**Group 1**

Pegah Naghahnejad

Gabriela Theis Marchan

Kyle Territo

Lina Rojas

Toheeb Balogun

Michael G. Khalil

**Data-Driven Predictive Modeling for Chemical Plants:**

**A Comparative Analysis of ML Models**

CSC 7333: Machine Learning Course

Instructor: Jianhua Chen

Louisiana State University

Table of Contents

[1. Motivation 3](#_Toc151909592)

[2. Key Research Question 4](#_Toc151909593)

[3. Methods 5](#_Toc151909594)

[3.1. Dataset 5](#_Toc151909595)

[3.1.1 Variables Description 6](#_Toc151909596)

[3.2. Design Implementation 7](#_Toc151909597)

[3.2.1. Unsupervised Learning 7](#_Toc151909598)

[3.2.2. Supervised Learning 10](#_Toc151909599)

[4. Experimental Design 14](#_Toc151909600)

[4.1. Results and Discussion 14](#_Toc151909601)

[4.1.1. Unsupervised Learning: DR and clustering 14](#_Toc151909602)

[4.1.2. Supervised Learning: Artificial Neural Network (ANN) 15](#_Toc151909603)

[4.1.3. Supervised Learning: Deep Neural Network (DNN) 18](#_Toc151909604)

[4.1.4. Supervised Learning: Random Forest (RF) 21](#_Toc151909605)

[4.1.5. Supervised Learning: Support Vector Machine (SVM) 22](#_Toc151909606)

[4.1.6. Comparisons between different machine learning methods 23](#_Toc151909607)

[5. Summary 24](#_Toc151909608)

[References 25](#_Toc151909609)

* [Jianhua Chen](https://moodle.lsu.edu/user/view.php?id=23427&course=39354) - Sat, Dec 2, 2023, 10:03 PM

Chemical plant modeling is an important topic with many useful applications. The most interesting part of this study is that it applied a combination of un-supervised learning and supervised learning.

* [Picture of Jianhua ChenJianhua Chen](https://moodle.lsu.edu/user/view.php?id=23427&course=39354) - Sat, Dec 2, 2023, 10:04 PM

Your written report is missing the ABSTACT ! The report clarity also needs improvement. The reader has to struggle through the report to “dig” out extactly what is the target variable (XMEAS10) for your supervised learning part. And yet it is not clear WHAT ARE the input variables to your supervised learning part. And it is also not clear whether the prediction of the target variable value at time t will use the PAST time (time t-1, t-2, …) values of the input variables. I guess not… Moreover, it is not so clear how you split the training/testing set in your supervised learning part – randomly, or use early time points as training, and later time points as testing?

* [Picture of Jianhua ChenJianhua Chen](https://moodle.lsu.edu/user/view.php?id=23427&course=39354) - Sat, Dec 2, 2023, 10:05 PM

The “Key research questions” section included some statements that sounded like key question studied in the project is the development of “PacMAP” and “HDBSCAN” methods, which is really misleading.  
The video presentation has better clarity compared to the written report.

**ABSTRACT**

Chemical plants generate vast amounts of complex data, encompassing numerous variables. This data, when leveraged through machine learning (ML), can significantly enhance process monitoring, safety, cost optimization, maintenance scheduling, and quality assurance. Process monitoring, in particular, offers insights into the plant's dynamic evolution, influenced by operational shifts or sensor malfunctions—patterns that are challenging to discern in large datasets without ML.

This project focuses on the application of machine learning to enhance operations in chemical plants. Leveraging unsupervised learning techniques such as dimension reduction and clustering, we aim to simplify complex datasets and identify inherent data patterns. Based on these findings, predictive modeling (supervised learning) can be done. In this case, we hope to find a key feature that signals a deviation from normal operating conditions towards a potential fault. Regression models can then be used for predicting future measurements. These predictions serve as “soft-sensors” for this fault-prone feature, facilitating preemptive maintenance and improving operational efficiency.

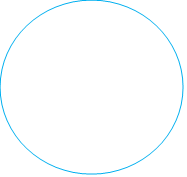
The primary objective of the project is to perform a comparative analysis of different types of regression models to construct the most suitable soft-sensor. Training and evaluation will be conducted using real plant data. Industry-standard machine learning tools and libraries in Python-- including scikit-learn, PyTorch, and TensorFlow-- will be utilized for the development and analysis of the models.

*DR + Clustering*

*Predictive Modeling*

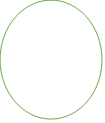
*Find Fault Contributors*

NORMAL





FAULTY



***Figure 1:*** *Overview of the schematic process*

# Motivation

Chemical plants generate vast amounts of complex data, encompassing numerous variables. This data, when leveraged through machine learning (ML), can significantly enhance process monitoring, safety, cost optimization, maintenance scheduling, and quality assurance. Process monitoring, in particular, offers insights into the plant's dynamic evolution, influenced by operational shifts or sensor malfunctions—patterns that are challenging to discern in large datasets without ML.

To effectively navigate this data landscape, unsupervised learning techniques, such as dimension reduction (DR) and clustering, are employed to reduce complexity and identify inherent data groupings. These techniques pave the way for the development of supervised learning models (classification and regression) so that we can identify faulty points corresponding to sensors measured in your variable. Classification models are crucial for differentiating between steady-state and faulty conditions in the plant, facilitating early fault detection. Conversely, regression models are pivotal in creating soft-sensors for plant operations, enabling the prediction of features correlated with faults, thus enhancing preventive measures and overall plant efficiency and reliability.

Aiming for superior performance and accuracy, a comparative analysis of several ML models will be conducted. This evaluation, based on criteria such as precision, computational efficiency, and reliability, will determine the most suitable model for predicting the target variable in a particular chemical plant case study.

This project focuses on the application of machine learning (ML) to enhance operations in chemical plants. Leveraging unsupervised learning techniques such as dimension reduction and clustering, we aim to simplify multivariable datasets that gather information related to operational conditions such as flow, rate, temperature, pressure, and level, and identify inherent data patterns. Subsequently, supervised learning models for classification and regression will be developed. Classification models will enable the early detection of faulty conditions, while regression models will create soft-sensors for predictive maintenance and enhanced operational efficiency. The project's primary objective is to conduct a comparative analysis of various ML models to determine the most suitable approach for predicting target variables in real-world chemical plant operations. Simulated plant data will be used for training and evaluation, with Industry-standard ML tools and libraries, such as scikit-learn, PyTorch, TensorFlow, and Python, which will be employed for model development and analysis.

# Key Research Question

Statistical Process Control (SPC) concepts and methods play a crucial role in monitoring the stability of manufacturing and process industries. Their primary goal is to evaluate whether a process remains in a “state of statistical control” over time. Previous studies in fault detection for multivariable processes employed methods such as Principal Component Analysis (PCA) and Partial Least Squares (PLS) for dimensionality reduction (MacGregor).

However, PCA, being a second-order method, focuses only on the mean and variance-covariance of data. While effective in many cases, PCA has limitations in providing high-order representations for non-Gaussian data, as often observed in our specific data. PCA’s objective is limited to decorrelating variables rather than ensuring their independence.

Recent advancements have introduced methods based on modified Independent Component Analysis (ICA). This approach utilizes modified ICA to extract dominant independent components from normal operational process data, showing superiority over PCA in fault detection and diagnosis (Lee).

Recognizing the limitations of existing methods and the importance of comprehensive multivariable statistical analysis, we propose the integration of another dimensionality reduction method: PaCMAP. Pairwise Controlled Manifold Approximation is a dimensionality reduction method that can be used for visualization, preserving both local and global structure of the data in original space. PaCMAP optimizes the low dimensional embedding using three kinds of pairs of points: neighbor pairs (pair\_neighbors), mid-near pair (pair\_MN), and further pairs (pair\_FP). PaCMAP aligns with our objective as we seek to comprehend the overall organization of the data, understand relationships between the near data points, and identify influential variables affecting the overall process. Additionally, PaCMAP maintains a pairwise-controlled approach during dimensionality reduction, providing better visualization capabilities. This choice is driven by our interest in obtaining a comprehensive view of the data’s structure and relationships, crucial for effective fault detection in industrial processes.

Our proposed clustering method is HDSCAB, a density-based spatial clustering algorithm. What distinguishes HDBSCAN is its utilization of varying epsilons values and integration of results to identify clustering that maximizes the stability over the epsilons (McInnes). Sensor –collected data can exhibit noise, resulting in non-uniform clusters that represent different stable states, and the faulty states. HDBSCAN proves to be a suitable clustering method for our proposals, given its capability in handling noisy data that may not allign any specific cluster.

# Methods

## Dataset

In this project, data from the Tennessee Eastman Process (TEP) will be used. The TEP simulation is often used when studying process control and fault detection as it represents a virtual plant. The process consists of a reactor, condenser, vapor-liquid separator, recycle compressor, and product stripper as shown in Figure 1(Chen). As this simulation is run, time series data is recorded for many different process features, or tags. In addition, disturbances are introduced, and operational set-points are changed to see how the plant will change over time. The data used in this project has disturbances introduced into the process.

A diagram of a chemical plant

Description automatically generated

Figure 1: Process flow diagram of the TE process.

### 3.1.1 Variables Description

This multivariable system comprises two types of variables: manipulated variables (Table 1) and measured variables (Table 2). All the variables correspond to continuous numerical variables, each with different scaling. Each variable has a total of 1995 time series records collected.

Table 1: Manipulated variables

|  |  |
| --- | --- |
| **Variable** | **Description** |
| XMV(1) | D feed flow (Stream 2) |
| XMV(2) | E feed flow (Stream 3) |
| XMV(3) | A feed flow (Stream 1) |
| XMV(4) | Total feed flow (Stream 4) |
| XMV(6) | Purge valve (Stream 9) |
| XMV(7) | Separator pot liquid flow (Stream 10) |
| XMV(8) | Stripper liquid product flow (Stream 11) |
| XMV(10) | Reactor cooling water flow |
| XMV(11) | Condenser Cooling Water flow |

Table 2: Manipulated variables

|  |  |  |
| --- | --- | --- |
| **Variable** | **Description** | **Units** |
| XMEAS(1) | A feed (stream 1) | Kscmh |
| XMEAS(2) | D feed (stream 2) | Kg/h |
| XMEAS(3) | E feed (stream 3) | Kg/h |
| XMEAS(4) | Total feed (stream 4) | Kscmh |
| XMEAS(5) | Recycle flow (stream 8) | Kscmh |
| XMEAS(6) | Reactor feed rate (stream 6) | Kscmh |
| XMEAS(7) | Reactor pressure | kPa gauge |
| XMEAS(8) | Reactor level | % |
| XMEAS(9) | Reactor temperature | °C |
| XMEAS(10) | Purge rate (stream 9) | Kscmh |
| XMEAS(11) | Product sep. temperature | °C |
| XMEAS(12) | Product sep. level | % |
| XMEAS(13) | Product sep. pressure | kPa gauge |
| XMEAS(14) | Prod. Sep. underflow (stream 10) | m3/h |
| XMEAS(15) | Stripper level | % |
| XMEAS(16) | Stripper pressure | kPa gauge |
| XMEAS(17) | Stripper unverflow (stream 11) | m3/h |
| XMEAS(18) | Stripper temperature | °C |
| XMEAS(19) | Stripper steam flow | Kg/h |
| XMEAS(20) | Compressor work | kW |
| XMEAS(21) | Reactor cooling water outlet temp | °C |
| XMEAS(22) | Separator cooling water outlet temp | °C |

## Design Implementation

### Unsupervised Learning

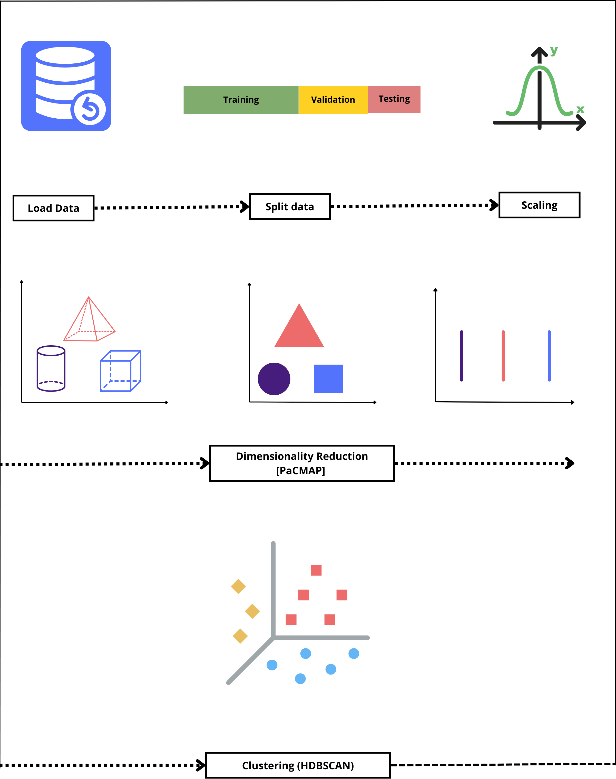


Figure 2: Unsupervised Learning approach.

In the pursuit of robust analysis through unsupervised learning, a crucial step involved splitting the dataset into training and test sets (Figure 2). Approximately 80% of the total data was designated for the training set, while the remaining 20% constituted the test set. Ensuring consistent results across model runs was prioritized by setting the random\_state parameter to 42.

For the input features, diverse sensor measurements were collected, registering different variables characterized by varying measurement units and orders of magnitude. Before embarking on dimensionality reduction, a meticulous preprocessing phase was implemented. This included addressing categorical values and scaling the data. The chosen scaling approach, Min-Max scaling (Eq. 1), stemmed from a prior dataset analysis revealing a limited number of outliers and distinct minimum and maximum values for each variable. This thoughtful preprocessing lays the groundwork for effective unsupervised learning, enhancing the dataset's suitability for subsequent analyses.

|  |  |
| --- | --- |
|  | (1) |

This thoughtful preprocessing lays the groundwork for effective unsupervised learning, enhancing the dataset's suitability for subsequent analyses. The subsequent sections delve into the thoughtful design of dimensionality reduction techniques and clustering methodologies, further enhancing the dataset's suitability for comprehensive unsupervised learning analyses.

#### Dimensionality reduction (DR)

##### Pairwise Controlled Manifold Approximation Projection (PaCMAP)

In the realm of dimensionality reduction, the Pairwise Controlled Manifold Approximation Projection (PaCMAP) algorithm was employed using the "pacmap" library in Python. This technique facilitates a concise representation of the dataset while preserving essential relationships. The algorithm was imported, and the scaled data underwent fitting and transformation with specific parameters detailed in Table 3.

Table 3: PaCMAP default parameters

|  |  |  |
| --- | --- | --- |
| **Parameters** | **Value selected** | **Explanation** |
| n\_components | 2 | To create a simple visual of the data to identify stable and faulty states, 2 components were used |
| n\_neighbors | 10 | The default value is used because the sample size is less than 10,000 |
| MN\_ratio | 0.5 | A low MN\_Ratio was chosen to prioritize capturing more glocal relationships, as MN\_ratio significantly influences the balance between local and global connections |

This approach aims to provide a clear, two-dimensional representation of the data, facilitating the identification of stable and faulty states.

##### PCA

Additionally, Principal Component Analysis (PCA) was another method applied to reduce the dimensionality of the dataset further.

Table 4: PCA default Parameters

|  |  |  |
| --- | --- | --- |
| **Parameters** | **Value selected** | **Explanation** |
| n\_components | 2 | To create a two-dimensional representation of the data. |
| copy | True | The input data is copied. |
| whiten | True | Components are whitened to ensure uncorrelated outputs. |

Therefore, PaCMAP and PCA techniques enhance understanding of the dataset's structure, providing valuable insights into global and local relationships among data points.

#### Clustering

##### HDBSCAN

Hierarchical Density-Based Clustering of Applications with Noise was executed in Python importing the ‘hdbscan’ library. The algorithm was applied to the dimensionality-reduced data considering the following parameters:

Table 5: HDBSCAN default Parameters

|  |  |  |
| --- | --- | --- |
| **Parameters** | **Value selected** | **Explanation** |
| min\_cluster\_size | 5 | After dimensionality reduction, the data suggests that the cluster should contain more than 5 data points. Hence, anything below this threshold can be considered an outliner |
| cluster\_selection\_epsilon | 0 | Given that the formed cluster exhibit numerous subclusters that required to be merged, the default minimum value is sensible to prevent the unnecessary creation of small clusters |

###### K-Means

K-means is a clustering algorithm, and it works selecting initial cluster centroids using sampling based on empirical probability distribution of the points contribution to the overall inertia. The algorithm was implemented in Python using the *sklearn.cluster* library.The parameters selected are as they follow:

Table 6: K-Means Parameters.

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Values** | **Explanation** |
| n\_clusters | 3 | Guided by the visual evaluation of dimensionality reduction performance, we made selections for the clusters |
| n\_init | auto' | We set this value to “auto” to let the algorithm automatically determine the number of runs with different centroid seeds |
| random\_state | 42 | This parameter ensures consistent results each time the code is executed |

#### Comparison of Dimensionality Reduction and Clustering

To qualitatively evaluate and compare the efficacy of dimensionality reduction and clustering methods, two prominent metrics were employed: the David Bouldin Index and the Silhouette Score. These metrics serve as valuable indicators to assess the ability of different methods to reveal intrinsic features of the data post dimensionality reduction.

##### Davies-Bouldin Index

The Davies-Bouldin Index provides a quantitative measure of the average similarity between clusters. It is calculated as the ratio of the inter-cluster distance to the average intra-cluster distance, defined as:

|  |  |
| --- | --- |
|  | (2) |

The explicit equation for the David Bouldin Index is given by Eq. (3):

|  |  |
| --- | --- |
|  | (3) |

Here is the number of clusters, represents the intra-cluster distance within cluster , and is the inter-cluster distance between cluster and .

The ‘davis\_bouldin\_score’ was implemented in Python using the scikit-learn library.

##### Silhouette Score

The Silhouette Score quantifies the fit of each data point within its assigned cluster and measures the separation between clusters. It is calculated using eq. 4,

|  |  |
| --- | --- |
|  | (4) |

Here, represents the average distance from -th data points within the same cluster, and is the smallest average distance from -th data points in a different cluster.

The Silhouette Score was computed in Python using the ‘silhouette\_score’ function from the scikit-learn library.

These metrics, applied to the results of dimensionality reduction and clustering methods, offer a comprehensive understanding of their performance in uncovering meaningful patterns within the dataset.

### Supervised Learning

In the preprocessing phase for supervised learning, the algorithm initiates by scaling the input features using a Min/Max Scaler. This crucial step standardizes the input data, ensuring uniformity across different variables with varying measurement units and orders of magnitude. Subsequently, the dataset undergoes a division into training and testing sets, allocating 80% for training and 20% for testing. This partitioning is fundamental in supervised learning workflows, providing distinct model training and evaluation subsets.

As machine learning algorithms exhibit varied performance depending on the nature of the training data, this study delves into the exploration of multiple approaches to identify the most effective one for the given problem. Four representative algorithms were thoroughly investigated—support vector regression (SVR), decision-based regression (DT), artificial neural networks, and deep neural networks. This comprehensive analysis aims to discern the strengths and weaknesses of each algorithm, laying the foundation for informed model selection in subsequent stages of the study. Hyperparameter optimization was conducted through a grid search, exploring parameters such as the number of estimators, maximum tree depth, and minimum samples required for node splitting. Performance evaluation encompassed both training and testing datasets, employing metrics like Root Mean Squared Error (RMSE) and R-squared (R2). RMSE gauges average prediction deviation, while R-squared measures the model's explanatory power.

#### Artificial Neural Network (ANN)

Artificial neural networks (ANN) are a type of machine learning method that constructs a model as a network of nodes connected by edges arranged in layers (also known as fully connected layers). Each node multiplies the outputs from the previous layer's nodes by their corresponding weights (represented by the edges), adds them, and passes the result through an activation function to feed the next layer of nodes. The predicted output is calculated for a given set of inputs to train ANN, the error between the predicted and true values is computed, and the weights are adjusted accordingly. The high tunability of ANNs, especially their architecture (number of nodes and layers), makes them flexible and thus attractive for a wide variety of applications. However, a considerable drawback is that the amount of data required to train the model effectively increases greatly with model complexity (Figure 3).

A screenshot of a computer

Description automatically generated

Figure 3: Representation of the Machine Learning ANN architecture.

Therefore, the PyTorch library defines the ANN model (Paszke et.al, 2019). This model architecture consists of three fully connected layers with ReLU activation functions, designed for a regression task. The ANN consists of input and output layers and one hidden layer (52, 64, 1). The algorithm iterates through 200 epochs, training the model using an Adam optimizer (Kingma, D. P., & Ba, J., 2014) and calculating the Mean Squared Error loss at each epoch.

#### Deep Neural Network (DNN)

Deep Neural Networks (DNN) represent a more advanced form of artificial neural networks with multiple layers (deep architecture). The deep layers enable DNNs to automatically learn hierarchical features from the data, potentially capturing complex patterns and representations. Despite their impressive capabilities, training deep neural networks requires substantial computational resources and large amounts of labeled data (Figure 4).

A screenshot of a computer

Description automatically generated

Figure 4: Representation of the Deep Learning architecture.

The deep learning model is trained using the PyTorch library (Paszke et.al, 2019). The deep learning architecture contains an input/output layer and four hidden layers (52, 64, 128, 256, 64, 1). Mean Squared Error (MSE) was used as a loss function and it was minimized using the Adam optimizer (Kingma, D. P., & Ba, J., 2014).

#### Random Forest (RF)

The most common implementations use the idea of random forest (RF) in which multiple models (decision tree models) are fit to the training data, and the final prediction is made via a voting mechanism. This approach, typically called ensemble methods, helps prevent overfitting by training multiple models, each with a different bias, and then averaging their predictions. Another advantage of DT is that they tend to perform well with high-dimensional data. Renowned for its robustness and predictive capabilities, the Random Forest algorithm was implemented through the scikit-learn library in Python.

Table 7: Parameters optimized for the RF implementation.

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Values** | **Explanation** |
| n\_estimators | 100,200,300 | **Number of tree in the forest:** To ensure a robust and stable model while minimizing overfitting at the lowest computational cost, this parameter was optimized |
| max\_depth | None, 10, 20,30 | **The maximum depth of the tree:** Optimizing this parameter is essential to capture intricate patterns in the training data throguh tree depth while preventing overfitting |
| min\_samples\_split | 2,5,10 | **The minimum number of samples required to split an internal node:** This parameter requires to be optimized to avoid the model capturing noise and generating overly detailed and complez trees |

#### Support Vector Machine (SVM)

A support vector machine (SVM) is an algorithm that tries to find the hyperplane that best separates two linearly separable regions. The idea of a SVM was first proposed by Boser et al.48 in 1996 and remains a powerful ML modeling method, especially for lowdimensional and small datasets. A useful extension of this method is the introduction of a kernel that transforms non-linearly separable data into linearly separable data by adding an extra dimension. The classification problem, that is, finding the boundary between two or more classes, can be reframed as a regression problem by setting the algorithm to find the two hyperplanes that contain both the training data and the predicted values while minimizing the distance between said hyperplanes.49 This latter implementation is called "Support Vector Regression. This model was created with a radial basis function (RBF) kernel within the Sklearn classification algorithm, performs a grid search with cross-validation (GridSearchCV) to find the best hyperparameters, and subsequently fits the model to the training data with the best hyperparameters.

# Experimental Design

## Results and Discussion

### Unsupervised Learning: DR and clustering

Two dimensionality reduction (PaCMAP and PCA) and two clustering algorithms (HBSCAN and K- Means) were used to determine the best parameter performance. The qualitatively evaluation and comparison of the efficacy of dimensionality reduction and clustering methods are shown in Table 8

Table 8: Qualitative evaluation on DR and clustering.

|  |  |  |  |
| --- | --- | --- | --- |
| Dimensionality Reduction | clustering algorithm | Davies Bouldin score | Silhouette score |
| PaCMAP | **HDBSCAN** | **0.336** | **0.734** |
| PaCMAP | K-Means | 0.605 | 0.627 |
| PCA | HDBSCAN | 2.450 | 0.302 |
| PCA | K-Means | 0.706 | 0.516 |

Overall, PaCMAP and HDBSCAN exhibited superior performance in aggregating points with greater similarities and organizing them into 5 distinct groups; this capability facilitates visualization and clear distinction between the system's different operating conditions and faulty states. This clustered reduced data is visualized in **Error! Reference source not found.**.

|  |  |
| --- | --- |
| A screenshot of a graph  Description automatically generated | |
| (a) | (b) |

Figure 5: DR and clustering. (a) HDBSCAN clustering on PaCMAP reduced data. (b) Timescale PaCMAP reduced data

The DR and clustering analysis presented in Figure 5 reveals insightful and significant patterns regarding the plant’s operations. There are five clusters—each representing a different operating condition of the plant. In addition, we can analyze the time series aspect of each of these operating regimes. In Figure 5 there is a side-by-side comparison of the dimensionally reduced data. In the subplot on the left, the data is colored by the HDBSCAN clustering labels. This represents the same image as before. However, the subplot on the right shows the data colored by the timescale. In this case, it ranges from the length of the datafile or 0 to 1950. With this comparison, the time in which the clusters occur becomes apparent. We can see that the process began with Cluster 1. This would represent one of the normal operating regimes. From here, we can see that the plant moves to Cluster 3, a similar normal operating condition, followed by Clusters 2 and 4. The latter two clusters represent faults introduced in the data. As we can see the plant moves away from Cluster 3 as the faults are introduced. Ultimately, the plant returned to Cluster 3 afterward. Finally, we can see another disturbance introduced with the final shift in the plant represented by Cluster 0.

Since TEP data is simulated benchmark data, we know how the data was produced. In this case, the main disturbance introduced into the virtual plant was the purge rate or XMEAS(10). This fault is responsible for the plant moving away from the normal operating condition represented by Cluster 3. Given that XMEAS(10) is identified as the faulty feature, we will employ supervised learning techniques to develop predictive models, or “soft sensors”, enabling precise and careful analysis for fault detection.

### Supervised Learning: Artificial Neural Network (ANN)

The best model with minimum MSE was chosen to predict the purge rate, as shown in Figure 6.

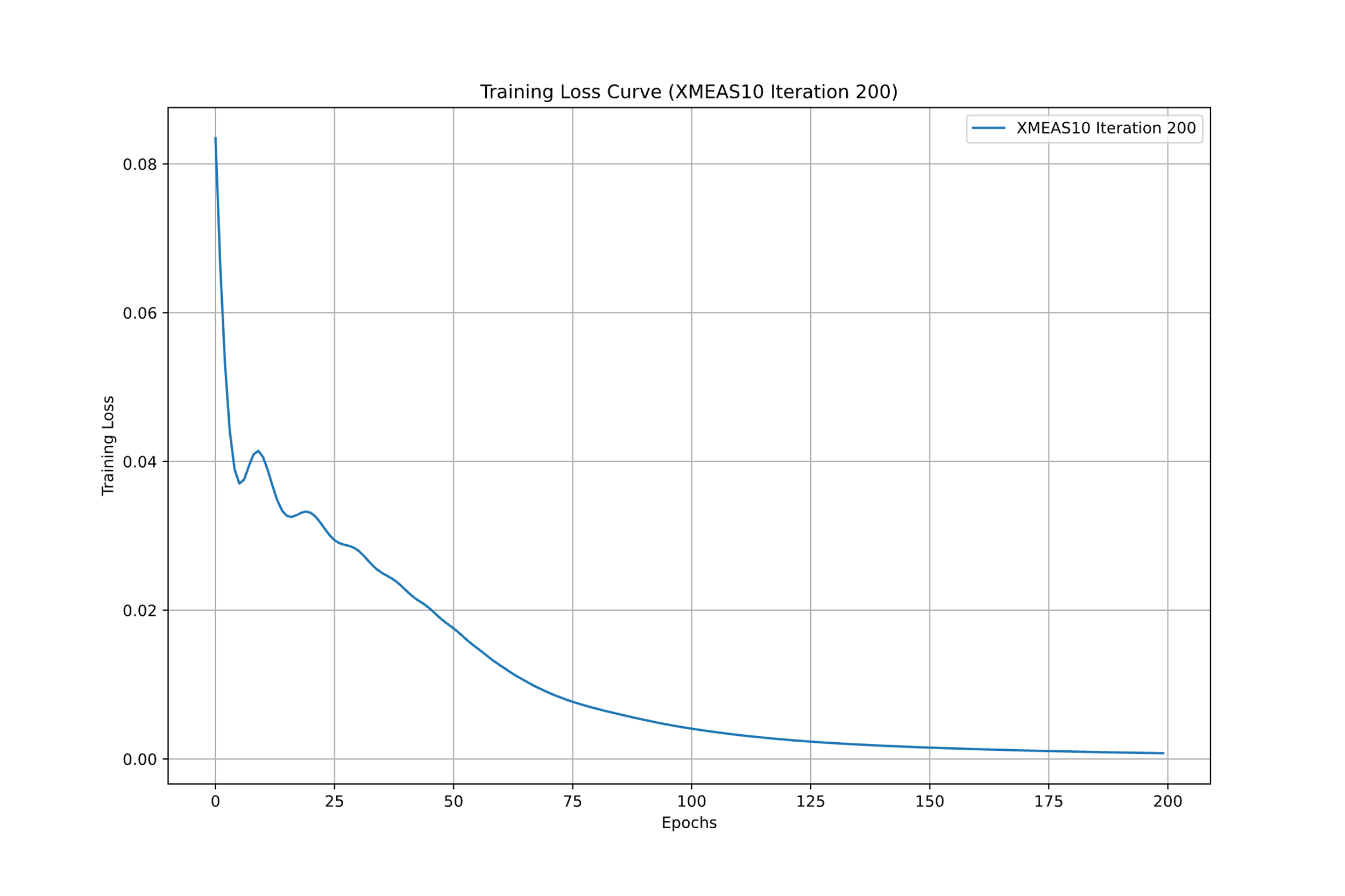


Figure 6: Loss function from 200 epochs using Adam optimizer to minimize the mean square error.

Figure 7shows the model evaluation and performance by plotting a skater plot between the real value of the XMEAS10 and the predicted one for the training and testing data. The Root Mean Squared Error on Test Data and Training Data were 0. 018101, and 0. 018405 respectively which ensures that the model is not overfitting. Also, the R-squared was 0.98. Additionally, Figure 8 shows the relative ranked relative importance weight for the predicted variables.

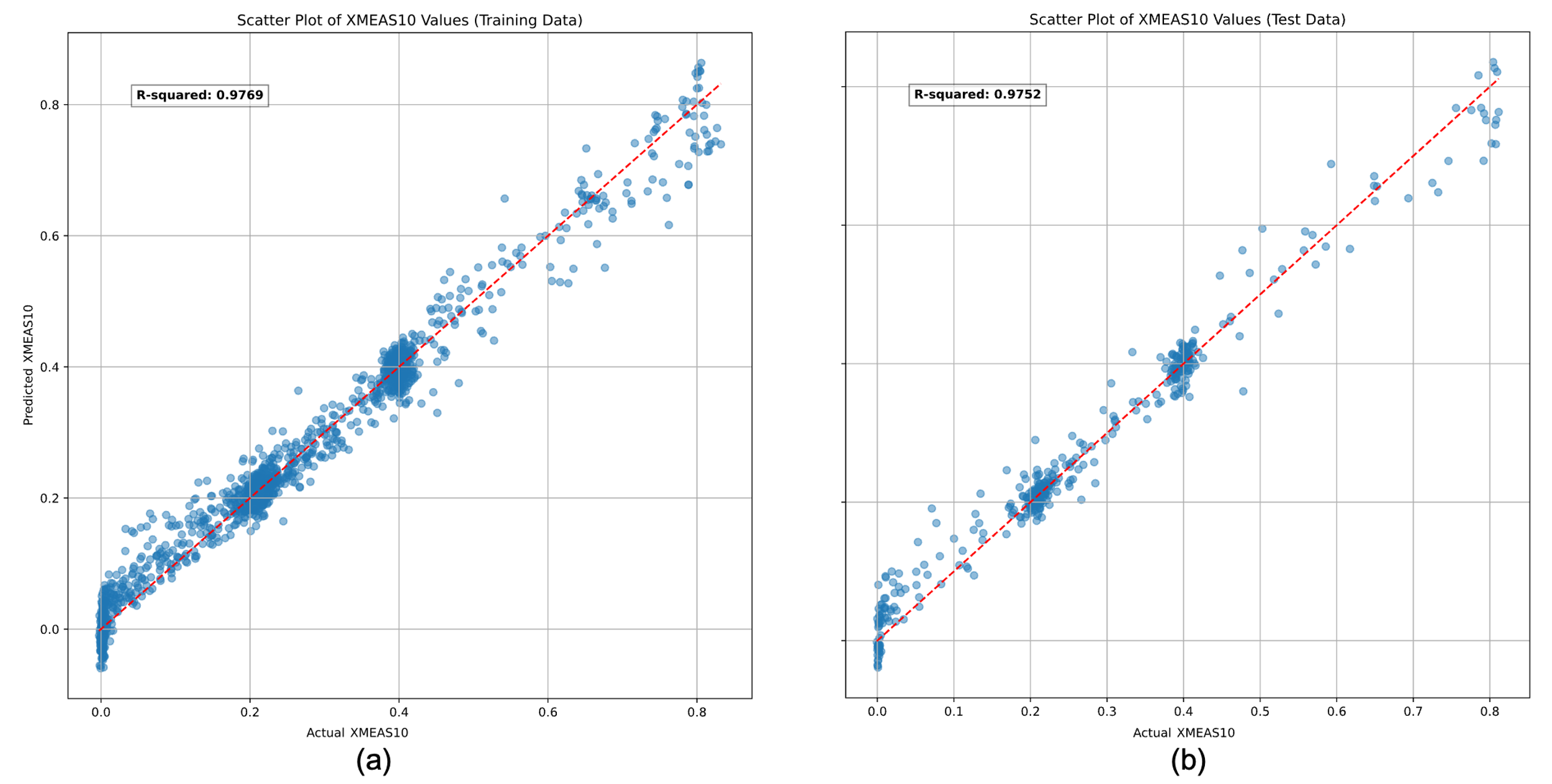


Figure 7: Comparison of observed and predicted XMEAS10 (purge rate) values by training and testing datasets with a coefficient of determination (R2) for the best model (a) for the training data and (b) for the testing data.

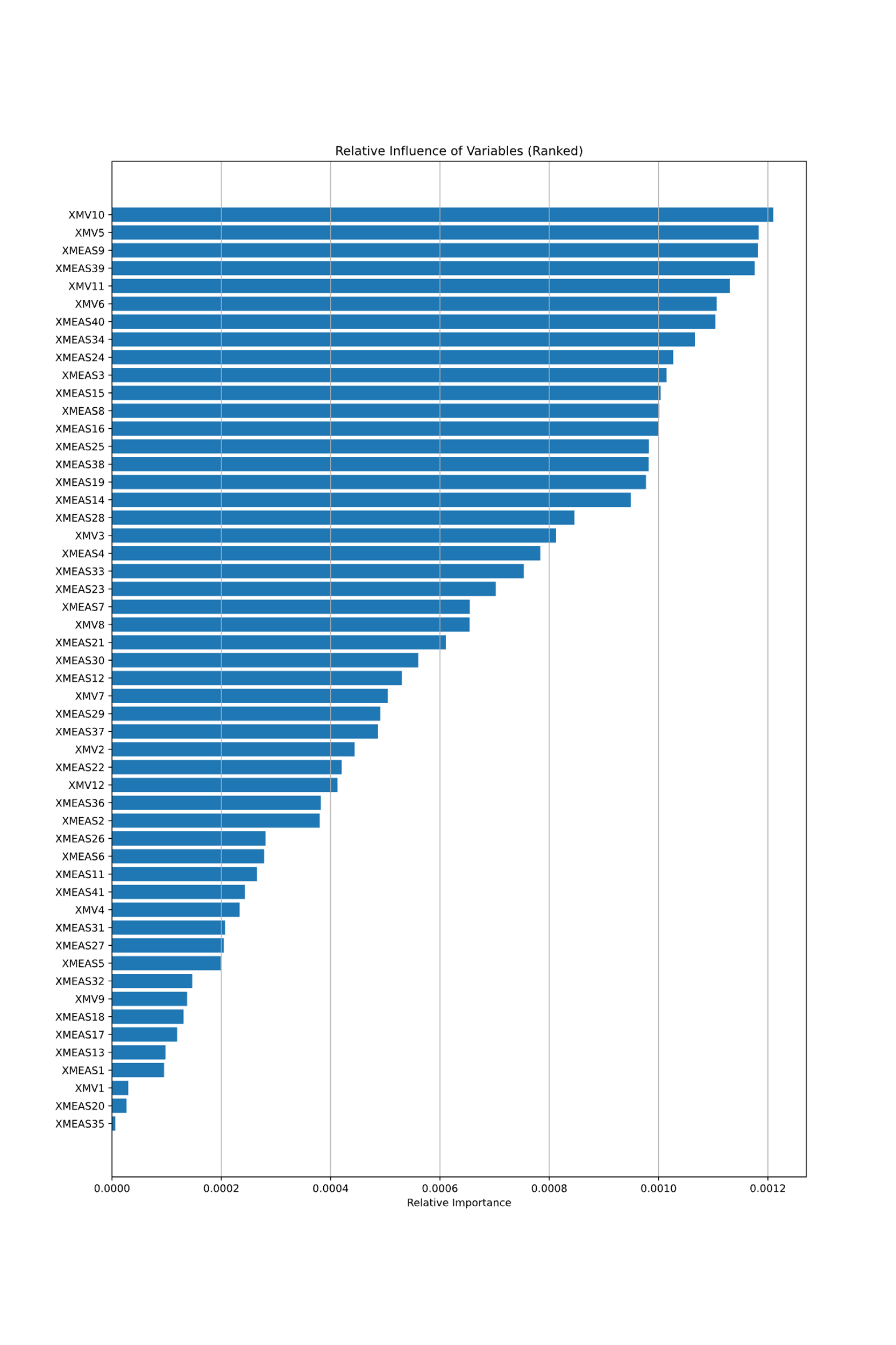


Figure 8: Relative influence of the variables including all the variables that have been used to train the model ML model.

### Supervised Learning: Deep Neural Network (DNN)

During training, the algorithm generates a loss curve for each epoch. The behavior of the optimizer is shown in Figure 9.

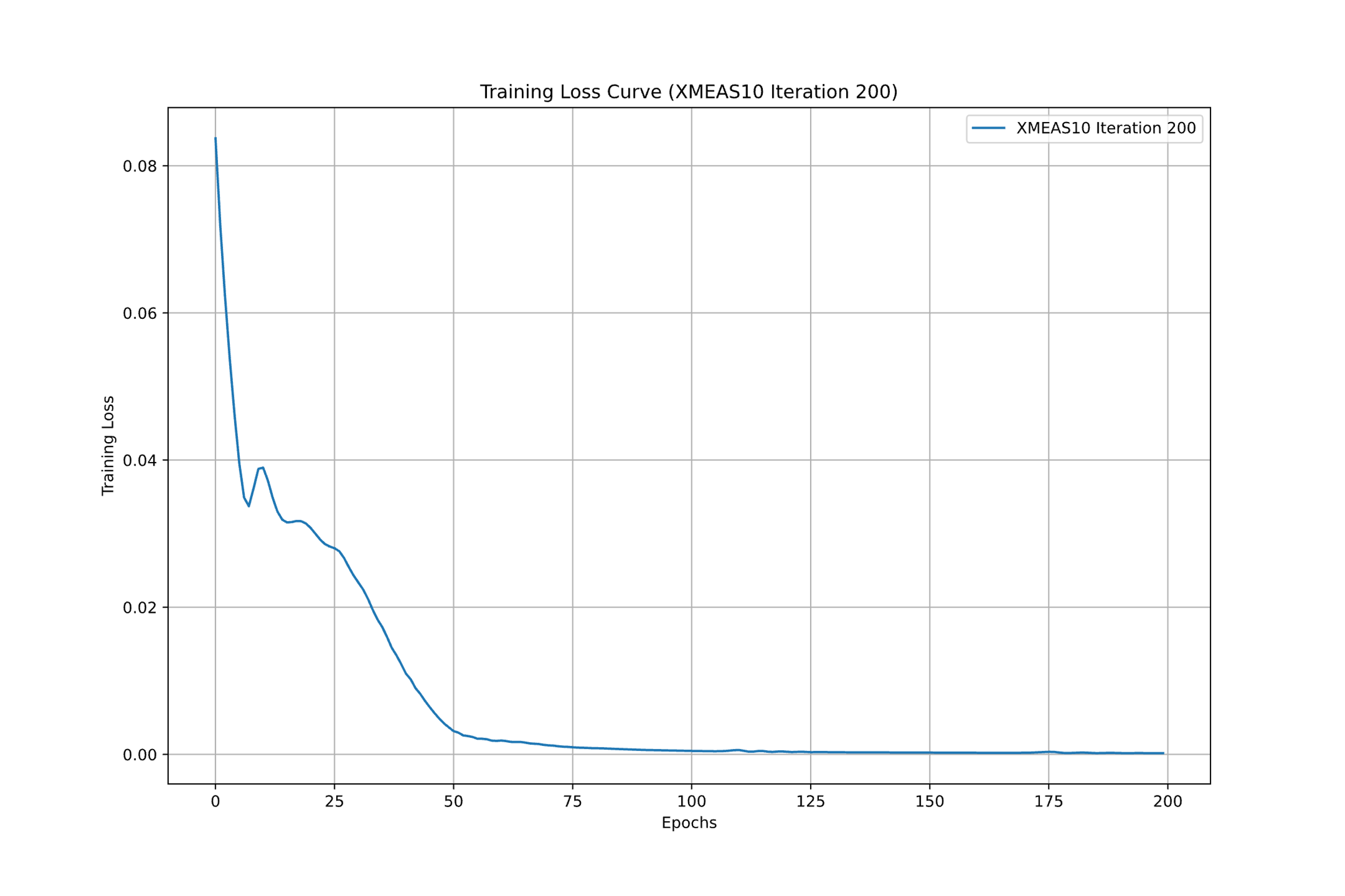


Figure 9: Loss function from 200 epochs using Adam optimizer to minimize the mean square error.

The trained model is evaluated on the test set, and performance metrics such as Root Mean Squared Error (RMSE) and R-squared (R2) are calculated. Additionally, scatter plots are generated to compare actual vs. predicted values for both training and test datasets as shown in Figure 10. The Root Mean Squared Error on Test Data and Training Data was 0.0125, and 0.0130 respectively which indicates that the model is not overfitting. Also, R-squared was very close to the one for both training and testing data sets.

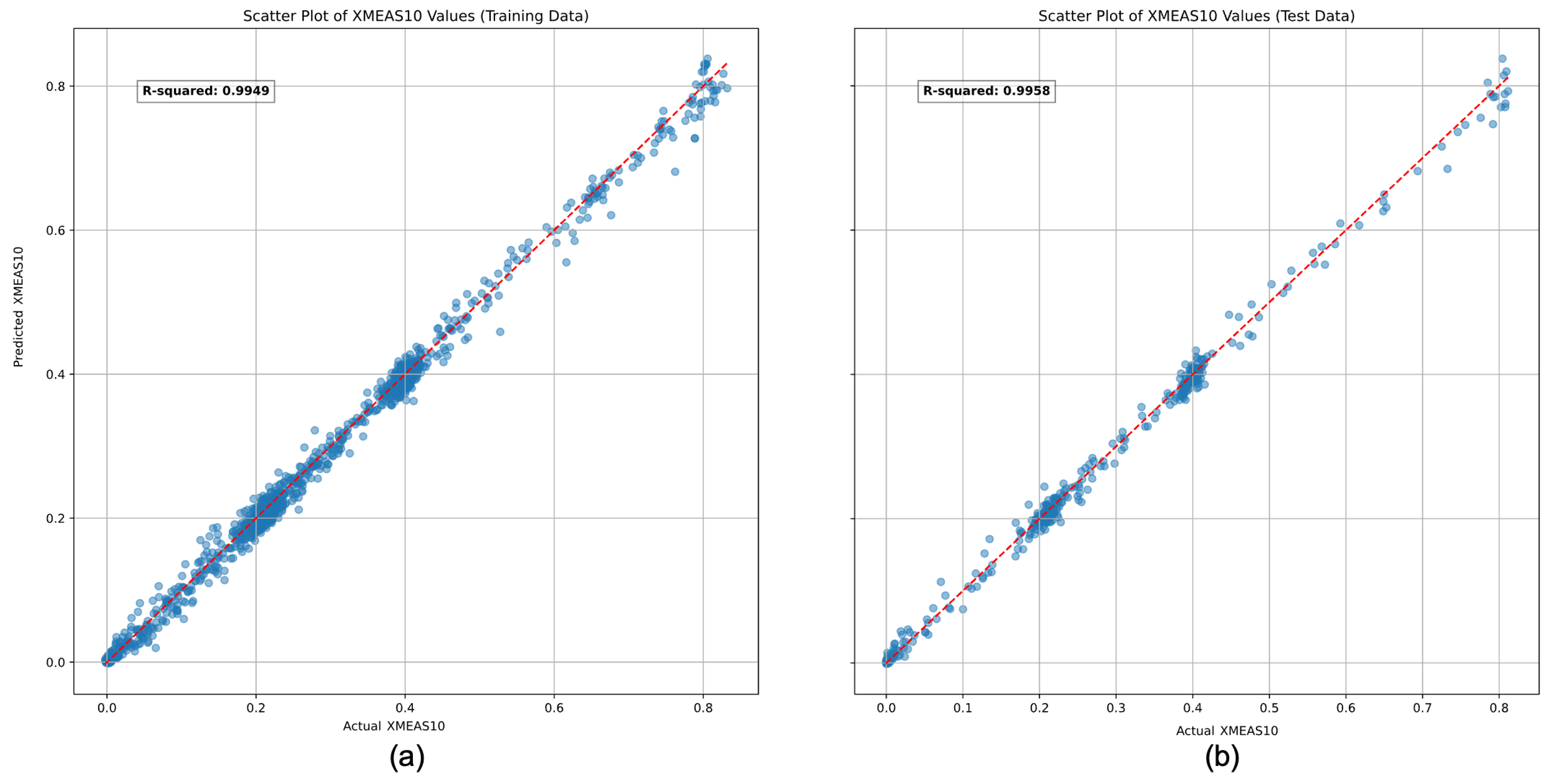


Figure 10: Comparison of observed and predicted XMEAS10 (purge rate) values by training and testing datasets with a coefficient of determination (R2) for the best model (a) for the training data and (b) for the testing data.

Furthermore, the algorithm explores the relative importance of input features by analyzing the weights of the first layer of the deep learning neural network. The absolute values of the weights are used to calculate the relative importance, and a bar plot is created to visualize the ranked influence of variables as shown in Figure 11.

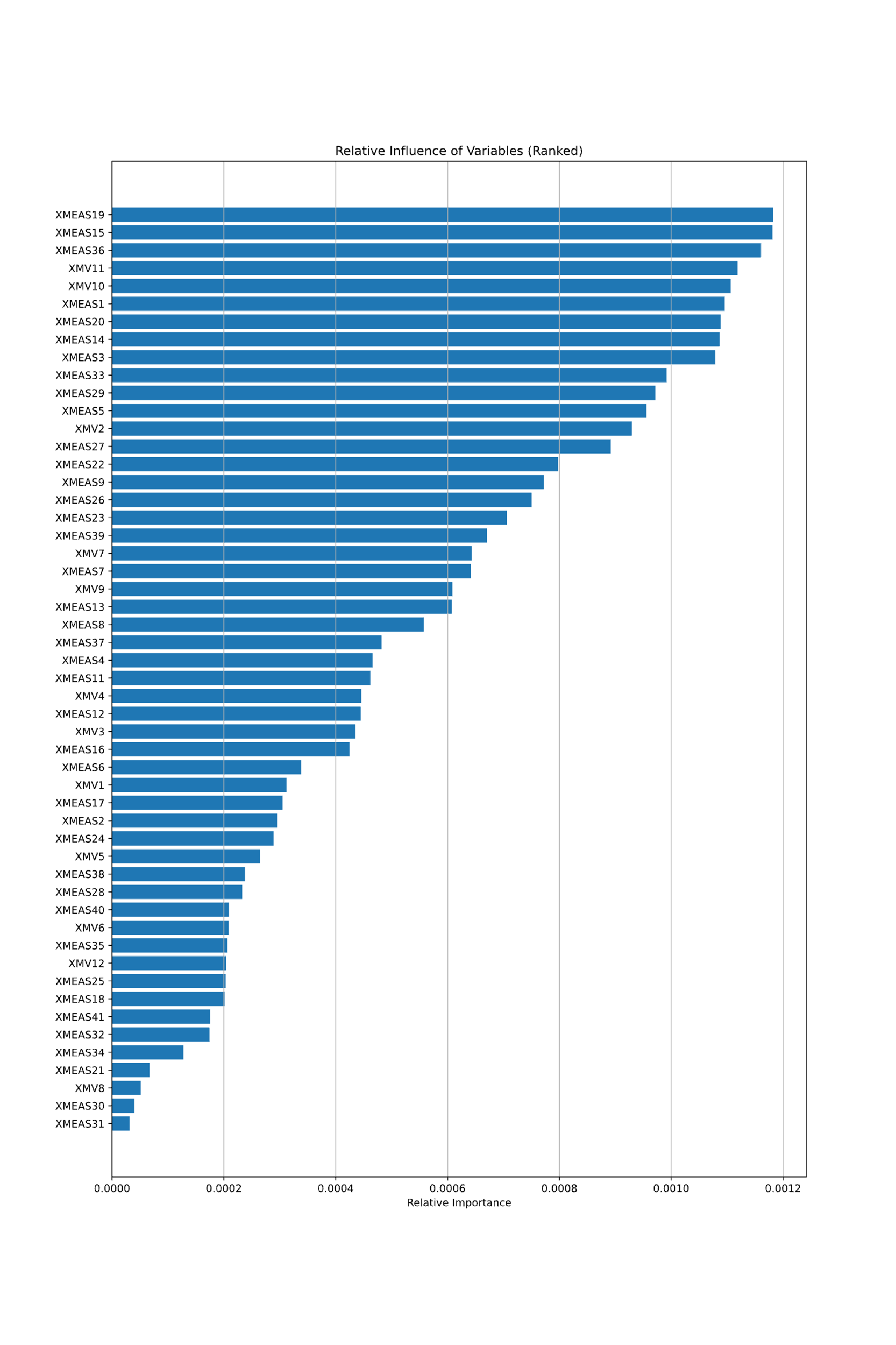


Figure 11: Relative influence of the variables including all the variables that have been used to train the model ML model.

### Supervised Learning: Random Forest (RF)

In our analysis, we utilized a Random Forest regression model to predict the target variable (XMEAS10) using a set of input features. Hyperparameter optimization was conducted through a grid search, exploring parameters such as the number of estimators, maximum tree depth, and minimum samples required for node splitting. Performance evaluation encompassed both training and testing datasets, employing metrics like Root Mean Squared Error (RMSE) and R-squared (R2). Visualizing results involved scatter plots comparing predicted and actual values for training and testing is shown in Figure below. The red dashed line in these plots signifies a perfect prediction. Additionally, R-squared values quantified the model's ability to capture underlying patterns.

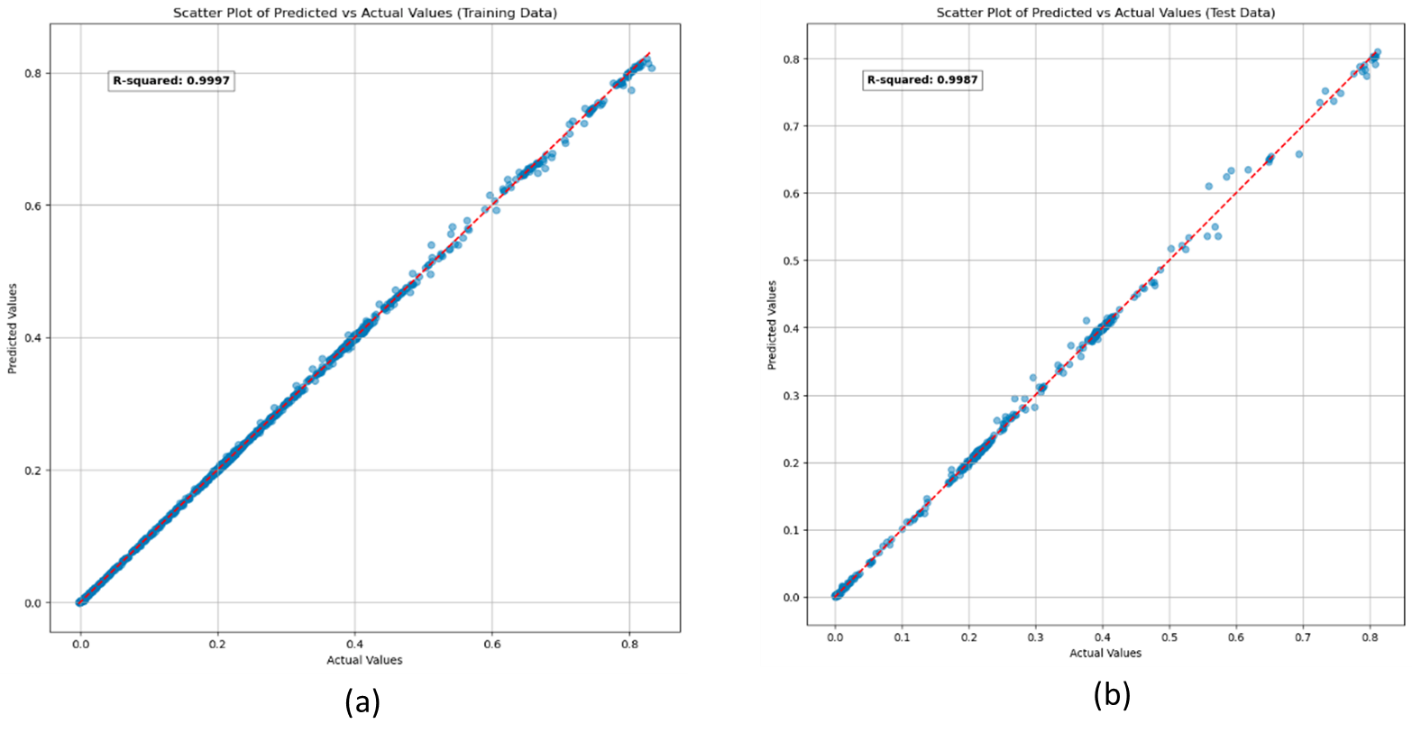


Figure 12: RF comparison of observed and predicted XMEAS10 (purge rate) values by training and testing datasets with a coefficient of determination (R2) for the best model (a) for the training data and (b) for the testing data.

Reported results demonstrate that the random forest model has been trained effectively, achieving high accuracy on both the training and testing datasets (Figure 12). In this case, training RMSE is 0.00288. A testing RMSE of 0.00685 indicates that the model performs well on unseen data, suggesting its ability to generalize to new observations. The training R2 score is exceptionally high at 0.99975. The testing R2 score of 0.9987 is also impressively high. This indicates that the model's performance extends beyond the training set and maintains a high level of explanatory power on new, unseen data. A high testing R2 and a low testing RMSE suggest that the model is robust and not overfitting the training data.

### Supervised Learning: Support Vector Machine (SVM)

A Support Vector Regression (SVR) model was also implemented. SVR is a type of Support vector machine (SVM) that supports linear and non-linear regression. The goal of SVR is to find a function that approximates the relationship between the input variables and a continuous target variable, while minimizing the prediction error. This model was created with a radial basis function (RBF) kernel within the Sklearn classification algorithm, performs a grid search with cross-validation (GridSearchCV) to find the best hyperparameters, and subsequently fits the model to the training data with the best hyperparameters.

This was followed by the model evaluation to predict the target variable for both the training and testing sets, calculated the Root Mean Squared Error (RMSE), and R-squared for both training and testing predictions. The scatter plots below (Figure 13) depict the training (right) and testing (Left) data, comparing actual vs predicted values.

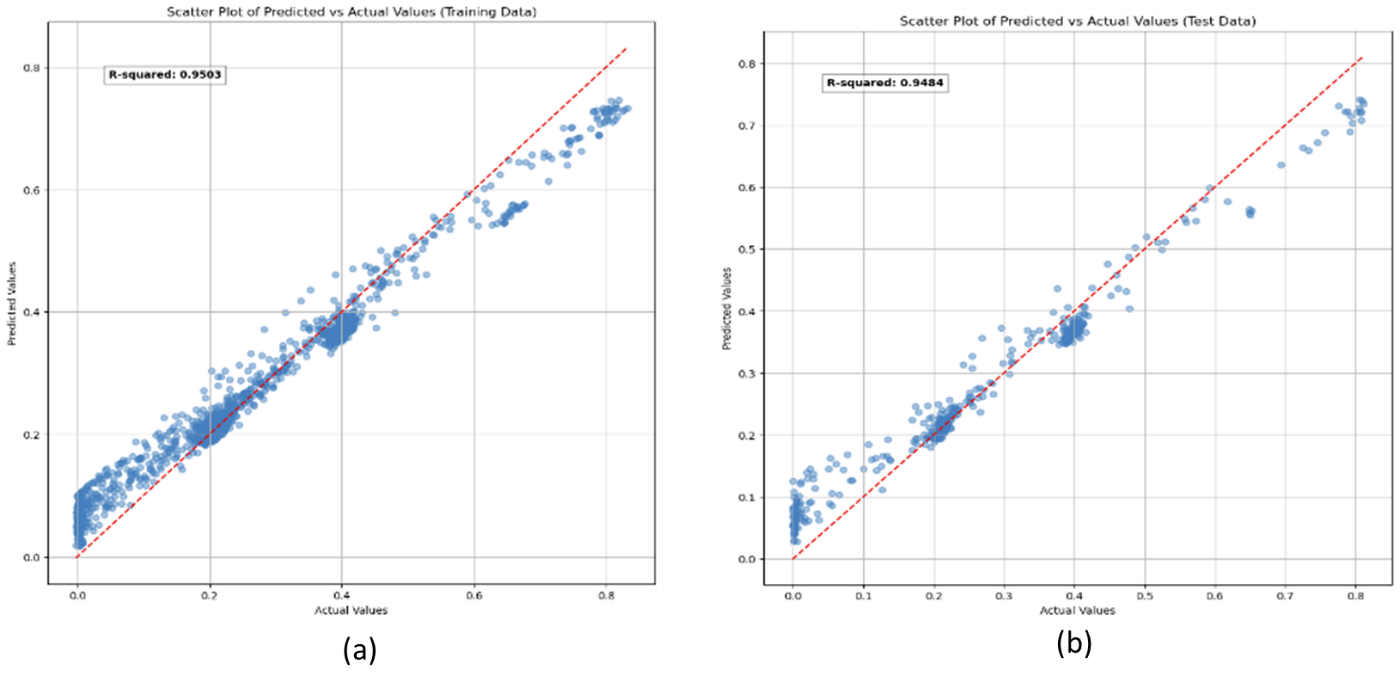


Figure 13: SVM comparison of observed and predicted XMEAS10 (purge rate) values by training and testing datasets with a coefficient of determination (R2) for the best model (a) for the training data and (b) for the testing data

The results shown above indicate that the support vector model has been trained successfully, achieving high accuracy on both the training and testing datasets with a training RMSE of 0.04048 and a testing RMSE of 0.04315. Notably, the training R2 score is high at 0.95025, and the testing R2 score of 0.94843 is also good. A high testing R2 and a low testing RMSE suggest that the model is robust and does not overfit the training data. RMSE value of 0.04315 represents the average magnitude of the residuals (prediction errors) between the actual and predicted values.

### Comparisons between different machine learning methods

The analysis utilized four supervised regression models, namely Supervised Random Forest, Machine Learning Neural Network, Deep Learning Neural Network, and Support Vector Regression (SVR) model, to predict the target variable XMEAS10.

The Supervised Random Forest excelled in handling complex relationships and maintaining robustness. The Machine Learning Neural Network demonstrated strong predictive power, while the Deep Learning Neural Network showed further refinement in capturing underlying patterns. The SVM model employed a Support Vector Regression (SVR) offers a distinctive approach by its ability to approximate relationships between input variables and the target with high flexibility, supporting both linear and non-linear regression.

Each method offers distinct advantages, with the Random Forest, Machine Learning Neural Network, and Deep Learning Neural Network showcasing their strengths in predictive modeling. The SVM model provides an alternative approach, focusing on minimizing prediction error and catering to the specific characteristics of the dataset. The choice among these models would depend on the RMSE as shown in Table 9.

Table 9: Evaluating and comparing the Root Mean Squared Error (RMSE) and R-squared (R2) across the supervised models.

|  |  |  |
| --- | --- | --- |
| **Model** | **RMSE** | **R2** |
| Random Forest | 0.0068 | 0.9987 |
| Machine Learning ANN | 0.018101 | 0.9752 |
| Deep Learning ANN | 0.0125 | 0.9958 |
| SVM | 0.0432 | 0.9484 |

Table 9 presents a comprehensive evaluation and comparison of the Root Mean Squared Error (RMSE) and R-squared (R2) metrics across various supervised models utilized in this study. These metrics serve as vital indicators of predictive accuracy and model performance. The Random Forest model, demonstrating an impressively low RMSE of 0.0068 coupled with a near-perfect R2 of 0.9987. The model's ability to minimize predictive error and explain variance in the dataset highlights its robustness and efficacy in handling complex relationships.

In contrast, The Machine Learning Artificial Neural Network (ANN) demonstrates a solid performance with an RMSE of 0.018101 and an R2 value of 0.9752. This signifies a good balance between predictive error and the model's ability to explain the variance in the data.

One of the standout performers in this comparison is the Deep Learning ANN model showcases even higher accuracy, boasting a notably lower RMSE of 0.0125 and an impressive R2 of 0.9958. These metrics denote exceptional predictive capabilities and a remarkable capacity to capture the underlying patterns within the dataset.

Conversely, the SVM model, while providing valuable insights, presents a higher RMSE of 0.0432 and an R2 of 0.9484. This indicates a comparatively higher predictive error and a relatively lower capacity to explain the variance in the data among the models evaluated in this study.

# Summary

In conclusion, the DR and clustering analysis reveals significant operational patterns in the plant, with 5 cluesters representing distinct operating conditions. The time series analysis allows us to identify the temporal evolution of these clusters highlighting the progression from normal operating states to introduced faults, particularly associated with the purge rate (XMEAS 10). Clustering scores (David-Bouldin Score and Silhouette Score) indicate that PacMAP as dimensionality reduction method and HDBSCAN as clustering are optimal in aggregating data points with greater similarities and organizing them into distinct groups.

The comparison of RMSE and R2 across these supervised models underscores the diverse performances, showcasing strengths and weaknesses in predictive accuracy and explanatory power. The Random Forest model stands out with exceptionally low predictive error and high explanatory capabilities, while the Deep Learning ANN model demonstrates remarkable accuracy in capturing intricate patterns within the dataset.

This insight further supports the efficacy of the analysis and underlines the potential for utilizing unsupervised learning techniques to identify patterns to later use supervise methods to develop “soft sensors” for precise fault detection. Such an approach holds promise in enhancing the plant’s operational analysis faults.

# References

Huang, Haiyang, et al. “Towards a Comprehensive Evaluation of Dimension Reduction Methods for Transcriptomic Data Visualization.” *Nature News*, Nature Publishing Group, 19 July 2022, **Error! Hyperlink reference not valid.**.

Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Lee, Jong-Min, et al. *Fault Detection and Diagnosis Based on Modified Independent Component ...*, aiche.onlinelibrary.wiley.com/doi/10.1002/aic.10978. Accessed 21 Nov. 2023.

MacGregor, J.F, and T. Kourti. “Statistical Process Control of Multivariate Processes.” Redirecting, doi.org/10.1016/0967-0661(95)00014-L. Accessed 21 Nov. 2023.

McInnes, Leland, et al. “HDBSCAN: Hierarchical Density Based Clustering.” *Journal of Open Source Software*, 21 Mar. 2017, joss.theoj.org/papers/10.21105/joss.00205.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., & Chintala, S. (2019). Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32.

“Sklearn.Ensemble.Randomforestclassifier.” *Scikit*, scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html. Accessed 22 Nov. 2023.

Xiaolu Chen, June 9, 2019, "Tennessee Eastman simulation dataset", IEEE Dataport, doi: <https://dx.doi.org/10.21227/4519-z502>.

YingfanWang. “Yingfanwang/PACMAP: PaCMAP: Large-Scale Dimension Reduction Technique Preserving Both Global and Local Structure.” *GitHub*, github.com/YingfanWang/PaCMAP#install-from-pypi-via-pip. Accessed 24 Nov. 2023.

**Repositories:**

PacMAP:

<https://github.com/YingfanWang/PaCMAP#install-from-pypi-via-pip>

Information. HDBSCAN:

<https://scikit-learn.org/stable/modules/generated/sklearn.cluster.HDBSCAN.html>

Project code:

<https://github.com/KyleTerrito/CSC7333.git>

Link to presentation:

[Presentation\_group1.mp4](https://lsumail2-my.sharepoint.com/:v:/r/personal/gtheis2_lsu_edu/Documents/Semestres/2.Fall%202023/CSC%207333%20Machine%20Learning/Group%20project/Written%20Project/Presentation_group1.mp4?csf=1&web=1&e=EQxD1B&nav=eyJyZWZlcnJhbEluZm8iOnsicmVmZXJyYWxBcHAiOiJTdHJlYW1XZWJBcHAiLCJyZWZlcnJhbFZpZXciOiJTaGFyZURpYWxvZy1MaW5rIiwicmVmZXJyYWxBcHBQbGF0Zm9ybSI6IldlYiIsInJlZmVycmFsTW9kZSI6InZpZXcifX0%3D)