technische universität
dortmund

Master Thesis

Stacking Ensemble Methods for Anomaly Detection

Haritha Thiyaagu

March 20, 2024

Academic Advisors
Simon Klüttermann
Prof. Dr. Emmanuel Müller

Technical University of Dortmund
Department of Computer Science
Chair of Data Science and Data Engineering

Contents

1 Introduction **1**

1.1 Background and Context 1

1.2 Research Objectives 2

1.3 Structure of this Thesis 2

2 Existing Methods **3**

2.1 Anomaly Detection Algorithms 3

2.1.1 Auto-Encoder 3

2.1.2 Deep Support Vector Data Description 4

2.1.3 Isolation Forest 5

2.1.4 K-Nearest Neighbor 5

2.1.5 One-Class Support Vector Machine 6

2.2 Ensemble Methods 7

2.2.1 Mean Ensemble Method 7

2.2.2 Greedy Ensemble Method 8

2.2.3 Gaussian Mixture Model 9

2.3 Normalization Methods 10

2.3.1 Min-Max Scaling 10

2.3.2 Z-score Normalization 10

2.4 Metrics of Comparison and Evaluation 10

2.4.1 ROC Scores 11

2.4.2 Wilcoxon Test 11

2.4.3 Nemenyi Test and Critical Difference Plots 12

3 Experimental Setup and Contributions **12**

3.1 Overview of Data Sets 13

3.2 Contribution - Correlation Maximization Methods 13

3.3 Overview of the Models 14

4 Analysis and Results **18**

4.1 Individual Base Model Scores and Mean 18

4.2 Ensemble Scores of Existing Methods 20

4.3 Comparison of the Correlation Maximization Methods 22

4.4 Comparison against Base Models and Existing Ensembles 25

4.5 Statistical Tests 28

5 Conclusion and Outlook **31**

List of Tables **33**

List of Figures **34**

Bibliography **35**

Appendix **38**

5.1 Additional Figures 38

5.2 Additional Tables 42

1 Introduction

Anomaly Detection plays a pivotal role in Data Science across a wide range of applications like fraud detection and medical diagnosis, in domains including finance, cyber-security, healthcare, and manufacturing. The primary objective is to differentiate irregular patterns in data to identify potential anomalies that require further investigation. The detection of such anomalies helps in timely intervention and crucial decision-making in diverse applications.

Several conventional Anomaly Detection techniques are in use, each method focusing on anomalies of specific characteristics. However, there are also many challenges and obstacles in Anomaly Detection. There exists a necessity to detect previously unseen outliers without any prior knowledge about them. Moreover, in most cases, there could occur different kinds of anomalies in a data set and a single anomaly detection algorithm might not be effective in detecting all of them. This thesis addresses these challenges in Anomaly Detection and examines how the utilization of Stacking Ensemble Methods can increase its effectiveness. The performance of a set of anomaly detection algorithms and ensemble methods is evaluated over multiple data sets. Further on, the thesis also suggests and explores the performance of a novel ensemble method, based on the correlation between the outlier scores, and compares it with the existing methods. Hence, this thesis aims to enhance the efficacy of unsupervised anomaly detection by experimenting with robust and scalable stacking ensemble methods.

1.1 Background and Context

Anomalies, often referred to as outliers, represent data points that deviate considerably from the expected behavior. There are many challenges faced in anomaly detection processes in practical applications which are addressed in this thesis. With only limited labeled data being available for most of the applications, unsupervised Anomaly Detection [18] methods are required to be further explored. Traditionally, several Anomaly Detection techniques are in use for different applications, each specialized for detecting a specific type of anomaly, that works well on specific domains, with data satisfying specific assumptions. These methods may not all be viable to all applications, in general. Ensemble Methods [21] are effective in improving classification accuracy, especially in the context of Anomaly Detection, by combining the results of many detection techniques.

This problem establishes the context for the current research, emphasizing the importance of effective, application-specific Anomaly Detection techniques using Stacking Ensemble Methods, especially when labeled training data is scarce and the

available detection tools are manifold. The following sections delve into the research objectives and the structure of this thesis.

1.2 Research Objectives

This thesis aims to advance Anomaly Detection methodologies using Stacking Ensemble Methods. The initial phase involves a comparative analysis of some traditional unsupervised Outlier Detection algorithms such as K-Nearest Neighbor, One-Class Support Vector Machine, Isolation Forest, Auto-encoder, and Deep Support Vector Data Description, applied to a set of unlabeled data sets and the ensemble mean of their anomaly scores aggregated from these base models. Subsequently, the research moves on to the Ensemble Methods, implementing well-established stacking techniques such as Gaussian Mixture Models, K-Nearest Neighbors, Auto-encoders, and Greedy Method using the base models. The performances of these Ensembles are experimented with varied parameters and normalization methods for a comprehensive comparison among these Ensemble Methods, as well as against the individual base models.

Further on, an Ensemble Method called the Correlation Maximization method which is based on the maximization of weighted correlation between the generated and true labels is developed. This method is also assessed with different parameters and normalization methods to identify cases where each of the experiments performs well. These experiments are also compared with the standard Ensemble Methods used and their base models. The performances of all these models are evaluated using Visualization Methods and standard Statistical Tests like the Wilcoxon Test and Nemenyi Test. Thereby, this thesis aspires to contribute to the enhancement of Anomaly Detection, enabling more effective decision-making in varied data-driven contexts.

1.3 Structure of this Thesis

This thesis is structured to provide a comprehensive understanding of the existing methods of Anomaly Detection and Stacking Ensemble Methods and a discussion of the experiments conducted. Section 2 briefs on the basic concepts of Anomaly Detection and Ensemble Methods and the need to combine them. It also describes the standard detection models and the methods of comparison and evaluation used in this study. Section 3 elaborates on the experimental setup and the data sets used and further details the Correlation Maximization method and its different versions. Section 4 gives a comparative analysis of all the detection models over the collection of data sets using Visualization Methods and Statistical Tests. Finally, Section 5 summarizes the results of this research and the scope for further work.

2 Existing Methods

There exist several contemporary Anomaly Detection algorithms and Ensemble Methods, each tailored for distinct use cases and scenarios. This section navigates through the methods used in this thesis.

2.1 Anomaly Detection Algorithms

Anomaly Detection [18] involves identifying observations that deviate substantially from the remaining data. Anomalies can take various forms, such as unexpected spikes, outliers, or irregularities in the data. For a data set X of dimension d with n data points x_i, anomalies are the data points in $X_{anomaly}$, a subset of X, which are inconsistent with the other data points in X. Detecting such exceptional instances may hold critical insights and signal potential problems in many applications. Anomaly Detection algorithms can be parametric or non-parametric *(as suggested by Samariya)* and based on Statistical Models, Distance (Neighborhood, density, clustering), Classification, and Deep learning among others [37].

Anomaly detection methods can be broadly classified based on the availability of data as supervised, semi-supervised, and unsupervised methods [37]. Acquiring accurately labeled training data for supervised Anomaly Detection can be challenging in most scenarios. Hence, there is a necessity to find abnormal samples in such data sets without much knowledge about how the anomalies might look like, where unknown types of anomalies might appear, or where anomalies have not been recorded. This absence of labeled data leads to the development of unsupervised Anomaly Detection methods involving innovative techniques for identifying anomalies without prior knowledge. Auto-Encoder (2.1.1), Deep Support Vector Data Description (2.1.2), Isolation Forest (2.1.3), K-Nearest Neighbor (2.1.4), and One-Class Support Vector Machine (2.1.5) are used as base models in this thesis, as each of them have diverse characteristics and would help identify anomalies of different kinds.

2.1.1 Auto-Encoder

Auto-encoder (*AE*) *(as explained by Aggarwal)* [17] is a feed-forward, multi-layer neural network used for unsupervised learning. It encodes the input data into a compact representation and then reconstructs it with minimal loss. For Anomaly detection, *AE* learns a compressed, informative representation of the normal data to flag anomalies during reconstruction. They consist of several layers of encoders and decoders. Encoders compress the data into a latent space representation and Decoders reconstruct the data

from this representation. For learning, the difference between the input and the reconstructed output is minimized during training.

For an input data point x_i and its corresponding reconstructed output x'_i, the loss function used for training is typically the Mean Squared Error (MSE) calculated as

$$\text{MSE}\left(x_i, x'_i\right) = \frac{1}{d}\sum_{j=1}^{d}\left(x_{i,j} - x'_{i,j}\right)^2$$

where d is the number of features in the data [31]. This reconstruction error is the anomaly score s_i for x_i - a higher s_i suggests that the data point deviates from the norm.

Some of the important hyperparameters to be considered include batch size (the number of samples processed in each iteration), epoch (number of iterations), and activation functions (to introduce non-linearity to the model to learn complex patterns in the data, like *sigmoid*, $ReLU$, $tanh$). By training Auto-Encoders with the outputs of different algorithms as features and then aggregating their reconstruction errors or latent representations, Auto-encoders are also used as an ensemble method [19].

2.1.2 Deep Support Vector Data Description

The Deep Support Vector Data Description ($Deep-SVDD$) *(as explained by Zhang)* [34] algorithm begins by encoding the input data into a lower-dimensional latent space using a deep neural network architecture. This latent representation captures crucial features of the data while filtering out noise and irrelevant information. The network is trained by minimizing the reconstruction error between the input data and its encoded compact representation, thereby learning to capture the intrinsic characteristics of normal data instances.

Once the network is trained, $Deep-SVDD$ defines a hyper-sphere in the latent space using support vector machines obtained from the trained network. These support vectors represent the data points closest to the center of the hyper-sphere and are crucial for defining its boundary. The radius of the hyper-sphere is determined such that it encloses a predefined percentage of the normal data instances, typically representing the majority of the data set. To calculate the anomaly score s_i for each data point x_i, $Deep-SVDD$ measures the Euclidean distance of z_i, the encoded representation of x_i from c, the center of the hyper-sphere [24]:

$$s_i = ||z_i - c||.$$

This distance serves as a measure of the point's deviation from the learned representation of normal data. Data points lying outside the hyper-sphere, that is, those with anomaly scores exceeding a predefined threshold ϵ, are classified as anomalies.

2.1.3 Isolation Forest

Isolation Forest ($IFor$) *(as explained by Liu)* [7] constructs a collection (forest) of decision trees called isolation trees by recursively partitioning the data set. In each split, a random feature is selected, and a random value within the range of the selected feature is chosen as the splitting point. The process continues until the data points are isolated into individual trees or reach a pre-defined maximum depth. The anomaly score for a data point is calculated based on the average path length across all isolation trees. Intuitively, anomalies being rare instances, are expected to be isolated early in the tree (closer to the root) and have shorter average path lengths, making them stand out. If a data point is isolated quickly in many trees, it is likely to be an anomaly. The scoring function is defined as

$$s(x) = 2^{-\frac{E(h(x))}{c(n)}}$$

where $E(h(x))$ is the average path length for data point x over all trees, $h(x)$ is the path length for x in a single tree, and $c(n)$ is the normalization factor, which is the average path length for an unsuccessful search in a binary tree with n data points. The anomaly score is then normalized, and a pre-defined threshold is applied to classify instances as normal or anomalous. If $s(x, n)$ is higher than the threshold, the point is considered normal and otherwise, it is labeled as an anomaly.

Isolation Forest is efficient, scalable, and particularly effective for high-dimensional data. It works well for both global and local anomaly detection scenarios, where anomalies are rare and different from the majority of instances. However, its performance can be influenced by the choice of hyperparameters [39], such as the number of trees in the forest.

2.1.4 K-Nearest Neighbor

The data is represented as points in a multi-dimensional space, defined by the number of features used in the analysis, and this enables the evaluation of the distance between the data points. For any two data points x and y in a d-dimensional space, the Euclidean distance is calculated as

$$d(x, y) = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2 + ... + (x_d - y_d)^2}$$

where x_i and y_i are the values of the i-th feature of the data points, respectively.

K-Nearest Neighbor algorithm ($K-nn$) *(as discussed by Kataria)*[14] assumes that the amount that points differ from one another depends on the distances between them. That is, the distance between them characterizes how similar these data points are. $K - nn$ typically calculates the Euclidean distance from each data point x_i to all the other data points in the data set $x_{i=1,...,n}$ and assigns the K nearest neighbors $x_{i_1}, ..., x_{i_K}$ to each of

them ($K < n$), based on the shortest distances. The anomaly score s_i for each data point x_i is the distance to its K-th nearest neighbors:

$$s_i = d(x_i, x_{i_K}).$$

Intuitively, the points in dense regions will have many points near them, leading to a small s_i [3]. A data point is then labeled anomalous if its anomaly score exceeds ϵ, an evaluated threshold: $s_i > \epsilon$.

$K - nn$ is also used as an ensemble method by considering the scores generated by different anomaly detection algorithms as feature vectors and then using the $K - nn$ algorithm to determine the combined score based on the nearest neighbors in the feature space [25]. By doing this, the diversity among the different algorithms is leveraged to improve outlier detection performance.

2.1.5 One-Class Support Vector Machine

Unlike traditional Support Vector Machines that are designed for binary classification tasks, One-Class Support Vector Machine ($OC - SVM$) [4] is trained only on the normal class of data points, to create a boundary (hyper-sphere) that encapsulates the normal instances (inliers) in the feature space for Anomaly Detection. $OC - SVM$ maps the input data into a higher-dimensional space and finds a hyper-plane that best separates the normal instances from the origin or center of the feature space, such that as many normal instances are included as possible while maintaining a maximal margin from the origin. Instances lying on the side of the hyper-plane opposite to the origin are considered outliers or anomalies.

The optimal hyper-plane of the $OC - SVM$ in the transformed feature space is $w \cdot \phi(x) + b = 0$, where w is the weight vector, $\phi(x)$ is the mapping function and b is the bias term. For a new data point x, the decision function is

$$f(x) = w \cdot \phi(x) + b.$$

The signed distance of the data point to the hyper-plane is taken as its anomaly score. Given a new point x, the anomaly score $s(x)$ is computed as $s(x) = f(x) - threshold$, where the *threshold* is a predefined value. A point is considered normal if $s(x)$ is positive and an anomaly, if otherwise. Intuitively, points within the hyper-sphere are considered inliers, and the others, outliers *(as pointed out by Chalapathy)* [22]. $OC - SVM$ is also effective when the normal class dominates the data set, making it suitable for scenarios where anomalies are rare. However, its performance depends on appropriate parameter tuning, particularly kernel function and regularization parameters.

2.2 Ensemble Methods

Ensemble methods [2] construct a set of base models and combine them to provide improved results to advanced problems and are, therefore, very popularly used in Machine learning. This helps by averaging out the errors from bias and variance, aggregating knowledge from different models with different specialties.

There exist several types of parallel and sequential Ensemble methods like Voting, Bagging, Boosting, Randomization, and Stacking [21]. In voting ensemble methods [2], predictions are combined by taking a majority vote of the individual base models. Bagging [26] or Bootstrap Aggregating involves training multiple instances of the same base model on different subsets of the training data, typically sampled with replacement, with the final prediction being the average of the predictions from each model. Boosting [26] methods train a sequence of weak learners sequentially, where each subsequent learner focuses more on the misclassified instances of the previous learner by increasing their weight in each iteration. Randomization methods [2] introduce randomness during the training process, such as feature randomization or instance randomization, to diversify the models for enhanced generalization. Stacking, also known as Stacked Generalisation, predicts using a meta-learner, and has been shown to have better performance, especially when dealing with complex data sets and heterogeneous base models and so, is further explored in this thesis.

Stacking Ensemble Methods [26] train many base algorithms using the data set and then generate a new data set with the results of these base models, which is then used as the input to the combiner or meta-learner algorithm. This meta-learner learns the combined weighted predictions of the base models to generate the final prediction. For N base models for Anomaly Detection, each of the algorithms $Y_1, ..., Y_n$ gives an Anomaly Score s_i for a data set X. The final anomaly score is $Y_{\text{ensemble}} = f(s_1, ..., s_n)$, where f is the ensemble method over the concatenated scores. Stacking, with its emphasis on model diversity and flexibility, and meta-learner optimization, improves accuracy and performance, especially in complex and noisy problems. Mean Ensemble (2.2.1), Greedy Ensemble (2.2.2), K-Nearest Neighbor Ensemble (2.1.4), Auto-encoder Ensemble (2.1.1), and Gaussian Mixture Model Ensemble (2.2.3) are the existing Ensemble methods explored in this thesis, while comparing them with the proposed Correlation Maximization Ensemble (3.2), which is a stacking ensemble based on the analysis of correlation between the outlier scores.

2.2.1 Mean Ensemble Method

The Mean Ensemble ($Mean\_base$) model score is averaged over the different base models of the ensemble. If the individual base models or components of the ensemble are poorly

derived models, the irrelevant scores from many diverse components might reduce the overall anomaly score. However, this simple ensemble method, which takes the average of predictions from each of the base models, is used for comparison.

2.2.2 Greedy Ensemble Method

The Greedy Ensemble Method [11] (*Greedy*) estimates the performance of individual algorithms or base models based on their correlation with a constructed label vector. Accuracy and diversity are the two conditions to ensure that an ensemble method outperforms its base models. This method optimizes diversity among the models with uncorrelated errors and is similar to boosting but in an unsupervised scenario. Due to the absence of true labels, the method constructs an approximate label vector from the base models themselves [36].

A label vector is created by aggregating the top k most abnormal samples identified by each of the base model algorithms to be true anomalies, assigning them a value close to 1, and 0 otherwise. The algorithm then proceeds by selecting subsets of models that maximize a weighted correlation with the constructed label vector, considering correlation as a measure to estimate the performance of each model relative to the constructed vector. The Weighted Pearson Correlation between the base model anomaly scores vector X and the constructed label vector Y is calculated as

$$\rho_\omega(X,Y) = \frac{\mathrm{Cov}_\omega(X,Y)}{\sigma_\omega(X)\sigma_\omega(Y)}$$

where

$$\mathrm{Cov}_\omega(X,Y) = \frac{1}{\Omega}\sum_i \omega_i \left(X_i - E_\omega(X)\right)\left(Y_i - E_\omega(Y)\right)$$

is the Weighted Covariance between vectors X and Y and

$$\sigma_\omega(X) = \sqrt{\frac{1}{\Omega}\sum_i \omega_i \left(X_i - E_\omega(X)\right)^2} \quad \text{and} \quad \sigma_\omega(Y) = \sqrt{\frac{1}{\Omega}\sum_i \omega_i \left(Y_i - E_\omega(Y)\right)^2}$$

are the Weighted Standard Deviations of the vectors X and Y, respectively, for weight ω_i, the weighted mean of X and Y, $E_\omega(X)$ and $E_\omega(Y)$, respectively, and the sum of the weights assigned to the objects, $\Omega = \sum_i \omega_i$. The ensemble is initialized with the model having the highest weighted Pearson correlation with the constructed target vector. Subsequent models are selected or discarded based on their ability to improve the correlation with the target vector while also maximizing diversity within the ensemble. At each iteration, the method evaluates whether including a new model improves the ensemble's correlation with the target vector, weights ω_i corresponding to each model i taking values of either 0 (when the model is not included) or 1 (when the model is

included). If the inclusion results in a higher correlation, the model is added to the ensemble and if not, the model is discarded. That is, models that maximize a weighted correlation to the constructed vector are chosen in a greedy manner. This process continues until all available models have been considered [11]. The anomaly score,

$$s\_ensemble = \sum_{i=1}^{m} \mathbb{1}_i \cdot s_i$$

is the weighted sum of the m base model scores, where the weight of a model i is denoted by the indicator function, $\mathbb{1}_i$.

2.2.3 Gaussian Mixture Model

The Gaussian Mixture Model (GMM) [13] is a probabilistic model that assumes that the observed data is generated from a mixture of several Gaussian distributions to learn complex data structures. The model is then trained using Expectation-Maximization which iteratively maximizes a lower bound of the likelihood. In the context of Anomaly Detection, GMM models the underlying probability distribution of the data set X with n samples and d dimensions as a linear combination of g Gaussian components:

$$P(X \mid \theta) = \sum_{i=1}^{g} \pi_i \, \mathcal{N}\left(X \mid \mu_i, \Sigma_i\right)$$

where θ represents the model parameters or mixture coefficients, π_i is the weight of the i-th component, indicating the proportion of the data set it covers, and $\mathcal{N}\left(X \mid \mu_i, \Sigma_i\right)$ is the Gaussian distribution of i with mean μ and covariance matrix Σ. This likelihood function is maximized with the Expectation-Maximization algorithm as

$$\ln P(X \mid \theta) = \sum_{j=1}^{n} \ln \left(\sum_{i=1}^{g} \pi_i \mathcal{N}\left(X_j \mid \mu_i, \Sigma_i\right) \right).$$

The anomaly score, s_i for x_i is calculated as the negative logarithm of its likelihood under the GMM. The higher the anomaly score of a point, the lower the probability that the point is generated by the mixture model, and the more likely the data point is an anomaly. Instances with low likelihoods under this model are considered potential anomalies. GMM is used as an ensemble method by training it on a feature space of scores from different anomaly detection algorithms, to improve the robustness and generalization of anomaly detection.

2.3 Normalization Methods

Normalization [32] is a crucial pre-processing step for Anomaly Detection, transforming data to a standard scale or distribution. This ensures that each feature contributes proportionally to the model's learning process, preventing certain features from dominating due to their larger scales and vice versa, and thus, enhancing the robustness and stability of the models. Some of the common Normalization methods to make features comparable and facilitate the convergence of optimization algorithms are Min-Max Scaling, Z-score Normalization, and Robust Scaling. The presence of outliers in the data is to be considered when choosing an appropriate normalization method.

2.3.1 Min-Max Scaling

Min-Max Scaling *(according to Henderi)* [32] or 0-1 normalization transforms each feature to a specific range, usually $[0, 1]$. The normalised value for a data point x_i in feature j is evaluated as

$$x_{ij}^{\text{min-max-norm}} = \frac{x_{ij} - \min_j}{\max_j - \min_j}.$$

This method is preferred when the data distribution is not necessarily or not known to be normal, but is sensitive to outliers that might affect the scaling.

2.3.2 Z-score Normalization

Z-score Normalization *(according to Henderi)* [32] or Standardization is more robust to outliers when compared to Min-Max Normalization as it centers the data around the mean, reducing the impact of the extreme values. The normalized value for a data point x_ij in feature j is calculated as the Z-score:

$$x_{ij}^{\text{z-score-norm}} = \frac{x_{ij} - \mu_j}{\sigma_j}$$

where μ_j is the mean of the feature j and σ_j, its standard deviation.

2.4 Metrics of Comparison and Evaluation

The metrics or anomaly scores used to evaluate the trained models and the statistical tests and the visualization methods used to compare and analyze them are discussed in this section. Besides the following metrics, other Graphical Methods [23] are also used for the comparison of models, which include Bar Plots to illustrate the maximum scores and the frequency of the best performance, Heat Maps to visualize Nemenyi Test (2.4.3) results, Candlestick charts [12] to compare the mean performance of the algorithms,

Kernel Density Estimate Plots to understand the distribution of the scores for each of the algorithms, and Scatter Plots and Stacked Plots to plot the difference in scores for pairs of algorithms.

2.4.1 ROC Scores

Receiver Operation Characteristic (ROC) score [9] is an evaluation metric for the performance and effectiveness of Anomaly Detection models. ROC is the probability of random anomaly samples being assigned a higher anomaly score than a random normal one. ROC analysis gives a trade-off between the true positive rate TPR (sensitivity) and the false positive rate FPR (1− specificity) across various decision thresholds. TPR is the ratio of true positives TP to the sum of true positives and false negatives FN and FPR is the ratio of false positives FP to the sum of false positives and true negatives TN:

$$TPR = \frac{TP}{TP + FN} \quad \text{and} \quad FPR = \frac{FP}{FP + TN}$$

The ROC curve is then constructed by plotting TPR against FPR at all possible threshold values. Adjusting the threshold values allows for tuning the balance between sensitivity and specificity. The area under the ROC curve ($AUC - ROC$) is a scalar value (between 0 and 1) which quantifies the ability of the model to differentiate between normal and anomalous data points. The $AUC - ROC$ score ranges from 0 to 1 - the higher the $AUC - ROC$, the better the model performance *(as acknowledged by Zhao)* [21].

2.4.2 Wilcoxon Test

The Wilcoxon signed-rank test [1] is a non-parametric statistical test used to compare paired samples or repeated measurements on a single sample. It is used to assess whether there is a significant difference between the ROC scores of different Anomaly Detection models by comparing the ranks of the ROC scores obtained from every pair of models.

The Null Hypothesis is taken that there is no significant difference between the paired scores, while the Alternative Hypothesis suggests a significant difference. For each pair of ROC scores, their differences and the ranks of these differences are calculated in terms of their absolute values. These signed ranks are used to compute the test statistic, which follows a specific distribution under the Null Hypothesis, allowing for the determination of the $p - value$. The test statistic is the sum of the ranks for positive differences and the $p - value$ gives the probability of observing a score when the Null Hypothesis is true. If the $p - value$ is lesser than a chosen Significance Level (typically, $\alpha = 0.05$), the Null Hypothesis is rejected, indicating a significant difference in the performance of the models

(suggesting that one model consistently outperforms the other) [10]. The Wilcoxon test assumes that the differences between pairs of samples are symmetrically distributed. This test is applicable when the data may not follow a normal distribution, making it suitable for comparing the performance of Anomaly Detection methods without assuming specific distribution characteristics, but can be used to compare only two methods at a time.

2.4.3 Nemenyi Test and Critical Difference Plots

The Nemenyi test [20] is a post-hoc statistical test, often used in conjunction with the Friedman test, which is employed for comparing multiple methods or models across different data sets. After conducting the Friedman test and determining that there are significant differences among the methods, the Nemenyi test is applied to identify specific pairs of methods that differ significantly, based on Critical Difference CD. The CD is the minimum average rank difference required for the two methods to be considered significantly different. If the average rank difference between the two methods exceeds the critical difference, they are deemed to have a significant performance difference.

The critical difference is calculated using a critical value from the Studentized range distribution or q distribution (a probability distribution used in statistical hypothesis testing) and a factor based on the number of methods and data sets as

$$CD = q_\alpha \cdot \sqrt{\frac{p\,(p+1)}{6N}}$$

where q_α is the critical value from the Studentized range distribution for a chosen significance level (typically, $\alpha = 0.05$), p is the number of methods compared and N is the number of data sets.

A Critical Difference plot [5] is generated using these CD values to visually represent the significant differences between methods. Methods connected by a line in the plot are not significantly different, while those without a connecting line are significantly different. This illustrates the differences in performance among multiple anomaly detection methods and aids in identifying the most effective method for a given context. The vertical bars or lines associated with each model show their relative ranking in terms of performance.

3 Experimental Setup and Contributions

The experimental setup including the processing of the data sets, base model anomaly algorithms used, ensemble methods the scores are combined with, the other variations of

these methods that are tried out, and the tests used to compare and evaluate them are discussed in this section.

All the experiments in this thesis are coded in the object-oriented, high-level programming language *Python*3.8.8 [38]. The packages used include *os* for interacting with the operating system, *time* for time-related functions like tracking execution time, *NumPy* [28] for numerical operations and array manipulations, *Matplotlib* [6] for creating visualizations and plots to analyze data, *pandas* [29] for data manipulation and analysis, *Seaborn* [33] for statistical data visualization, *PyOD* [27] for implementing various anomaly detection algorithms and utilities for model evaluation and ensemble learning, *SciPy* [30] for scientific computing functions and tools like statistical tests and optimization algorithms, $scikit-learn$ [8] for machine learning algorithms and tools for data mining and data analysis tasks, and *TensorFlow* [15] for deep learning and neural network framework.

3.1 Overview of Data Sets

The data sets used in this thesis are sourced from the Yano (version 0.92) [35] repository, a comprehensive Python module providing easy access to data sets specifically curated for unsupervised anomaly detection research. It is a collection of 181 anomaly detection data sets over which 21 distinct outlier detection algorithms and ensembles are applied.

Test and train samples and the true labels from each of the data sets are extracted for further experiments and used with all the models compared. Some models are incompatible with some data sets, producing faulty or erroneous results - such cases are ignored for respective combinations, and ensembles are built from the remaining sub-models [36]. For explanation within this report, fetal heart rate signals data set - *cardio* [16], (originally consisting of 1831 data points across 21 features with 176 (9.6%) outliers) is explored in detail, although, all 181 data sets are used for experimentation. From the *cardio* data set, 296 train samples and 352 test samples are extracted for experimentation in this thesis.

3.2 Contribution - Correlation Maximization Methods

Correlation Maximization (CM) is a novel method tested in this thesis, as an extension of the Greedy Method (2.2.2). This method also constructs an approximate label vector from the base models and then, leverages the correlation between them to estimate the performance of the algorithms. Two approaches are used to construct the label vector, considering the top k ranking samples (with the k highest anomaly scores (2.4.1)) for each of the base models:

- *Unique* - Samples having the top k highest anomaly scores in any one of the base models are labeled anomalous, giving them a value of 1, and the others, 0 (normal).
- *Count* - The number of base models (count) in which each sample gets the top k anomaly scores is taken as the label. As a result, the constructed label values range between 0 and n, where n is the number of base models considered for the ensemble.

After constructing the label vector, instead of selecting a subset of base models, all models are included with varying weights, taking continuous values between 0 and 1. This way, every model contributes to the ensemble in varying ranges, for a more robust anomaly score. The ensemble is initialized with equal weights $\frac{1}{n}$ assigned to each of the n base models. The weights are then optimized such that the correlation value is maximized, using two alternate approaches, instead of the greedy approach:

- *Computational Optimization* - Correlation is maximized computationally using the *minimize* function from the *scipy.optimize* [30] package in *Python* [38]. Here, the negative correlation is minimized to maximize the positive correlation.
- *Approximation* - This is a mathematical optimization technique to reduce computation time, arbitrarily assuming that the average of the prediction scores is zero and approximating (normalizing) the weights to be bounded within a certain limit. By doing this, the weight of a base model is proportionally taken to be equal to the Covariance between its predicted scores and the constructed labels. In contrast to *Computational Optimization*, this method is only approximative, although it reduces computational time by more than 60%.

Both these approaches aim to get the global maximum correlation as the greedy approach might lead to local minima as well. The anomaly score,

$$s\_ensemble = \sum_{i=1}^{m} \omega_i \cdot s_i$$

is the weighted sum of the m base model scores, where $\omega_i = [0, 1]$ is the weight of a model. Experiments with a combination of these approaches along with tuning for varying hyperparameters like k and different normalization methods are conducted in this thesis.

3.3 Overview of the Models

Each of the 181 data sets is initially trained with 5 independent anomaly detection base models (2.1) using the *PyOD* library [27] functions with the default parameters:

- **1**: Auto-encoder - *AE* (2.1.1)

- **2**: Deep Support Vector Data Description - $Deep - SVDD$ (2.1.2)
- **3**: Isolation Forest - $IFor$ (2.1.3)
- **4**: K-Nearest Neighbor - $K - nn$ (2.1.4)
- **5**: One-Class Support Vector Machine - $OC - SVM$ (2.1.5)

The results of the individual anomaly detection models are evaluated and compared and then, 26 ensemble models (2.2) (as listed below) are constructed, aggregating the results of these 5 base models, along with some corresponding hyperparameter tuning:

- **6**: Simple Mean of Scores, $Mean(s) = \frac{\sum_{i=1}^{n} s_i}{n}$ - $Mean\_base$
- **7**: Greedy Ensemble with the number of anomalies for the constructed label vector, $k = 5$, using $0 - 1 Normalization$ (2.3.1) over the scores - $Greedy$ (2.2.2)
- **8**: K-Nearest Neighbor Ensemble with the number of neighbors considered, $k = 5$ or $K = n - 1$ when $n < k$, where n is the sample size - $K - nn\_ensemble$ (2.1.4)
- **9**: Auto-Encoder Ensemble defined with 3 layers of encoders, 3 layers of decoders over a batch size of 256 for 100 epochs, using both *sigmoid* and *ReLU* activation functions - $AE\_ensemble$ (2.1.1)
- Correlation Maximization Ensemble (3.2)
  - using $0 - 1\, Normalization$ (2.3.1)
    - with Computational Optimization for weights and
      - the label vector constructed using the 'unique' approach for varying numbers of anomalies k
        - **10**: $k = 5$ - $CM\_01\_CU\_5$
        - **11**: $k = 20$ - $CM\_01\_CU\_20$
        - **12**: $k = 55$ - $CM\_01\_CU\_55$
      - the label vector constructed using the 'count' approach for varying numbers of anomalies k
        - **13**: $k = 5$ - $CM\_01\_CC\_5$
        - **14**: $k = 20$ - $CM\_01\_CC\_20$
        - **15**: $k = 55$ - $CM\_01\_CC\_55$
    - with Approximation of weights and
      - the label vector constructed using the 'unique' approach for varying numbers of anomalies k

- **16**: k = 5 - $CM\_01\_AU\_5$
- **17**: k = 20 - $CM\_01\_AU\_20$
- **18**: k = 35 - $CM\_01\_AU\_35$
- **19**: k = 55 - $CM\_01\_AU\_55$

- the label vector constructed using the 'count' approach for varying numbers of anomalies k
  - **20**: k = 5 - $CM\_01\_AC\_5$
  - **21**: k = 20 - $CM\_01\_AC\_20$
  - **22**: k = 35 - $CM\_01\_AC\_35$
  - **23**: k = 55 - $CM\_01\_AC\_55$

- using $Z - score\,Normalization$ (2.3.2)
  - with Computational Optimization for weights and
    - **24**: the label vector constructed using the 'unique' approach for numbers of anomalies k = 55 - $CM\_Z\_CU\_55$
    - **25**: the label vector constructed using the 'count' approach for numbers of anomalies k = 55 - $CM\_Z\_CC\_55$
  - with Approximation of weights and
    - **26**: the label vector constructed using the 'unique' approach for numbers of anomalies k = 5 - $CM\_Z\_AU\_5$
    - **27**: the label vector constructed using the 'count' approach for numbers of anomalies k = 5 - $CM\_Z\_AC\_5$

- Gaussian Mixture Model Ensemble (2.2.3)
  - using $0-1\,Normalization$ of scores (2.3.1) with varying number of components, g
    - **28**: g = 2 - GMM_01_2
    - **29**: g = 5 - GMM_01_5
    - **30**: g = 50 - GMM_01_50
  - **31**: using $Z - score\,Normalization$ of scores (2.3.2) with number of components, g = 2 - GMM_Z_2

For the proposed Correlation Maximization ensemble, the Computational Optimization method is pitted against the Approximation method with varying parameters like different values of contamination, $k = \{5, 20, 35, 55\}$ and different label construction methods, 'unique' and 'count'. Having normalized the scores to $0 - 1$ initially, the entire set of experiments is then repeated using Z-score normalization. The

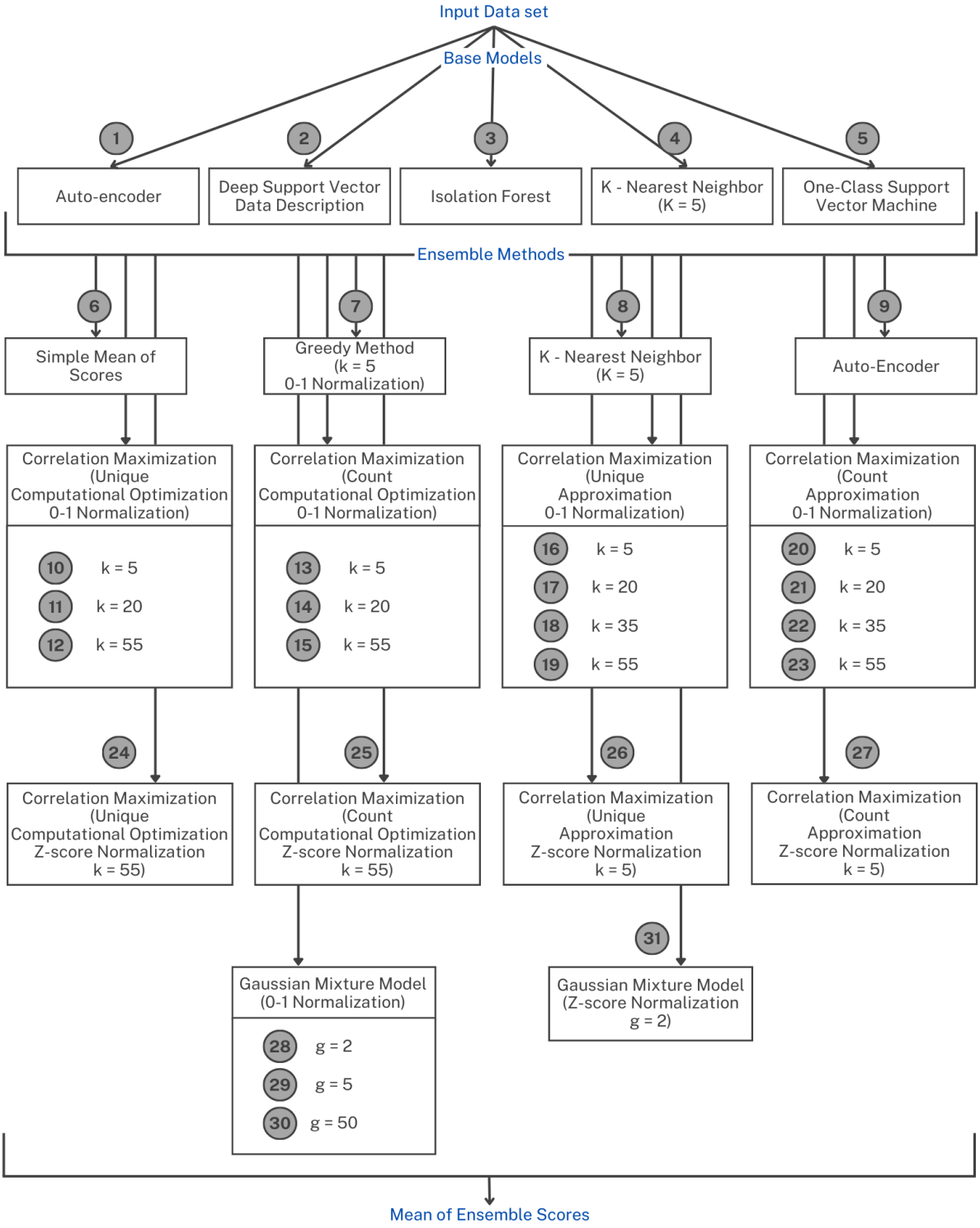


Figure 1: Architecture of Models

effects of hyperparameter tuning - varying k values, normalization methods, approximated label construction methods, and optimization approaches - are studied. The Gaussian Mixture Model Ensemble is initially analyzed with $0-1$ normalized scores for different number of components, $g = \{2, 5, 50\}$ and is then repeated with Z-score normalization for $g = 2$

Results from each of the ensemble models are analyzed and their corresponding *ROC* scores (2.4.1) are obtained. Finally, the mean scores of all of the models, including both the base models and the ensembles are taken as a final measure - $Mean\_all$ - to arrive at a decision. The architecture of the models analyzed in this thesis is illustrated in figure 1. The notations mentioned for each of the ensembles in this section are used throughout the report. Each model listed is also enumerated for reference against the architecture (1).

Having constructed all the 31 models for each of the data sets, their respective *ROC* scores are derived and the various aspects of the performance of these models are explored (2.4). Bar charts are generated to visualize the number of data sets in which each of the ensemble methods with respect to which set of parameters has performed the best and also, the mean scores of every model over all the data sets. Scores from pairs of models are compared using the Wilcoxon Test (2.4.2) and Stacked Bar Plots. The scores from a group of models are compared using the Nemenyi Test (2.4.3) for which, the results are assessed with Heat Maps and Critical Difference Plots. Candlestick diagrams are generated to compare the performance of the models and Kernel Density Estimates are plotted to understand the distribution of the scores. The optimal choice of the number of anomalies k for label generation in the proposed Correlation Maximization method is investigated. After analysis, the results of the comparative performance of each of the models are summarized.

4 Analysis and Results

This section describes all the significant findings from the experiments in section 3 - the base models, the existing ensemble methods, the proposed ensemble approaches, and the corresponding hyperparameter tuning. The anomaly scores of all the models for the data set *cardio* are initially discussed and are then rolled out for all the data sets. In the end, the results of all the statistical tests and the methods of comparison are analyzed, and the significance of the proposed Correlation Maximization Ensemble is highlighted. All the scores are rounded to 5 decimal places.

4.1 Individual Base Model Scores and Mean

The *ROC* scores for all the 181 data sets (3.1) are obtained for the 5 chosen base model algorithms (2.1) with the default parameters in the *PyOD* package. No one model is found to perform uniformly the best in all the data sets. About 37% of the data sets get the highest *ROC* score on using $K - nn$ while the other models have their unique share of data sets on which they have performed the best, as shown in the pie chart figure

2. $K - nn$ being a non-parametric method, depending only on the value of K, performs effectively with smaller-sized high-dimensional data sets while the rest may suffer due to the curse of dimensionality. On the other hand, when dealing with noisy data with irrelevant features that might require complex decision boundaries, more complex methods like $Deep - SVDD$, AE, $IFor$, and $OC - SVM$ would perform better. Meanwhile, on data with spatial or temporal dependencies and multiple nodes, Auto-encoders might perform better. However, they require a large amount of data to train effectively and are sensitive to model architecture and hyperparameters.

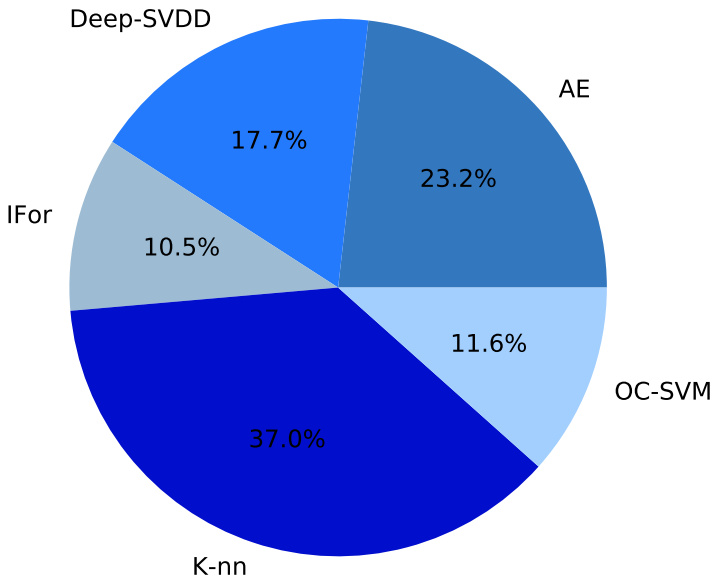


Figure 2: Percentage of data sets in which each of the base models has performed the best

As each of the algorithms functions differently, with different understandings of the data, it is crucial to study methods of combining the learning from such individual base models, for a more robust detection of anomalies.

The simplest ensemble used is the $Mean\_base$ method, which averages the understanding from each of the methods. To understand this further, a closer look is taken at data set *cardio* - the ROC scores of all the models for the data set *cardio* are illustrated in figure 11 in the appendix and the exact values are listed here, in table 1. The $Mean\_base$ score, 0.94728, is higher than most of the individual base models - almost equal to the highest score 0.94751 by AE and is relatively higher than that of the lowest, 0.63152, by $Deep - SVDD$. This warrants the need for better ways to combine

the predictions of the base models with more complex stacking ensemble methods, possibly with other combinations of parameters.

Table 1: *ROC* scores from all the models for data set *cardio*

	Model	*ROC* **score**
Base models	AE	0.94751
	$Deep-SVDD$	0.63152
	$IFor$	0.93705
	$K-nn$	0.93240
	$OC-SVM$	0.94654
Ensembles	$Mean\_base$	0.94728
	$Greedy$	0.94635
	$K-nn\_ensemble$	0.92956
	$AE\_ensemble$	0.94315
	$CM\_01\_CU\_5$	0.94350
	$CM\_01\_CU\_20$	0.94464
	$CM\_01\_CU\_55$	0.94231
	$CM\_01\_CC\_5$	0.94047
	$CM\_01\_CC\_20$	0.94793
	$CM\_01\_CC\_55$	0.94373
	$CM\_01\_AU\_5$	0.94728
	$CM\_01\_AU\_20$	0.94764
	$CM\_01\_AU\_35$	0.94757
	$CM\_01\_AU\_55$	0.94738
	$CM\_01\_AC\_5$	0.94744
	$CM\_01\_AC\_20$	0.94764
	$CM\_01\_AC\_35$	0.94754
	$CM\_01\_AC\_55$	0.94754
	$CM\_Z\_CU\_55$	0.94231
	$CM\_Z\_CC\_55$	0.94373
	$CM\_Z\_AU\_5$	0.94606
	$CM\_Z\_AC\_5$	0.94412
	GMM_01_2	0.09750
	GMM_01_5	0.11709
	GMM_01_50	0.33787
	GMM_Z_2	0.09759
	$Mean\_all$	0.82443

4.2 Ensemble Scores of Existing Methods

All the data sets are further trained and tested with the chosen existing stacking ensemble methods (from section 2.2. GMM is initially tried with an increasing number of components - 2, 5, and 50 - and $Min-Max\,Scaling$ but also using $Z-score\,Normalization$ for 2 components. The number of data sets in which each

model gives the highest *ROC* score is displayed in the bar chart figure 3. Among the 181 data sets, 68 get their highest score with *Greedy* method, closely followed by $K - nn\_ensemble$ for 63 data sets. GMM does not show promising results for any of their variation in the number of components and the normalization techniques.

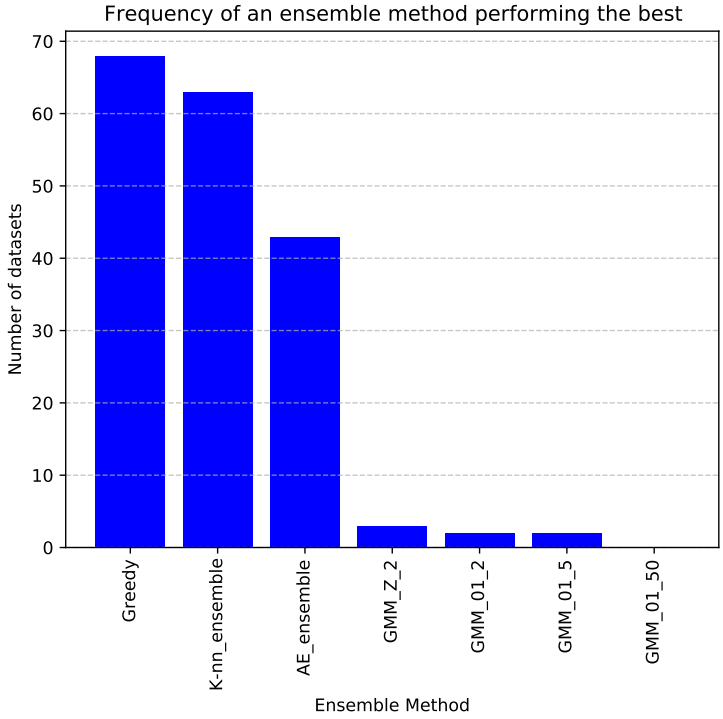


Figure 3: Number of data sets for which each of the existing ensemble methods has the highest score

Figure 4 is the candlestick diagram showing the decreasing mean of *ROC* scores across all the 31 models experimented with in this thesis, to understand the variability in the performance of the models. The green bars contain values from the first and third quartiles ($25th$ and $75th$ percentile) of the *ROC* scores. These bars indicate their average range of performance across data sets and the thin tails at the ends show the extent of the outliers with respect to the mean scores, from its first and the fourth quantile. These outliers indicate instances when the model has performed exceptionally well or poorly compared to the rest of the models. The two ends of the tails denote the highest and the lowest score.

In compliance with figure 3, *Greedy* has the highest mean (0.8084) among the existing ensemble methods, closely followed by $AE\_ensemble$ and $K - nn\_ensemble$ with mean *ROC* scores of 0.80100 and 0.97980, respectively. All three of these models have scores spread across a comparable range. The GMM models seem to under-perform, in terms of the mean, maximum, and minimum score achieved. Their highest scores are much lower than the lowest mean score from the rest of the models, which makes them undesirable

with the used set of parameters. As the number of their components increases, the mean also increases. The length of their candlesticks also decreases with the increasing number of components, with lesser variability, indicating a more consistent performance across different data sets.

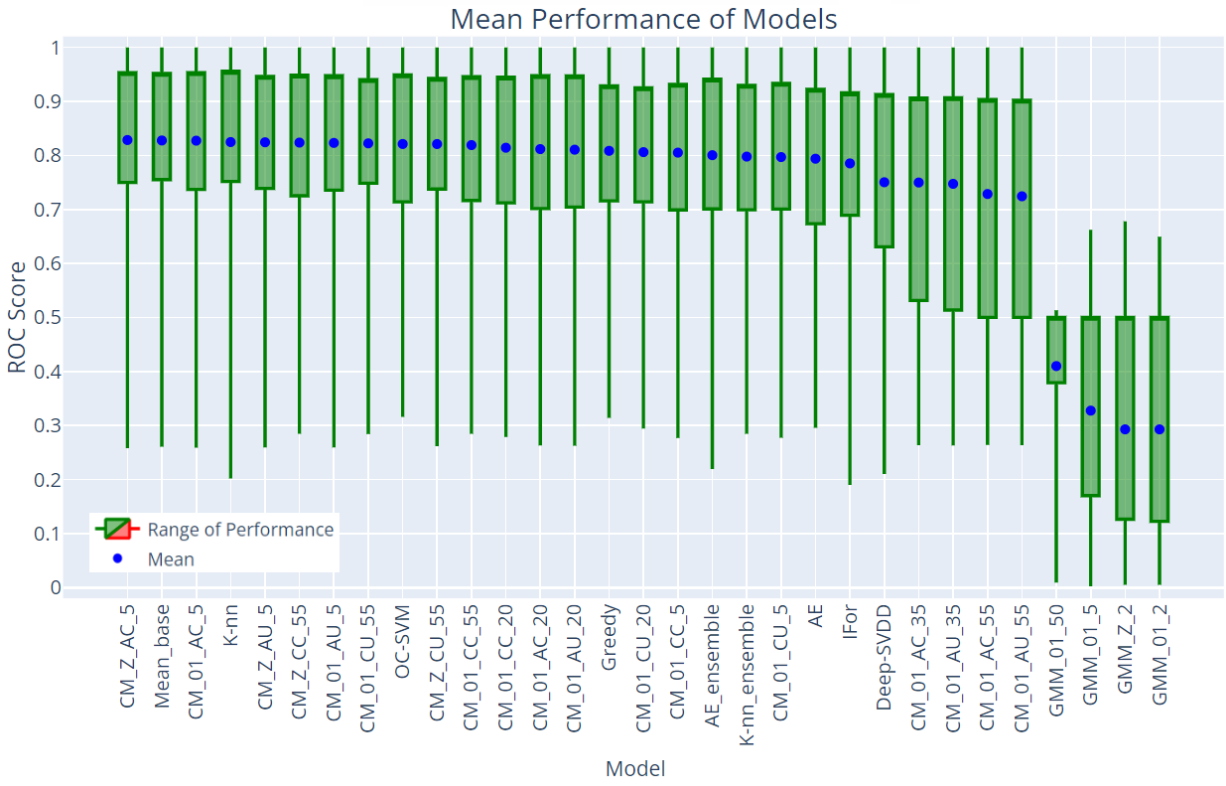


Figure 4: Mean of the *ROC* scores for each model - sorted by descending order of mean

4.3 Comparison of the Correlation Maximization Methods

The 18 Correlation Maximization models (3.2) are run for all the data sets and their mean *ROC* scores are generated and plotted, along with that of the rest of the models, in figure 5 (Grouped by method). Figure 4 is also used to compare the results of the *CM* ensembles with those of the other models.

From both figures, variations of the *CM* ensembles outperform most of the base models and the existing methods discussed. The highest mean among all the models is reached by *CM_Z_AC_5* (0.82840), which is a better score than that of *Mean_base* (0.82760). This is closely followed by *CM_01_AC_5* and then by *CM_Z_AU_5*. The range of their mean scores is between 0.72430 and 0.82840, indicating better performance than most models. The relatively comparable length and distribution of the candlesticks for the *CM* models show their consistency in performance. For the combination of $0-1$ *Scaling* with *Computational Optimization*, for both *Count* and *Unique* approaches for label generation, the mean increases with increasing value of k,

which is the number of anomalous points per data set chosen for label generation. But the same experiments repeated with *Approximation* method show decreasing means with increasing k. The 4 CM models using $Z - score\ normalization$ all perform well and are very similar to the ones using $0 - 1$ *Scaling*.

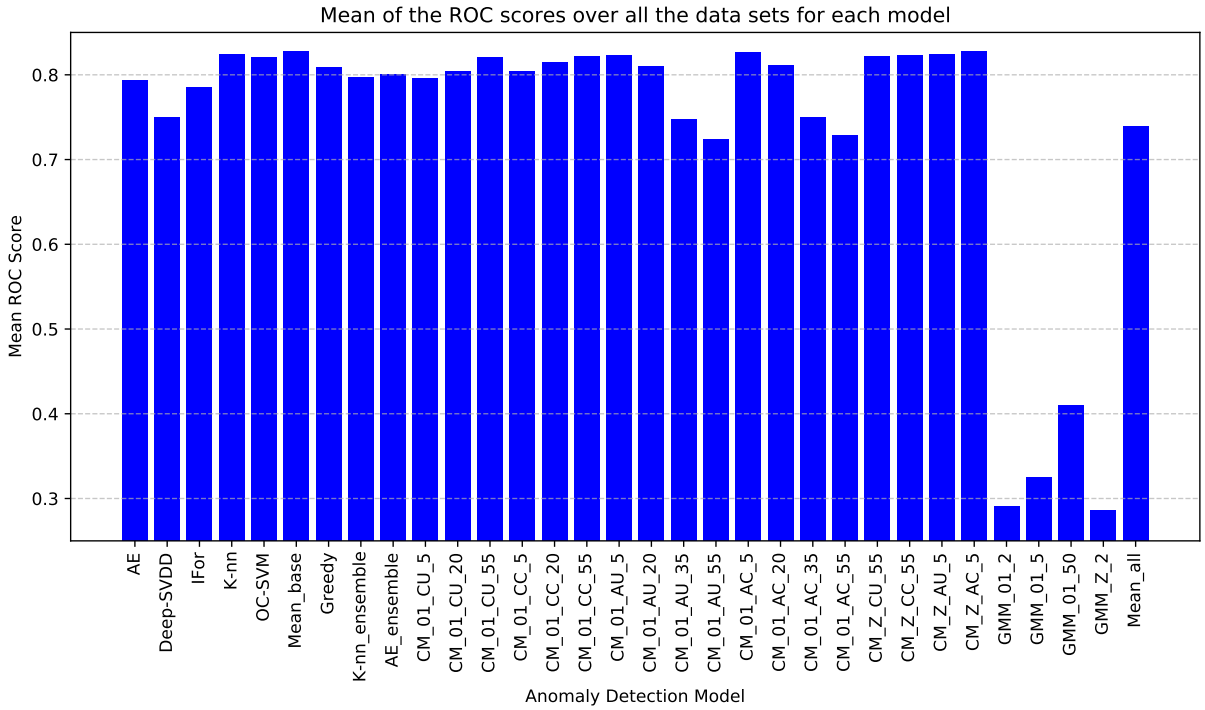


Figure 5: Mean of the *ROC* scores for each model - grouped by method category

With a very small mean difference, it is hard to compare the effects of changing the label generation approach from *Count* to *Unique*. Figure 6 shows the number of data sets in which each combination of the better-performing ensemble methods (with the highest mean score from each group) has performed best, or has the highest anomaly score, in the decreasing order of the frequency. Figure 12 in the appendix illustrates the same for all the 31 models, including the base models.

Individual base models have a higher frequency of wins. but it is not uniform, as each model specializes in identifying different kinds of anomalies. The simple combining function *Mean_base* has the highest wins among the ensembles - but this method is also unreliable as it gives equal weightage to both the meaningful and the irrelevant models. Next to this, *Greedy*, $CM\_Z\_AC\_$, and $CM\_01\_CU\_55$ have the next highest wins. The difference in the mean scores for the highest-ranking CM methods is very small, yet, the top-ranking means are with the *Approximation* approach, first for the *Count* method of label generation with a small k value and then for the *Unique* method for a small k. From figure 12, *Computational Optimization* approach with $0 - 1$ *Scaling* wins in most data sets, comparatively. It is still difficult to draw a concrete conclusion about the best

ensemble method with the best combination of parameters from analyzing their mean and average number of wins.

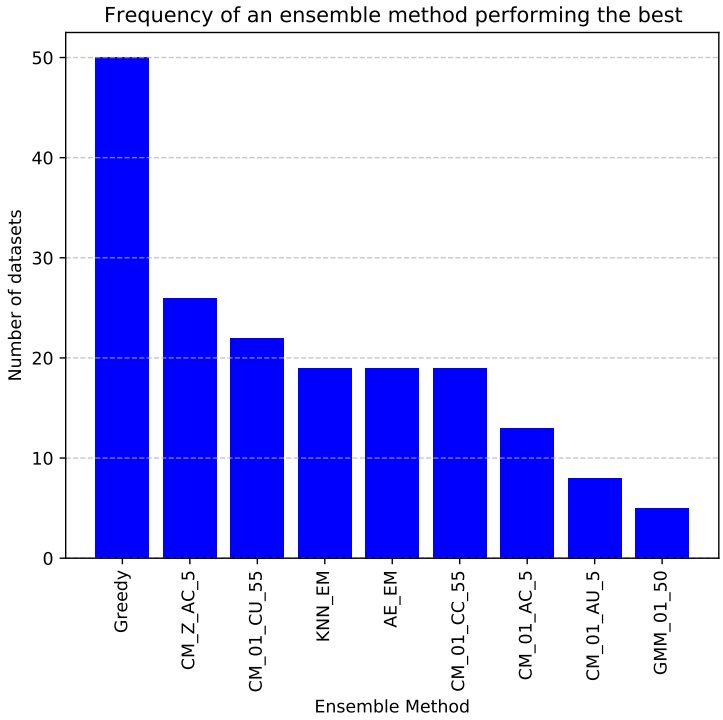
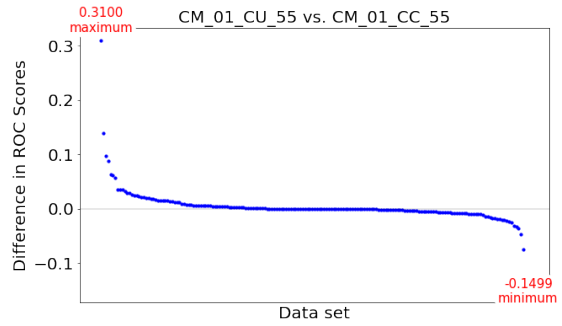


Figure 6: Count of data sets in which ensemble methods with the highest mean from each group have the highest ROC score

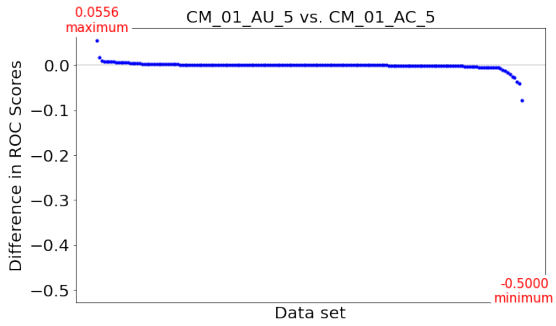
The models with the highest mean score from each of the group of combinations of the CM models, corresponding to the most optimal k value, are chosen as the better-performing CM models for further analysis. Figure 2 shows Scatter plots displaying the difference in ROC scores between pairs of the best-performing CM methods. The line $y = 0$ being close to the middle of the plot implies the differences are not very extreme. Extreme values above and below the $y = 0$ line denote large differences between the scores for specific data sets - positive extreme values above the line signify that, between the two methods compared, the first one has a larger value than the second one, and vice versa, for negative extremes. The maximum and the minimum of the differences are also highlighted.

In all four comparisons, the majority of the differences are around 0, showing that the ensembles give similar outputs in most data sets. On comparing $CM\_01\_CU\_55$ with $CM\_01\_CC\_55$ (2a) and $CM\_01\_AU\_5$ with $CM\_Z\_AU\_5$ (2d), both the plots have comparatively fewer extreme values equally distributed above and below the zero line, indicating that both the methods are similar and equally have instances when one score is higher than the other. This means that switching the label generation method from *Unique* to *Count* with all the other parameters being the same, for

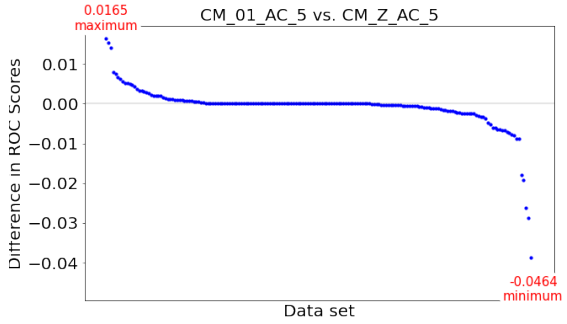
Computational Optimization, and also changing only the normalization method with all the other parameters remaining the same, for *Approximation* − *Unique*, does not affect the results much. For the comparison between *CM_01_AU_5* and *CM_01_AC_5* (2b), the zero line is closer to the top of the plot, but does not have relatively many negative extreme values, but one, which could be a potential outlier. Comparing *CM_01_AC_5* with *CM_Z_AC_5* (2c), the zero line is closer to the top of the plot, with many negative extreme values, signifying the existence of several data sets in which scores of the latter are higher than that of the former model. This means that, for the *Approximation* approach, switching between *Count* and *Unique* and also between the two normalization methods for *Count* beings noticeable differences between the models.



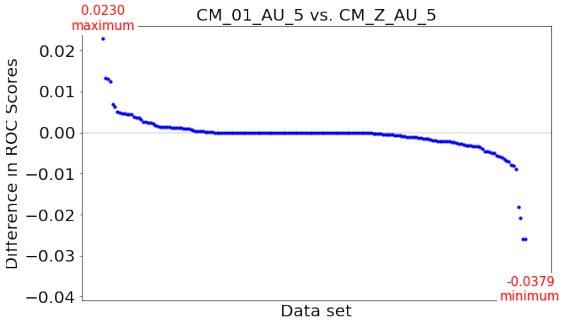
a *CM_01_CU_55* vs. *CM_01_CC_55*



b *CM_01_AU_5* vs. *CM_01_AC_5*



c *CM_01_AC_5* vs. *CM_Z_AC_5*



d *CM_01_AU_5* vs. *CM_Z_AU_5*

Table 2: Scatter plots showing difference in *ROC* scores between two models for each data set

4.4 Comparison against Base Models and Existing Ensembles

To understand the distribution of the *ROC* scores for the models, their *Kernel Density Estimates* are plotted. Figure 13 in the Appendix shows the entire distribution of all the models, whereas, figure 7 shows the peaks in the density of some

of the better-performing models. The density along the Y-axis denotes the probability of observing a value at a particular value of the score.

Only the GMM models seem to peak at around 0.5, indicating that most of their scores are around the value of 0.5. Most of the other models peak between 0.75 and 1.0, which is the highest score achievable. On having a closer look at the peaks of the better-performing models from figure 7, the more commonly occurring scores are between 0.9 and 0.95. In this range, the base model $K-nn$ has the highest peak, followed by the base model $OC-SVM$, ensembles $Greedy$, $K-nn\_ensemble$, $AE\_ensemble$, the CM models, and finally by the GMM method, in that order. This implies that the base models $K-nn$ and $OC-SVM$ have scores around the highest values consistently for many data sets. This might not necessarily be favorable as the base models might fail to capture some anomalies of other aspects that they do not specialize in. Among the ensembles, $CM\_Z\_AC\_5$ has more consistent peak scores. The CM ensembles (more specifically $CM\_01\_AC\_5$, $CM\_01\_CU\_55$, $CM\_01\_AU\_5$, $CM\_01\_CC\_55$, in that order) perform consistently, giving a robust and a higher range of anomaly scores. They are closely followed by the $Greedy$ ensemble - but this is unreliable as it might stop at local minima, instead of the global optimal value. $K-nn\_ensemble$ and $AE\_ensemble$ also have peaks along the same range but have a relatively lesser density at these scores.

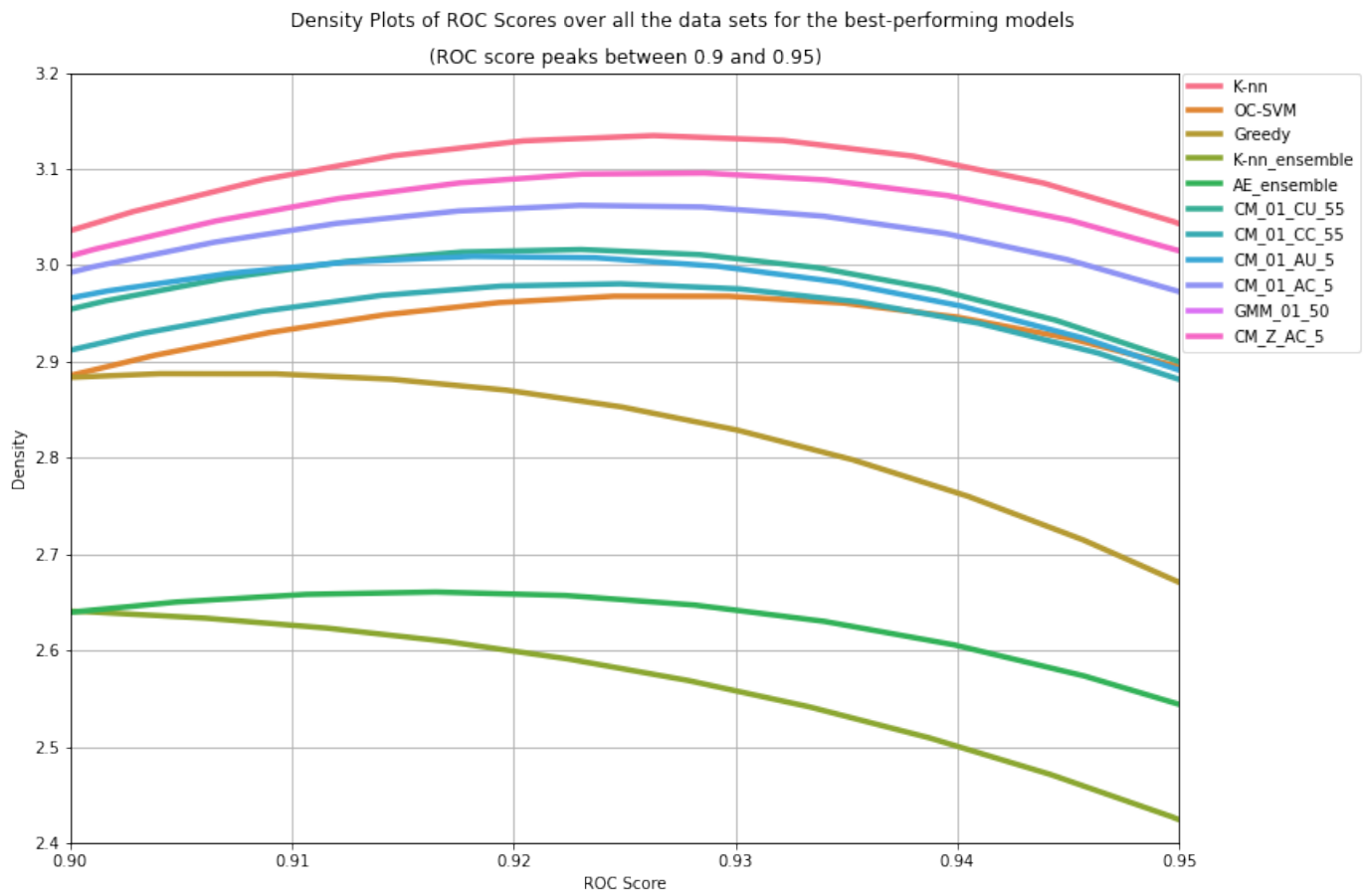


Figure 7: Kernel Density Estimate Plots of ROC Scores for the best-performing models

To compare any two models, a Stacked plot can also be used, plotting the ordered sum of their ROC scores. Figure 8 portrays the sum of the scores from the base model with the highest mean score $K-nn$ and the ensemble with the highest mean score $CM\_01\_CU\_55$ across each data set. For each data set, the height of the stacked bar plots corresponding to the individual models appears relatively equal. In more than 40% of the data sets, the sum of scores is above 1.75, indicating good performance. Unless specific values are looked at for the sake of the argument, the graph shows similar performance between $K-nn$ and $CM\_01\_CU\_55$. Yet, the ensemble is more robust, because of the culmination of different aspects in understanding a data set.

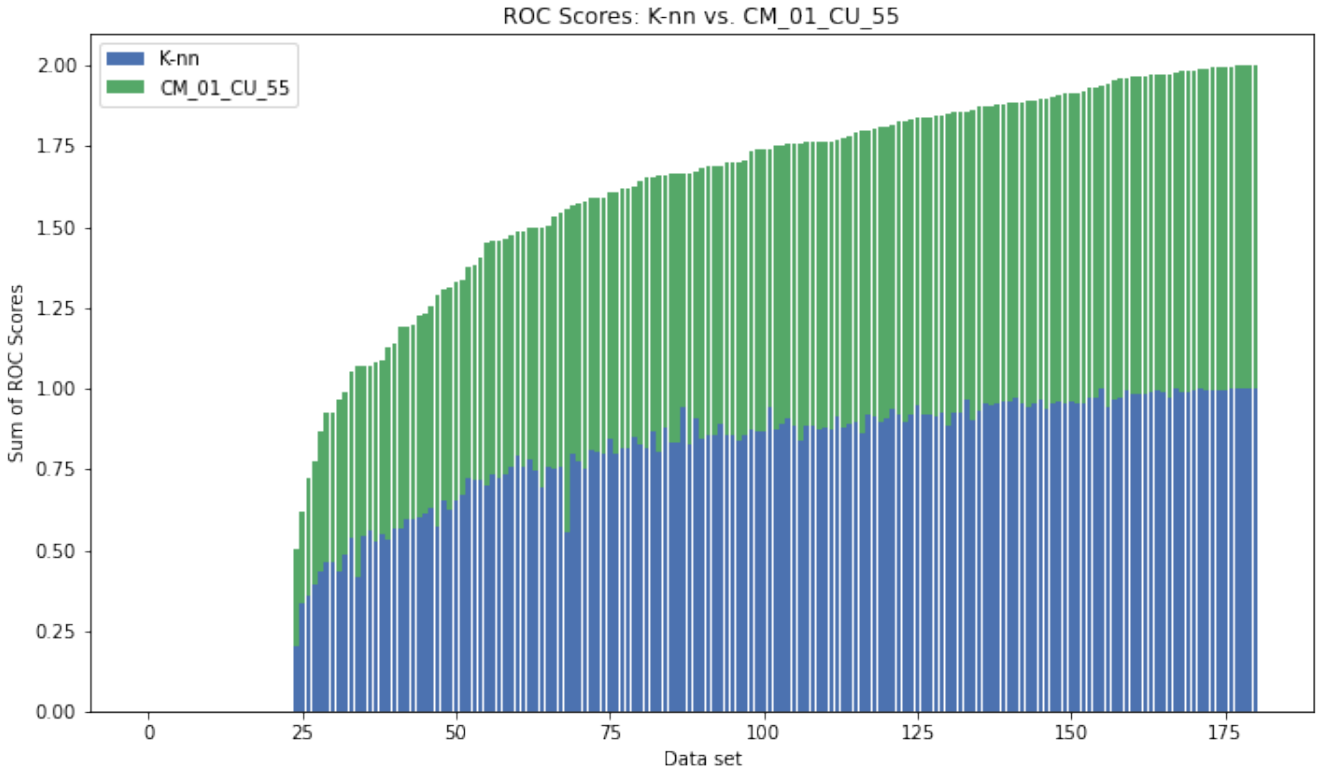


Figure 8: Stacked plot showing the sum of *ROC* scores from two methods

4.5 Statistical Tests

Statistical tests are used for evaluating and interpreting the results of the multiple models. Wilcoxon Tests (2.4.2) are conducted to statistically compare results from any two methods. The better-performing models are picked and combinations of their results are compared. The results of these Wilcoxon tests are listed in table 3. Supporting the argument from the examination of the scatter plots for the difference in scores from subsection 4.3, changing the label generation method from *Unique* to *Count* does not make much of a difference between CM ensembles using *Computational Optimization* for a k value of 55 with $0-1$ *Scaling*. Similarly, there is no statistically significant difference between the performance of the same CM models using *Computational Optimization* for a k value of 55 when the normalization methods are swapped. On the other hand, changing from *Computational Optimization* to *Approximation* between $CM\_01\_CU\_55$ and $CM\_01\_AU\_55$ results in a statistically significant difference between their performances. A similar difference is observed when changing the normalization method of the CM ensemble while using *Approximation* with $k = 5$. Comparing the model with the highest mean score *Mean_base* and the one with the lowest, GMM_01_50, shows an expected significant

difference. Comparing specific combinations of the CM ensemble with that of the GMM ensemble also shows significant differences.

Table 3: Wilcoxon Test Results with 5% significance level

Models Compared		**Wilcoxon Statistic**	***p*-value**	**Statistically significant difference between performances**
Mean_base	*GMM_01_50*	104.0	$1.17872e-26$	yes
CM_01_CU_55	*CM_01_CC_55*	4921.0	0.64735	no
CM_01_CU_55	*CM_01_AU_55*	4389.0	0.00215	yes
CM_01_CU_55	*GMM_01_50*	112.0	$1.37082e-26$	yes
CM_01_AU_5	*CM_Z_AU_5*	2856.5	0.02221	yes
CM_01_CU_55	*CM_Z_CU_55*	2594.5	0.44749	no

To compare more than just two, but a set of models at a time, Post-hoc Nemenyi tests are performed for two subsets of the models - one comparing the better-performing CM methods against *AE_ensemble* and *Greedy* and the other comparing all the better-performing CM methods (with varying parameters) among each other. Critical Difference plots are built to visualize the results of both the Nemenyi tests - figure 9 for the former subset of models and figure 10 for the latter. The relationship in terms of correlation between the performance of these subsets of models is also evaluated using Heat maps, in which cells with shades close to *red* (close to 1) show higher correlation between model performances and the ones with shades close to *blue* (close to 0) show lesser correlation - figure 14 and figure 15, respectively.

Models are ranked along the x-axis based on the *median* of their performance and the dotted red line indicates the critical difference (CD). Pairs of methods with ranks that differ by more than the CD are significantly different with 5% significance. The vertical black lines connecting multiple models denote that all those models are not significantly different. Similarly, models, for which such vertical lines do not overlap, are significantly different.

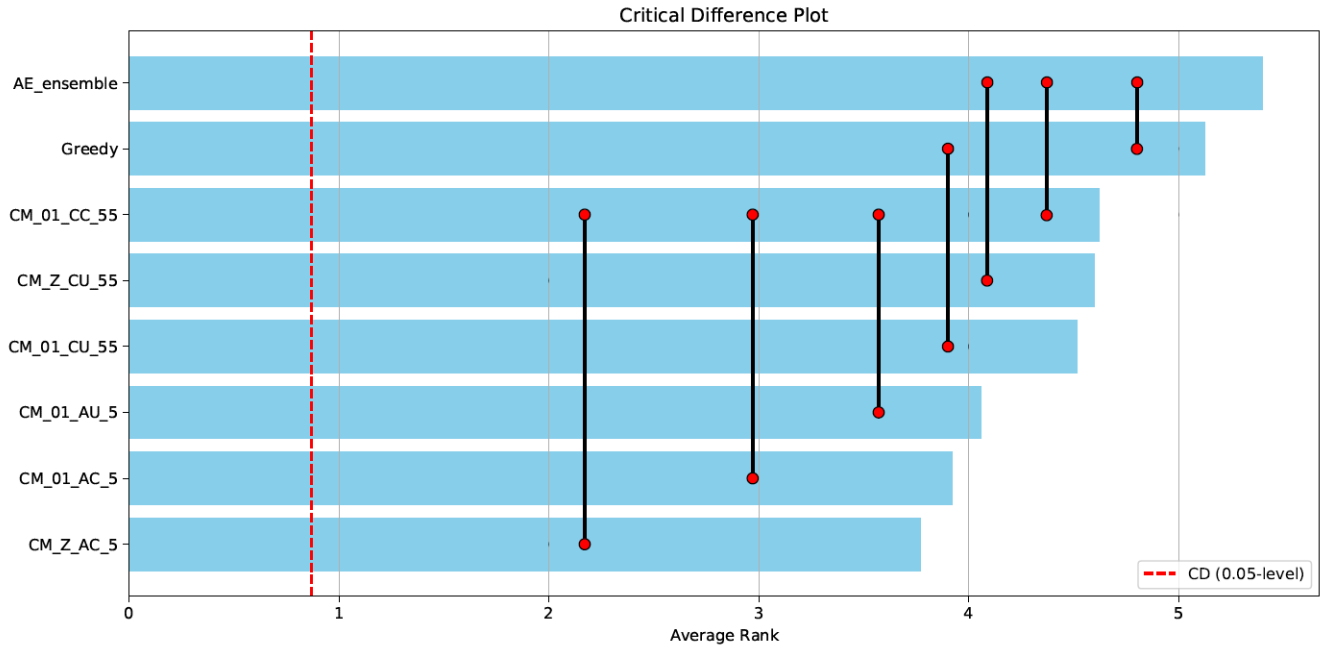


Figure 9: Critical Difference Diagram for the best-performing ensembles

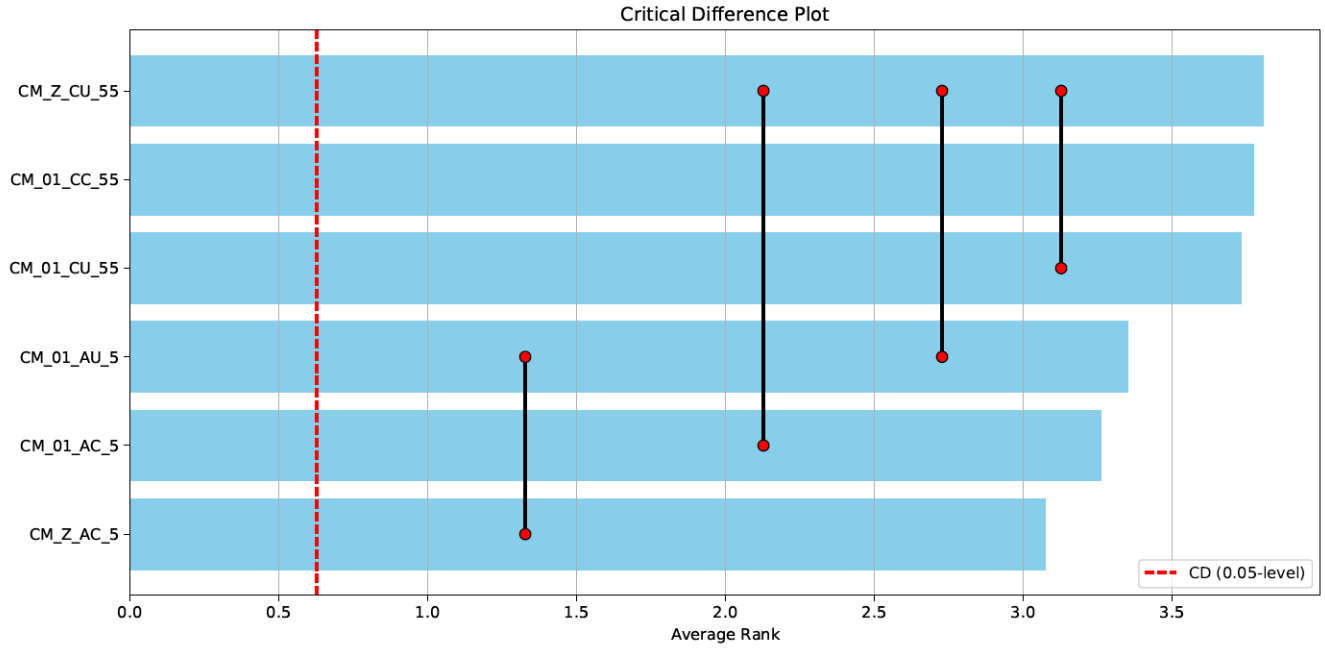


Figure 10: Critical Difference Diagram for the best-performing Correlation Maximization ensembles

From figure 9, it is observed that there is a statistically significant difference between the performance of the set of CM models compared with the combination of *AE_ensemble* and *Greedy*. A comparatively higher average rank for the existing ensembles as compared to the CM ensembles indicates that the former models have performed worse on average across multiple datasets or experimental conditions when

compared to the latter models. Figure 10 explains that there is a significant difference between the performance of the CM models using the *Computational Optimization* approach and *Approximation* approach but there is no significant difference observed when changing the method of normalization for the same set of parameters. The Heat Maps also resonate with this observation, showing a higher correlation between CM models for variations of *Unique* and *Count* approaches and variations of the normalization methods over the same set of parameters. The CM models using the *Computational Optimization* approach for either normalization methods and a higher value of $k = 55$ show better performance in the previous subsections of analysis but have a higher average rank in the CD diagram - this suggests that the performance of these models may be sub-optimal under certain conditions and could benefit from further optimization or refinement, such as feature selection, parameter tuning, or algorithmic enhancements.

5 Conclusion and Outlook

Anomaly detection, a crucial process in data analysis, involves finding a subset of a data set that behaves differently when compared with the rest of the data. This thesis aims to enhance the effectiveness of anomaly detection techniques by trying to tackle the challenges often encountered in practice - the limited availability of labeled data and the individual anomaly detection algorithms detecting only specific types of anomalies efficiently and not all of them. For this purpose, stacking ensemble methods are employed for multiple data sets and anomaly detection algorithms as these approaches help to combine the strengths of the individual models to potentially improve overall outlier detection performance. Furthermore, an ensemble method called *Correlation Maximization* is formulated and evaluated with different combinations of parameters, against the existing methods.

5 diverse base anomaly detection models, 8 different existing ensemble methods, and 18 different variations (in parameters) of the suggested *Correlation Maximization* ensemble are applied to 181 different data sets, and their results are analyzed and evaluated in comparison to each other. The anomaly scores of each of the models for all the data sets are calculated and statistical tests are carried out for the assessment of their performance. It is found that, though the base models give a fairly high anomaly score in most data sets, they are comparatively less reliable, as they might miss identifying specific anomalies in some cases. Among the ensemble methods employed, the existing ensembles tested give better results in a few of the data sets. More focus is laid on the *Correlation Maximization* method in comparison with these results.

The suggested *Correlation Maximization* method seems to outperform the existing methods in several cases, under different conditions, in terms of mean anomaly score across all data sets. In the suggested *Computational Optimization* approach, a higher value of anomalies to be induced (k) seems favorable in most cases. Whereas, in the other *Approximation* approach, a lower value of k is to be preferred. Both the approaches tried for label generation and normalization do not seem to have a considerable effect on the results - all combinations are found to perform well with specific subject data. Although it is difficult to pin down one specific combination of parameters with which the method universally performs the best, the *Correlation Maximization* does portray promising results worth further exploration.

The experiments in this thesis are to be further extended in future studies to modify and refine existing anomaly detection methods and their corresponding parameters to get more advanced base models with an improved overall understanding of the data and its anomalies. Additionally, the suggested *Correlation Maximization* ensemble is to be further experimented with different combinations of parameters and testing methods to find the optimal number of anomalies to be induced (k value), in order to be more case-specific to the subject data. More complex stacking combinations of models including neural networks, with different normalization techniques and parameters, are to be tried out. It is also crucial to identify niches where specific methods or anomaly detection rules work best, to get a more robust, accurate, condition-specific, and improved performance from the Anomaly Detection Models. This could possibly aid in further enhancing the efficacy and applicability of such anomaly detection techniques in real-world scenarios.

List of Tables

1 *ROC* scores from all the models for data set *cardio* 20

2 Scatter plots showing difference in *ROC* scores between two models for each data set 25

3 Wilcoxon Test Results with 5% significance level 29

4 Mean of the Ensemble model *ROC* scores over all the data sets 42

List of Figures

1 Architecture of Models 17

2 Percentage of data sets in which each of the base models has performed the best 19

3 Number of data sets for which each of the existing ensemble methods has the highest score 21

4 Mean of the *ROC* scores for each model - sorted by descending order of mean 22

5 Mean of the *ROC* scores for each model - grouped by method category . . 23

6 Count of data sets in which ensemble methods with the highest mean from each group have the highest *ROC* score 24

7 Kernel Density Estimate Plots of ROC Scores for the best-performing models 27

8 Stacked plot showing the sum of *ROC* scores from two methods 28

9 Critical Difference Diagram for the best-performing ensembles 30

10 Critical Difference Diagram for the best-performing Correlation Maximization ensembles 30

11 *ROC* scores (> 0.9250) from all the models for data set *cardio*, sorted by decreasing score 38

12 Count of data sets in which each model has the highest *ROC* score 39

13 Kernel Density Estimate Plots of *ROC* Scores for all models 39

14 Heat Map to visualize the Nemenyi Post-hoc Test results for the best-performing ensembles 40

15 Heat Map to visualize the Nemenyi Post-hoc Test results for the best-performing Correlation Maximization ensembles 41

Bibliography

[1] Frank Wilcoxon. "Individual comparisons by ranking methods". In: *Breakthroughs in Statistics: Methodology and Distribution*. Springer, 1992, pp. 196–202.

[2] Thomas G Dietterich. "An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization". In: *Machine learning* 40 (2000), pp. 139–157.

[3] Sridhar Ramaswamy, Rajeev Rastogi, and Kyuseok Shim. "Efficient algorithms for mining outliers from large data sets". In: *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*. 2000, pp. 427–438.

[4] Bernhard Schölkopf et al. "Estimating the support of a high-dimensional distribution". In: *Neural computation* 13.7 (2001), pp. 1443–1471.

[5] Janez Demšar. "Statistical comparisons of classifiers over multiple data sets". In: *The Journal of Machine learning research* 7.1 (2006), pp. 1–30.

[6] J. D. Hunter. "Matplotlib: A 2D graphics environment". In: *Computing in Science & Engineering* 9 (2007). DOI: 10.1109/MCSE.2007.55.

[7] Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. "Isolation forest". In: *2008 eighth ieee international conference on data mining*. IEEE. 2008, pp. 413–422.

[8] F. et al. Pedregosa. "Scikit-learn: Machine Learning in Python". In: *Journal of Machine Learning Research* 12 (2011).

[9] Larry Shoemaker and Lawrence O Hall. "Anomaly detection using ensembles". In: *Multiple Classifier Systems: 10th International Workshop, MCS 2011, Naples, Italy, June 15-17, 2011. Proceedings 10*. Springer. 2011, pp. 6–15.

[10] Ikewelugo Cyprian Anaene Oyeka, Godday Uwawunkonye Ebuh, et al. "Modified Wilcoxon signed-rank test". In: *Open Journal of Statistics* 2.2 (2012), pp. 172–176.

[11] Erich Schubert et al. "On evaluation of outlier rankings and outlier scores". In: *Proceedings of the 2012 SIAM international conference on data mining*. SIAM. 2012, pp. 1047–1058.

[12] Michael C Thomsett. *Bloomberg visual guide to candlestick charting*. Vol. 1. John Wiley & Sons, 2012.

[13] Michael Glodek, Martin Schels, and Friedhelm Schwenker. "Ensemble Gaussian mixture models for probability density estimation". In: *Computational statistics* 28 (2013), pp. 127–138.

[14] Aman Kataria and MD Singh. "A review of data classification using k-nearest neighbour algorithm". In: *International Journal of Emerging Technology and Advanced Engineering* 3.6 (2013), pp. 354–360.

[15] Martin et al. Abadi. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. 2015. URL: https://www.tensorflow.org/.

[16] Charu C Aggarwal and Saket Sathe. "Theoretical foundations and algorithms for outlier ensembles". In: *Acm sigkdd explorations newsletter* 17.1 (2015), pp. 24–47.

[17] Charu C Aggarwal. *Outlier analysis second edition*. 2016.

[18] Markus Goldstein and Seiichi Uchida. "A comparative evaluation of unsupervised anomaly detection algorithms for multivariate data". In: *PloS one* 11.4 (2016), e0152173.

[19] Alvin Chiang et al. "A study on anomaly detection ensembles". In: *Journal of Applied Logic* 21 (2017), pp. 1–13.

[20] Niki Veček, Matej Črepinšek, and Marjan Mernik. "On the influence of the number of algorithms, problems, and independent runs in the comparison of evolutionary algorithms". In: *Applied Soft Computing* 54 (2017), pp. 23–45.

[21] Zhiruo Zhao. "Ensemble methods for anomaly detection". PhD thesis. Syracuse University, 2017.

[22] Raghavendra Chalapathy, Aditya Krishna Menon, and Sanjay Chawla. "Anomaly detection using one-class neural networks". In: *arXiv preprint arXiv:1802.06360* (2018).

[23] Robert Grant. *Data visualization: charts, maps, and interactive graphics*. Chapman and Hall/CRC, 2018.

[24] Lukas Ruff et al. "Deep one-class classification". In: *International conference on machine learning*. PMLR. 2018, pp. 4393–4402.

[25] Xiaoyi Gu, Leman Akoglu, and Alessandro Rinaldo. "Statistical analysis of nearest neighbor methods for anomaly detection". In: *Advances in Neural Information Processing Systems* 32 (2019).

[26] Rising Odegua. "An empirical study of ensemble techniques (bagging, boosting and stacking)". In: *Proc. Conf.: Deep Learn. IndabaXAt*. 2019.

[27] Yue Zhao, Zain Nasrullah, and Zheng Li. "Pyod: A python toolbox for scalable outlier detection". In: *arXiv preprint arXiv:1901.01588* (2019).

[28] Charles R. et al. Harris. "Array programming with NumPy". In: *Nature* 585 (2020). DOI: 10.1038/s41586-020-2649-2.

[29] Wes McKinney and Team. *pandas-dev/pandas: Pandas*. Version latest. 2020. DOI: 10.5281/zenodo.3509134. URL: https://doi.org/10.5281/zenodo.3509134.

[30] Pauli et al. Virtanen. "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python". In: *Nature Methods* 17 (2020). DOI: 10.1038/s41592-019-0686-2.

[31] Zhen Cheng et al. “Improved autoencoder for unsupervised anomaly detection”. In: *International Journal of Intelligent Systems* 36.12 (2021), pp. 7103–7125.

[32] Henderi Henderi, Tri Wahyuningsih, and Efana Rahwanto. “Comparison of Min-Max normalization and Z-Score Normalization in the K-nearest neighbor (kNN) Algorithm to Test the Accuracy of Types of Breast Cancer”. In: *International Journal of Informatics and Information Systems* 4.1 (2021), pp. 13–20.

[33] Michael L. Waskom. “seaborn: statistical data visualization”. In: *Journal of Open Source Software* 6 (2021). DOI: 10.21105/joss.03021. URL: https://doi.org/10.21105/joss.03021.

[34] Zheng Zhang and Xiaogang Deng. “Anomaly detection using improved deep SVDD model with data structure preservation”. In: *Pattern Recognition Letters* 148 (2021), pp. 1–6.

[35] Simon Klüttermann. *Yano*. 2023. URL: https://doi.org/10.5281/zenodo.7520448.

[36] Simon Klüttermann and Emmanuel Müller. “Evaluating and Comparing Heterogeneous Ensemble Methods for Unsupervised Anomaly Detection”. In: *2023 International Joint Conference on Neural Networks (IJCNN)*. IEEE. 2023, pp. 1–8.

[37] Durgesh Samariya and Amit Thakkar. “A comprehensive survey of anomaly detection algorithms”. In: *Annals of Data Science* 10.3 (2023), pp. 829–850.

[38] Guido Van Rossum et al. *Python Programming Language*. Python Software Foundation. Wilmington, Delaware, United States, 2023. URL: https://www.python.org//.

[39] Hongzuo Xu et al. “Deep isolation forest for anomaly detection”. In: *IEEE Transactions on Knowledge and Data Engineering* (2023).

Appendix

5.1 Additional Figures

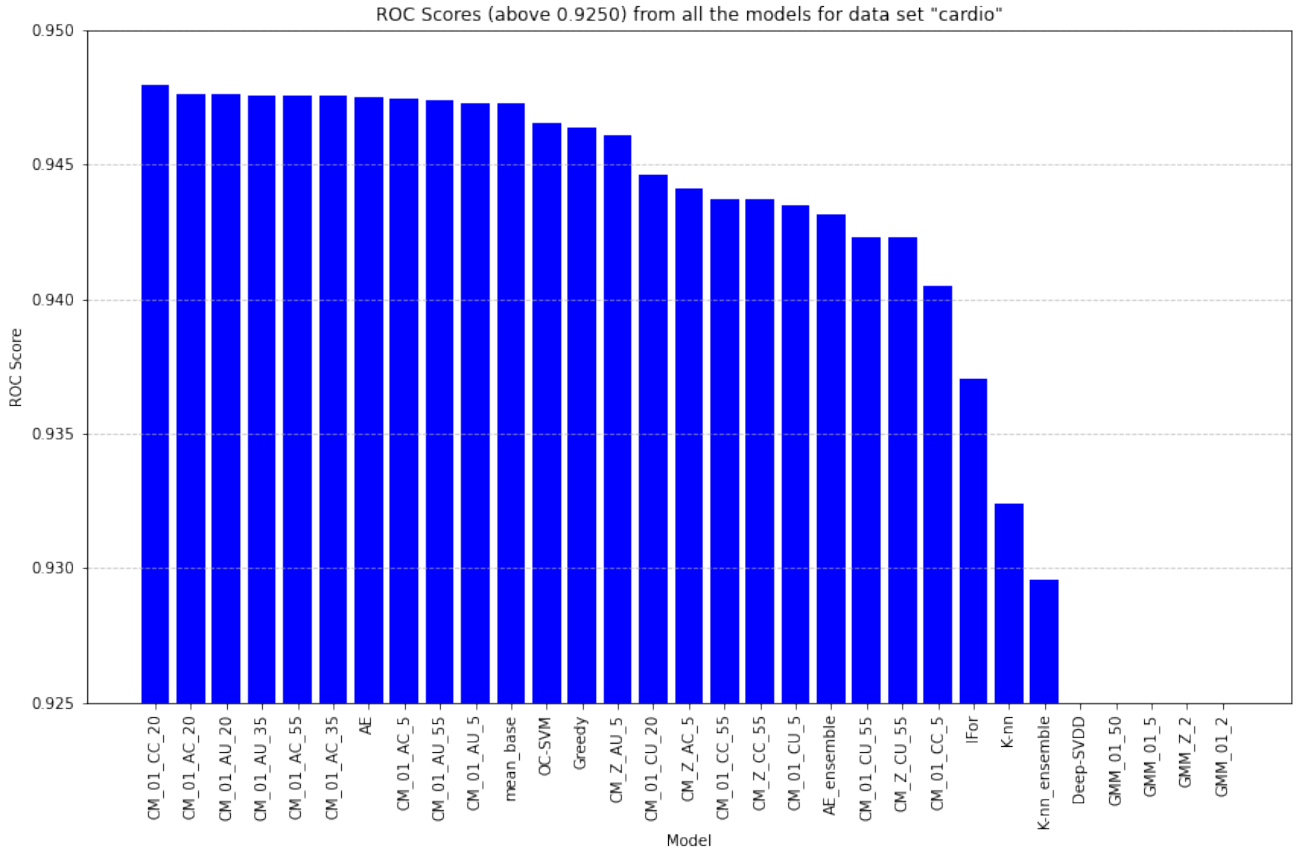


Figure 11: *ROC* scores (> 0.9250) from all the models for data set *cardio*, sorted by decreasing score

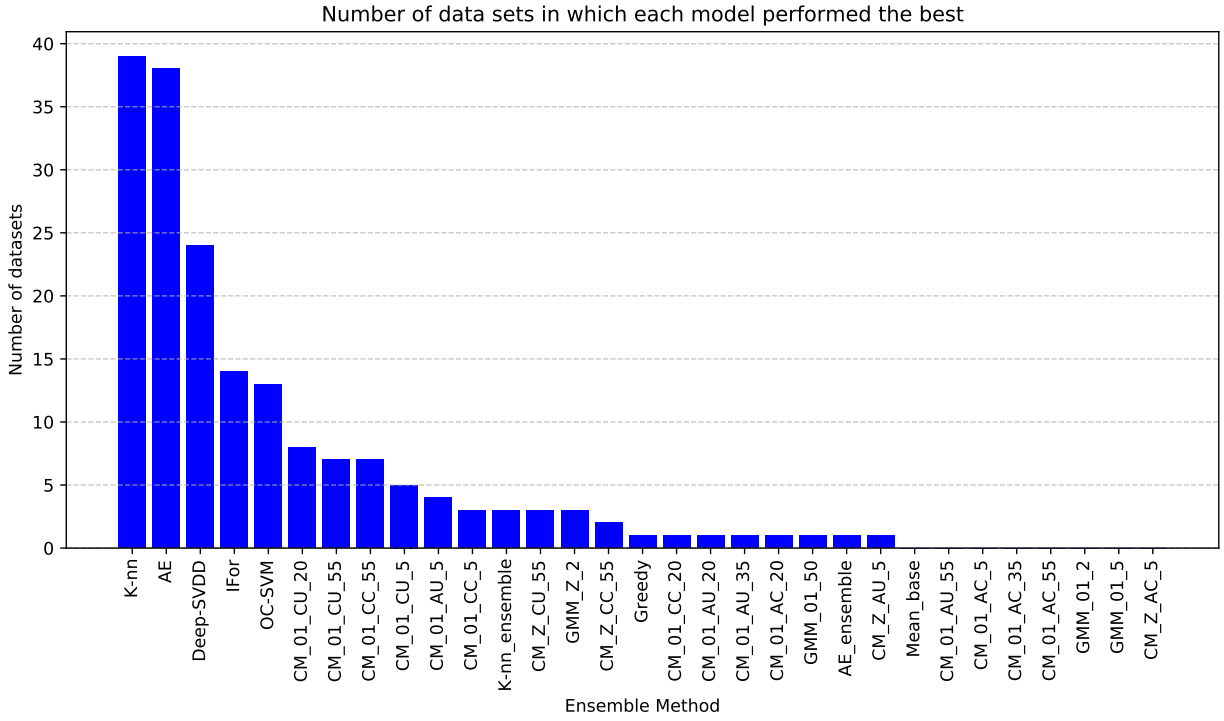


Figure 12: Count of data sets in which each model has the highest *ROC* score

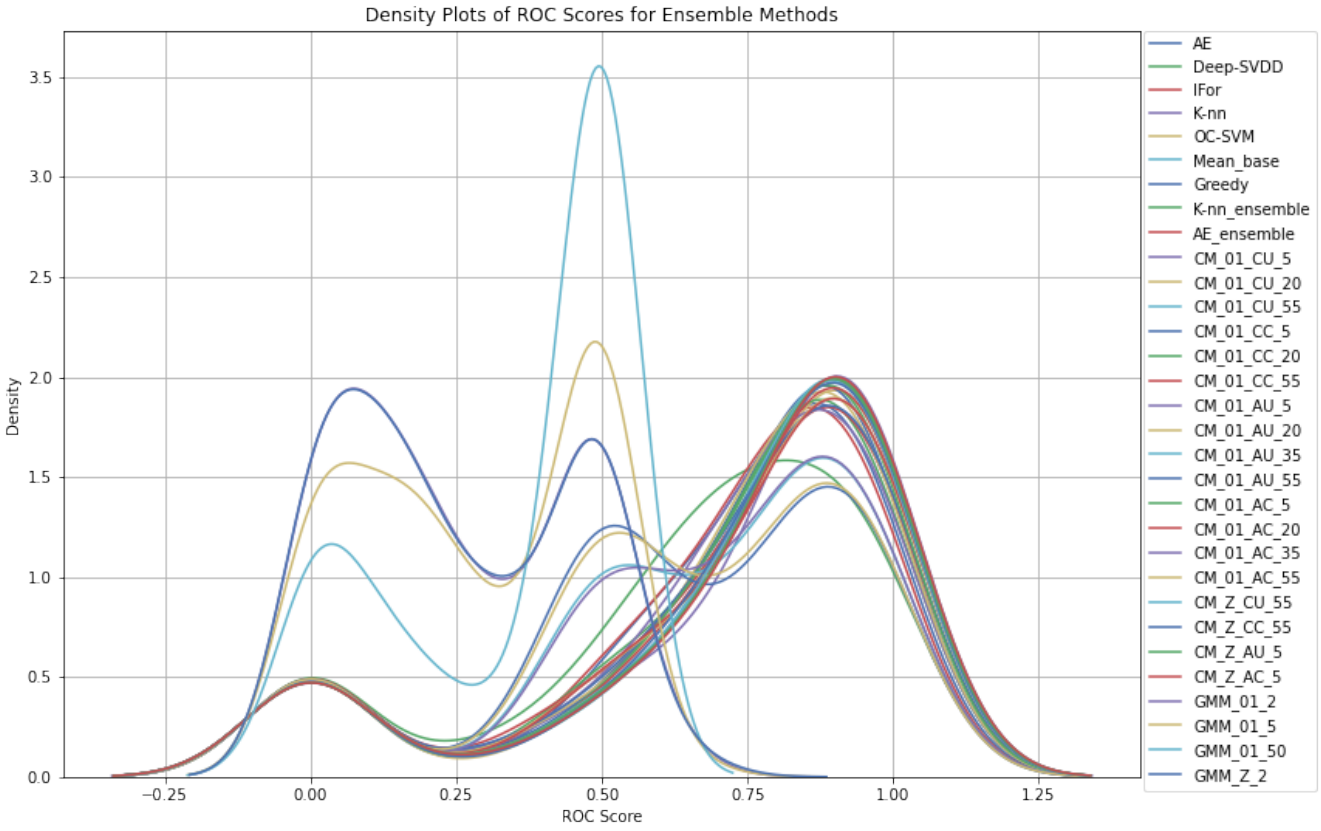


Figure 13: Kernel Density Estimate Plots of *ROC* Scores for all models

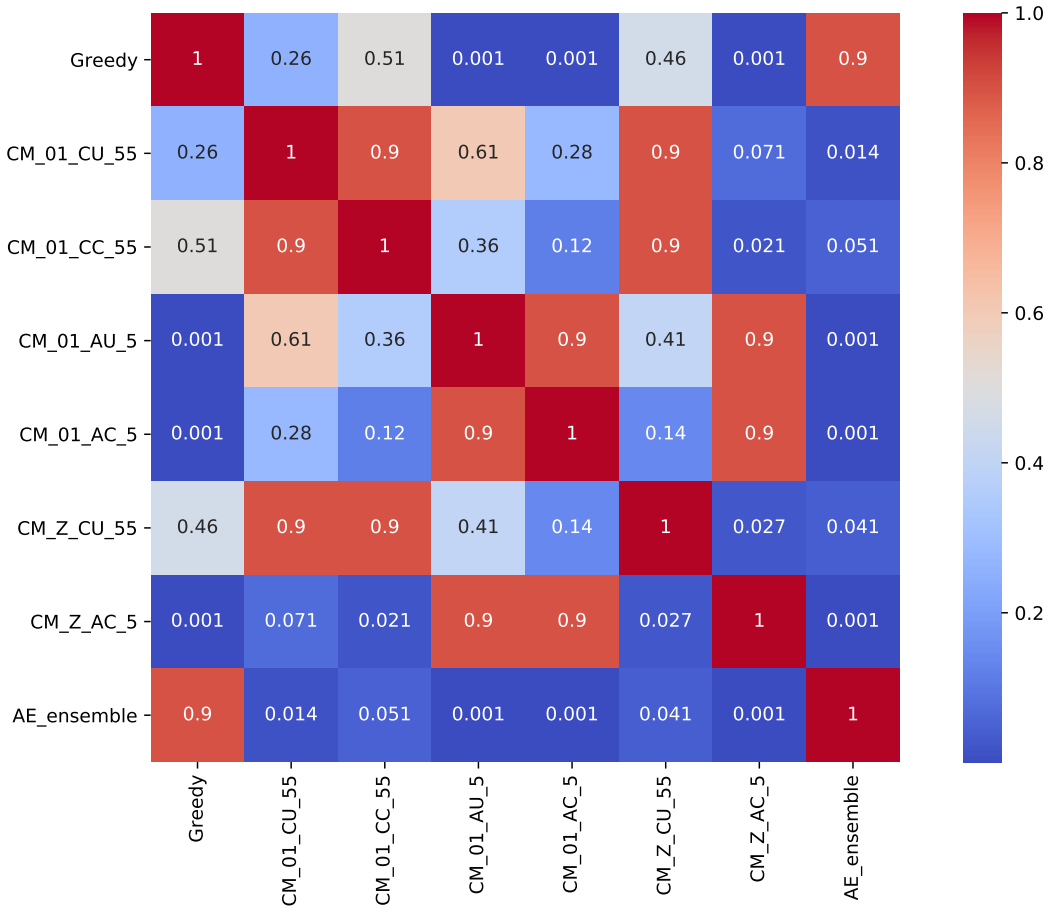


Figure 14: Heat Map to visualize the Nemenyi Post-hoc Test results for the best-performing ensembles

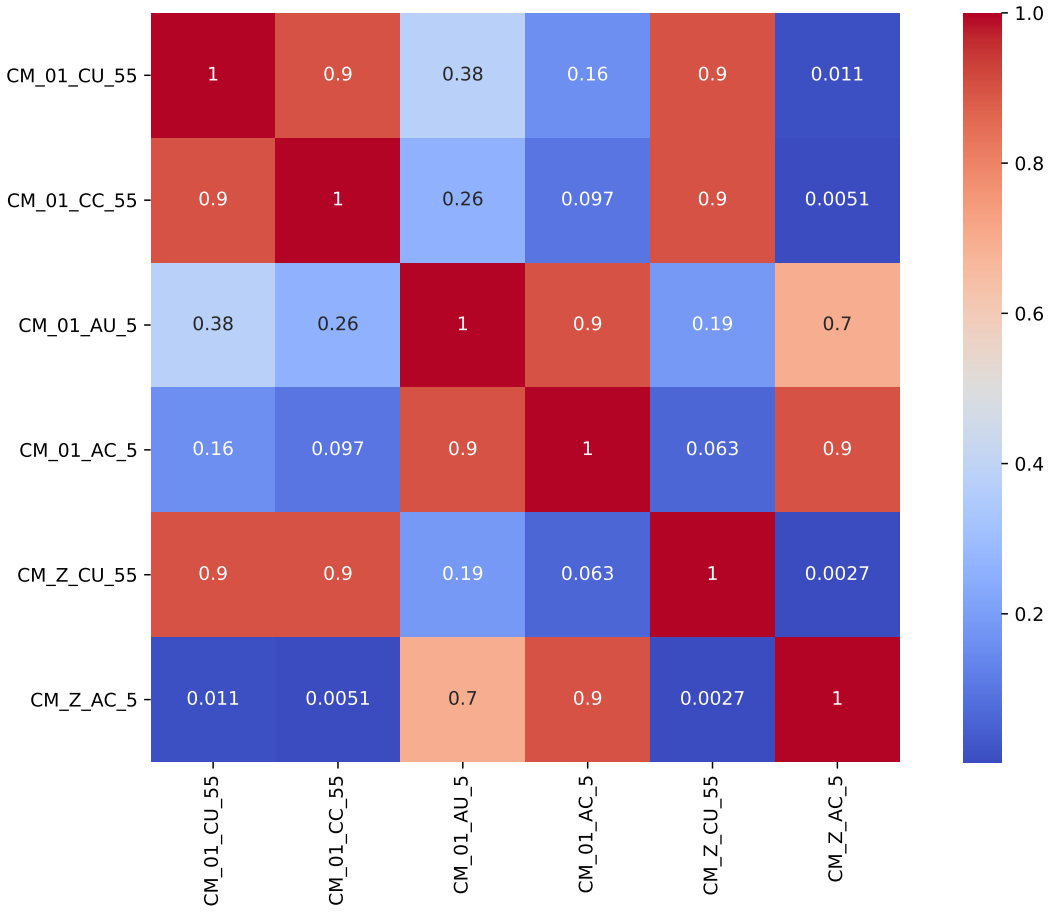


Figure 15: Heat Map to visualize the Nemenyi Post-hoc Test results for the best-performing Correlation Maximization ensembles

5.2 Additional Tables

Table 4: Mean of the Ensemble model *ROC* scores over all the data sets

Model	*ROC* **score**
$Mean\_base$	0.82760
$Greedy$	0.80840
$K-nn\_ensemble$	0.79780
$AE\_ensemble$	0.80100
$CM\_01\_CU\_5$	0.79650
$CM\_01\_CU\_20$	0.80480
$CM\_01\_CU\_55$	0.82120
$CM\_01\_CC\_5$	0.80410
$CM\_01\_CC\_20$	0.81500
$CM\_01\_CC\_55$	0.82220
$CM\_01\_AU\_5$	0.82310
$CM\_01\_AU\_20$	0.81060
$CM\_01\_AU\_35$	0.74730
$CM\_01\_AU\_55$	0.72430
$CM\_01\_AC\_5$	0.82720
$CM\_01\_AC\_20$	0.81160
$CM\_01\_AC\_35$	0.74970
$CM\_01\_AC\_55$	0.72850
$CM\_Z\_CU\_55$	0.82170
$CM\_Z\_CC\_55$	0.82330
$CM\_Z\_AU\_5$	0.82420
$CM\_Z\_AC\_5$	0.82840
GMM_01_2	0.29280
GMM_01_5	0.32670
GMM_01_50	0.41020
GMM_Z_2	0.28830

Eidesstattliche Versicherung

(Affidavit)

Thiyaagu, Haritha

Name, Vorname
(surname, first name)

230289

Matrikelnummer
(student ID number)

☐ Bachelorarbeit
(Bachelor's thesis)

■ Masterarbeit
(Master's thesis)

Titel
(Title)

Stacking Ensemble Methods for Anomaly Detection

Ich versichere hiermit an Eides statt, dass ich die vorliegende Abschlussarbeit mit dem oben genannten Titel selbstständig und ohne unzulässige fremde Hilfe erbracht habe. Ich habe keine anderen als die angegebenen Quellen und Hilfsmittel benutzt sowie wörtliche und sinngemäße Zitate kenntlich gemacht. Die Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

I declare in lieu of oath that I have completed the present thesis with the above-mentioned title independently and without any unauthorized assistance. I have not used any other sources or aids than the ones listed and have documented quotations and paraphrases as such. The thesis in its current or similar version has not been submitted to an auditing institution before.

Mannheim, 20.03.2024

Ort, Datum
(place, date)

Haritha Thiyaagu

Unterschrift
(signature)

Belehrung:
Wer vorsätzlich gegen eine die Täuschung über Prüfungsleistungen betreffende Regelung einer Hochschulprüfungsordnung verstößt, handelt ordnungswidrig. Die Ordnungswidrigkeit kann mit einer Geldbuße von bis zu 50.000,00 € geahndet werden. Zuständige Verwaltungsbehörde für die Verfolgung und Ahndung von Ordnungswidrigkeiten ist der Kanzler/die Kanzlerin der Technischen Universität Dortmund. Im Falle eines mehrfachen oder sonstigen schwerwiegenden Täuschungsversuches kann der Prüfling zudem exmatrikuliert werden. (§ 63 Abs. 5 Hochschulgesetz - HG -).

Die Abgabe einer falschen Versicherung an Eides statt wird mit Freiheitsstrafe bis zu 3 Jahren oder mit Geldstrafe bestraft.

Die Technische Universität Dortmund wird ggf. elektronische Vergleichswerkzeuge (wie z.B. die Software „turnitin") zur Überprüfung von Ordnungswidrigkeiten in Prüfungsverfahren nutzen.

Die oben stehende Belehrung habe ich zur Kenntnis genommen:

Official notification:
Any person who intentionally breaches any regulation of university examination regulations relating to deception in examination performance is acting improperly. This offense can be punished with a fine of up to EUR 50,000.00. The competent administrative authority for the pursuit and prosecution of offenses of this type is the Chancellor of TU Dortmund University. In the case of multiple or other serious attempts at deception, the examinee can also be unenrolled, Section 63 (5) North Rhine-Westphalia Higher Education Act (*Hochschulgesetz, HG*).

The submission of a false affidavit will be punished with a prison sentence of up to three years or a fine.

As may be necessary, TU Dortmund University will make use of electronic plagiarism-prevention tools (e.g. the "turnitin" service) in order to monitor violations during the examination procedures.

I have taken note of the above official notification:*

Mannheim, 20.03.2024

Ort, Datum
(place, date)

Haritha Thiyaagu

Unterschrift
(signature)

***Please be aware that solely the German version of the affidavit ("Eidesstattliche Versicherung") for the Bachelor's/ Master's thesis is the official and legally binding version.**