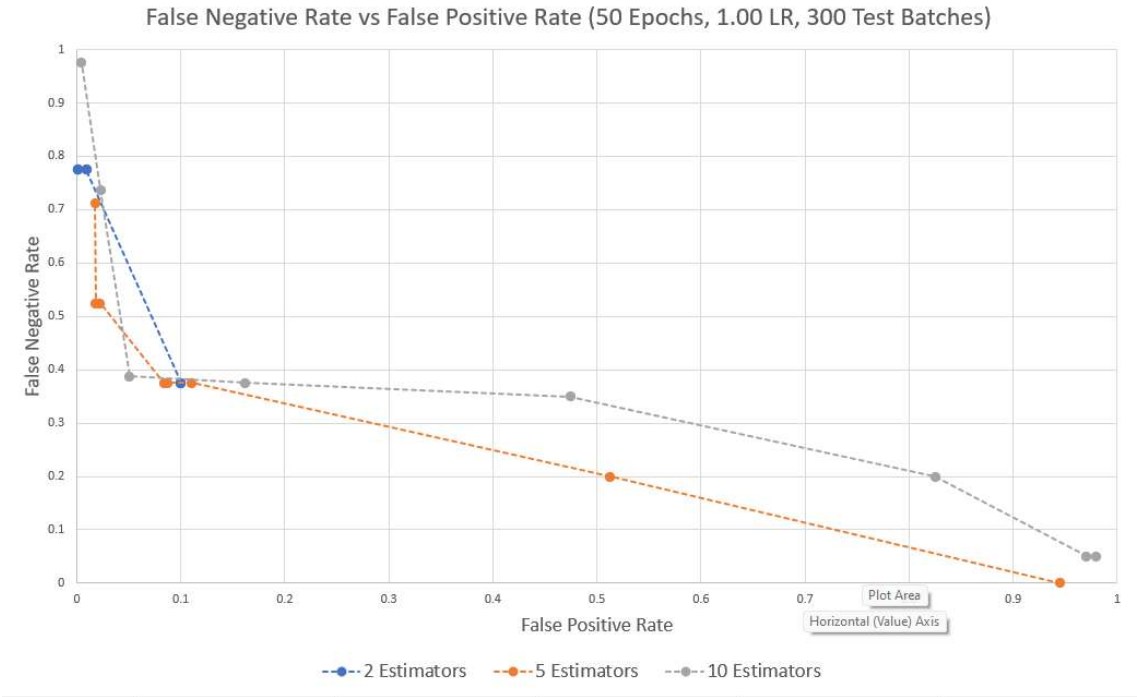

Figure 16: False Negatives vs False Positives (learning rate of 1.00 varying the thresholds)

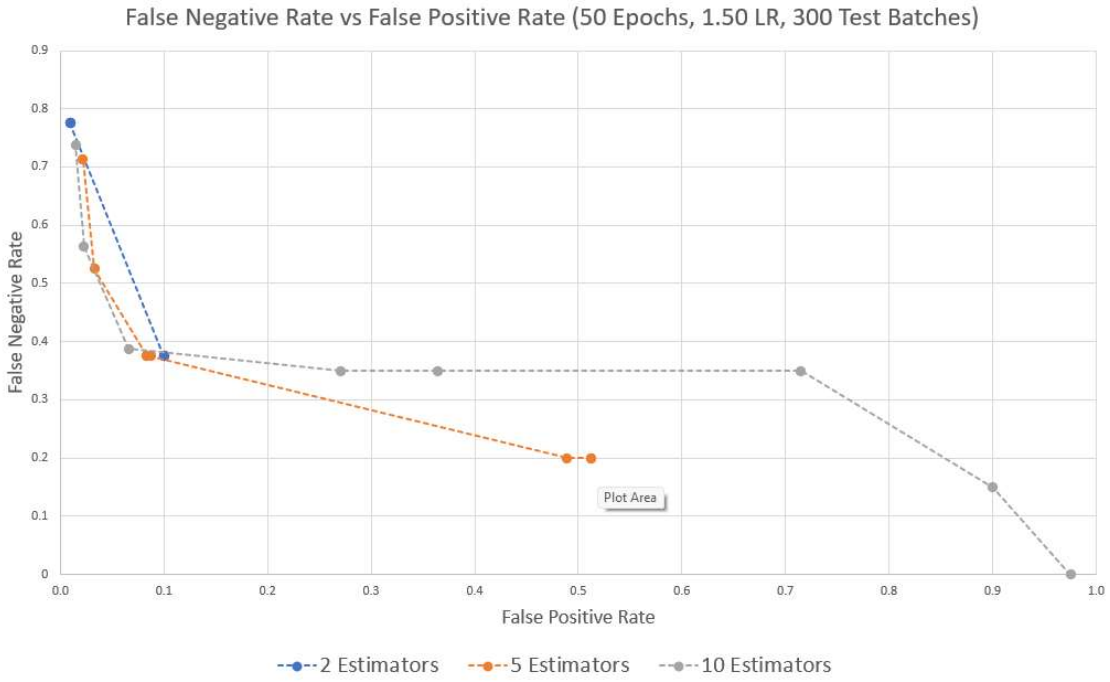

Figure 17: False Negatives vs False Positives (learning rate of 1.50 varying the thresholds)

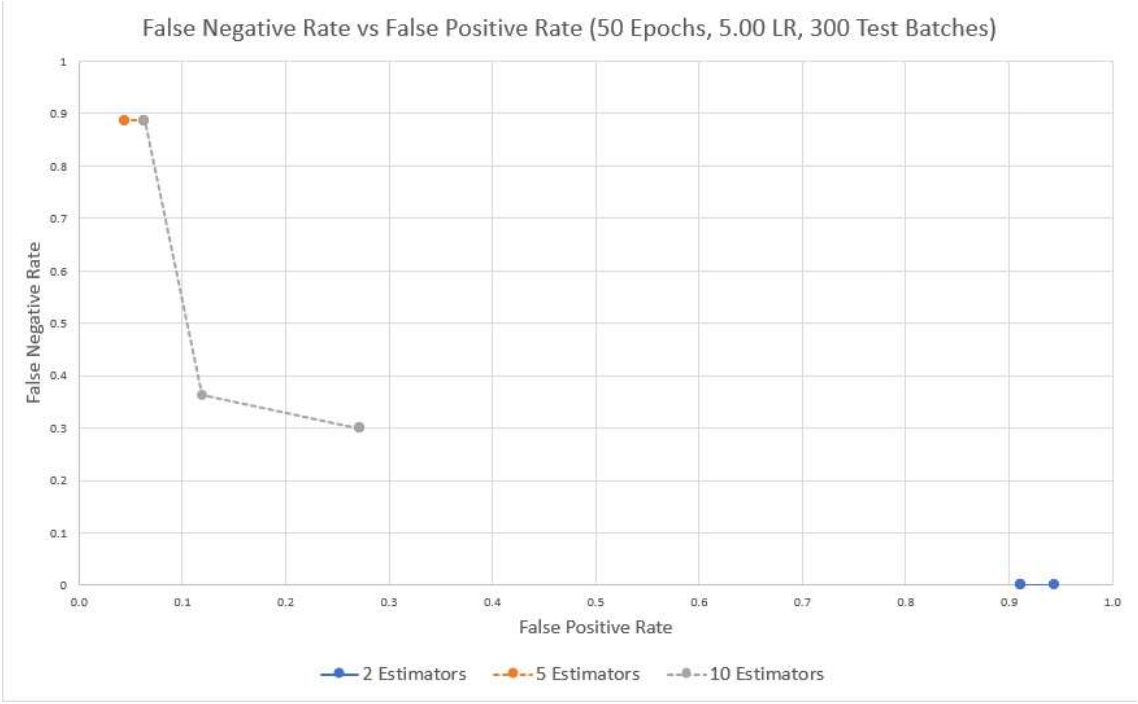

Figure 18: False Negatives vs False Positives (learning rate of 5.00 varying the thresholds)

6.2 Appendix D: Graphical Results for 1000 test batches (authors: A. Khan & K. Patel)

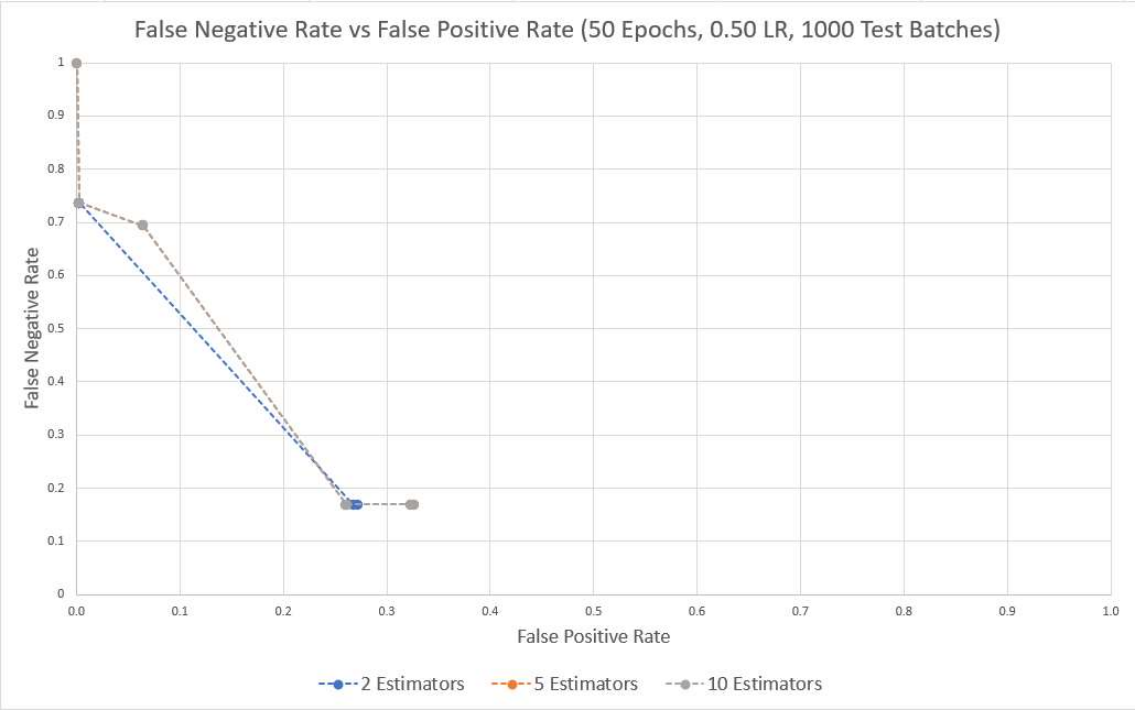

Figure 18: False Negatives vs False Positives (learning rate of 0.50 varying the thresholds)

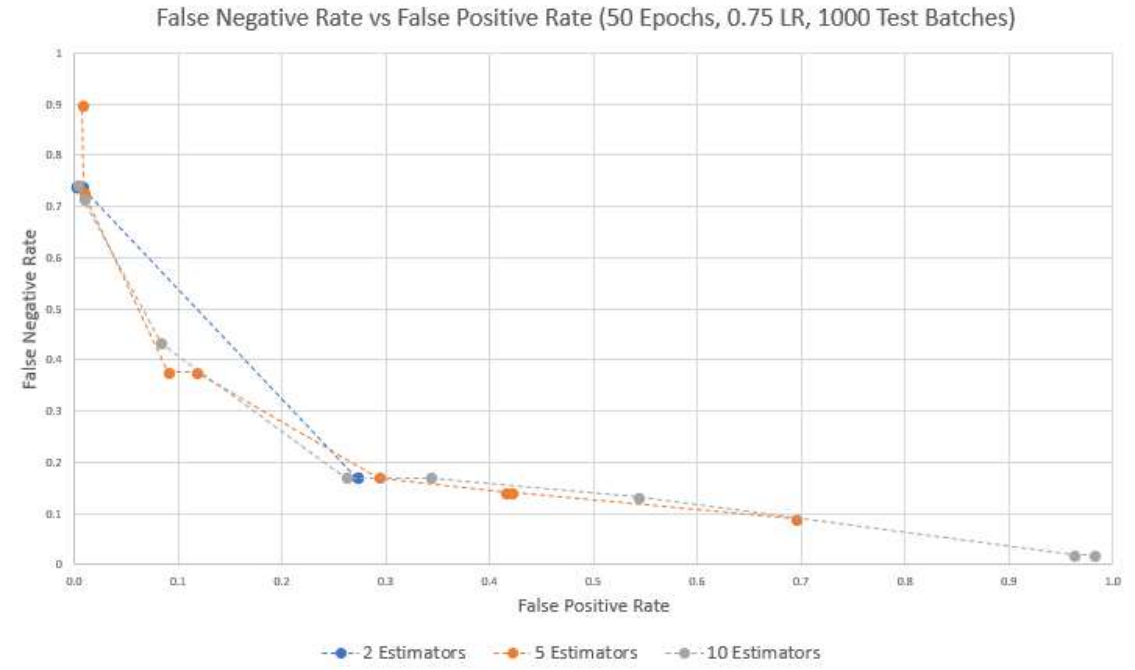

Figure 19: False Negatives vs False Positives (learning rate of 0.75 varying the thresholds)

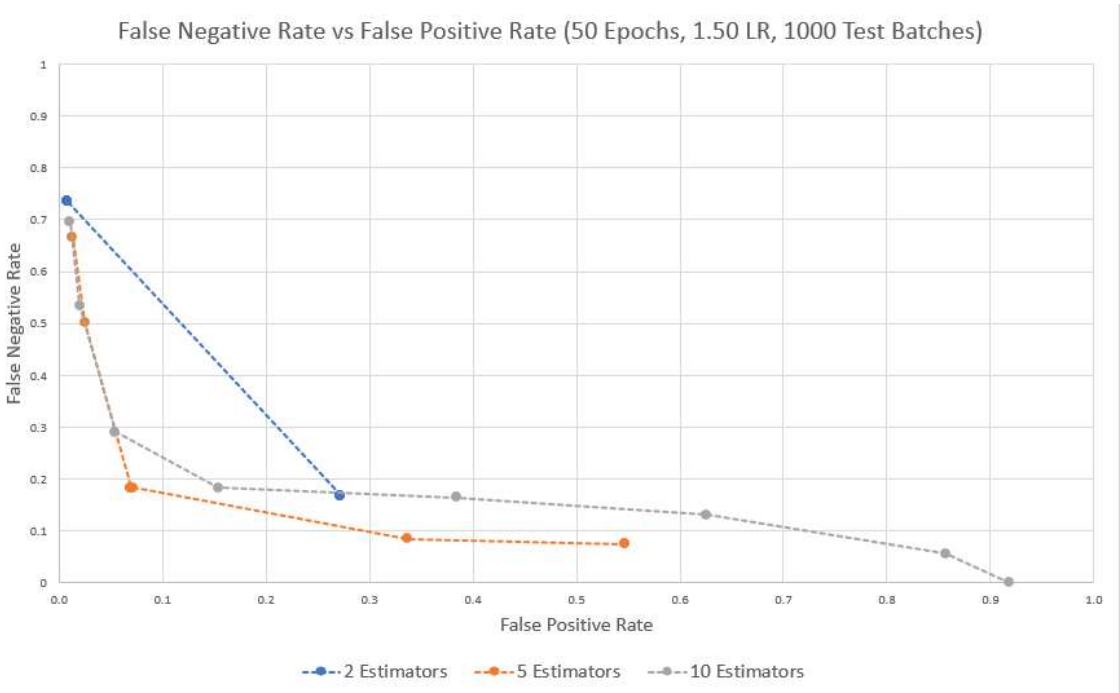

Figure 20: False Negatives vs False Positives (learning rate of 1.50 varying the thresholds)

6.3 Appendix E: Spreadsheets of data collected (author: Equally amongst all members)

Testing Parameters		Results		
Hyperparameters	Threshold	FP Rate	FN Rate	Average Accuracy
Learning Rate = 0.50 Estimators = 2 Test Batches = 100	25%	0.092357257920171800	0	N/A
	30%	0.083855378557365300	0	N/A
	35%	0.001431895471630570	1	N/A
	40%	0.001431895471630570	1	N/A
	45%	0.001431895471630570	1	N/A
	50%	0.001431895471630570	1	N/A
	55%	0.001431895471630570	1	N/A
	60%	0.001431895471630570	1	N/A
Learning Rate = 0.50 Estimators = 5 Test Batches = 100	25%	0.345892249865759000	0	N/A
	30%	0.342402004653660000	0	N/A
	35%	0.082781456953642300	0	N/A
	40%	0.080723107213173400	0	N/A
	45%	0.026937533560050100	1	N/A
	50%	0.023984249149812000	1	N/A
	55%	0.001431895471630570	1	N/A
	60%	0.000715947735815285	1	N/A
Learning Rate = 0.50 Estimators = 10 Test Batches = 100	25%	0.991677107571147000	0	N/A
	30%	0.905405405405405000	0	N/A
	35%	0.559871129407553000	0	N/A
	40%	0.196527653481295000	0	N/A
	45%	0.059960622874530100	0.4230769231	N/A
	50%	0.021836405942366200	1	N/A
	55%	0.008322892428852690	1	N/A
	60%	0.008322892428852690	1	N/A

Figure 21: False Positive & False Negative Rates (LR = 0.50, Test Batches = 100, w/ varying thresholds)

Testing Parameters		Results		
Hyperparameters	Threshold	FP Rate	FN Rate	Average Accuracy
Learning Rate = 0.75 Estimators = 2 Test Batches = 100	25%	0.092357257920171800	0	N/A
	30%	0.092357257920171800	0	N/A
	35%	0.009933774834437080	1	N/A
	40%	0.001431895471630570	1	N/A
	45%	0.001431895471630570	1	N/A
	50%	0.001431895471630570	1	N/A
	55%	0.001431895471630570	1	N/A
	60%	0.001431895471630570	1	N/A
Learning Rate = 0.75 Estimators = 5 Test Batches = 100	25%	0.881331662788616000	0	N/A
	30%	0.529622337569357000	0	N/A
	35%	0.522999821013066000	0	N/A
	40%	0.192679434401288000	0	N/A
	45%	0.157597995346339000	0.4615384615	N/A
	50%	0.049847861106139200	0.4615384615	N/A
	55%	0.012529085376767400	1	N/A
	60%	0.010918202971183100	1	N/A
Learning Rate = 0.75 Estimators = 10 Test Batches = 100	25%	0.991677107571147000	0	N/A
	30%	0.980937891533918000	0	N/A
	35%	0.626006801503490000	0	N/A
	40%	0.369429031680687000	0	N/A
	45%	0.077590835868981500	0	N/A
	50%	0.044299266153570700	0.4230769231	N/A
	55%	0.013066046178628900	1	N/A
	60%	0.001610882405584390	1	N/A

Figure 22: False Positive & False Negative Rates (LR = 0.75, Test Batches = 100, w/ varying thresholds)

Testing Parameters		Results		
Hyperparameters	Threshold	FP Rate	FN Rate	Average Accuracy
Learning Rate = 1.00 Estimators = 2 Test Batches = 100	25%	0.092357258	0	N/A
	30%	0.092357258	0	N/A
	35%	0.092357258	0	N/A
	40%	0.009933775	1	N/A
	45%	0.009933775	1	N/A
	50%	0.001431895	1	N/A
	55%	0.001431895	1	N/A
	60%	0.001431895	1	N/A
Learning Rate = 1.00 Estimators = 5 Test Batches = 100	25%	0.962412744	0	N/A
	30%	0.436907106	0	N/A
	35%	0.101306605	0	N/A
	40%	0.074995525	0	N/A
	45%	0.070699839	0	N/A
	50%	0.016914265	0.461538462	N/A
	55%	0.012887059	0.461538462	N/A
	60%	0.011992125	0.461538462	N/A
Learning Rate = 1.00 Estimators = 10 Test Batches = 100	25%	0.991677108	0	N/A
	30%	0.98684446	0	N/A
	35%	0.843296939	0	N/A
	40%	0.499552533	0	N/A
	45%	0.159208878	0	N/A
	50%	0.047521031	0	N/A
	55%	0.02013603	0.461538462	N/A
	60%	0.002684804	1	N/A

Figure 23: False Positive & False Negative Rates (LR = 1.00, Test Batches = 100, w/ varying thresholds)

Testing Parameters		Results		
Hyperparameters	Threshold	FP Rate	FN Rate	Average Accuracy
Learning Rate = 1.50 Estimators = 2 Test Batches = 100	25%	0.0923572579201718	0	N/A
	30%	0.0923572579201718	0	N/A
	35%	0.0923572579201718	0	N/A
	40%	0.0923572579201718	0	N/A
	45%	0.0099337748344371	1	N/A
	50%	0.0099337748344371	1	N/A
	55%	0.0099337748344371	1	N/A
	60%	0.0099337748344371	1	N/A
Learning Rate = 1.50 Estimators = 5 Test Batches = 100	25%	0.4369071057812770	0	N/A
	30%	0.4369071057812770	0	N/A
	35%	0.4105960264900660	0	N/A
	40%	0.0749955253266511	0	N/A
	45%	0.0706998389117594	0	N/A
	50%	0.0706998389117594	0	N/A
	55%	0.0263110792912117	0.4615384615	N/A
	60%	0.0160193305888670	0.4615384615	N/A
Learning Rate = 1.50 Estimators = 10 Test Batches = 100	25%	0.9822802935385710	0	N/A
	30%	0.8846429210667620	0	N/A
	35%	0.7039556112403790	0	N/A
	40%	0.3271881152675850	0	N/A
	45%	0.2351888312153210	0	N/A
	50%	0.0561124037945230	0	N/A
	55%	0.0202255235367818	0.4230769231	N/A
	60%	0.0110076964381600	0.4615384615	N/A

Figure 24: False Positive & False Negative Rates (LR = 1.50, Test Batches = 100, w/ varying thresholds)

Testing Parameters		Results		
Hyperparameters	Threshold	FP Rate	FN Rate	Average Accuracy
Learning Rate = 5.00 Estimators = 2 Test Batches = 100	25%	0.9622337569357430	0	N/A
	30%	0.9622337569357430	0	N/A
	35%	0.9378915339180230	0	N/A
	40%	0.9378915339180230	0	N/A
	45%	0.9378915339180230	0	N/A
	50%	0.9378915339180230	0	N/A
	55%	0.9378915339180230	0	N/A
	60%	0.9378915339180230	0	N/A
Learning Rate = 5.00 Estimators = 5 Test Batches = 100	25%	0.0288168963665652	0.8461538462	N/A
	30%	0.0288168963665652	0.8461538462	N/A
	35%	0.0288168963665652	0.8461538462	N/A
	40%	0.0288168963665652	0.8461538462	N/A
	45%	0.0288168963665652	0.8461538462	N/A
	50%	0.0271165204940039	0.8461538462	N/A
	55%	0.0271165204940039	0.8461538462	N/A
	60%	0.0271165204940039	0.8461538462	N/A
Learning Rate = 5.00 Estimators = 10 Test Batches = 100	25%	0.1195632718811520	0	N/A
	30%	0.1195632718811520	0	N/A
	35%	0.1195632718811520	0	N/A
	40%	0.1195632718811520	0	N/A
	45%	0.1195632718811520	0	N/A
	50%	0.1195632718811520	0	N/A
	55%	0.0867191695006264	0	N/A
	60%	0.0288168963665652	0.8461538462	N/A

Figure 25: False Positive & False Negative Rates (LR = 5.00, Test Batches = 100, w/ varying thresholds)

Testing Parameters		Results		
Hyperparameters	Threshold	FP Rate	FN Rate	Average Accuracy
Learning Rate = 0.50 Estimators = 2 Test Batches = 300	25%	0.099821002386634800	0.375	N/A
	30%	0.091974940334128800	0.375	N/A
	35%	0.001491646778042950	0.775	N/A
	40%	0.001491646778042950	0.775	N/A
	45%	0.001491646778042950	0.775	N/A
	50%	0.001491646778042950	0.775	N/A
	55%	0.001491646778042950	0.775	N/A
	60%	0.001491646778042950	0.775	N/A
Learning Rate = 0.50 Estimators = 5 Test Batches = 300	25%	0.264021479713603000	0.375	N/A
	30%	0.259427207637231000	0.375	N/A
	35%	0.086068019093078700	0.375	N/A
	40%	0.084397374701670600	0.375	N/A
	45%	0.028371121718377000	0.775	N/A
	50%	0.026789976133651500	0.775	N/A
	55%	0.001491646778042950	0.775	N/A
	60%	0.000387828162291169	1	N/A
Learning Rate = 0.50 Estimators = 10 Test Batches = 300	25%	0.979116945107398000	0.05	N/A
	30%	0.888842482100238000	0.2	N/A
	35%	0.537589498806682000	0.3125	N/A
	40%	0.182995226730310000	0.375	N/A
	45%	0.068019093078758900	0.5125	N/A
	50%	0.025178997613365100	0.75	N/A
	55%	0.008591885441527440	0.775	N/A
	60%	0.007368735083532220	0.775	N/A

Figure 26: False Positive & False Negative Rates (LR = 0.50, Test Batches = 300, w/ varying thresholds)

Testing Parameters		Results		
Hyperparameters	Threshold	FP Rate	FN Rate	Average Accuracy
Learning Rate = 0.75 Estimators = 2 Test Batches = 300	25%	0.099821002386634800	0.375	N/A
	30%	0.099821002386634800	0.375	N/A
	35%	0.009337708830548920	0.775	N/A
	40%	0.001491646778042950	0.775	N/A
	45%	0.001491646778042950	0.775	N/A
	50%	0.001491646778042950	0.775	N/A
	55%	0.001491646778042950	0.775	N/A
	60%	0.001491646778042950	0.775	N/A
Learning Rate = 0.75 Estimators = 5 Test Batches = 300	25%	0.865632458233890000	0.2	N/A
	30%	0.500686157517899000	0.3375	N/A
	35%	0.494272076372315000	0.3375	N/A
	40%	0.181354415274463000	0.375	N/A
	45%	0.148418854415274000	0.525	N/A
	50%	0.060023866348448600	0.525	N/A
	55%	0.016855608591885400	0.775	N/A
	60%	0.015900954653937900	0.9625	N/A
Learning Rate = 0.75 Estimators = 10 Test Batches = 300	25%	0.979116945107398000	0.05	N/A
	30%	0.963484486873508000	0.05	N/A
	35%	0.585859188544152000	0.3125	N/A
	40%	0.294212410501193000	0.375	N/A
	45%	0.084278042959427200	0.375	N/A
	50%	0.047374701670644300	0.5125	N/A
	55%	0.013484486873508300	0.75	N/A
	60%	0.002207637231503580	0.775	N/A

Figure 27: False Positive & False Negative Rates (LR = 0.75, Test Batches = 300, w/ varying thresholds)

Testing Parameters		Results		
Hyperparameters	Threshold	FP Rate	FN Rate	Average Accuracy
Learning Rate = 1.00 Estimators = 2 Test Batches = 300	25%	0.099821002	0.375	N/A
	30%	0.099821002	0.375	N/A
	35%	0.099821002	0.375	N/A
	40%	0.009337709	0.77500	N/A
	45%	0.009337709	0.77500	N/A
	50%	0.001491647	0.77500	N/A
	55%	0.001491647	0.77500	N/A
	60%	0.001491647	0.77500	N/A
Learning Rate = 1.00 Estimators = 5 Test Batches = 300	25%	0.94522673	0	N/A
	30%	0.5125	0.20000	N/A
	35%	0.110292363	0.37500	N/A
	40%	0.087112172	0.37500	N/A
	45%	0.083651551	0.37500	N/A
	50%	0.022195704	0.52500	N/A
	55%	0.018257757	0.52500	N/A
	60%	0.017601432	0.71250	N/A
Learning Rate = 1.00 Estimators = 10 Test Batches = 300	25%	0.979116945	0.05000	N/A
	30%	0.97022673	0.05000	N/A
	35%	0.824940334	0.20000	N/A
	40%	0.474194511	0.35000	N/A
	45%	0.161903341	0.37500	N/A
	50%	0.050924821	0.38750	N/A
	55%	0.023060859	0.73750	N/A
	60%	0.004295943	0.97500	N/A

Figure 28: False Positive & False Negative Rates (LR = 1.00, Test Batches = 300, w/ varying thresholds)

Testing Parameters		Results		
Hyperparameters	Threshold	FP Rate	FN Rate	Average Accuracy
Learning Rate = 1.50 Estimators = 2 Test Batches = 300	25%	0.0998210023866348	0.375	N/A
	30%	0.0998210023866348	0.375	N/A
	35%	0.0998210023866348	0.375	N/A
	40%	0.0998210023866348	0.375	N/A
	45%	0.0093377088305489	0.775	N/A
	50%	0.0093377088305489	0.775	N/A
	55%	0.0093377088305489	0.775	N/A
	60%	0.0093377088305489	0.775	N/A
Learning Rate = 1.50 Estimators = 5 Test Batches = 300	25%	0.5125596658711210	0.2	N/A
	30%	0.5125000000000000	0.2	N/A
	35%	0.4893496420047730	0.2	N/A
	40%	0.0871420047732697	0.375	N/A
	45%	0.0836515513126491	0.375	N/A
	50%	0.0836515513126491	0.375	N/A
	55%	0.0330250596658711	0.525	N/A
	60%	0.0215393794749403	0.7125	N/A
Learning Rate = 1.50 Estimators = 10 Test Batches = 300	25%	0.9750596658711210	0	N/A
	30%	0.9003281622911690	0.15	N/A
	35%	0.7150656324582330	0.35	N/A
	40%	0.3641109785202860	0.35	N/A
	45%	0.2703460620525050	0.35	N/A
	50%	0.0659307875894988	0.3875	N/A
	55%	0.0227028639618138	0.5625	N/A
	60%	0.0147673031026252	0.7375	N/A

Figure 29: False Positive & False Negative Rates (LR = 1.50, Test Batches = 300, w/ varying thresholds)

Testing Parameters		Results		
Hyperparameters	Threshold	FP Rate	FN Rate	Average Accuracy
Learning Rate = 5.00 Estimators = 2 Test Batches = 300	25%	0.9443019093078750	0	N/A
	30%	0.9443019093078750	0	N/A
	35%	0.9113961813842480	0	N/A
	40%	0.9113961813842480	0	N/A
	45%	0.9113961813842480	0	N/A
	50%	0.9113961813842480	0	N/A
	55%	0.9113961813842480	0	N/A
	60%	0.9113961813842480	0	N/A
Learning Rate = 5.00 Estimators = 5 Test Batches = 300	25%	0.0634844868735083	0.8875	N/A
	30%	0.0634844868735083	0.8875	N/A
	35%	0.0634844868735083	0.8875	N/A
	40%	0.0634844868735083	0.8875	N/A
	45%	0.0634844868735083	0.8875	N/A
	50%	0.0440930787589498	0.8875	N/A
	55%	0.0440930787589498	0.8875	N/A
	60%	0.0440930787589498	0.8875	N/A
Learning Rate = 5.00 Estimators = 10 Test Batches = 300	25%	0.2714797136038180	0.3	N/A
	30%	0.2714797136038180	0.3	N/A
	35%	0.2714797136038180	0.3	N/A
	40%	0.2714797136038180	0.3	N/A
	45%	0.2714797136038180	0.3	N/A
	50%	0.2714797136038180	0.3	N/A
	55%	0.1194809069212410	0.3625	N/A
	60%	0.0634844868735083	0.8875	N/A

Figure 30: False Positive & False Negative Rates (LR = 5.00, Test Batches = 300, w/ varying thresholds)

Testing Parameters		Results		
Hyperparameters	Threshold	FP Rate	FN Rate	Average Accuracy
Learning Rate = 0.50 Estimators = 2 Test Batches = 1000	25%	0.272106774490772000	0.1690140845	N/A
	30%	0.266980954851637000	0.1690140845	N/A
	35%	0.002433198851387010	0.7370892019	N/A
	40%	0.002433198851387010	0.7370892019	N/A
	45%	0.002433198851387010	0.7370892019	N/A
	50%	0.002433198851387010	0.7370892019	N/A
	55%	0.002433198851387010	0.7370892019	0.9962
	60%	0.002433198851387010	0.7370892019	0.9962
Learning Rate = 0.50 Estimators = 5 Test Batches = 1000	25%	0.325789224149498000	0.1690140845	0.6745
	30%	0.322452521312853000	0.1690140845	0.6778
	35%	0.261524148604041000	0.1690140845	0.7387
	40%	0.260253875674273000	0.1690140845	0.7399
	45%	0.064282966713481900	0.6948356808	0.9345
	50%	0.063764122840759600	0.6948356808	0.935
	55%	0.002433198851387010	0.7370892019	0.9962
	60%	0.000429388032597708	1.0000000000	0.9977
Learning Rate = 0.50 Estimators = 10 Test Batches = 1000	25%	0.325789224149498000	0.1690140845	0.6745
	30%	0.322452521312853000	0.1690140845	0.6778
	35%	0.261524148604041000	0.1690140845	0.7387
	40%	0.260253875674273000	0.1690140845	0.7399
	45%	0.064282966713481900	0.6948356808	0.9345
	50%	0.063764122840759600	0.6948356808	0.935
	55%	0.002433198851387010	0.7370892019	0.9962
	60%	0.000429388032597708	1.0000000000	0.9977

Figure 31: False Positive & False Negative Rates (LR = 0.50, Test Batches = 1000, w/ varying thresholds)

Testing Parameters		Results		
Hyperparameters	Threshold	FP Rate	FN Rate	Average Accuracy
Learning Rate = 0.75 Estimators = 2 Test Batches = 1000	25%	0.272106774490772000	0.1690140845070420	N/A
	30%	0.272106774490772000	0.1690140845070420	N/A
	35%	0.007559018490522150	0.7370892018779340	N/A
	40%	0.002433198851387010	0.7370892018779340	N/A
	45%	0.002433198851387010	0.7370892018779340	0.9962
	50%	0.002433198851387010	0.7370892018779340	0.9962
	55%	0.002433198851387010	0.7370892018779340	0.9962
	60%	0.002433198851387010	0.7370892018779340	0.9962
Learning Rate = 0.75 Estimators = 5 Test Batches = 1000	25%	0.694866129335253000	0.0892018779342723	0.3063
	30%	0.421560646586812000	0.1408450704225350	0.579
	35%	0.415933874242979000	0.1408450704225350	0.5846
	40%	0.293084168999973000	0.1690140845070420	0.7072
	45%	0.117017184466887000	0.3755868544600930	0.8825
	50%	0.090010466333294500	0.3755868544600930	0.9094
	55%	0.009097658940663900	0.7276995305164310	0.9895
	60%	0.007353270058235750	0.8967136150234740	0.991
Learning Rate = 0.75 Estimators = 10 Test Batches = 1000	25%	0.982010430550958000	0.0187793427230046	0.0198
	30%	0.963090520364622000	0.0187793427230046	0.0387
	35%	0.543381609668387000	0.1314553990610320	0.4574
	40%	0.343009473373469000	0.1690140845070420	0.6573
	45%	0.261363128091817000	0.1690140845070420	0.7388
	50%	0.083453353252167000	0.4319248826291080	0.9159
	55%	0.008614597403991510	0.7136150234741780	0.99
	60%	0.002674729619723220	0.7417840375586850	0.9959

Figure 32: False Positive & False Negative Rates (LR = 0.75, Test Batches = 1000, w/ varying thresholds)

Testing Parameters		Results		
Hyperparameters	Threshold	FP Rate	FN Rate	Average Accuracy
Learning Rate = 1.00 Estimators = 2 Test Batches = 1000	25%	0.2721067744907720	0.16901408450	0.7281
	30%	0.2721067744907720	0.16901408450	0.7281
	35%	0.2721067744907720	0.16901408450	0.7281
	40%	0.0075590184905222	0.73708920190	0.9911
	45%	0.0075590184905222	0.73708920190	0.9911
	50%	0.0024331988513870	0.73708920190	0.9962
	55%	0.0024331988513870	0.73708920190	0.9962
	60%	0.0024331988513870	0.73708920190	0.9962
Learning Rate = 1.00 Estimators = 5 Test Batches = 1000	25%	0.9591991913192050	0.00000000000	N/A
	30%	0.5470492991134920	0.07511737089	N/A
	35%	0.2824210328571300	0.16901408450	N/A
	40%	0.0720835159723402	0.17840375590	N/A
	45%	0.0695071877767540	0.18309859150	N/A
	50%	0.0146260298603594	0.50234741780	0.988
	55%	0.0123985794412588	0.50234741780	0.988
	60%	0.0107883743190174	0.66666666670	0.988
Learning Rate = 1.00 Estimators = 10 Test Batches = 1000	25%	0.9820104305509580	0.01877934272	0.0198
	30%	0.9271382182185760	0.01877934272	0.0746
	35%	0.7185629813842390	0.07511737089	0.2827
	40%	0.4517162102927880	0.14553990610	0.5489
	45%	0.1453120666982740	0.20657277000	0.8546
	50%	0.0409260468569690	0.29577464790	0.9586
	55%	0.0144918461001726	0.63380281690	0.9843
	60%	0.0024779267714493	0.96244131460	0.9957

Figure 33: False Positive & False Negative Rates (LR = 1.00, Test Batches = 1000, w/ varying thresholds)

Testing Parameters		Results		
Hyperparameters	Threshold	FP Rate	FN Rate	Average Accuracy
Learning Rate = 1.50 Estimators = 2 Test Batches = 1000	25%	0.27210677449077200	0.16901408451	0.7281
	30%	0.27210677449077200	0.16901408451	0.7281
	35%	0.27210677449077200	0.16901408451	0.7281
	40%	0.27210677449077200	0.16901408451	0.7281
	45%	0.00755901849052215	0.73708920188	0.9911
	50%	0.00755901849052215	0.73708920188	0.9911
	55%	0.00755901849052215	0.73708920188	0.9911
	60%	0.00755901849052215	0.73708920188	0.9911
Learning Rate = 1.50 Estimators = 5 Test Batches = 1000	25%	0.54708508144954200	0.07511737089	0.4538
	30%	0.54704929911349200	0.07511737089	0.4538
	35%	0.33672967339672700	0.08450704225	0.6637
	40%	0.07210140714036510	0.18309859155	0.9277
	45%	0.06950718777675400	0.18309859155	0.9303
	50%	0.06950718777675400	0.18309859155	0.9303
	55%	0.02519076457906550	0.50234741784	0.9739
	60%	0.01301582473811800	0.66666666667	0.9857
Learning Rate = 1.50 Estimators = 10 Test Batches = 1000	25%	0.91826419887822300	0.00000000000	0.0835
	30%	0.85648599568822800	0.05633802817	0.145
	35%	0.62563625466288500	0.13145539906	0.3753
	40%	0.38393552023043800	0.16431924883	0.6165
	45%	0.15363145982985500	0.18309859155	0.8463
	50%	0.05452333455589640	0.29107981221	0.945
	55%	0.01974290391548210	0.53521126761	0.9793
	60%	0.01037687745444460	0.69483568075	0.9883

Figure 21: False Positive & False Negative Rates (LR = 1.50, Test Batches = 1000, w/ varying thresholds)