CSE514 – Data Mining

Assignment 2

Name: ***Jinheng Zhang (510492)***
Yuxiao Wang (509794)

Date: ***04/01/2023***

Introduction

1. Description of the Problem and the impact of solving it

This question gives us 26 English letters based on 20 different written fonts, and they are converted into a dataset with 16 numeric attributes. Our task is to classify them accurately and determine which letter exactly belongs to provided handwritten mention font letter by implementing a classification algorithm. We can identify the correct letter in most of the handwritten fonts with different writing habits by modeling 26 letters in 20 different fonts the model. In this project, we will work with the Letter Recognition dataset from the UCI Machine Learning Repository. The dataset contains 20,000 samples of 26 capital letters of the English alphabet. The goal of this project is to perform binary and multi-class classification on the dataset using various machine learning models and dimension reduction techniques.

The practical impact of solving this problem is that it can be applied to various fields such as Optical Character Recognition, where the system is designed to recognize printed or handwritten characters automatically. It can also be applied in the field of computer vision for automatic reading of documents, sorting mail, and many more.

2. Motivation for multiple classifiers

Using multiple classifiers can help to improve the accuracy of predictions and provide more robust models. However, selecting the best classifier for a given task depends on various factors.

Some factors that should be considered in determining a classifier as the "best" include:

- **For model accuracy and the tradeoff between model generalization ability and fit:**
    - To solve this problem better, the first come into our consideration is the accuracy of the model. Thus, we should select models with higher accuracy. If a classifier has a great performance with a high correct rate on the training samples, it is most likely that has a better performance of the training dataset. However, we should notice a classifier cannot be considered a good model if it has a large bias. And it does not necessary to get the food result on the test data(overfitting). In conclusion, the tradeoff between model generalization ability and fitting is equally essential.
- **For the complexity of the classification function and the size of the training data:**
    - It really depends on the complexity of the classification problem. There are two cases: If is a simple classification problem, a classifier with a strong fitting ability and weak generalization ability is able to obtain a small portion of the training dataset. Otherwise, when it faces a complex classification problem, then the classifier learning requires numbers of training data and a learning algorithm with strong generalization

ability. In a word, the standard for a good classifier is if it will be able to automatically adjust the balance between the fitting power and generalization power based on the complexity of the problem and the size of the training set.

- **For the homogeneity of the input feature vectors and their relationship:**
  - Several classifiers require that the input feature's datatype should be numeric and normalized to a similar range. The rest of them required each feature vector should be independent of the other. Thus, a good classifier needs to fit the input feature vectors, which means that a good classifier should be selected based on the types of feature vector and their relationship.
- **Interpretability:**
  - Interpretability is crucial for understanding how the classifier works and how it is making predictions. The more interpretable a model is, the easier it is to understand and explain its predictions.
- **Computational Complexity:**
  - The computational complexity of a classifier is also an important consideration, especially when dealing with large datasets. A classifier that is computationally efficient can save time and resources.
- **Robustness:**
  - The robustness of a classifier is its ability to handle noisy or incomplete data. A robust classifier can perform well even when the data is not perfect.
- **Scalability:**
  - A classifier that can scale to handle larger datasets or more complex tasks is advantageous.
- **Bias and Variance Trade-Off:**
  - A good classifier should balance bias and variance. A model with high bias will underfit the data, while a model with high variance will overfit the data.
- **Hyperparameter Tuning:**
  - The selection of optimal hyperparameters for a classifier can significantly impact its performance. A good classifier should have well-tuned hyperparameters.

These factors, among others, should be considered when selecting the best classifier for a particular task.

3. Motivation for dimension reduction (With the Analysis of Pros & Cons)

Dimension reduction is a critical step in machine learning, especially when dealing with high-dimensional datasets. The motivation for dimension reduction is to reduce the number of features in the dataset while retaining the most relevant information.

Some factors that should be considered in determining a dimension reduction method as "good" or "bad" include:

- **Accuracy:**
    - For accuracy, we can determine whether we have selected a good dimension reduction method by comparing the accuracy of the model before and after dimension reduction.
- **Model Complexity:**
    - One of the main objectives of reducing dimension is reducing the complexity of the model. The better the dimension reduction method we selected, the better the task will perform. If we have a high correlation between data attributes, we should consider one of these features as the redundant date. And Filter methods will be a great choice in this case.
- **Type or different physical meaning (nature) of Data:**
    - In general, the type of data will require different dimension reduction methods. For data, we have count data, continuous data, categorical data, and distance data and etc. All of them required different dimension reduction methods. Linear methods such as PCA (principal component analysis) are often used to preserve the overall structure of the data. In contrast, the nonlinear methods perform better in expressing the local interactions of the data. Good dimension reduction methods need to match the nature of the processing data.
- **Preserving Relevant Information:**
    - The primary goal of dimension reduction is to preserve the most relevant information in the dataset. A good method should retain as much information as possible while reducing the number of features.
- **Computational Complexity:**
    - The computational complexity of the dimension reduction method is an essential consideration, especially when dealing with large datasets. A good method should be computationally efficient and scalable.
- **Interpretability:**
    - The interpretability of the reduced features is essential for understanding how the reduced dataset relates to the original dataset. A good method should provide interpretable features that are easy to understand and explain.
- **Robustness:**
    - The robustness of a dimension reduction method is its ability to handle noisy or incomplete data. A good method should be able to handle noisy or incomplete data without losing too much information.
- **Consistency:**
    - A good dimension reduction method should produce consistent results across different datasets or subsets of data.
- **Independence of the Classifier:**
    - A good dimension reduction method should be independent of the classifier used for the final prediction. It should not depend on the specific characteristics of the classifier.

There are several dimension reduction methods available, including simple quality filtering, filter methods, wrapper feature selection, embedded methods, and feature extraction. The best method depends on the specific characteristics of the dataset and the task at hand. A good method should balance the above factors and provide the best possible reduction in the number of features while retaining the most relevant information.

4. Description of chosen dimension reduction methods

- **VarianceThreshold ()** --- *Simple Quality Filtering*:
    - This method involves filtering out features based on a simple quality threshold, such as a minimum correlation coefficient or a minimum variance threshold.
    - It removes all the features whose variance does not meet a certain threshold. The threshold can be set using the threshold parameter, which is by default set to 0. For three different pairs and 26-classification, I setted different threshold values and tried to reduce the dimensions to 4. VarianceThreshold is useful for removing features with low variance, which often contain less information and are less useful for modeling.
- **SelectKBest (chi2, k=4)** --- *Filter Methods*:
    - These methods involve selecting features based on their statistical properties, such as their correlation with the target variable.
    - It selects the top k features with the highest scores from a set of features using the chi-squared statistical test. The chi2 parameter is used to specify the scoring function to be used. In this case, it is the chi-squared test. The second value of k is 4, but it can be changed to any positive integer.
- **RFE (estimator=RandomForestClassifier ())** --- *Wrapper Feature Selection*:
    - This method involves selecting features based on their performance in a specific classifier.
    - It selects the top n features with the highest importance scores, based on the recursive feature elimination (RFE) algorithm. The estimator parameter is used to specify the machine learning algorithm to be used for feature importance ranking. In this case, it is the RandomForestClassifier. The 'n_features_to_select' parameter is used to specify the number of features to be selected.
- **SelectFromModel (estimator-LinearSVC ())** --- *Embedded Methods*:
    - These methods involve selecting features based on their importance to the specific classifier used.
    - It selects the top k features with the highest scores from a set of features based on the importance scores provided by a machine learning algorithm. The estimator parameter is used to specify the machine learning algorithm to be used for feature importance ranking. In this case, it is the LinearSVC. The 'max_features' parameter is used to specify the maximum number of features to be selected.
- **PCA ()** --- *Feature Extraction*:
    - This method involves creating new features based on the existing features.
    - It reduces the number of dimensions in a dataset by projecting it onto a lower-dimensional space. The 'n_components' parameter is used to specify the number of principal components to be retained after the dimensionality reduction. In this case, it is set to 4. PCA is often used for reducing the complexity of a dataset while preserving the important features.

5. Speculate on the binary classification problems

For this project, we have simplified the classification problem to three binary classification problems:

- H vs K
- M vs Y
- R vs V

For the third problem, we chose to classify the letters R and V.

It is difficult to predict which pair of letters will be the easiest or hardest to classify without testing the classifiers. However, we can speculate that some letters may be easier to classify than others based on their visual similarities. For example, the letters H and K have similar shapes, so it may be more challenging to differentiate between them than it is to differentiate between the letters M and Y, which have more distinct shapes. Similarly, the letters R and V have some visual similarities, such as the diagonal lines, but they also have some differences, such as the curve in the letter R. Therefore, it may be challenging to classify these letters accurately, and we will need to test the classifiers to see how well they perform.

6. (Bonus Points) Advantages/disadvantages of using a multi-class classifier

Using a multi-class classifier instead of a set of binary classifiers has several advantages and disadvantages.

- Advantages:
    - Simplified workflow: A single multi-class classifier can be trained on all the classes simultaneously, reducing the complexity of the workflow.
    - Improved computational efficiency: Training a single classifier is often faster than training multiple binary classifiers.
    - Consistent predictions: A multi-class classifier can ensure that predictions are consistent across all classes.
    - Potentially higher accuracy: A multi-class classifier can consider all the classes together, which can result in higher accuracy for some datasets.
- Disadvantages:
    - Lower interpretability: A multi-class classifier can be more complex and harder to interpret than multiple binary classifiers, making it harder to understand how the model is making predictions. Limited flexibility: Multi-class classifiers may not allow for fine-tuned control over the classification process for each class, which can result in suboptimal performance for some classes.
    - Imbalanced datasets: Multi-class classifiers may not perform as well when the dataset is imbalanced, with significantly more examples for some classes than others.

- Higher risk of overfitting: A multi-class classifier can be more prone to overfitting, especially when the number of classes is high.

In summary, the choice between using a multi-class classifier or multiple binary classifiers depends on the specific needs of the problem at hand, the available data, and the resources available for the project. If interpretability, fine-tuned control, or performance on imbalanced datasets are critical, then multiple binary classifiers may be the best option. However, if computational efficiency and consistent predictions are more important, a multi-class classifier may be preferred.

Result

1. Description of the classifier:

1. **K-nearest neighbors**
    a. **Description:** K nearest neighbors is a machine learning algorithm used for both classification and regression tasks. The algorithm usually works by finding the k-closest training example in the feature space to give a test example and using the label to predict the label of the best example. The value of k is a hyperparameter that can be chosen based on the problem at hand. A small value of k will result in a more flexible model with high variance and lower bias.
    b. **Pros:** KNN is easier to understand, and t is a non-parametric algorithm, it doesn't need to make any assumptions about the underlying distribution of data. KNN also does not require a training phase, it can be used for both supervised and unsupervised learning tasks. It generally ignores the outliers and noise data. Moreover, KNN can be used for both classification and regression tasks, and it is easy to be updated when the data changes.
    c. **Cons:** KNN is computationally expensive when it deals with large datasets or high-dimensional feature spaces, it will become extremely time-consuming. KNN also performance can be influenced by the curse of dimensionality, which make the model become less meaningful. KNN is also very sensitive to irrelevant features such as noisy data, that will lead the model to a bad performance.
2. **Decision Tree**
    a. **Description:** A decision Tree is a graphical representation of a decision-making process or a decision-making algorithm that use a tree as a model of decision and their possible consequences. It is usually used in operations research, management, and AI to help visualize and analyze complex decision problems.
    b. **Pros:** Decision trees are easy to understand and visualize. And it is also very flexible, it can be used in a wide range of tasks such as classification, regression, and clustering. And it can deal with different data types such as categorical and numerical data. A decision tree is also highly interpretable. It is worth saying that decision trees are efficient, they can be constructed rapidly when it faces bigger datasets. It can deal with noisy data and missing values without extensive data preprocessing.
    c. **Cons:** Decision trees can easily overfit the training data, which will lead to bad overperformance it is hard to handle the small difference in the training data. Decision trees also struggle with high-dimensional data, such as the number of possible splices growing exponentially with the number of features. And decision trees possibly miss valuable information when it deals with a larger dataset.
3. **Random Forest**
    a. **Description:** Random Forest combines multiple decision trees to create a robust and accurate predictive model. In a random forest, multi possessions are built on n different subsets of the training data, with different random selections of features used at each node。

b. **Pros**: Random Forest has high prediction accuracy and is less sensitive to outliers and noisy data than individual decision trees. The random forest can provide information on important features and improve interpretability. It also has a great performance in reducing overfitting.
c. **Cons:** Random Forest can be compositionally expensive, especially when it deals with a large dataset and a large number of features, it has limited interpretability. It can become less interpretable than individual decision trees although random forests provide information on feature importance. The random forest can be biased toward features with more levels and categories. Random forest is different to tune, and it may be very time-consuming at predicting new data, especially when it is facing a large dataset.

4. **SVM:**
   a. **Description:** A support vector machine is a two-class classification model. The basic model of it is an interval-maximizing linear classifier defined on a feature space, it also includes kernel trucks, which make it essentially a nonlinear classifier. The learning strategy of a support vector machine is interval maximization and the learning algorithm of it is an optimization algorithm for solving convex quadratic programming.
   b. **Pros:** SVM is effective in high-dimensional spaces, it usually performs well in high-dimensional spaces, which makes them ideal for problems with a large number of features. And SVM is also robust to overfitting, it has a regularization parameter that helps to avoid overfitting. SVM also handles non-linear data, and it is possible to separate non-linearly separable classes.
   c. **Cons:** SVM training time is long. When the SMO algorithm is used, the time complexity is high. It is also sensitive to the choice of the kernel because it is highly dependent on the choice of the kernel function. If we choose the wrong kernel, it will lead to poor performance. Moreover, it is also computationally expensive, and when it deals with a large dataset, will be extremely time-consuming. SVM also has difficulty in handling noisy data. It is very sensitive to the noise in the data, it may lead to overfitting or underfitting, which will get a low accuracy of the model. Lastly, VM is also performing limited when it applies to multi-class classification problems.
5. **MLP Classifier:**
   a. **Description:** MLP classifier is a type of artificial neural network that Is used for classification tasks in machine learning. It is a feedforward neural network that consists of multiple layers of perceptions, which are computational units that learn to transform input data into output data through a series of mathematical operations.
   b. **Pros:** MLP classifier can model complex non-linear relationships between input features and target variables. MLP also has high accuracy, especially of the high accuracy on a variety of input data types, such as categorical, continuous, and ordinal data. MLP also has the ability to robust the noisy value.
   c. **Cons:** MLP classifier requires a large dataset, especially for deep neural networks. It is also a time-consuming model, in which training time is especially long. MLP classifier is also easily overfitted to the training data when it deals with many

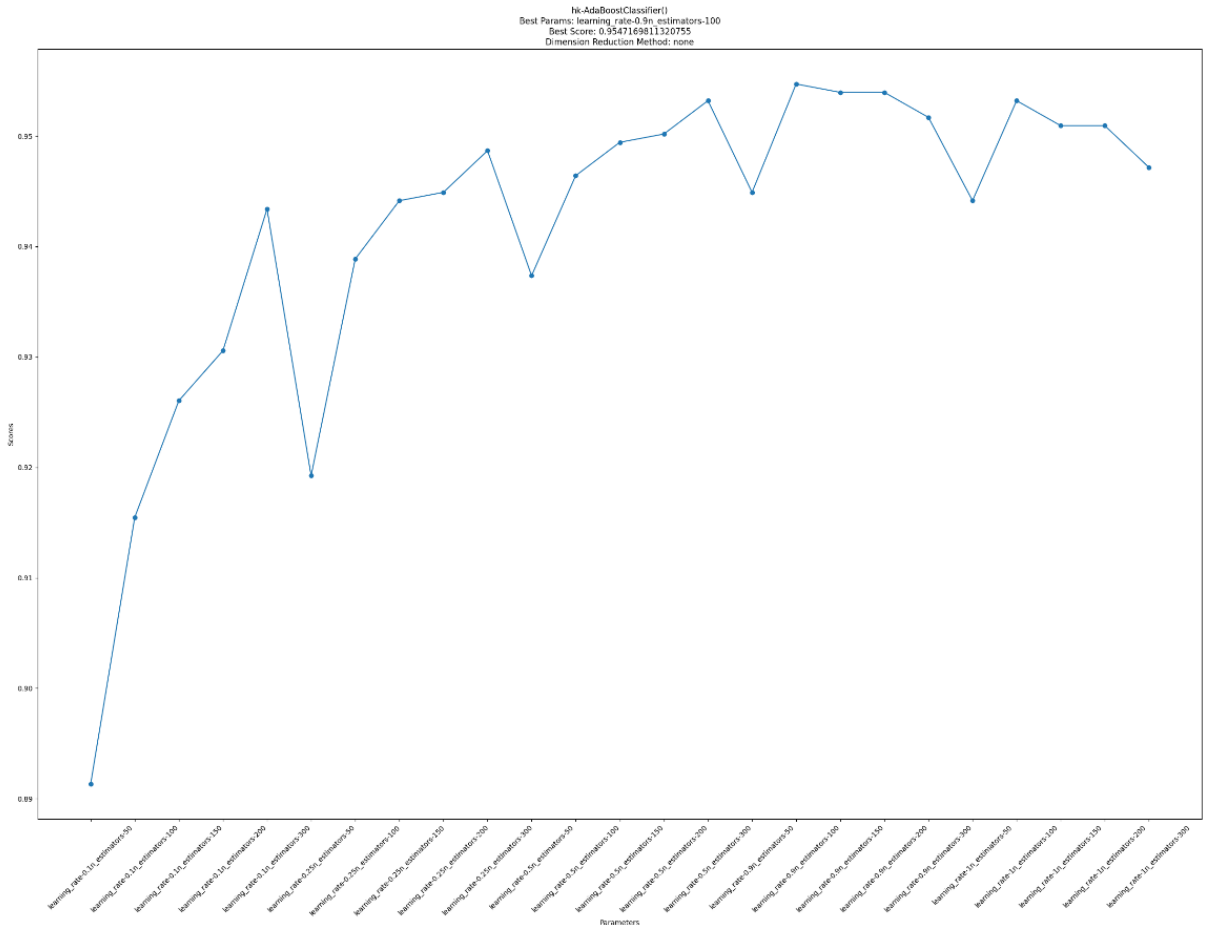
parameters. MLP classifier may also have several hyperparameters that need to be tuned, such as the number of hidden layers, the number of neurons per layer, and the learning rate. If we want to find the optimal values for this hyperparameter, it will be very time-consuming and computationally expensive.

6. **Adaboost:**
    a. **Description:** The AdaBoost algorithm combines a few weal classifiers to create a new stronger classifier. It works by assigning weight to each instance in the training data. After it, it will increase the weight of misclassified instances and decrease the weight of correctly classified instances. And it will keep iterating for the required loop, and each iteration will focus on the misclassified instance from the previous iteration. After finishing all of them, the final classifier will be created.
    b. **Pros:** Adaboost has high accuracy compared with the other machine learning models. It can utilize a wide range of classification algorithms such as decision trees, support vector machines, and neural networks. Moreover, it is less prone to overfitting than other algorithms, it performs well in generalizing to new data. Adaboost can also automatically select the most important feature for the model and reduce manual feature selection.
    c. **Cons:** Adaboost is very sensitive to outliers. Outliers can have a significant influence on performances. Adaboost also performs badly when the data is imbalanced. It is also computationally expensive. When it faces large datasets and complex models, it will be greatly time-consuming, Adaboost mat performs poorly when data contains a lot of noise which can cause the model to overfit. Adaboost also be influenced by the bias model.

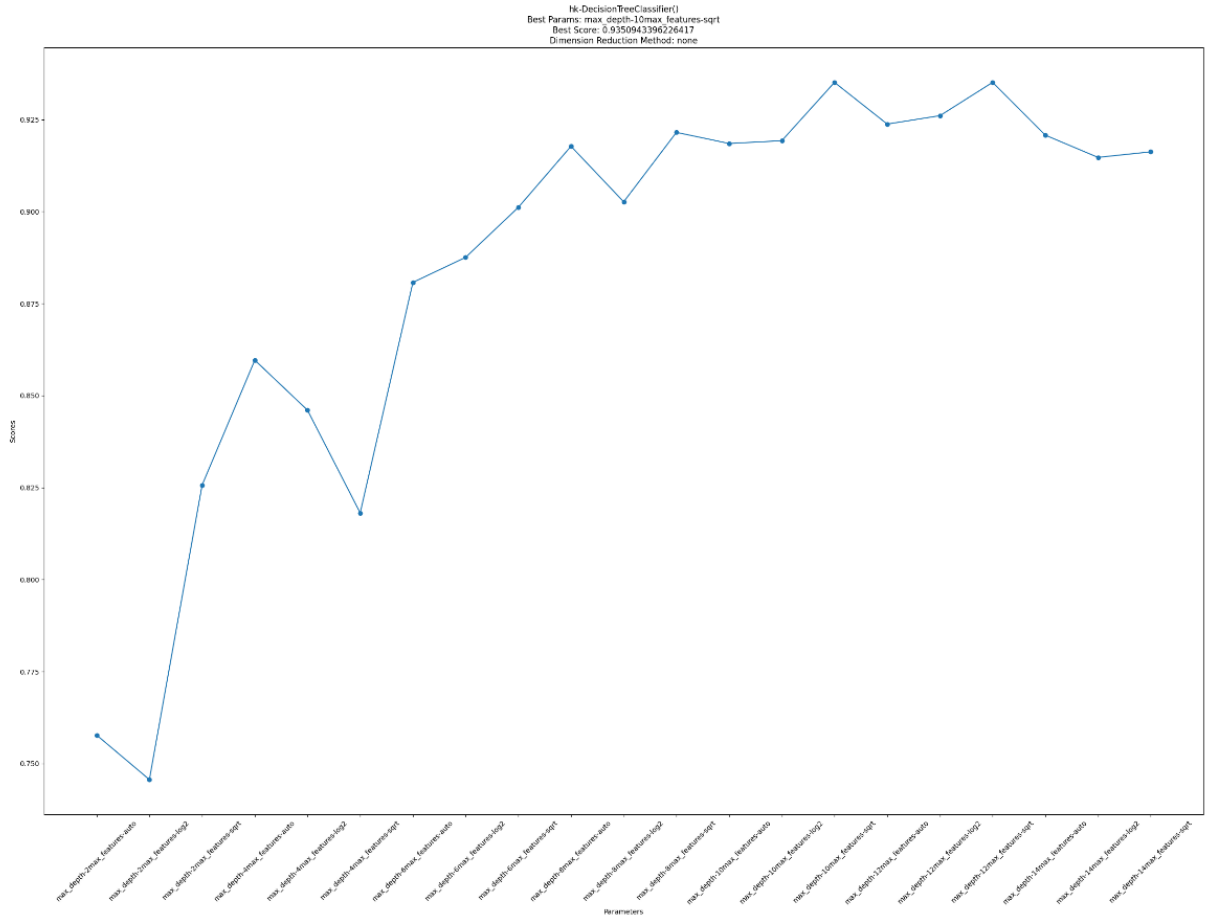
2. Graph

2.1 Graph the cross-validation results *(without dimension reduction)*

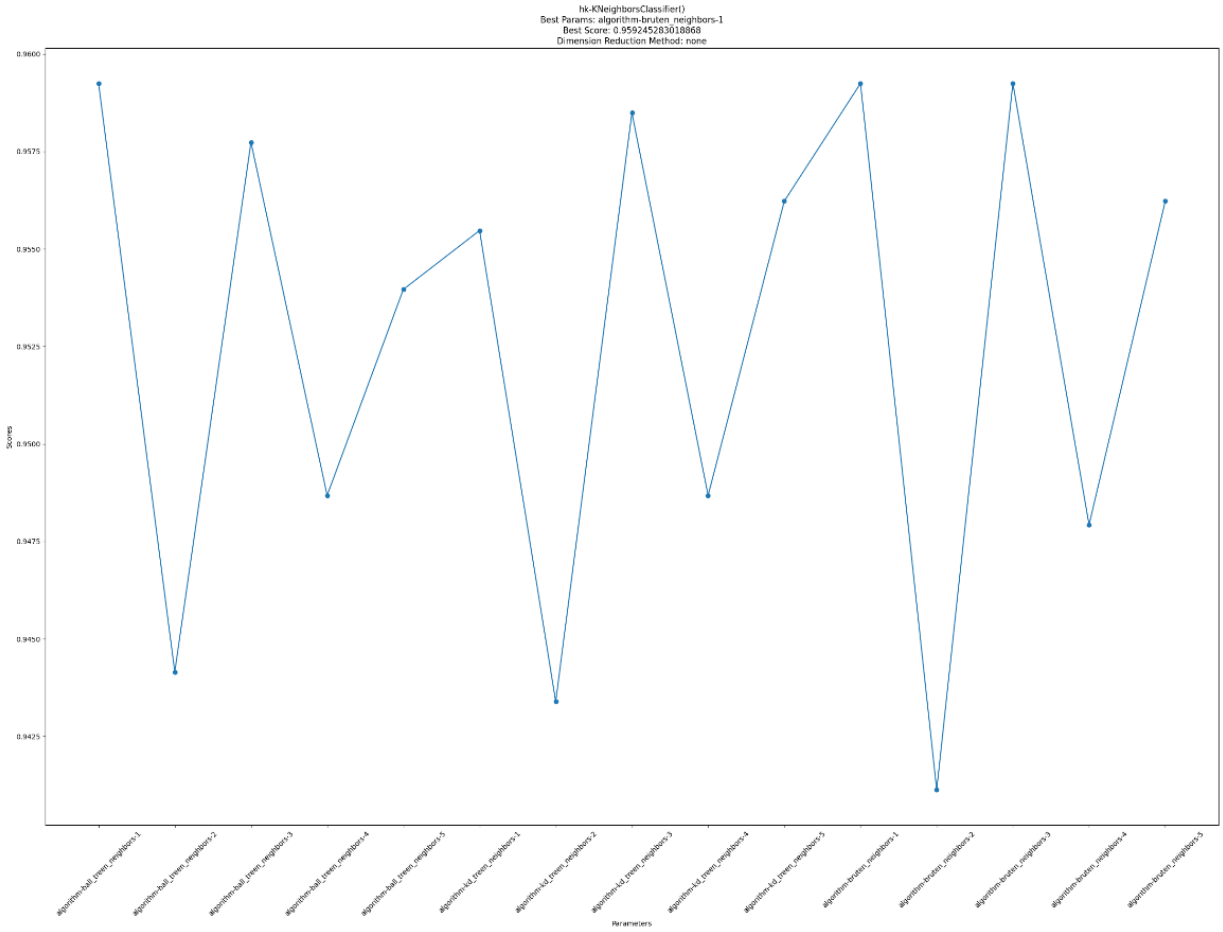
HK- AdaBoost Classifier



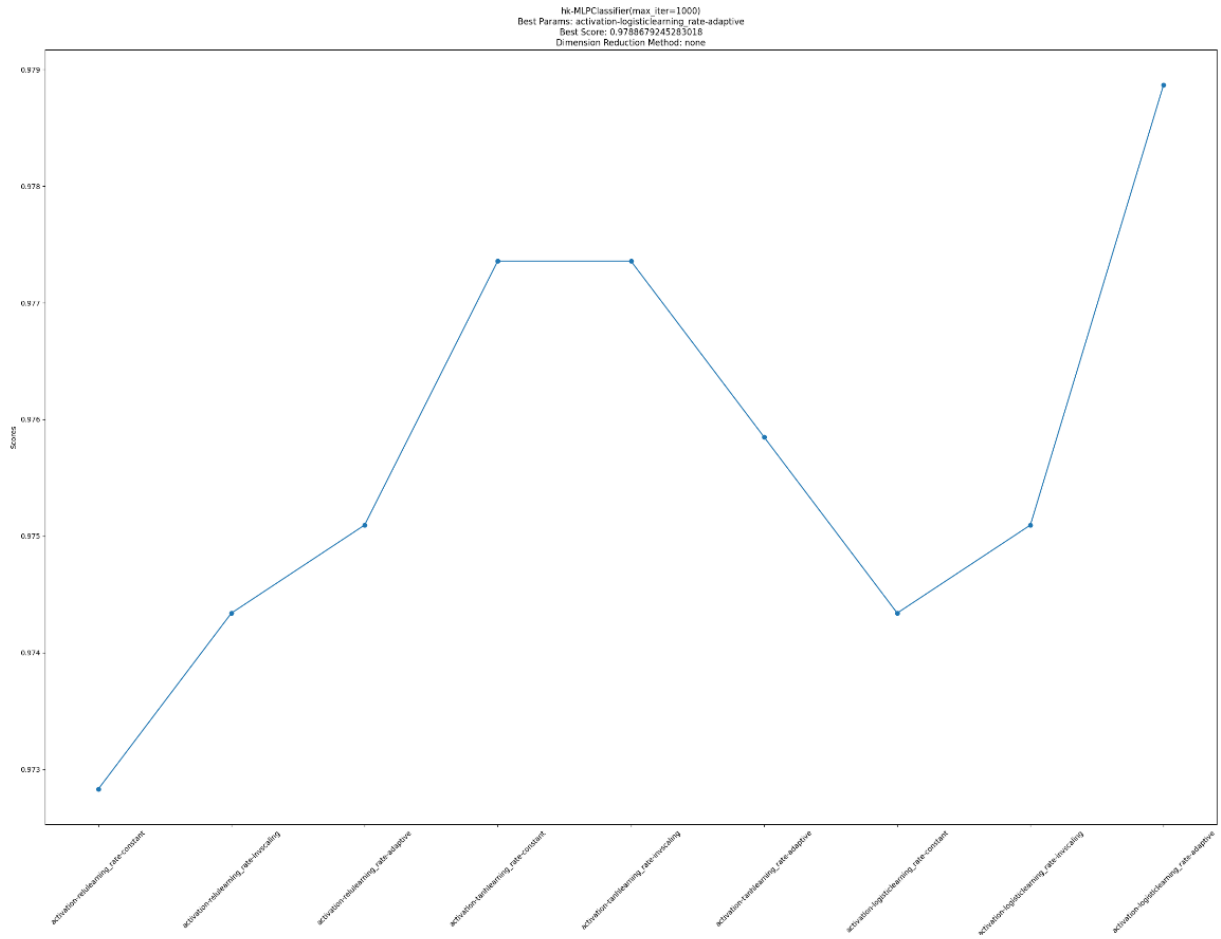
HK – Decision Tree Classifier



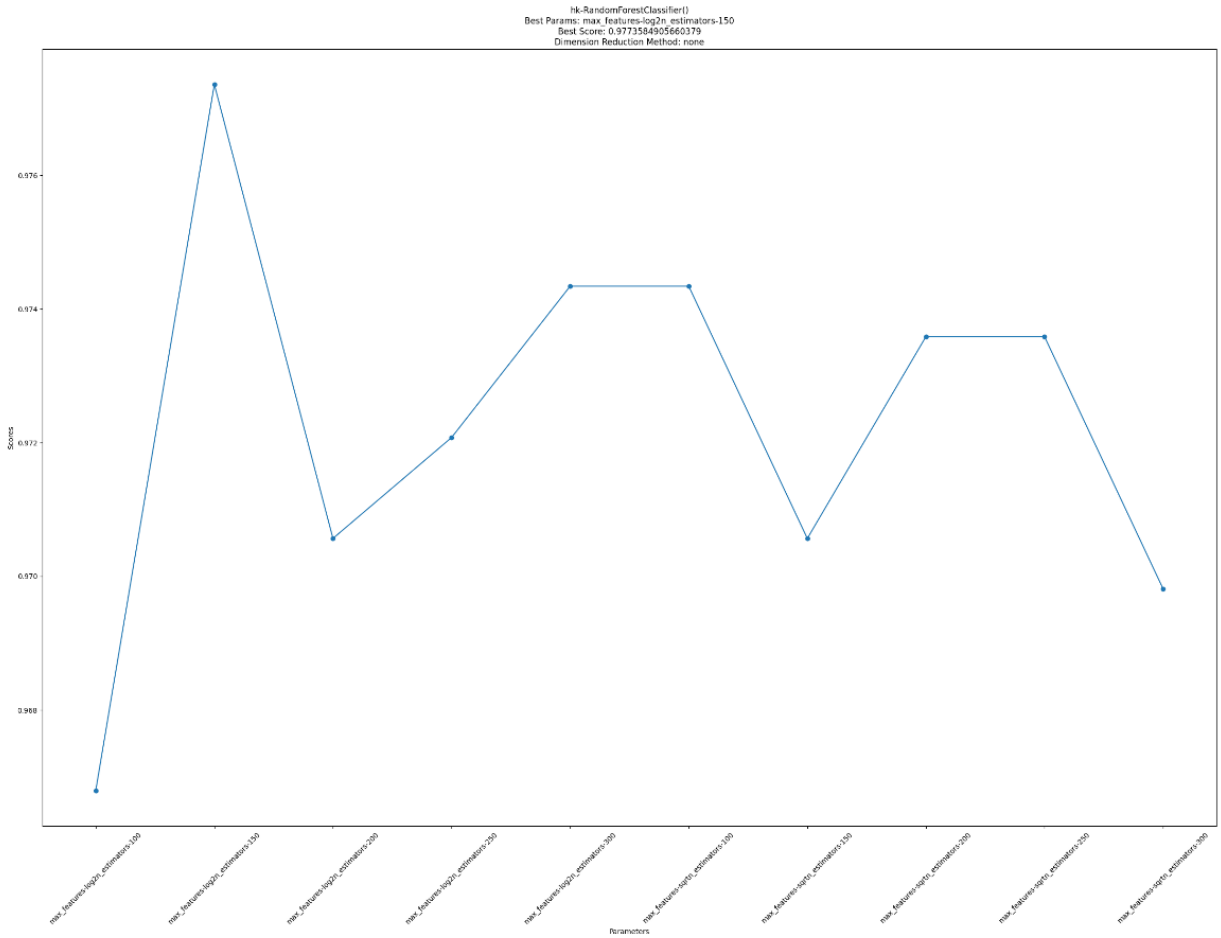
HK – K neighbors Classifier



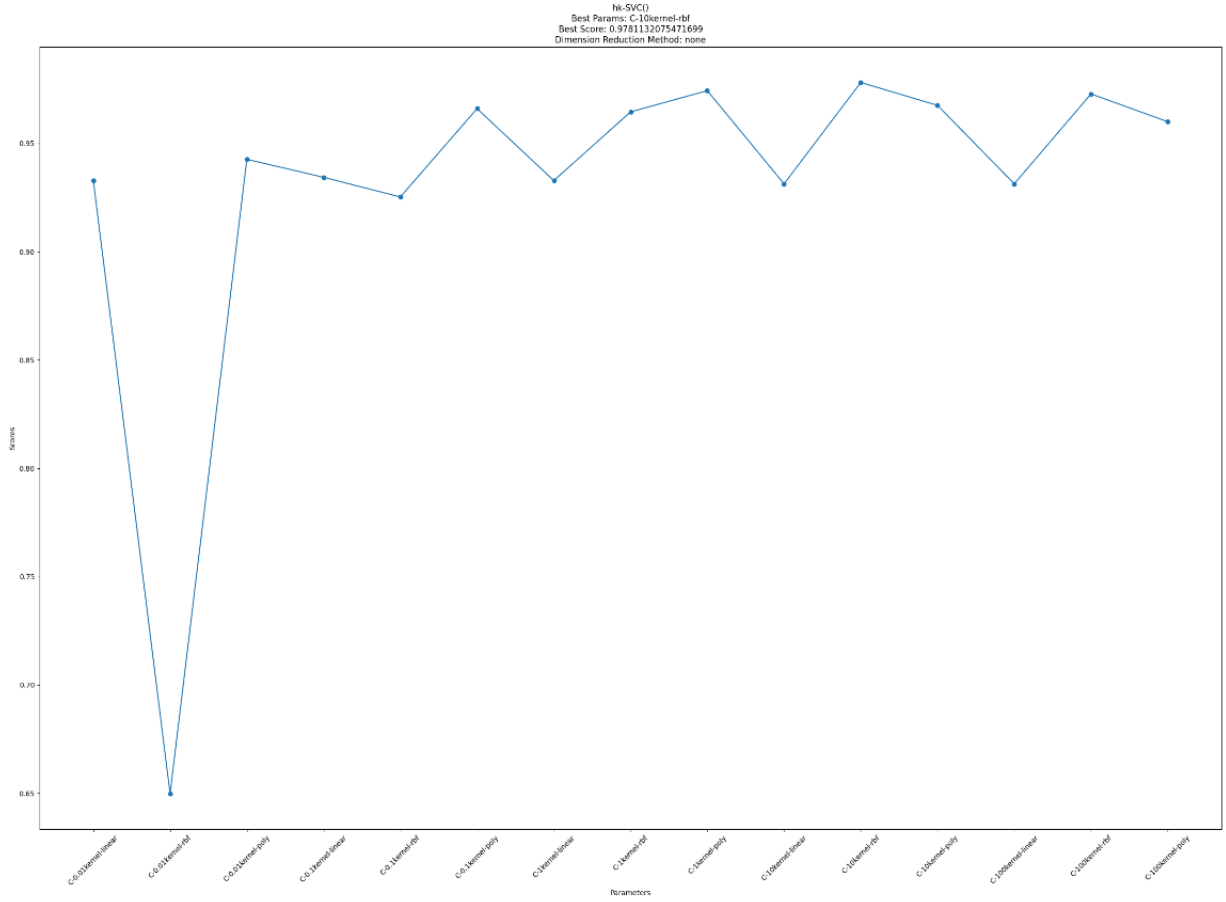
HK – MLP Classifier



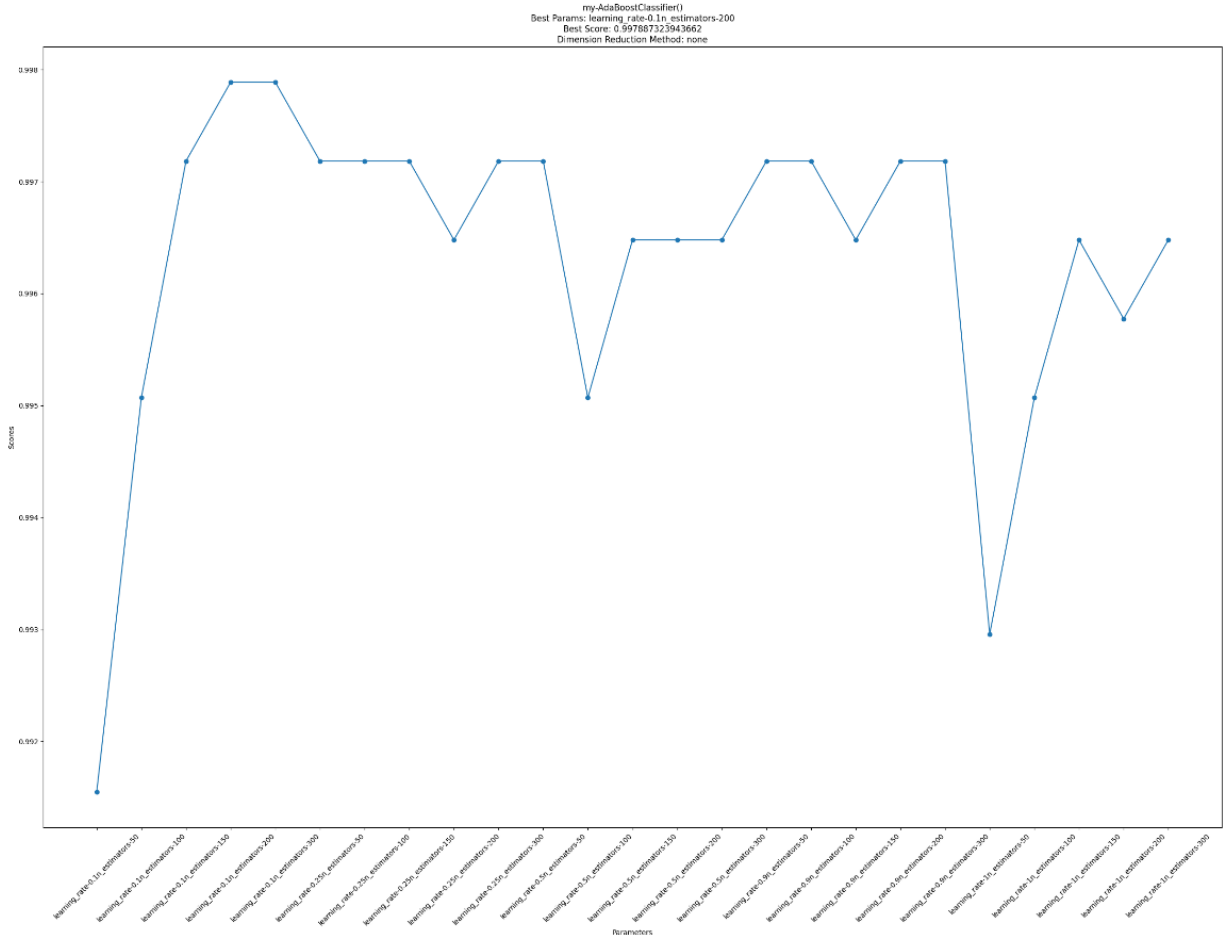
HK – Random Forest Classifier



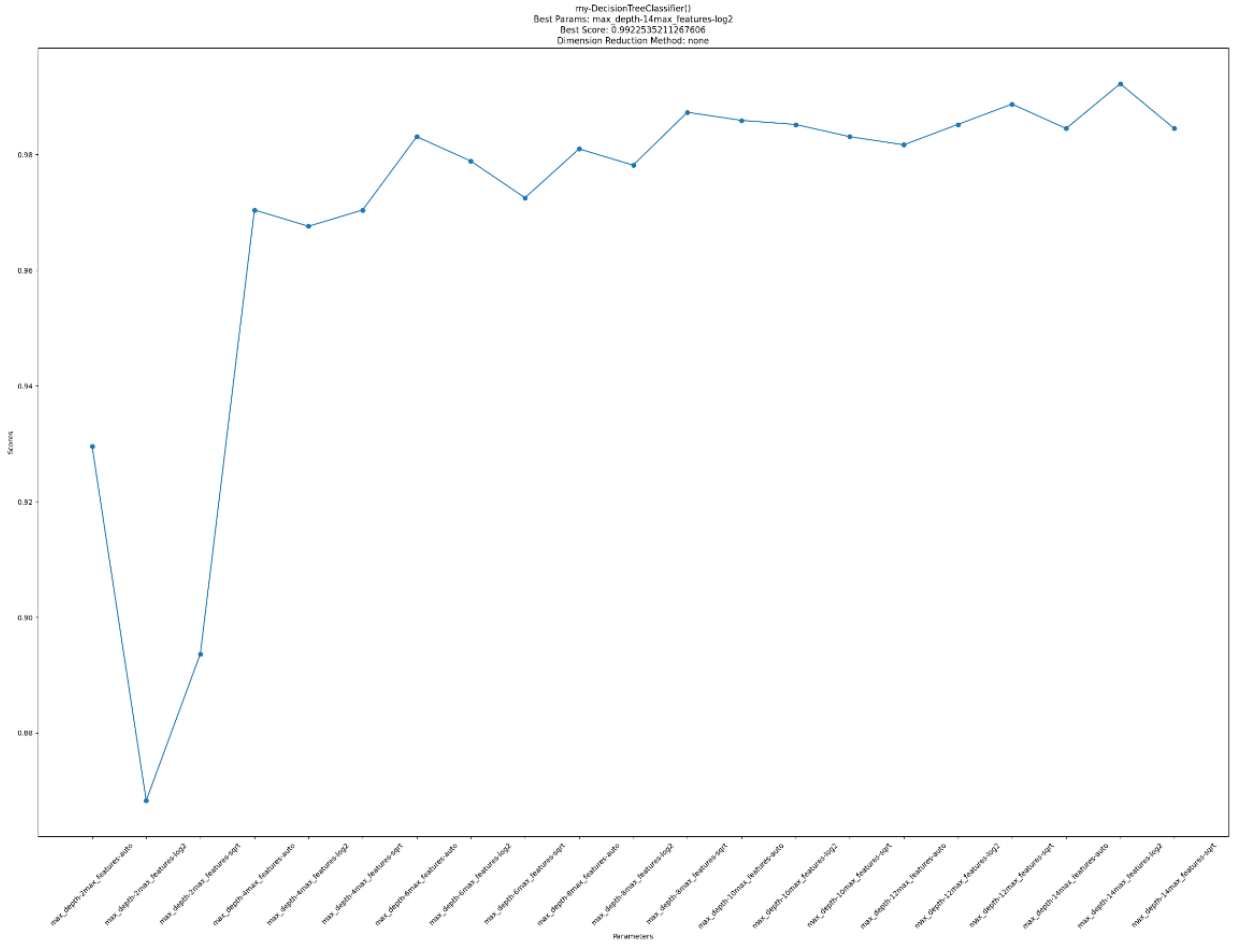
HK- SVC


hk-SVC()
Best Params: C-10kernel-rbf
Best Score: 0.9781132075471699
Dimension Reduction Method: none
Scores
0.95
0.90
0.85
0.80
0.75
0.70
0.65
C-0.01kernel-linear
C-0.01kernel-rbf
C-0.01kernel-poly
C-0.1kernel-linear
C-0.1kernel-rbf
C-0.1kernel-poly
C-1kernel-linear
C-1kernel-rbf
C-1kernel-poly
C-10kernel-linear
C-10kernel-rbf
C-10kernel-poly
C-100kernel-linear
C-100kernel-rbf
C-100kernel-poly
Parameters

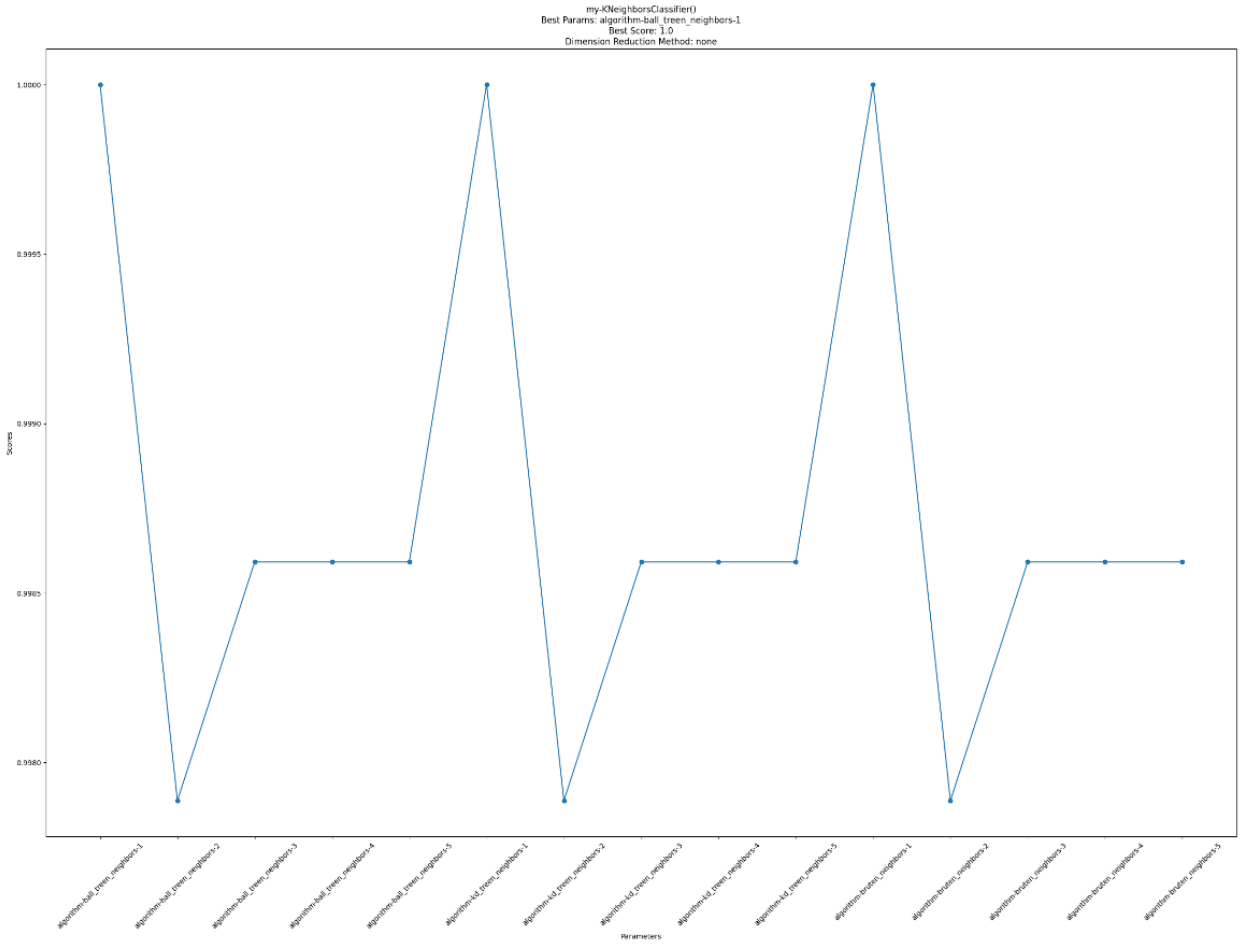
MY – AdaBoost Classifier



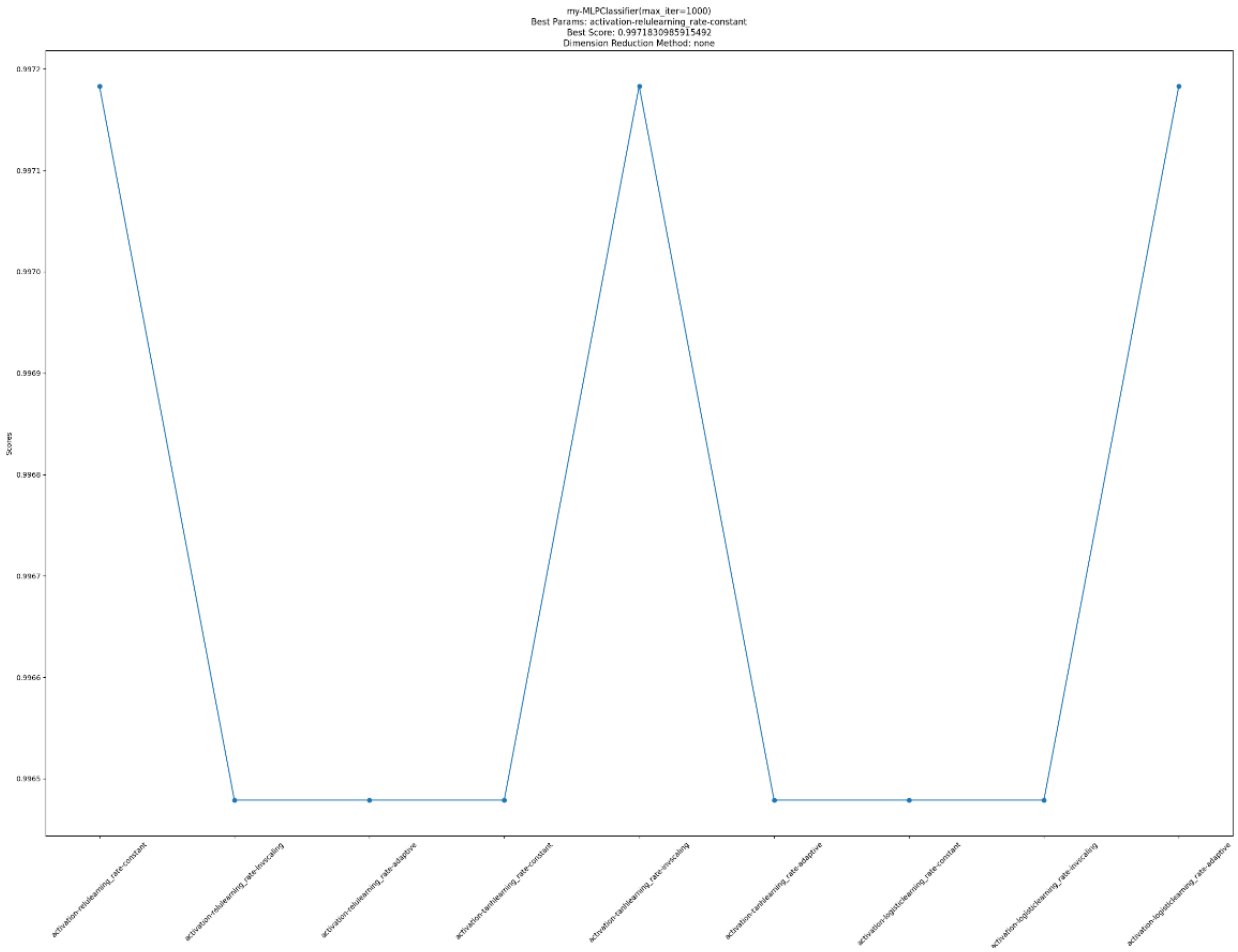
MY – Decision Tree Classifier



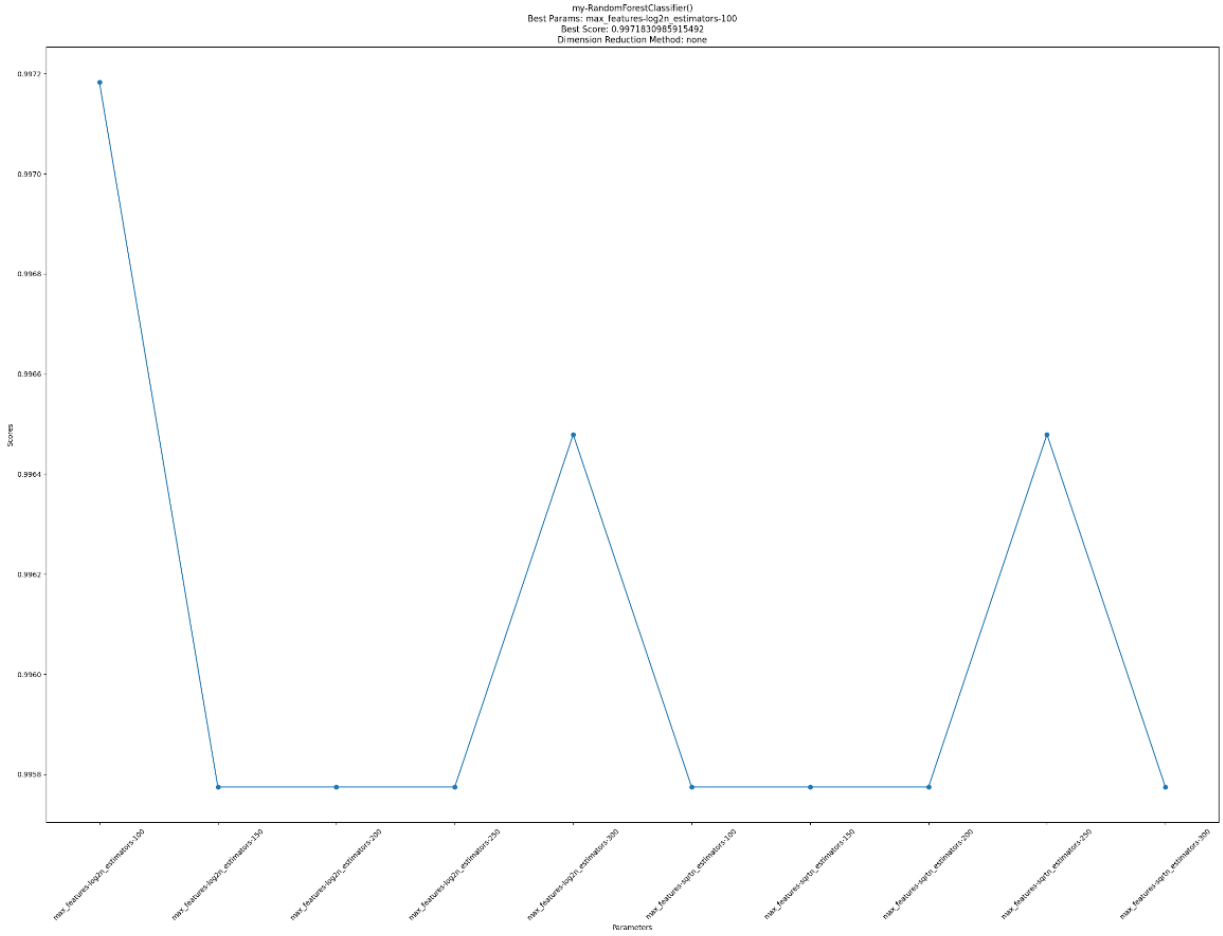
MY – K Neighbors Classifier



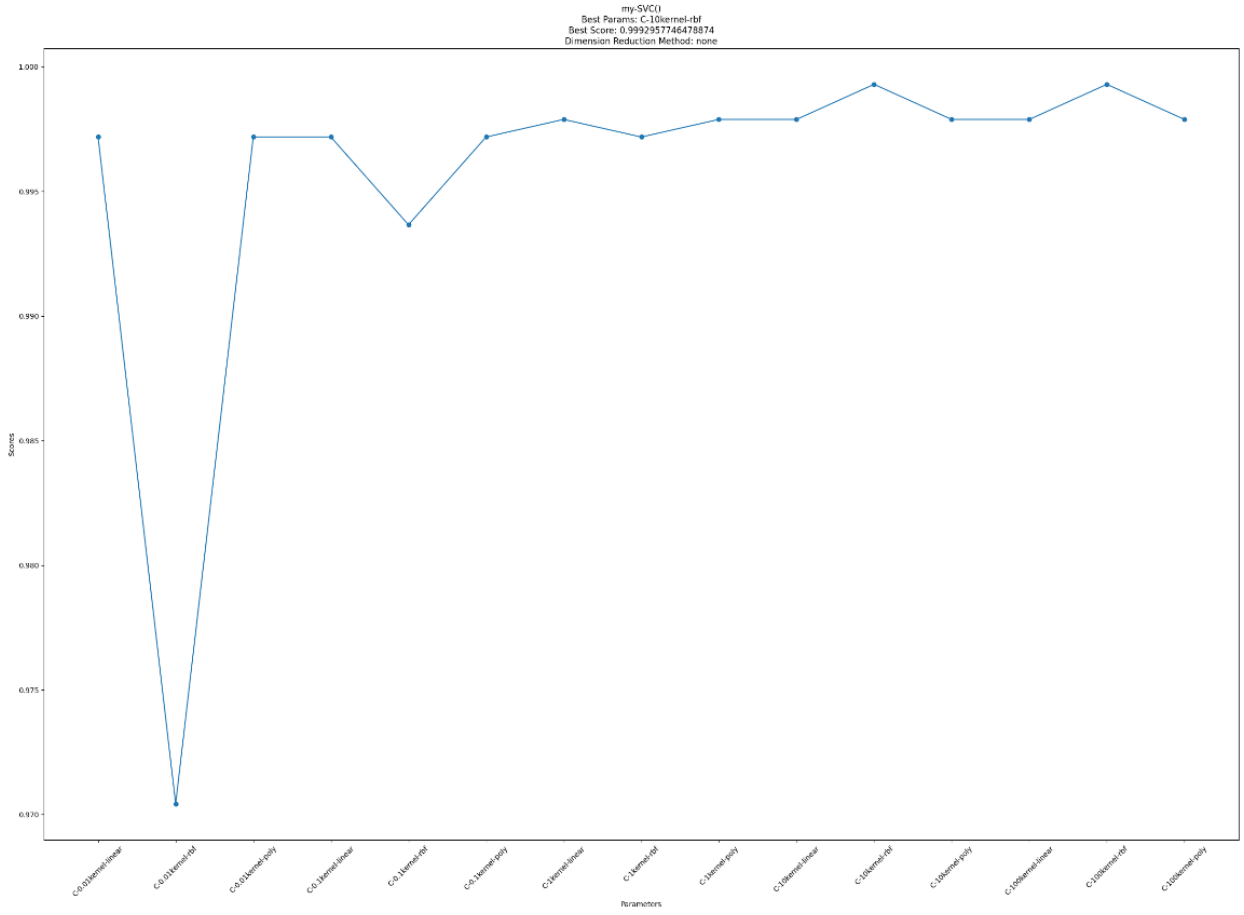
MY – MLP Classifier



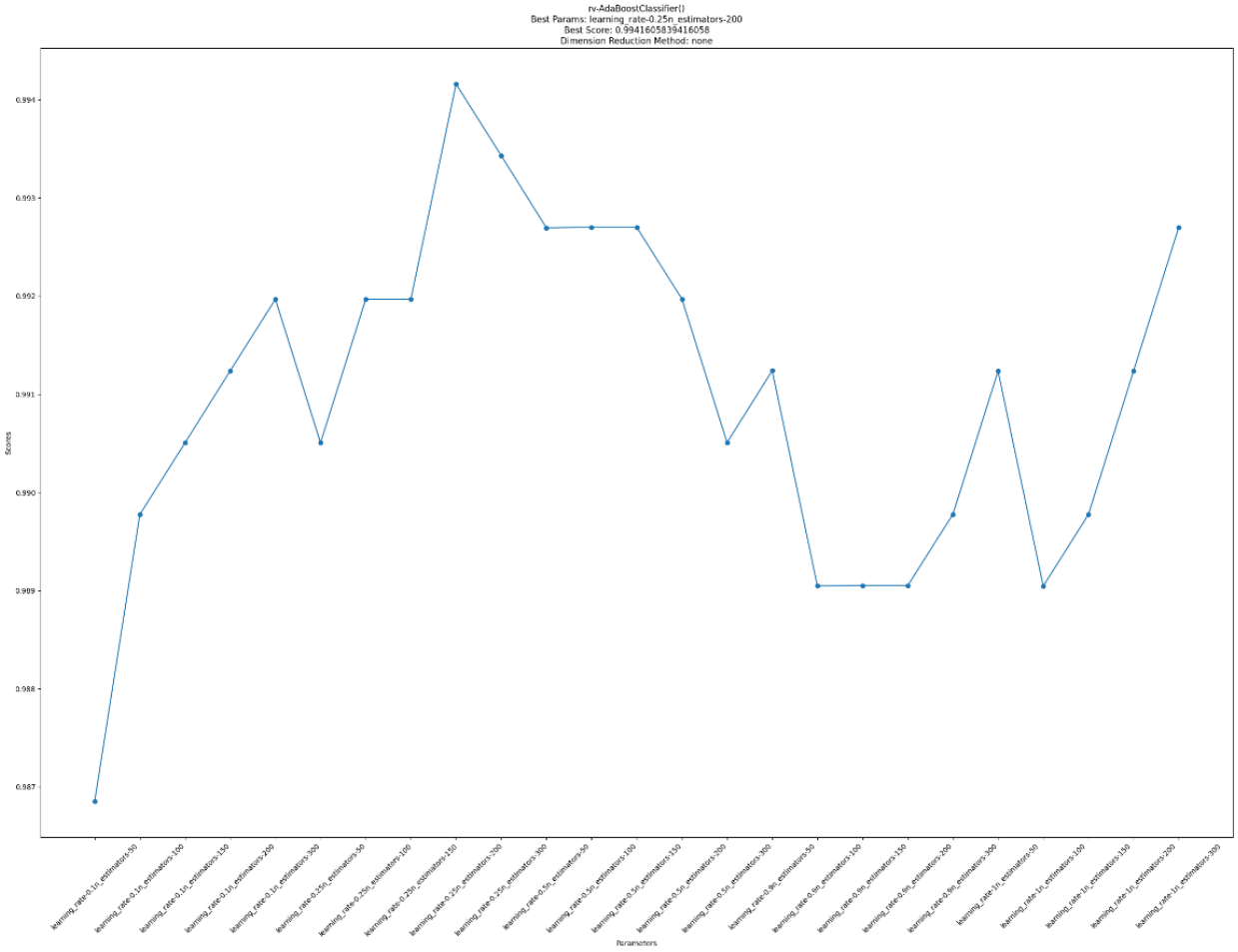
MY – Random Forest Classifier



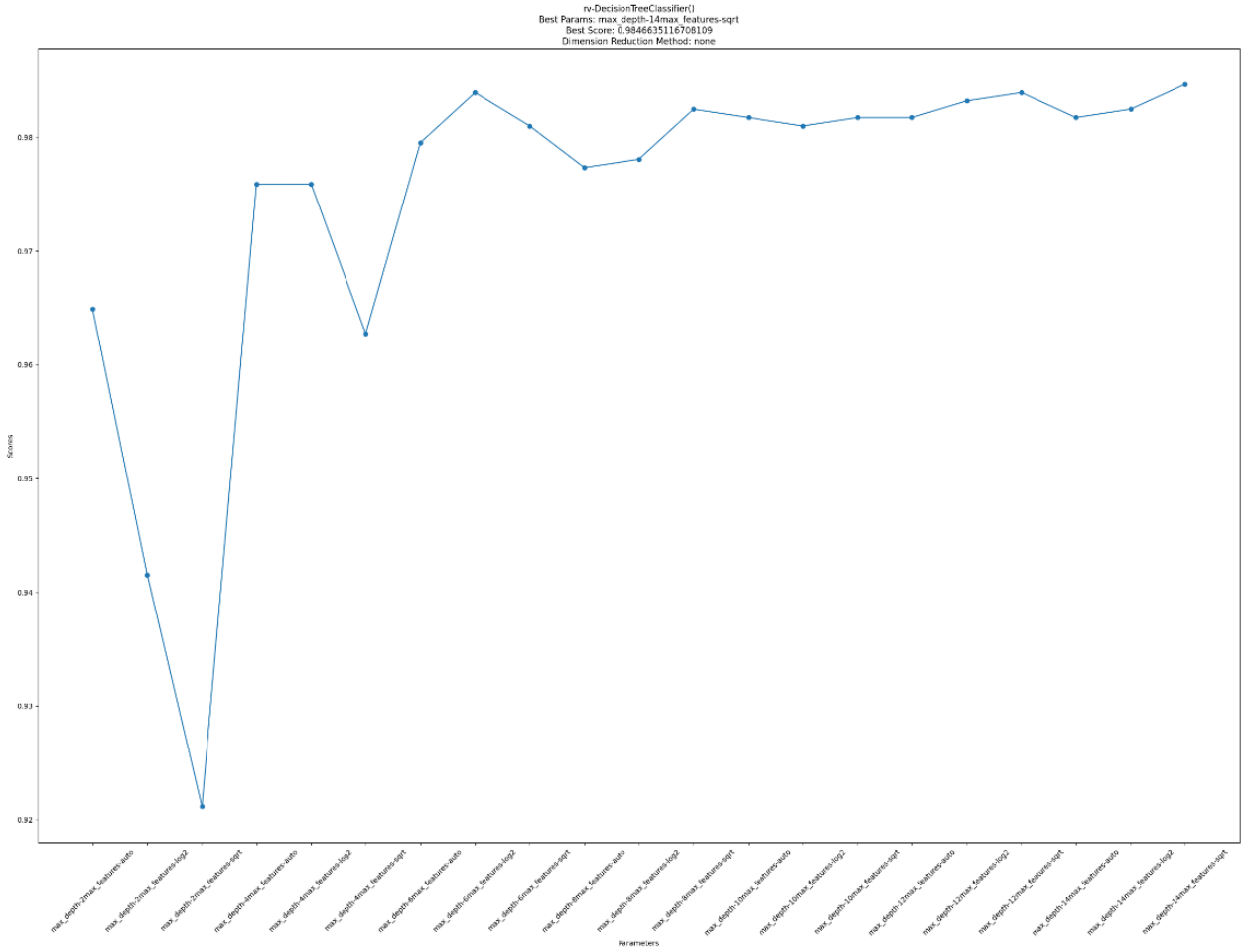
MY- SVC


my-SVC()
Best Params: C-10kernel-rbf
Best Score: 0.9992957746478874
Dimension Reduction Method: none
Scores
1.000
0.995
0.990
0.985
0.980
0.975
0.970
C-0.01kernel-linear
C-0.01kernel-rbf
C-0.01kernel-poly
C-0.1kernel-linear
C-0.1kernel-rbf
C-0.1kernel-poly
C-1kernel-linear
C-1kernel-rbf
C-1kernel-poly
C-10kernel-linear
C-10kernel-rbf
C-10kernel-poly
C-100kernel-linear
C-100kernel-rbf
C-100kernel-poly
Parameters

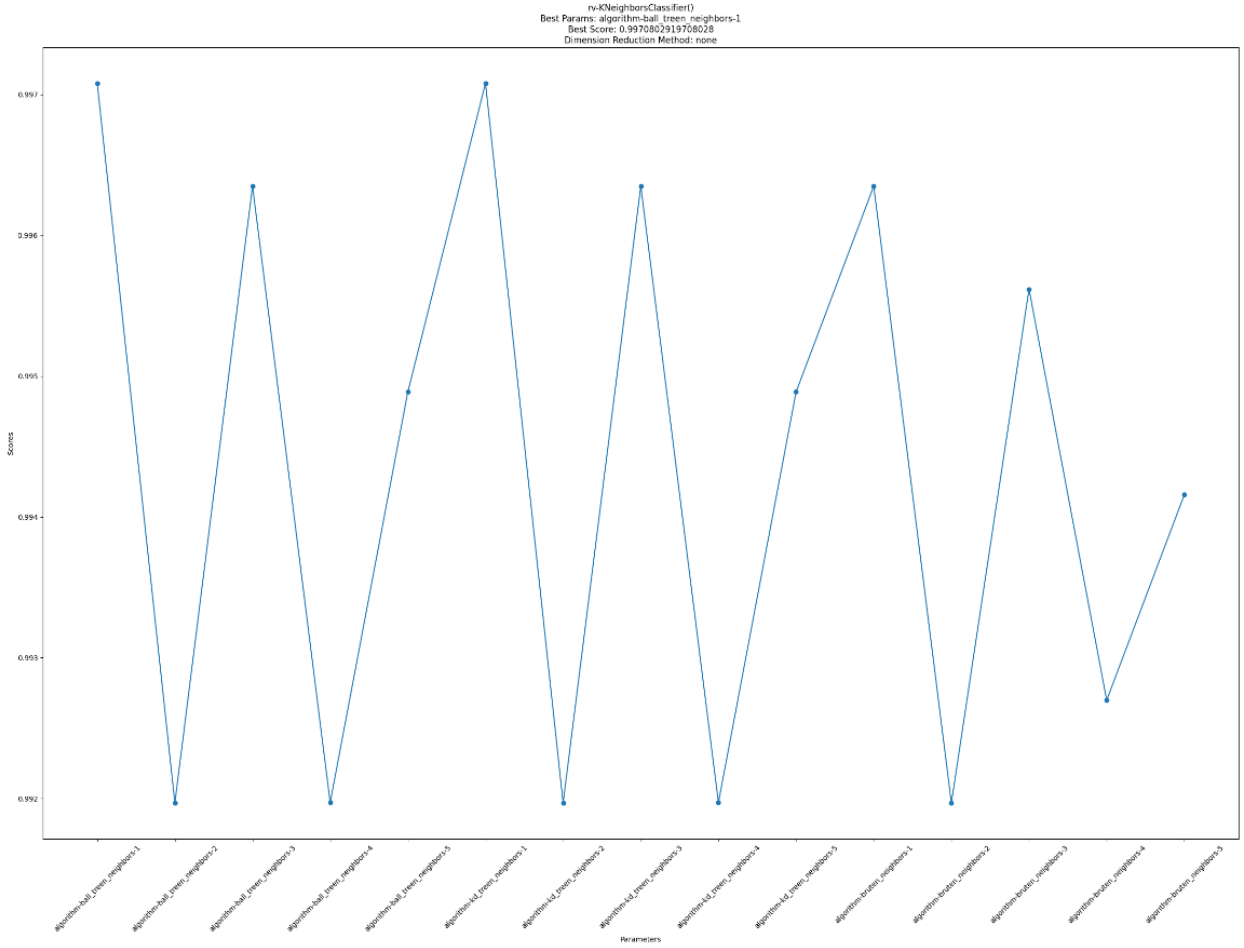
RV – AdaBoost Classifier



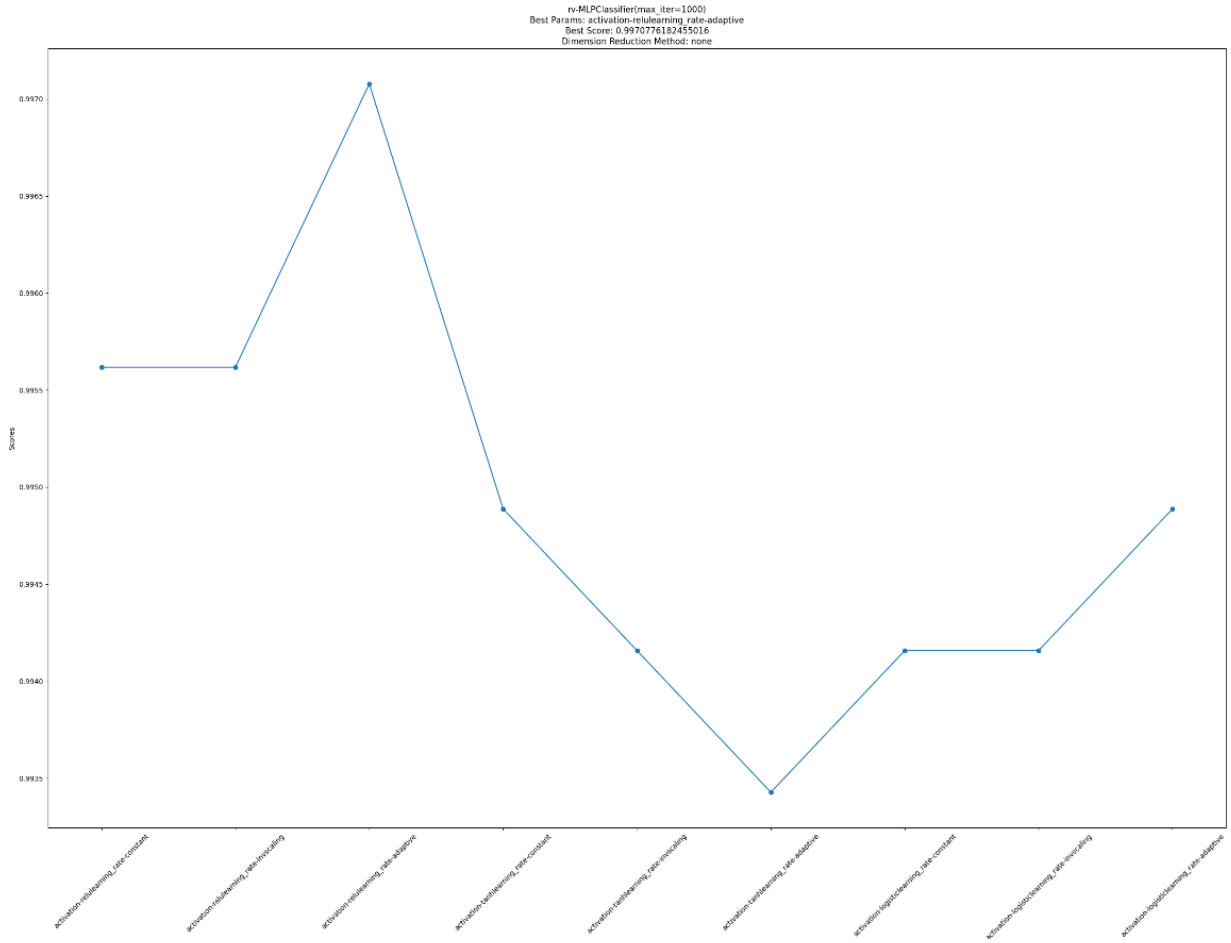
RV – Decision Tree Classifier



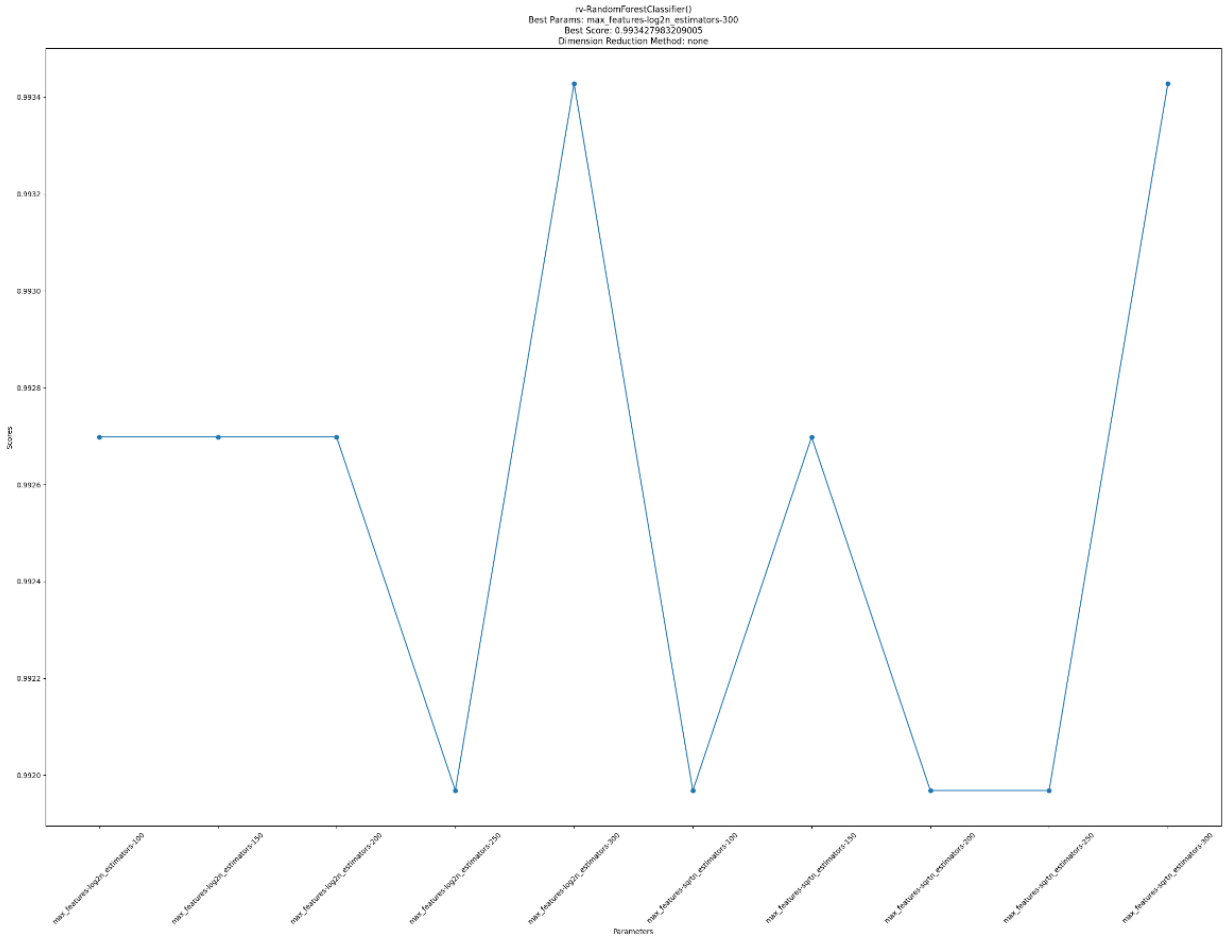
RV – K Neighbors Classifier



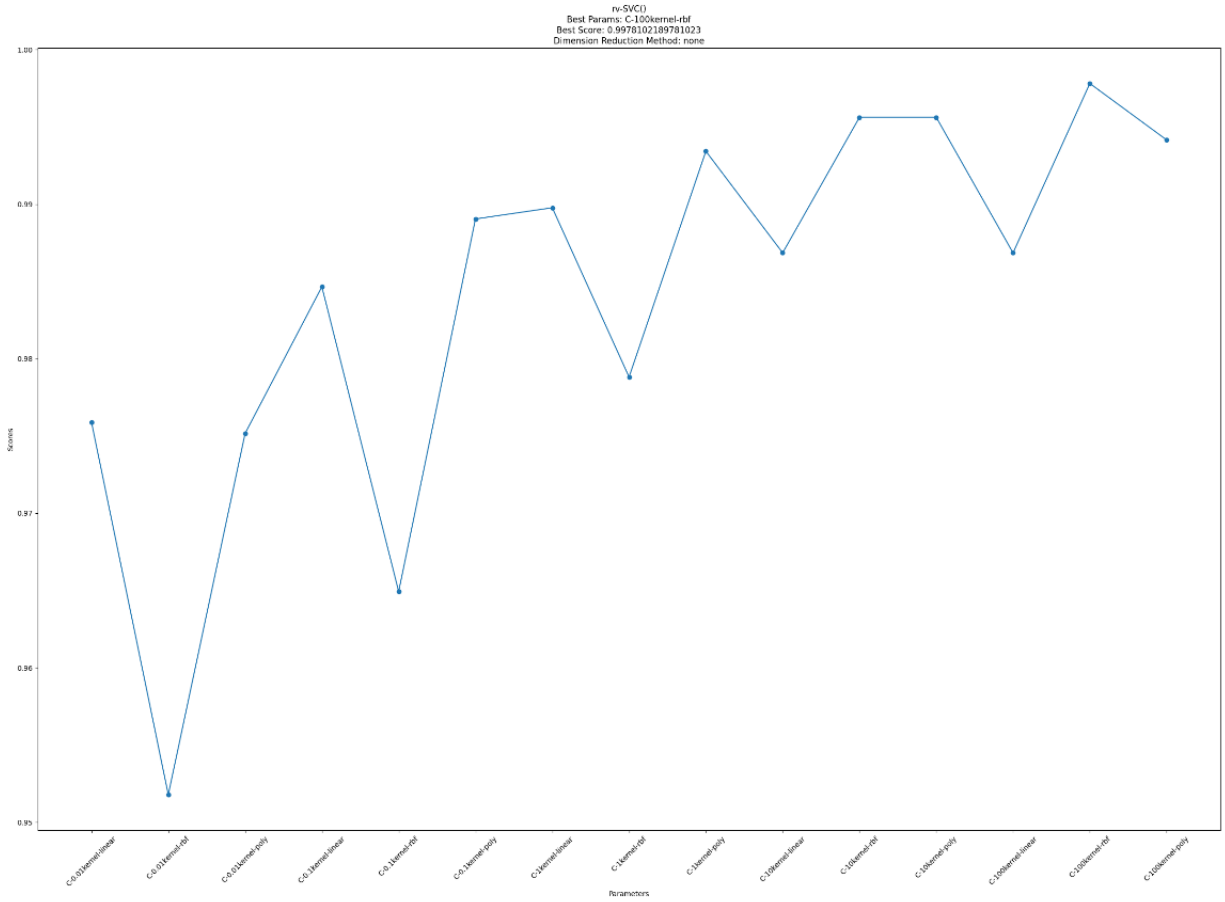
RV – MLP Classifier



RV – Random Forest Classifier

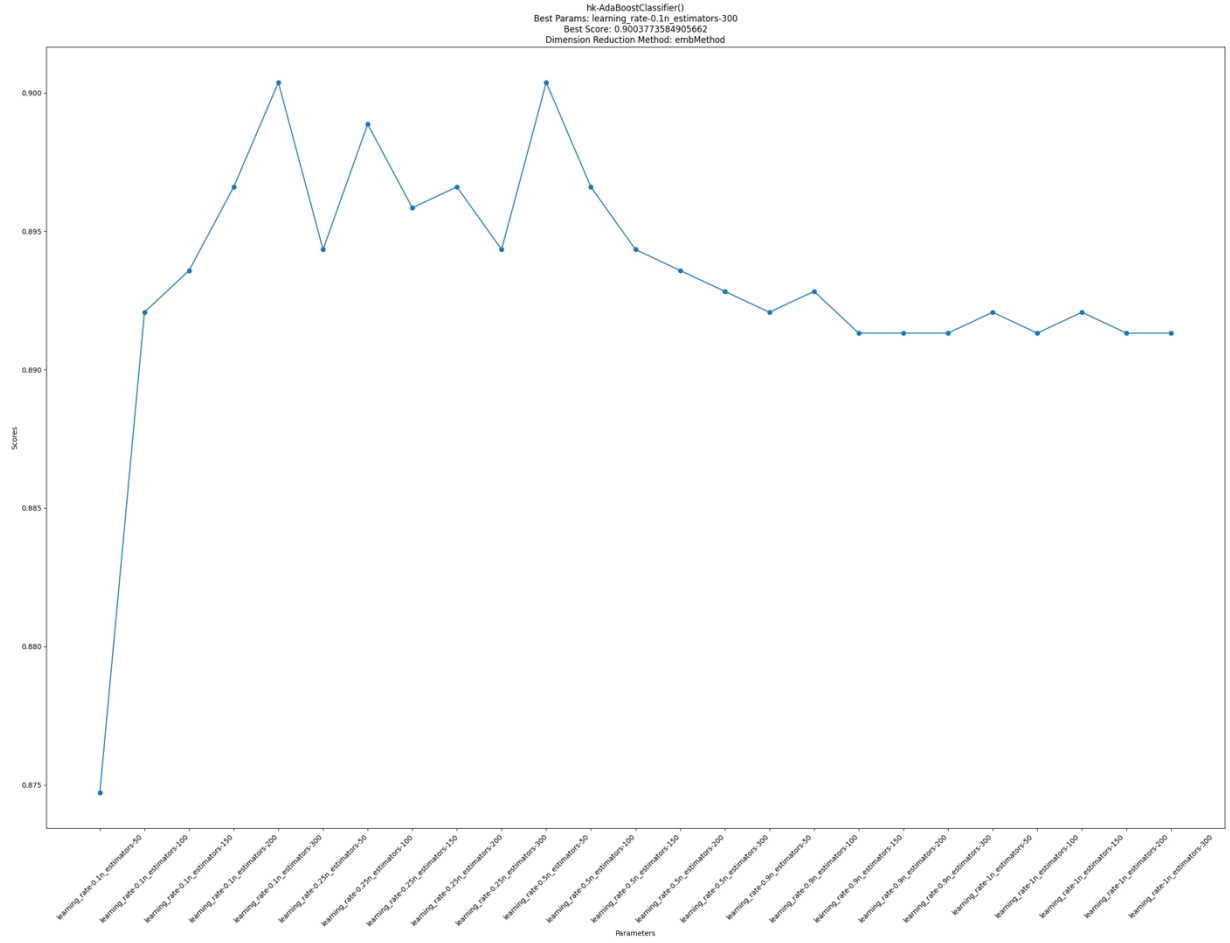


RV- SVC

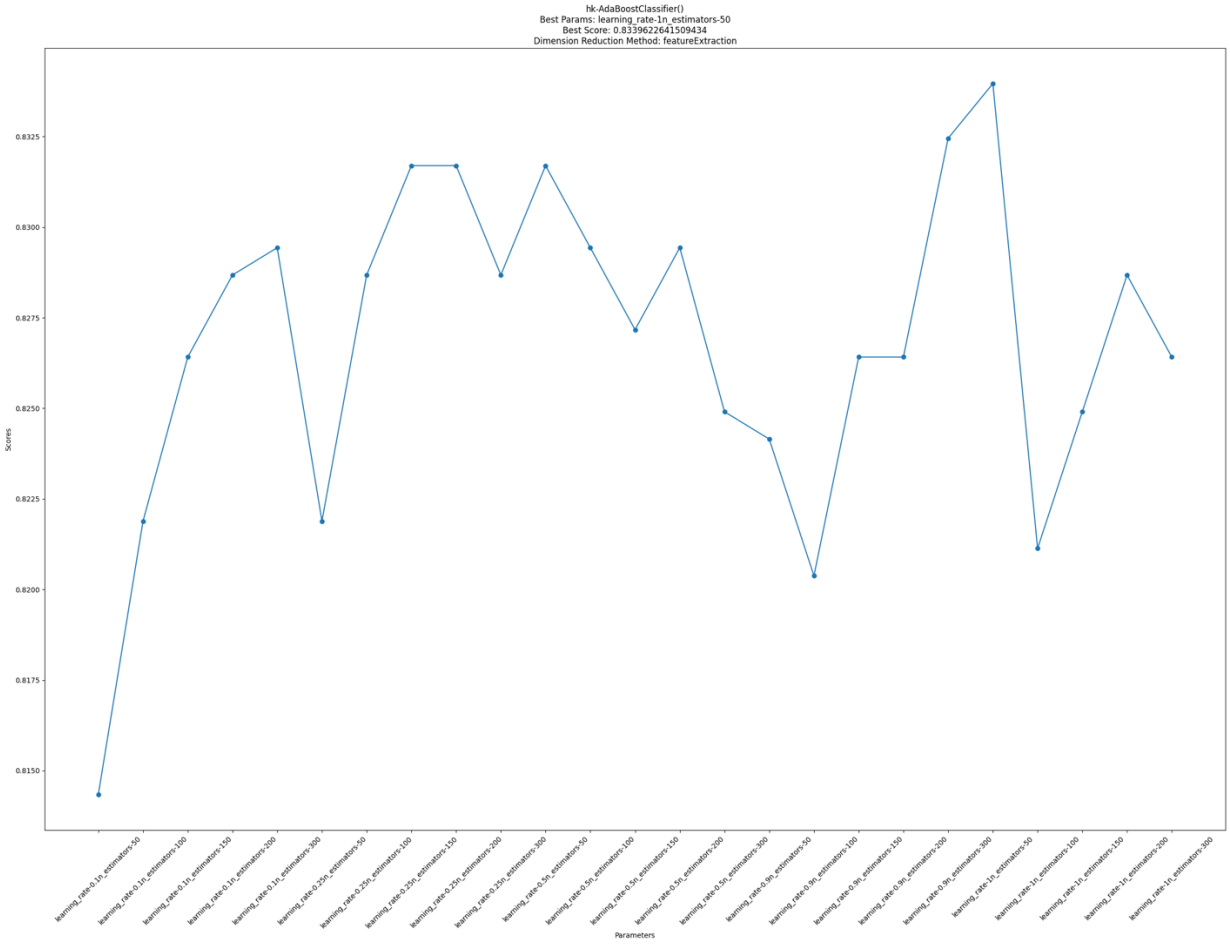

rv-SVC()
Best Params: C-100kernel-rbf
Best Score: 0.9978102189781023
Dimension Reduction Method: none
Scores
Parameters
1.00
0.99
0.98
0.97
0.96
0.95
C-0.01kernel-linear
C-0.01kernel-rbf
C-0.01kernel-poly
C-0.1kernel-linear
C-0.1kernel-rbf
C-0.1kernel-poly
C-1kernel-linear
C-1kernel-rbf
C-1kernel-poly
C-10kernel-linear
C-10kernel-rbf
C-10kernel-poly
C-100kernel-linear
C-100kernel-rbf
C-100kernel-poly

2.2 Graph the cross-validation results *(with dimension reduction)*

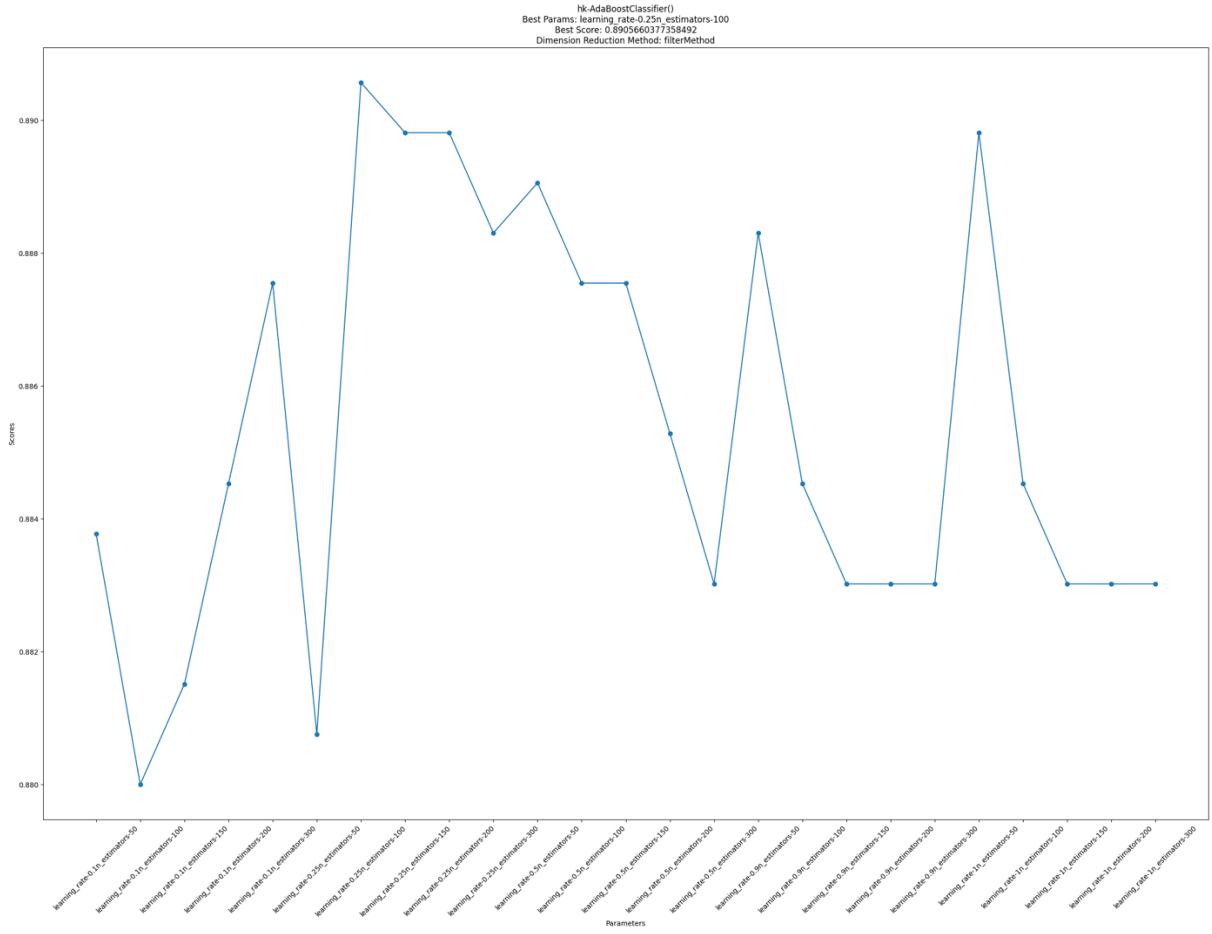
HK – AdaBoost Classifier – Embedded Method



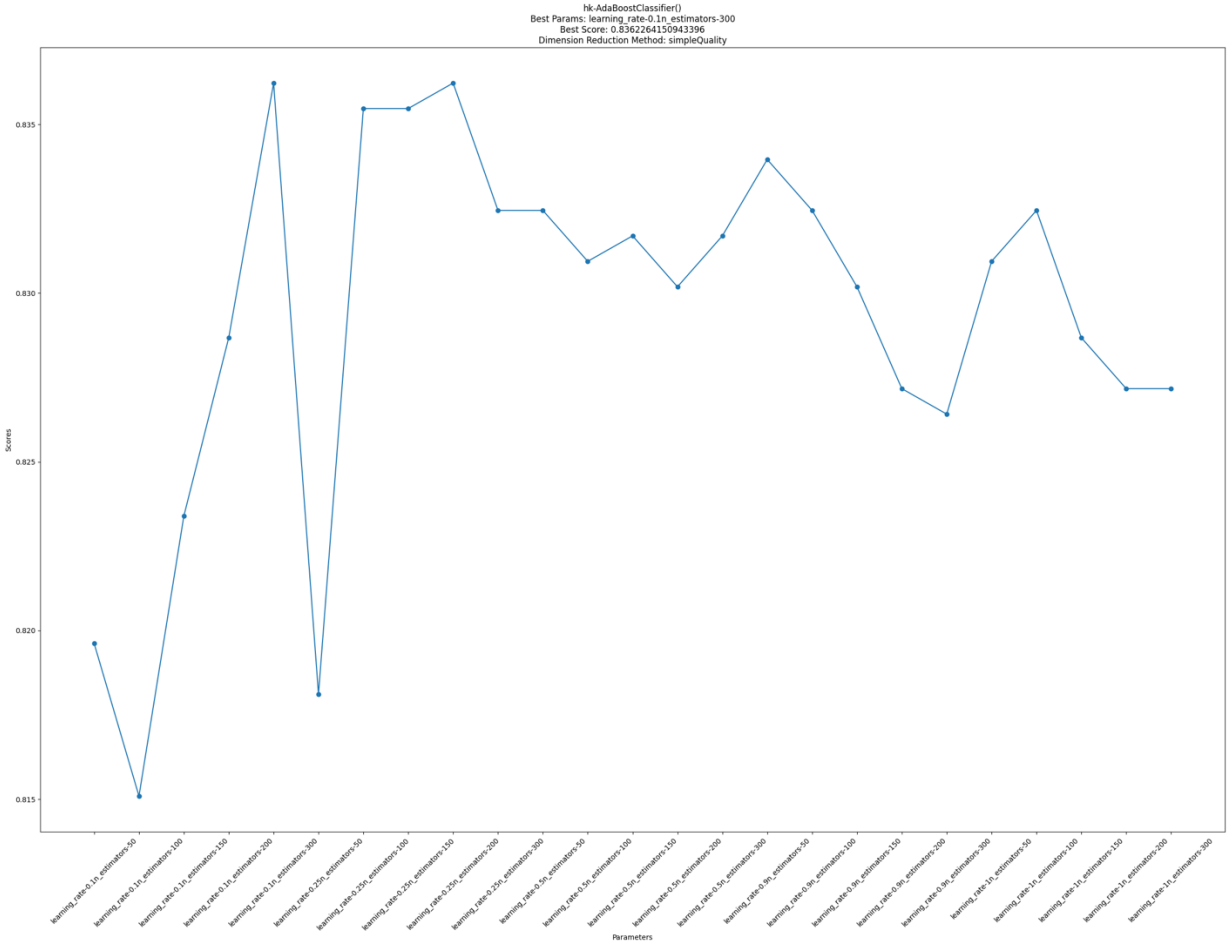
HK – AdaBoost Classifier – Feature Extraction



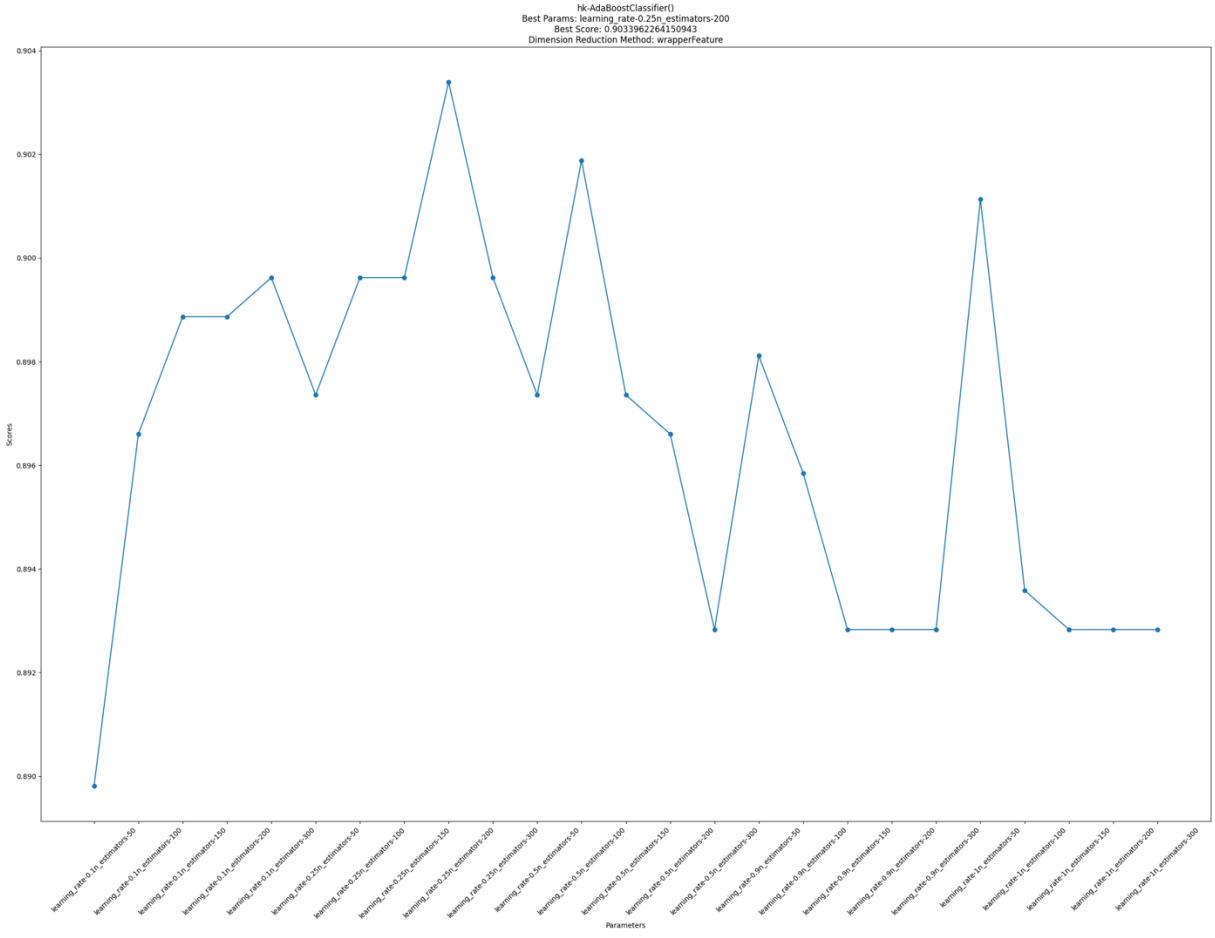
HK – AdaBoost Classifier – Filter Method



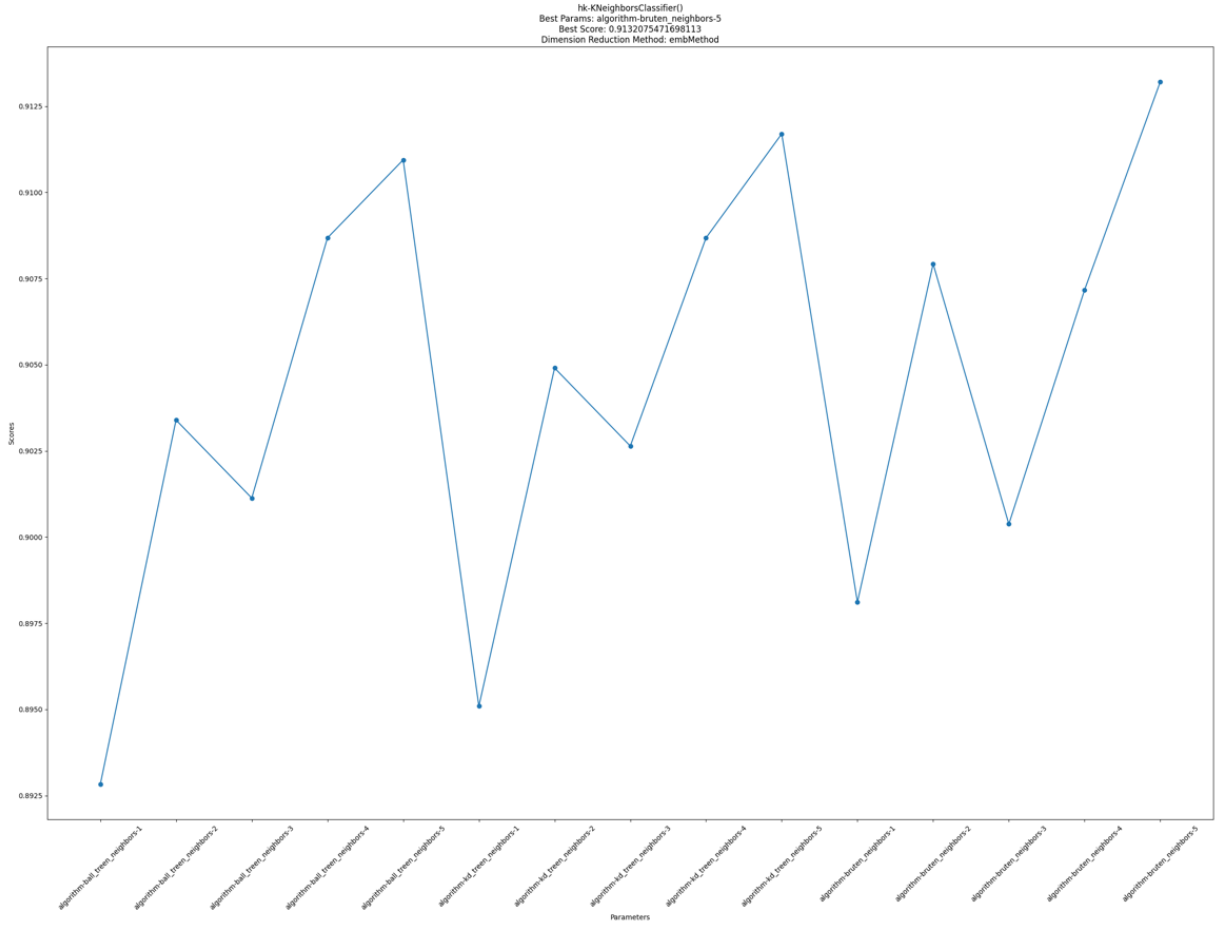
HK – AdaBoost Classifier – Simple Quality Filtering



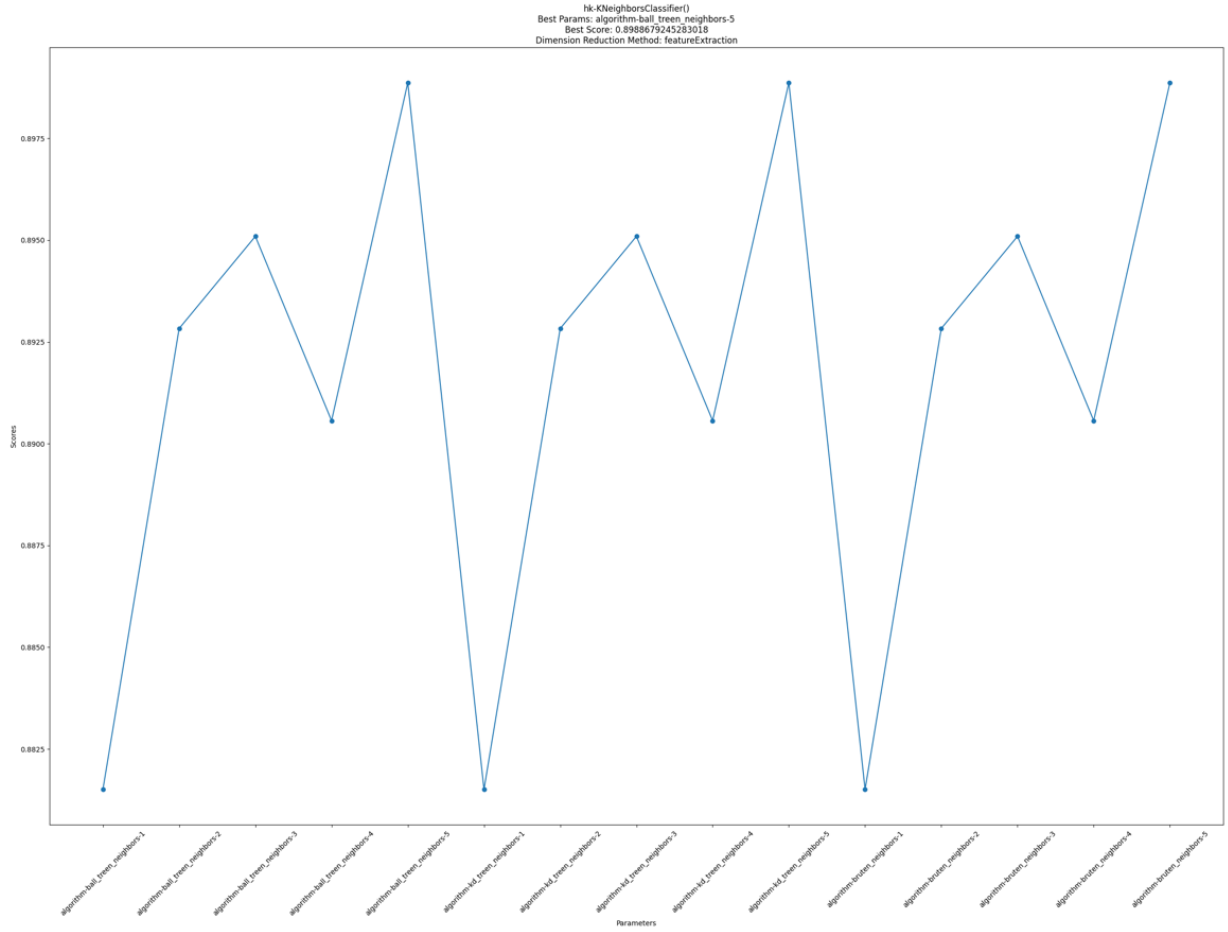
HK – AdaBoost Classifier – Wrapper Feature Selection



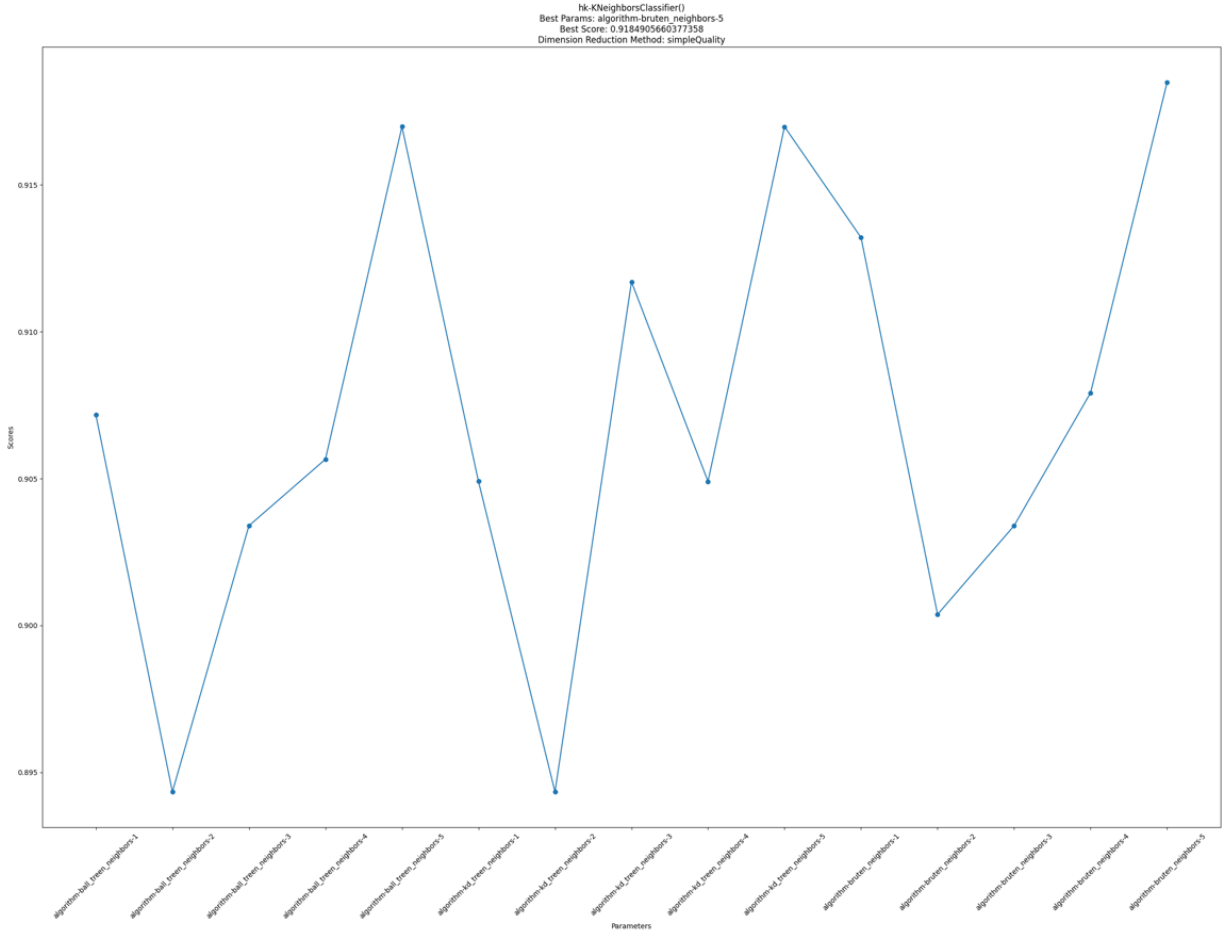
HK – K Neighbors Classifier – Embedded Methods



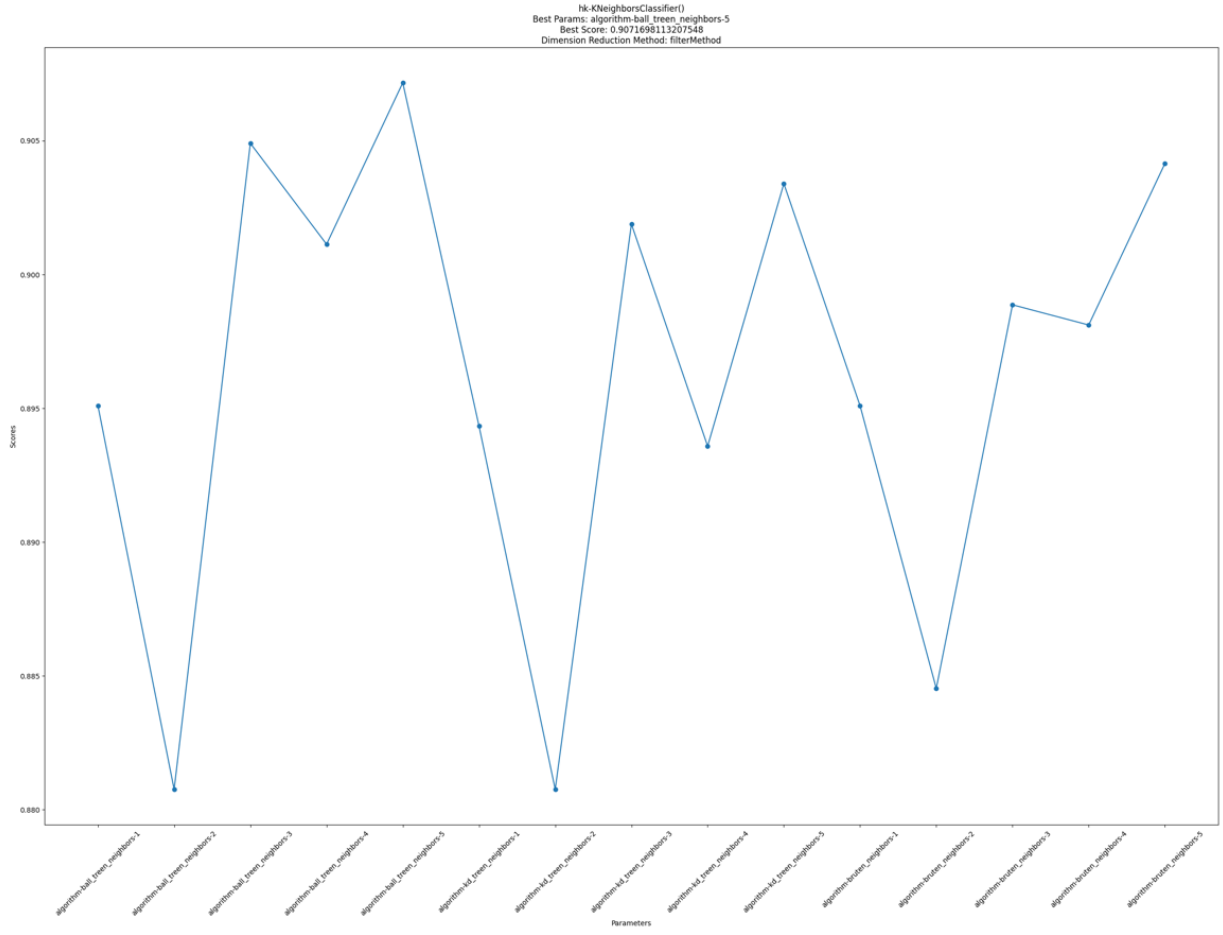
HK – K Neighbors Classifier – Feature Extraction



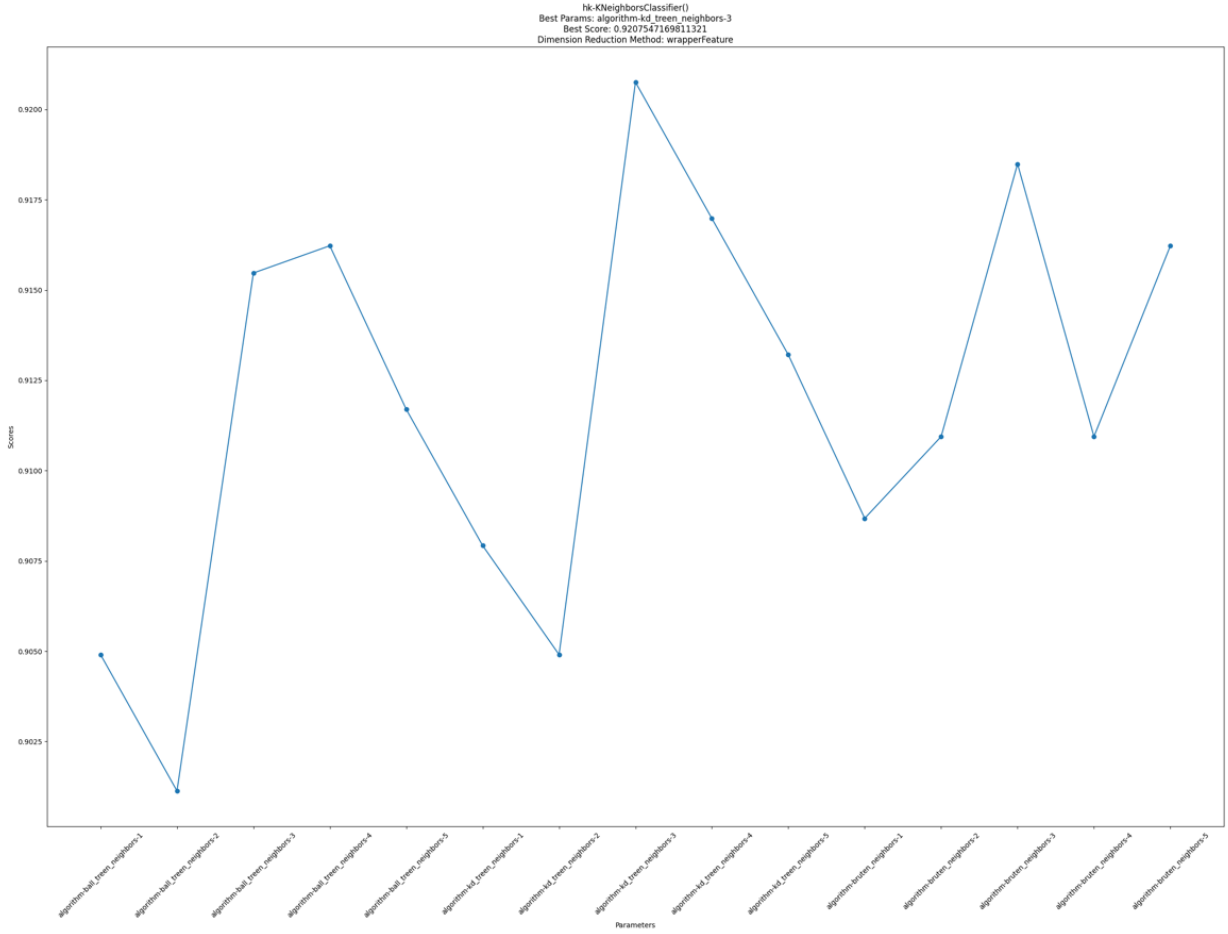
HK – K Neighbors Classifier – Simple Quality Filtering



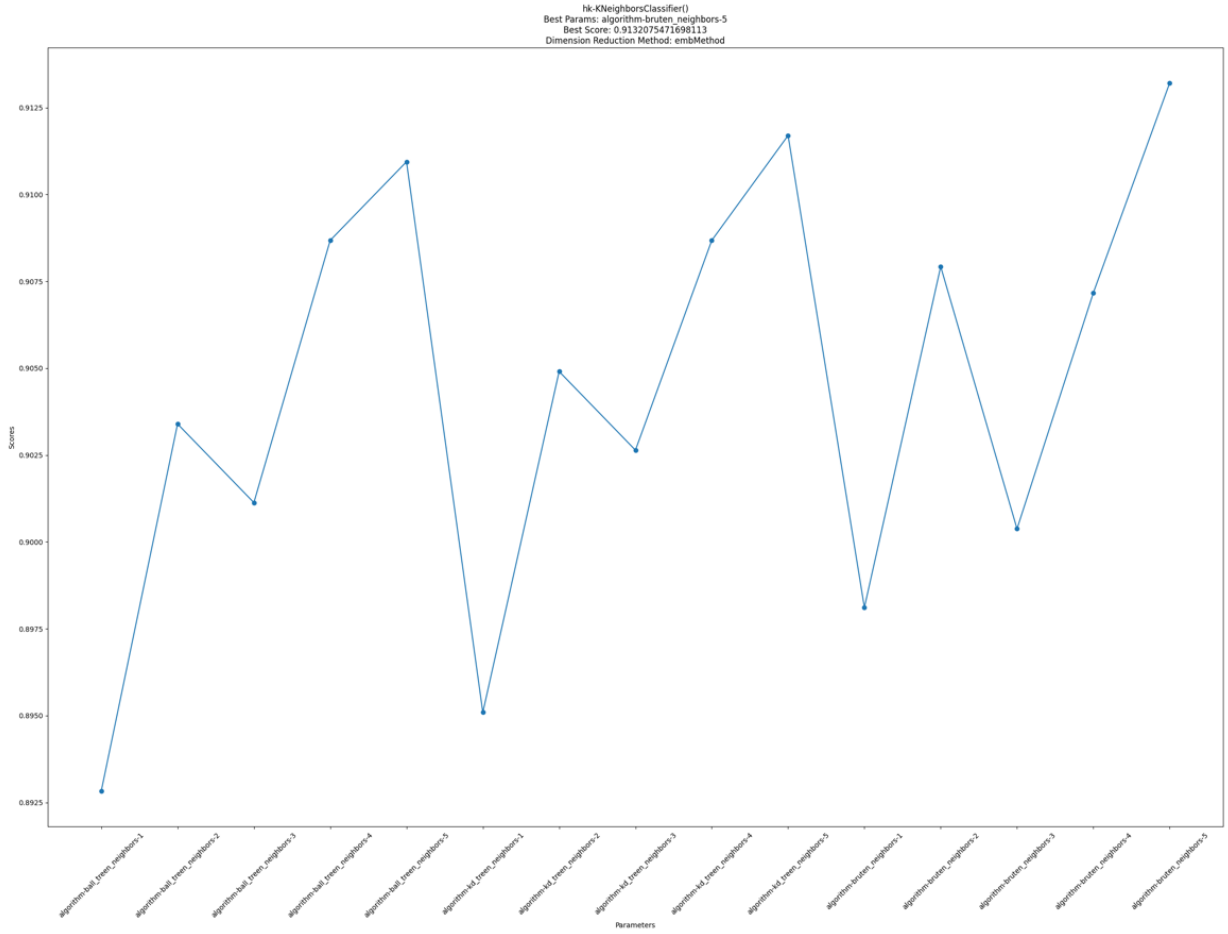
HK – K Neighbors Classifier – Filter Methods



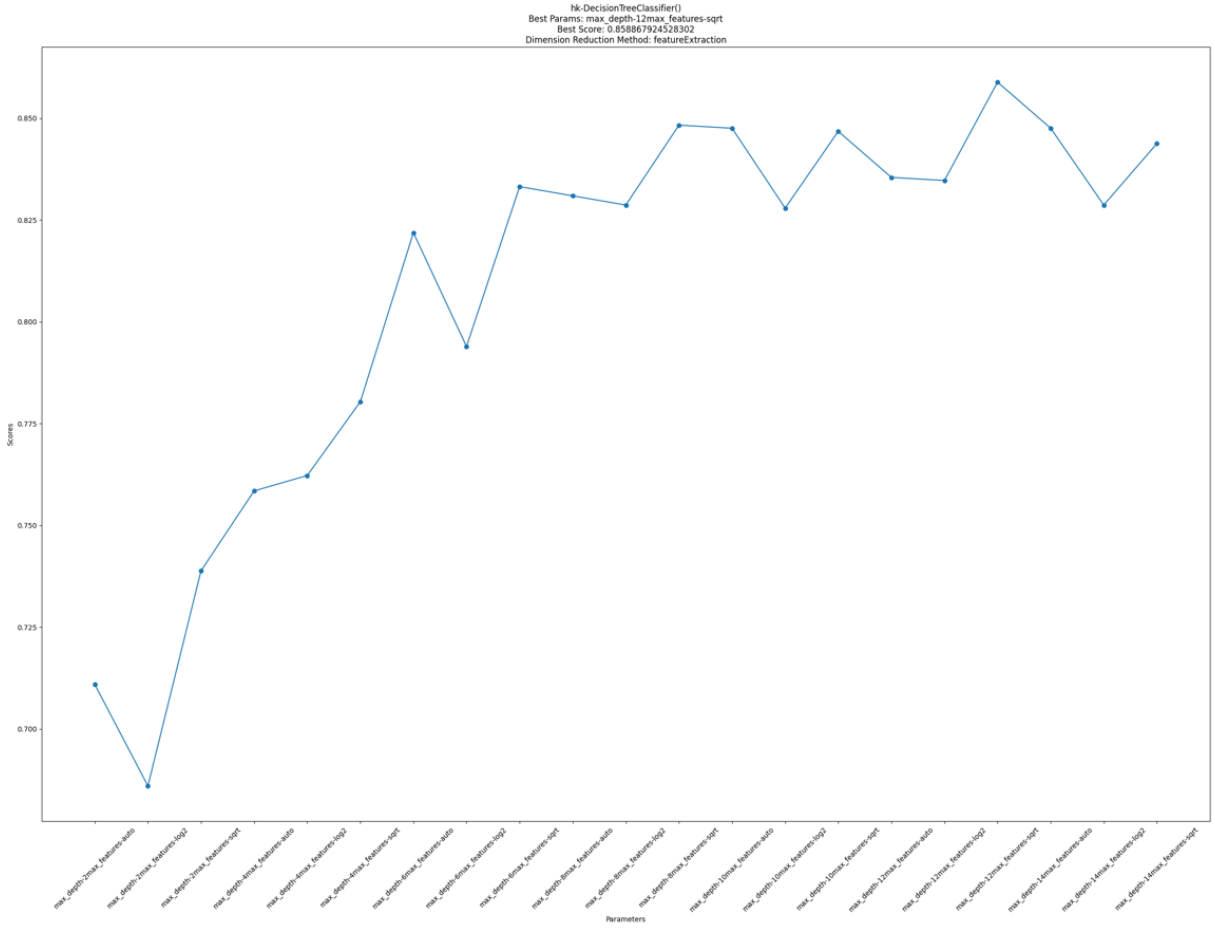
HK – K Neighbors Classifier – Wrapper Feature Selection



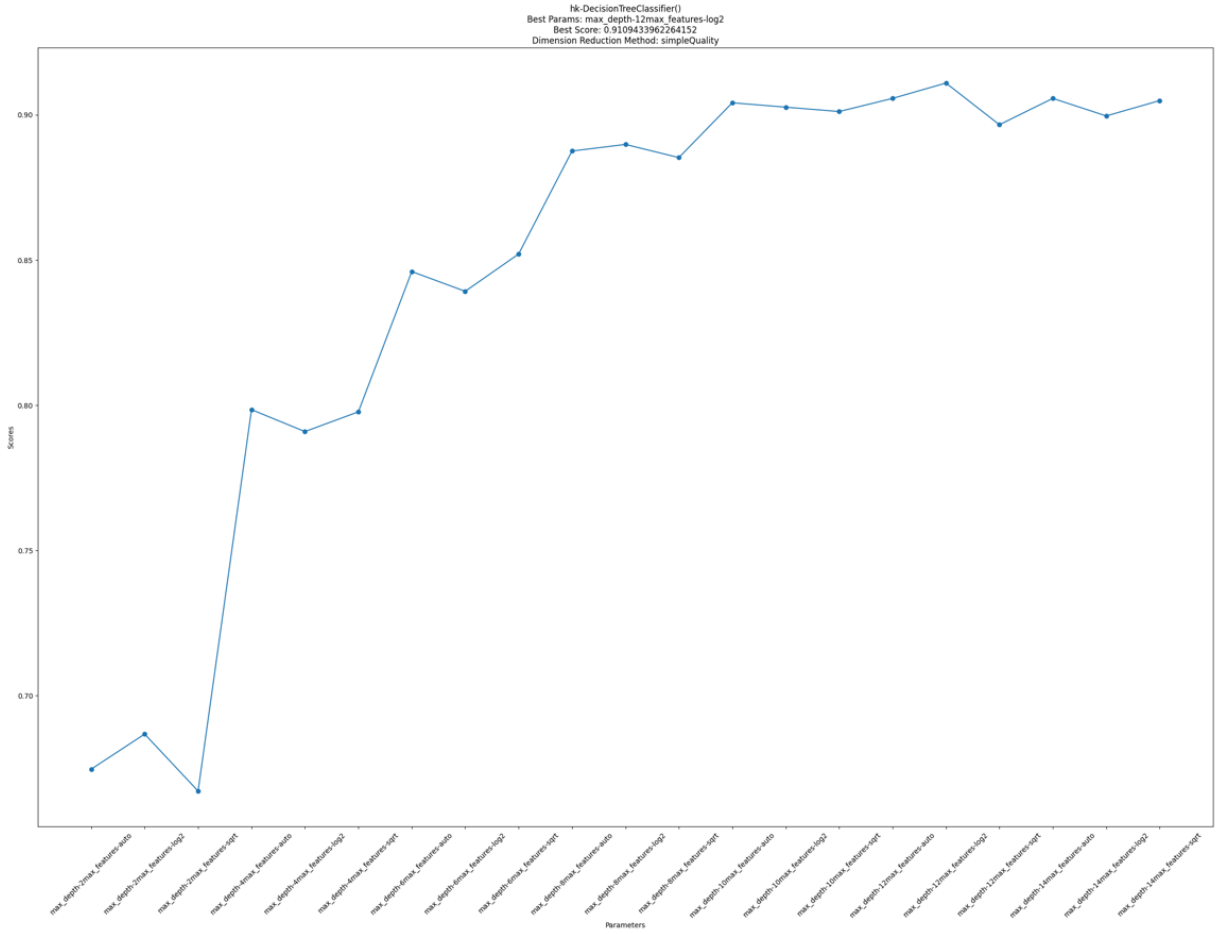
HK – Decision Tree Classifier – Embedded Methods



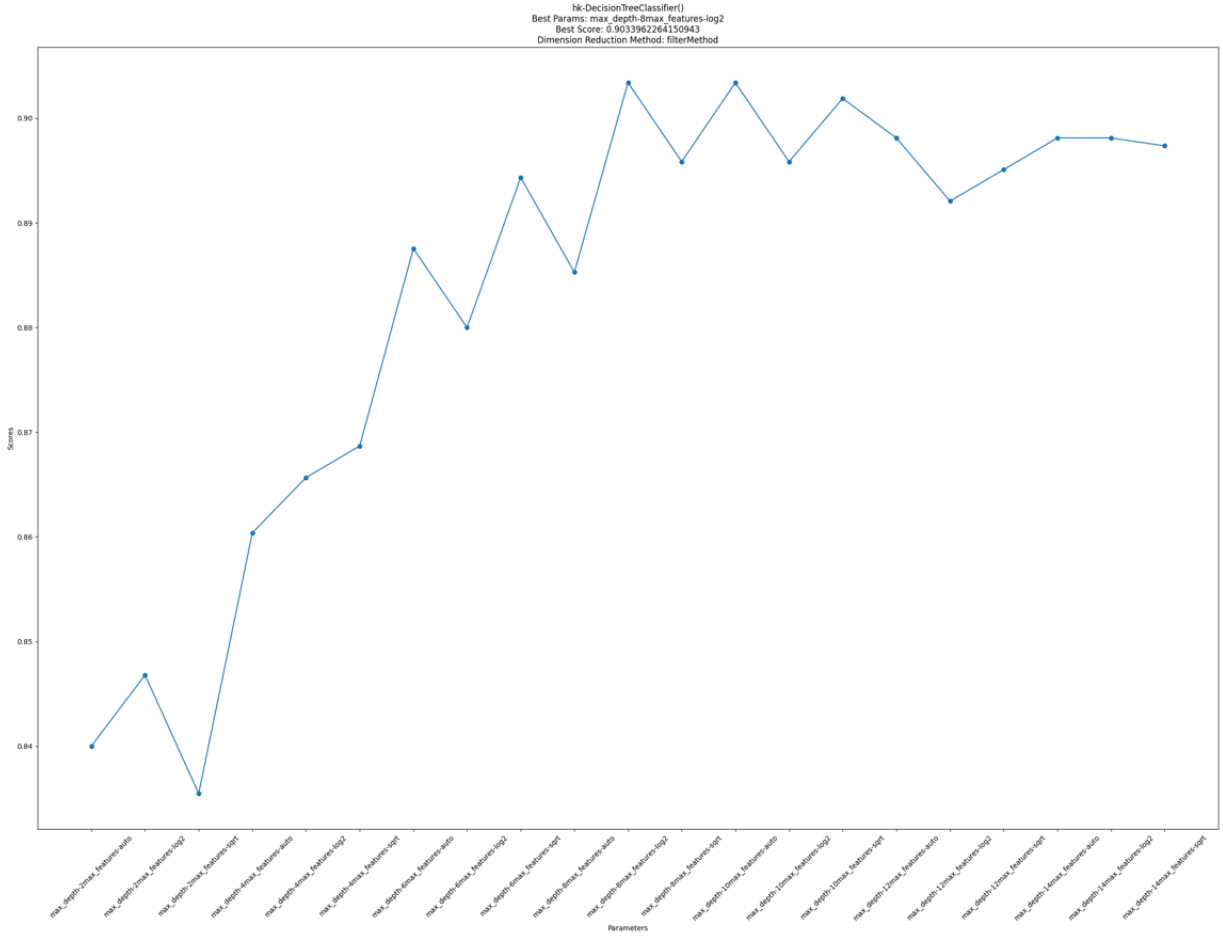
HK – Decision Tree Classifier – Feature Extraction



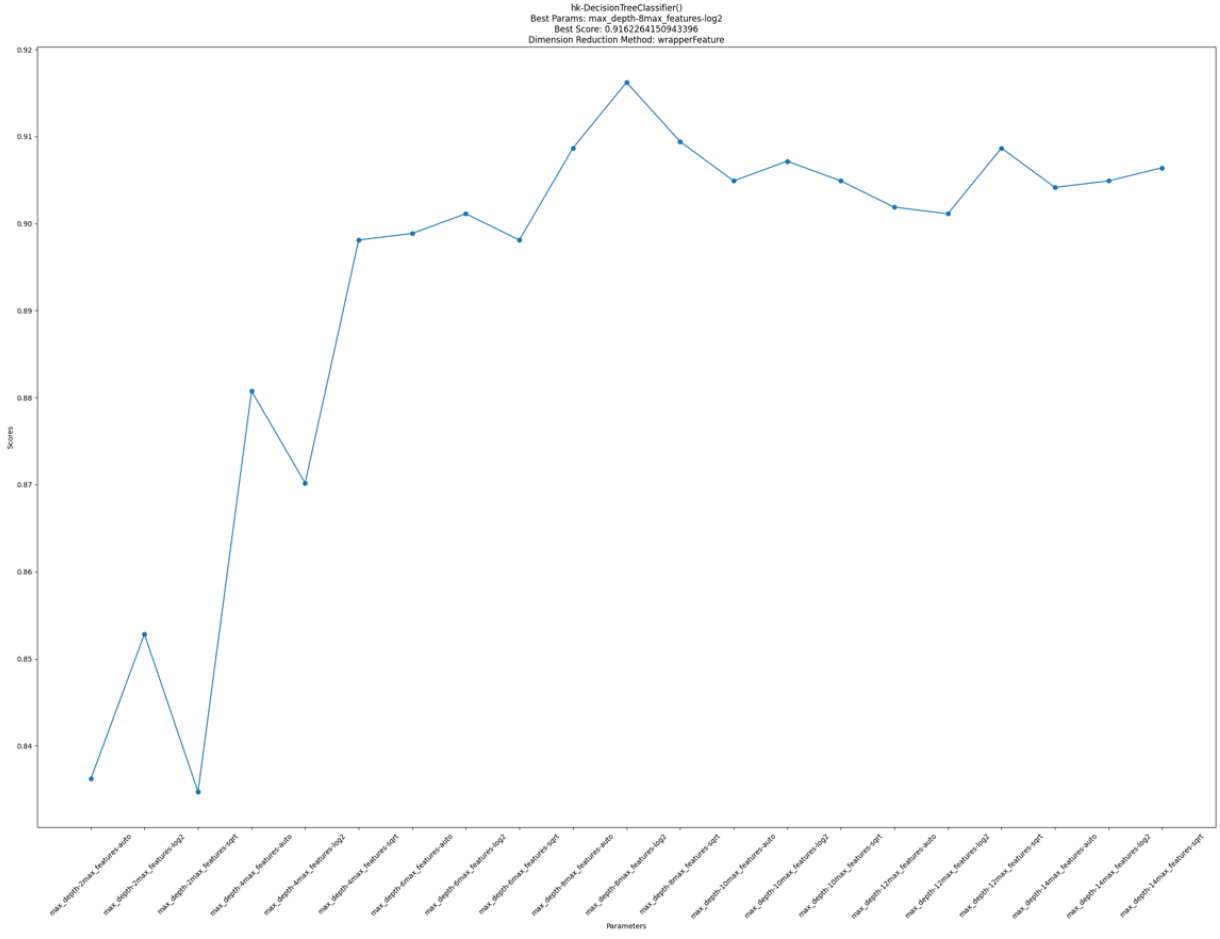
HK – Decision Tree Classifier – Simple Quality Filtering



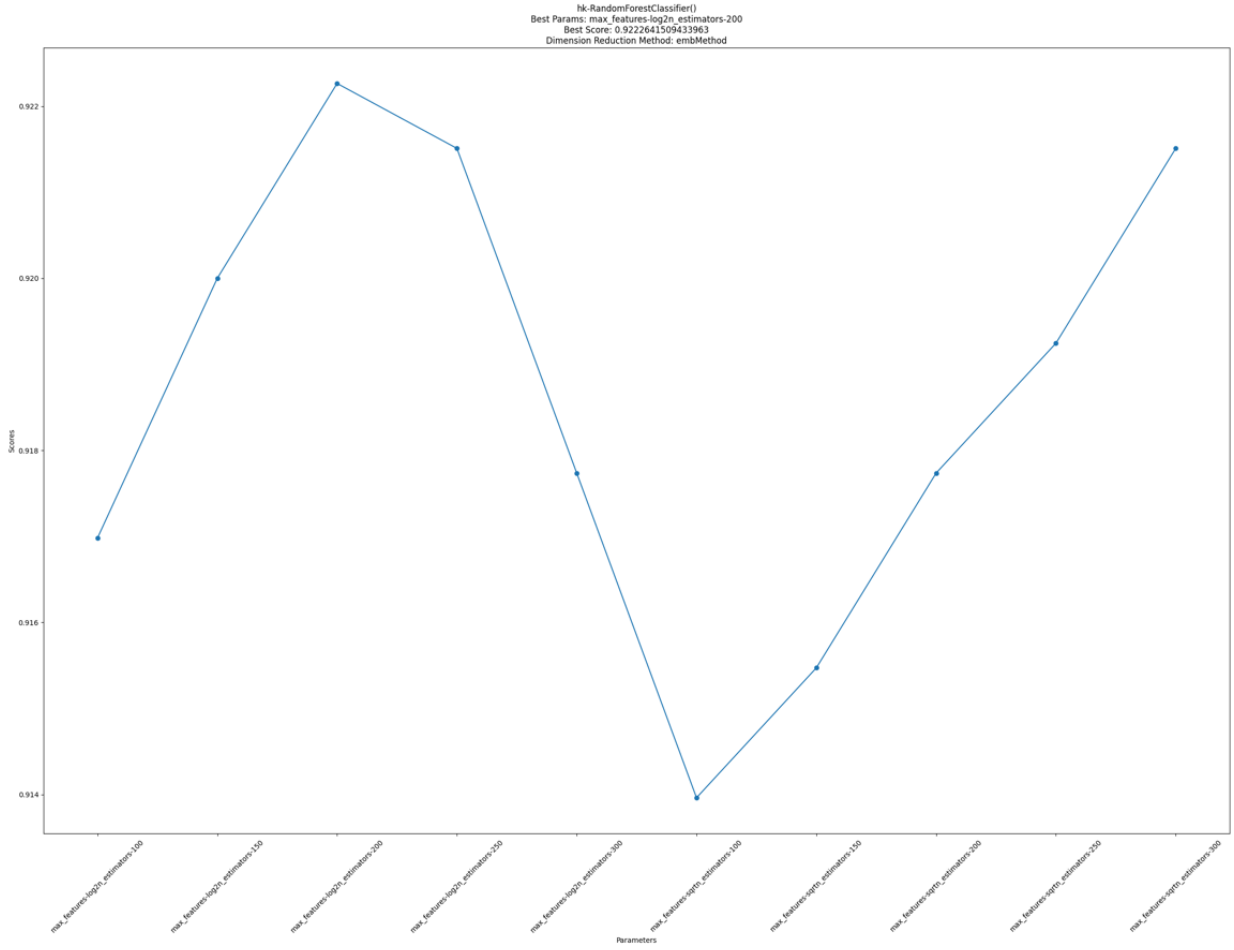
HK – Decision Tree Classifier – Filter Methods



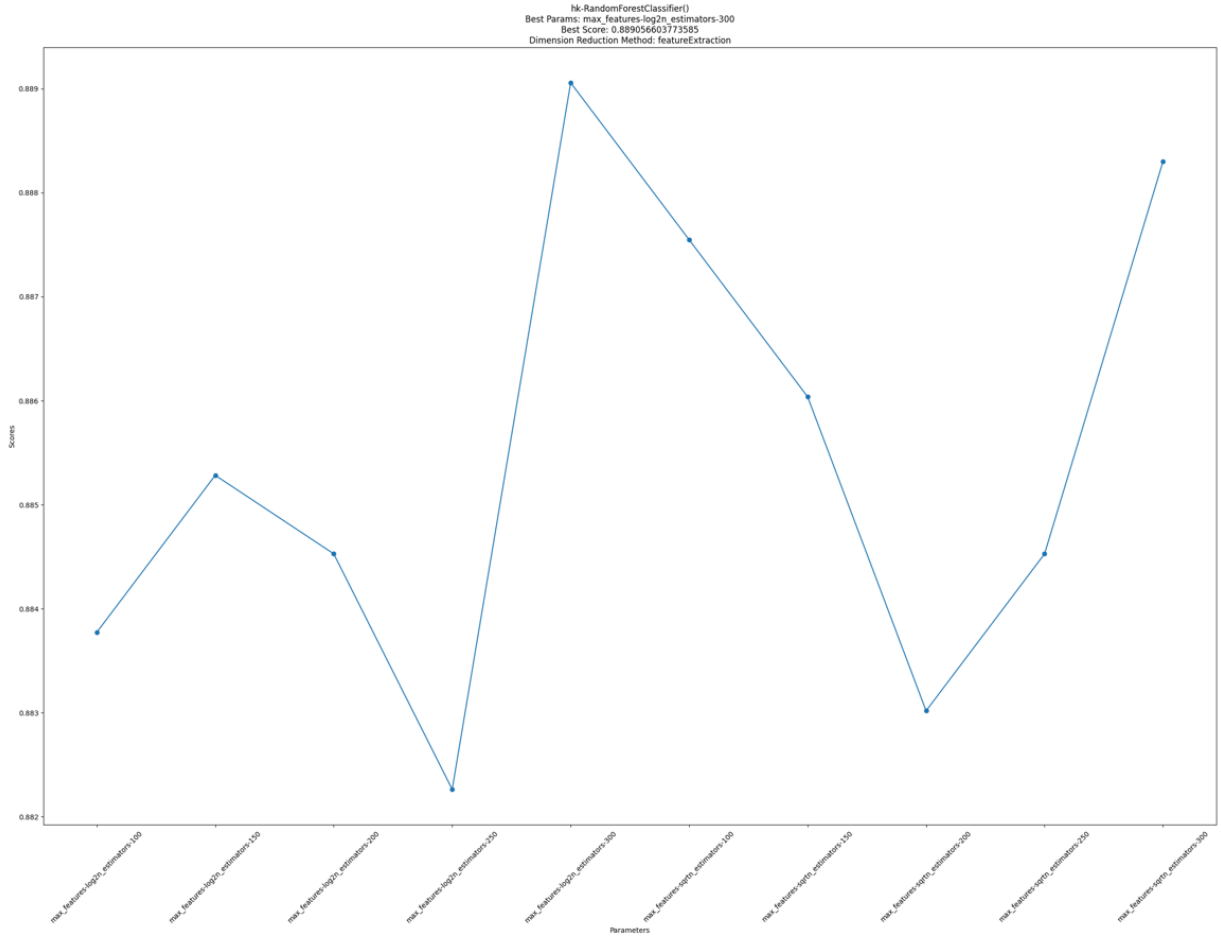
HK – Decision Tree Classifier – Wrapper Feature Selection



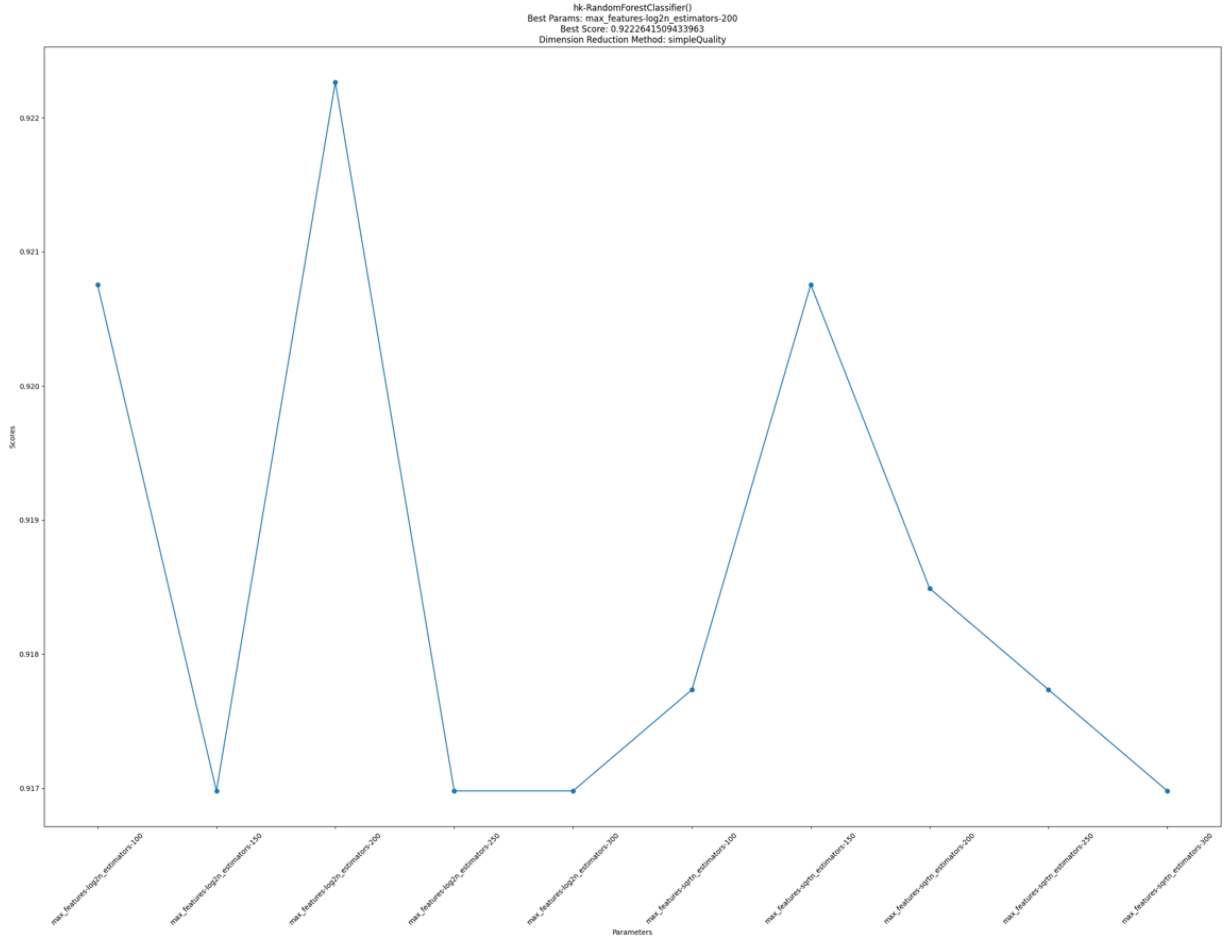
HK – Random Forest Classifier – Embedded Methods



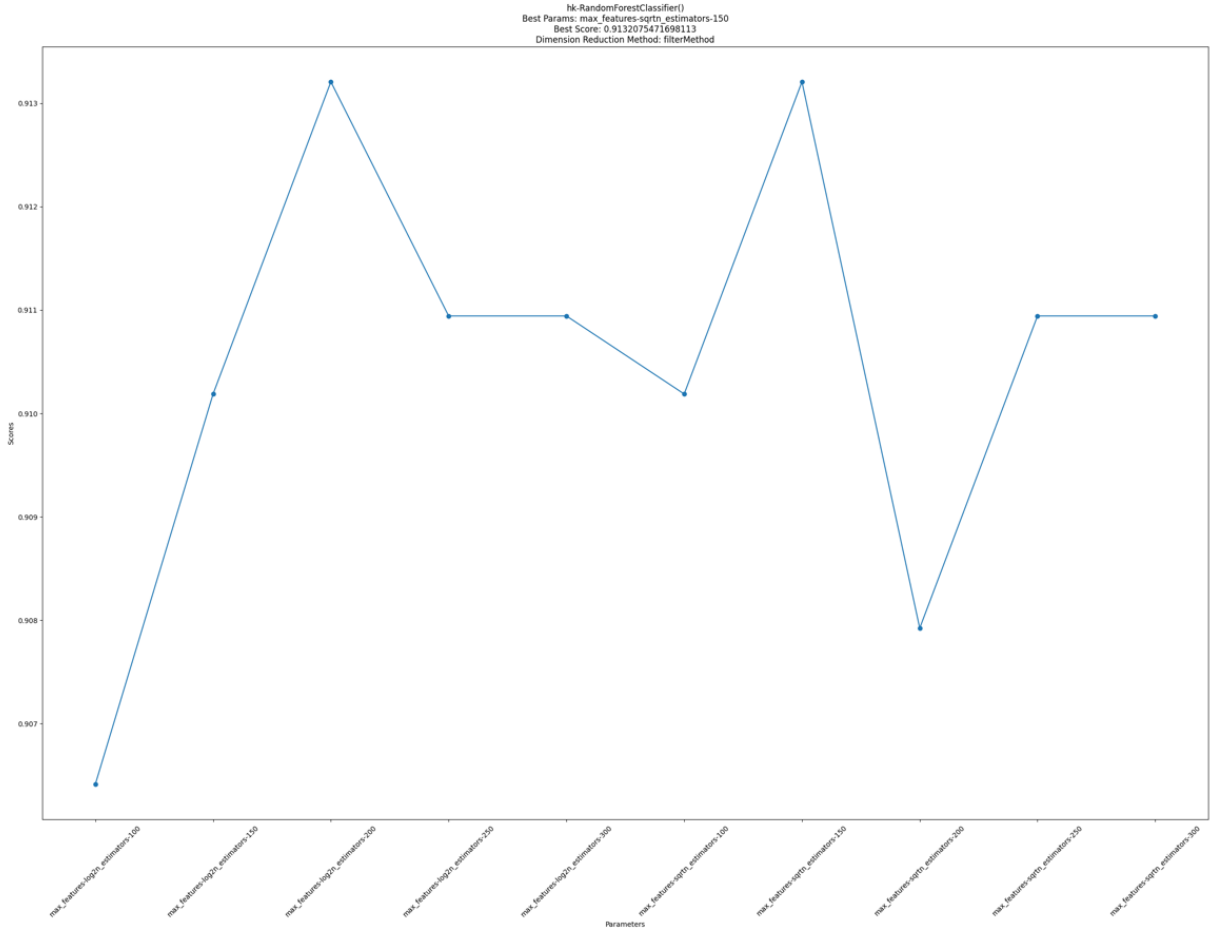
HK – Random Forest Classifier – Feature Extraction



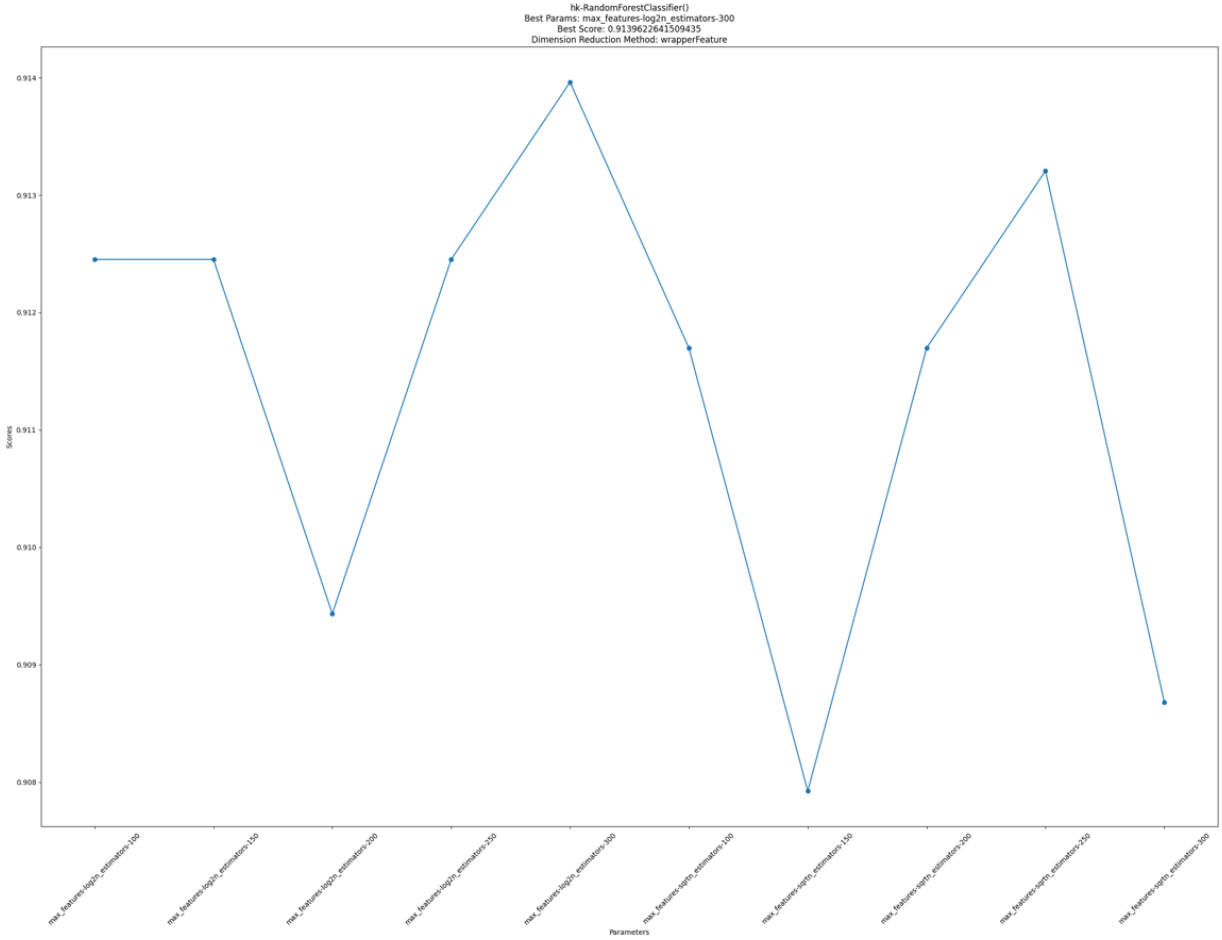
HK – Random Forest Classifier – Simple Quality Filtering



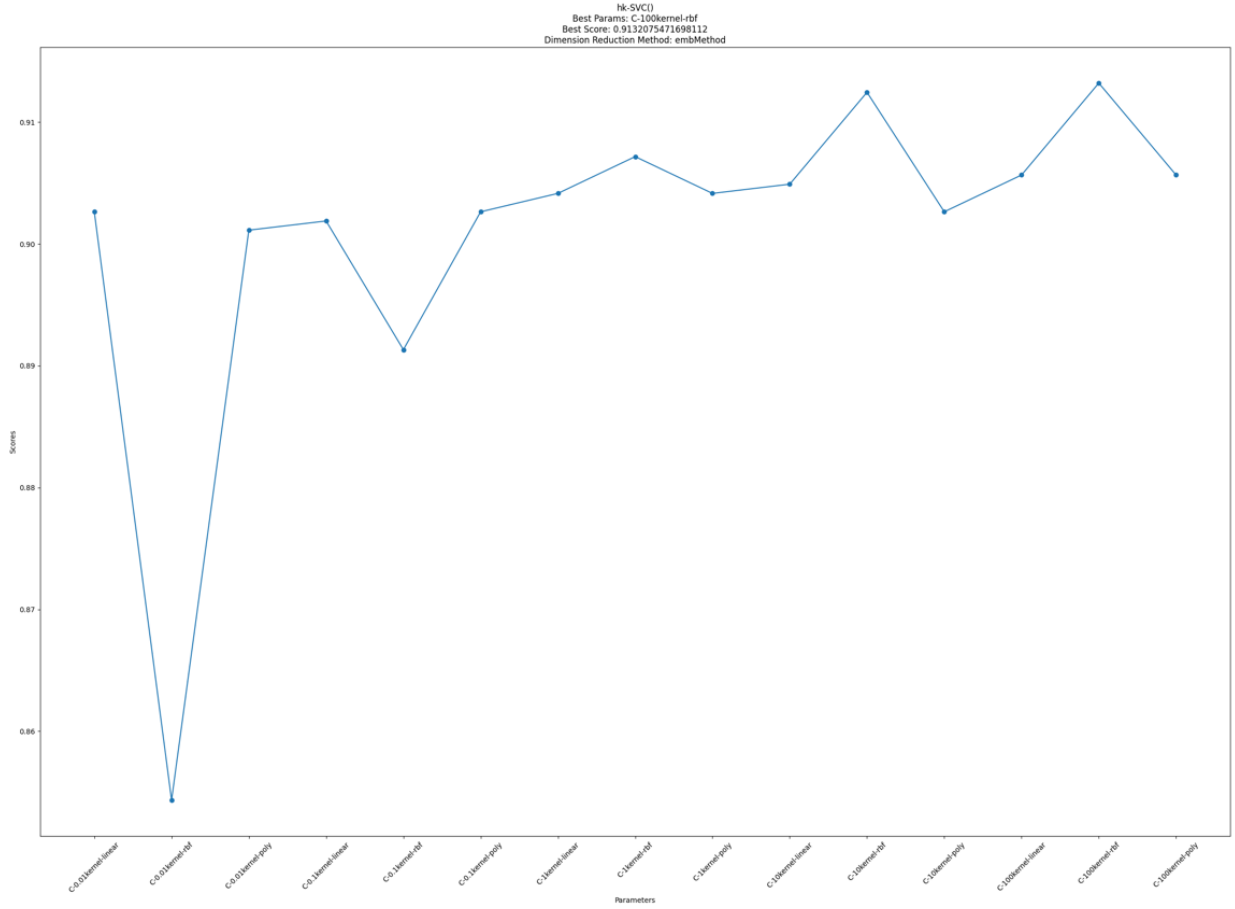
HK – Random Forest Classifier – Filter Methods



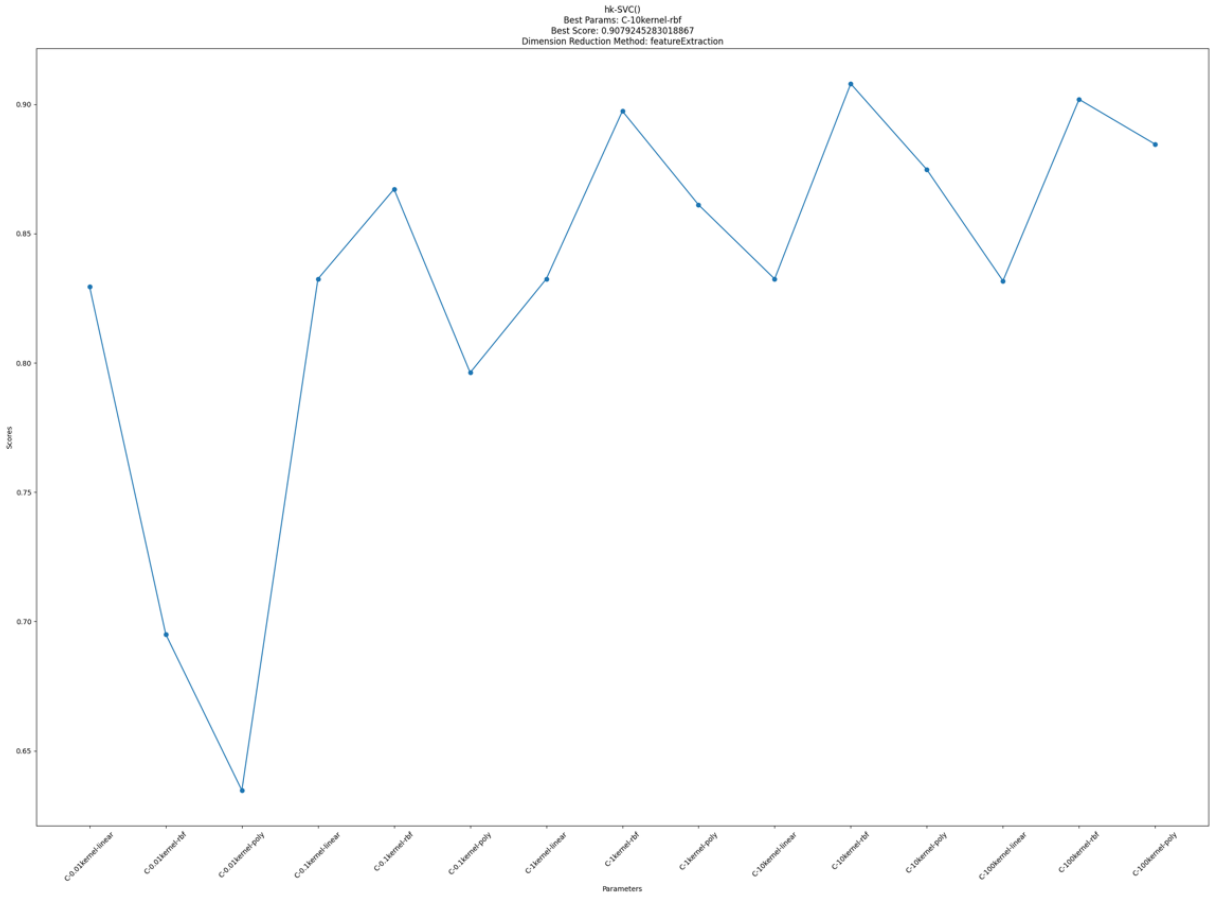
HK – Random Forest Classifier – Wrapper Feature Selection



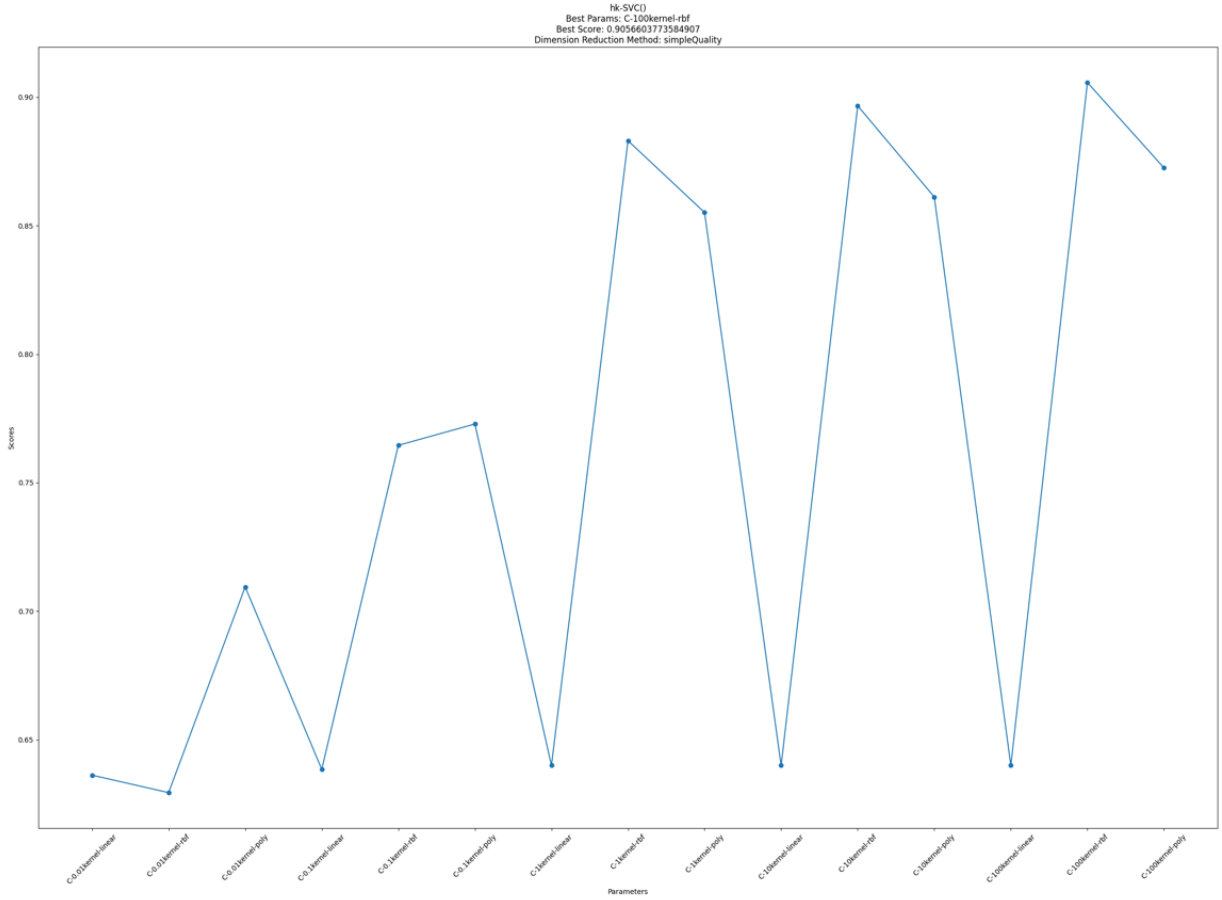
HK – SVM – Embedded Methods



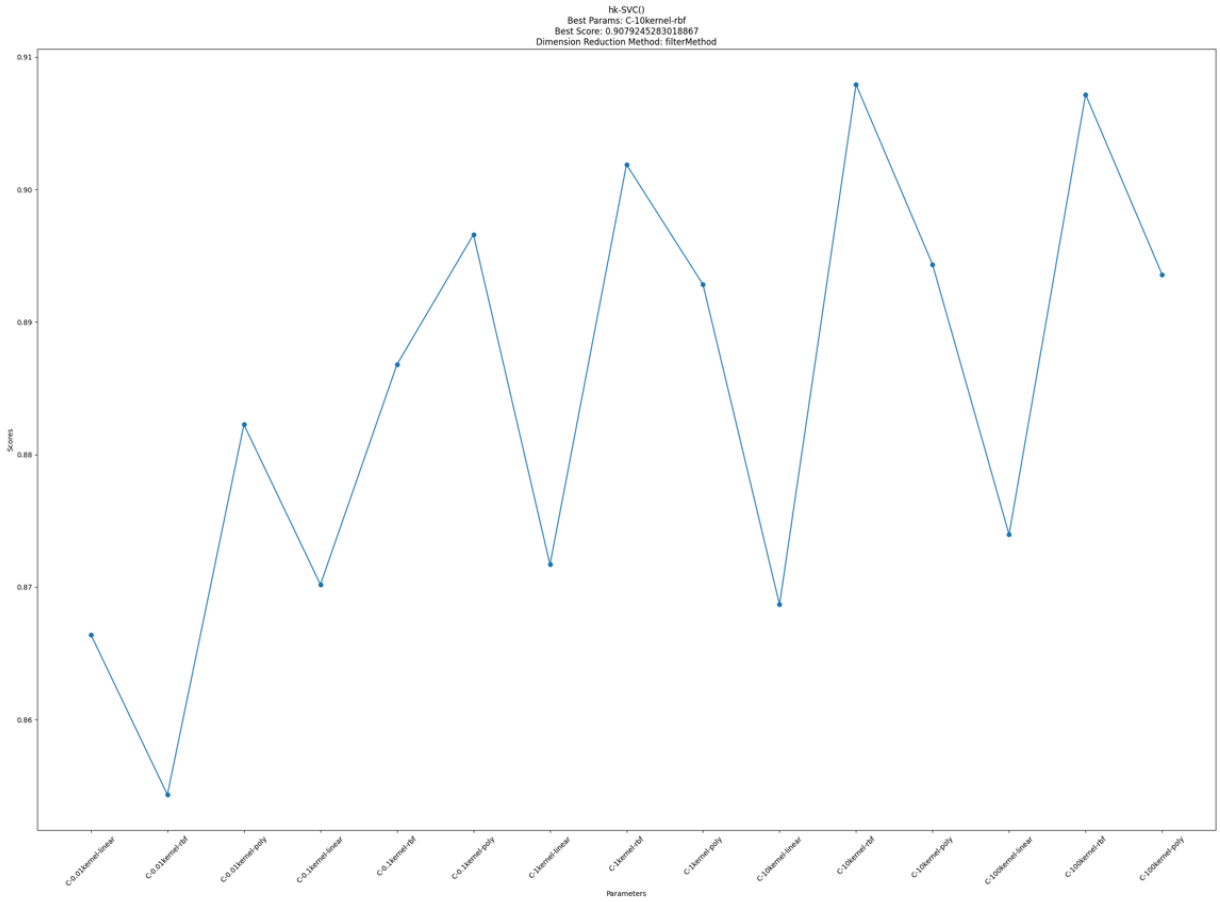
HK – SVM – Feature Extraction



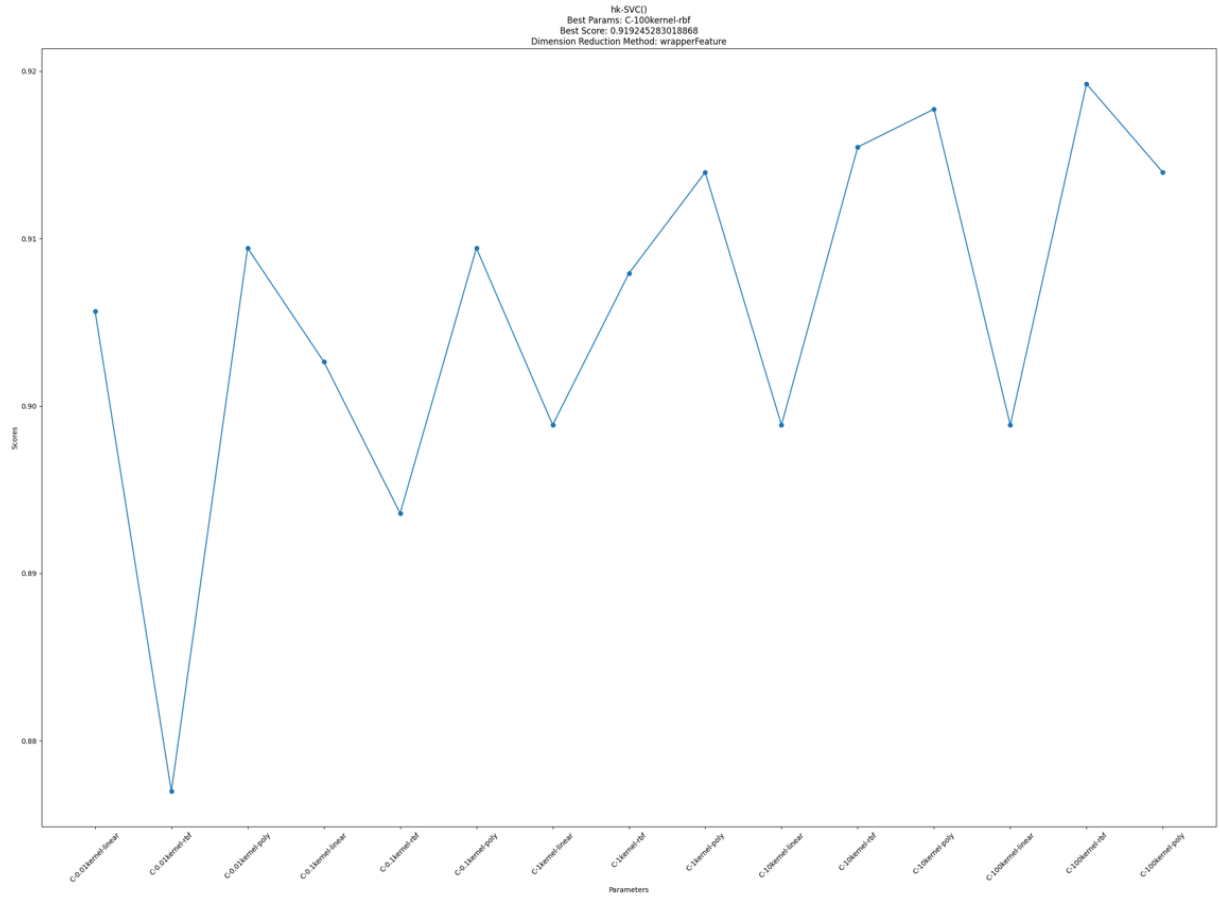
HK – SVM – Simple Quality Filtering



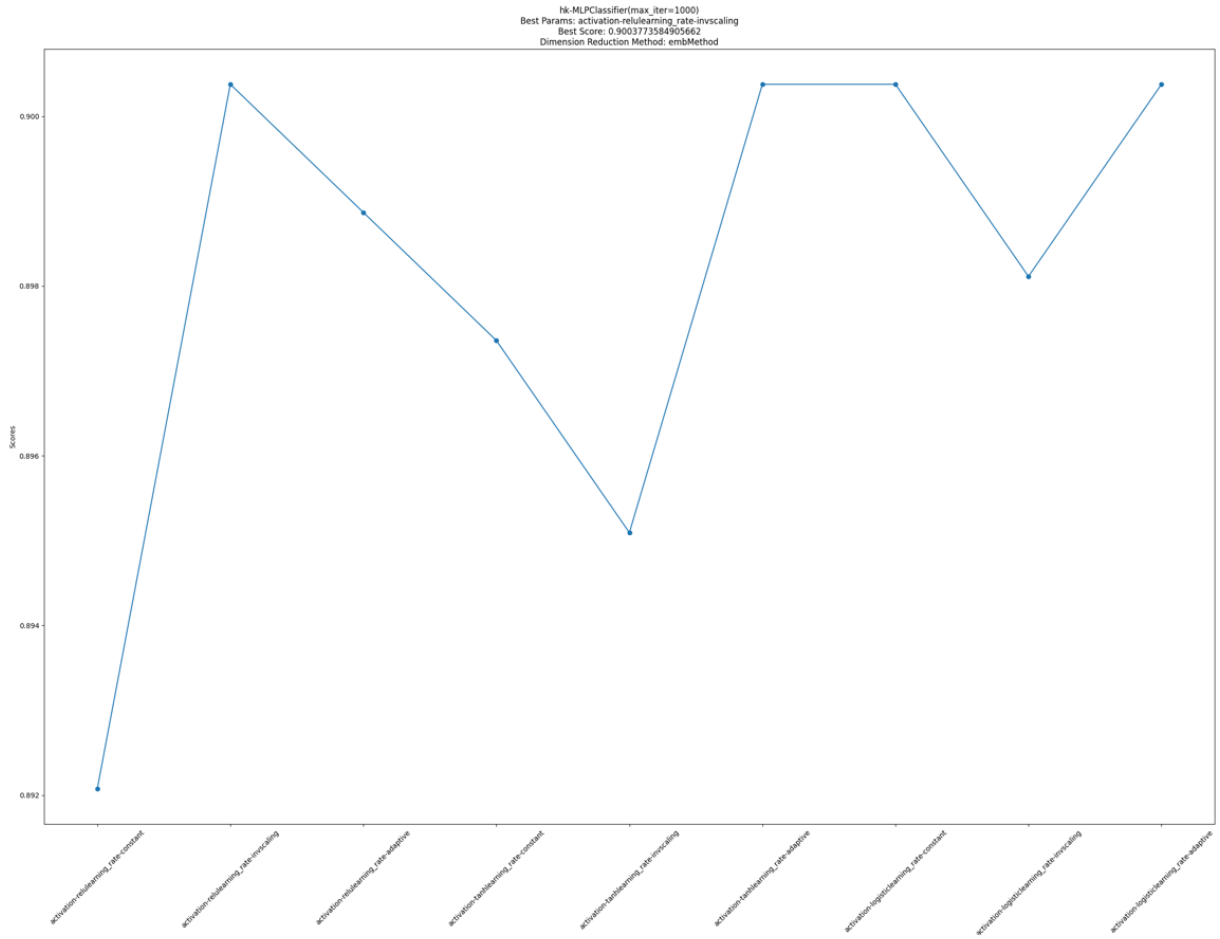
HK – SVM – Filter Methods



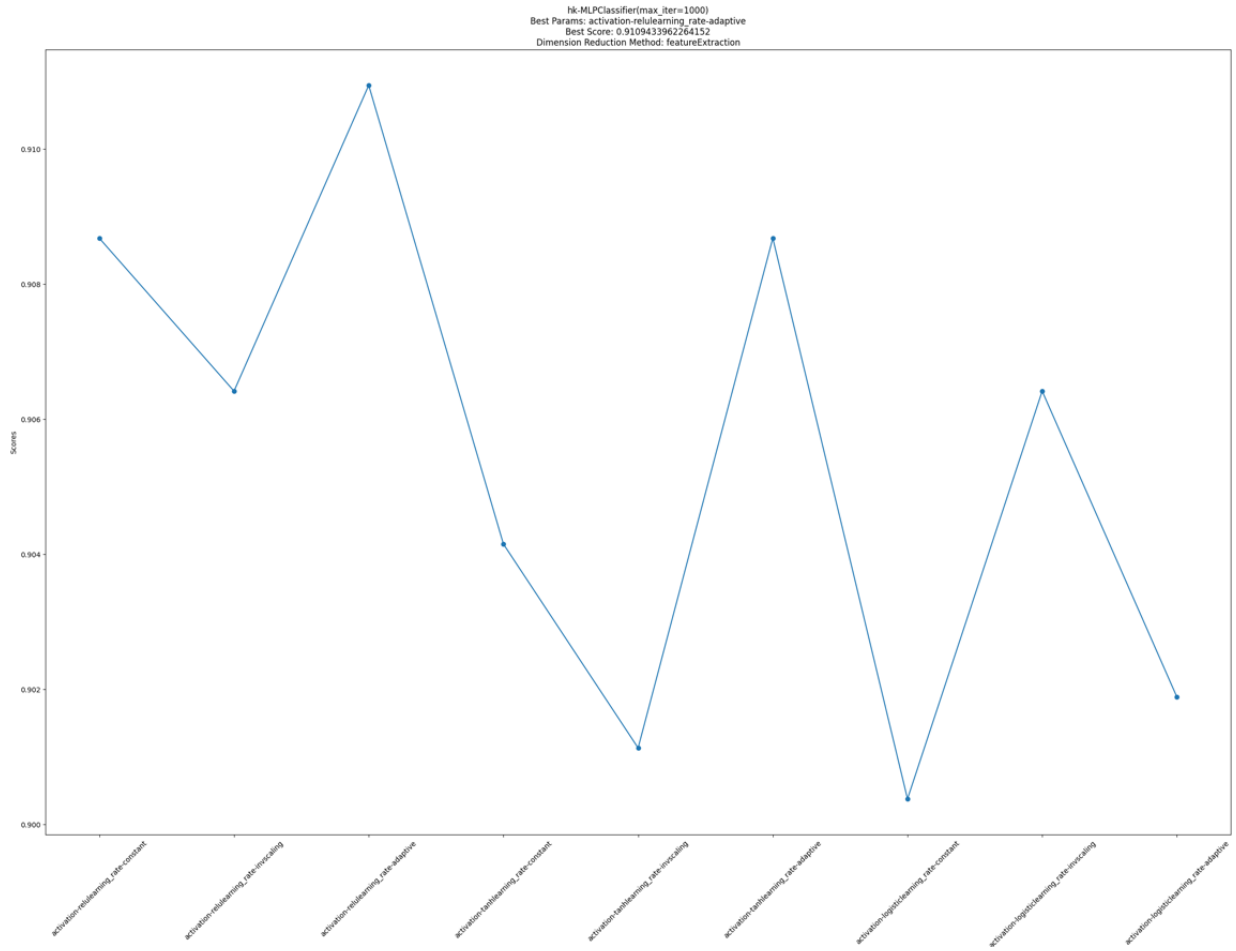
HK – SVM – Wrapper Feature Selection



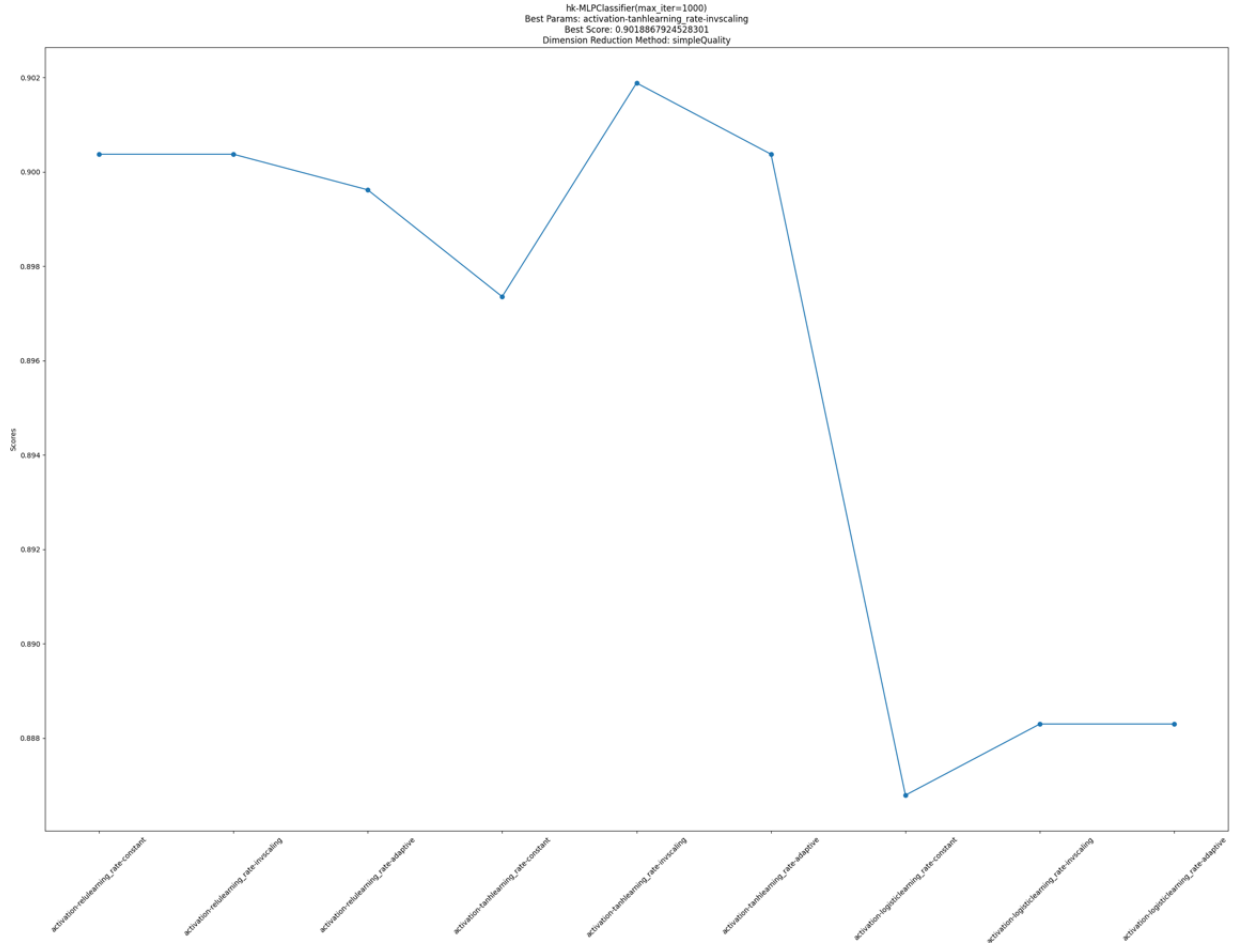
HK – MLP Classifier – Embedded Methods



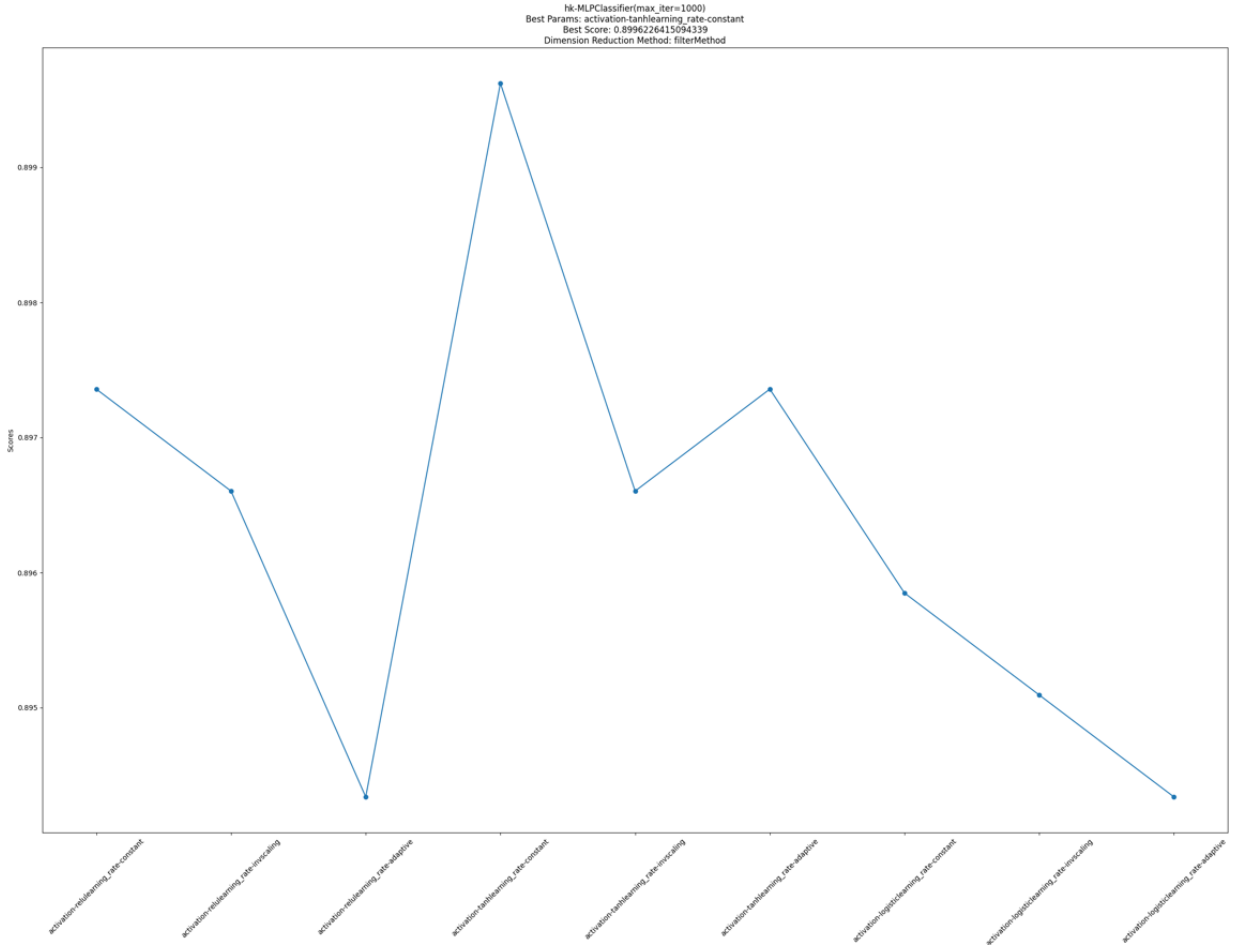
HK – MLP Classifier – Feature Extraction



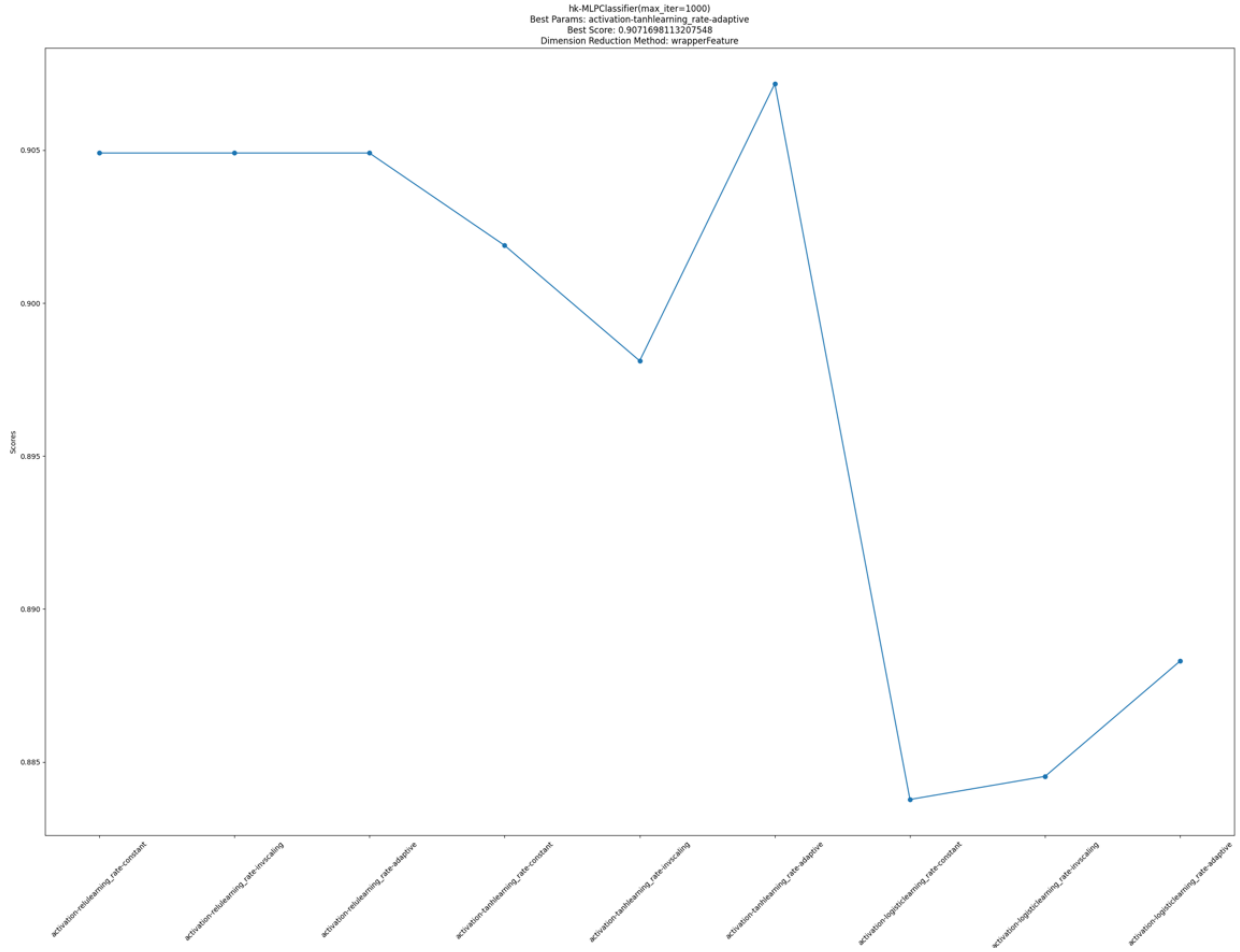
HK – MLP Classifier – Simple Quality Filtering



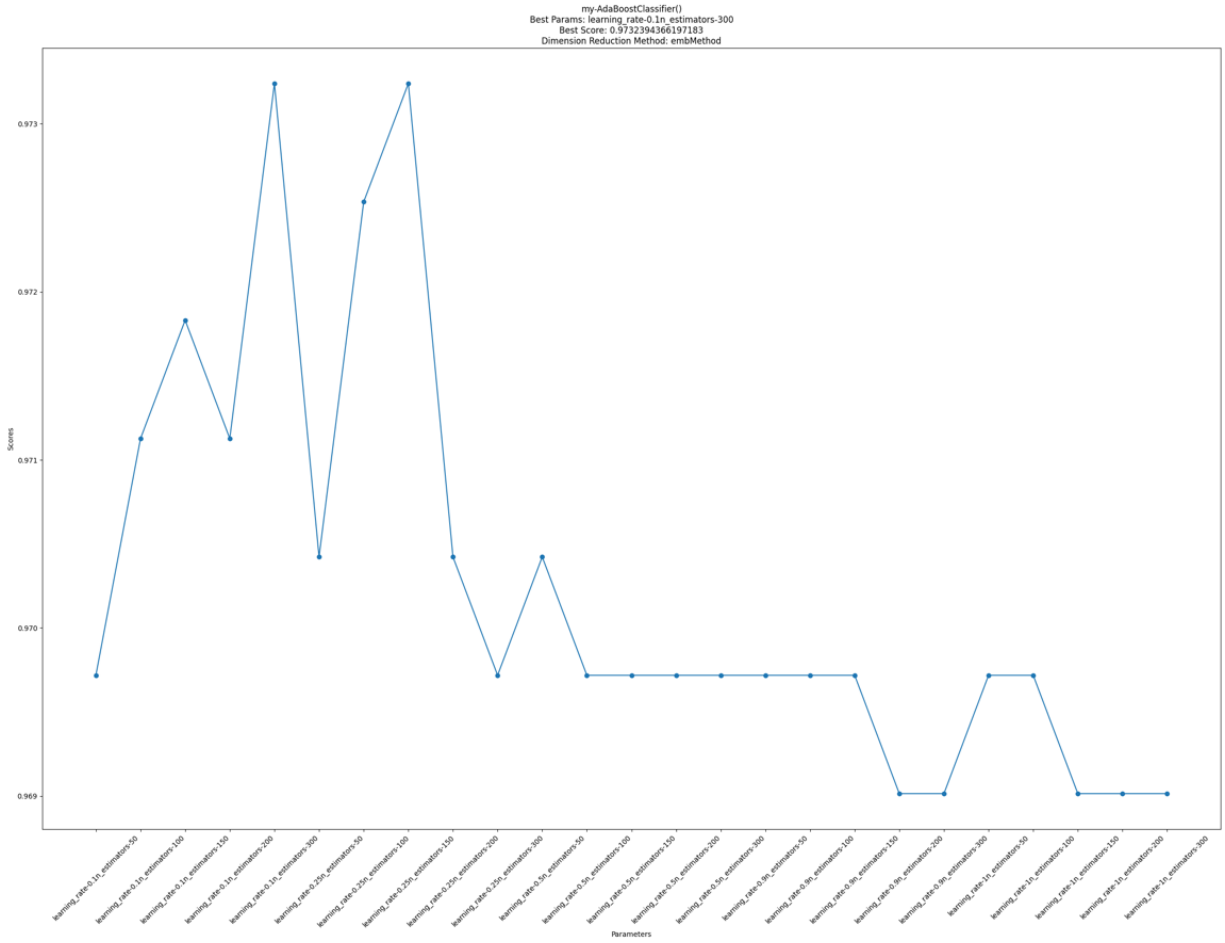
HK – MLP Classifier – Filter Methods



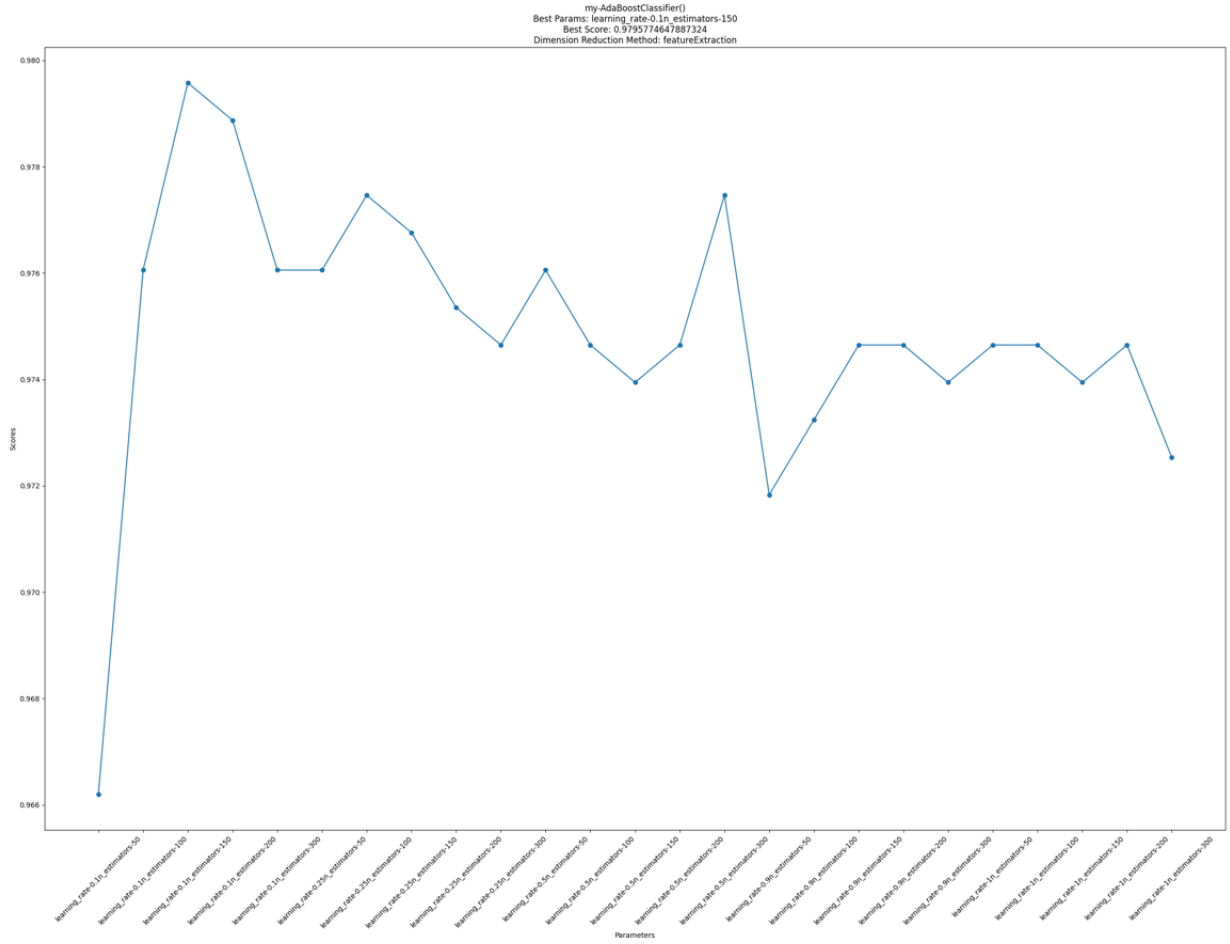
HK – MLP Classifier – Wrapper Feature Selection



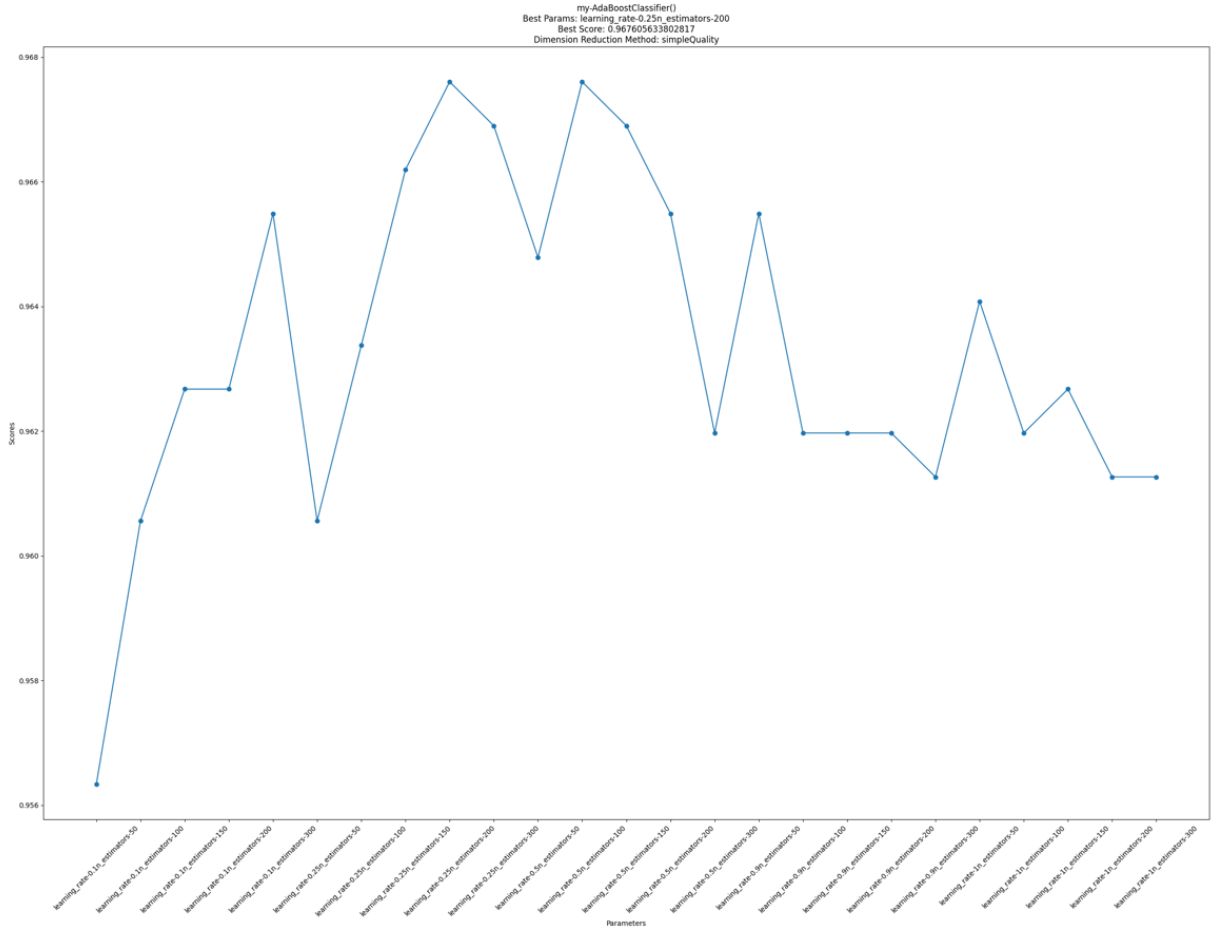
MY – AdaBoost Classifier – Embedded Methods



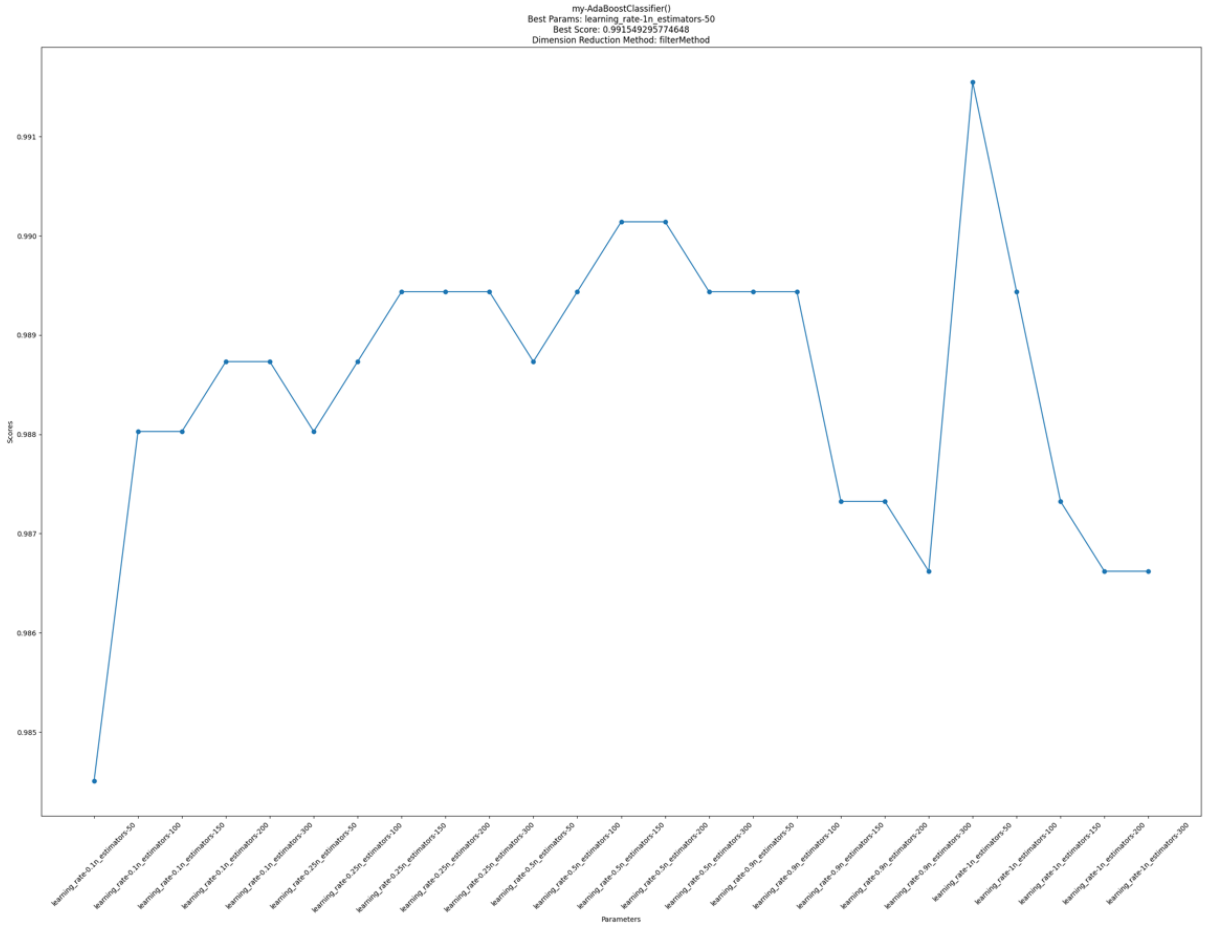
MY – AdaBoost Classifier – Feature Extraction



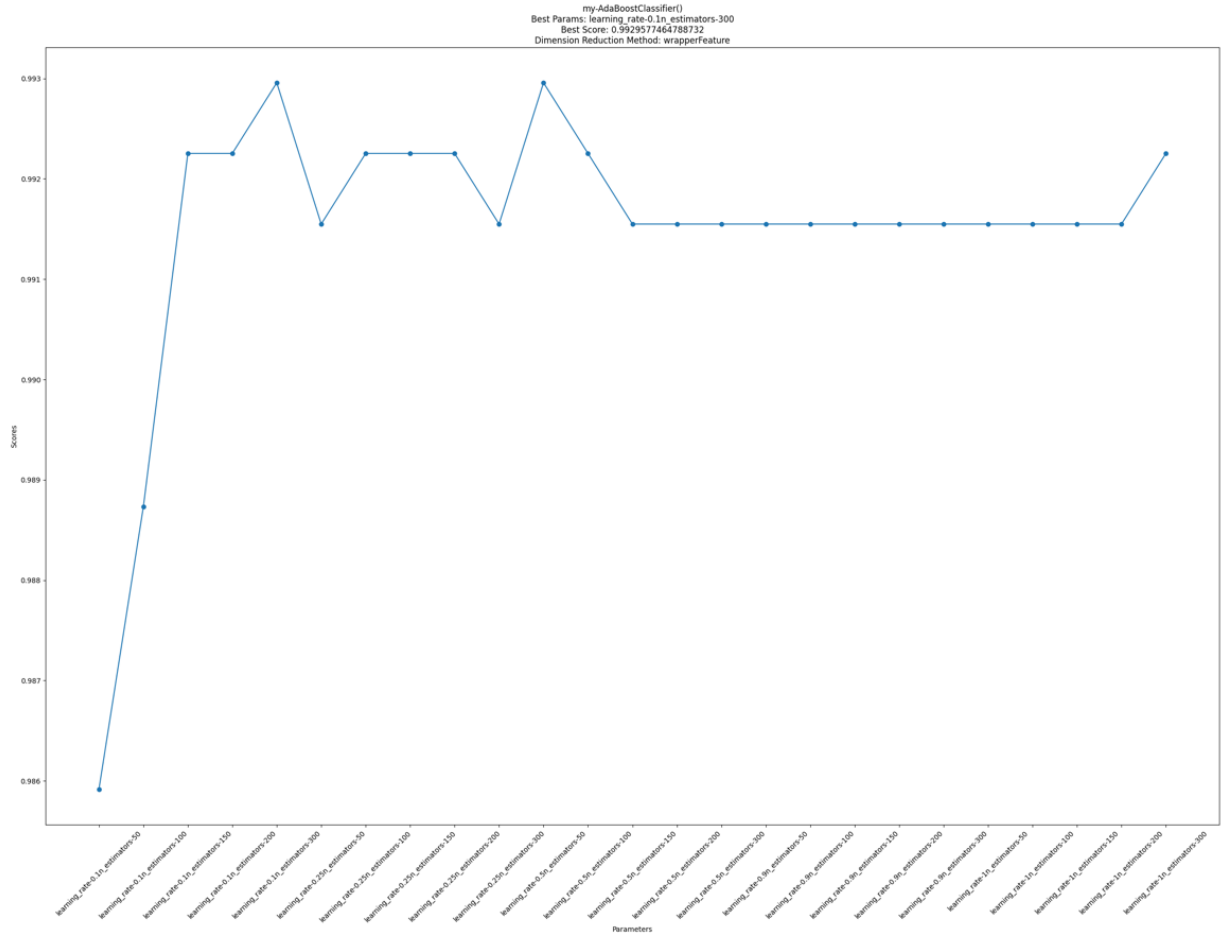
MY – AdaBoost Classifier – Simple Quality Filtering



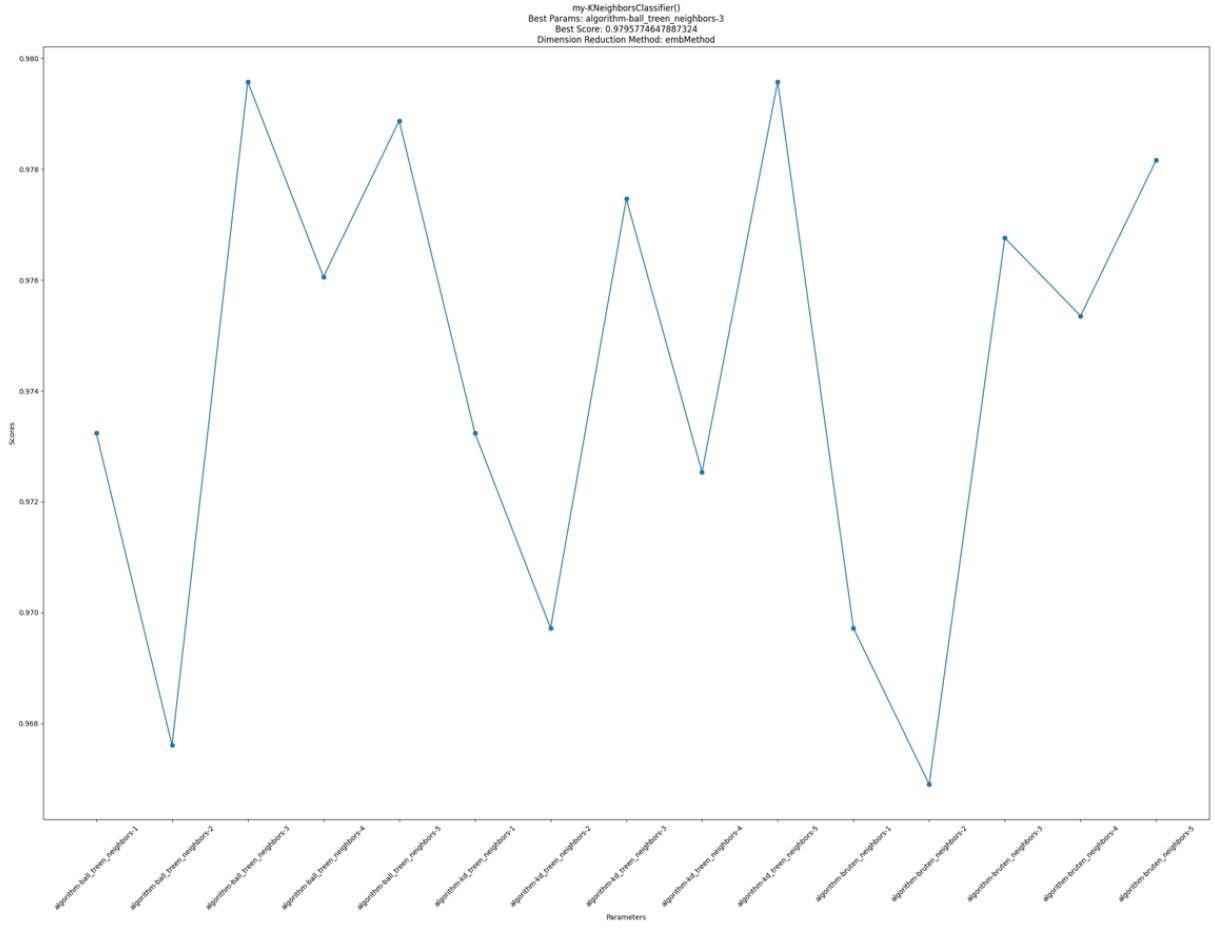
MY – AdaBoost Classifier – Filter Methods



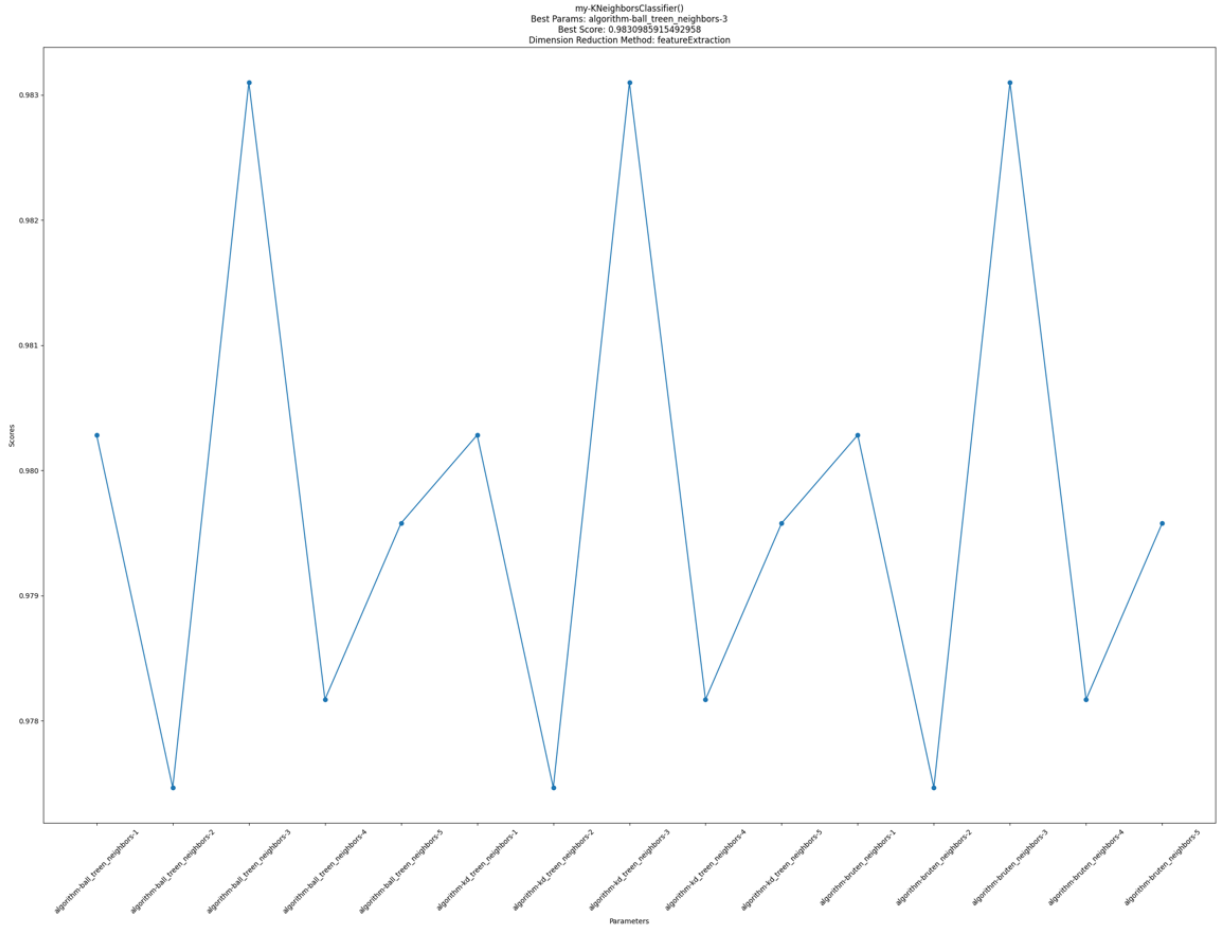
MY – AdaBoost Classifier – Wrapper Feature Selection



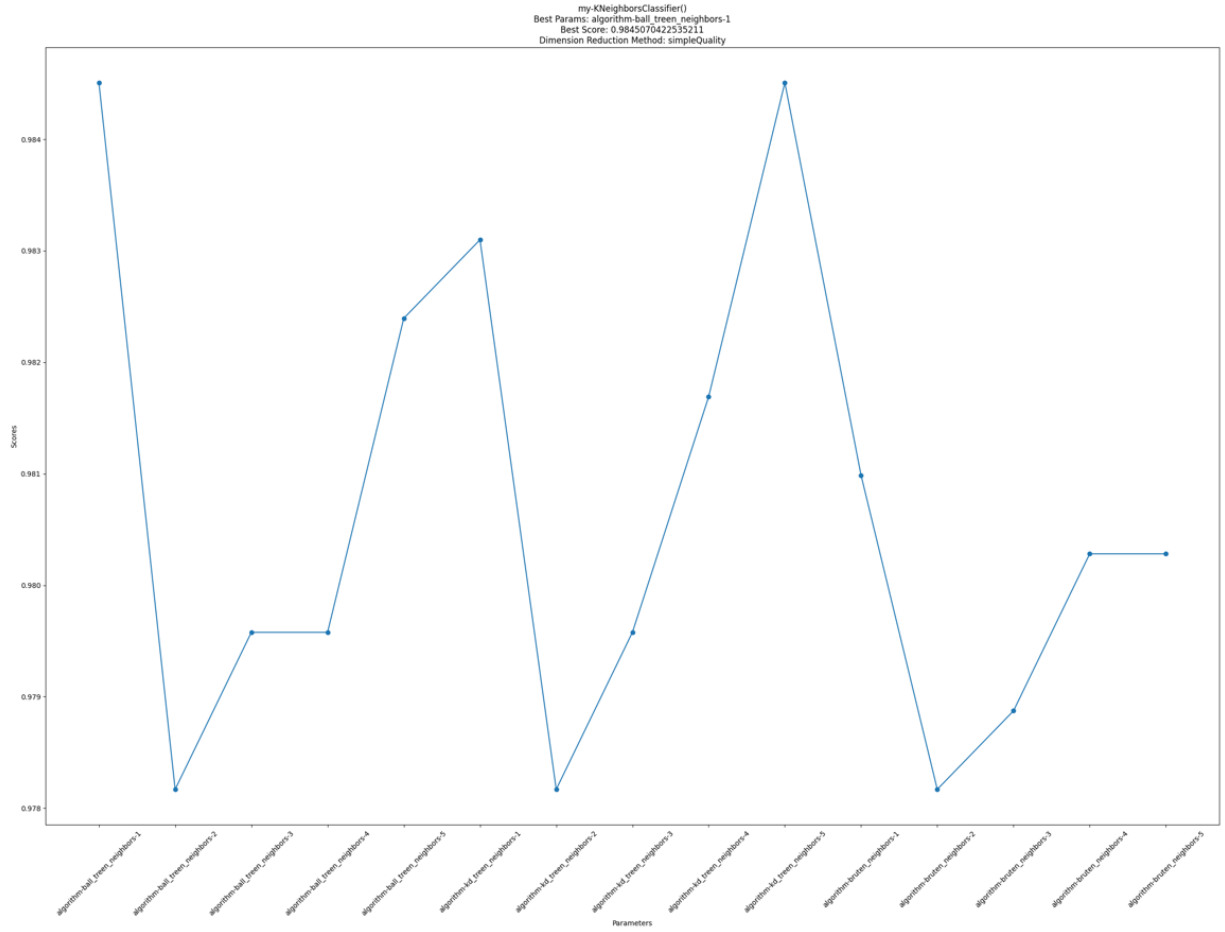
MY – K Neighbors Classifier – Embedded Methods



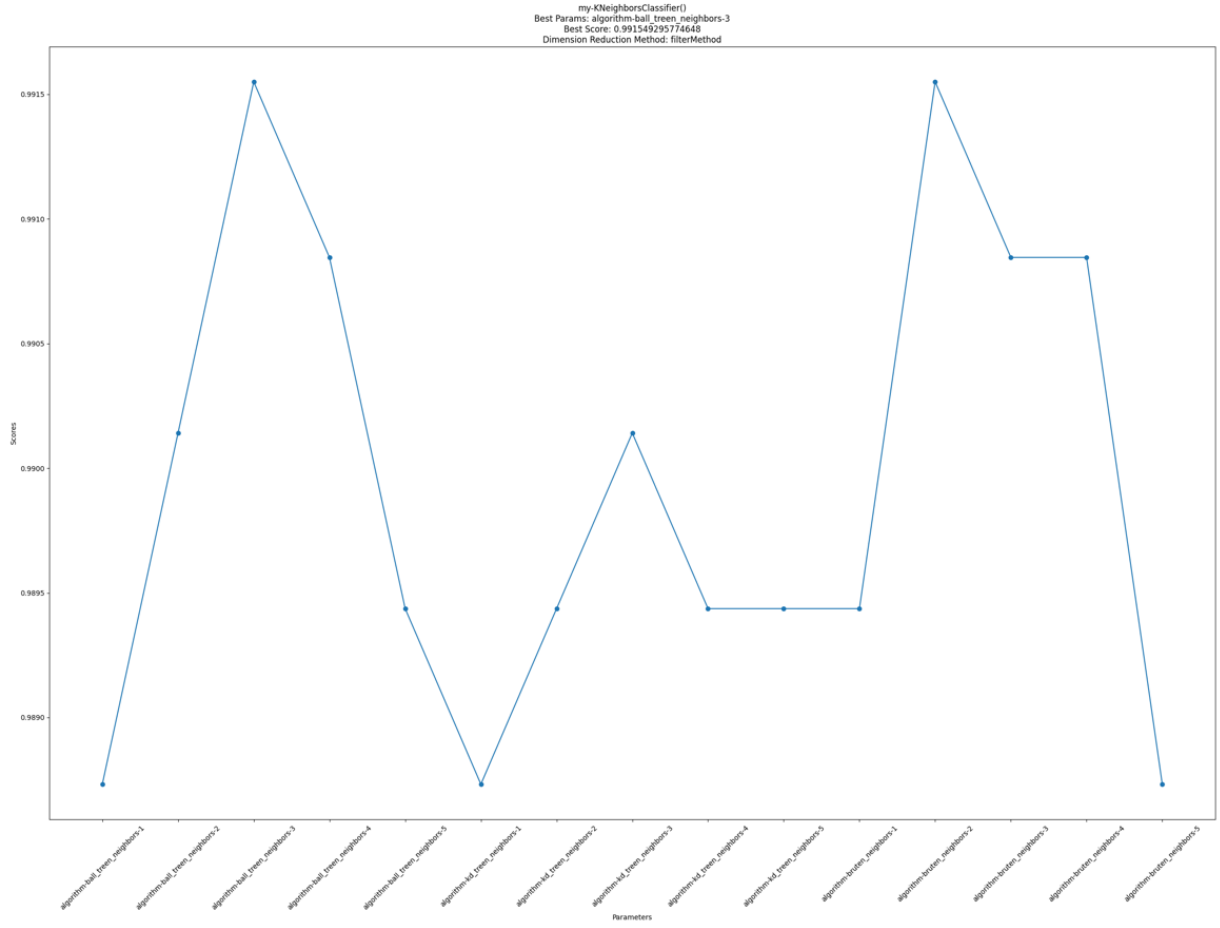
MY – K Neighbors Classifier – Feature Extraction



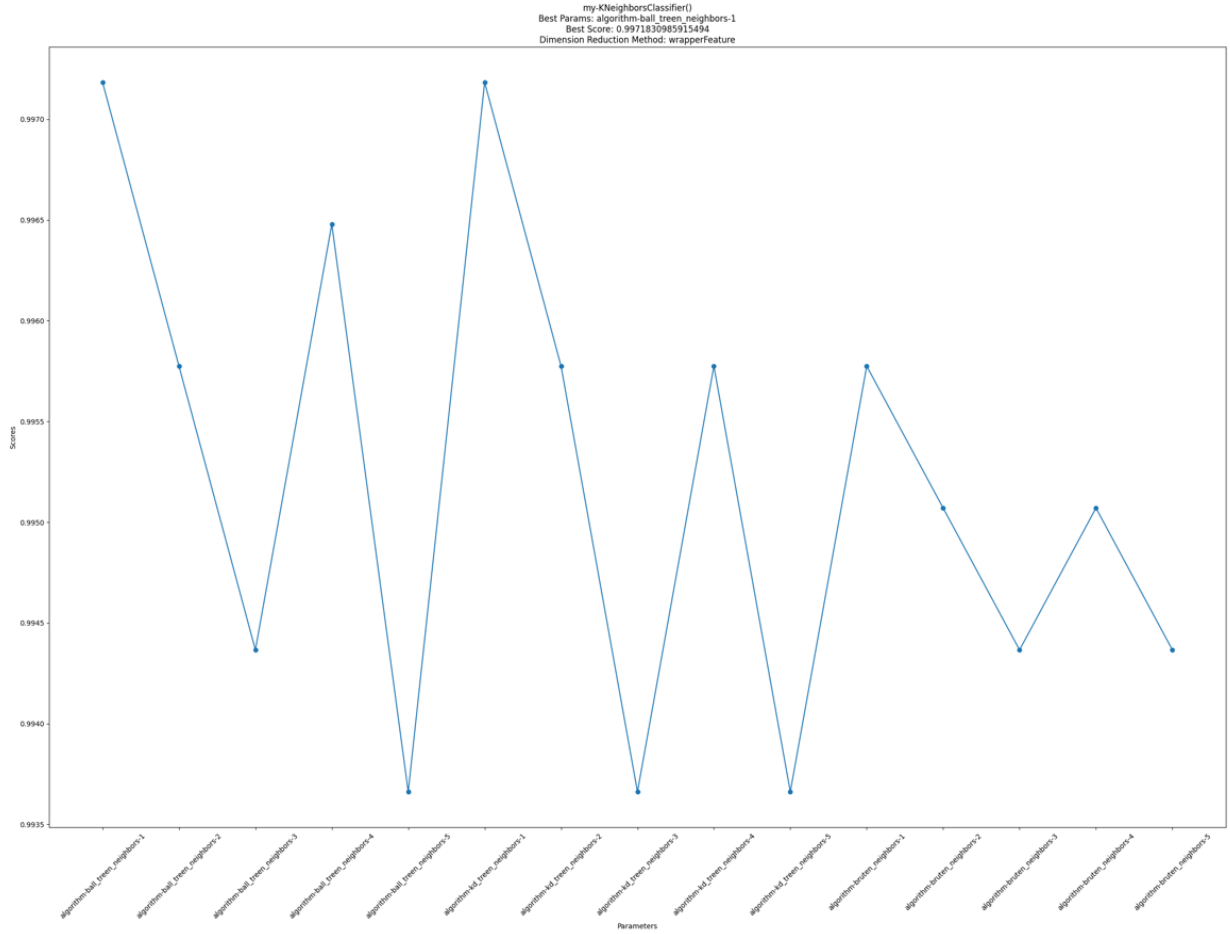
MY – K Neighbors Classifier – Simple Quality Filtering



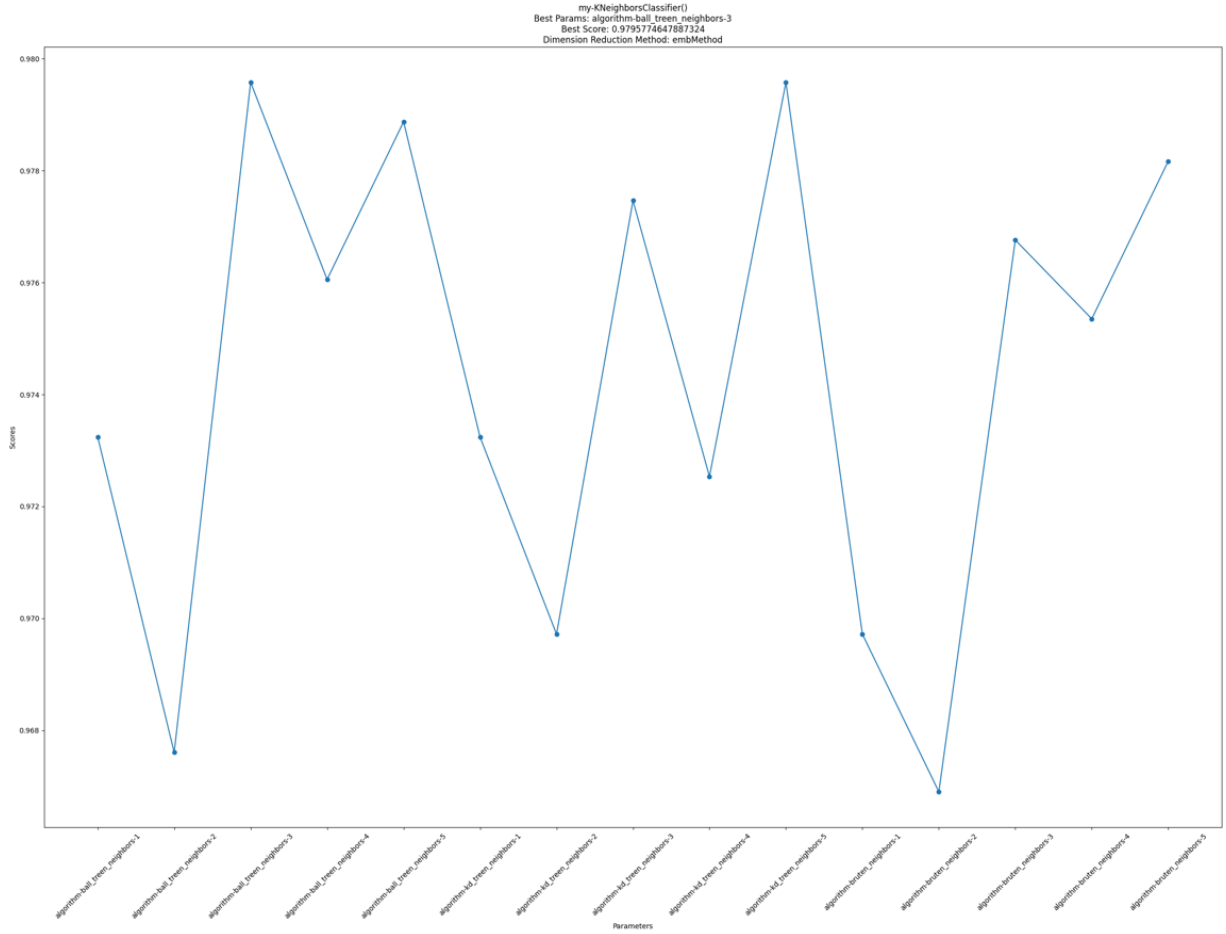
MY – K Neighbors Classifier – Filter Methods



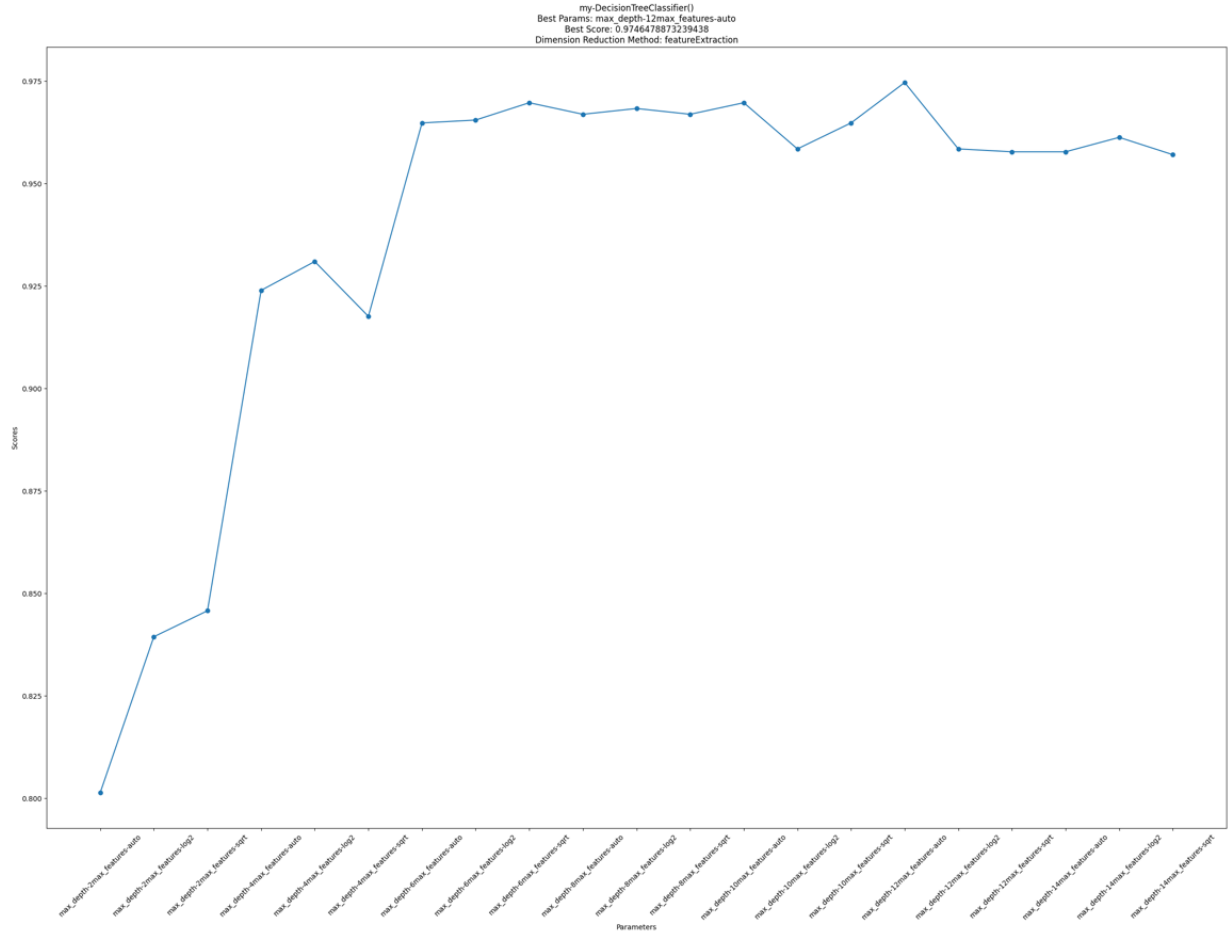
MY – K Neighbors Classifier – Wrapper Feature Selection



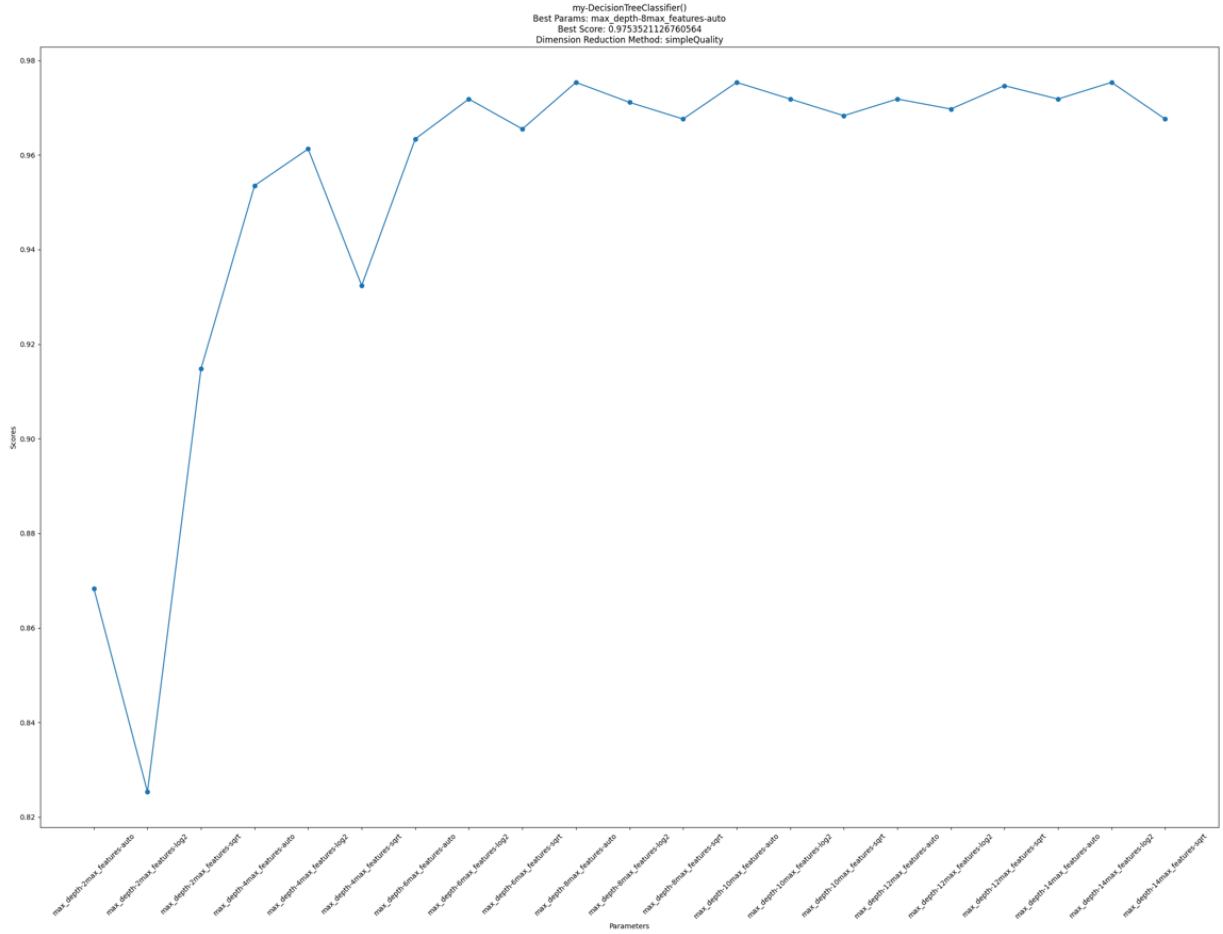
MY – Decision Tree Classifier – Embedded Methods



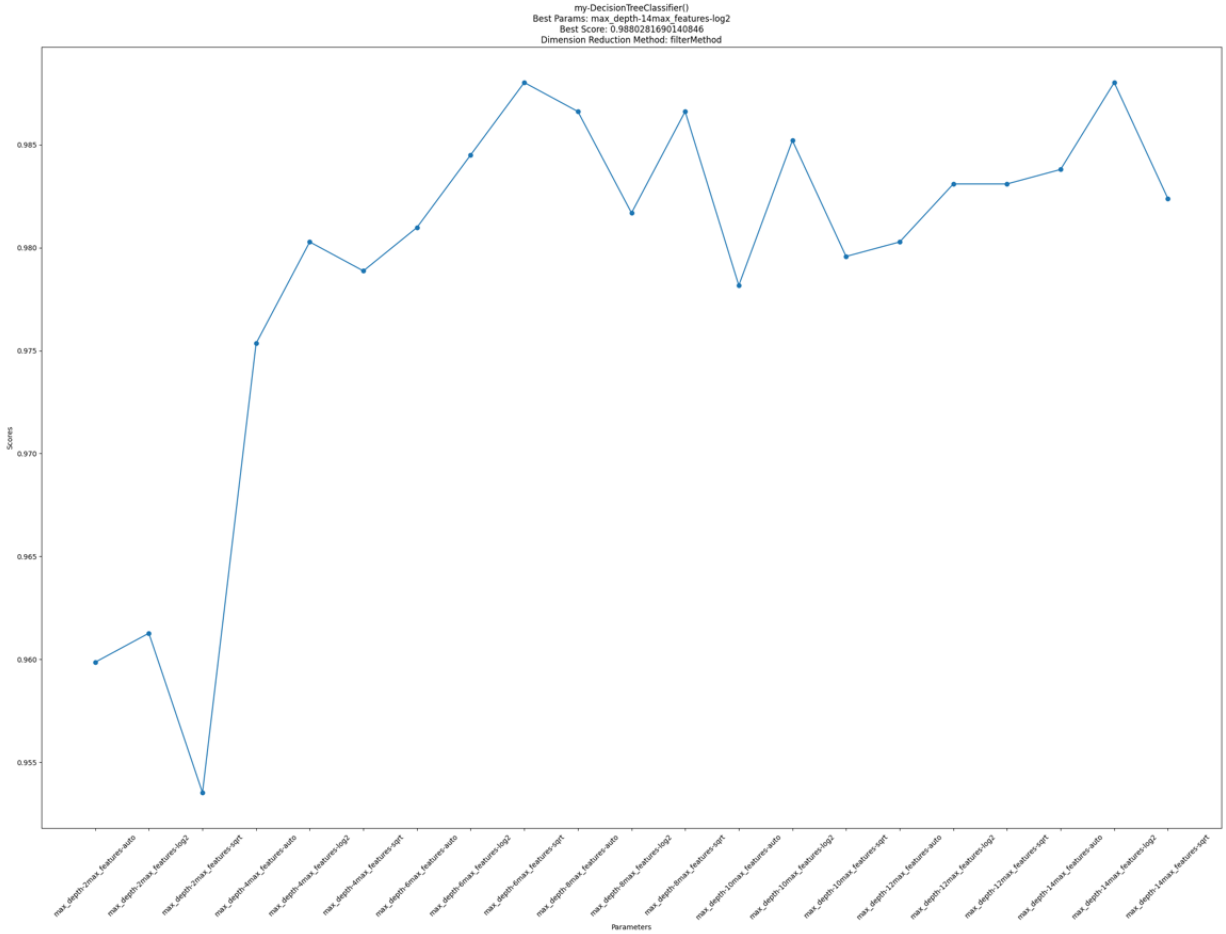
MY – Decision Tree Classifier – Feature Extraction



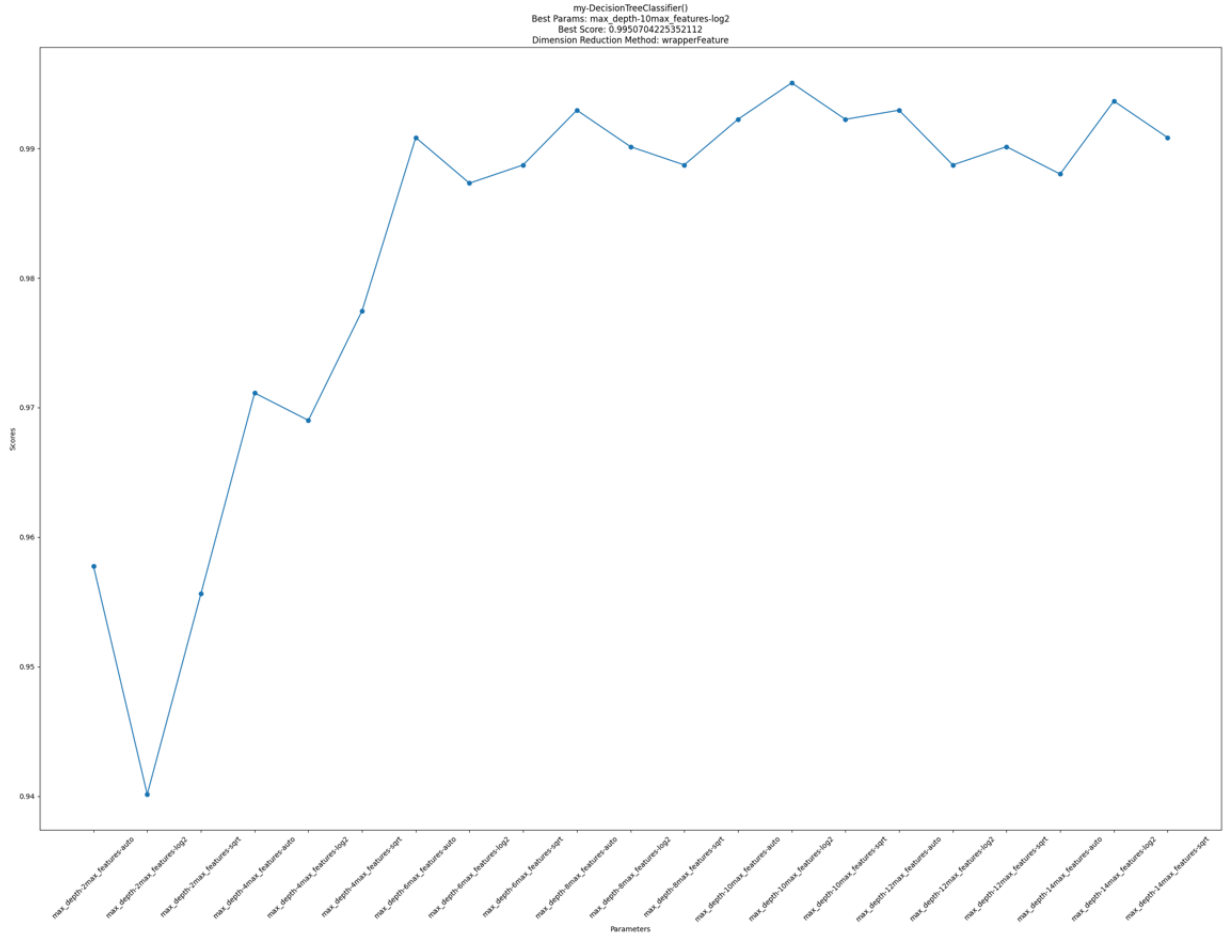
MY – Decision Tree Classifier – Simple Quality Filtering



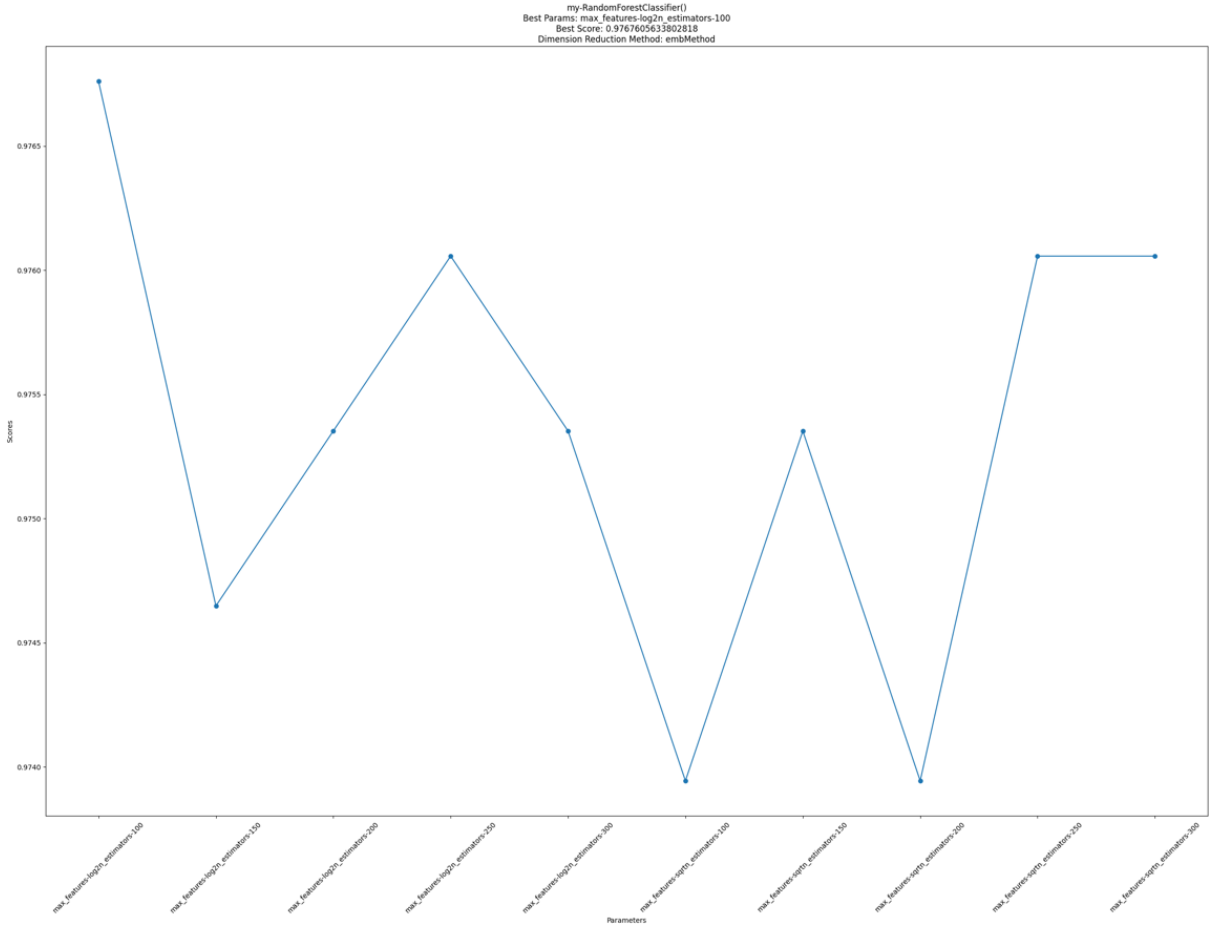
MY – Decision Tree Classifier – Filter Methods



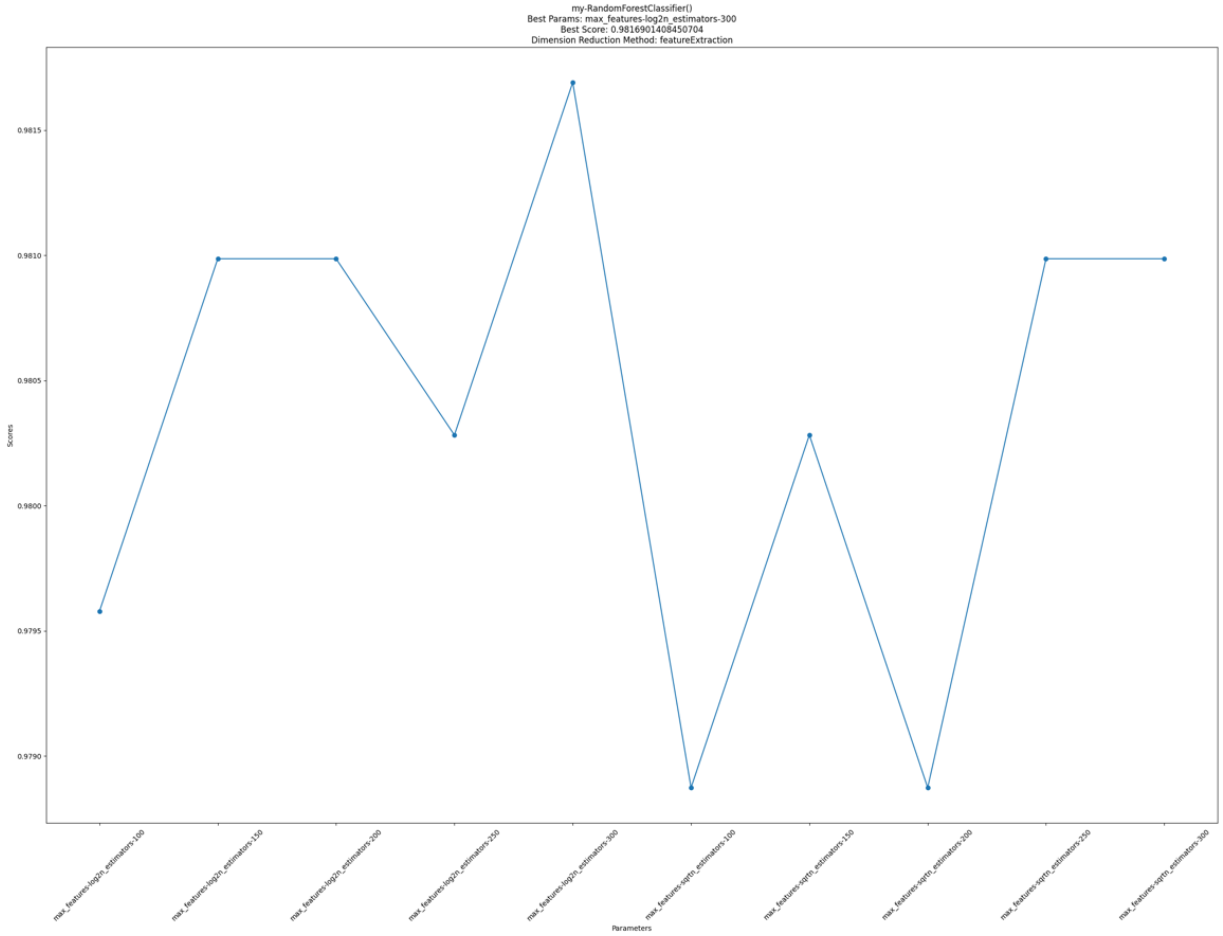
MY – Decision Tree Classifier – Wrapper Feature Selection



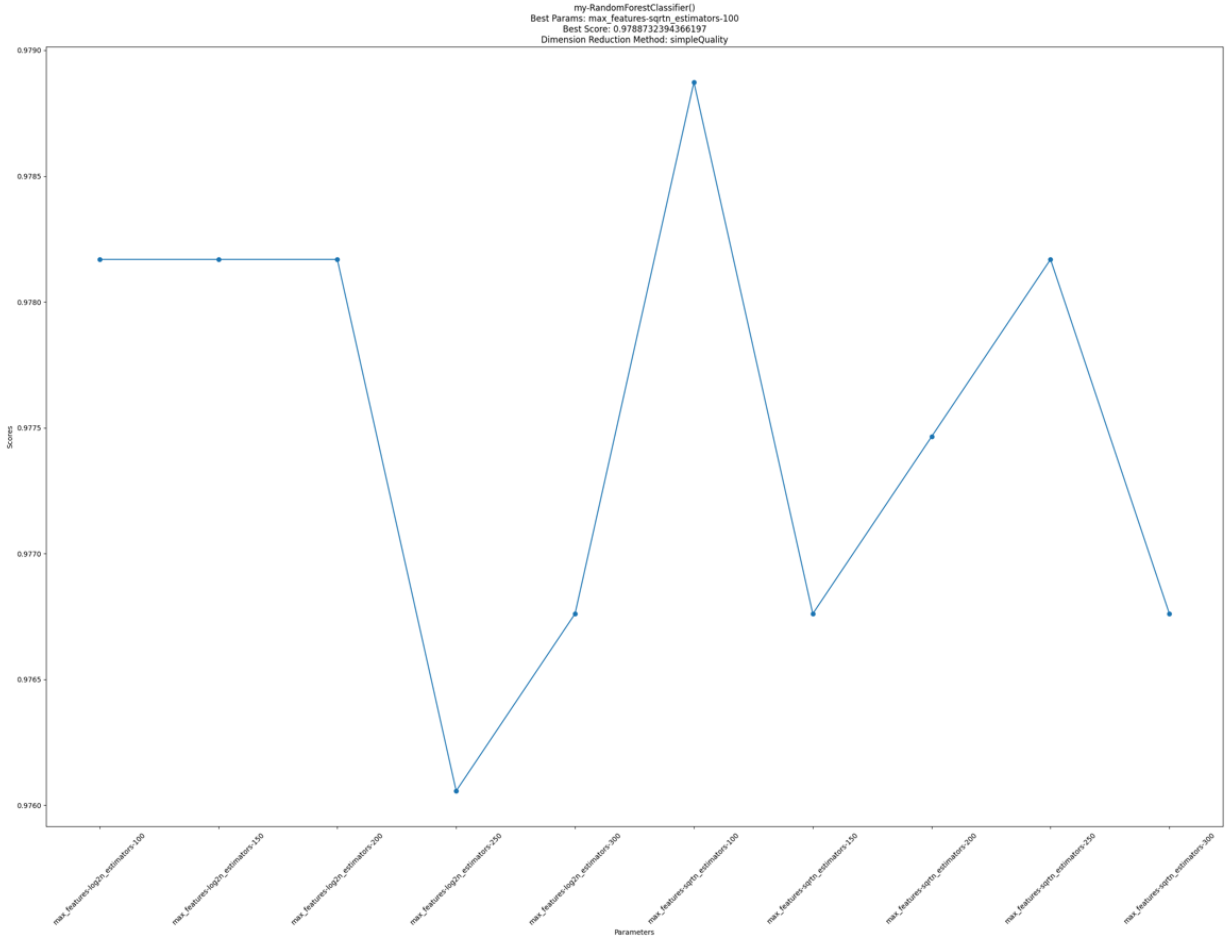
MY – Random Forest Classifier – Embedded Methods



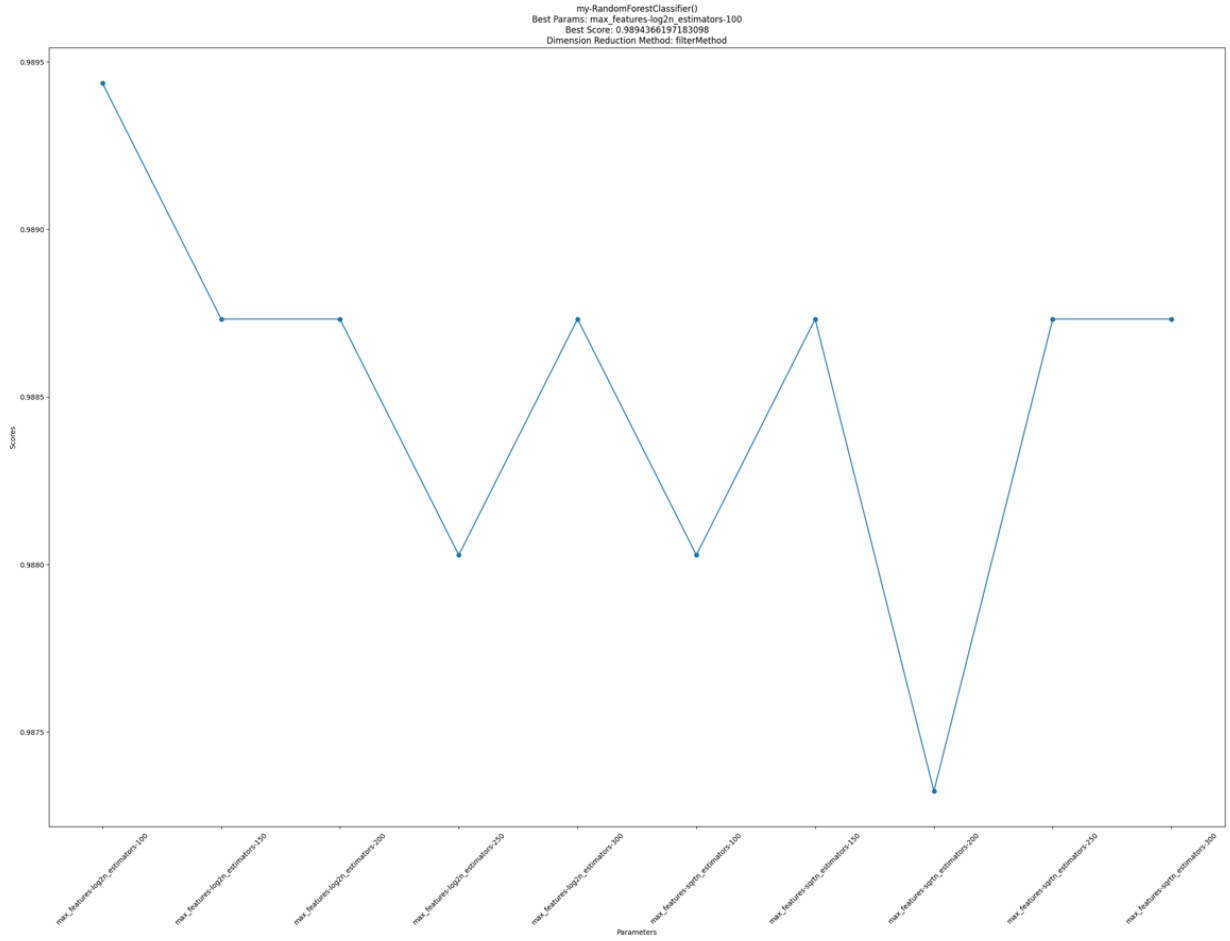
MY – Random Forest Classifier – Feature Extraction



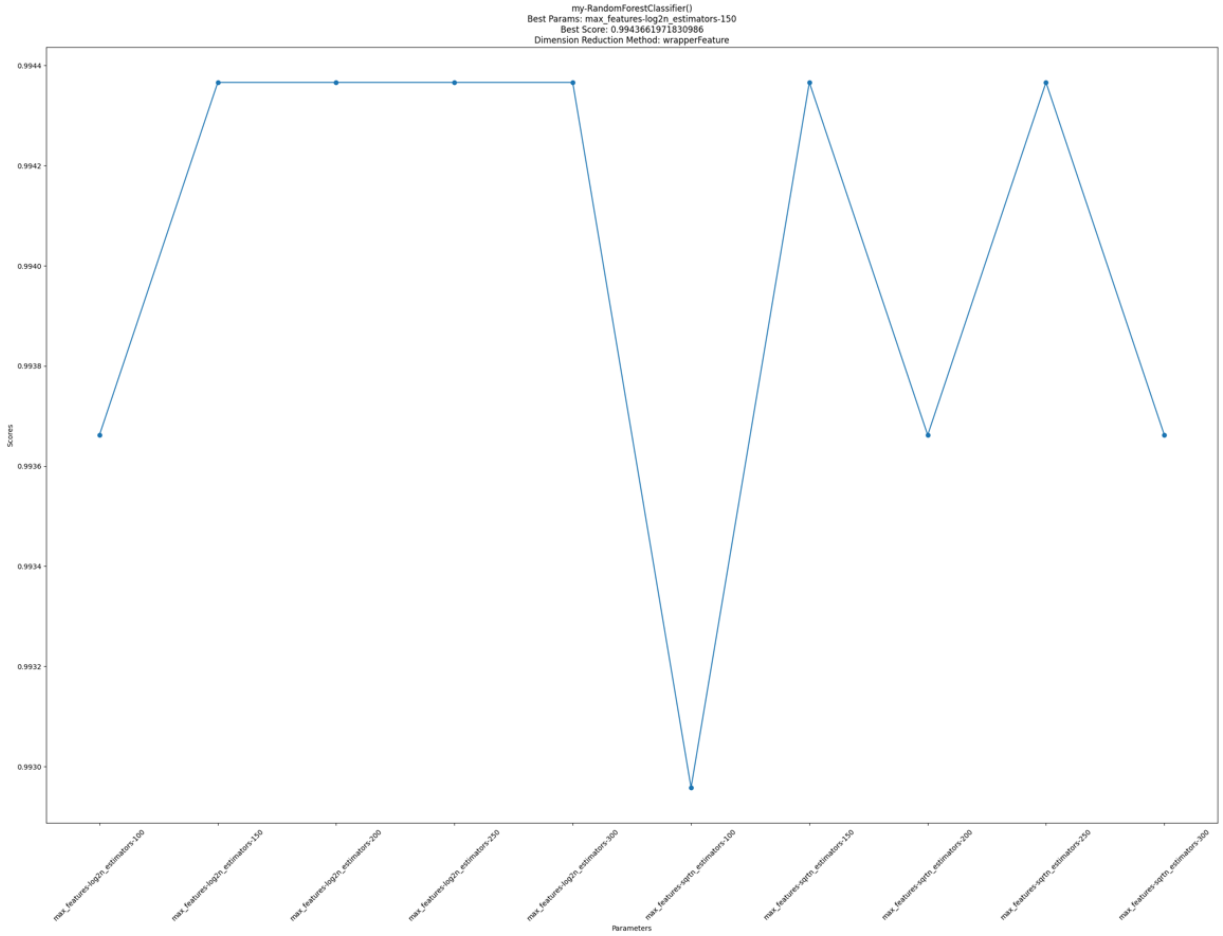
MY – Random Forest Classifier – Simple Quality Filtering



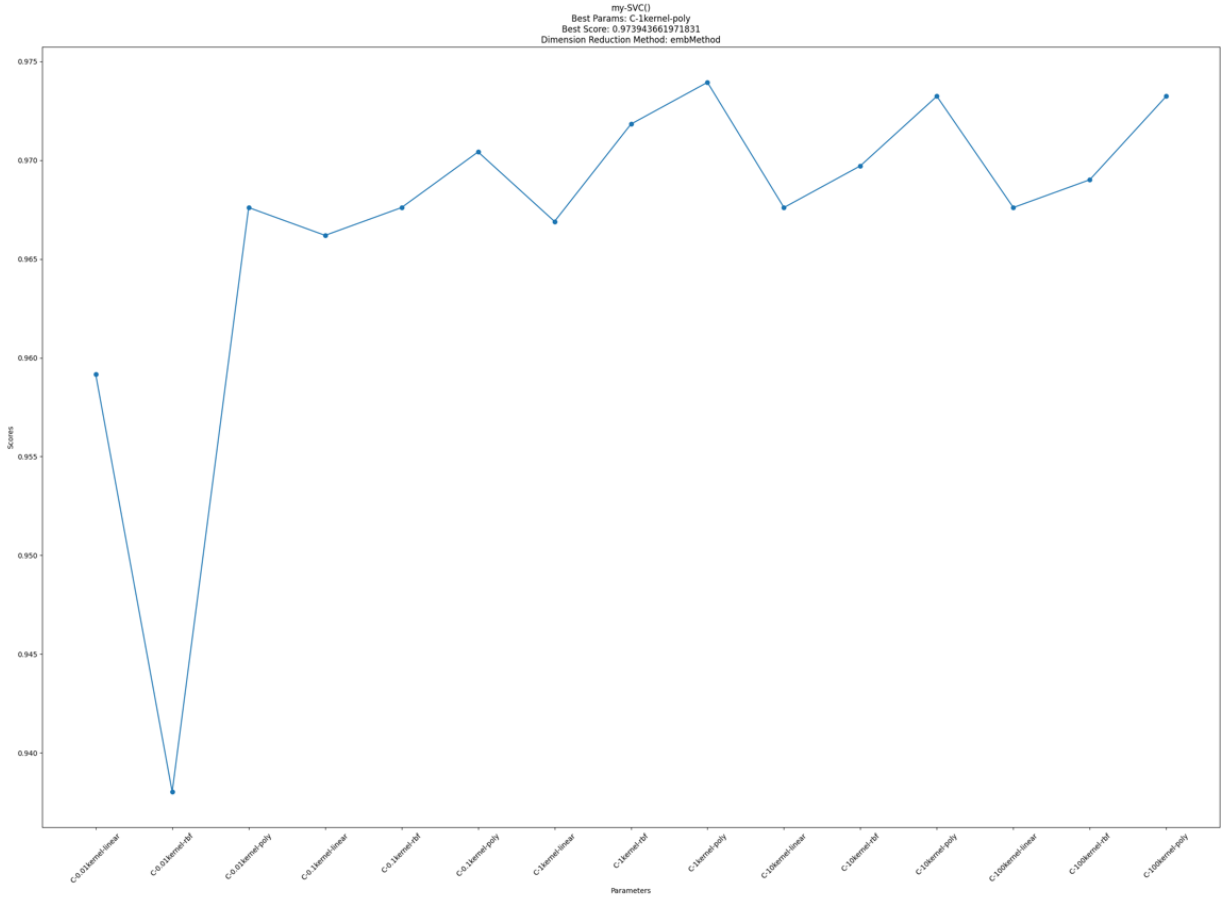
MY – Random Forest Classifier – Filter Methods



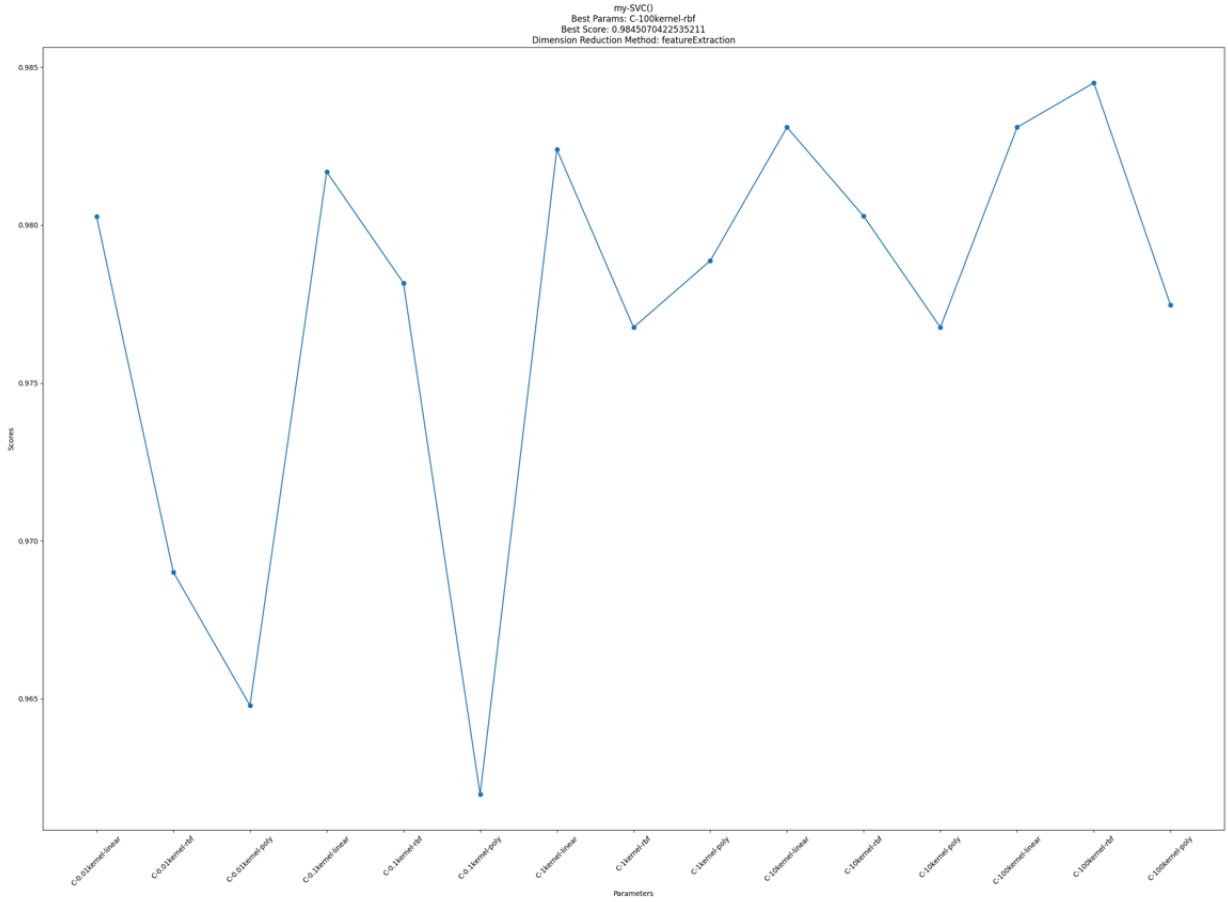
MY – Random Forest Classifier – Wrapper Feature Selection



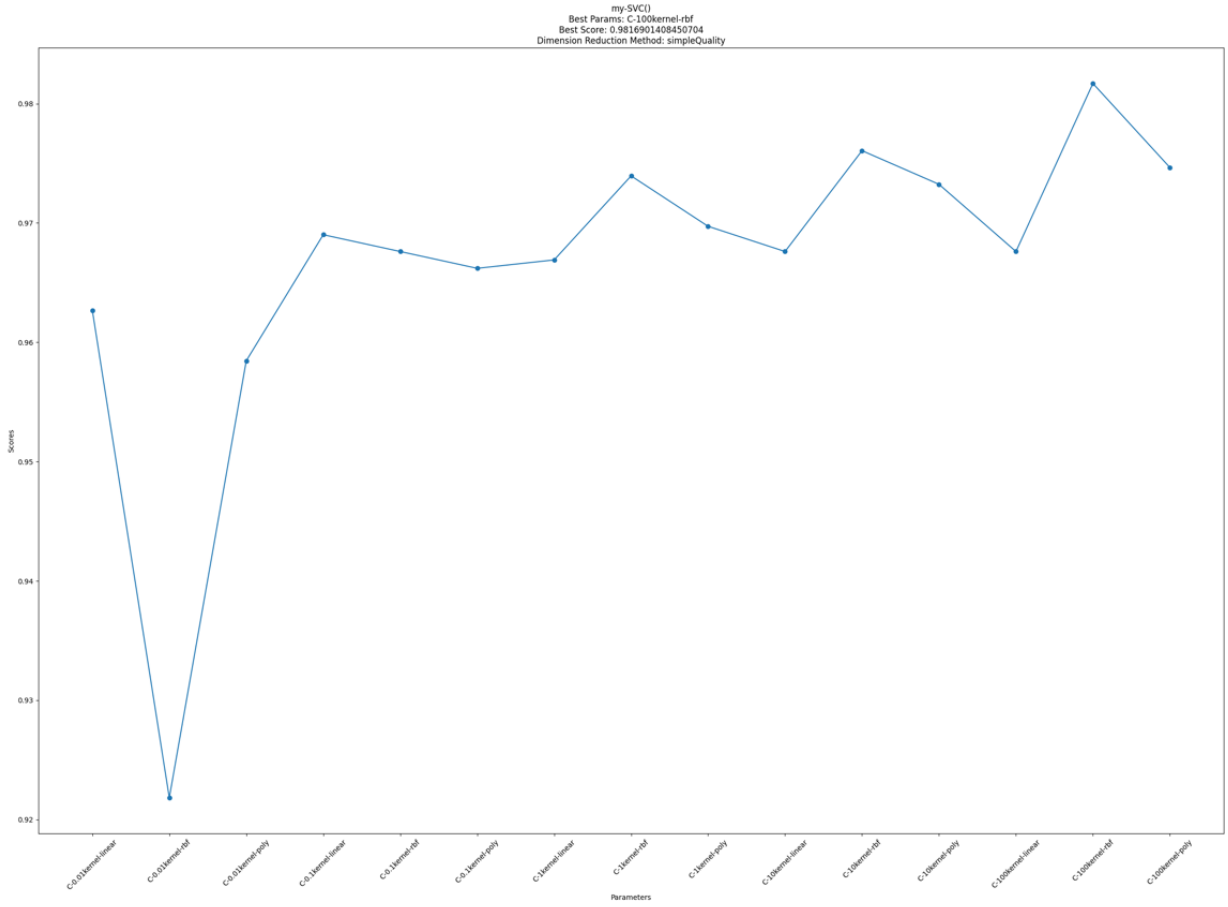
MY – SVM – Embedded Methods



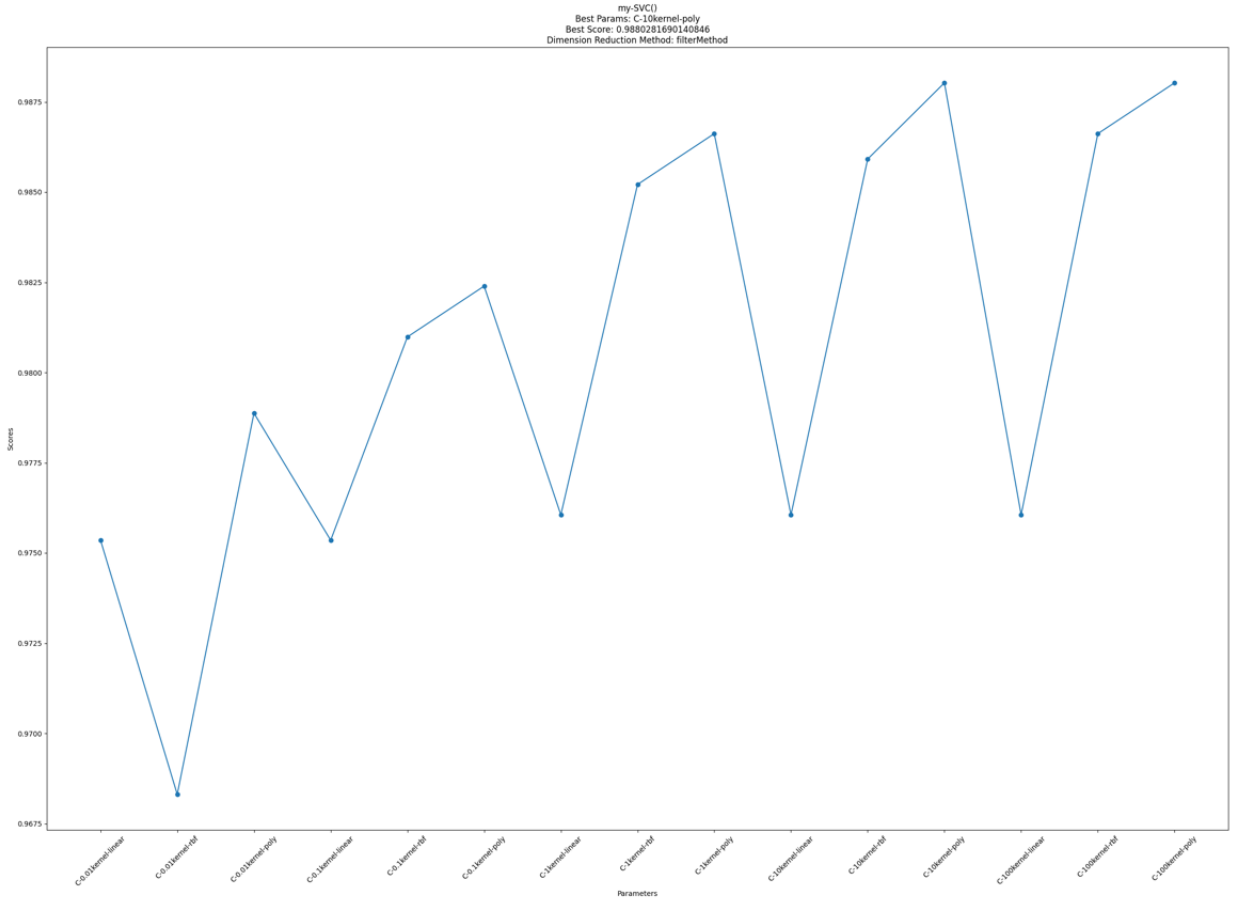
MY – SVM – Feature Extraction



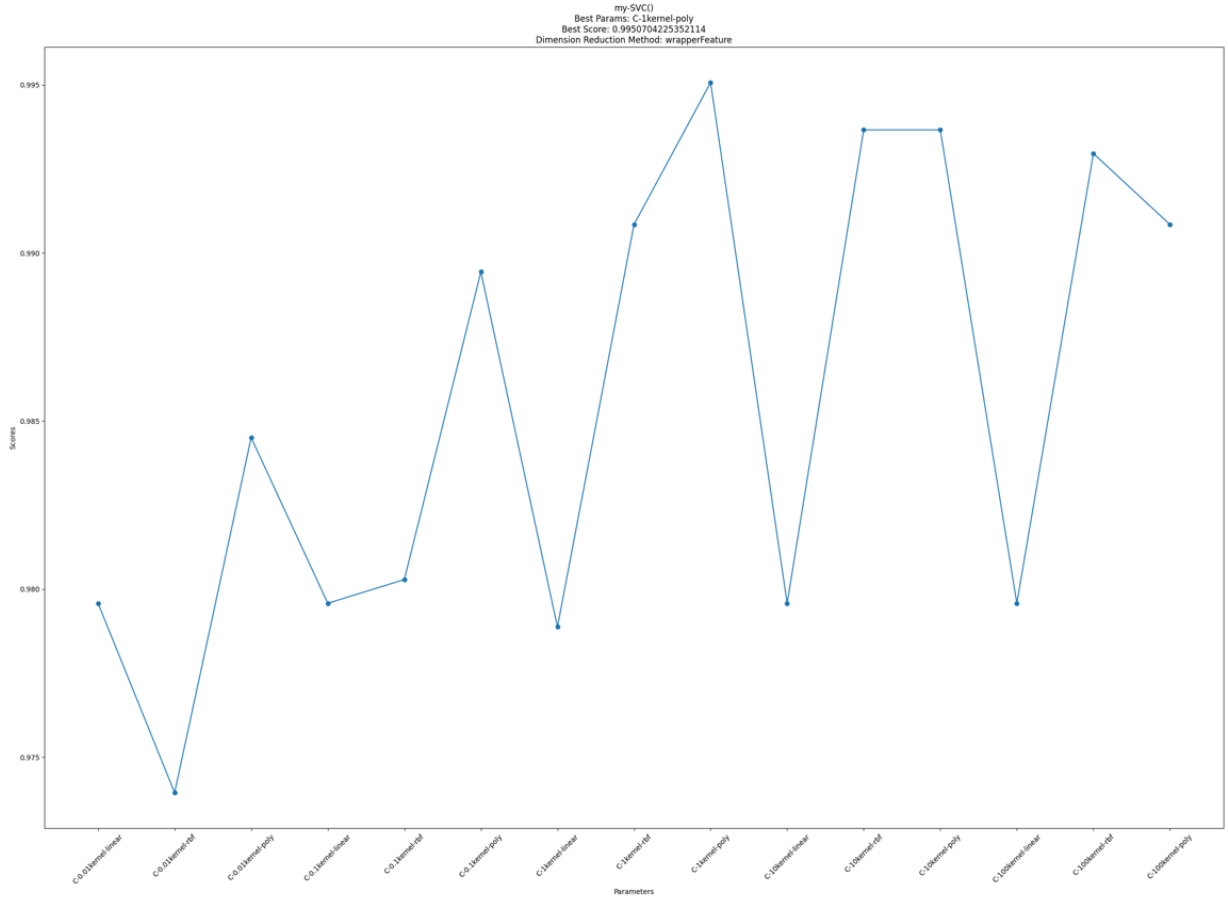
MY – SVM – Simple Quality Filtering



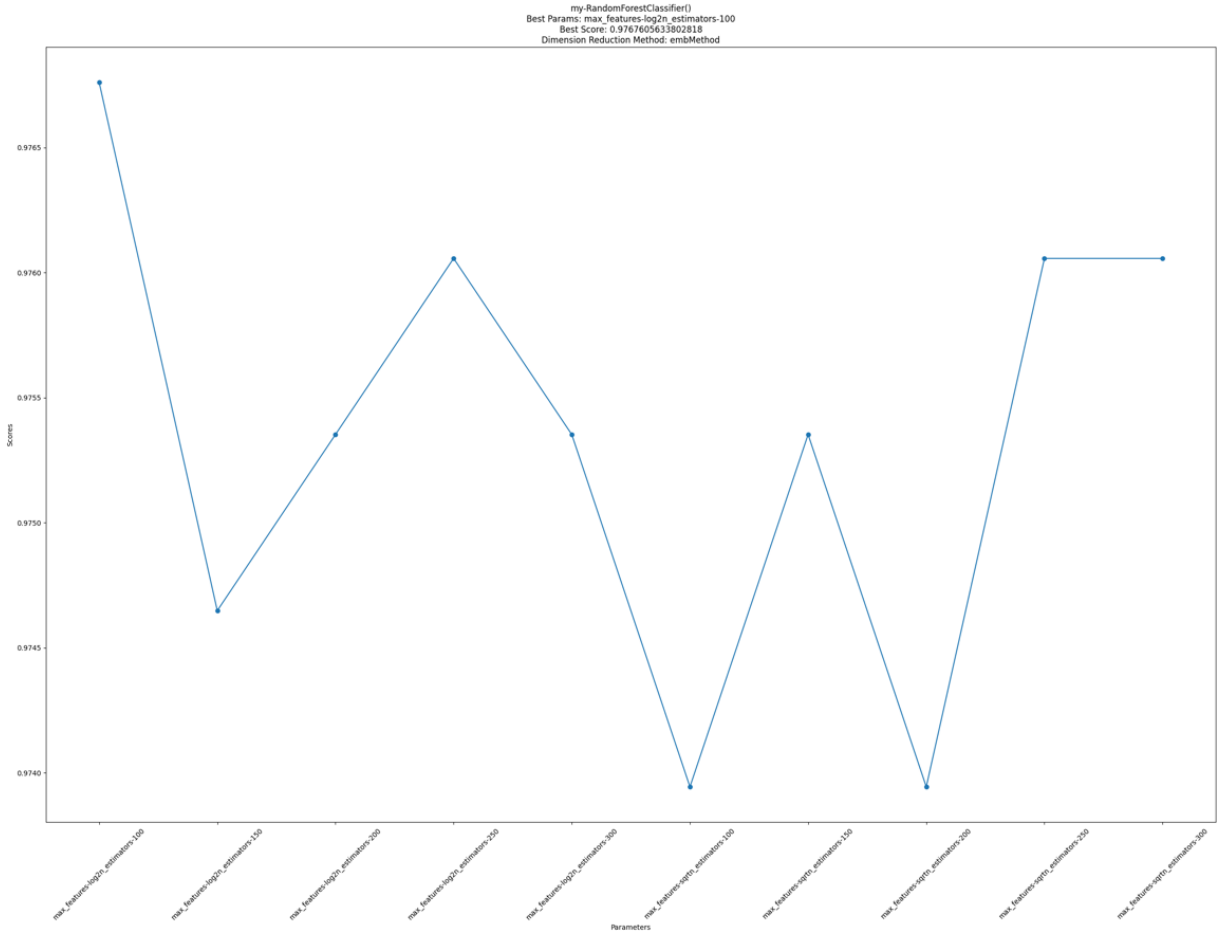
MY – SVM – Filter Methods



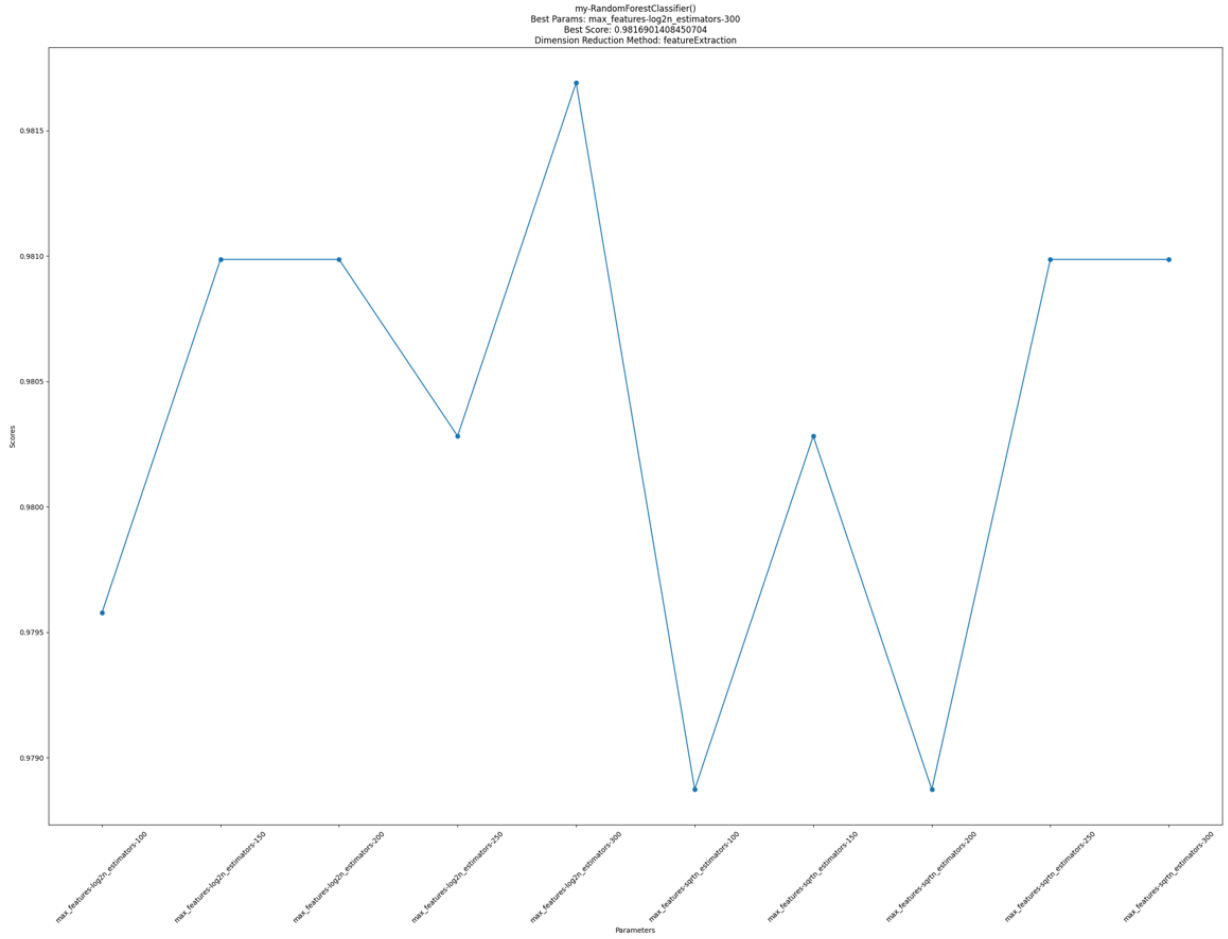
MY – SVM – Wrapper Feature Selection



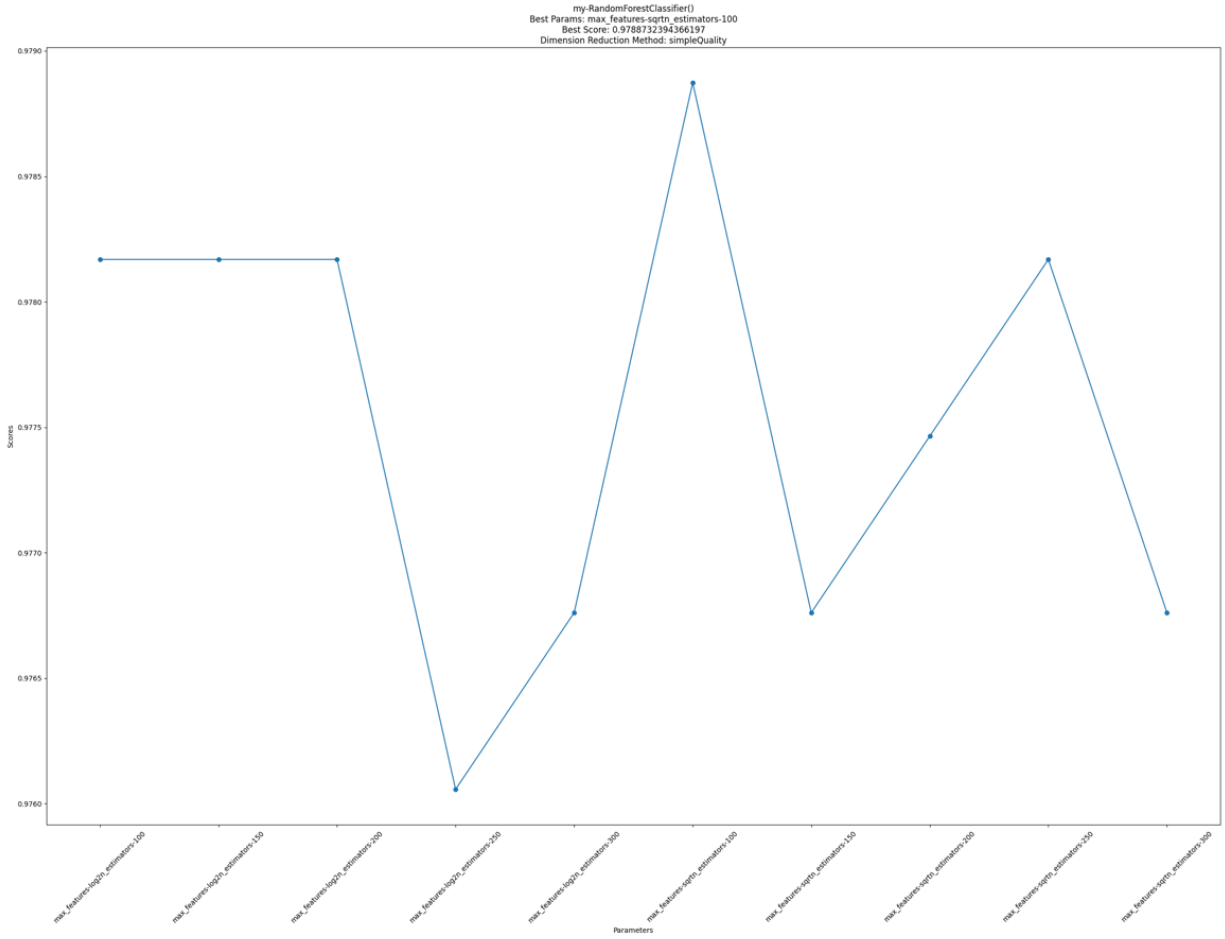
MY – MLP Classifier – Embedded Methods



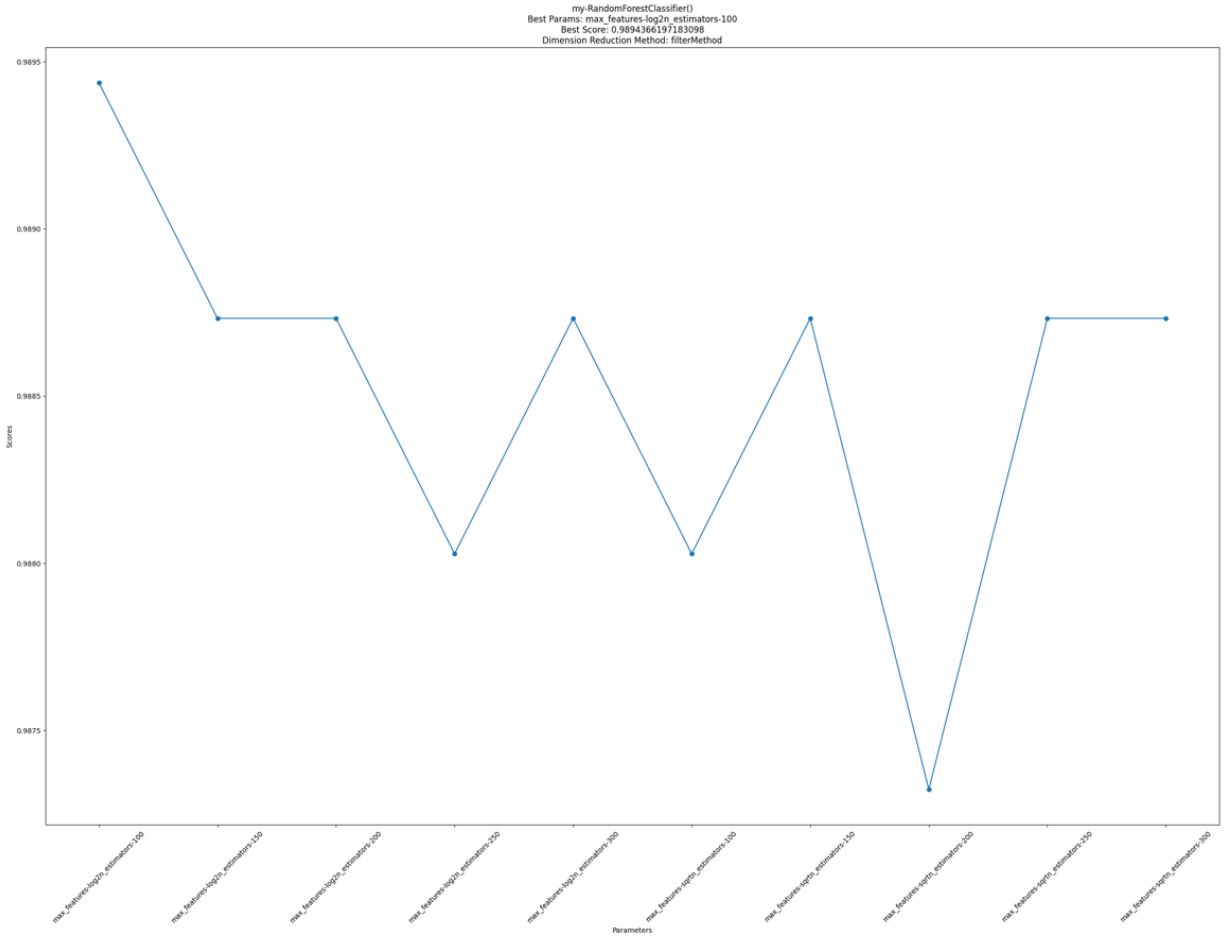
MY – MLP Classifier – Feature Extraction



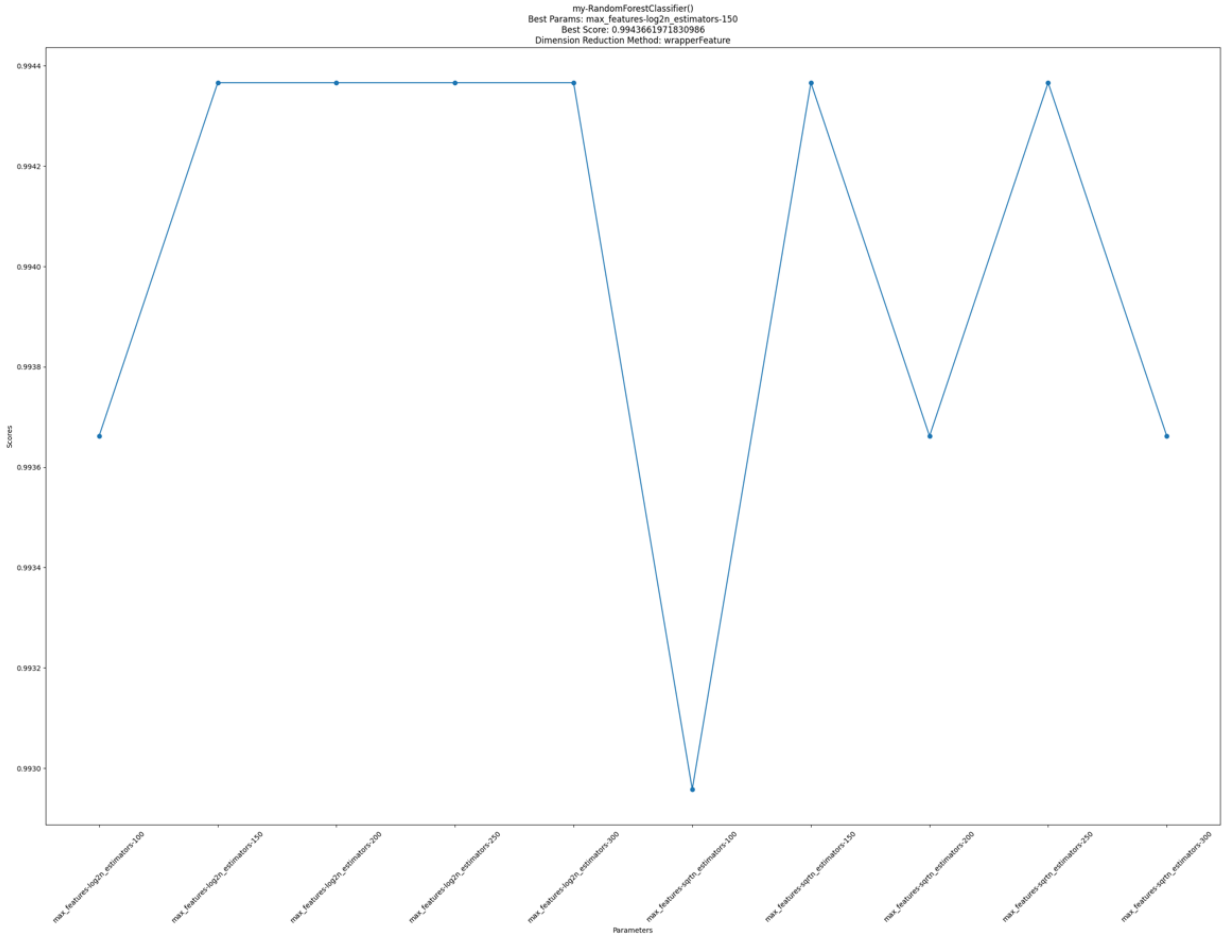
MY – MLP Classifier – Simple Quality Filtering



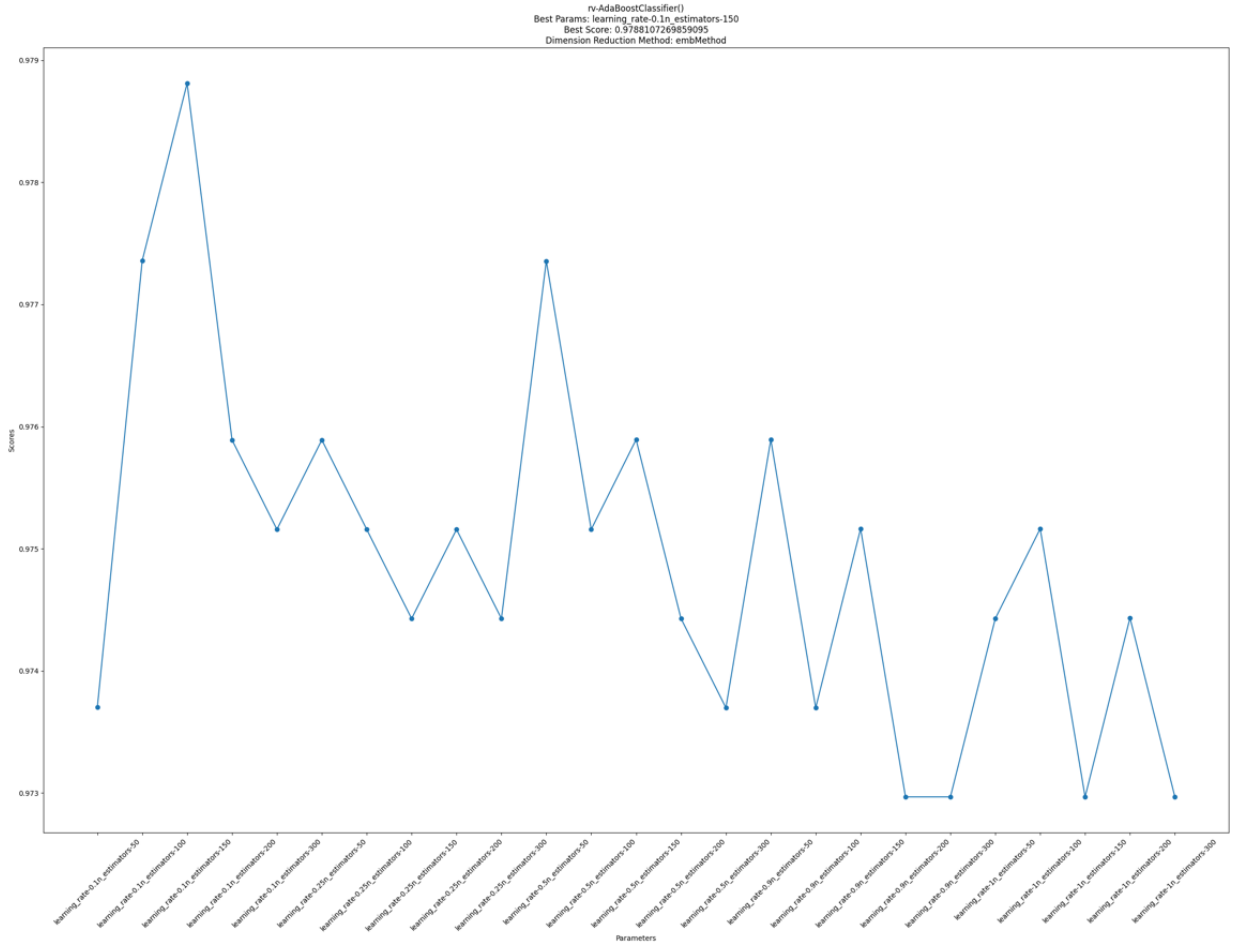
MY – MLP Classifier – Filter Methods



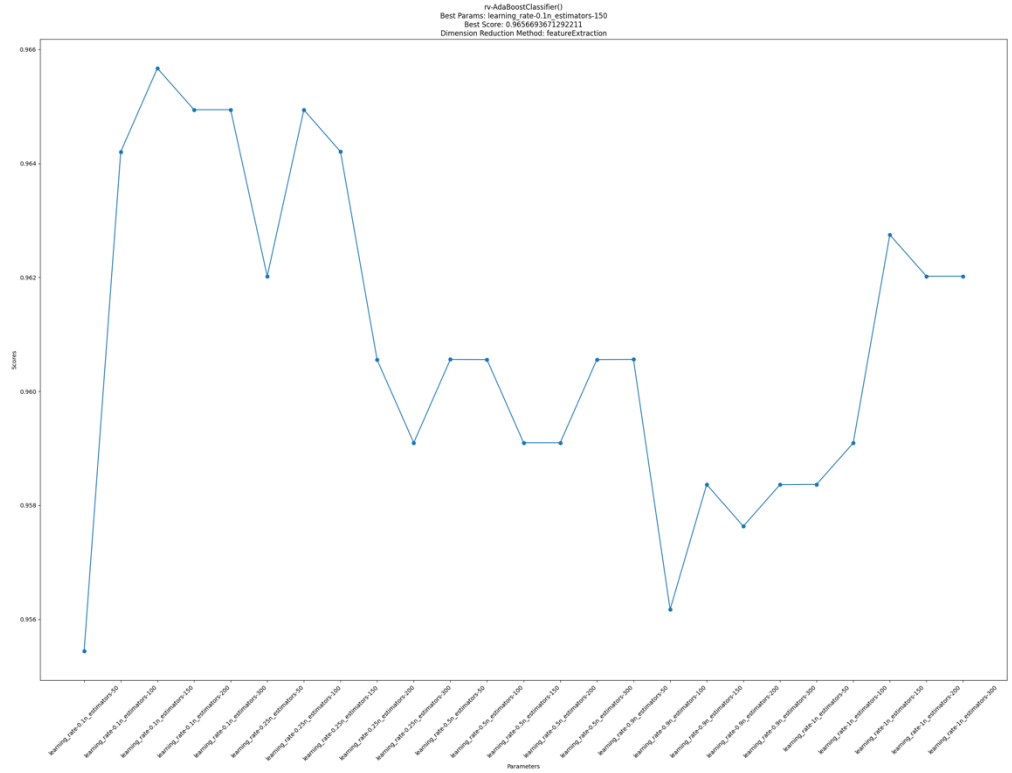
MY – MLP Classifier – Wrapper Feature Selection



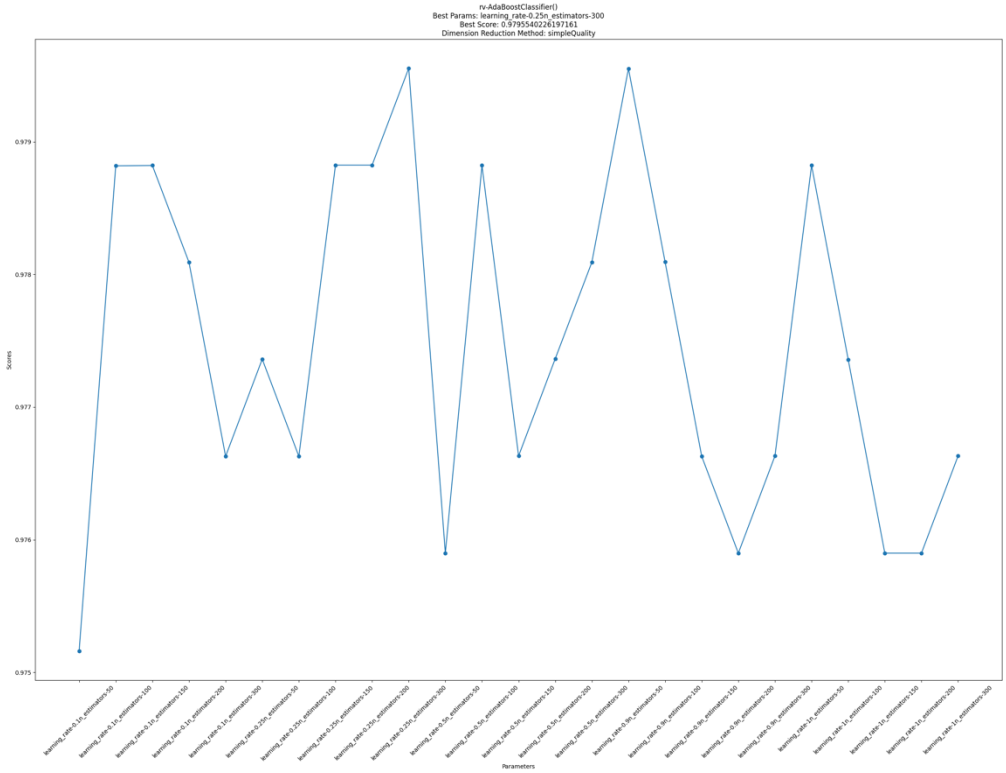
RV – AdaBoost Classifier – Embedded Methods



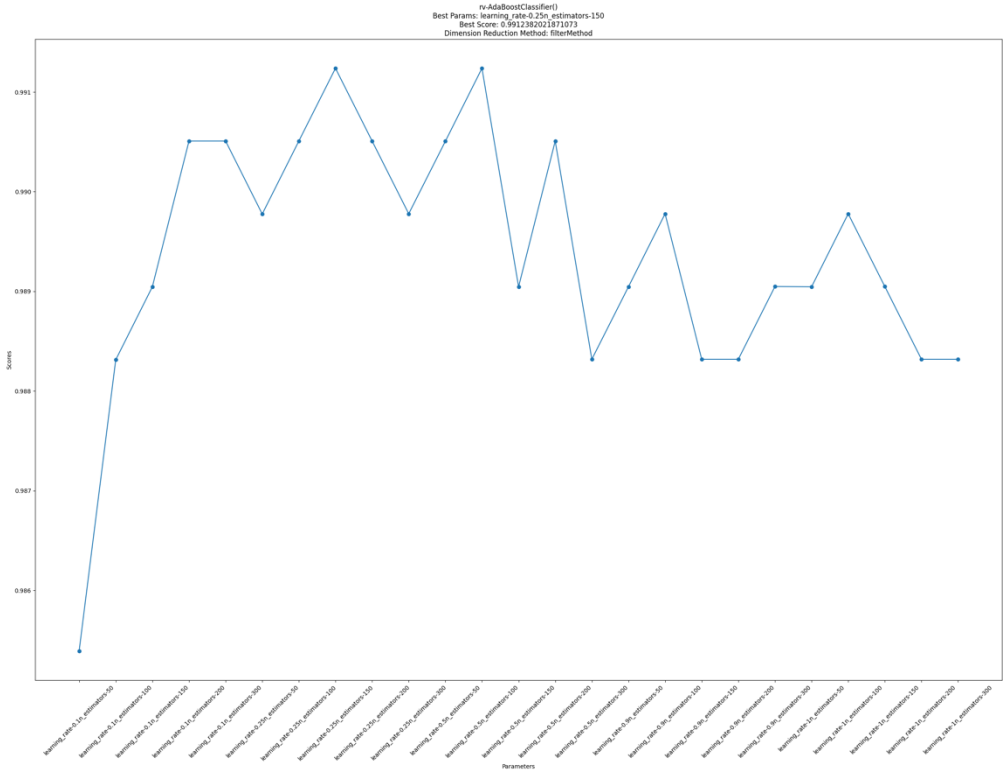
RV – AdaBoost Classifier – Feature Extraction



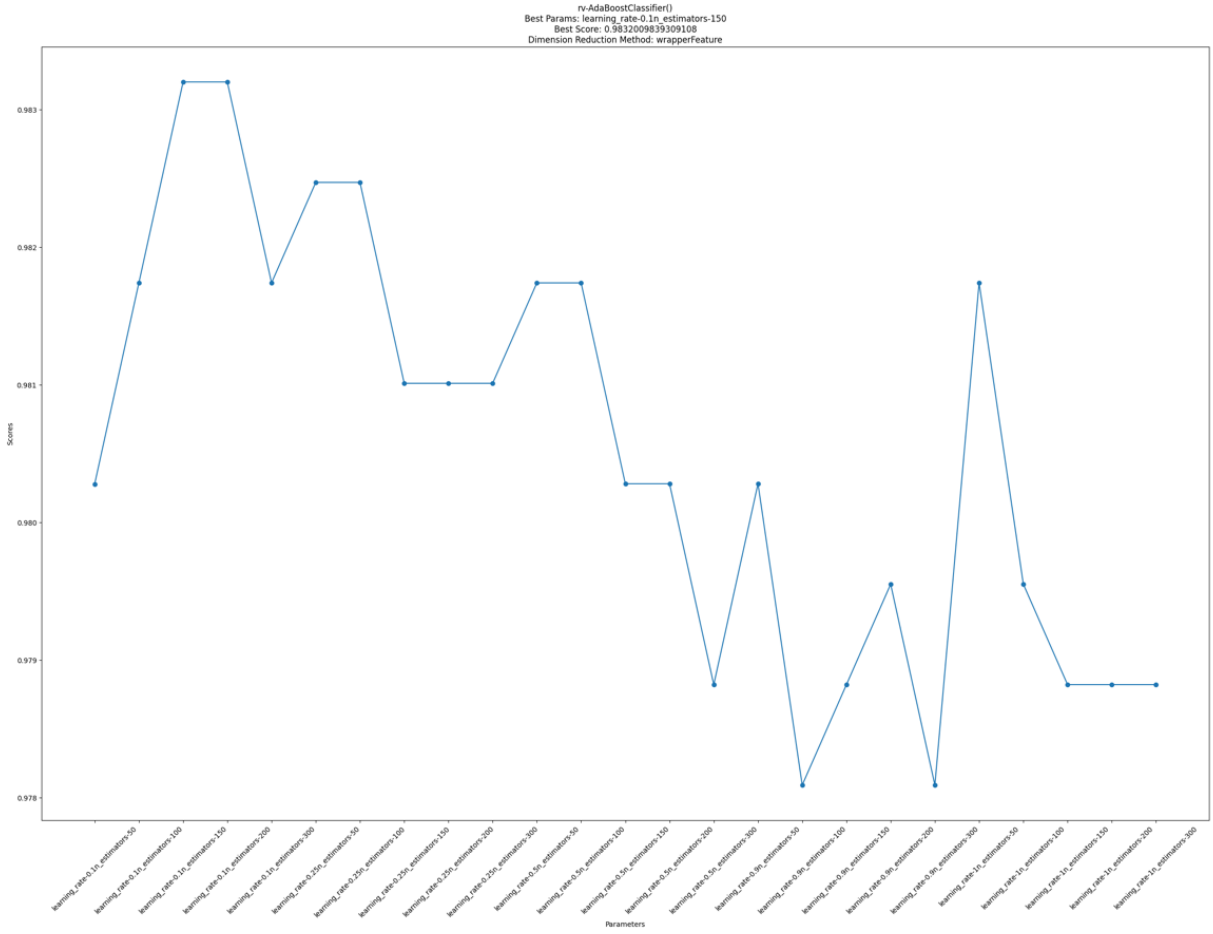
RV – AdaBoost Classifier – Simple Quality Filtering



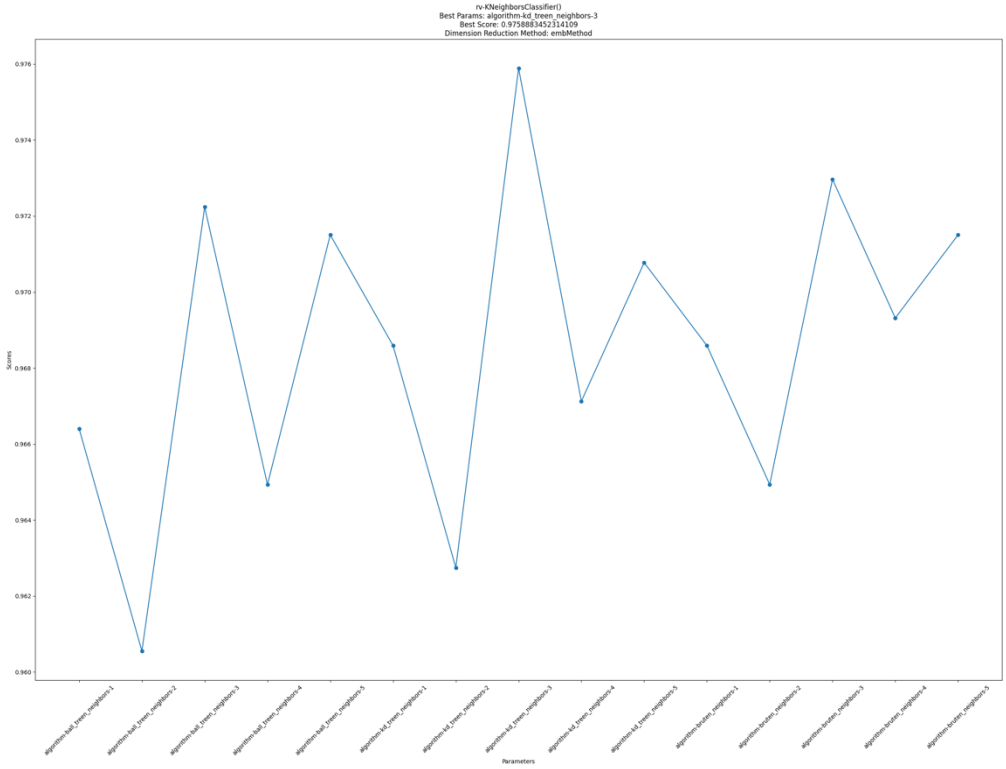
RV – AdaBoost Classifier – Filter Methods



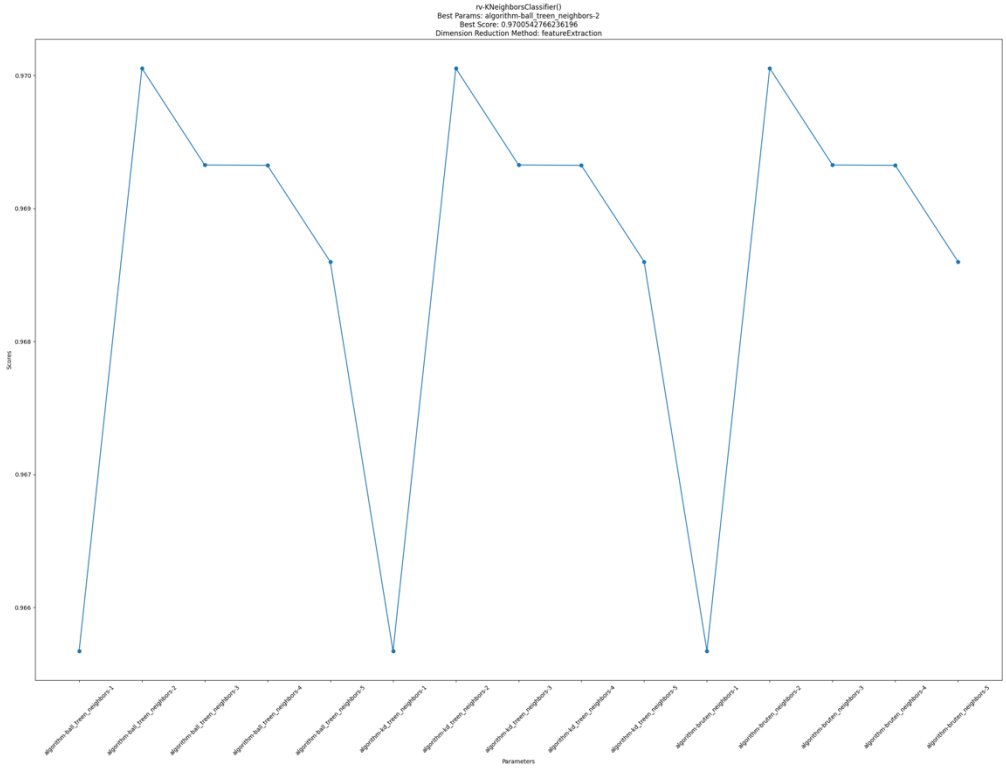
RV – AdaBoost Classifier – Wrapper Feature Selection



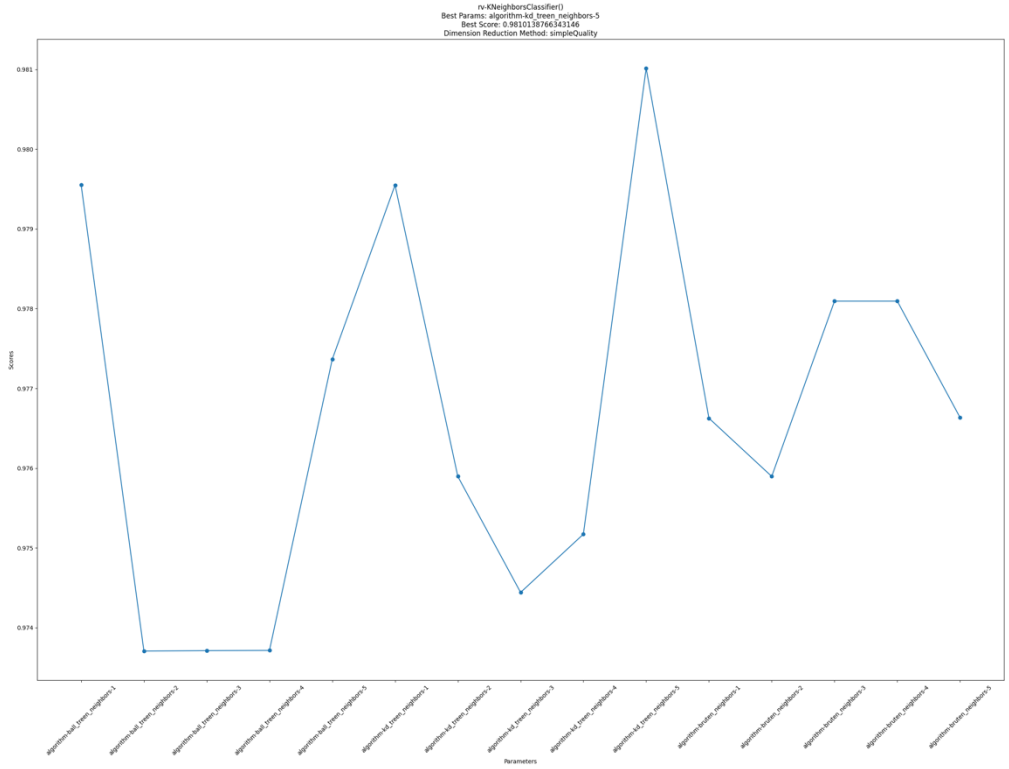
RV – K Neighbors Classifier – Embedded Methods



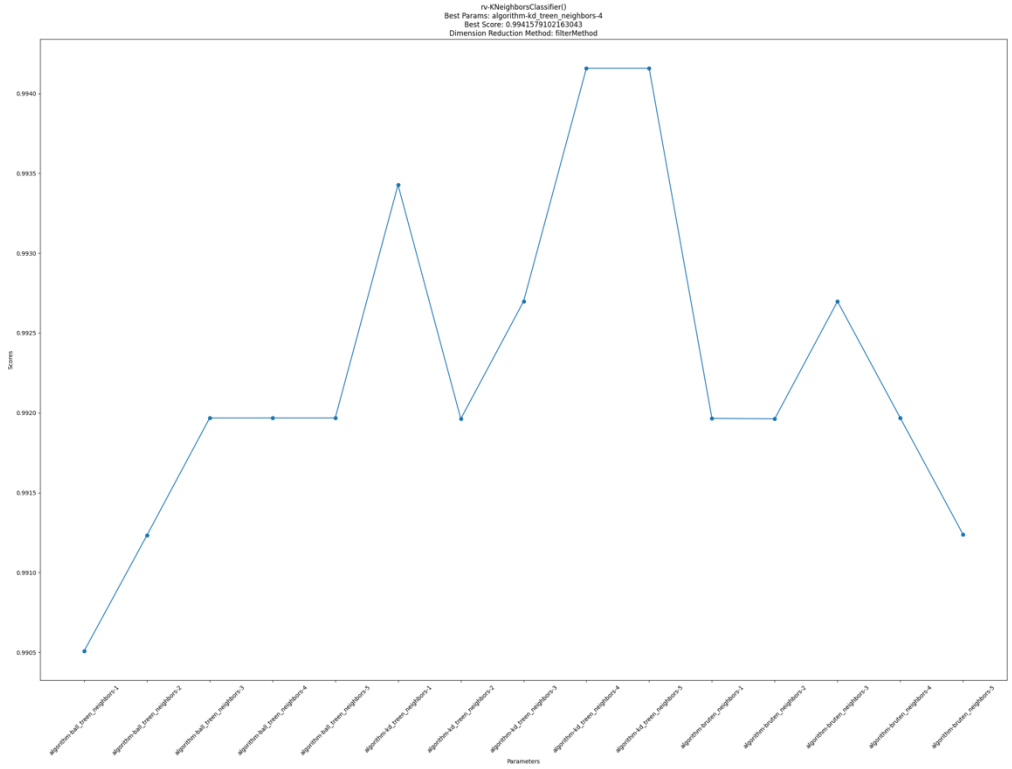
RV – K Neighbors Classifier – Feature Extraction



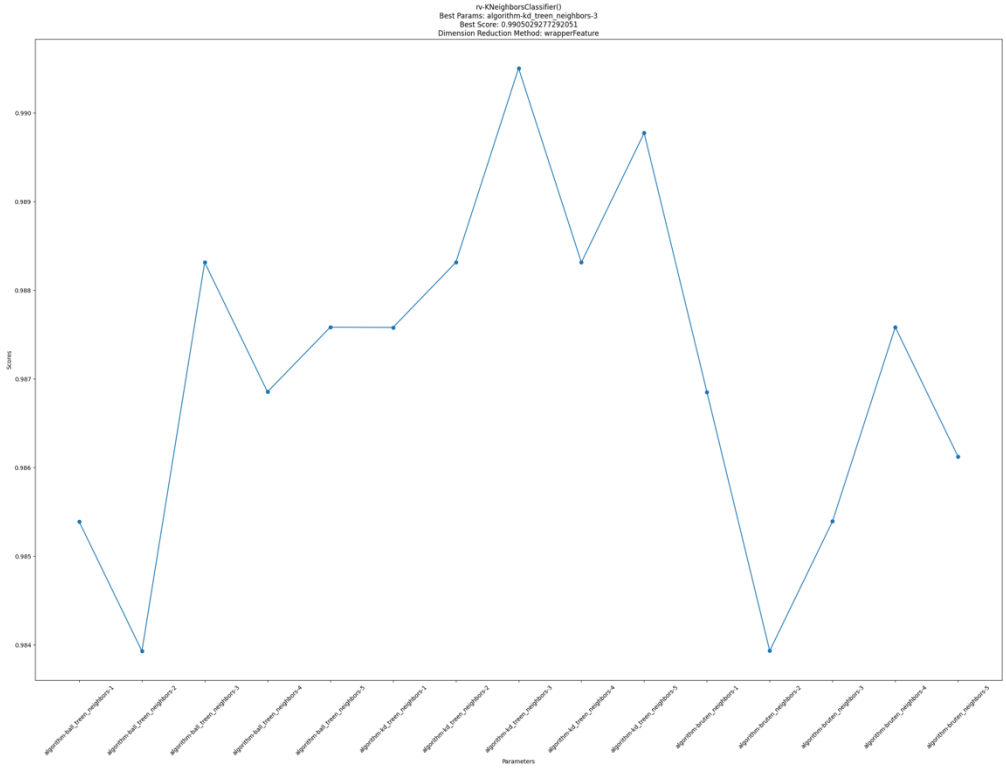
RV – K Neighbors Classifier – Simple Quality Filtering



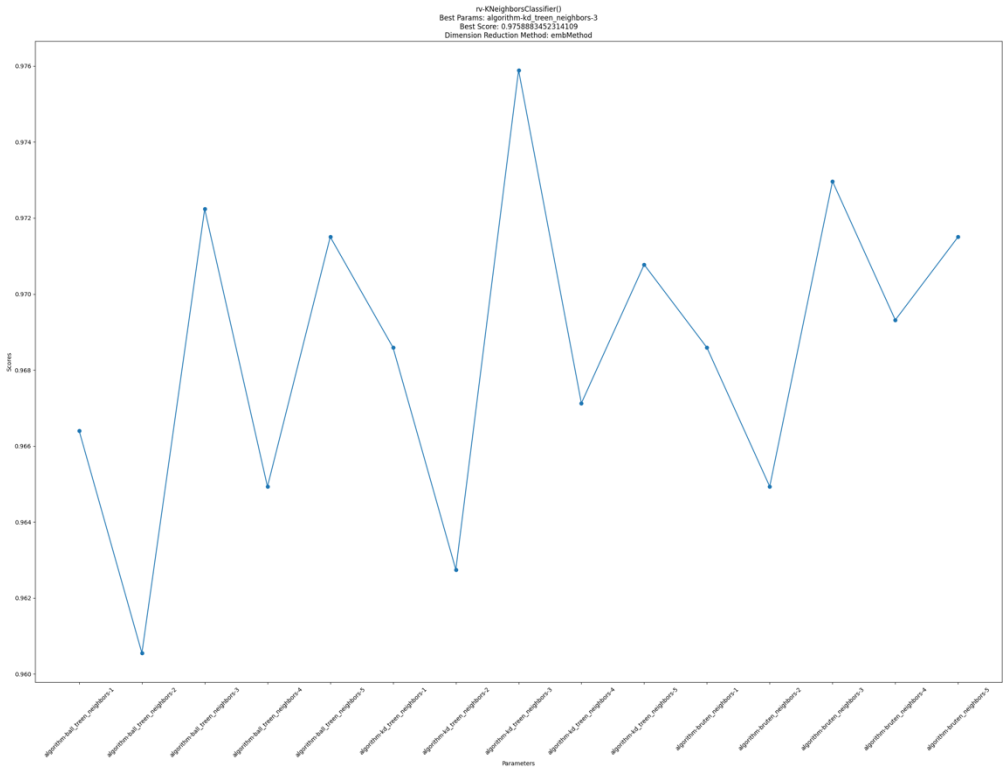
RV – K Neighbors Classifier – Filter Methods



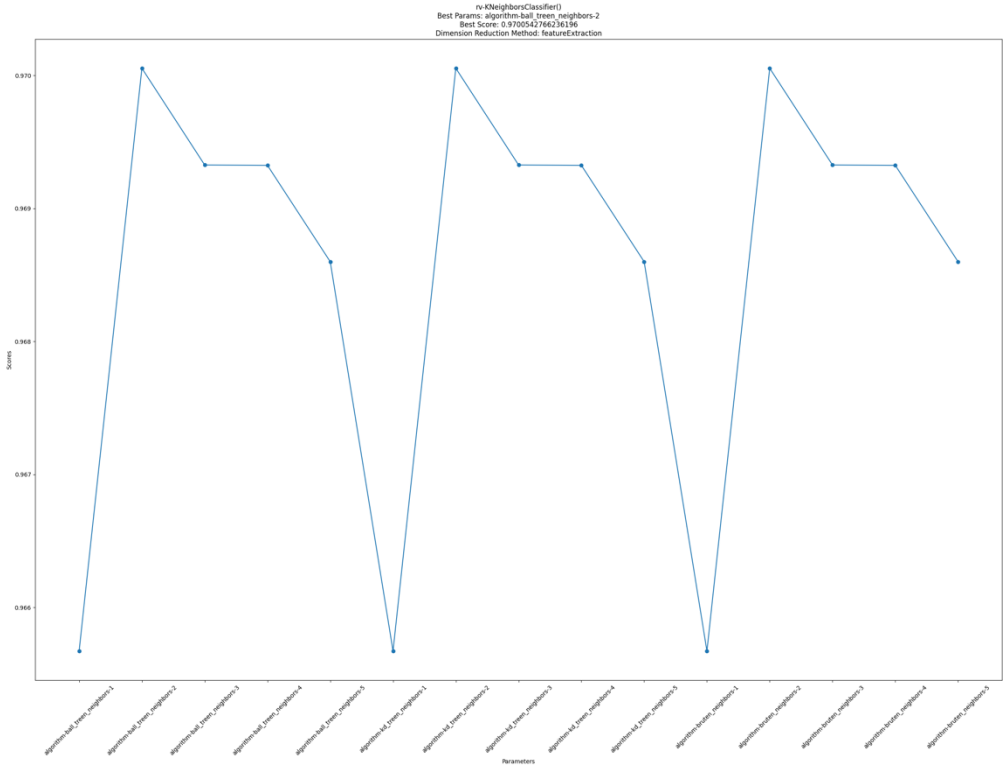
RV – K Neighbors Classifier – Wrapper Feature Selection



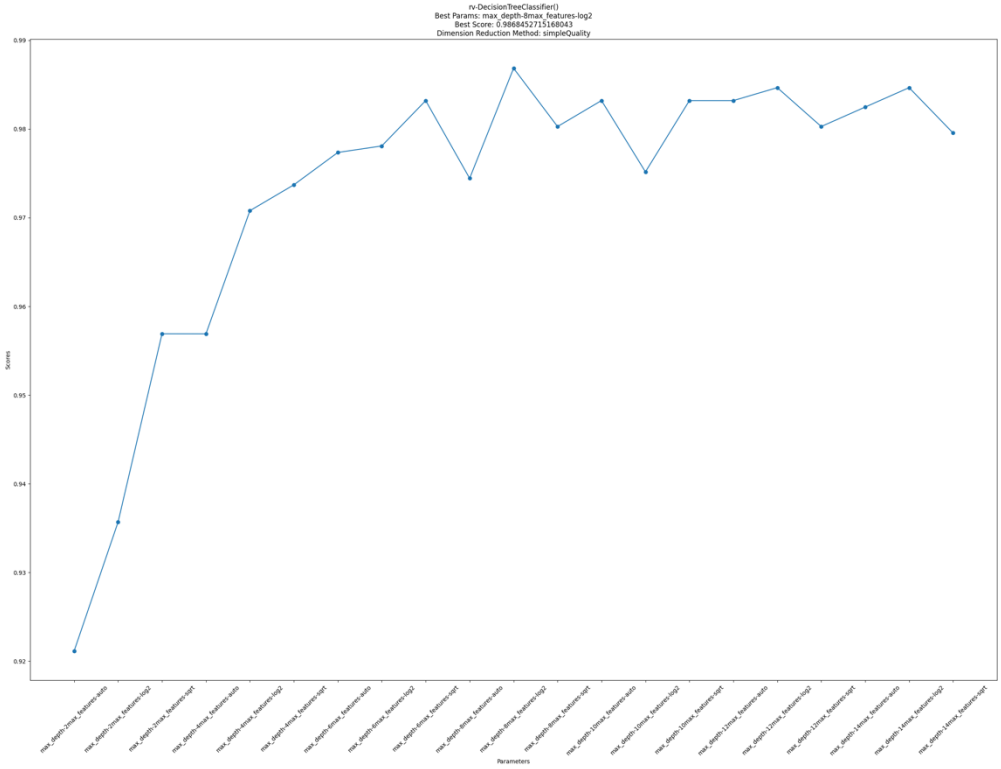
RV – Decision Tree Classifier – Embedded Methods



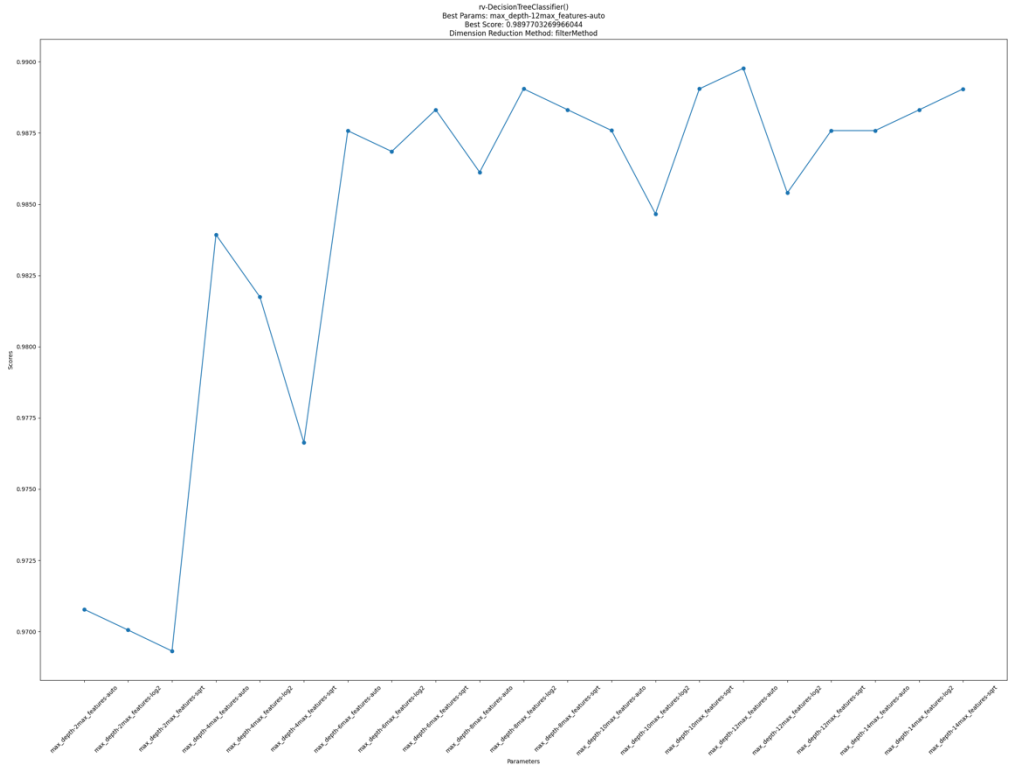
RV – Decision Tree Classifier – Feature Extraction



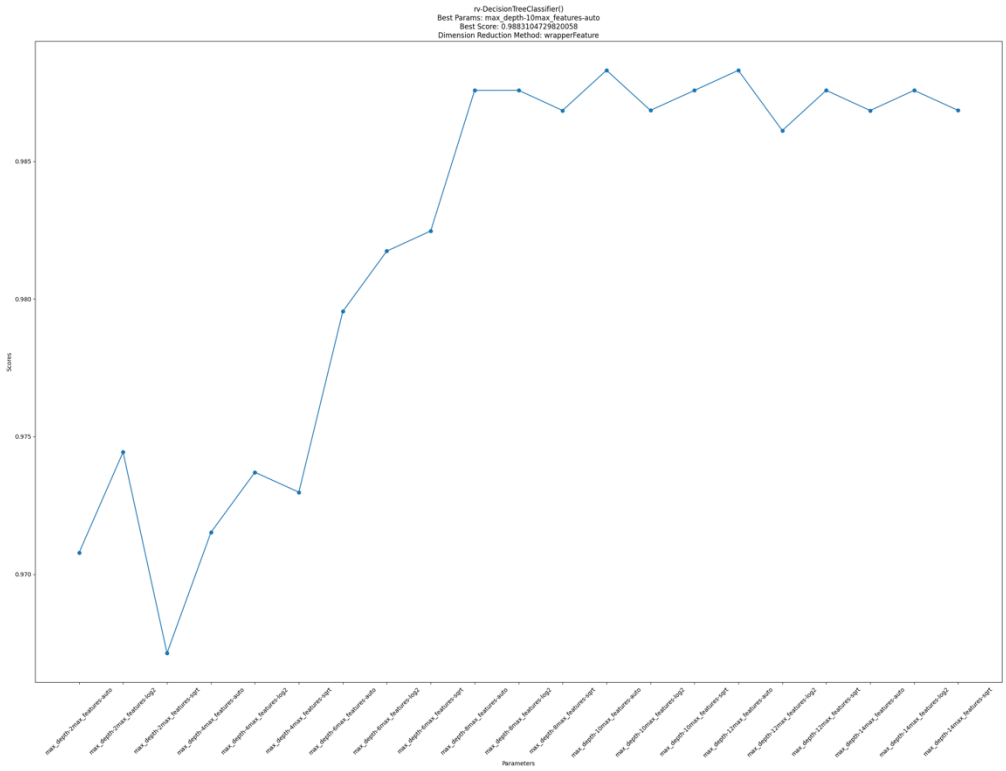
RV – Decision Tree Classifier – Simple Quality Filtering



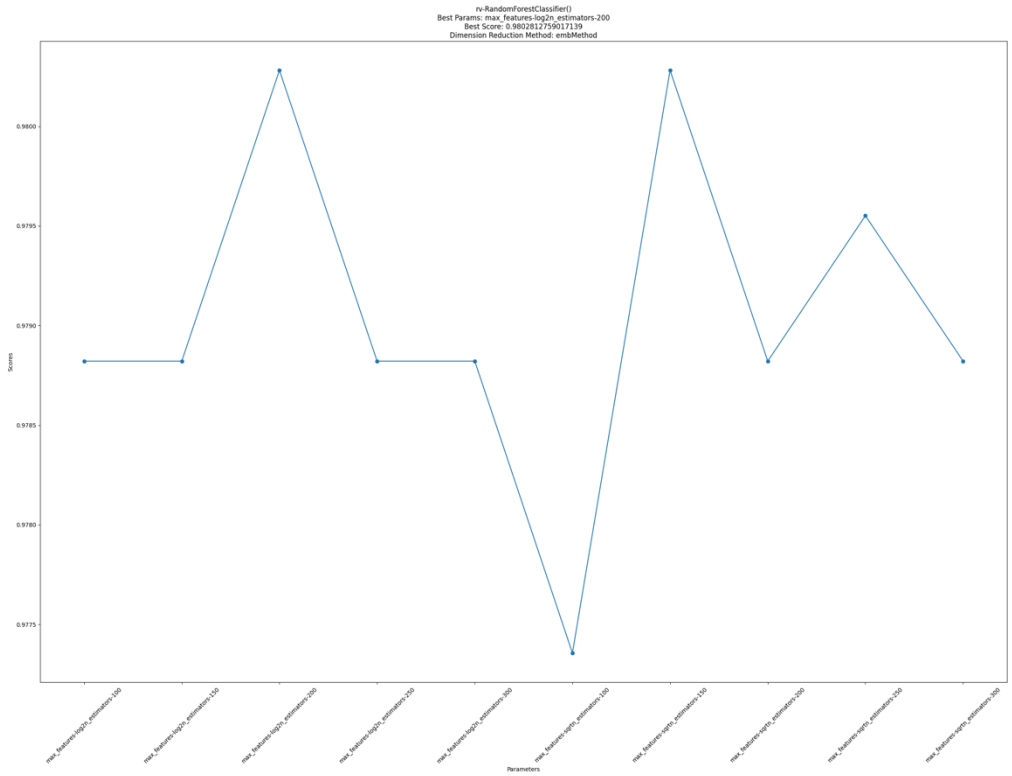
RV – Decision Tree Classifier – Filter Methods



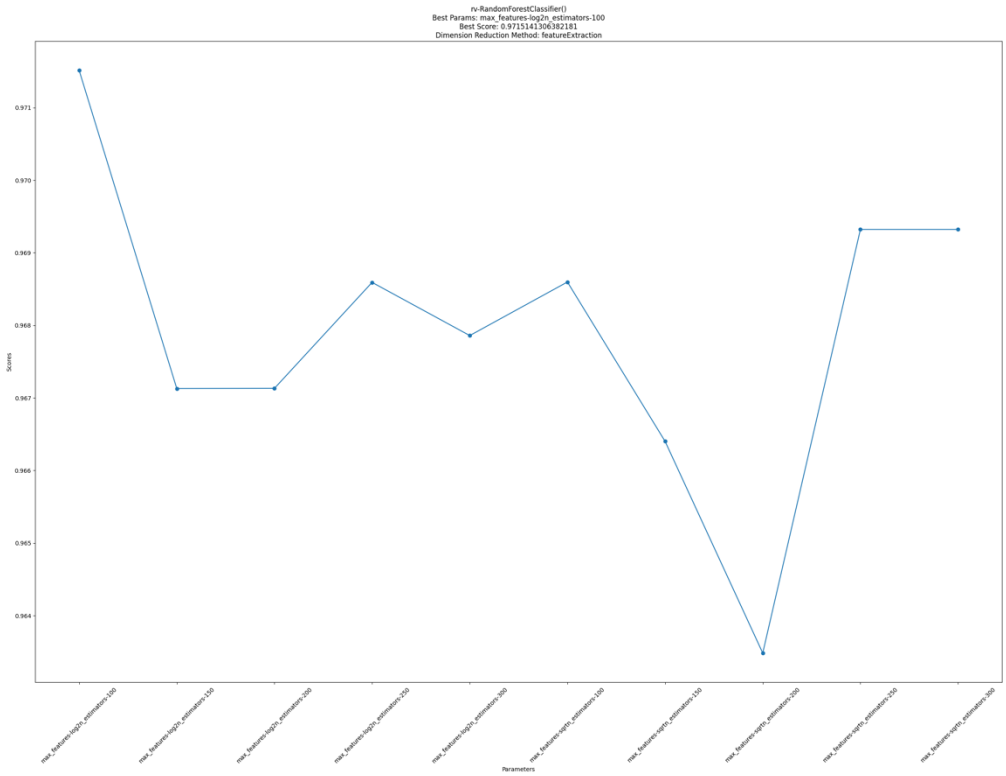
RV – Decision Tree Classifier – Wrapper Feature Selection



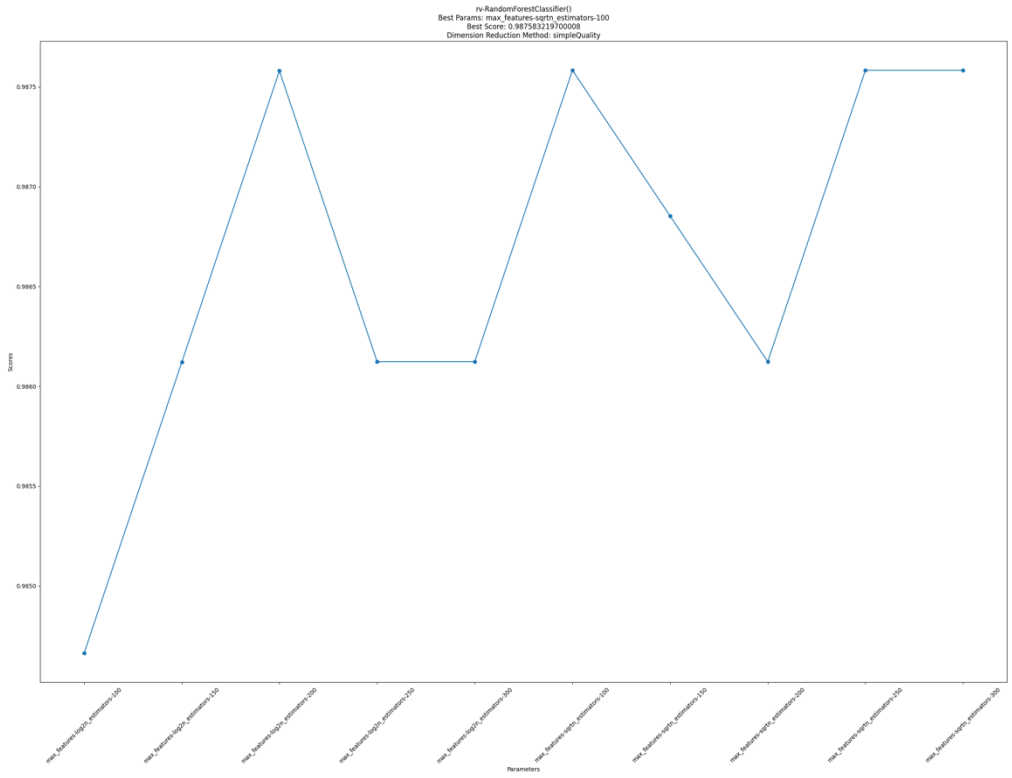
RV – Random Forest Classifier – Embedded Methods



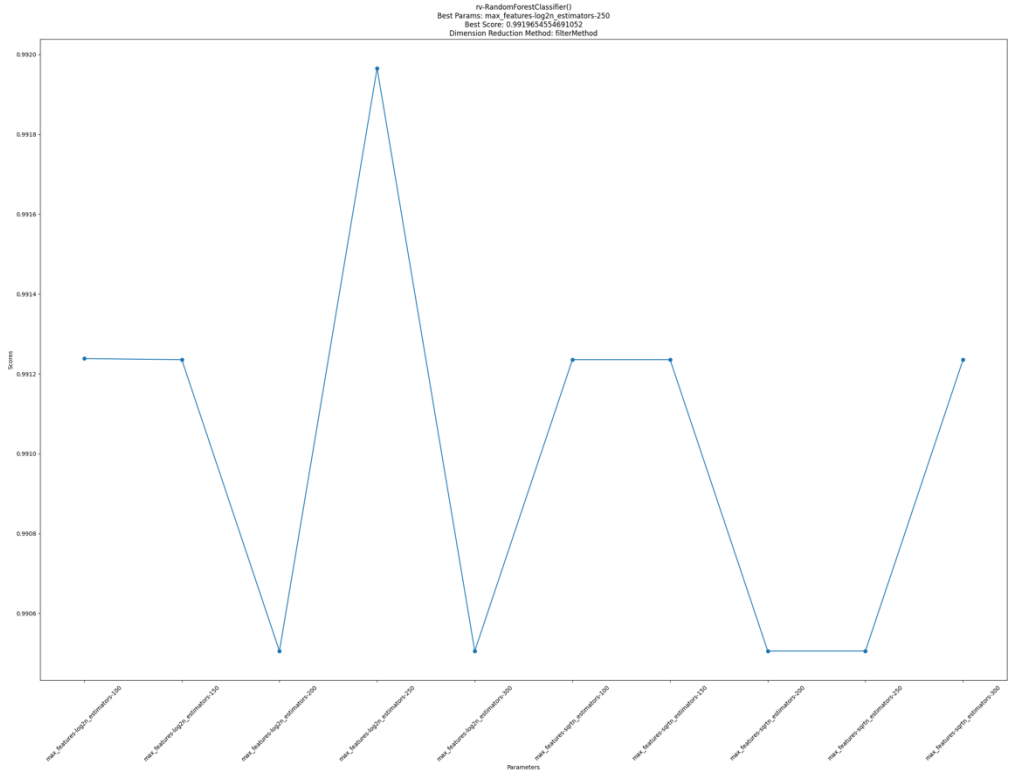
RV – Random Forest Classifier – Feature Extraction



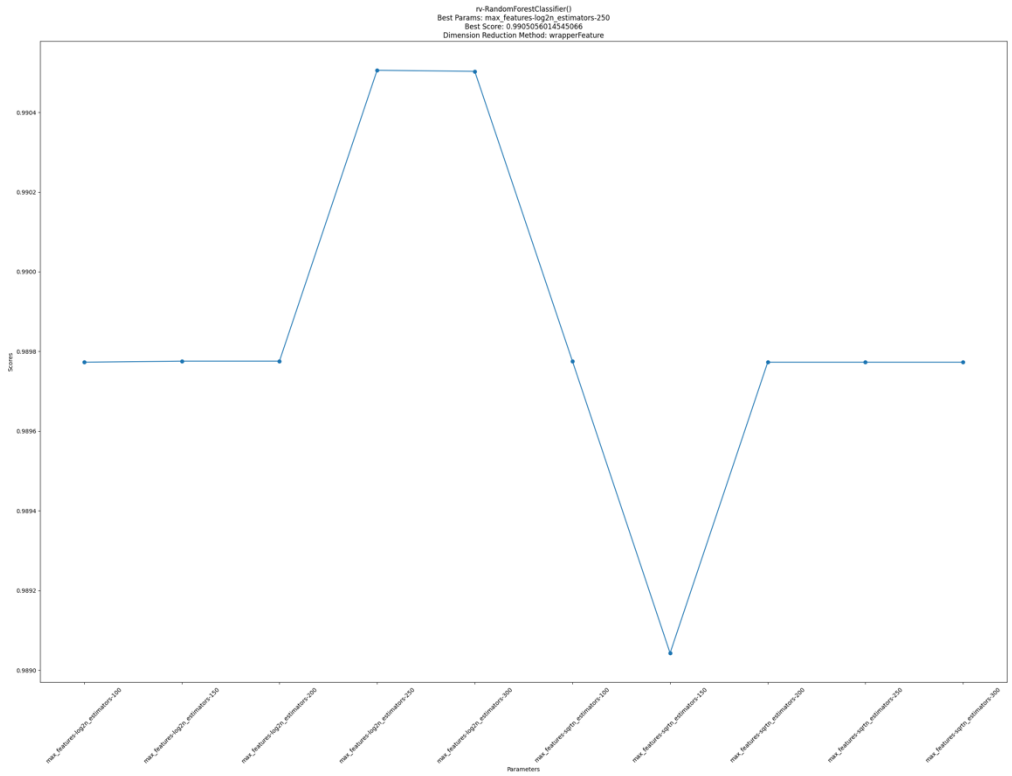
RV – Random Forest Classifier – Simple Quality Filtering



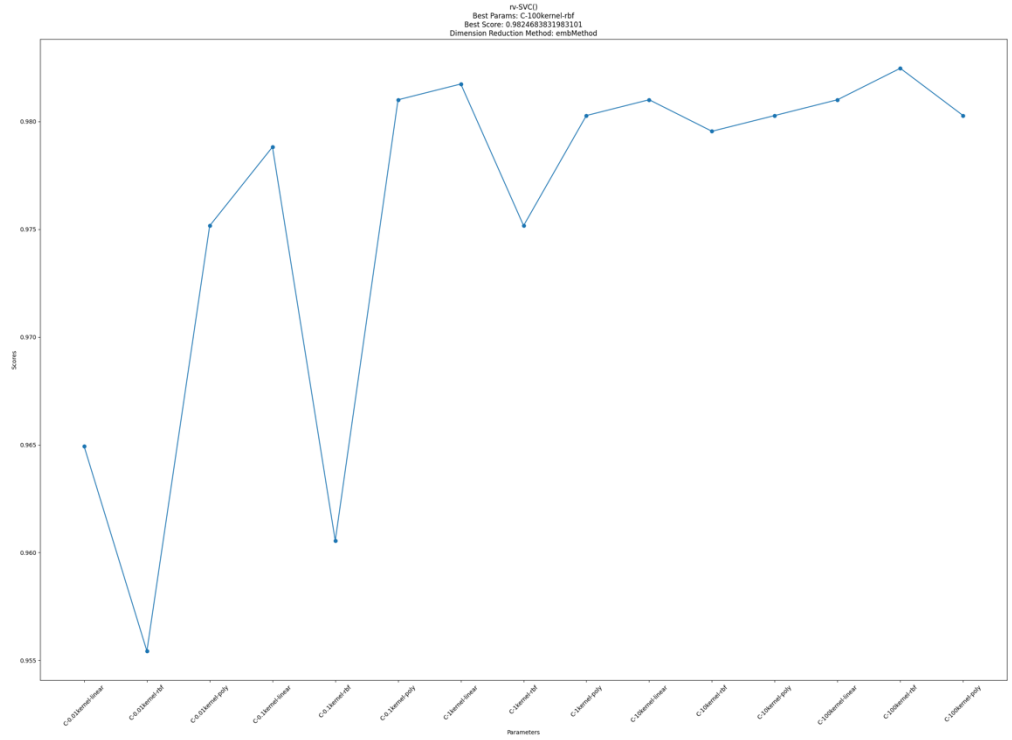
RV – Random Forest Classifier – Filter Methods



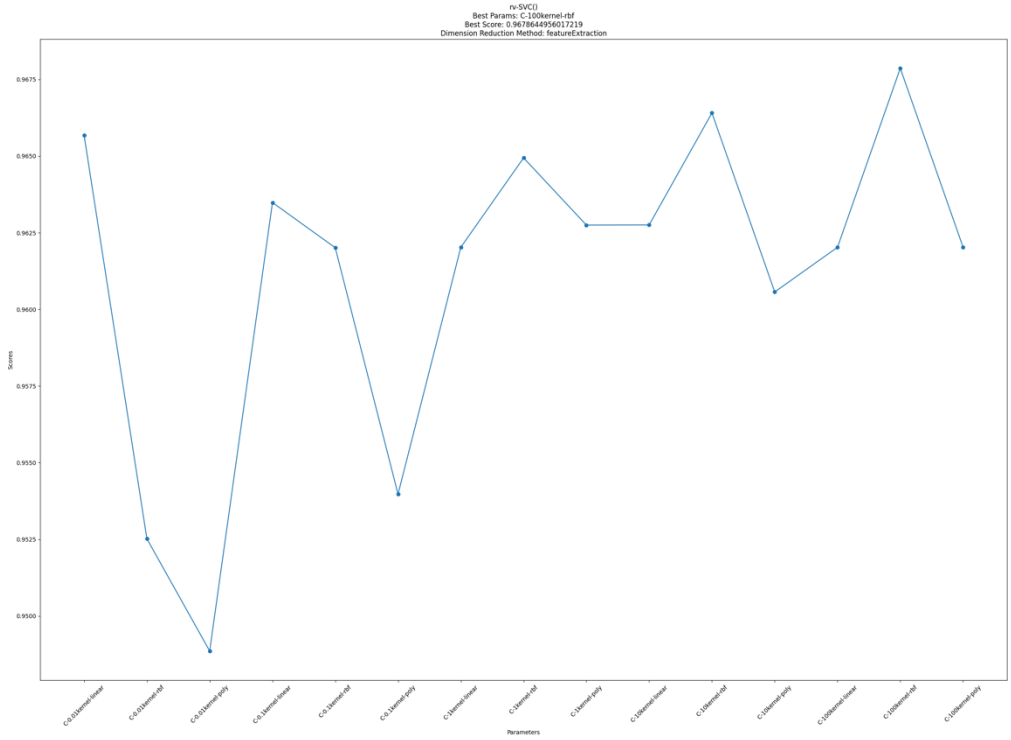
RV – Random Forest Classifier – Wrapper Feature Selection



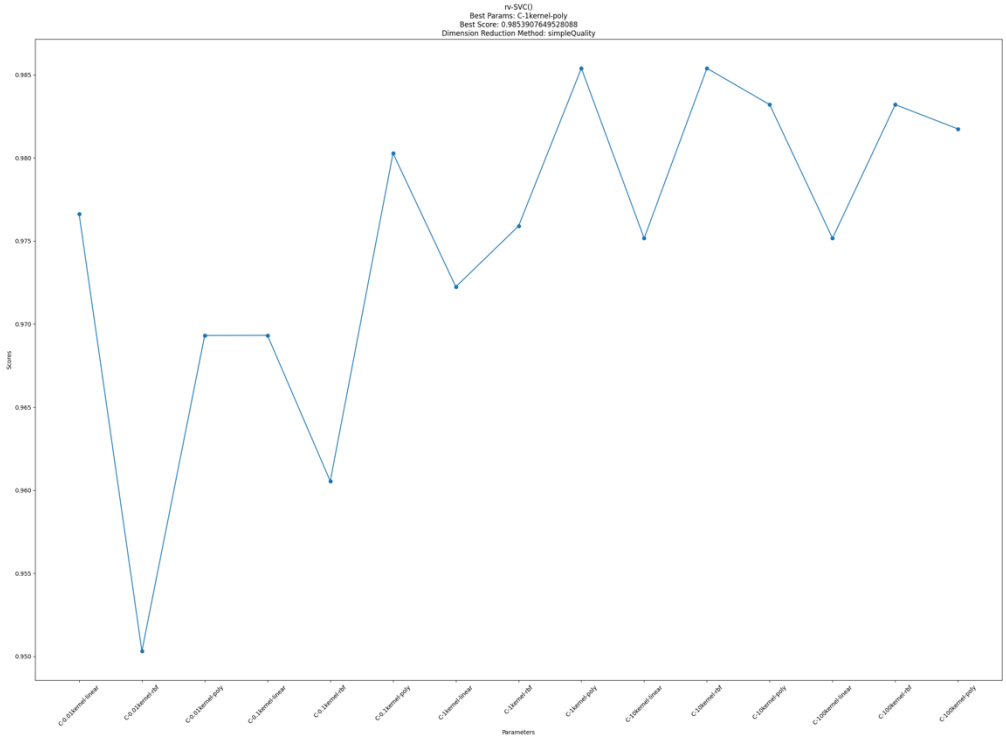
RV – SVM – Embedded Methods



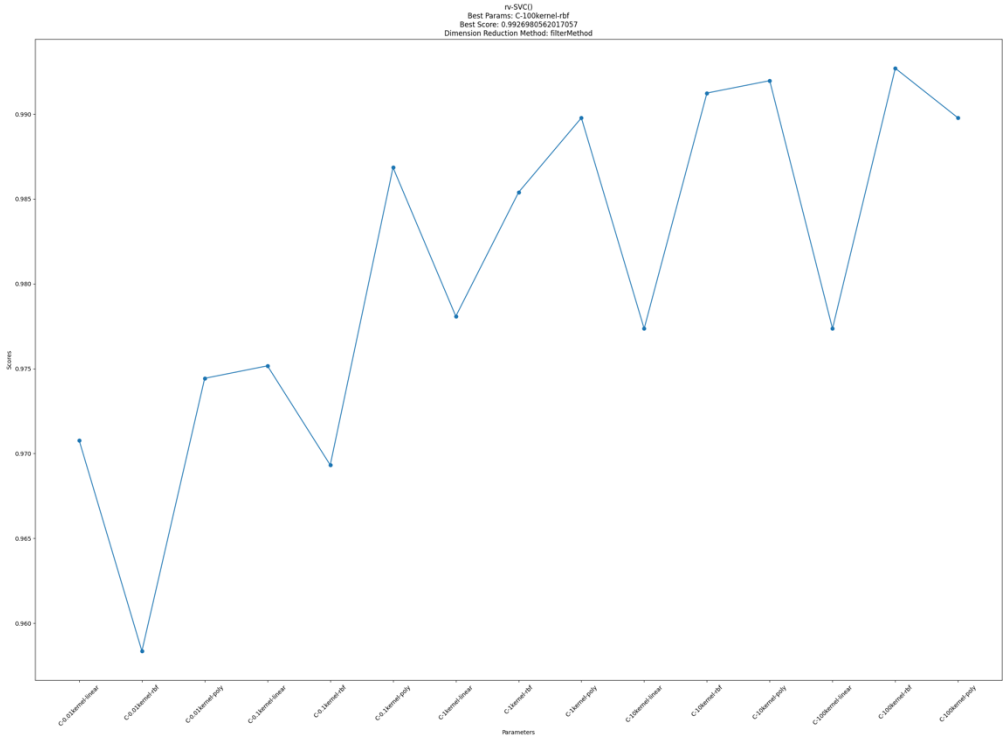
RV – SVM – Feature Extraction



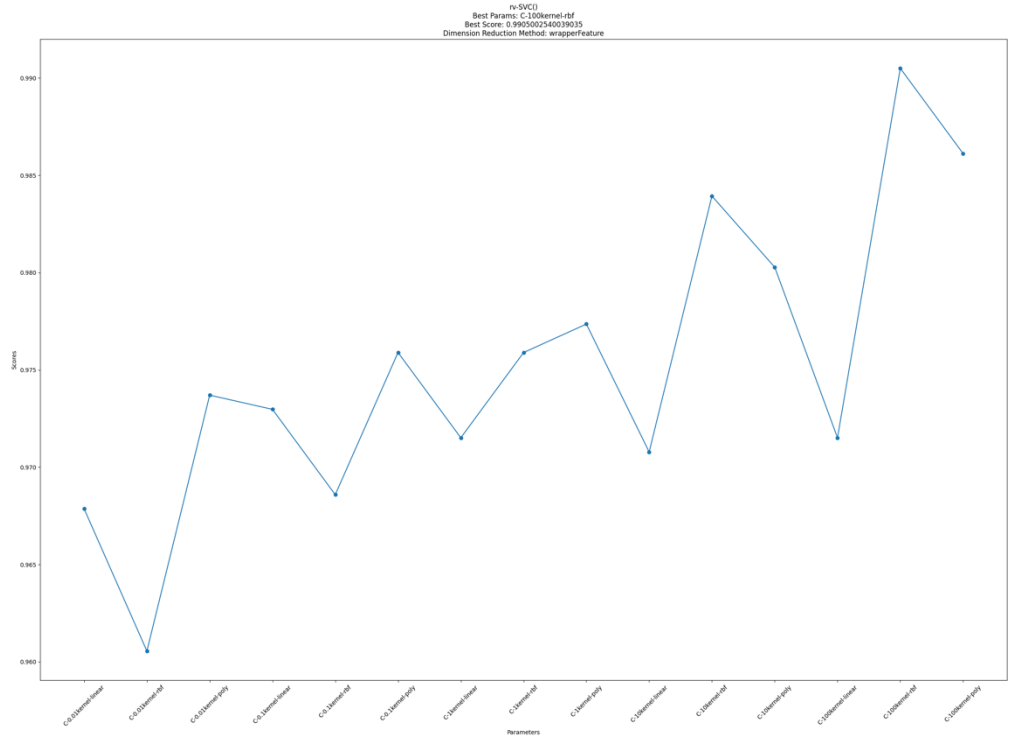
RV – SVM – Simple Quality Filtering



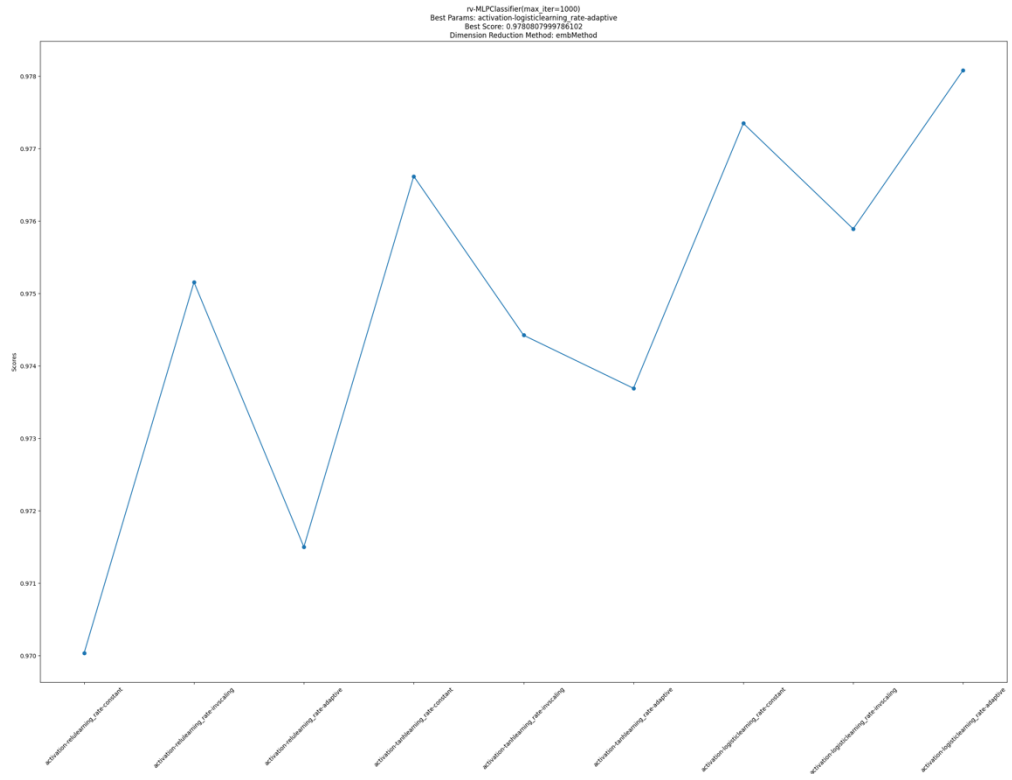
RV – SVM – Filter Methods



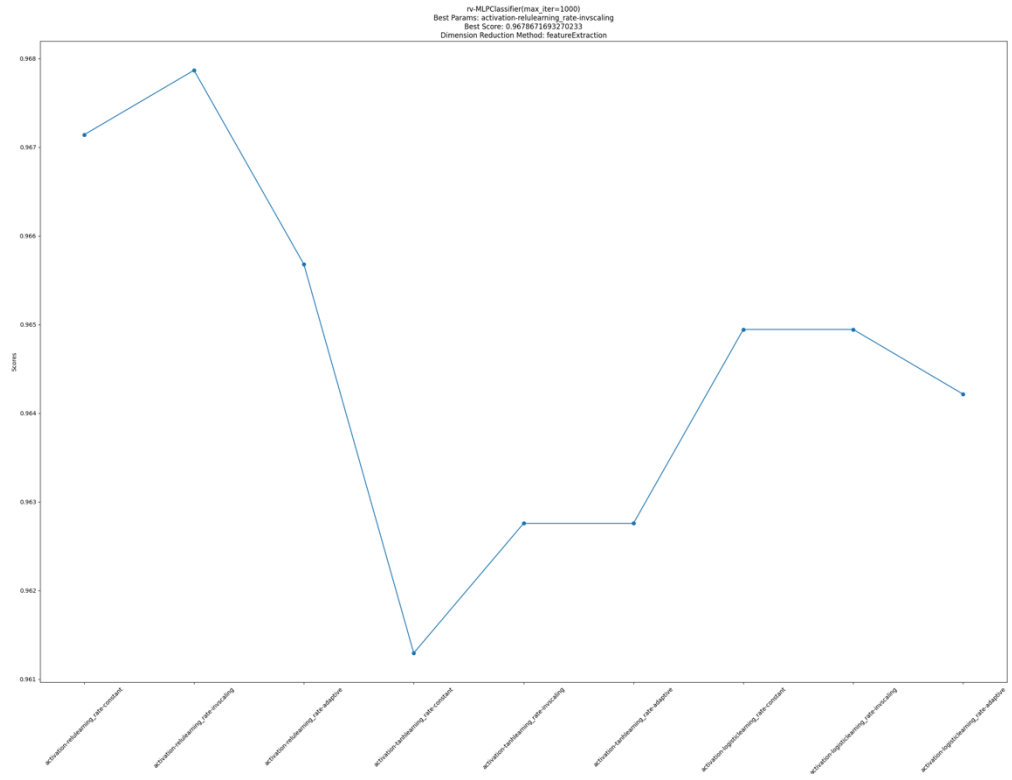
RV – SVM – Wrapper Feature Selection



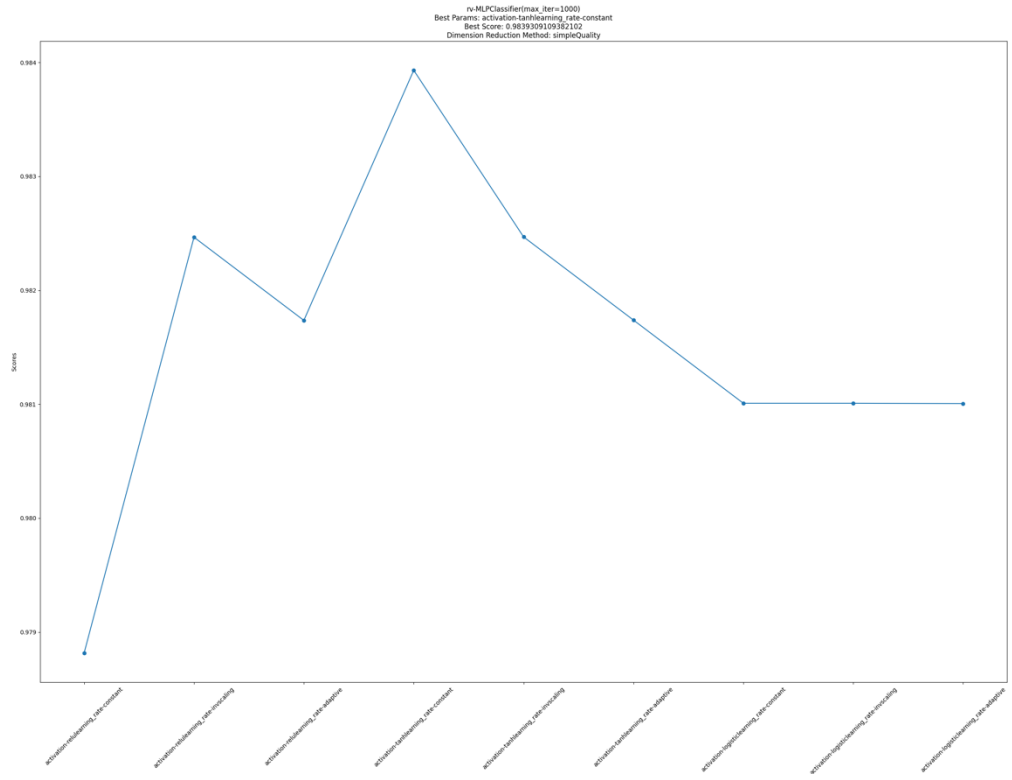
RV – MLP Classifier – Embedded Methods



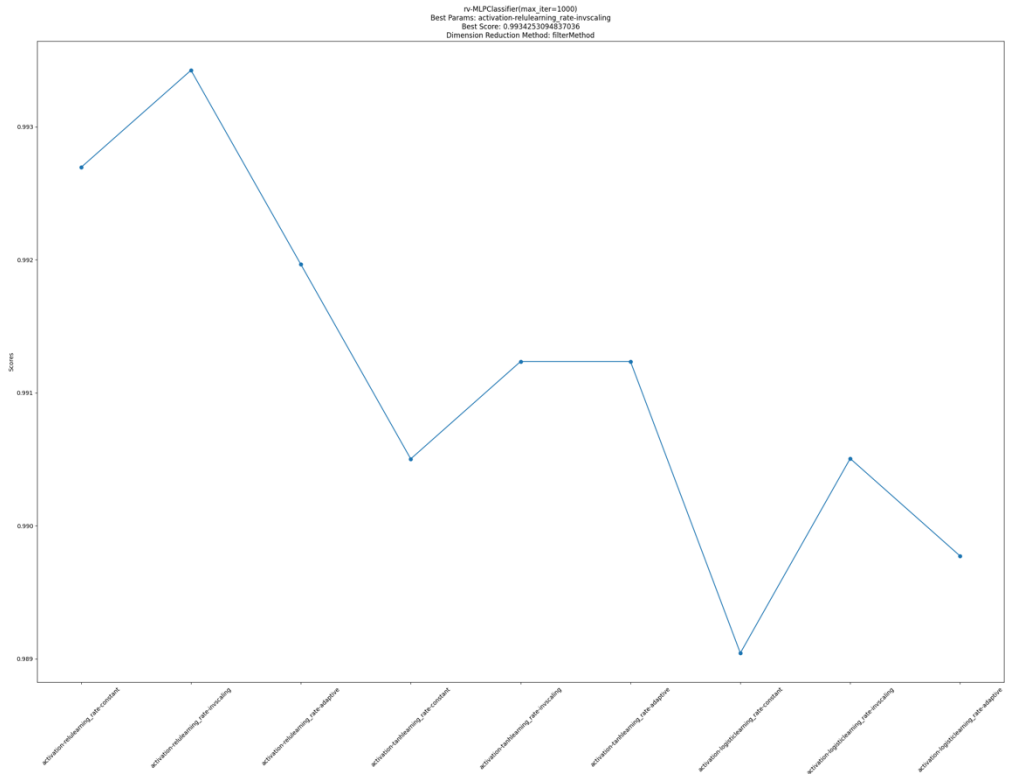
RV – MLP Classifier – Feature Extraction



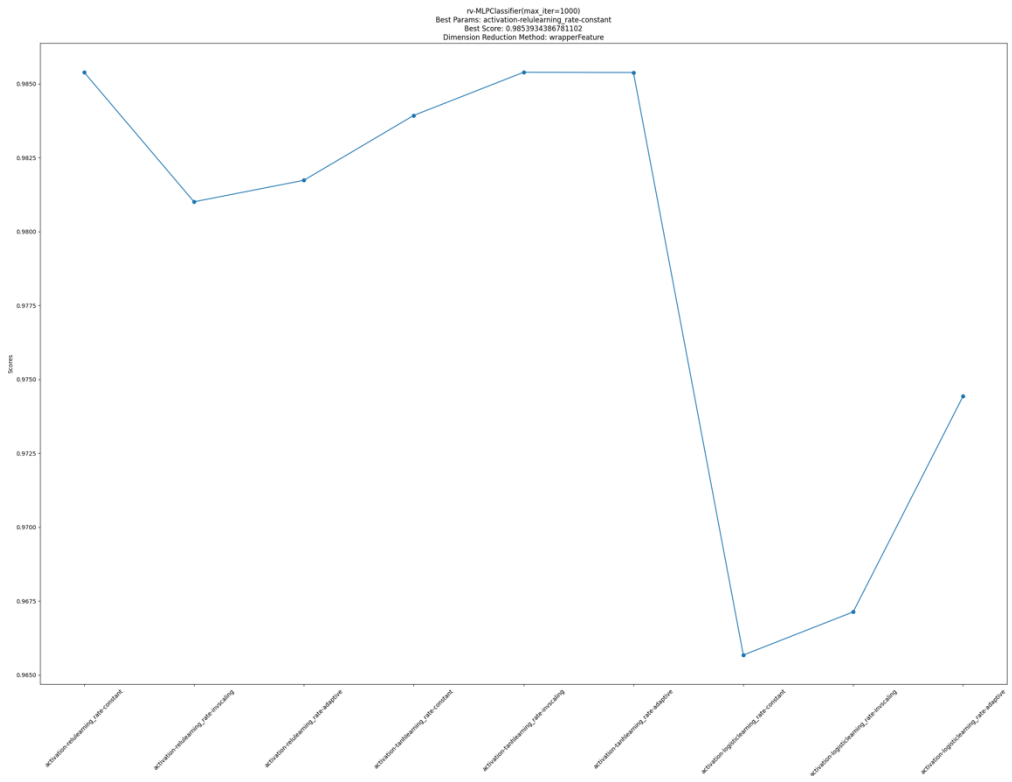
RV – MLP Classifier – Simple Quality Filtering



RV – MLP Classifier – Filter Methods



RV – MLP Classifier – Wrapper Feature Selection



2.3 for additional hyperparameters tuned

Please check the graphs above. For every model, we set more than 3 values for a categorical hyperparameter and more than 5 values for a numerical one.

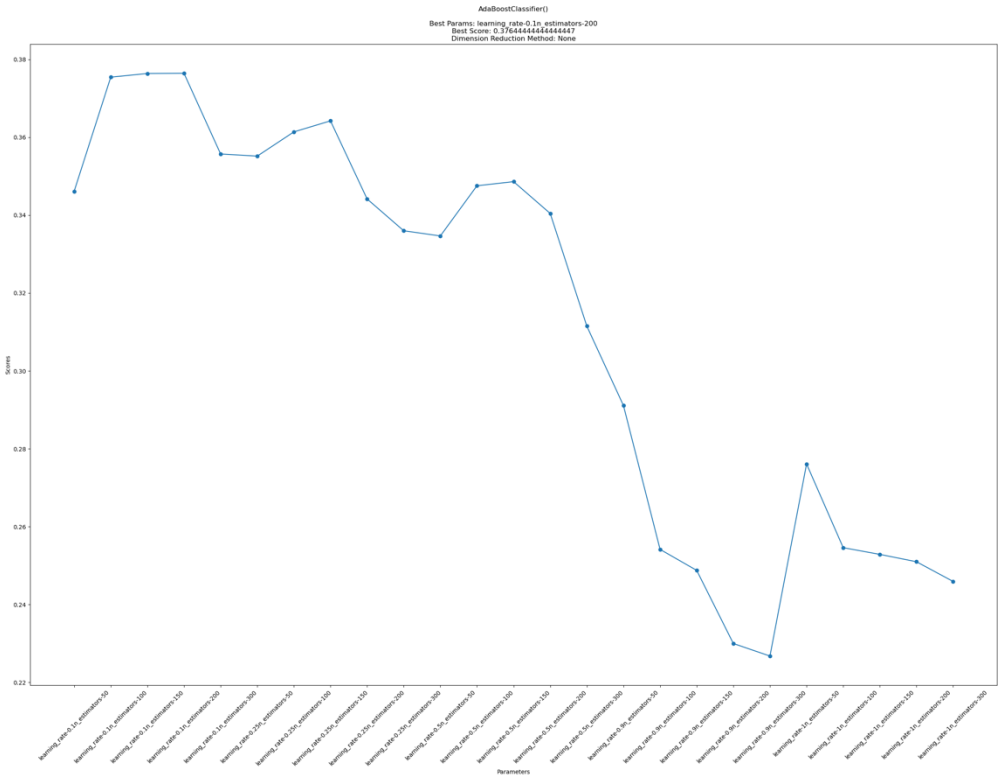
2.4 for additional classifiers

Please check the graphs above. We applied all models that were listed on the assignment (k-nearest neighbors, SVM, Decision tree, Artificial Neural Network, Random Forest Classifier and Naïve Bayes Classifier).

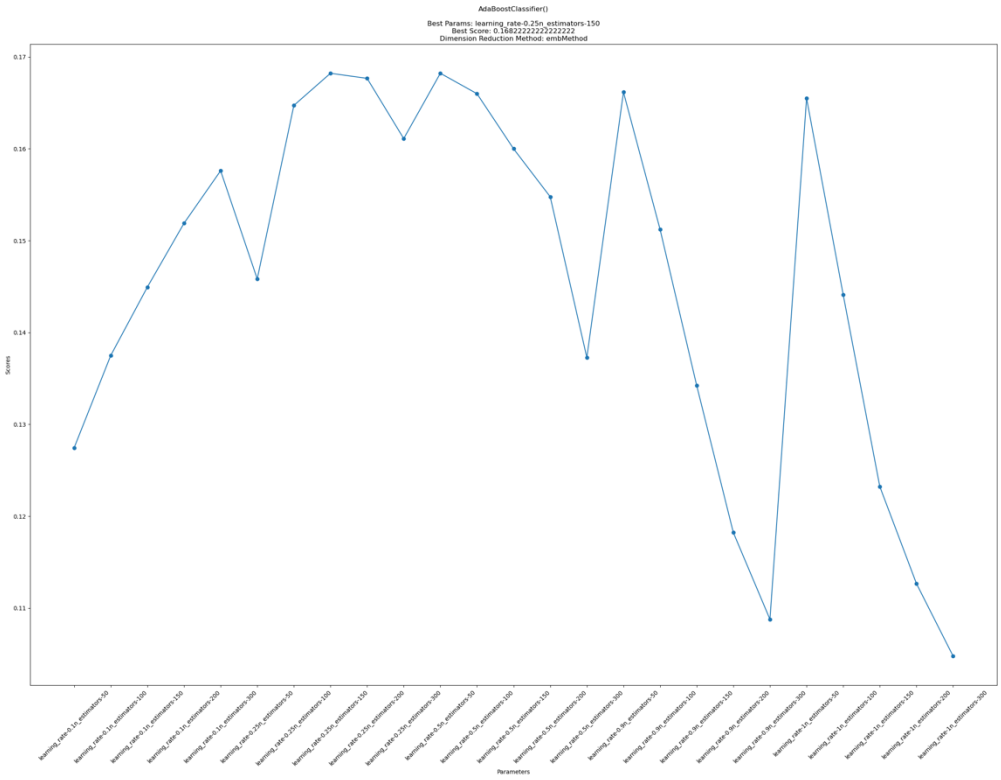
2.5 For multi-class classification

*(**None** means without dimension reduction)*

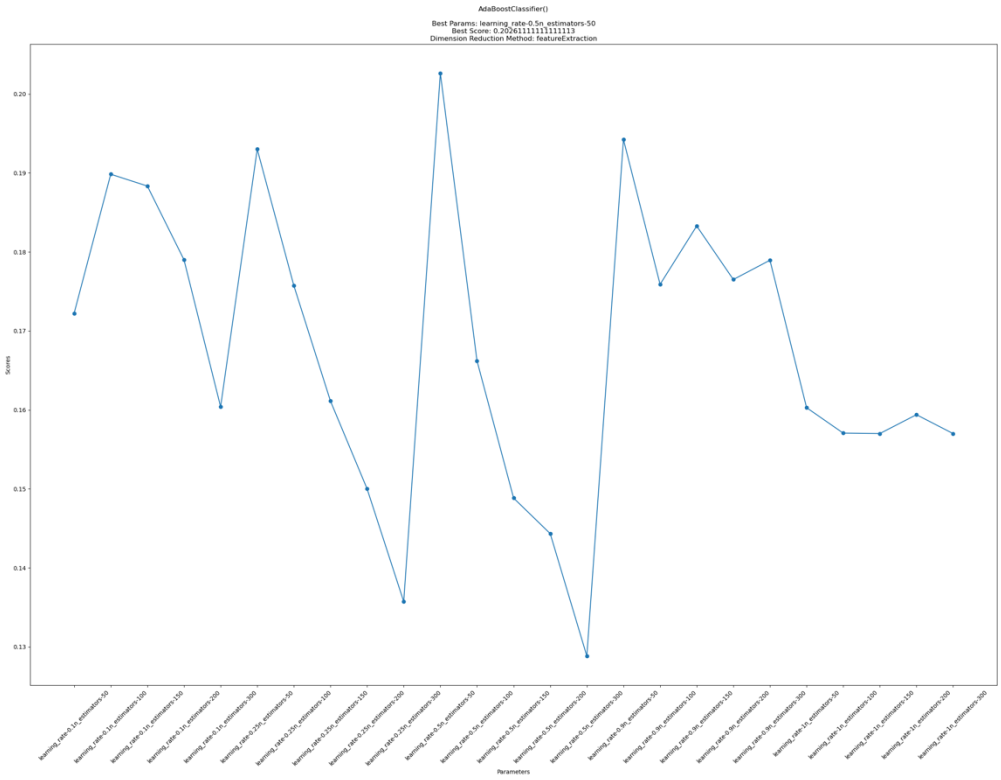
AdaBoost Classifier – None



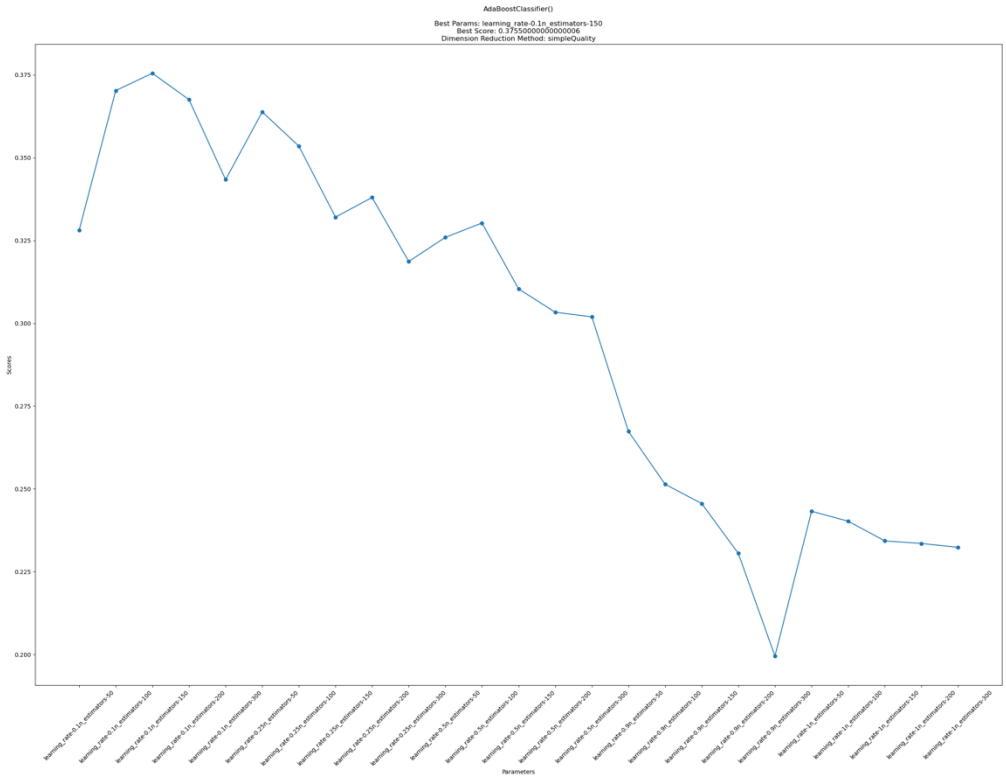
AdaBoost Classifier – Embedded Methods



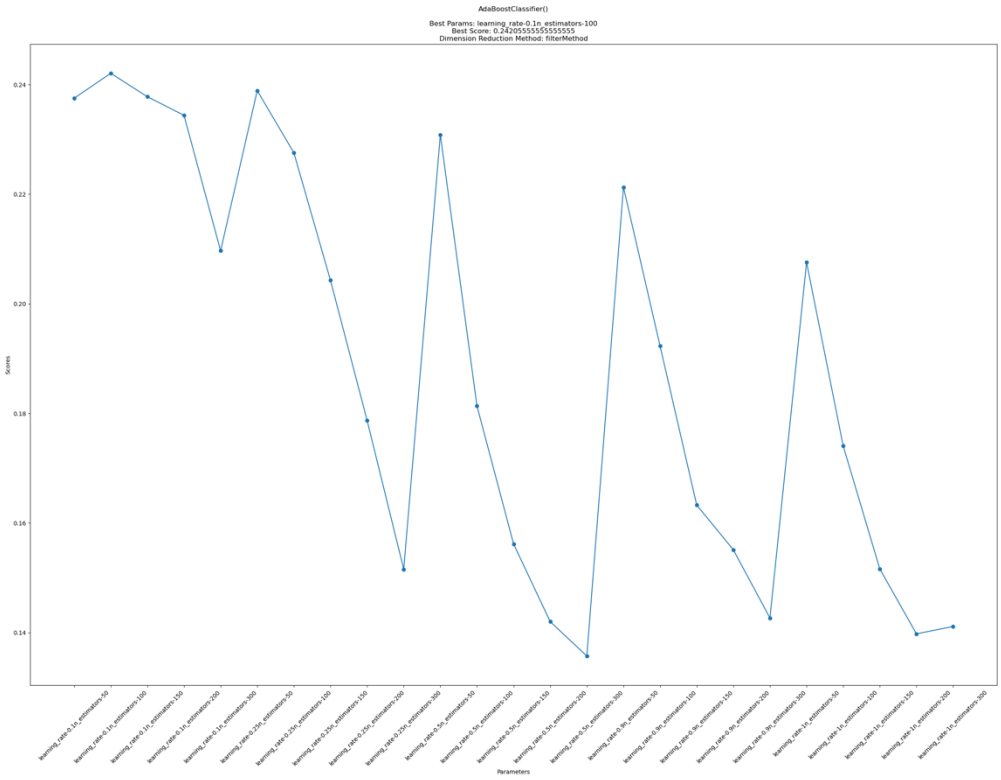
AdaBoost Classifier – Feature Extraction



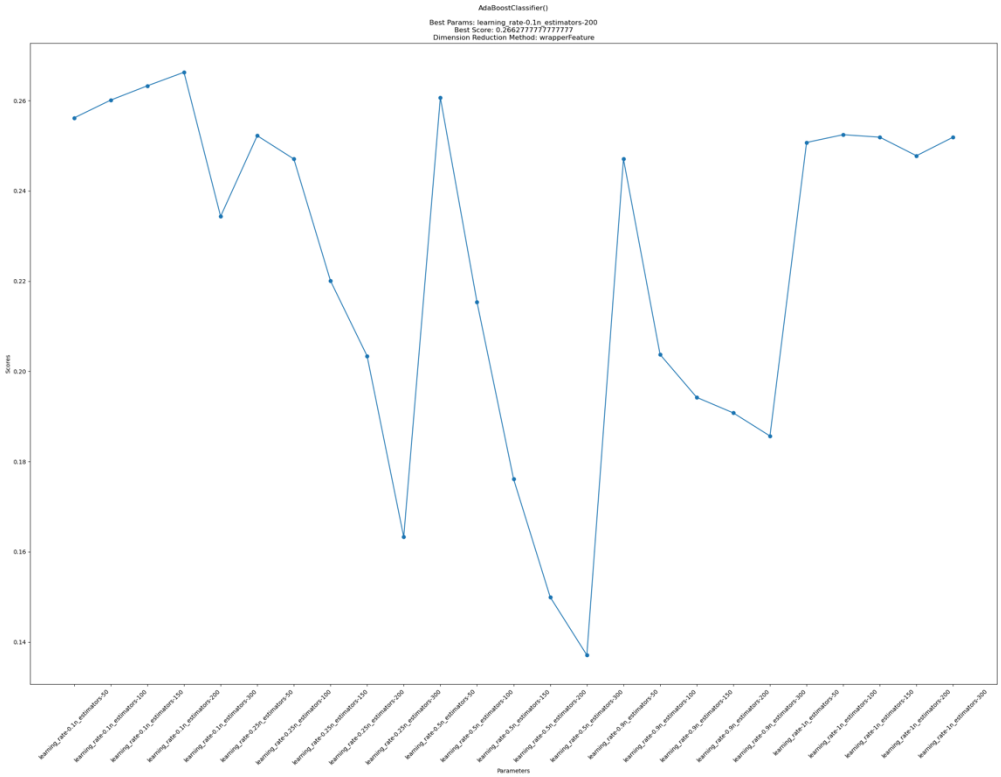
AdaBoost Classifier – Simple Quality Filtering



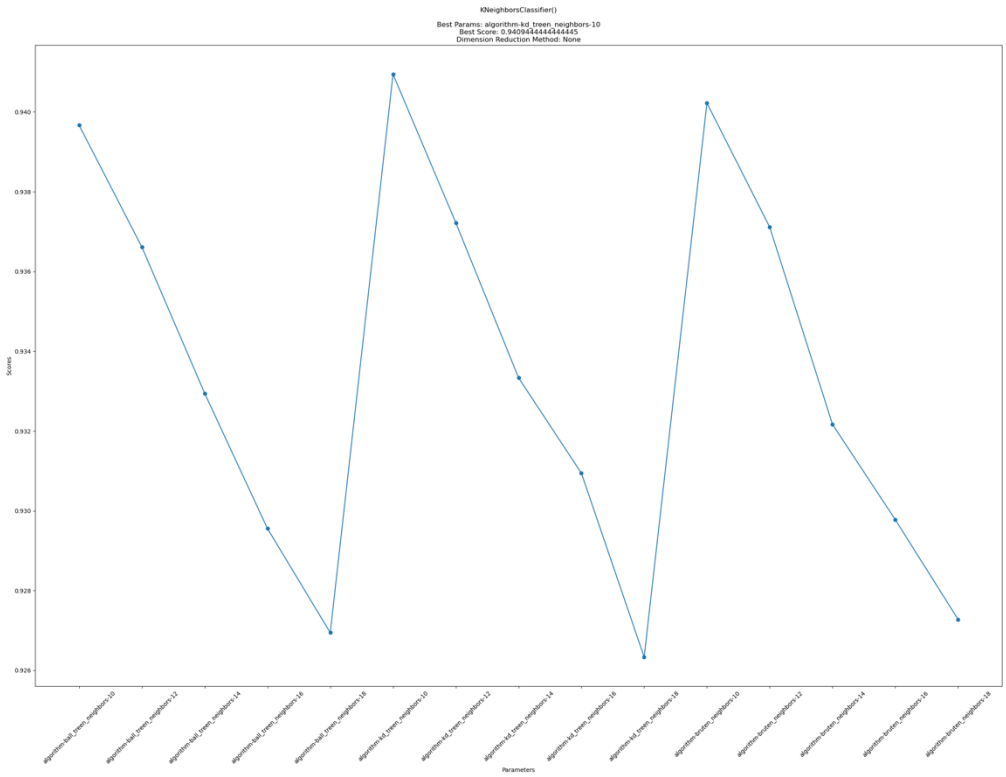
AdaBoost Classifier – Filter Methods



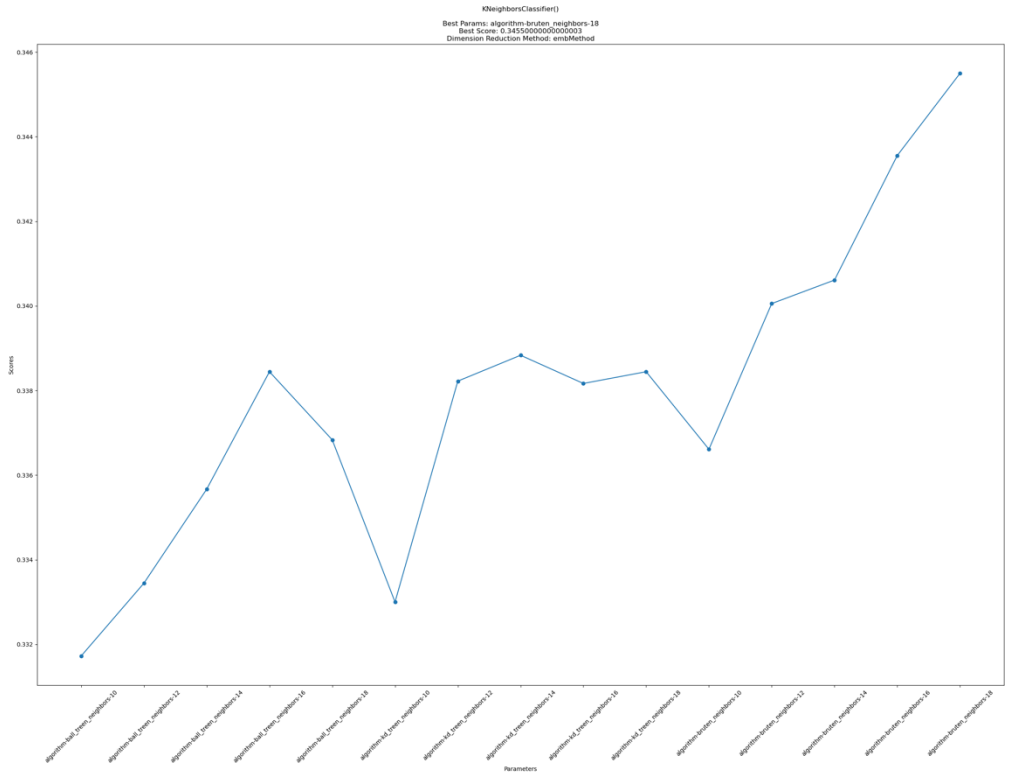
AdaBoost Classifier – Wrapper Feature Selection



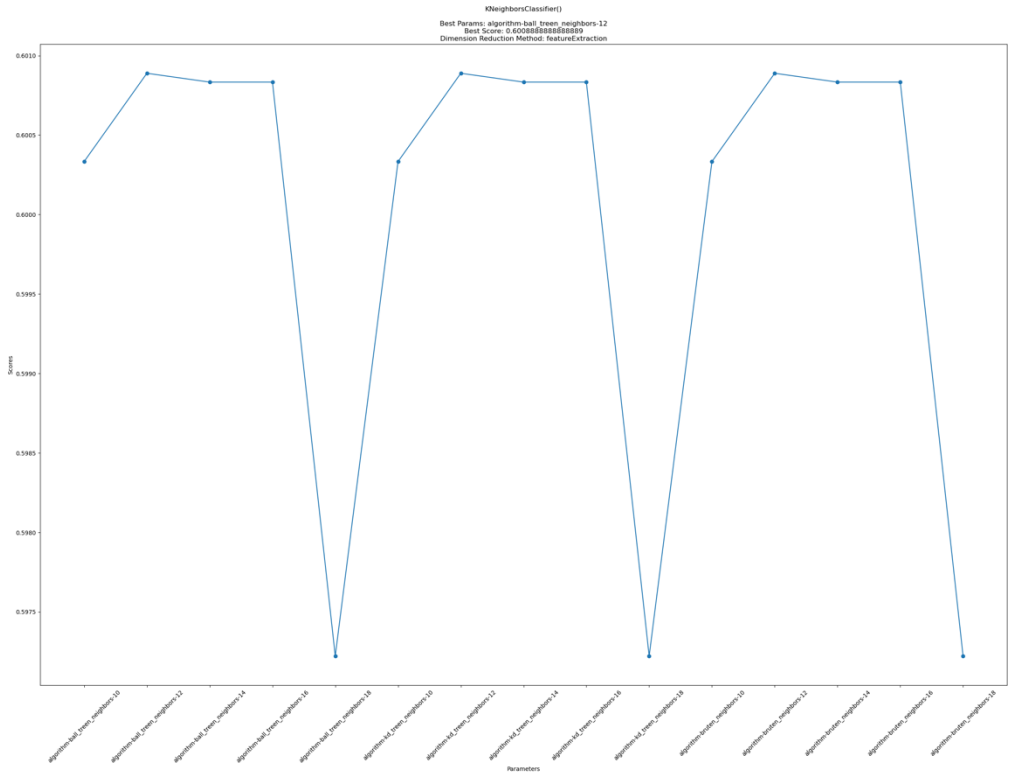
K Neighbors Classifier – None



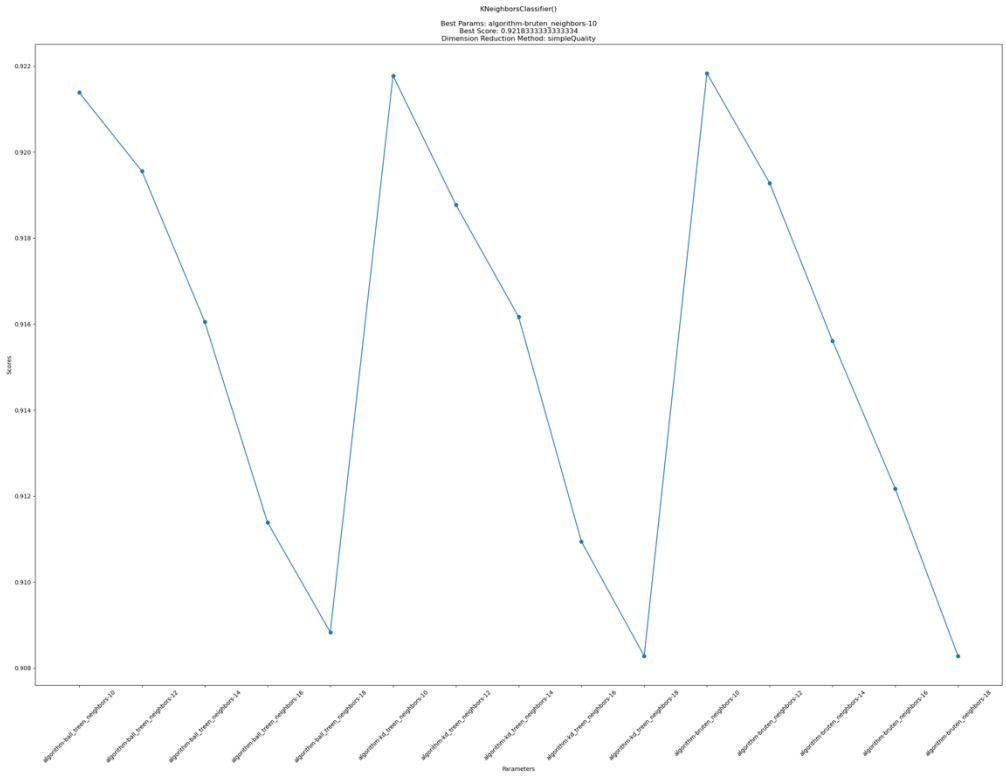
K Neighbors Classifier – Embedded Methods



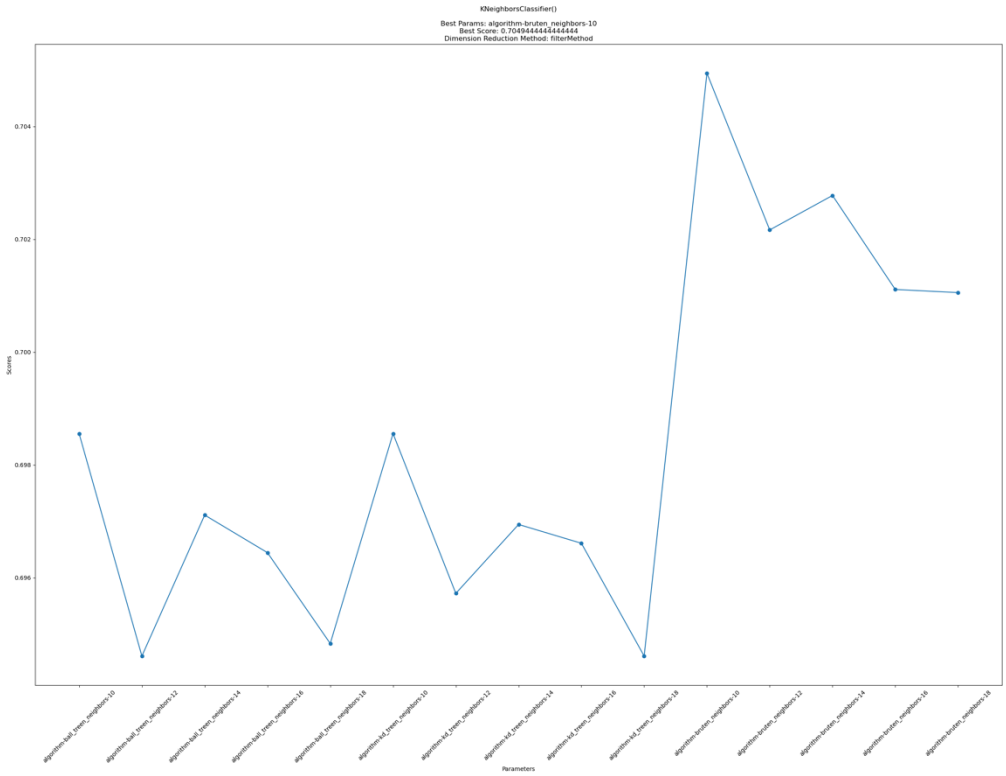
K Neighbors Classifier – Feature Extraction



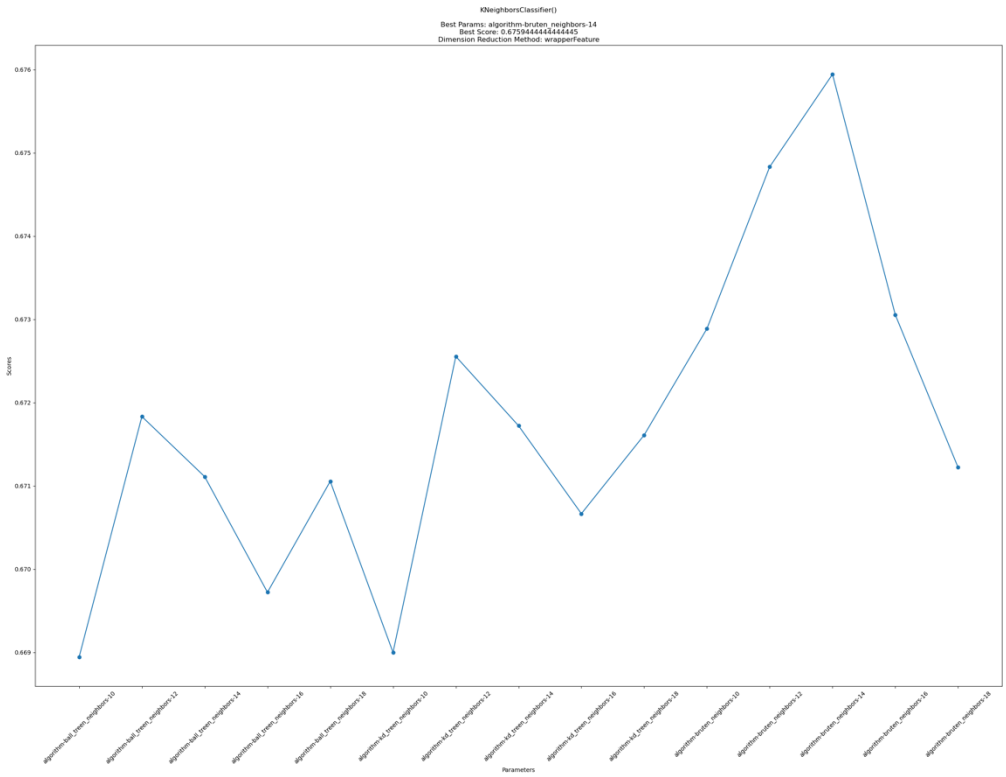
K Neighbors Classifier – Simple Quality Filtering



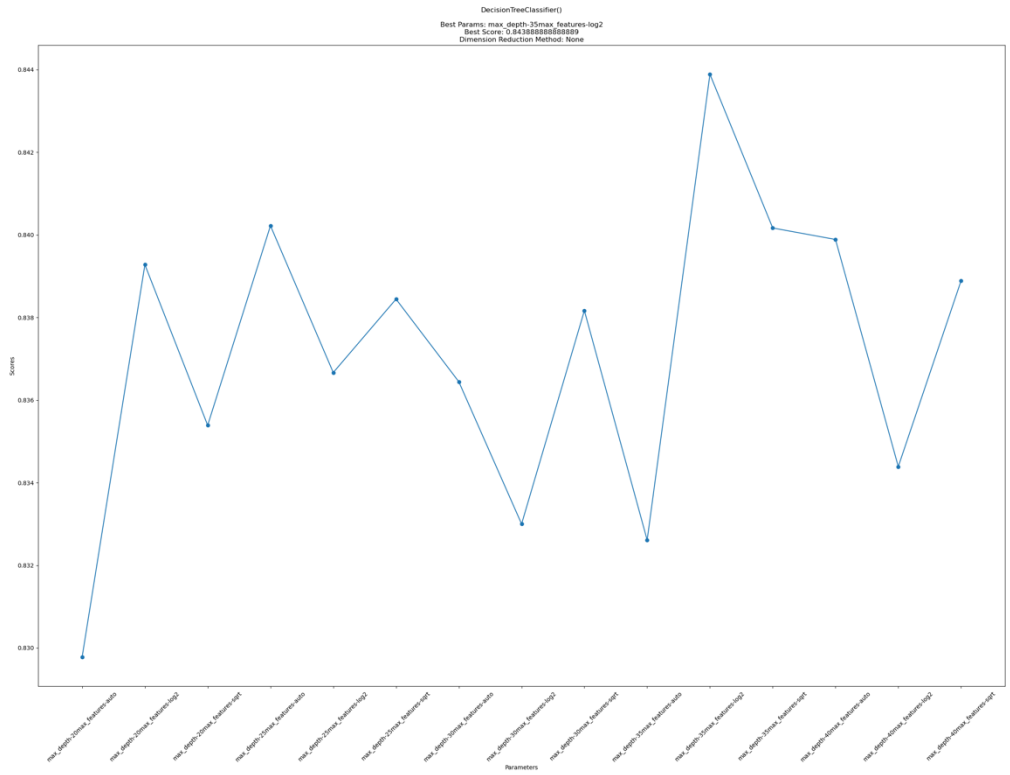
K Neighbors Classifier – Filter Methods



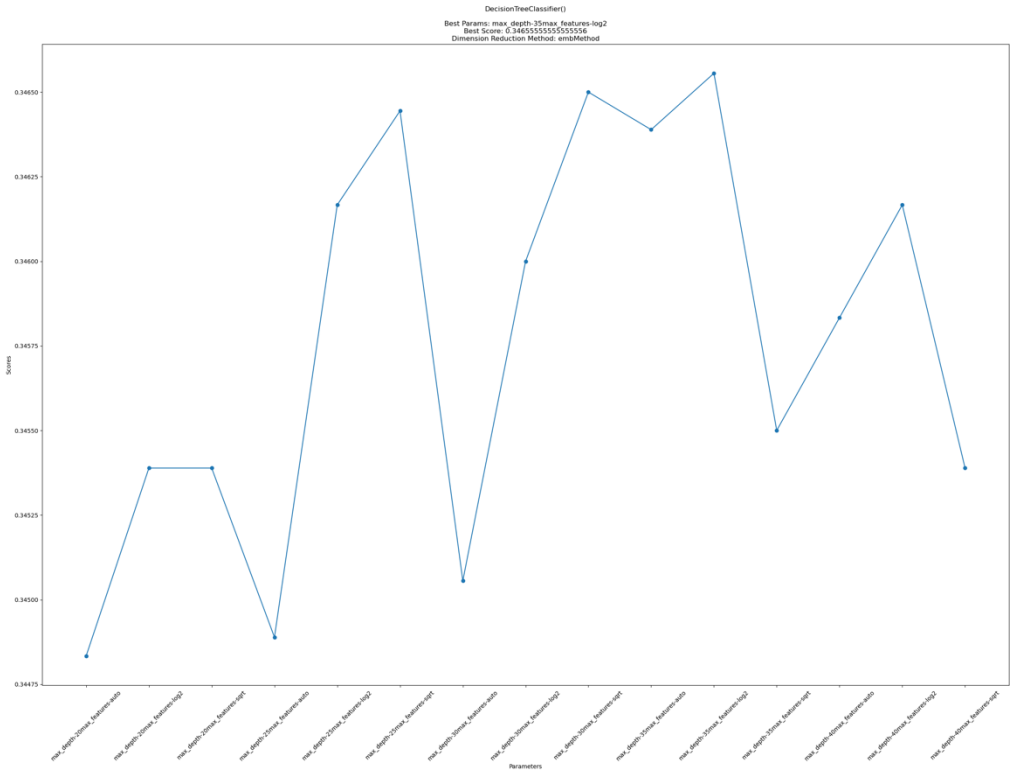
K Neighbors Classifier – Wrapper Feature Selection



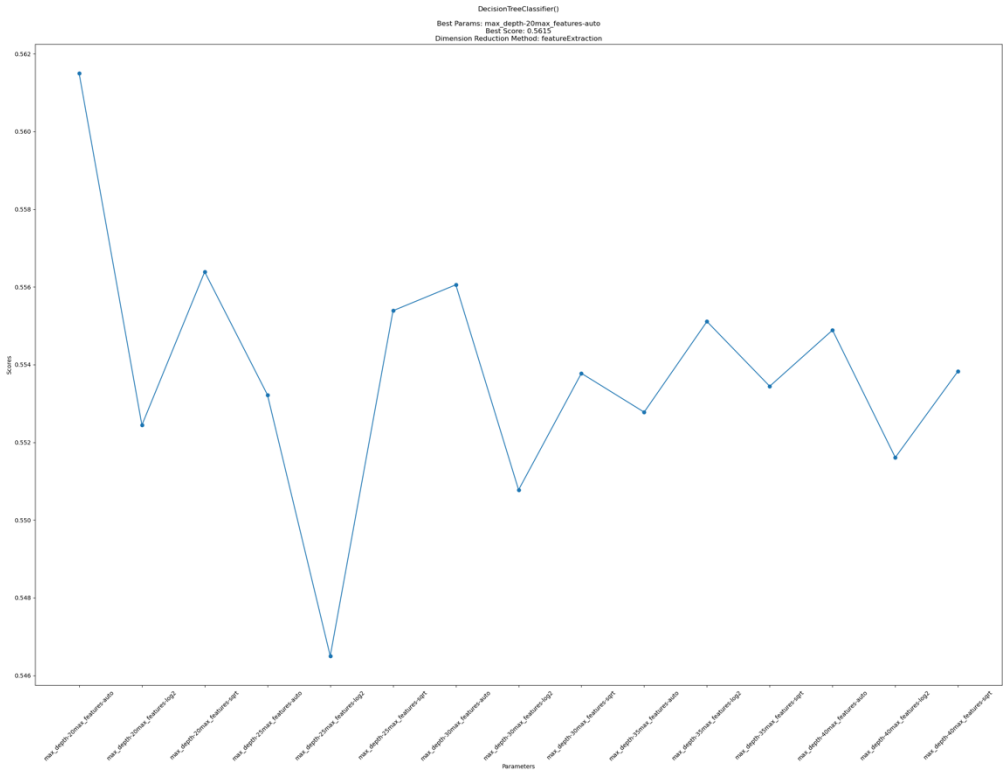
Decision Tree Classifier – None



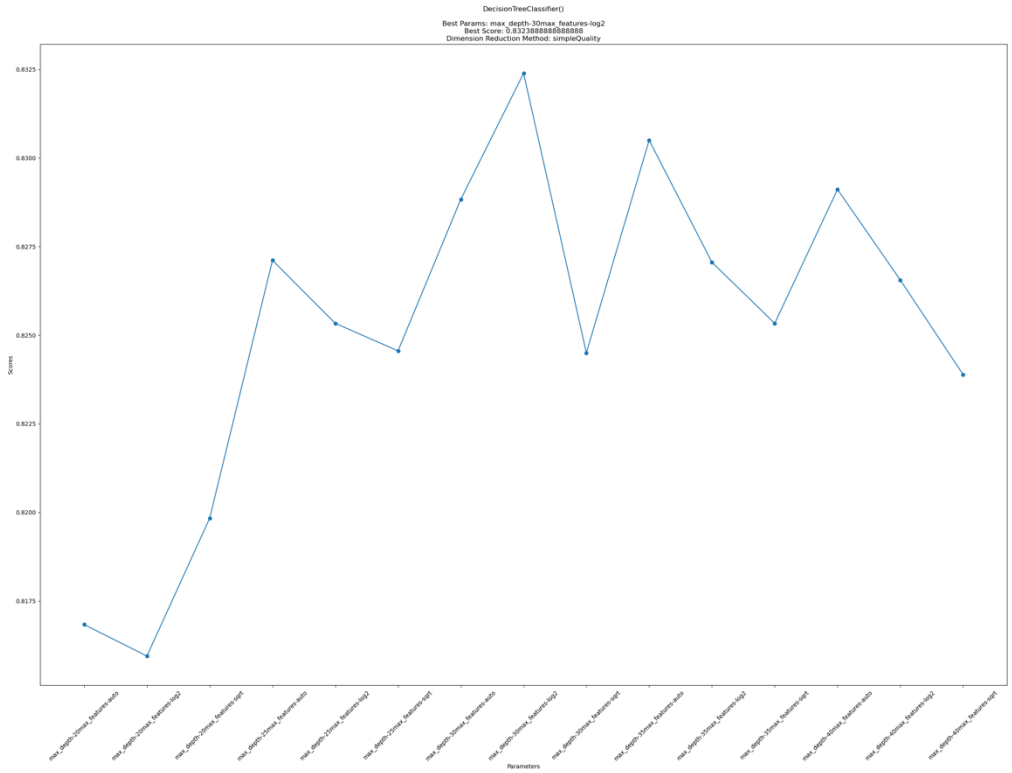
Decision Tree Classifier – Embedded Methods



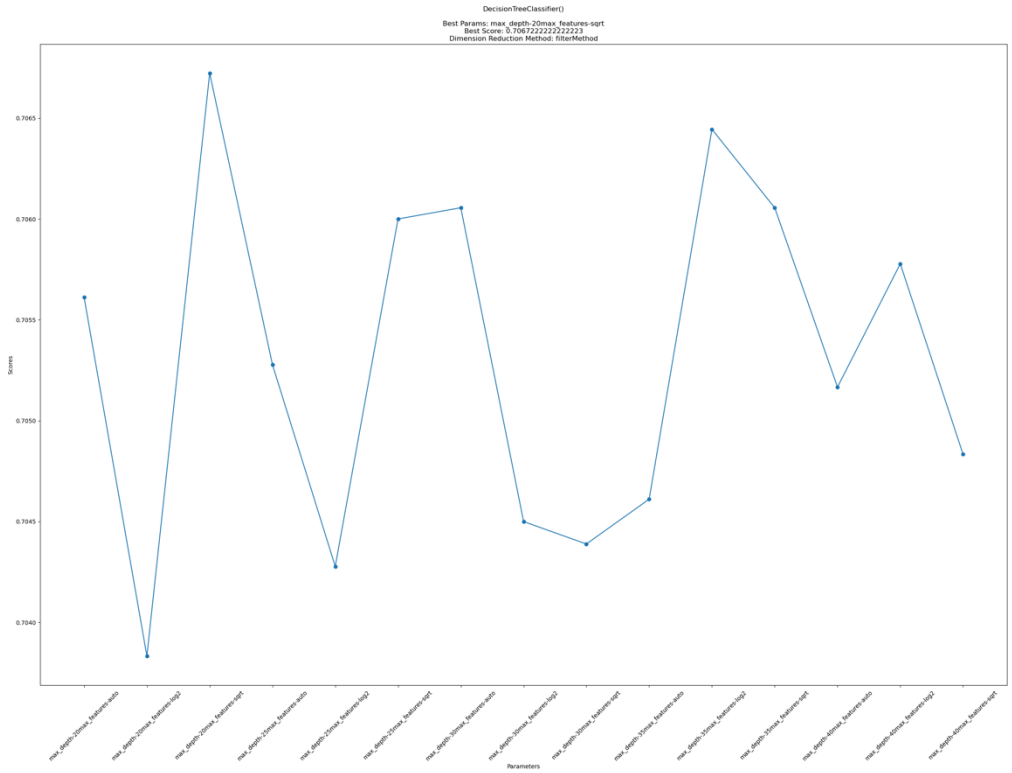
Decision Tree Classifier – Feature Extraction



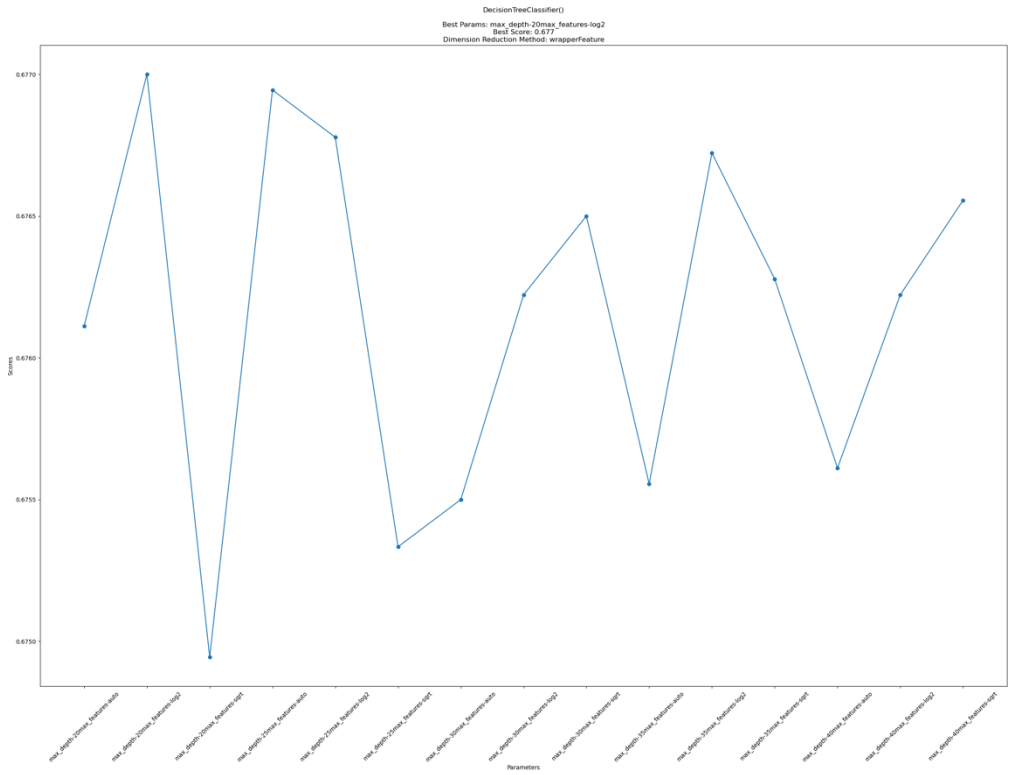
Decision Tree Classifier – Simple Quality Filtering



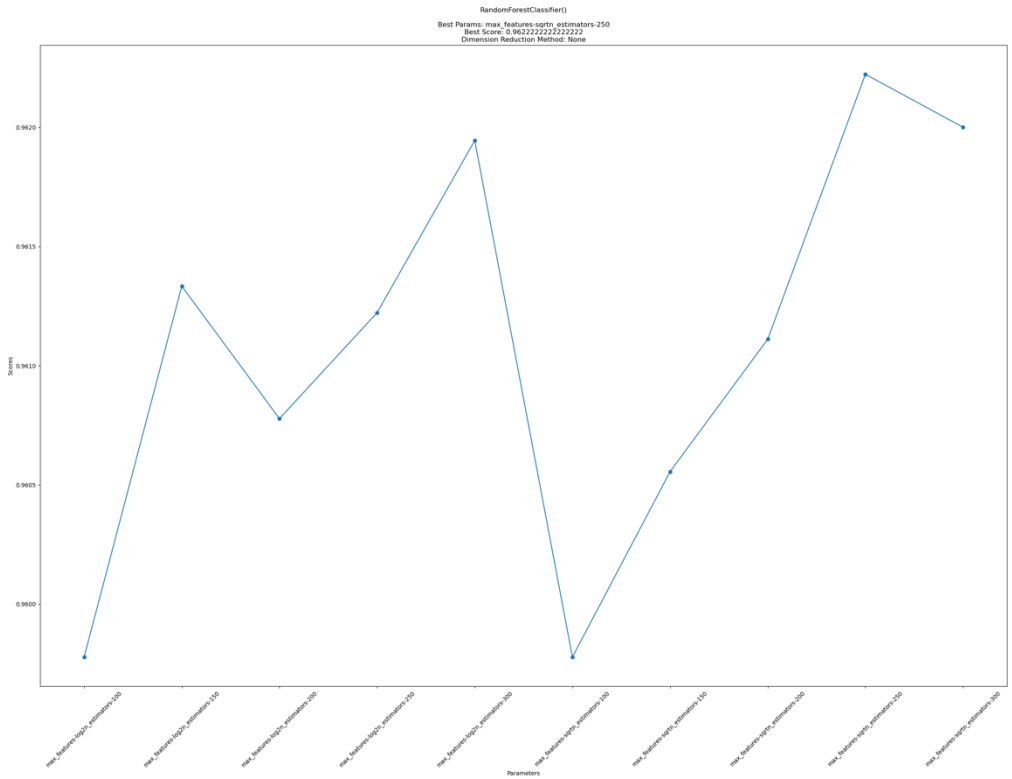
Decision Tree Classifier – Filter Methods



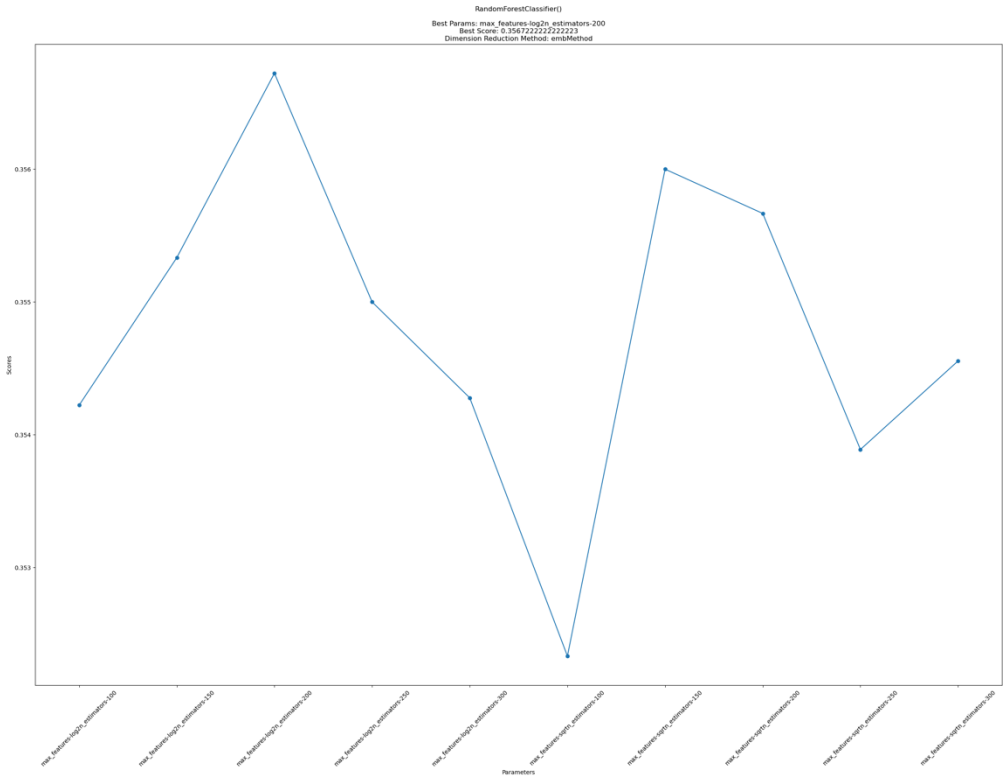
Decision Tree Classifier – Wrapper Feature Selection



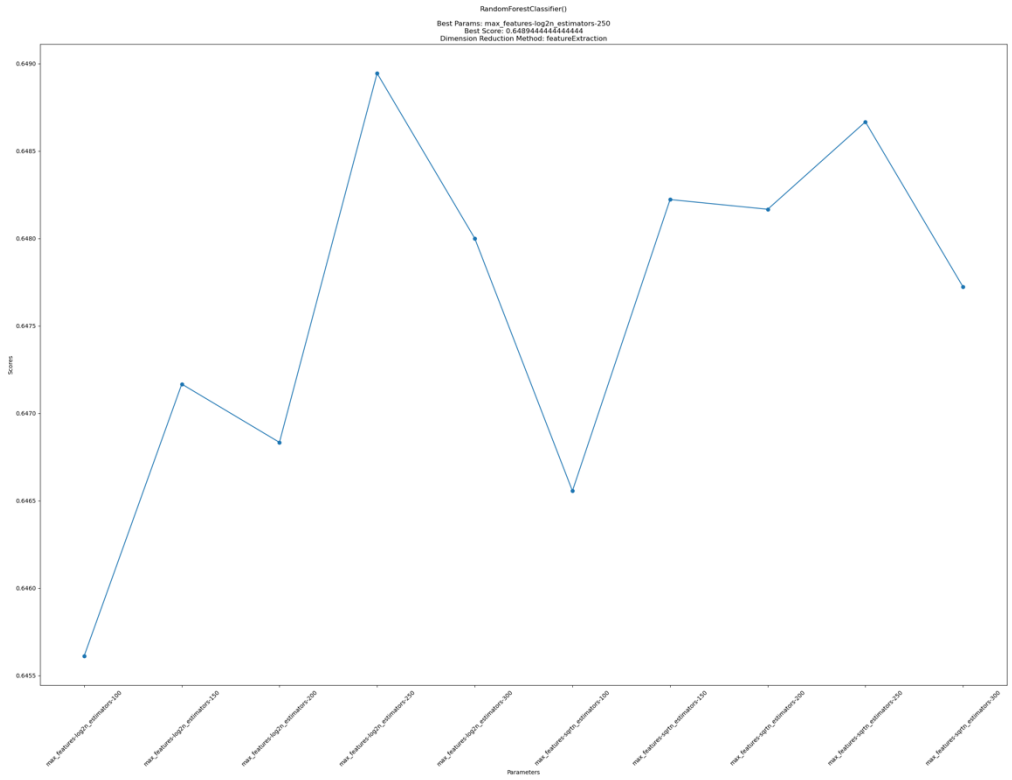
Random Forest Classifier – None



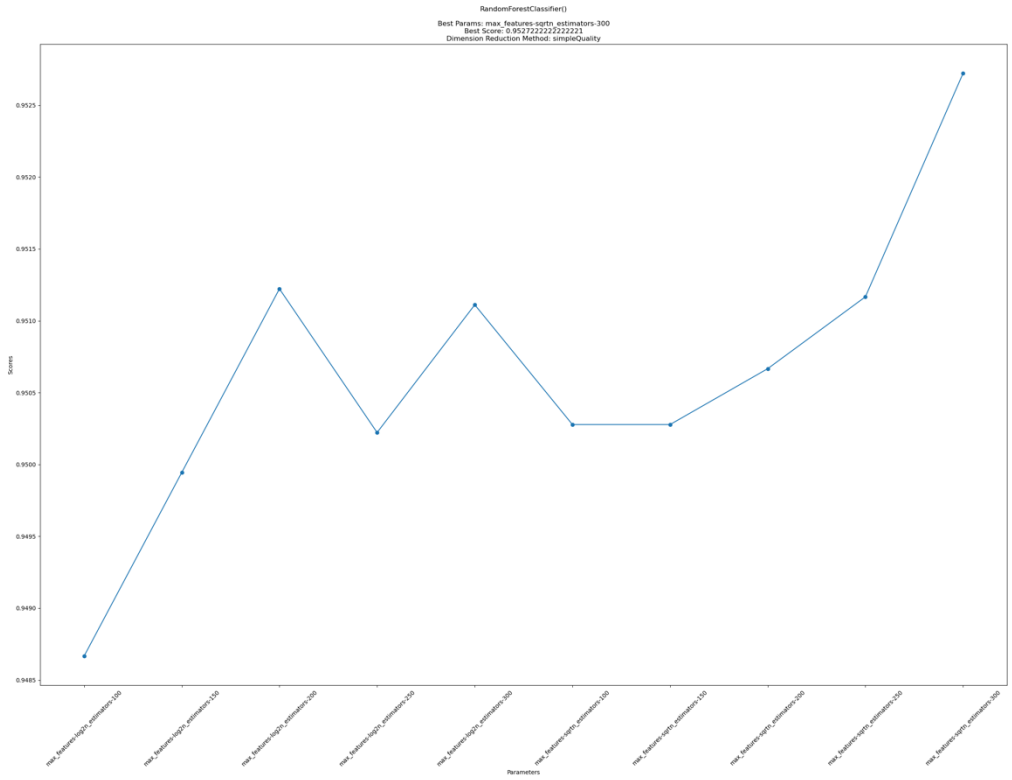
Random Forest Classifier – Embedded Methods



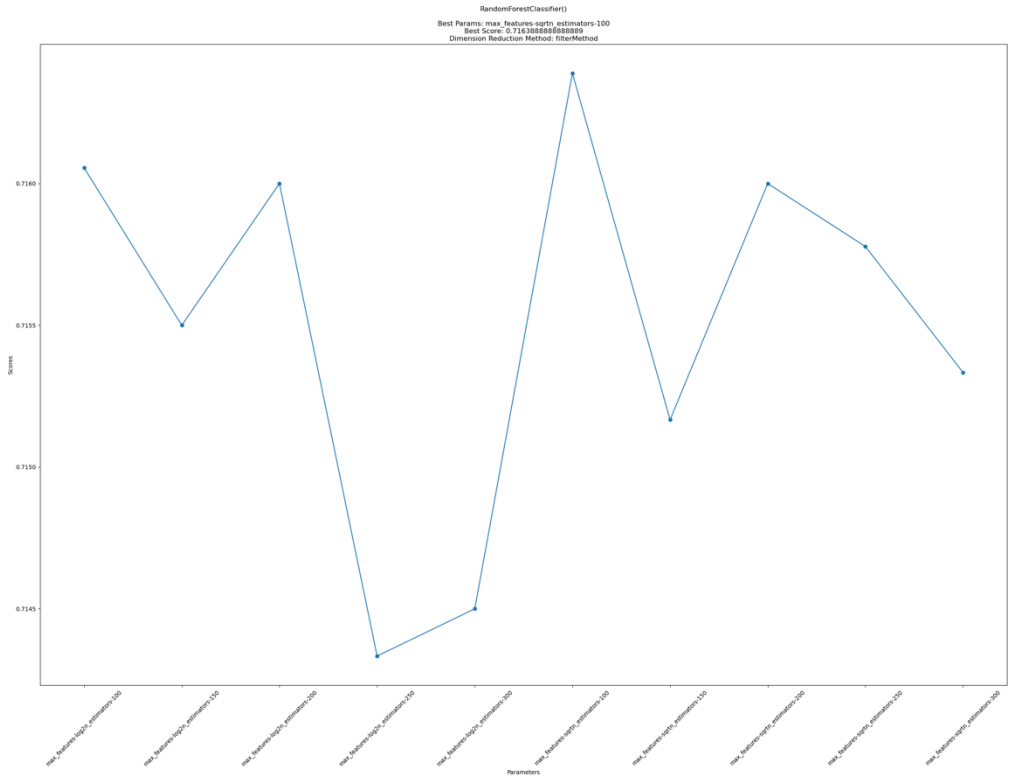
Random Forest Classifier – Feature Extraction



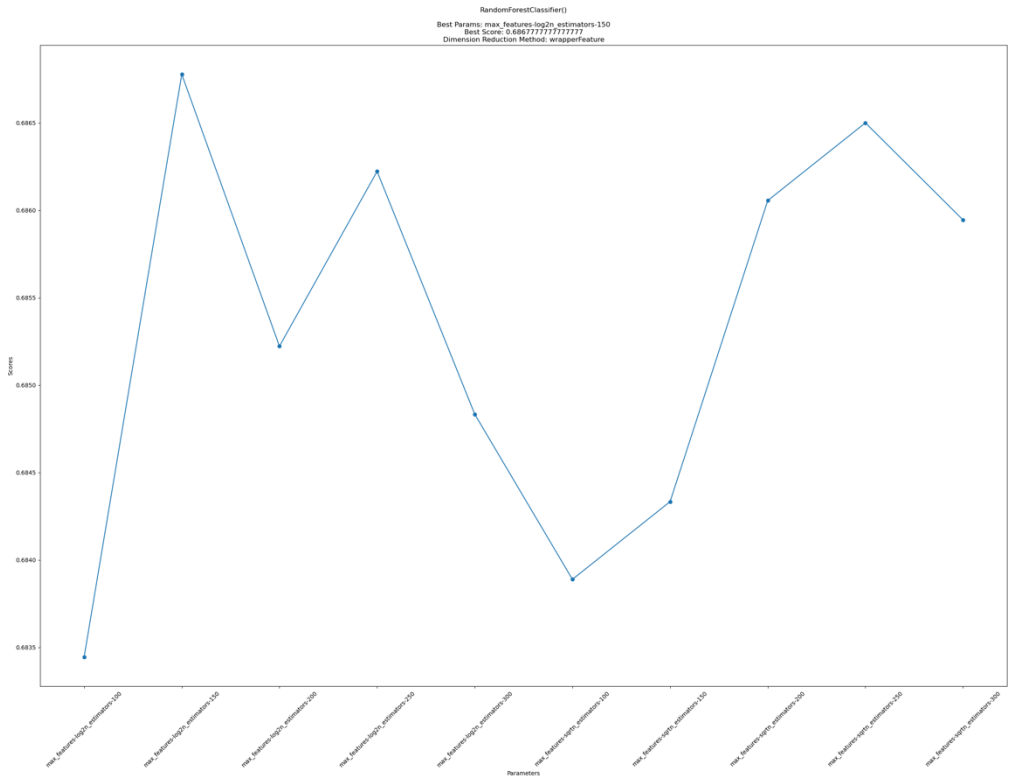
Random Forest Classifier – Simple Quality Filtering



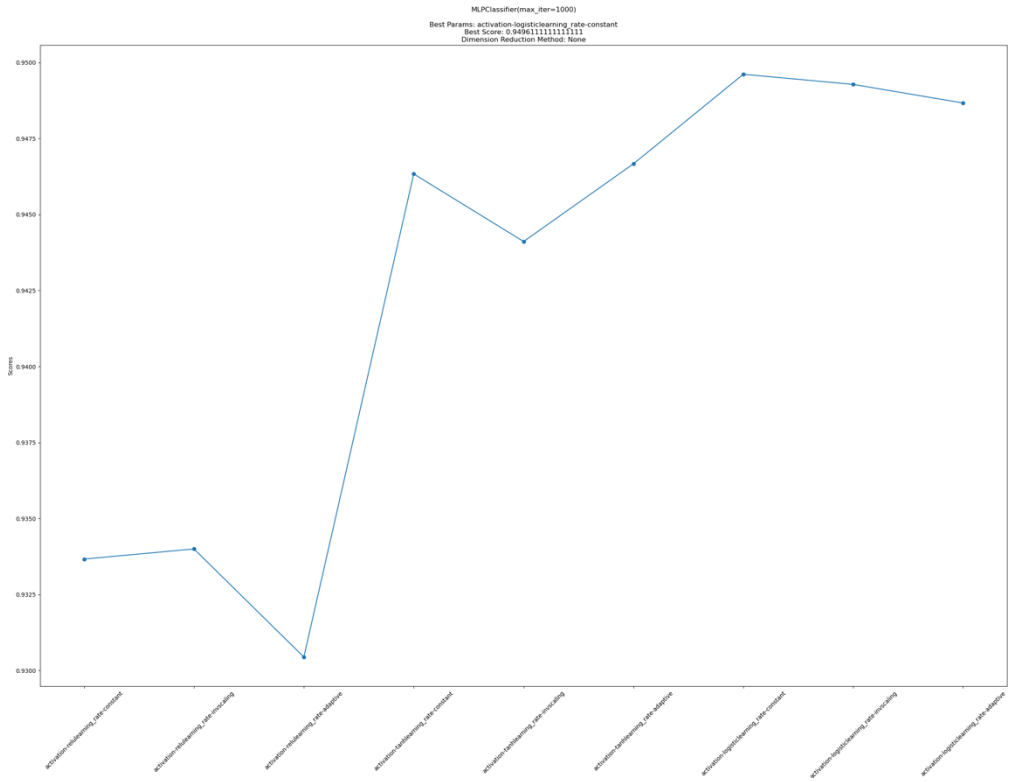
Random Forest Classifier – Filter Methods



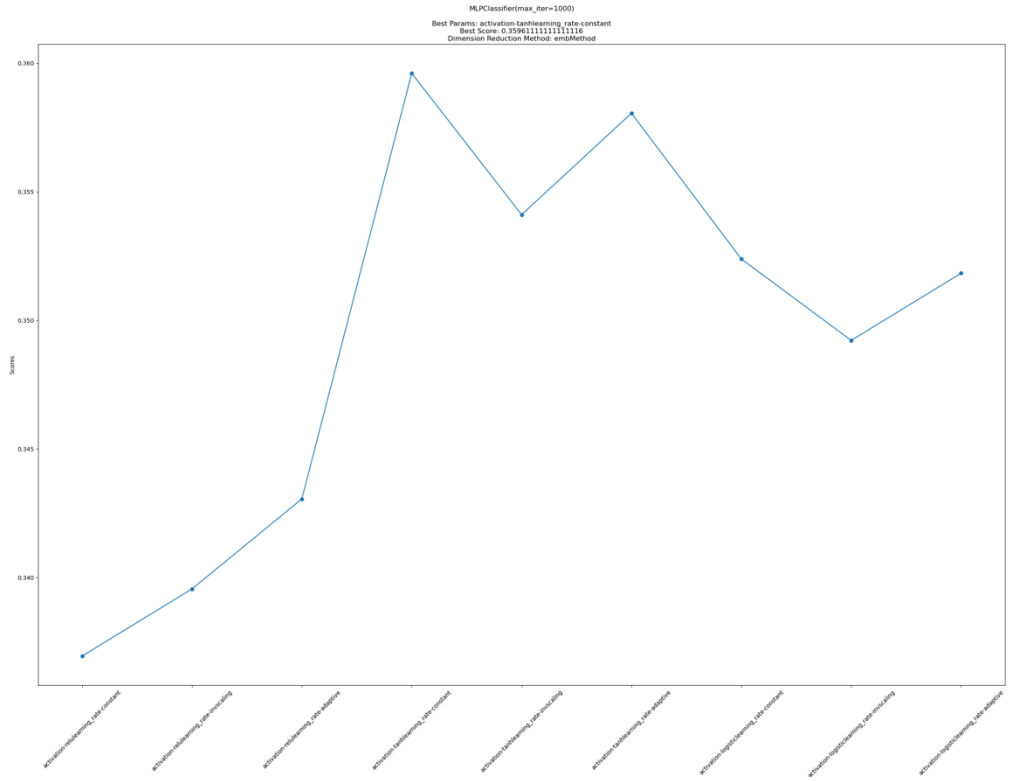
Random Forest Classifier – Wrapper Feature Selection



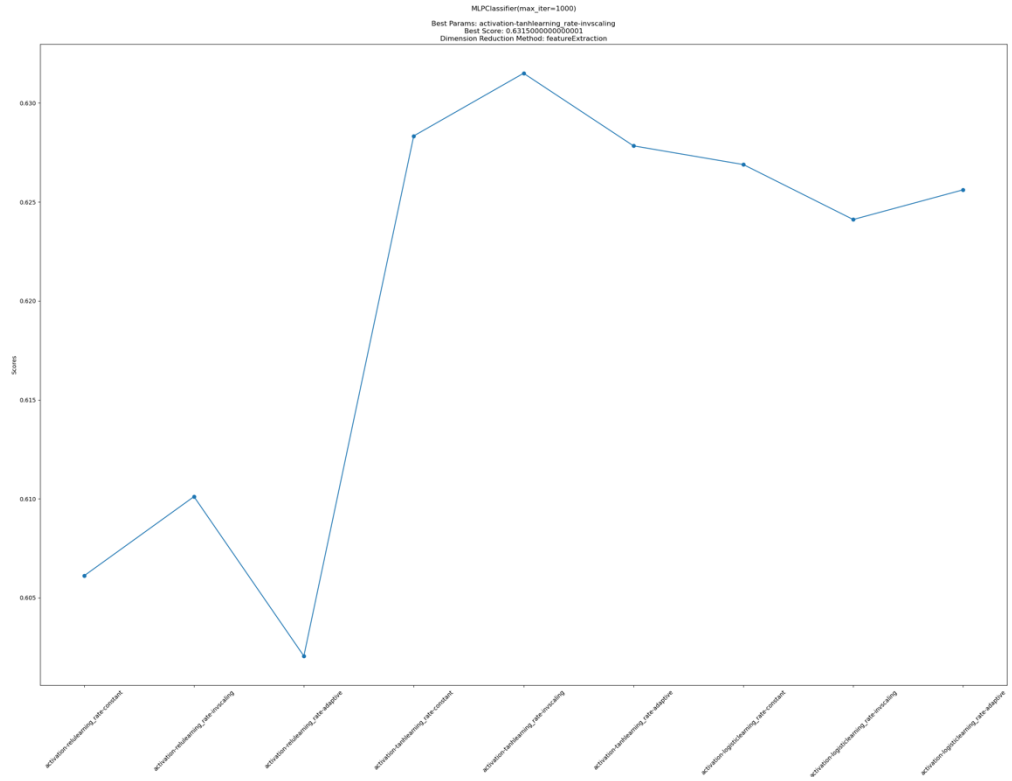
MLP Classifier – None



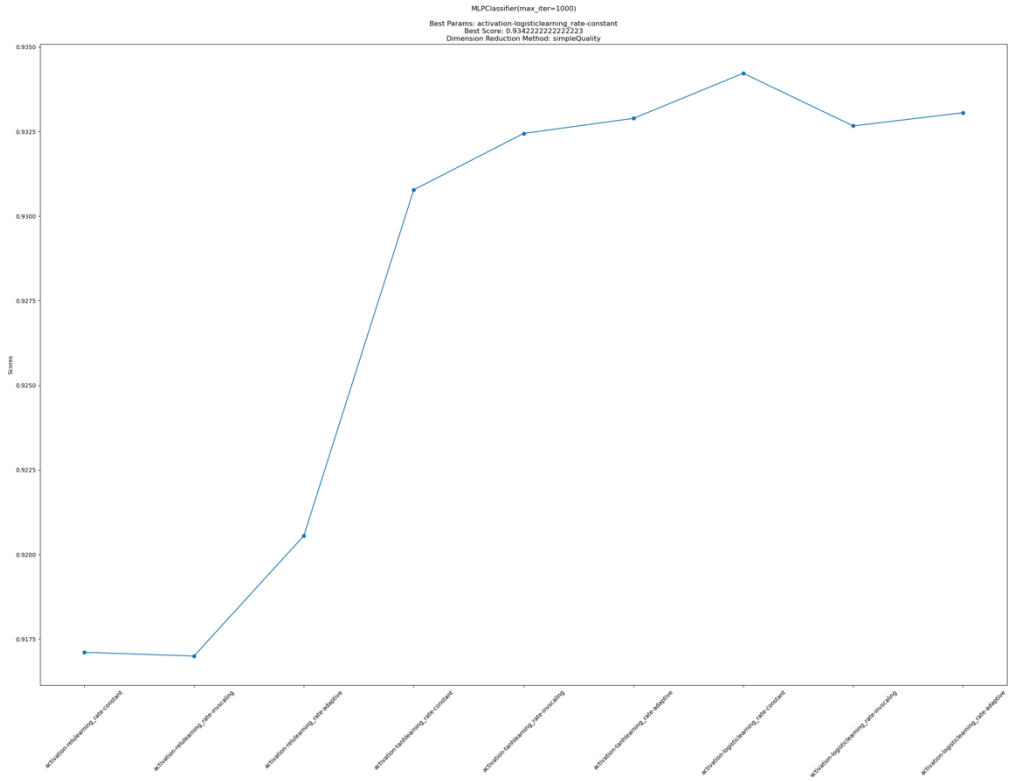
MLP Classifier – Embedded Methods



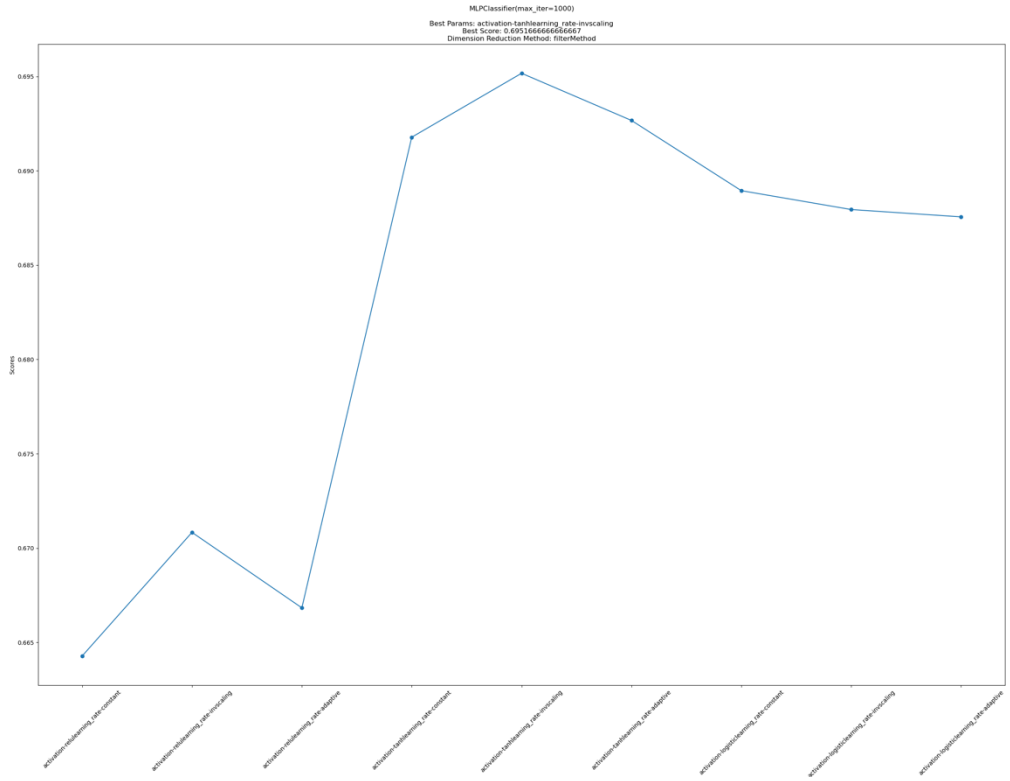
MLP Classifier – Feature Extraction



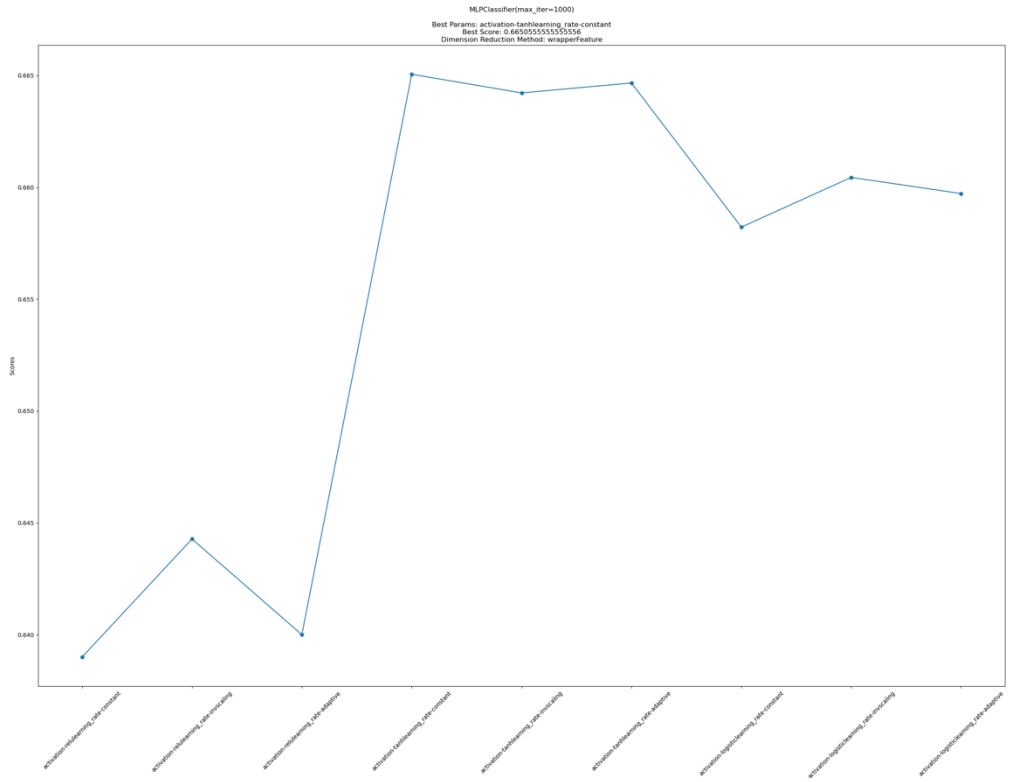
MLP Classifier – Simple Quality Filtering



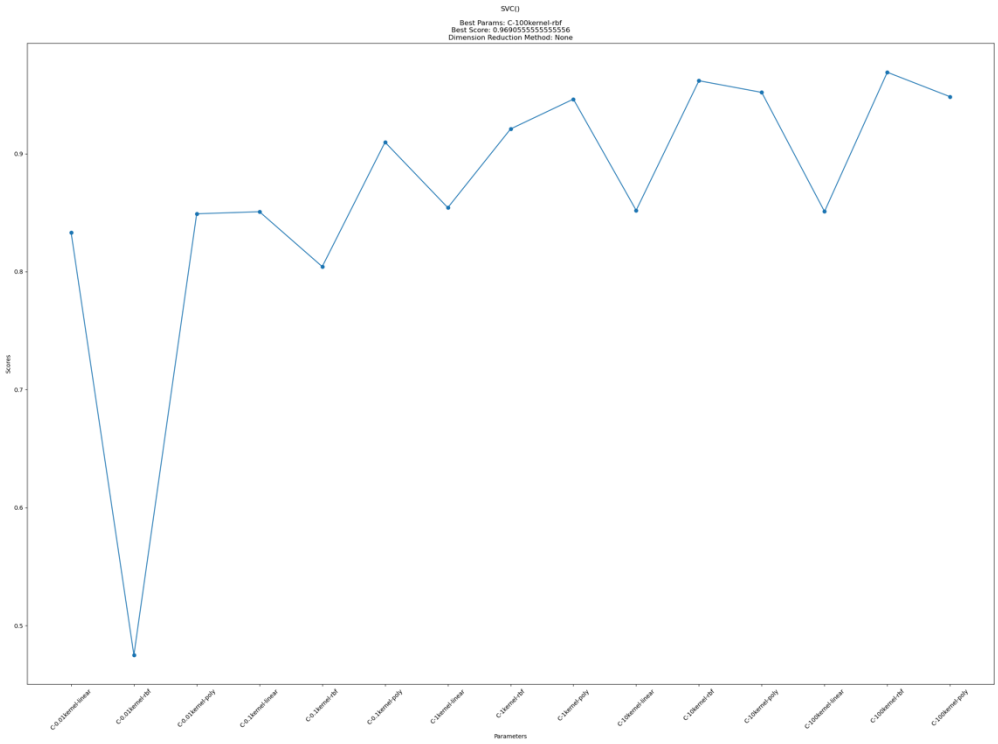
MLP Classifier – Filter Methods



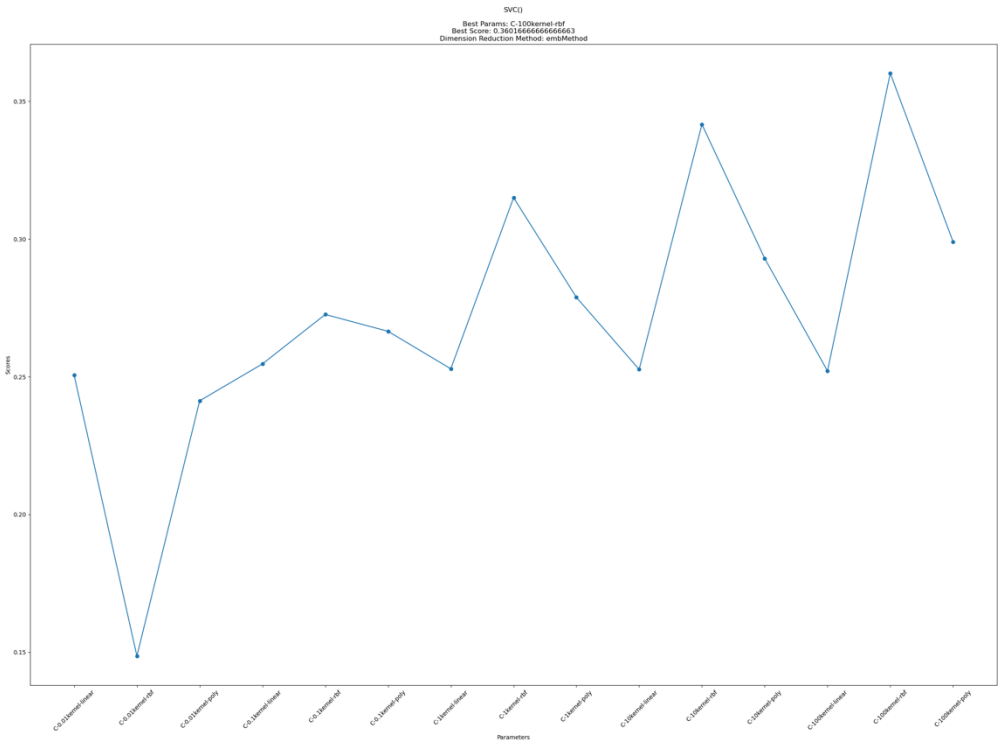
MLP Classifier – Wrapper Feature Selection



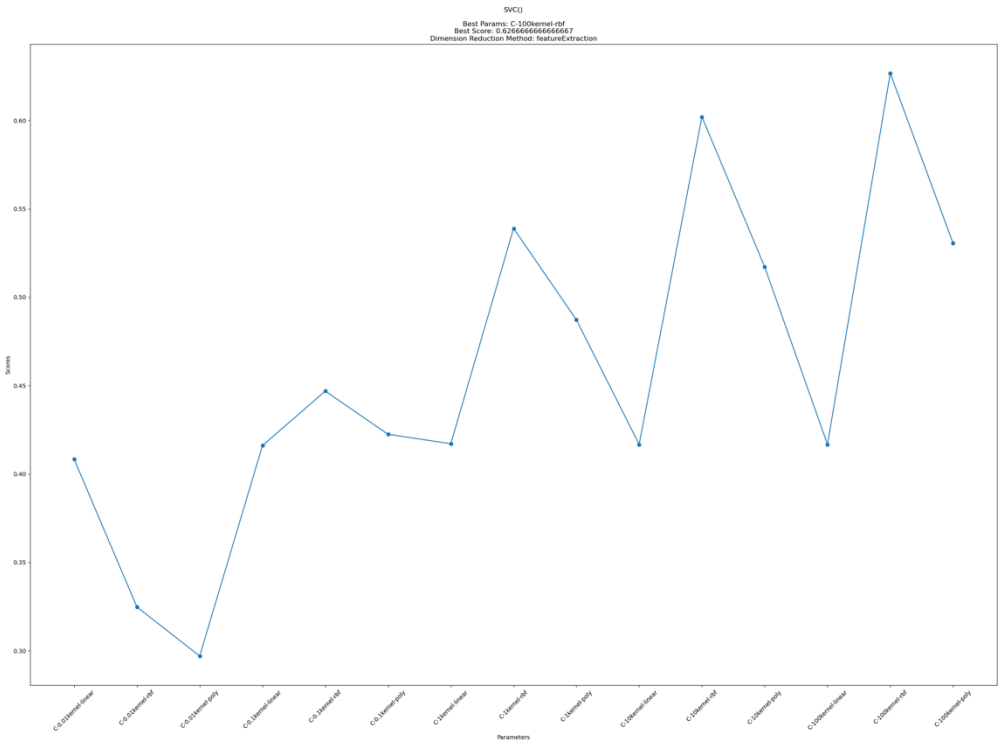
SVM – None


SVC()
Best Params: C-100kernel-rbf
Best Score: 0.9690555555555556
Dimension Reduction Method: None
Scores
0.9
0.8
0.7
0.6
0.5
C-0.01kernel-linear
C-0.01kernel-rbf
C-0.01kernel-poly
C-0.1kernel-linear
C-0.1kernel-rbf
C-0.1kernel-poly
C-1kernel-linear
C-1kernel-rbf
C-1kernel-poly
C-10kernel-linear
C-10kernel-rbf
C-10kernel-poly
C-100kernel-linear
C-100kernel-rbf
C-100kernel-poly
Parameters

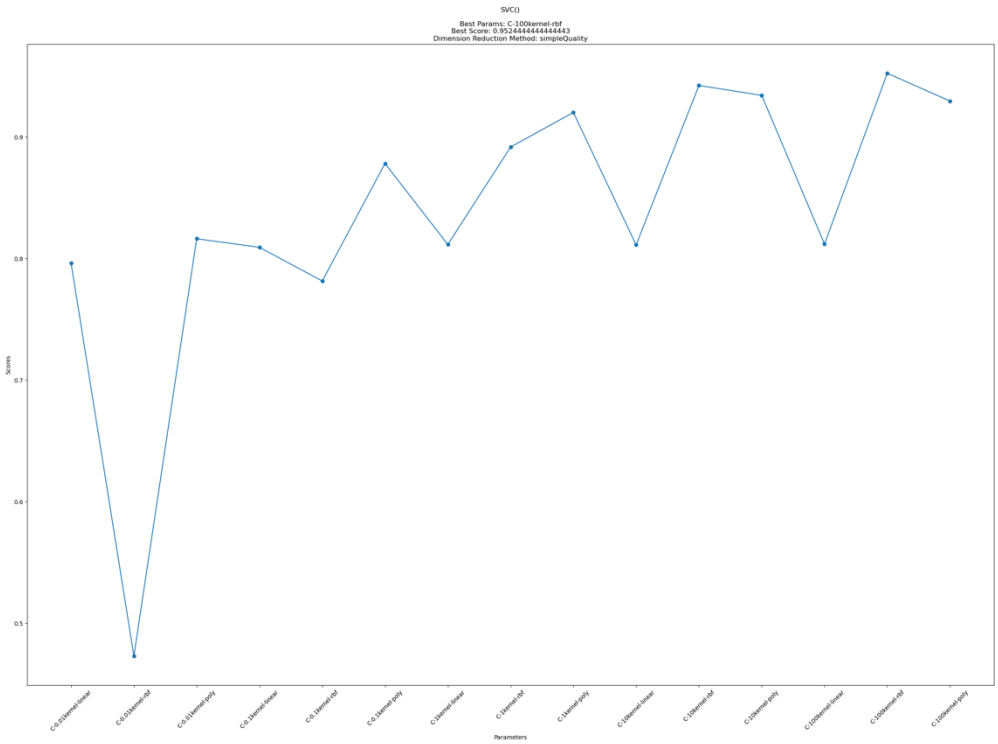
SVM Classifier – Embedded Methods



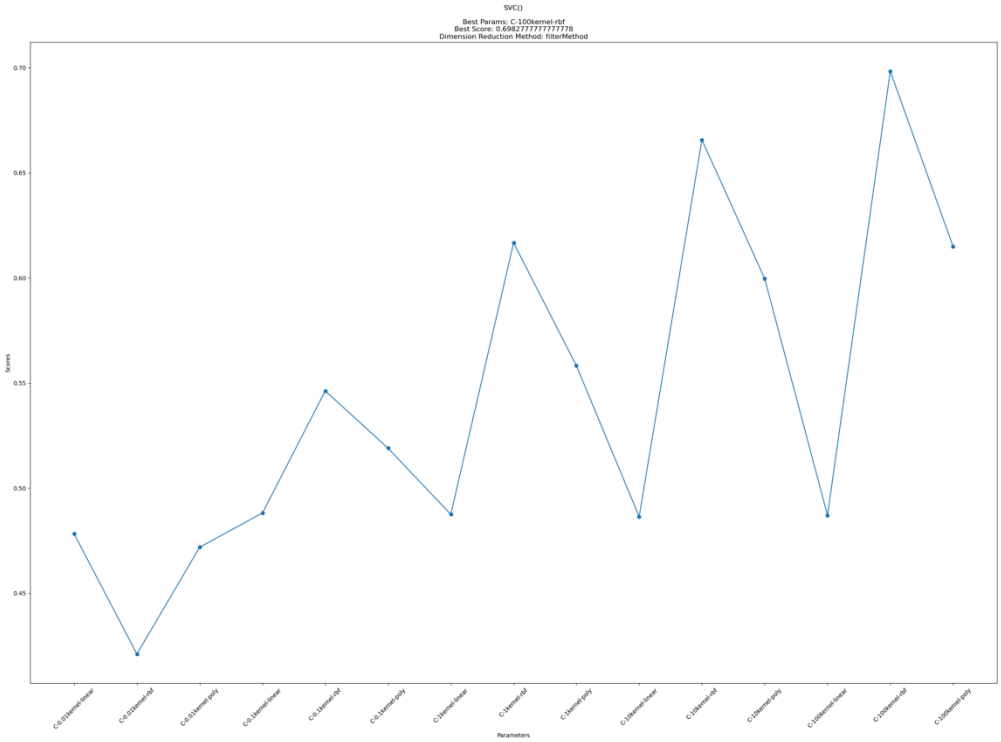
SVM Classifier – Feature Extraction



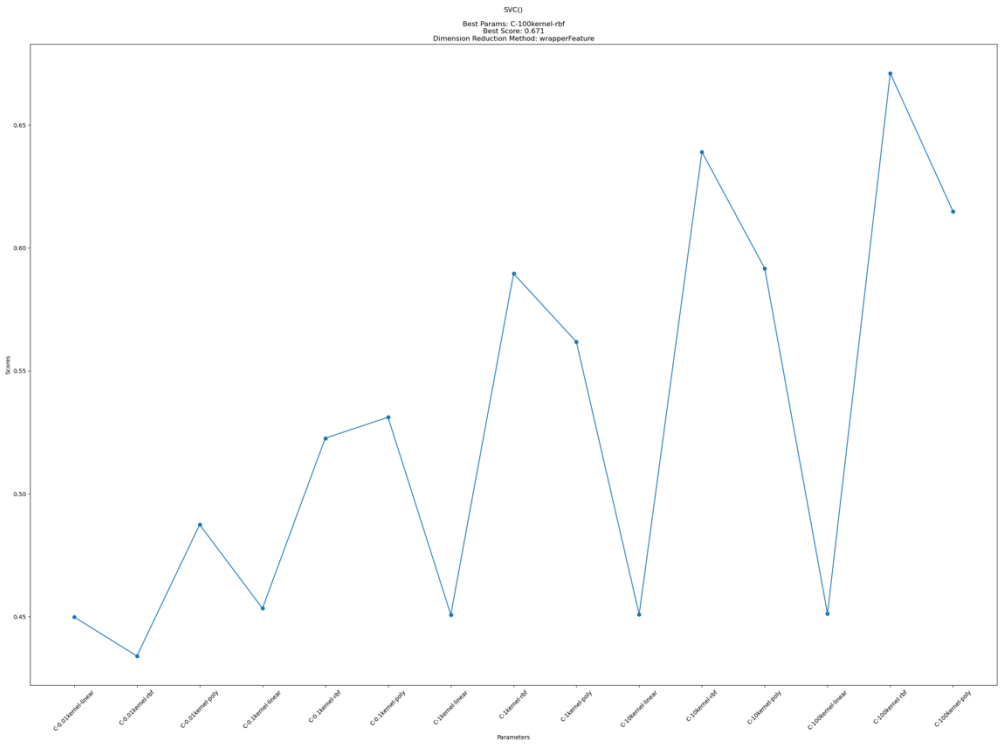
SVM Classifier – Simple Quality Filtering



SVM Classifier – Filter Methods



SVM Classifier – Wrapper Feature Selection



Discussion

1. Comparison of performance and run time

Pairs	*Models*	*Training Time*	*Run Time (sec)*	*Test Scores*
H & K	K Neighbors Classifier	0.48	0.04	0.92
	Decision Tree Classifier	0.12	0.01	0.91
	Random Forest Classifier	16.82	0.16	0.95
	SVM	0.89	0.02	0.98
	MLP Classifier	142.98	0.01	0.96
	AdaBoost Classifier	13.8	0.10	0.86
M & Y	K Neighbors Classifier	0.45	0.05	0.99
	Decision Tree Classifier	0.10	0.00	1.0
	Random Forest Classifier	15.61	0.27	0.99
	SVM	0.28	0.03	0.99
	MLP Classifier	84.98	0.01	1.0
	AdaBoost Classifier	14.01	0.12	0.98
R & V	K Neighbors Classifier	0.51	0.05	0.985
	Decision Tree Classifier	0.09	0.01	0.98
	Random Forest Classifier	16.01	0.08	0.985
	SVM	0.32	0.02	0.99
	MLP Classifier	102.12	0.01	1.0
	AdaBoost Classifier	13.79	0.17	0.98
26 Classification	K Neighbors Classifier	1.48	0.41	0.79
	Decision Tree Classifier	0.42	0.25	0.81
	Random Forest Classifier	79.93	0.39	0.71
	SVM	1.52	0.40	0.74
	MLP Classifier	358.21	0.90	0.89
	AdaBoost Classifier	51.98	0.66	0.32

2. Performance and run-time comparison (after dimension reduction)

Pairs	*Models*	*Training Time*	*Run Time (sec)*	*Test Scores*
H & K	K Neighbors Classifier	0.37	0.04	0.92
	Decision Tree Classifier	0.12	0.00	0.91
	Random Forest Classifier	20.78	0.36	0.92
	SVM	0.89	0.05	0.92
	MLP Classifier	74.4	0.00	0.92
	AdaBoost Classifier	12.8	0.09	0.86
M & Y	K Neighbors Classifier	0.37	0.04	0.99
	Decision Tree Classifier	0.10	0.00	0.98
	Random Forest Classifier	18.69	0.26	0.99
	SVM	0.38	0.06	0.99
	MLP Classifier	44.33	0.01	0.985
	AdaBoost Classifier	13.21	0.13	0.98
R & V	K Neighbors Classifier	0.37	0.04	0.985
	Decision Tree Classifier	0.10	0.01	0.98
	Random Forest Classifier	18.68	0.08	0.985
	SVM	0.35	0.02	0.99
	MLP Classifier	62.55	0.02	0.985
	AdaBoost Classifier	13.10	0.10	0.98
26 Classification	K Neighbors Classifier	1.48	0.49	0.70
	Decision Tree Classifier	0.42	0.21	0.67
	Random Forest Classifier	79.93	0.36	0.77
	SVM	1.52	0.25	0.7135
	MLP Classifier	258.21	0.89	0.86
	AdaBoost Classifier	51.98	0.61	0.27

3. Lesson learned:

a. What model would you choose for this problem

We could choose ANN (MLP Classifier) for this problem.

b. Why?

Because ANN has higher accuracy compared with other classifiers. One thing that needs to be clarified for ANN is: it might need more training time. Although the training time of the ANN model is higher than other classifiers, the amount of data in this dataset is small enough and the dimensionality is properly small. Thus, ANN can still perform well in this question and have proper time-consuming. We believe it is worthwhile spending more time to achieve higher accuracy.

c. How did dimension reduction affect the accuracy and/or run times of the different classifiers?

This dimension reduction reduces the time-consuming, and it also reduces the accuracy. First, dimension reduction reduces the data dimension from 16 to 4, reduces the data set dimension and it greatly reduces the complexity of the data, so it can reduce the run time. However, it also brings a question about that much useful information from the data is removed. That is the major reason why dimension reduction caused the decrease if the accuracy of the model.

d. What would you do differently if you were given this same task for a new dataset? Include at least one additional topic of discussion.

First, we will clean the dataset first without using it directly. We need to remove the noise data. And we also need to convert the data into the proper data type, supplement the null value and fix other problems to make the dataset into a proper, clean, and trainable data set. Moreover, we will choose the appropriate classifier which is suitable for the complexity of the dataset. Due to the large time consuming, ANN is not suitable for the large dataset.

e. (+4 bonus points) for including the additional classifiers in your discussion

We compared every models among the six models and thought the ANN is the best.

f. (+2 bonus points) for including the multi-class classifier in your discussion

The classifiers mentioned above can all handle multi-class classification, but some may be better suited for certain types of data or problems. For this task, I will choose ANN (MLP Classifier). Because although it takes a longer time to train than other models, it makes a better performance than others either. And a few minutes are totally acceptable.