Optimizing Fast Randomized One-Class Classification (FROCC) for Enhanced Binary Classification

Thesis submitted by

Pawan Kumar Rajotiya

2011CS50290

under the guidance of

Prof. Amitabh Bagchi

in partial fulfilment of the requirements

for the award of the degree of

Bachelor and Master of Technology

Department Of Computer Science And Engineering

INDIAN INSTITUTE OF TECHNOLOGY DELHI

July 2024

Abstract

The Fast Randomized One-Class Classification (FROCC) model is a novel and efficient algorithm for binary classification tasks, known for its simplicity and effectiveness. This thesis explores various enhancements and optimizations to the FROCC model, aiming to improve its performance and robustness. The research is structured into several key phases, each addressing different aspects of model optimization and comparison with traditional baseline models.

Initially, the study investigates the impact of tuning key hyperparameters, such as dimension and epsilon, on the performance of the FROCC model. Through extensive experimentation on multiple binary datasets, optimal configurations are identified, leading to significant performance gains. A combined decision function is then proposed, integrating the strengths of two independently trained FROCC models to enhance classification accuracy.

To further refine the model, vector optimization techniques are employed. By ranking and prioritizing the most effective classifier dimensions, the decision function is optimized to use only the top-performing vectors, resulting in improved predictive accuracy. The optimized FROCC model is rigorously evaluated against traditional baseline models, including Support Vector Machine (SVM) and CatBoost, demonstrating competitive or superior performance.

This thesis also explores various kernel functions to enhance the FROCC model further. The results underscore the potential of FROCC as a robust and efficient alternative to traditional binary classifiers, with significant improvements achievable through targeted optimizations.

The findings of this research contribute to the field of machine learning by providing a comprehensive understanding of FROCC's capabilities and limitations, along with practical strategies for its enhancement. The proposed methods and insights pave the way for future research and applications of FROCC in diverse binary classification tasks.

KEYWORDS: Machine learning, binary classification, FROCC

TABLE OF CONTENTS

CHAPTER 1

Introduction

In recent years, the use of machine learning in real-world problems is increasing. The main idea behind machine learning is to learn patterns from existing data and make predictions on unfamiliar data. Binary Classification is a classification task in machine learning where the goal is to classify the input data into two different classes or categories. We train the model to learn the patterns or features of the two classes and try to predict the input data belongs to which class. This problem is prevalent across various domains, including healthcare (e.g., disease diagnosis), finance (e.g., fraud detection), and technology (e.g., spam detection). The effectiveness of binary classification models directly impacts decision-making processes and the overall performance of systems relying on these models.

The main challenge of the binary classification is training. It requires a lot of labeled data for training. If we consider the model for detecting the image as a dog or cat then we need tens of thousands of images known to contain dog and cat for training. The need for large datasets leads to time-intensive training. Hence, the model must be efficient to train and predict in a reasonable time. If the two classes do not have significant differences in the training data it may lead to lower accuracy. So, training data needs to be selected carefully.

In recent years, a lot of binary classification methods have been proposed. Some commonly used methods are decision trees, random forests, Bayesian networks, Support Vector Machines, etc. Each classifier is best only a select domain depending on features, noise in the data. We will not discuss each of them here. Our study considers the Support Vector Machine (SVM) and CatBoost as baseline models to compare our proposed model. The details of the two are given in next chapter. However, these methods have certain limitations, particularly when dealing with complex and high-dimensional data. They often require extensive parameter tuning.

In recent years, kernel-based methods have gained attention for their ability to handle non-linear relationships in data. The FROCC (Fast Random projection-based One-Class Classification) [BVBB21] model is a relatively novel approach that leverages kernel functions to project data into

higher-dimensional spaces, facilitating better separation of classes. Originally designed for one-class classification problems, the potential of FROCC in binary classification has not been fully explored.

In this study we explore the possibility of using FROCC in binary classification. In chapter 2, we review some related literature required for our study. In chapter 3, we discuss the proposed models and some optimization methods to improve the FROCC models performance further. In chapter 4, we show our results and experiments. We discuss our findings and propose some future works in chapter 5

CHAPTER 2

Literature Review

2.1 Binary Classification

Binary classification is a problem where the outcomes are restricted to two possible outcomes. It is widely used in email spam detection, credit card fraud detection, and medical diagnosis such as detecting cancer cells [LK24]. In this study, we consider SVM and CatBoost as baseline models.

2.1.1 Support Vector Machines (SVM)

It is a supervised machine-learning algorithm. It aims to find the optimal hyperplane that separates the data points into two classes. The hyperplane is chosen to maximize the margin, which is the distance between the hyperplane and the nearest data points from each class. The dimension of the hyperplane depends on the number of features. It is very efficient in the case of high-dimensional cases where the number of dimensions exceeds the number of samples. It is also memory efficient as it only uses subsets of the training points in the decision function. Only support vectors are used to define the decision function. By using kernel functions, SVM can handle non-linear data.

Limitations of SVM are 1) Choice of kernel function affects the performance of SVM. 2) Training can be time-consuming for large datasets. Hence, SVM requires careful tuning of kernel parameters.

2.1.2 CatBoost

It is a gradient-boosting technique that handles categorical features efficiently [PGV$^+$19]. It constructs decision trees on each iteration. On each iteration, the decision trees are trained to correct the errors of the previous one. It does not require much preprocessing in the case of categorical features. It uses regularization to avoid overfitting. Its training time is fast due to its efficient implementation.

The main limitation of CatBoost is memory consumption. It requires significant memory resources in case of large datasets or high dimensional data.

2.2 One-Class Classification

Unlike the binary classification, One-class classification (OCC) involves identifying all instances of the target class. It is most used where normal data i.e. data with only one class is available only at training time. After training, it can identify the class from the data while treating everything else as an anomaly. Real-world uses include fraud detection [ZYW+18], monitoring the operational status of nuclear power plants as normal, etc.

2.2.1 FROCC

FROCC is a random projection-based method that only preserves the boundary [BVBB21]. FROCC uses random projection and interval trees to classify the point as normal or an outlier.

A small example taken from the FROCC paper is shown in Figure 2.1. In this example, point q is an outlier as it lies outside the boundary formed by e and f. The splitting of the interval $[a, d]$ into $[a, \mathbf{b}]$ and $[c, d]$ depends on the choice of epsilon.

The reason for choosing FROCC is its efficiency in terms of training and testing time as compared to the traditional model as discussed in the paper. FROCC only traverses the data once for training purposes.

In our study, we do not change the internal working of the FROCC model. The only change we need in the internal working of the FROCC model is its decision function. The decision function of the FROCC model works by computing the score as the fraction of the projection vectors agreeing out of all vectors. Hence, the score depends on the choice of vectors. So, if we can optimize these vectors we may be able to improve FROCC performance further. Changes done to the decision function and results obtained are shown in chapter 4.

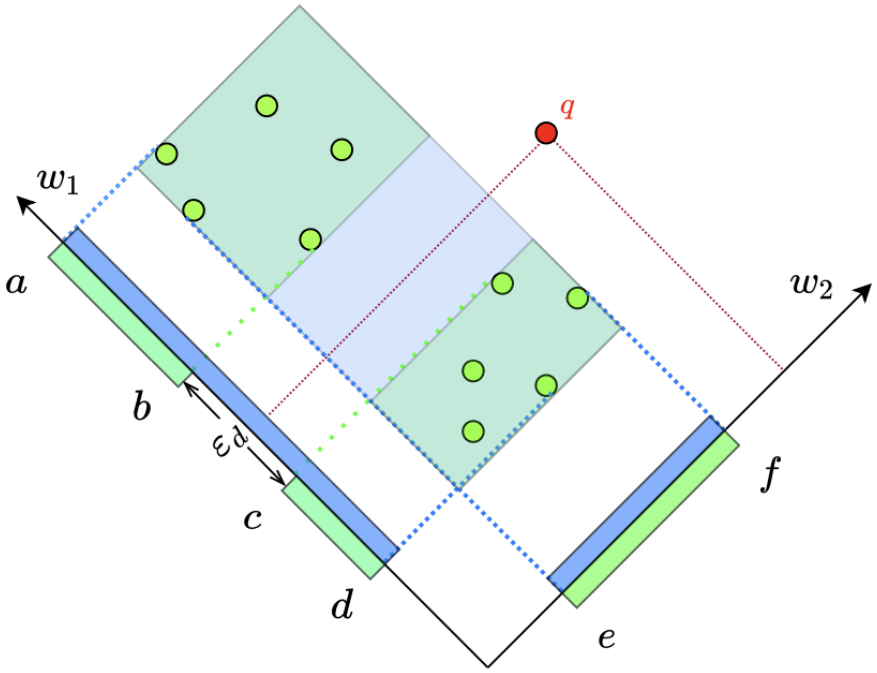

Figure 2.1: Working of FROCC taken from FROCC paper

2.3 Hyperparameter Optimization

In the machine learning pipeline, hyperparameter optimization is a crucial step for achieving better performance. These parameters are set before the training process. Effective hyperparameter optimization can enhance models' predictive power and robustness.

There are several techniques for hyperparameter optimization. The few techniques we used in our study are described briefly here. 1) Grid Search: It is an exhaustive search over a predefined set of hyperparameters. It ensures all combinations are considered, hence, it is time-consuming and resource-intensive. 2) Random Search: It selects parameters randomly from a defined distribution. It only considers a fixed number of combinations, hence, it may miss optimal combinations but it is fast and practical for high-dimensional data. 3) Bayesian Optimization: It takes an objective function and uses it to select the most promising parameters. It requires fewer evaluations to determine optimal parameters as it uses past evaluations to make informed decisions about the next set of parameters.

CHAPTER 3

Methodology

3.1 Introduction

This chapter details the methodologies employed in the study, focusing on the design, implementation, and optimization of FROCC models for binary classification. It also explains the process of combining the outputs of two FROCC models and the techniques used for evaluating and comparing their performance against baseline models like SVM and CatBoost.

3.2 Training

Since we are going for binary classification, we need binary datasets. We used OpenML datasets for this study. OpenML datasets come with meta-data and are structured and uniformly formatted to allow automated processing. The OpenML provides an API to directly download the dataset using the unique ID of the dataset. The study utilizes multiple binary classification datasets identified by their OpenMl dataset IDs: 23381, 1063, 40994, 6332, 1510, 1480, 29, 15, 1464, 37, 50, 31. These datasets are with varying numbers of features both numerical and categorical. Also, these are frequently used datasets in various works. The brief details of the dataset are provided in the Table 3.1

As we can see in Table 3.1, all datasets have different numbers of features. FROCC uses numerical features as input so we use the One-Hot Encoder to encode the non-numerical features to numerical values. In the next step, we split the data into training and test data. The test data is 20 percent of the complete data and the rest is training data. Next, the labels are converted to numerical values, 1 for the positive class and 0 for the negative class. Next, we split the training data into two sets i.e. positive and negative sets.

After getting the data ready, we start our process of exploring the use of FROCC as a binary classifier as well as optimizing the model. Our complete study can be broken into a few steps. Evaluation and discussion of the findings are done in the next chapter.

Dataset Id	Name	Features	Training Samples
15	breast-w	9	699
29	credit-approval	15	690
31	credit-g	20	1000
37	diabetes	8	768
50	tic-tac-toe	9	958
1063	kc2	21	522
1464	blood-transfusion-service-center	4	748
1480	ilpd	10	583
1510	wdbc	30	569
6332	cylinder-bands	37	540
23383	dresses-sales	12	500
40994	climate-model-simulation-crashes	18	540

Table 3.1: OpenML Datasets

3.2.1 Step 1

We train two separate FROCC models on the two training sets. We use grid search and random search to optimize the hyperparameters. We consider the best possible combination of hyperparameters.

3.2.2 Step 2

We train two separate FROCC models on the two training sets. The difference from Step 1 is that we took the scores of the two models and combined them to get a combined score. We use the below-given function for combination.

$$CombinedScore = (w1 * positiveScore + w2 * negativeScore)/(w1 + w2)$$

We use two different techniques to find the optimal weights.

1. **SLSQP:** Sequential Least-Squares Quadratic Programming is an optimization algorithm that is particularly effective for solving nonlinear constrained optimization problems. SLSQP can handle a variety of constraints, including linear and nonlinear equality and inequality constraints.
2. **Binary Cross-Entropy Loss:** Binary cross-entropy, also known as log loss, is a common loss function used in binary classification tasks. Its probabilistic nature, differentiability, and ability to penalize incorrect predictions make it a popular choice in machine learning.

3.2.3 Step 3

In this step, we try to optimize the decision function of the FROCC model. As described before, the decision function of the FROCC model evaluates scores as fractions of vectors agreeing to the total number of vectors. We propose to rank the vectors according to the frequency of correct predictions given by vectors. If we only consider the top-ranking vectors then the error in prediction can be minimized.

Ranking of vectors can be done easily. We change the original decision function of FROCC to store the number of correct predictions given by each vector. We find the agreement ratio for each vector.

$$agreementRatio = correctAgreement/totalAgreements]$$

We sort the vectors according to the agreement ratio. Then we define a new decision function that uses *k* top-ranking vectors or vectors above a certain threshold from these sorted vectors and calculate the scores.

CHAPTER 4

Experimental Results

In this chapter, we present the experimental results obtained from the evaluation done at each step described in the previous chapter. All the experiments are done on the same laptop Apple M3 16GB RAM.

4.1 Step 1

In this step, we compare the AUC value of the two models trained on the different classes but tested on the same test data. Figure 4.1 shows how the hyperparameters affect the performance of the FROCC model for a given dataset (ID: 50). The dimension parameter in the case of the FROCC model determines the number of random projection vectors. The other parameter we optimized is epsilon which determines the distance between intervals. In Figure 4.1, we considered ten possible values of epsilon for each value of dimension. Each point in the figure shows the sum of accuracy for each epsilon value for a given dimension. The accuracy of positive and negative models is very different. This indicates that data is not balanced for both classes. Hence, hyperparameter tuning is crucial before going further in our study. We can conclude that the role of hyperparameter tuning is a necessary step to be taken before going further for any work.

4.2 Step 2

In this step, we compare the combined models obtained from combination with individual models for each optimization technique. Figure 4.2 shows the comparison of the AUC value for the three models with the baseline models. In this case, the optimization is done using the binary cross-entropy loss technique. We can see that the combined model works better than the individual models only for a few datasets. Its results even degrade for a few datasets. The combination is worse than individual models which suggests that the combination function performed is not

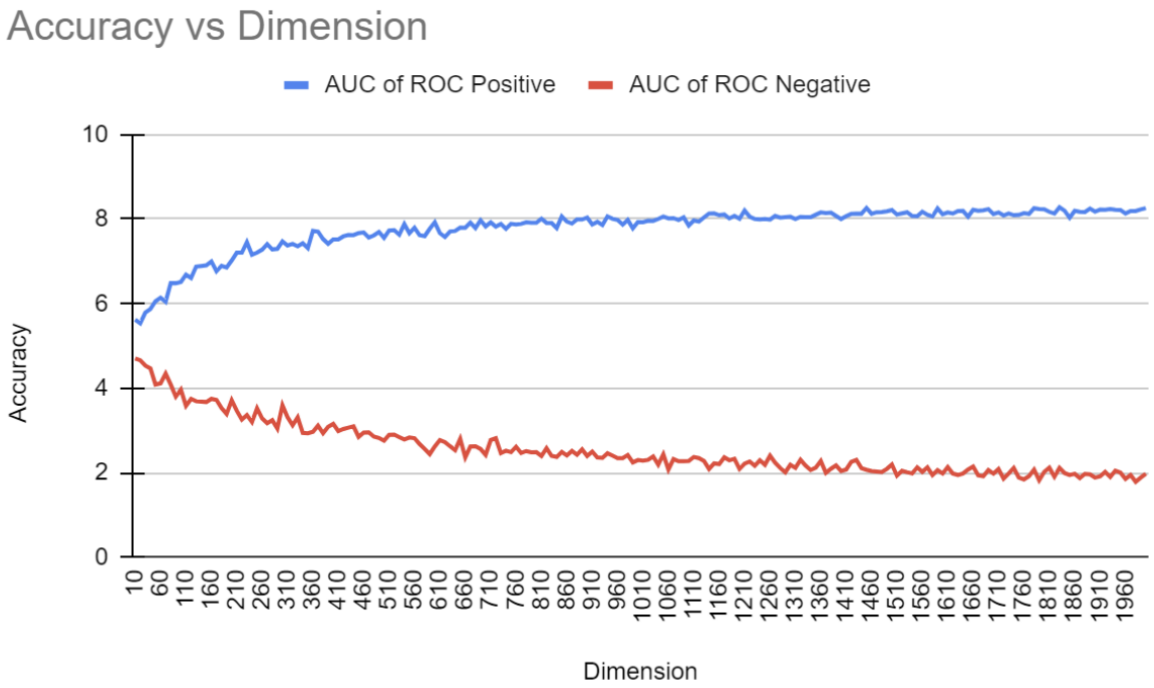

Figure 4.1: AUC value for different parameters

good. Even for the worst-case combination the combined model should at least be equal to the better-performing model among the two individual models.

Figure 4.3 shows the comparison with weight optimization done using the SQLSP technique. In this figure, we can see the combined performed a lot better than the individual models. There was even a case where it competes with baseline models. Except for one or two cases where it performs worse than the best individual model, its performance is excellent. From this, we can conclude that our proposed model has the potential to compete with baseline models. We need further fine-tuning to achieve the desired results.

4.3 Step 3

In this section, we discuss the findings of vector optimization done on the FROCC model. We show the results in this section in 2 parts. The AUC value of the negative model is calculated with the inverse score for better comparison such that both the AUC values represent the model performance for predicting positive class. Figure 4.4 shows the comparison of the individual models with and without vector optimization. From the results, we can see that the AUC value of the optimized positive model is most time better than its counterpart. The AUC value of the

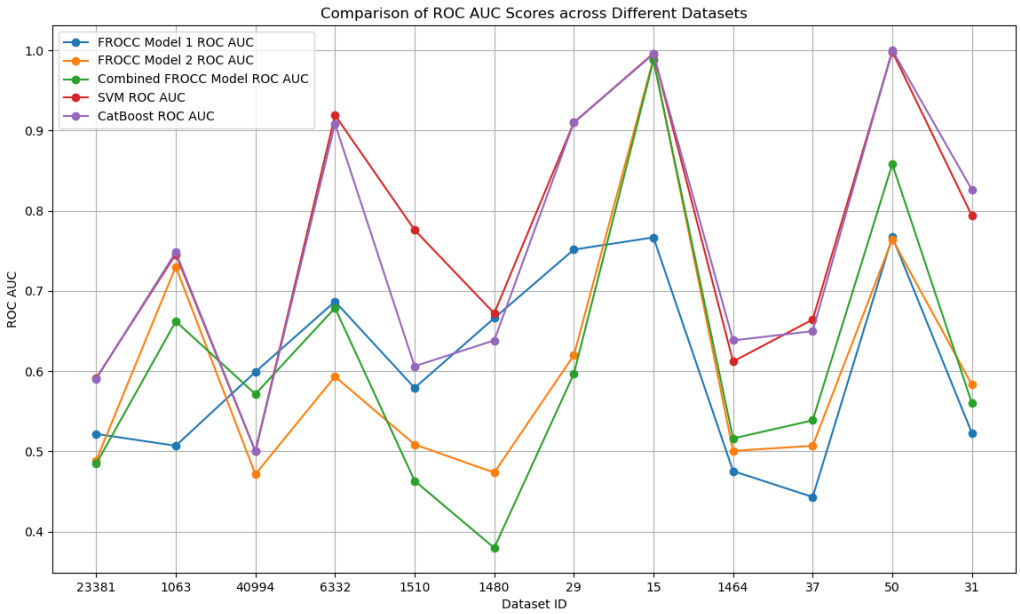

Figure 4.2: Weight optimization using binary cross-entropy

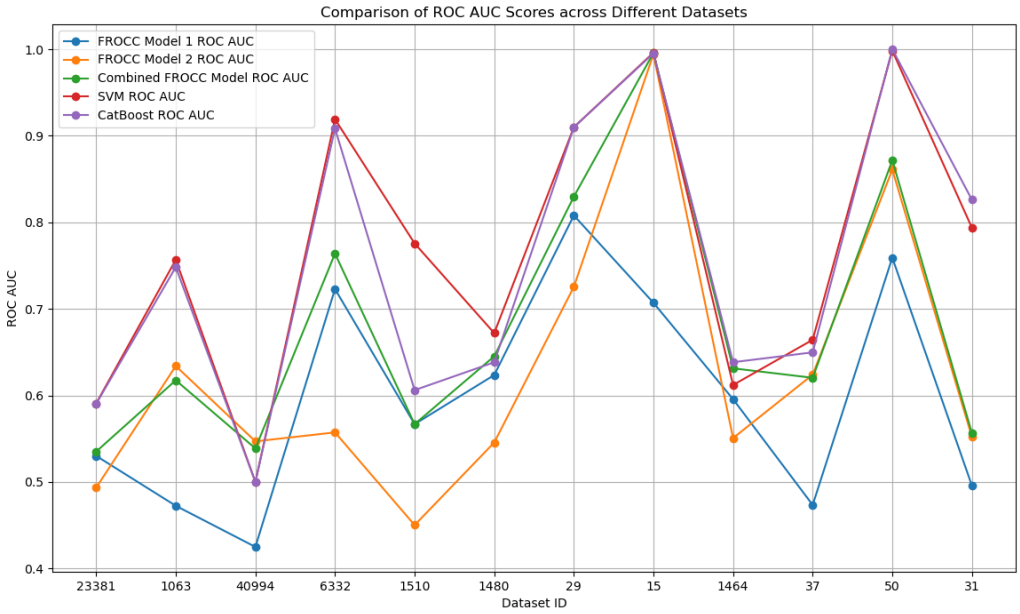

Figure 4.3: Weight optimization using SLSQP

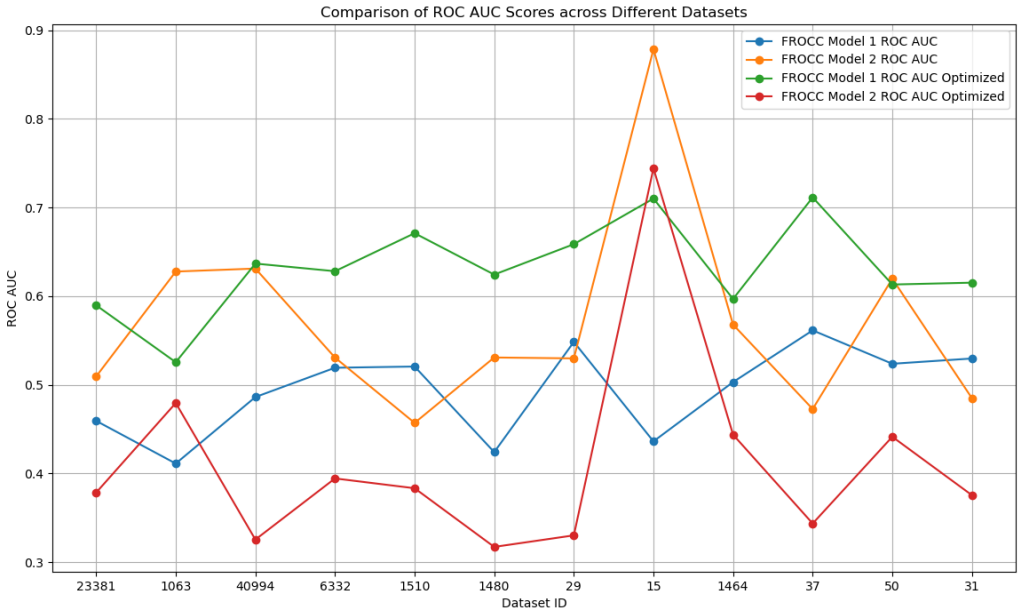

Figure 4.4: Comparison with individual models after Vector Optimization

negative model is most time lower than its counterpart which is in line with our expectations.

Figure 4.5 shows the comparison of combined models with the baseline models. In this case, we can see our model performs excellently. In a few cases, its results exceed that of the baseline models.

We can conclude from the results that vector optimization can indeed improve the performance of the FROCC model. The combination models can even compete with traditional models.

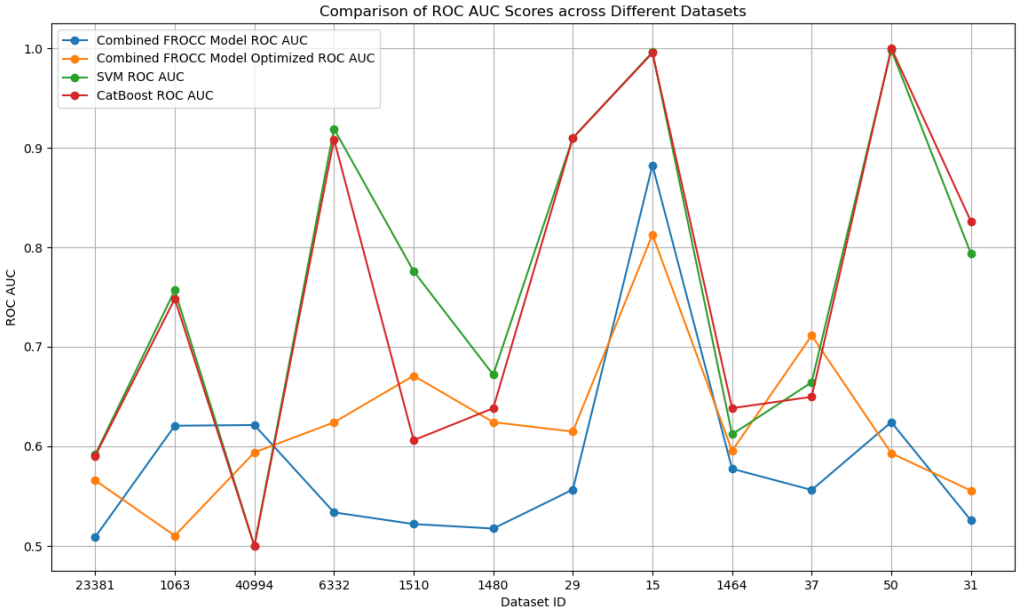

Figure 4.5: Comparison with baseline models after Vector Optimization

CHAPTER 5

Conclusion and Future Works

In this chapter, we discuss the implications of our findings and propose future work to improve the FROCC model's performance further. From the results shown in the last chapter, we can conclude that FROCC has a lot of potential for binary classification tasks. Just with correct weight optimization for our combined function shown in Figure 4.3, the combined model can show better results than the individual models. After our proposed vector optimization, results show that our combined model can compete with baseline models and can even exceed them in a few cases. This shows that the combination model and vector optimization of the FROCC model can be an excellent choice for binary classification. It even leverages the speed up for training and testing time provided by FROCC. Below we propose some future work for the usage of FROCC in binary classification.

5.1 Future Work

We propose to use advanced hyperparameter optimization techniques for an efficient tuning process so that we can save time during the training process. For the combination, we need to optimize the weights, so we need efficient methods such as parallel processing, etc to reduce time and for better handling of large datasets. Also, we can try different combination functions to optimize the combination of the FROCC models.

REFERENCES

[BVBB21] Arindam Bhattacharya, Sumanth Varambally, Amitabha Bagchi, and Srikanta Bedathur. Frocc: Fast random projection-based one-class classification, 2021.

[LK24] Chongmin Lee and Jihie Kim. Biomarker based cancer classification using an ensemble with pre-trained models, 2024.

[PGV+19] Liudmila Prokhorenkova, Gleb Gusev, Aleksandr Vorobev, Anna Veronika Dorogush, and Andrey Gulin. Catboost: unbiased boosting with categorical features, 2019.

[ZYW+18] Panpan Zheng, Shuhan Yuan, Xintao Wu, Jun Li, and Aidong Lu. One-class adversarial nets for fraud detection, 2018.