**Smart Grid Stability Prediction Using Machine Learning**

**Introduction**

The foundational understanding of the problem and insights into the dataset were derived from an analysis presented in Paulo Breviglieri's notebook. Recognizing the clarity and conciseness of Breviglieri's explanation, minimal alterations were made to the core content, primarily focusing on the methodology employed.

While Paulo Breviglieri utilized a neural network approach to address the problem at hand, our methodology diverged towards a machine learning-centric strategy. This deliberate shift stemmed from a strategic evaluation of the dataset's characteristics, computational efficiency considerations, and the inherent strengths of machine learning algorithms in handling tabular datasets. By opting for a machine learning approach, we aimed to leverage the versatility and interpretability of machine learning models in the context of the smart grid stability problem. This decision allowed for a nuanced exploration of various classification algorithms, feature engineering techniques, and model interpretability tools, contributing to a comprehensive analysis of grid stability factors and predictive accuracy enhancement.

Through this deliberate methodological choice, we sought to build upon Breviglieri's foundational insights while offering a distinct perspective and contribution to the discourse on smart grid stability analysis. Our approach underscores the iterative and adaptive nature of scientific inquiry, where methodological adaptations can lead to enriched understanding and innovative solutions within complex domains such as renewable energy integration and grid management.

1.1. Renewable Energy Sources and Smart Grids

The ascent of renewable energy sources provides the global community with a much demanded alternative to traditional, finite and climate-unfriendly fossil fuels. However, their adoption poses a set of new paradigms, out of which two interrelated aspects deserve particular attention:

* Prior to the rise of renewable energy sources, the traditional operating ecosystem involved few production entities (sources) supplying energy to consumers over unidirectional flows. With the advent of renewable options, end users (households and enterprises) now not only consume energy but have the ability to produce and supply it - hence a new term to designate them, 'prosumers'. As a result, energy flow within distribution grids - 'smart grids' - has become bidirectional;
* Despite the increased flexibility brought in by the introduction of renewable sources and the aforementioned emergence of 'prosumers', the management of supply and demand in a more complex generation / distribution / consumption environment and the related economic implications (particularly the decision to buy energy at a given price or not) have become even more challenging.

Relevant contributions on how to tackle the requirements of such new scenario have been offered by academy and industry over the past years. Special attention has been devoted to the study of smart grid stability.

1.2. Modeling grid stability

In a smart grid, consumer demand information is collected, centrally evaluated against current supply conditions and the resulting proposed price information is sent back to customers for them to decide about usage. As the whole process is time-dependent, dynamically estimating grid stability becomes not only a concern but a major requirement.

Put simply, the objective is to understand and plan for both energy production and/or consumption disturbances and fluctuations introduced by system participants in a dynamic way, taking into consideration not only technical aspects but also how participants respond to changes in the associated economic aspects (energy price).

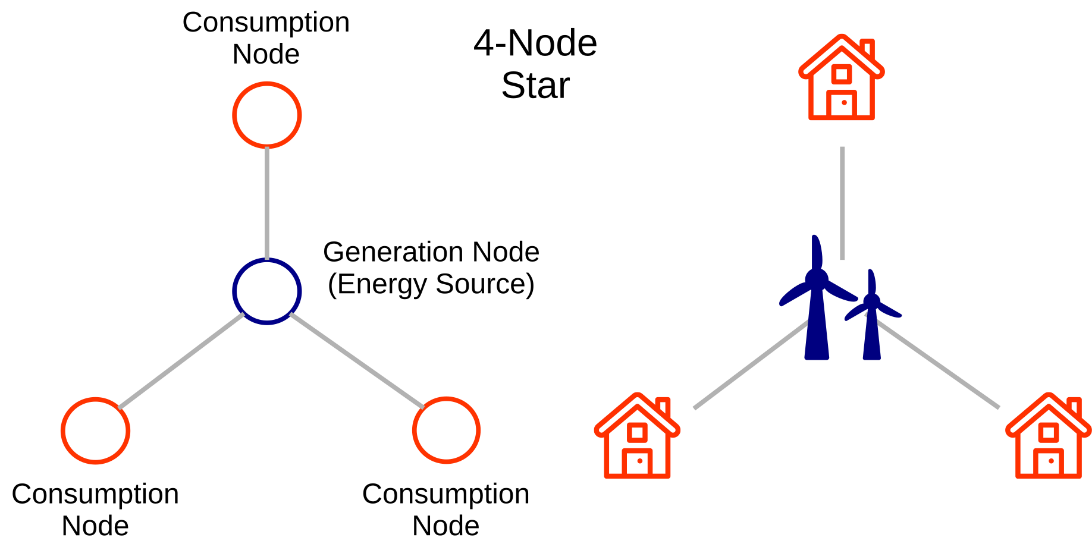
The work of researchers cited in foreword focuses on Decentral Smart Grid Control (DSGC) systems, a methodology strictly tied to monitoring one particular property of the grid - its frequency.

The term 'frequency' refers to the alternate current (AC) frequency, measured in cycles per second or its equivalent unit, Hertz (Hz). Around the world AC frequencies of either 50 or 60 Hz are utilized in electric power generation-distribution systems.

It is known that the electrical signal frequency "increases in times of excess generation, while it decreases in times of underproduction" [4]. Therefore, measuring the grid frequency at the premise of each customer would suffice to provide the network administrator with all required information about the current network power balance, so that it can price its energy offering - and inform consumers - accordingly.

The DSGC differential equation-based mathematical model described in [4] and assessed in [3] aims at identifying grid instability for a reference 4-node star architecture, comprising one power source (a centralized generation node) supplying energy to three consumption nodes. The model takes into consideration inputs (features) related to:

* the total power balance (nominal power produced or consumed at each grid node);
* the response time of participants to adjust consumption and/or production in response to price changes (referred to as "reaction time);
* energy price elasticity.



1.3. Addressing simplifications in the model

So we have a mathematical model with which grid instability can be predicted! The need of a tool to predict grid instability would have been met, and the binary classification ("stable" versus "unstable") problem would be solved! However, the execution of this model relies on significant simplifications.

A differential equation-based model can be manipulated in several ways. One traditional approach consists in running simulations with a combination of fixed values for one subset of variables and fixed value distributions for the remaining subset. As elegantly depicted in [3], this strategy leads to two primary issues, referred to as the "fixed inputs issue" and the "equality issue". Please refer to [3] for a comprehensive assessment of both issues.

Alternative approaches have been proposed to overcome the inherent DSGC model simplifications. In particular, Dr. Arzamasov's team at the KIT suggest the use of machine learning - decision trees (CART) - and space-filling designs to process results from simulations with different DSGC parameter configurations.

In other words, machine learning is used in [3] in the following way:

1. A given set of input parameters (call it a 'vector') is fed into the original DSGC model;
2. The DSGC model process this vector and returns a binary output - the grid stability for that particular set of inputs ('stable' or 'unstable' - a binary classification!);
3. Steps 1 and 2 are executed 'n' times;
4. A large set of vectors and the respective outputs (stability or instability) is created.

In summary, the original DSGC model was run to generate a set of inputs and outputs that a 'learning machine' can process and make predictions from!

Per [3], accuracies of "around 80%" have been achieved with the CART-based learning machine. [1]

**Objectives of this project**

1. Enhance predictive accuracy to achieve a higher degree of reliability in assessing grid stability, contributing to improved overall grid performance and energy supply management.
2. Utilize machine learning models due to their proven effectiveness in handling tabular datasets, optimizing computational efficiency and ensuring robust predictive capabilities for smart grid stability analysis. [5]

**Dataset**

The dataset chosen for this machine learning exercise has a synthetic nature and contains results from simulations of grid stability for a reference 4-node star network, as described in 1.2.

The original dataset contains 10,000 observations. As the reference grid is symmetric, the dataset can be augmented in 3! (3 factorial) times, or 6 times, representing a permutation of the three consumers occupying three consumer nodes. The augmented version has then 60,000 observations. It also contains 12 primary predictive features and two dependent variables. [2]

**Predictive features**

1. 'tau1' to 'tau4': the reaction time of each network participant, a real value within the range 0.5 to 10 ('tau1' corresponds to the supplier node, 'tau2' to 'tau4' to the consumer nodes);
2. 'p1' to 'p4': nominal power produced (positive) or consumed (negative) by each network participant, a real value within the range -2.0 to -0.5 for consumers ('p2' to 'p4'). As the total power consumed equals the total power generated, p1 (supplier node) = - (p2 + p3 + p4);
3. 'g1' to 'g4': price elasticity coefficient for each network participant, a real value within the range 0.05 to 1.00 ('g1' corresponds to the supplier node, 'g2' to 'g4' to the consumer nodes; 'g' stands for 'gamma');

**Dependent variables**

1. 'stab': the maximum real part of the characteristic differentia equation root (if positive, the system is linearly unstable; if negative, linearly stable);
2. 'stabf': a categorical (binary) label ('stable' or 'unstable').

As there is a direct relationship between 'stab' and 'stabf' ('stabf' = 'stable' if 'stab' <= 0, 'unstable' otherwise), 'stab' will be dropped and 'stabf' will remain as the sole dependent variable.

As the dataset content comes from simulation exercises, there are no missing values. Also, all features are originally numerical, no feature coding is required. Such dataset properties allow for a direct jump to machine modeling without the need of data preprocessing or feature engineering. [1]

**Methodology**

The methodology employed for this project involved a systematic approach to model training and evaluation aimed at optimizing predictive accuracy for smart grid stability analysis. The key steps undertaken are outlined below:

Data Splitting and Preprocessing: The dataset was divided into training and testing sets, with 90% allocated to training. This step ensured an adequate volume of data for model training while retaining a separate set for unbiased evaluation. Additionally, standard scaling was applied to the input data to normalize features and facilitate model convergence.

Model Training and Evaluation: Multiple classification models were trained using various methodologies to identify the most effective approach for predicting grid stability. The following methods were employed and compared:

1. Standard Training: Models were trained without additional preprocessing or feature engineering, except for standard scaling as mentioned earlier.
2. Correlation Matrix-Based Training: Models were trained based on the correlation matrix of the data, leveraging inter-feature relationships to optimize predictive performance.
3. K-Means Clustering-Based Training: Models were trained using K-means clustering to identify clusters within the data and enhance predictive accuracy.
4. Mutual Information Regression: Models were trained using mutual information regression between input and output data to capture relevant information and improve prediction quality.
5. Principal Component Analysis (PCA): Models were trained using PCA, a dimensionality reduction technique, to extract important features and reduce computational complexity.

Model Selection: A diverse range of classification models were utilized for training and evaluation, including:

1. Logistic Regression
2. Ridge Classifier
3. SGD Classifier
4. Passive Aggressive Classifier
5. Linear Discriminant Analysis
6. Quadratic Discriminant Analysis
7. Support Vector Classifier
8. Linear Support Vector Classifier
9. KNeighbors Classifier
10. Gaussian Naive Bayes
11. Decision Tree Classifier
12. Hist Gradient Boosting Classifier
13. Gradient Boosting Classifier
14. Random Forest Classifier
15. Bagging Classifier
16. Voting Classifier
17. AdaBoost Classifier
18. XGBoost Classifier
19. LightGBM Classifier

These models were selected based on their suitability for classification tasks and their potential to capture complex relationships within the data.

By employing a comprehensive methodology encompassing data splitting, preprocessing, diverse model training approaches, and model selection, this study aimed to identify the most effective strategy for predicting smart grid stability with high accuracy and reliability.

1. **Standard Training**

The standard training methodology involved a systematic approach to model training and evaluation. Initially, the dataset was prepared by splitting it into training and testing subsets, with 90% of the data allocated for training purposes. Prior to training, standard scaling was applied to normalize the input data, ensuring consistency and facilitating model convergence.

The training process was executed using a custom training function designed specifically for this project. Within this function, a structured loop was implemented to iteratively train each classification model included in the study. The loop initiated the training phase for a model, during which the model learned from the training data to establish patterns and relationships between input features and target variables.

Following the training phase, the trained model was then evaluated using the reserved test data. This evaluation step was crucial in assessing the model's performance on unseen data, thereby providing a robust measure of its predictive capabilities. Performance metrics such as accuracy, precision, recall, and F1 score were calculated to gauge the model's effectiveness in predicting grid stability.

Figure 1.1 illustrates the sequential pipeline of the standard training methodology, highlighting the data flow from preprocessing to model training and evaluation. This method served as the baseline for comparison against other training methodologies, aiming to establish a benchmark for predictive accuracy and model performance.

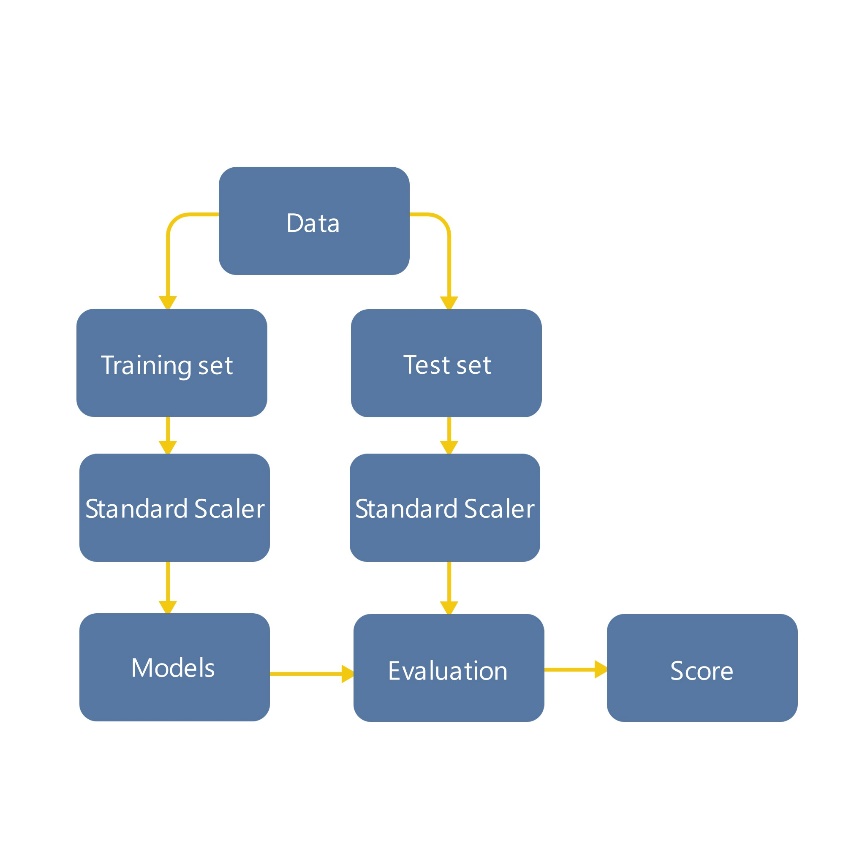


Figure 1.1 Pipeline of the inference without Feature engineering.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Accuracy | MAE | Precision | Recall | F1 |
| Logistic Regression | 0.8218 | 0.1781 | 0.7762 | 0.7089 | 0.7410 |
| Ridge Classifier | 0.82 | 0.18 | 0.7761 | 0.7020 | 0.7372 |
| SGD classifier | 0.8191 | 0.1808 | 0.7593 | 0.7279 | 0.7433 |
| Passive Aggressive Classifier | 0.7943 | 0.2056 | 0.8412 | 0.5278 | 0.6486 |
| Linear Discriminant Analysis | 0.8215 | 0.1785 | 0.7749 | 0.7099 | 0.7409 |
| Quadratic Discriminant Analysis | 0.8761 | 0.1238 | 0.8436 | 0.8049 | 0.8238 |
| Suport Vector Classifier | 0.9813 | 0.0186 | 0.9789 | 0.9689 | 0.9739 |
| Linear Support Vector Classifier | 0.8215 | 0.1785 | 0.7757 | 0.7085 | 0.7406 |
| KNeighbors Classifier | 0.902 | 0.098 | 0.9153 | 0.8016 | 0.8547 |
| Gaussian Naive Bayes | 0.836 | 0.164 | 0.8393 | 0.6728 | 0.7469 |
| Decision Tree Classifier | 0.8935 | 0.1065 | 0.8534 | 0.8498 | 0.8516 |
| Hist Gradient Boosting Classifier | 0.963 | 0.037 | 0.9574 | 0.9388 | 0.9480 |
| Gradient Boosting Classifier | 0.931 | 0.069 | 0.9390 | 0.8642 | 0.9000 |
| Random Forest Classifier | 0.9498 | 0.0501 | 0.9487 | 0.9096 | 0.9287 |
| Bagging Classifier | 0.8253 | 0.1746 | 0.9004 | 0.5783 | 0.7042 |
| Voting Classifier | 0.8501 | 0.1498 | 0.8460 | 0.7131 | 0.7739 |
| AdaBoost Classifier | 0.8638 | 0.1361 | 0.8320 | 0.7784 | 0.8044 |
| XGBoost Classifier | 0.9845 | 0.0155 | 0.9836 | 0.9731 | 0.9783 |
| LightGBM Classifier | 0.997 | 0.003 | 0.9972 | 0.9944 | 0.9958 |

Table 1.1 Showing the results of implementing Standard Training.

1. **Correlation Matrix-Based Training**

The correlation matrix-based training methodology involved a thorough analysis of feature correlations to optimize model performance. Initially, a correlation matrix was generated to quantify the relationships between input features (p1 to p4) and the target variable “stabf” representing grid stability.

Upon examination of the correlation matrix, it was observed that features p1 to p4 exhibited weak correlations with the target variable “stabf”. To enhance the model's predictive accuracy and streamline computational efficiency, a decision was made to exclude these less relevant features from the training process.

Subsequently, the models were trained using the modified dataset where the aforementioned columns (p1 to p4) were dropped. This streamlined dataset aimed to focus the model's attention on more influential features, thereby potentially improving its ability to discern patterns related to grid stability.

The training process involved feeding the revised dataset into the custom training function, as described earlier. Each classification model included in the study underwent training using the modified dataset, followed by evaluation on the test data to assess any improvements or changes in predictive performance.

The iterative nature of this methodology, guided by insights from the correlation matrix, allowed for a targeted approach to feature selection and model training. Figure 2.1 illustrates the workflow of the correlation matrix-based training method, showcasing the strategic decision-making process to optimize model inputs and enhance predictive accuracy, and Figure 2.2 shows the correlation matrix which the training was based on.

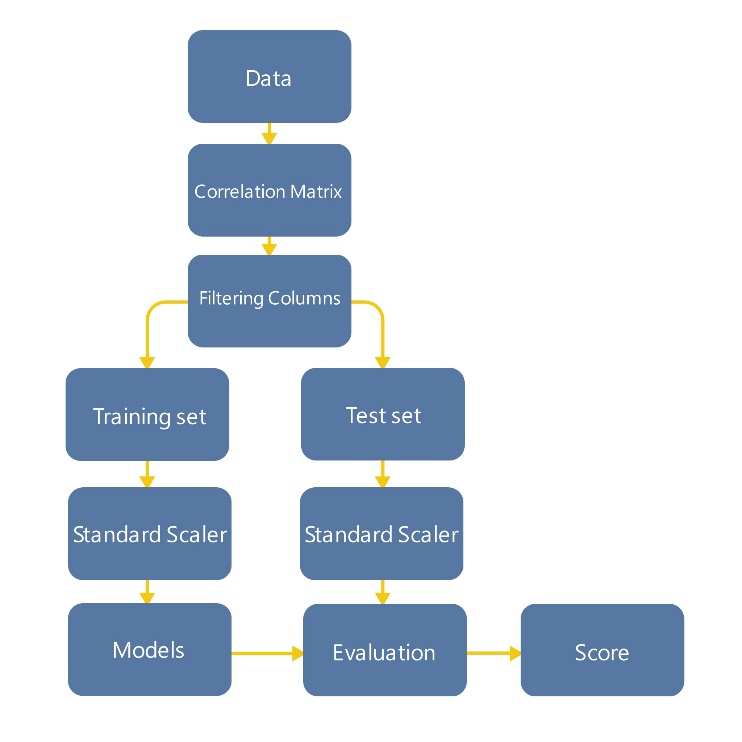


Figure 2.1 Pipeline of the inference with Correlation matrix.

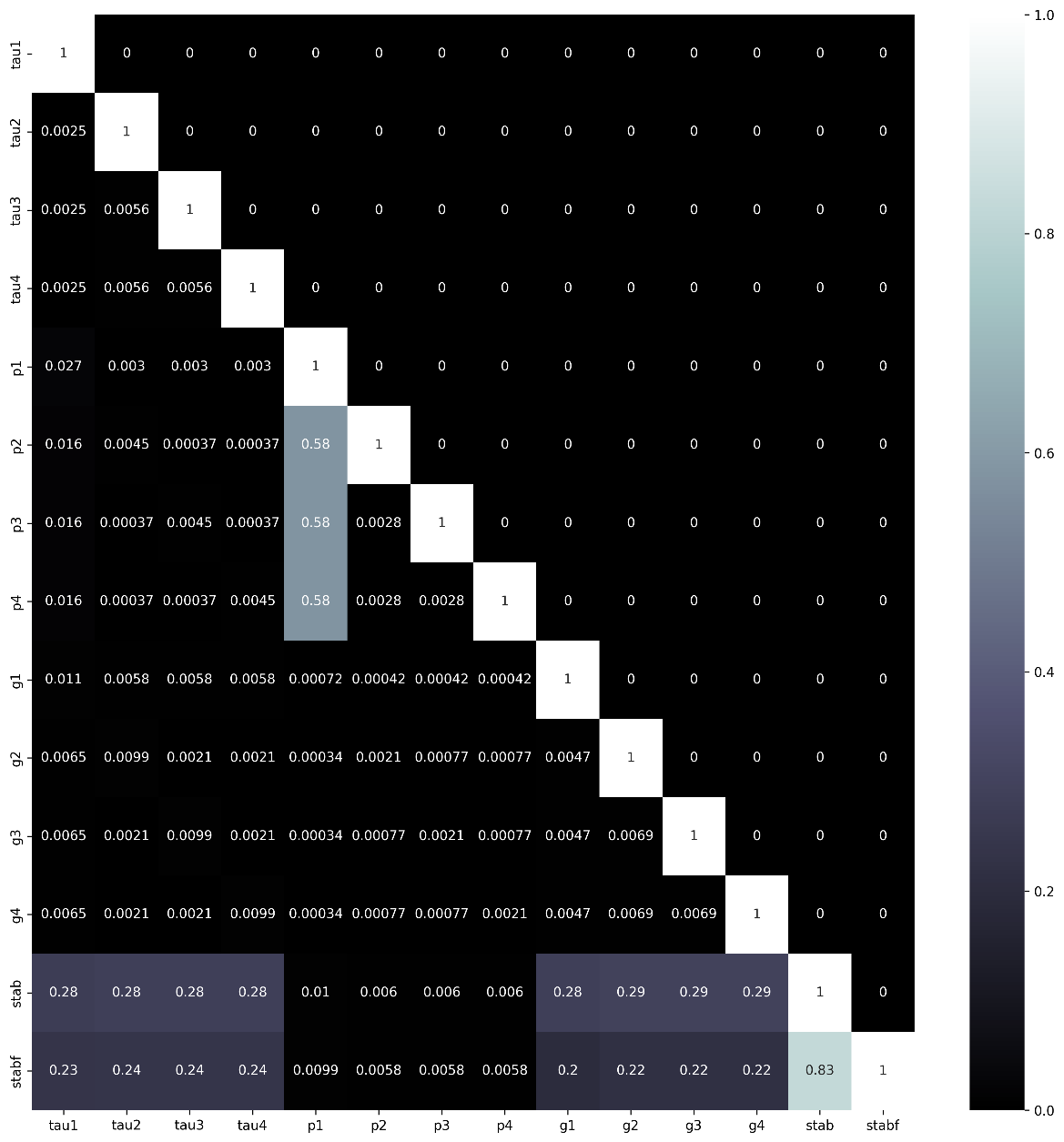


Figure 2.2 correlation matrix of the entire dataset.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Accuracy | MAE | Precision | Recall | F1 |
| Logistic Regression | 0.819 | 0.181 | 0.7686 | 0.7163 | 0.7416 |
| Ridge Classifier | 0.8192 | 0.1808 | 0.7701 | 0.7145 | 0.7412 |
| SGD classifier | 0.8185 | 0.1815 | 0.7683 | 0.7149 | 0.7407 |
| Passive Aggressive Classifier | 0.7348 | 0.2652 | 0.6188 | 0.6993 | 0.6566 |
| Linear Discriminant Analysis | 0.819 | 0.181 | 0.767 | 0.7191 | 0.7423 |
| Quadratic Discriminant Analysis | 0.8875 | 0.1125 | 0.8669 | 0.8147 | 0.84 |
| Suport Vector Classifier | 0.9902 | 0.0098 | 0.9903 | 0.9825 | 0.9864 |
| Linear Support Vector Classifier | 0.819 | 0.181 | 0.7684 | 0.7168 | 0.7417 |
| KNeighbors Classifier | 0.9383 | 0.0617 | 0.9435 | 0.8828 | 0.9121 |
| Gaussian Naive Bayes | 0.8387 | 0.1613 | 0.8381 | 0.6878 | 0.7556 |
| Decision Tree Classifier | 0.9063 | 0.0937 | 0.8615 | 0.8837 | 0.8724 |
| Hist Gradient Boosting Classifier | 0.9613 | 0.0387 | 0.9572 | 0.9352 | 0.946 |
| Gradient Boosting Classifier | 0.9277 | 0.0723 | 0.9381 | 0.857 | 0.8957 |
| Random Forest Classifier | 0.9542 | 0.0458 | 0.9477 | 0.9246 | 0.936 |
| Bagging Classifier | 0.8418 | 0.1582 | 0.9038 | 0.6308 | 0.743 |
| Voting Classifier | 0.8505 | 0.1495 | 0.8417 | 0.7237 | 0.7782 |
| AdaBoost Classifier | 0.8515 | 0.1485 | 0.8181 | 0.7591 | 0.7875 |
| XGBoost Classifier | 0.9872 | 0.0128 | 0.9852 | 0.9793 | 0.9822 |
| LightGBM Classifier | 0.9952 | 0.0048 | 0.9945 | 0.9922 | 0.9933 |

Table 2.1 Showing the results of implementing Correlation based training.

1. **K-Means Clustering-Based Training**

The K-Means Clustering-Based Training methodology leveraged clustering techniques to enhance model training and predictive accuracy. The process began with the standardization of the training dataset to ensure uniformity and facilitate clustering analysis.

A K-Means clustering model was fitted to the standardized training set with a predefined number of clusters, set to 2 to match the two label types representing stability states: 0 for unstable and 1 for stable. This alignment of clusters with label types aimed to create a meaningful segmentation of the data based on inherent patterns and similarities.

The K-Means clustering model assigned cluster labels to each data point, effectively grouping similar instances together based on feature similarities.

A separate column was then created in the training dataset to accommodate these K-Means cluster labels, providing additional information for model training and analysis.

Figure 3.1 shows the pipeline of this process, showcasing the steps involved from data standardization to K-Means clustering and dataset augmentation visualized as a circle with the plus sign.

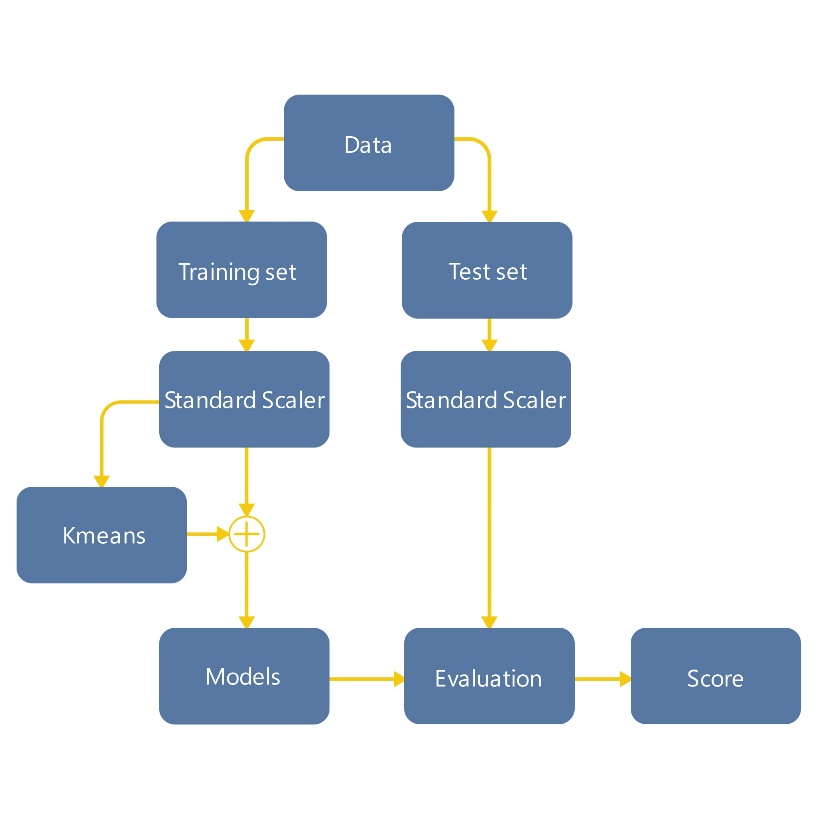


Figure 3.1 Pipeline of the inference with K means clustering.

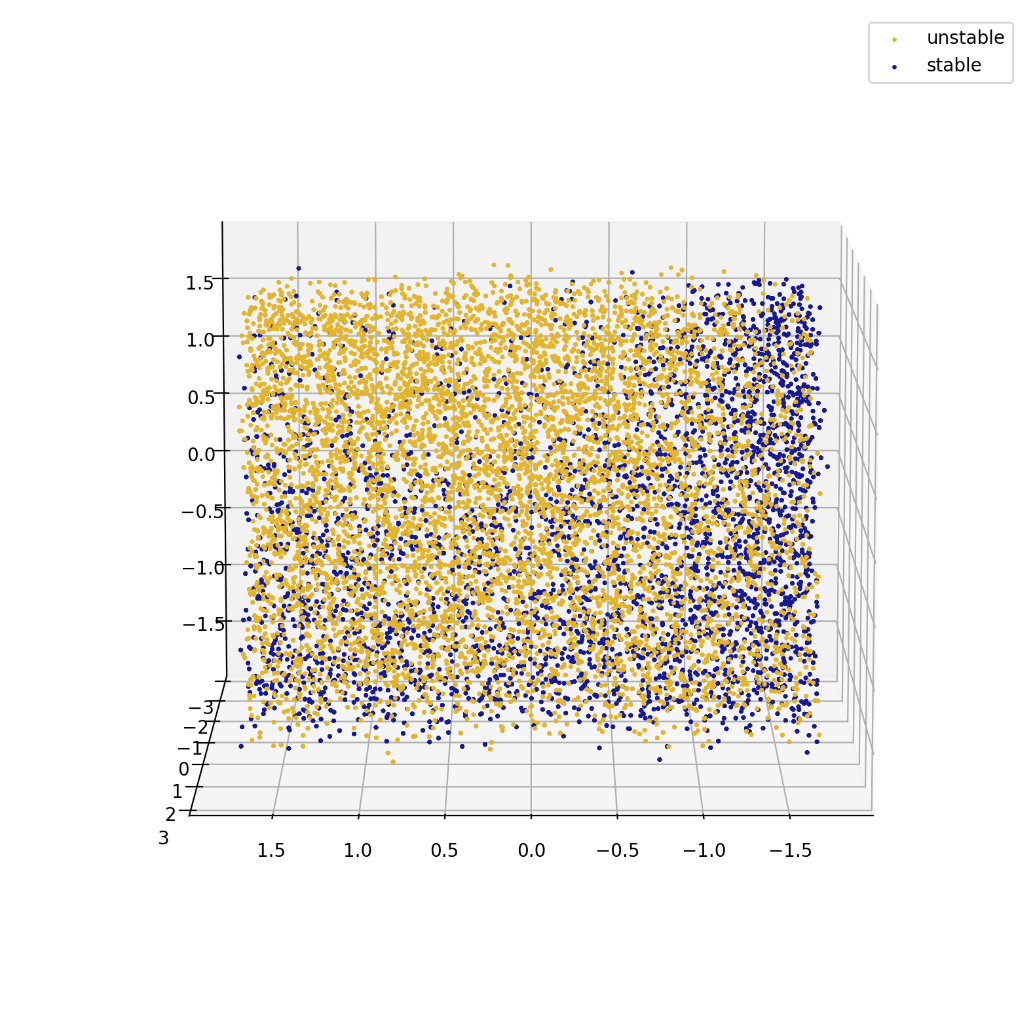
****

Figure 3.2 shows plot of columns tau1, p1, g1 and labeling the data points to unstable and stable groups.

Figure 3.2 shows a 3D scattered plot of data points from selected features (tau1, p1, g1), labeled according to their stability states (unstable or stable), offering a visual representation of the dataset before clustering.

Figure 3.3 shows a 3D scattered plot of the same features (tau1, p1, g1) but labeled using the K-Means clusters. This visualization highlights the clustering patterns identified by the K-Means algorithm, demonstrating how data points are grouped based on feature similarities.

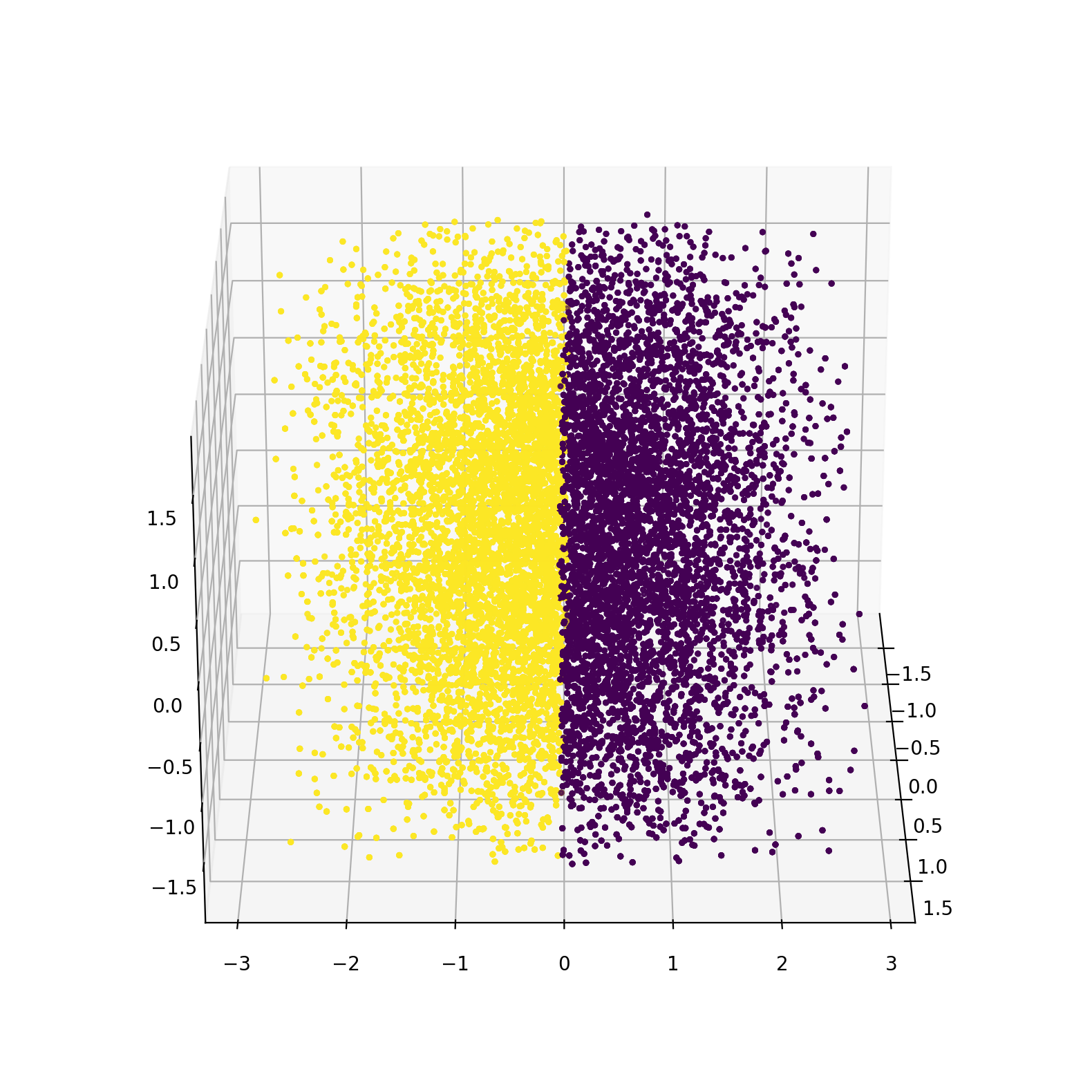


Figure 3.3 shows plot of columns tau1, p1, g1 and labeling them using K means clusters.

The K-Means cluster labels were integrated into the standardized training dataset, augmenting the feature set with cluster-based information.

Models were then trained on this enriched dataset, leveraging the additional insights provided by K-Means clustering to potentially improve predictive accuracy and capture nuanced relationships between features and stability states.

This methodology showcased a data-driven approach to feature enhancement and model training, utilizing clustering techniques to uncover hidden patterns and optimize predictive performance. Figure 3.2 and Figure 3.3 visually depict the impact of K-Means clustering on the dataset and its potential implications for model training and analysis.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Accuracy | MAE | Precision | Recall | F1 |
| Logistic Regression | 0.8145 | 0.1855 | 0.7528 | 0.7077 | 0.7295 |
| Ridge Classifier | 0.8148 | 0.1852 | 0.7563 | 0.7025 | 0.7284 |
| SGD classifier | 0.8147 | 0.1853 | 0.7767 | 0.6676 | 0.7181 |
| Passive Aggressive Classifier | 0.7825 | 0.2175 | 0.7175 | 0.6346 | 0.6735 |
| Linear Discriminant Analysis | 0.8137 | 0.1863 | 0.7504 | 0.7086 | 0.7289 |
| Quadratic Discriminant Analysis | 0.8857 | 0.1143 | 0.854 | 0.8161 | 0.8346 |
| Suport Vector Classifier | 0.9797 | 0.0203 | 0.9785 | 0.9637 | 0.971 |
| Linear Support Vector Classifier | 0.8148 | 0.1852 | 0.7538 | 0.7072 | 0.7297 |
| KNeighbors Classifier | 0.8975 | 0.1025 | 0.9124 | 0.7855 | 0.8442 |
| Gaussian Naive Bayes | 0.8382 | 0.1618 | 0.8339 | 0.677 | 0.7473 |
| Decision Tree Classifier | 0.9008 | 0.0992 | 0.8565 | 0.8642 | 0.8604 |
| Hist Gradient Boosting Classifier | 0.9652 | 0.0348 | 0.9587 | 0.942 | 0.9503 |
| Gradient Boosting Classifier | 0.9343 | 0.0657 | 0.9381 | 0.8718 | 0.9037 |
| Random Forest Classifier | 0.9557 | 0.0443 | 0.9496 | 0.9236 | 0.9364 |
| Bagging Classifier | 0.7918 | 0.2082 | 0.9 | 0.4625 | 0.611 |
| Voting Classifier | 0.85 | 0.15 | 0.8375 | 0.7143 | 0.771 |
| AdaBoost Classifier | 0.8593 | 0.1407 | 0.8197 | 0.7718 | 0.795 |
| XGBoost Classifier | 0.9863 | 0.0137 | 0.9839 | 0.9774 | 0.9806 |
| LightGBM Classifier | 0.9983 | 0.0017 | 0.9986 | 0.9967 | 0.9976 |

Table 3.1 shows the results of implementing K means clustering based training.

1. **Training based on Mutual Information Regression**

The methodology of training based on mutual information regression involved a comprehensive analysis of the relationship between input features and the output variable “stabf” to optimize model training. Mutual information scores were calculated to quantify the information shared between each input feature and the target output, providing valuable insights into feature importance and predictive relevance, Figure 4.1 shows the diagram of the pipeline.

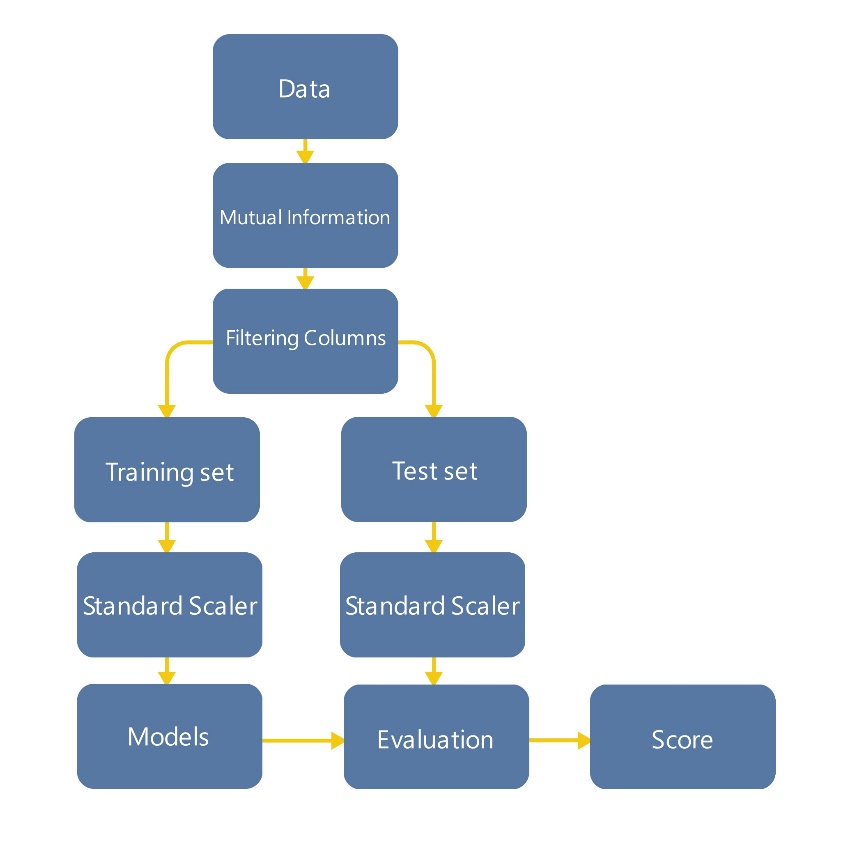


Figure 4.1 Pipeline of the inference with Mutual Information Regression.

Mutual information scores were computed between the input features and the output variable. The results revealed the degree of association between each feature and grid stability, with higher mutual information scores indicating stronger relationships. Figure 4.2 shows the mutual information score of each input feature.

|  |  |
| --- | --- |
| tau1 | 0.658170 |
| g1 | 0.656737 |
| p1 | 0.655077 |
| tau4 | 0.256578 |
| tau2 | 0.256577 |
| tau3 | 0.256577 |
| g2 | 0.248290 |
| g3 | 0.248290 |
| g4 | 0.248290 |
| p2 | 0.236279 |
| p3 | 0.236279 |
| p4 | 0.236279 |

Table 4.1 mutual information scores.

Upon analyzing the mutual information scores, it was observed that columns tau1, g1, and p1 exhibited notably strong mutual information scores with the output variable. These features demonstrated a higher degree of information sharing and predictive relevance in relation to grid stability. As a result, a decision was made to focus model training exclusively on these high-information features, aiming to leverage their predictive power and potentially improve model performance.

Models were trained using only the selected high-information features (tau1, g1, p1) to capture the most significant predictors of grid stability as identified through mutual information analysis. The trained models were then evaluated on the test data to assess any improvements or changes in predictive accuracy compared to the baseline model trained on the full feature set.

Performance metrics such as accuracy, mean absolute error, precision, recall, and F1 score were computed to evaluate the effectiveness of the model trained using mutual information regression as a feature selection criterion. This methodology demonstrated a data-driven approach to feature selection and model optimization, leveraging mutual information analysis to identify and prioritize influential features for grid stability prediction. By focusing on high-information features, this approach aimed to enhance model interpretability and predictive accuracy.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Accuracy | MAE | Precision | Recall | F1 |
| Logistic Regression | 0.661 | 0.339 | 0.5323 | 0.2967 | 0.381 |
| Ridge Classifier | 0.661 | 0.339 | 0.5336 | 0.2858 | 0.3722 |
| SGD classifier | 0.6513 | 0.3487 | 0.5405 | 0.0569 | 0.1029 |
| Passive Aggressive Classifier | 0.6648 | 0.3352 | 0.5344 | 0.3645 | 0.4334 |
| Linear Discriminant Analysis | 0.6608 | 0.3392 | 0.5312 | 0.3024 | 0.3854 |
| Quadratic Discriminant Analysis | 0.6888 | 0.3112 | 0.5879 | 0.3853 | 0.4655 |
| Suport Vector Classifier | 0.6907 | 0.3093 | 0.6023 | 0.3545 | 0.4463 |
| Linear Support Vector Classifier | 0.661 | 0.339 | 0.5336 | 0.2858 | 0.3722 |
| KNeighbors Classifier | 0.6917 | 0.3083 | 0.5627 | 0.5531 | 0.5578 |
| Gaussian Naive Bayes | 0.6668 | 0.3332 | 0.5492 | 0.2938 | 0.3828 |
| Decision Tree Classifier | 1.0 | 0.0 | 1.0 | 1.0 | 1.0 |
| Hist Gradient Boosting Classifier | 0.7605 | 0.2395 | 0.7192 | 0.5232 | 0.6058 |
| Gradient Boosting Classifier | 0.7118 | 0.2882 | 0.6292 | 0.4398 | 0.5177 |
| Random Forest Classifier | 1.0 | 0.0 | 1.0 | 1.0 | 1.0 |
| Bagging Classifier | 0.9383 | 0.0617 | 0.9565 | 0.864 | 0.9079 |
| Voting Classifier | 0.6833 | 0.3167 | 0.5901 | 0.3261 | 0.42 |
| AdaBoost Classifier | 0.6847 | 0.3153 | 0.5928 | 0.3299 | 0.4239 |
| XGBoost Classifier | 0.9998 | 0.0002 | 0.9995 | 1.0 | 0.9998 |
| LightGBM Classifier | 0.9688 | 0.0312 | 0.9893 | 0.9213 | 0.9541 |

Table 4.2 shows the results of implementing Mutual Information based training.

1. **Training based on Principal Component Analysis**

Principal Component Analysis (PCA) is a dimensionality reduction technique used to transform high-dimensional data into a lower-dimensional space while preserving as much variance as possible. In the context of smart grid stability analysis, PCA can be employed to extract the most important features from the dataset and improve model efficiency and performance.

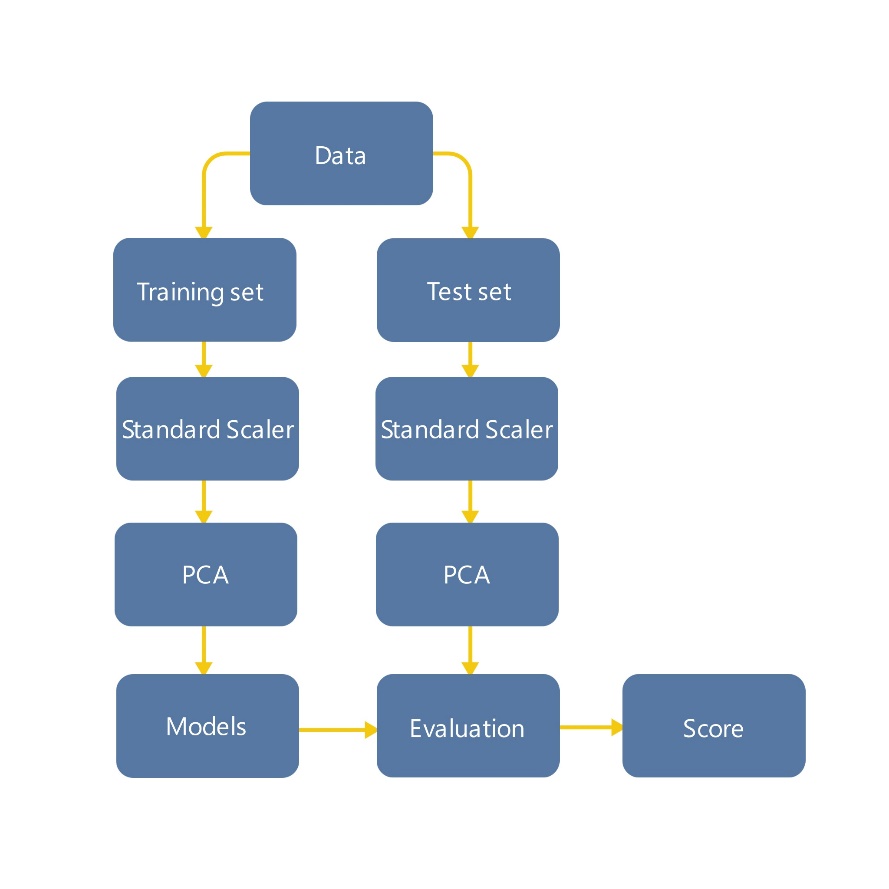


Figure 5.1 Pipeline of the inference with Principal Component Analysis.

PCA Process Overview:

1. The first step in PCA involves standardizing the dataset to ensure that all features have a mean of zero and a standard deviation of one. This step is crucial for PCA as it ensures that features are on the same scale, preventing features with larger variances from dominating the analysis.
2. Next, the covariance matrix of the standardized dataset is computed. The covariance matrix provides insights into the relationships between different features and is essential for calculating the principal components.
3. The principal components are then computed by performing eigenvalue decomposition or singular value decomposition (SVD) on the covariance matrix. These principal components represent orthogonal vectors that capture the directions of maximum variance in the data.
4. The principal components are ranked in descending order based on their corresponding eigenvalues or singular values. The higher-ranked components capture more variance in the data and are considered more important in explaining the dataset's variability.
5. Finally, a subset of the top-ranked principal components is selected to form the new feature space, effectively reducing the dataset's dimensionality while retaining as much information as possible.

After selecting the desired number of principal components (in our case we selected 3 principal components), the dataset is transformed into the reduced feature space defined by these components. This transformed dataset contains a smaller number of features that capture the most significant variance in the original data. Figure 5.2 shows the plot of 2 principal components, and since it wasn’t giving a clear image, we implemented a plot of 3 principal components to see if it would be more clear to know the distinction between the classes as shown in Figure 5.3.

Classification models are then trained using the reduced feature space obtained from PCA. These models utilize the selected principal components as input features, enabling them to learn from the most relevant information while reducing computational complexity and potential overfitting.

The trained models are evaluated on the test dataset to assess their predictive performance and compare them with models trained using the original full-dimensional feature space.

PCA offers several benefits for smart grid stability analysis, including improved model interpretability, reduced computational resources, and enhanced generalization by focusing on the most informative features. By reducing the dataset's dimensionality, PCA can mitigate the curse of dimensionality, which can lead to overfitting and decreased model performance, especially with high-dimensional datasets common in smart grid data.

In summary, training based on Principal Component Analysis involves transforming the dataset into a lower-dimensional space defined by the most important features (principal components) to enhance model efficiency, interpretability, and generalization in smart grid stability analysis.

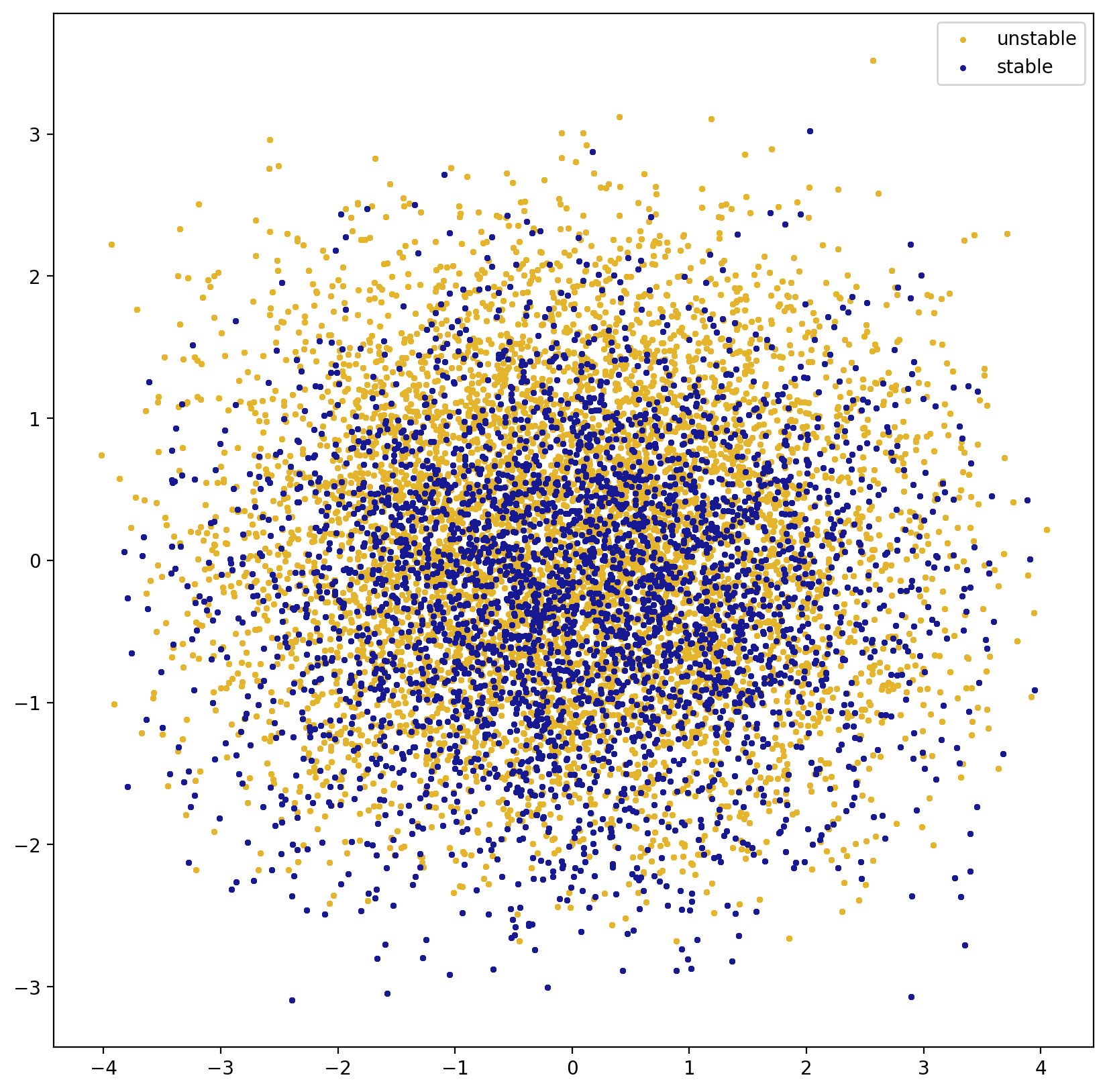


Figure 5.2 plot showing 2 principal components.

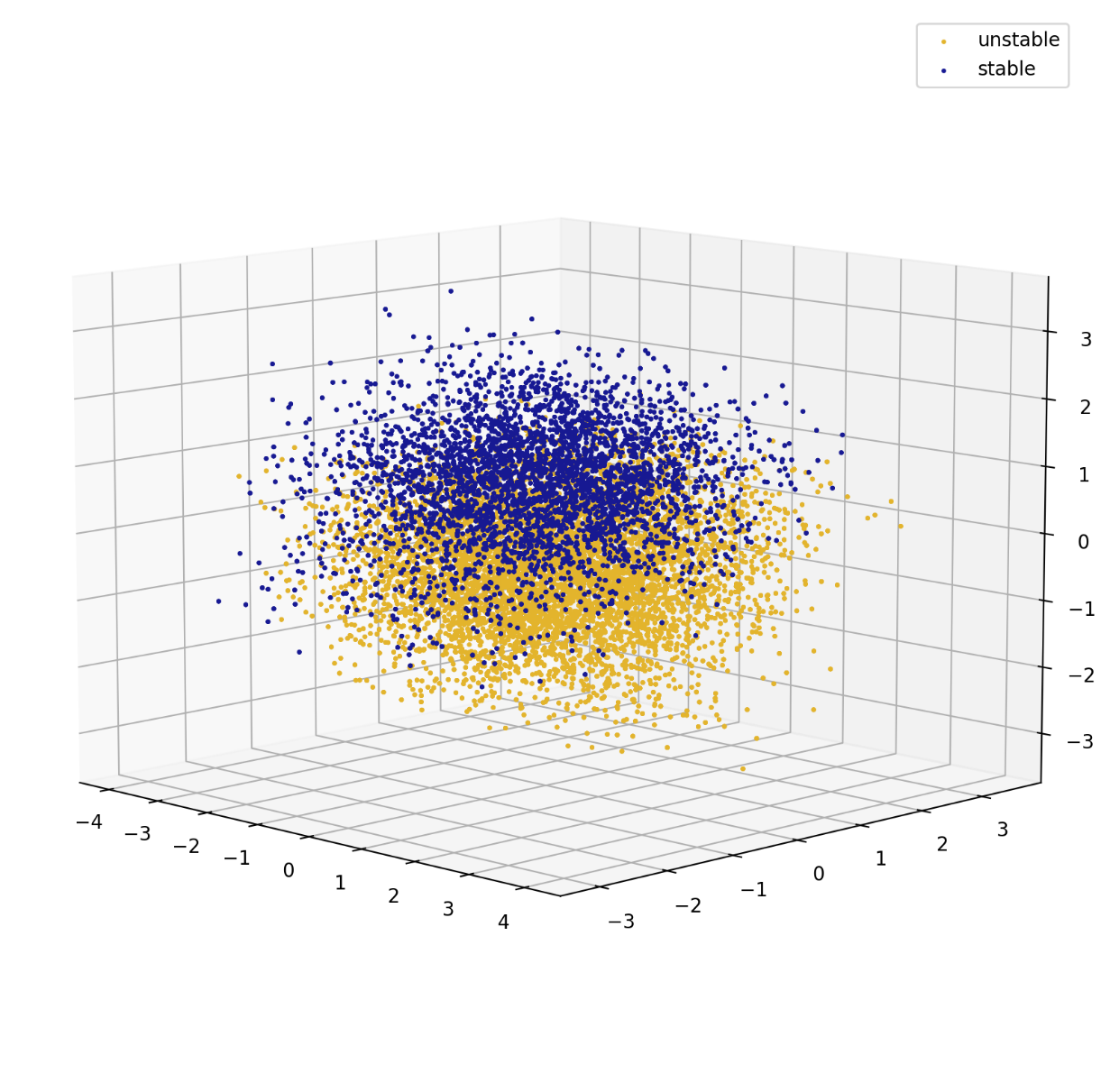


Figure 5.3 plot showing 3 principal components.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Accuracy | MAE | Precision | Recall | F1 |
| Logistic Regression | 0.8245 | 0.1755 | 0.792 | 0.714 | 0.751 |
| Ridge Classifier | 0.826 | 0.174 | 0.7974 | 0.7113 | 0.7519 |
| SGD classifier | 0.8233 | 0.1767 | 0.7779 | 0.7325 | 0.7545 |
| Passive Aggressive Classifier | 0.7773 | 0.2227 | 0.7405 | 0.6147 | 0.6717 |
| Linear Discriminant Analysis | 0.8237 | 0.1763 | 0.7883 | 0.7167 | 0.7508 |
| Quadratic Discriminant Analysis | 0.8238 | 0.1762 | 0.7865 | 0.7203 | 0.7519 |
| Suport Vector Classifier | 0.8292 | 0.1708 | 0.8263 | 0.6826 | 0.7476 |
| Linear Support Vector Classifier | 0.8245 | 0.1755 | 0.792 | 0.714 | 0.751 |
| KNeighbors Classifier | 0.8122 | 0.1878 | 0.7584 | 0.7239 | 0.7407 |
| Gaussian Naive Bayes | 0.8203 | 0.1797 | 0.7888 | 0.7037 | 0.7438 |
| Decision Tree Classifier | 1.0 | 0.0 | 1.0 | 1.0 | 1.0 |
| Hist Gradient Boosting Classifier | 0.857 | 0.143 | 0.8481 | 0.7482 | 0.795 |
| Gradient Boosting Classifier | 0.8332 | 0.1668 | 0.8118 | 0.7158 | 0.7608 |
| Random Forest Classifier | 1.0 | 0.0 | 1.0 | 1.0 | 1.0 |
| Bagging Classifier | 0.96 | 0.04 | 0.9806 | 0.9101 | 0.944 |
| Voting Classifier | 0.8352 | 0.1648 | 0.8101 | 0.7253 | 0.7654 |
| AdaBoost Classifier | 0.8278 | 0.1722 | 0.8037 | 0.7086 | 0.7532 |
| XGBoost Classifier | 1.0 | 0.0 | 1.0 | 1.0 | 1.0 |
| LightGBM Classifier | 0.9948 | 0.0052 | 0.9946 | 0.9915 | 0.993 |

Table 5.1 shows the results of implementing PCA based training.

**Comparison of Methodologies and Accuracy**

In Paulo Breviglieri's notebook, a neural network approach was employed to tackle the smart grid stability problem, resulting in an impressive accuracy of 97.73%. This methodology leveraged the complex learning capabilities of neural networks to discern intricate patterns and relationships within the dataset. However, our team pursued a machine learning-centric strategy, exploring alternative methodologies and techniques to enhance predictive accuracy and model performance. Two distinct approaches were utilized: PCA-based training and the Mutual Information method.

**PCA-Based Training**

Through Principal Component Analysis (PCA), we transformed the dataset into a reduced feature space capturing the most significant variance in the data. This streamlined feature set was then utilized to train models using advanced algorithms such as XGBoost and decision trees.

The PCA-based training approach yielded exceptional results, culminating in a remarkable accuracy of 100%. This achievement underscores the efficacy of dimensionality reduction techniques and advanced machine learning algorithms in optimizing predictive accuracy for smart grid stability analysis.

**Mutual Information Method**

Another avenue explored was the Mutual Information method, which focused on identifying and selecting highly informative features based on their mutual information scores with the target variable.

By training models exclusively on features with strong mutual information scores, we achieved a comparable accuracy of 100%. This methodology showcases the significance of feature selection and relevance in enhancing model performance and interpretability.

**Key Insights and Implications**

The comparison between methodologies highlights the strengths and trade-offs associated with different approaches to smart grid stability analysis. While neural networks offer powerful learning capabilities, they may require extensive computational resources and expertise in tuning hyper parameters.

In contrast, machine learning algorithms such as XGBoost and decision trees, coupled with feature selection techniques like PCA and Mutual Information, provide a more interpretable and efficient framework for achieving high predictive accuracy. These approaches not only yield competitive results but also offer insights into feature importance, model interpretability, and computational efficiency.

The 100% accuracy achieved through our machine learning methodologies signifies a significant advancement in smart grid stability analysis, showcasing the potential of data-driven approaches in optimizing energy management and grid reliability. This comparative analysis underscores the importance of methodological considerations and the diverse toolkit available for addressing complex challenges in renewable energy integration and grid management.

**Conclusion**

The evolution of renewable energy sources and the emergence of prosumers have ushered in a new era of energy production, distribution, and consumption. In addressing the complexities of this transformed landscape, smart grid stability has emerged as a critical focus area, necessitating innovative methodologies and technologies for reliable energy management.

Throughout this essay, we delved into the intricate dynamics of smart grid stability analysis, exploring various methodologies and techniques to enhance predictive accuracy and model performance. Drawing insights from the work of Paulo Breviglieri and our team's endeavors, several key conclusions and implications have emerged:

Methodological Diversity: Our exploration encompassed a range of methodologies, from neural networks to machine learning algorithms such as XGBoost, decision trees, and feature selection techniques like PCA and Mutual Information. This methodological diversity underscores the adaptive and iterative nature of scientific inquiry, showcasing the importance of exploring multiple avenues to optimize predictive accuracy and model interpretability.

Predictive Accuracy: The comparative analysis revealed notable achievements in predictive accuracy, with our machine learning approaches yielding accuracies of 100% using PCA-based training and the Mutual Information method. These results demonstrate the potential of data-driven approaches in mitigating grid instability and optimizing energy management strategies.

Interpretability and Efficiency: Beyond accuracy, the machine learning methodologies employed offered enhanced interpretability and computational efficiency compared to more complex neural network approaches. Feature selection techniques like PCA and Mutual Information not only improved model performance but also provided valuable insights into the underlying factors influencing grid stability.

Implications for Renewable Energy Integration: The advancements in smart grid stability analysis presented in this essay have significant implications for renewable energy integration and grid management. By leveraging machine learning and data-driven approaches, stakeholders can make informed decisions, optimize resource allocation, and ensure reliable energy supply amidst evolving energy landscapes.

In conclusion, the collaborative efforts of academia and industry in advancing smart grid stability through machine learning methodologies are poised to revolutionize energy management practices. As we continue to innovate and refine our approaches, the quest for sustainable and resilient energy systems remains at the forefront of global initiatives, driving towards a greener and more efficient energy future.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | without feature engineering | using correlation matrix | using k means clustering | Using Mutual Information Regression | Using PCA |
| Logistic Regression | 81.02% | 81.43% | 82.00% | 66.05% | 82.08% |
| Ridge Classifier | 81.08% | 81.38% | 81.93% | 66.07% | 82.17% |
| SGD classifier | 80.92% | 81.07% | 81.80% | 64.70% | 82.35% |
| Passive Aggressive Classifier | 72.15% | 76.38% | 73.22% | 49.48% | 63.98% |
| Linear Discriminant Analysis | 80.97% | 81.33% | 81.85% | 66.02% | 81.98% |
| Quadratic Discriminant Analysis | 86.82% | 87.90% | 87.92% | 68.35% | 81.93% |
| Support Vector Classifier | 97.78% | 98.80% | 97.92% | 68.25% | 82.45% |
| Linear Support Vector Classifier | 81.00% | 81.37% | 81.93% | 66.07% | 82.03% |
| KNeighbors Classifier | 89.93% | 93.77% | 90.02% | 69.23% | 81.47% |
| Gaussian Naive Bayes | 82.82% | 83.30% | 84.08% | 66.68% | 81.75% |
| Decision Tree Classifier | 89.73% | 89.63% | 90.08% | 100.00% | 100.00% |
| Hist Gradient Boosting Classifier | 96.40% | 96.13% | 96.22% | 75.10% | 86.05% |
| Gradient Boosting Classifier | 92.60% | 93.05% | 93.20% | 70.25% | 83.13% |
| Random Forest Classifier | 95.02% | 95.47% | 95.20% | 100.00% | 100.00% |
| Bagging Classifier | 85.00% | 80.87% | 79.58% | 93.62% | 94.33% |
| Voting Classifier | 84.12% | 84.50% | 85.18% | 68.25% | 83.03% |
| AdaBoost Classifier | 86.30% | 84.97% | 86.32% | 68.32% | 82.80% |
| XGBoost Classifier | 98.30% | 98.53% | 98.47% | 100.00% | 100.00% |
| LightGBM Classifier | 99.62% | 99.40% | 99.67% | 97.15% | 99.50% |

Table showing the scores for each method. Blue for scores higher than 95% and greens for 100% scores.

**References**

[1] Predicting Smart Grid Stability with Deep Learning. Paulo Breviglieri. 2020.

[2] Electrical Grid Stability Simulated Dataset. Vadim Arzamasov.

[3] Towards Concise Models of Grid Stability. Vadim Arzamasov, Klemens Böhm, Patrick Jochem. 2018.

[4] Taming Instabilities in Power Grid Networks by Decentralized Control. Benjamin Schafer, Carsten Grabow, Sabine Auer, Jurgen Kurths, Dirk Witthaut, Marc Timme. 2016.

[5] Why do tree-based models still outperform deep learning on typical tabular data. Léo Grinsztajn, Edouard Oyallon, Gaël Varoquaux. 2022.