**Last Task – Explainable AI (XAI)**

# **Objective**

The goal of this task is to explore and implement Explainable AI (XAI) techniques to interpret regression models trained on numeric datasets from the EU and USA networks. To apply XAI methods, to understand the decision-making process of these models and assess the effectiveness of XAI in providing insights into model behavior. The task also involves comparing the insights provided by XAI with those obtained from traditional models.

# **Understanding Explainable AI (XAI)**

## **Introduction**

To start, we will need to familiarize ourselves with the core concepts and techniques of Explainable AI (XAI) that are specifically applicable to regression models. Following is a breakdown of what each technique involves and how it can be used:

## **1. Feature Importance**

* **Concept**: Feature importance quantifies the contribution of each feature to the model’s predictions. It helps in identifying which features are most influential in the decision-making process.
* **Application**: Many models, such as Random Forests and Gradient Boosting, have built-in methods to compute feature importance. For example, in a Random Forest, feature importance can be determined by measuring the increase in the model’s prediction error when the feature's values are permuted.

## **2. Partial Dependence Plots (PDPs)**

* **Concept**: PDPs show the relationship between a specific feature and the target variable, averaging out the influence of all other features. This helps in visualizing how changes in a feature affect the model's predictions.
* **Application**: PDPs can be used to visualize non-linear relationships and interactions between features. For example, in a housing price model, a PDP might show how house prices change with the number of rooms, holding all other variables constant.

## **3. Individual Conditional Expectation (ICE) Plots**

* **Concept**: ICE plots are similar to PDPs but focus on individual data points. Instead of averaging the effects across the entire dataset, ICE plots show the effect of a feature on each instance.
* **Application**: ICE plots help in understanding how the model’s predictions for individual instances change when a particular feature is altered, which is useful for identifying heterogeneity in feature effects.

## **4. SHAP Values (Shapley Additive Explanations)**

* **Concept**: SHAP values provide a unified framework to explain the output of any machine learning model. They are based on cooperative game theory and represent the contribution of each feature to a particular prediction.
* **Application**: SHAP values can be used to explain individual predictions by showing how much each feature contributed positively or negatively to the final prediction. This method is model-agnostic and works well with complex models like neural networks.

## **5. LIME (Local Interpretable Model-agnostic Explanations)**

* **Concept**: LIME approximates the model locally around a prediction with a simpler, interpretable model. It helps in understanding why the model made a specific prediction for a particular instance.
* **Application**: LIME is useful for interpreting predictions from black-box models. For example, it can explain why a neural network predicted a certain house price by approximating the network’s behavior with a linear model in the vicinity of the prediction.

## **Recommended Resources for Learning**

### **Books**

* "Interpretable Machine Learning" by Christoph Molnar (available online): This book provides an in-depth guide to XAI techniques, including PDPs, ICE plots, SHAP, and LIME.
* "Machine Learning Interpretability" by Patrick Hall, Navdeep Gill, and Nicholas Schmidt: This book offers practical insights into applying XAI methods in real-world scenarios.

### **Academic Papers**

* "A Unified Approach to Interpreting Model Predictions" by Scott M. Lundberg and Su-In Lee: This paper introduces SHAP values and is fundamental to understanding feature importance in complex models.
* "Why Should I Trust You?": Explaining the Predictions of Any Classifier by Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin: This paper introduces LIME and provides a framework for interpreting black-box models.

### **Online Tutorials and Courses**

* **Kaggle**: Kaggle offers tutorials and notebooks that explain how to implement SHAP and LIME in Python.
* **Coursera**: Courses like "Interpretable Machine Learning" and "AI for Everyone" can provide foundational knowledge on XAI.
* **Blogs and Articles**: Websites like Towards Data Science and Analytics Vidhya frequently publish articles on XAI techniques with code examples.

# **Data Preprocessing for both Datasets**

Data preprocessing is an important step, as it refers to the cleaning, transforming, and integrating of data in order to make it ready for analysis. The goal of data preprocessing is to improve the quality of the data and to make it more suitable for the specific task.

**We have used channel 1 (GSNR\_1) as the target variable.**

## **Library Imports**

The code begins by mounting Google Drive to access the dataset stored there. It then imports essential libraries such as pandas for data handling and StandardScaler from sklearn for normalization.

## **Data Loading**

The datasets are read from Excel files containing numeric data from the EU and USA networks, and the initial rows are previewed to verify successful loading.

## **Attribute and Target Selection**

The code identifies 306 attribute columns, including power, ASE, and NLI measurements, along with spans and total distance. The target column is 'GSNR\_1', which is the focus of the regression task. At first, we have 306 attribute columns but then we remove the frequency columns from the attribute columns as these columns have almost zero correlation with other attribute columns.

## **Data Normalization**

Features and the target variable are normalized using StandardScaler to scale the data. This ensures all features are on a similar scale, preventing any single feature from disproportionately influencing the model's performance.

## **Custom Train-Test Split**

The custom\_train\_test\_split function splits the data path-wise, recognizing that the dataset changes every 3000 samples. It splits each block into training and testing sets, ensuring that the data is correctly partitioned. This approach is particularly useful for datasets with structured changes or segments.

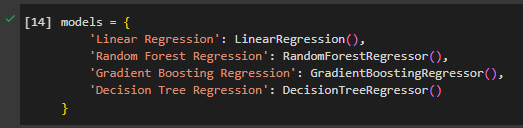
## **Resulting Data Shapes**

After splitting, the training data comprises 30,000 samples with 230 features, while the testing set has 6,000 samples. The labels (target values) are split correspondingly, ensuring the model has adequate data for training and evaluation.

# **Model Training and Evaluation Regression Models**

## **Model Initialization**

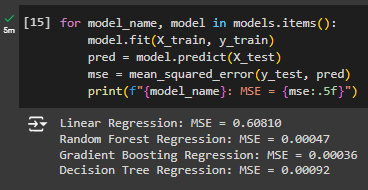
Four different regression models were initialized for the task: Linear Regression, Random Forest Regressor, Gradient Boosting Regressor, and Decision Tree Regressor. These models were chosen for their diverse approaches to regression, ranging from simple linear methods to more complex tree-based algorithms.



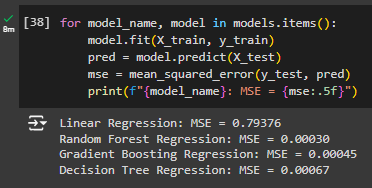
## **Model Training and MSE Calculation**

Each model was trained on the training data (X\_train, y\_train) and then used to predict the target variable on the test set (X\_test). The performance of each model was evaluated using the Mean Squared Error (MSE) metric, which measures the average squared difference between the predicted and actual values. The MSE values were printed for comparison, highlighting how well each model performed in predicting the target variable from the EU dataset.

### **Model Results for EU Dataset**



### **Model Results for USA Dataset**



# **Explainable AI (XAI) Implementation for EU Dataset**

## **Library Installation and Import**

The code begins by installing and importing necessary libraries for Explainable AI (XAI), including SHAP, LIME, and PDPbox. These tools are essential for interpreting the predictions of machine learning models and understanding how features influence model decisions.

## **Feature Selection**

A set of key features (Power\_1, NLI\_1, ASE\_1, No. Spans, Total Distance(m)) was selected for interpretation. These features were chosen based on their relevance to the problem, and the SHAP analysis focused on them to provide clearer insights.

## **SHAP Analysis**

For each regression model (Linear Regression, Random Forest Regressor, Gradient Boosting Regressor, Decision Tree Regressor), SHAP (SHapley Additive exPlanations) was applied to generate insights into the model’s decision-making process:

* **Global Interpretability with SHAP Summary Plots**: SHAP summary plots were generated to visualize the impact of each feature on the model’s predictions across the test dataset. These plots help in understanding which features are most influential in the model’s predictions.
* **Feature Dependence with SHAP Dependence Plots**: SHAP dependence plots were created for each of the selected features. These plots illustrate how changes in a specific feature influence the prediction, providing a more detailed view of feature interactions.

### **SHAP Summary Plot Explanation**

This SHAP summary plot visualizes the impact of the selected features (Power\_1, NLI\_1, Total Distance(m), No. Spans, and ASE\_1) on the model's output. Here's a brief breakdown of the plot:

**1. Features on the Y-axis:** The features are listed on the Y-axis in order of importance, with the most influential feature at the top (Power\_1) and the least influential at the bottom (ASE\_1).

**2. SHAP Values on the X-axis:** The X-axis represents the SHAP values, which indicate the impact of each feature on the model's prediction. A positive SHAP value pushes the prediction higher, while a negative SHAP value pushes it lower.

**3. Color Coding:** The color of each dot represents the feature value, with red indicating high feature values and blue indicating low feature values.

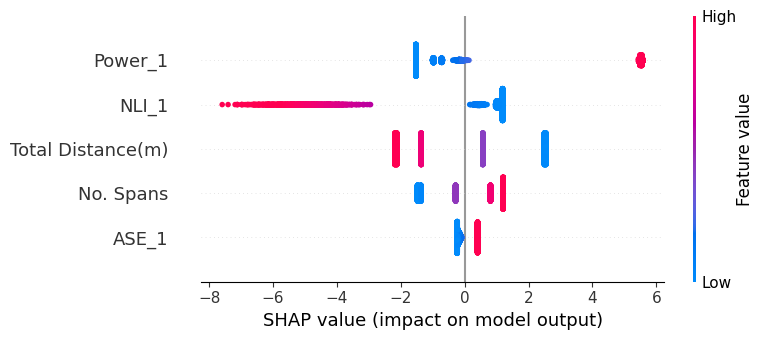
**4. Feature Impact:**

* **Power\_1:** High values (red dots) are associated with a significant positive impact on the model's prediction, while low values (blue dots) tend to push the prediction lower.
* **NLI\_1:** This feature has a consistently negative impact on the model's output, with little variation, indicating that higher values of NLI\_1 generally lead to lower predictions.
* **Total Distance(m), No. Spans, and ASE\_1**: These features have varying impacts on the model's prediction, with some values leading to positive impacts and others to negative impacts, suggesting complex interactions with other features.

**5. Insights:**

* Power\_1 has the most significant influence on the model’s predictions, particularly when it has high values.
* NLI\_1 consistently lowers the prediction, regardless of its value, suggesting it's a strong negative predictor.
* Other features show more mixed impacts, suggesting their influence depends on the context or interaction with other features.

This plot provides a global overview of feature importance and their contribution to the model's output, helping in understanding which features drive the model's predictions and how.



### **Visualization**

Each SHAP summary and dependence plot was generated with specific figure sizes to ensure clarity and focus on the most relevant aspects of the data. These visualizations offer a comprehensive overview of how each model interprets the input features and makes predictions, aiding in comparing model behavior across different algorithms.

## **LIME (Local Interpretable Model-agnostic Explanations) Analysis**

The LIME (Local Interpretable Model-agnostic Explanations) framework is used to provide local interpretability for machine learning models. It explains how specific features influence the prediction for a single instance in the test set. The LimeTabularExplainer is initialized with the training data and is used to generate an explanation for the first instance in the test set. The explanation is filtered to focus on a set of key features (features\_to\_plot). The filtered explanations highlight how much each of these selected features contributed to the prediction. The results are then printed and can also be visualized directly in a notebook for easier interpretation.

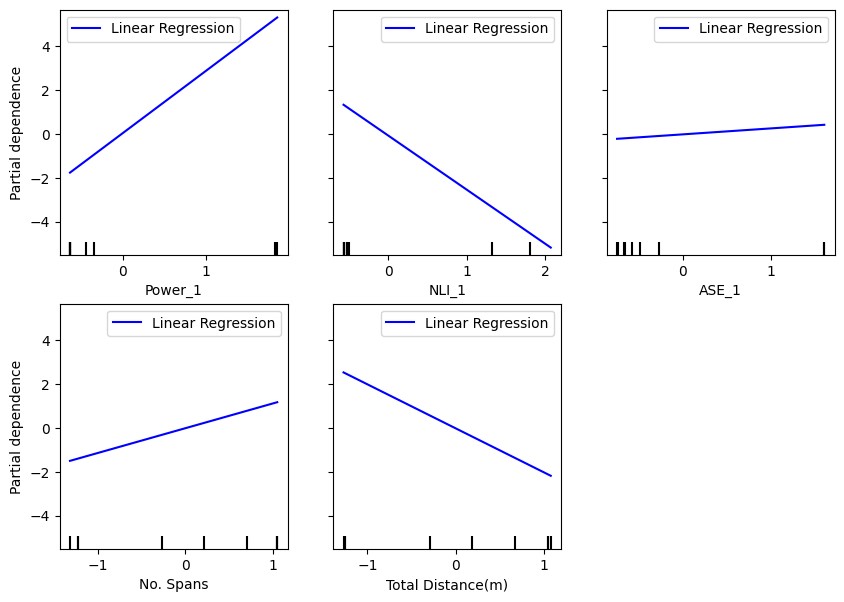
## **PDP (Partial Dependence Plots) Analysis**

In this analysis, I evaluated multiple models on a dataset to understand the influence of specific features using Partial Dependence Plots (PDPs). The process involved training each model on normalized training data (X\_train, y\_train) and then predicting outcomes on the test set (X\_test). For each model, I calculated the Mean Squared Error (MSE) to gauge its performance.

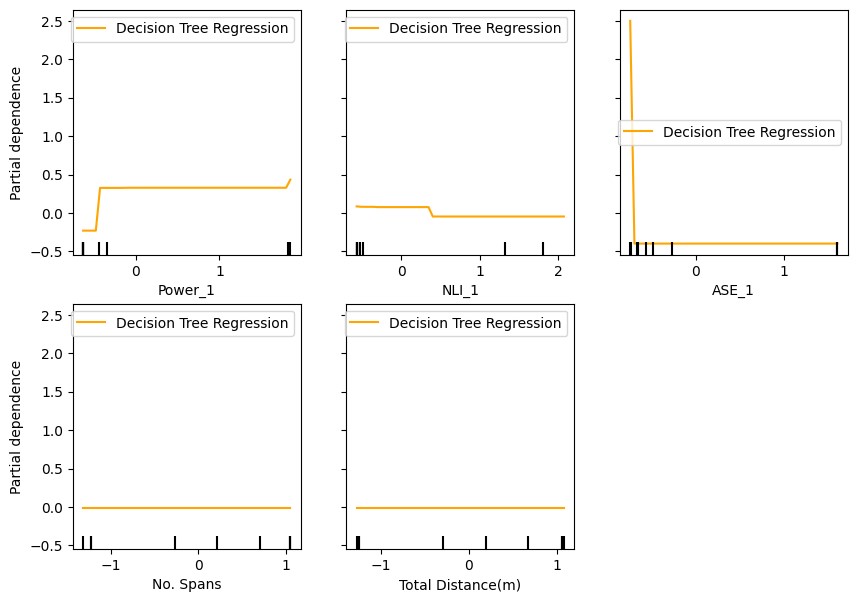
After training, I generated PDPs for selected features, which depict the marginal effect of each feature on the predicted outcome while holding other features constant. This was done using PartialDependenceDisplay with a high grid resolution for smoother visualizations. Each model's PDP was plotted with a distinct color for clear comparison. The PDPs provide insights into how different models interpret the influence of specific features, revealing their importance and contribution to the prediction outcomes across different algorithms.

By visually comparing the PDPs, it was possible to identify which features had the most significant impact on predictions and how their effects varied between models. This approach aids in understanding model behavior and enhances interpretability, especially in complex models.

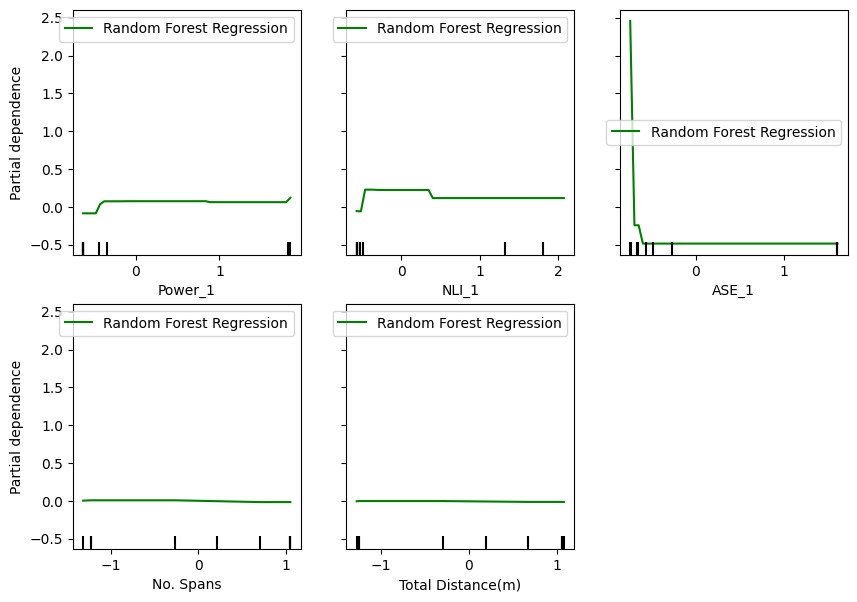
### **PDP Plots for Linear Regression**



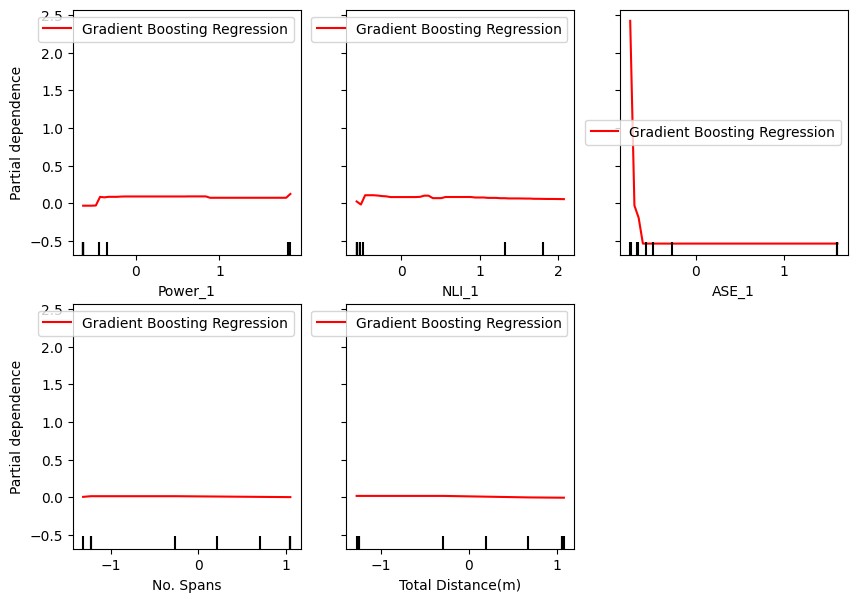
### **PDP Plots for Decision Tree**



### **PDP Plots for Random Forest**



### **PDP Plots for Gradient Boost**



## **ICE (Individual Conditional Expectation) Analysis**

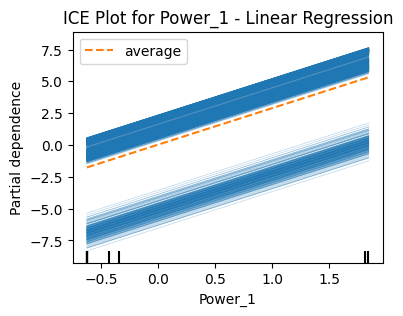
In this analysis, Individual Conditional Expectation (ICE) plots were generated to visualize the influence of specific features on the predictions of various models. ICE plots provide a detailed view of how changes in a feature value affect the model's prediction for individual instances, as opposed to the global average effect shown in Partial Dependence Plots (PDPs).

For each model, after fitting it to the training data (X\_train, y\_train), ICE plots were generated for selected features. The plots include both the PDP (average line) and the ICE curves for individual instances, allowing us to observe not just the overall trend but also the variability in predictions across different data points.

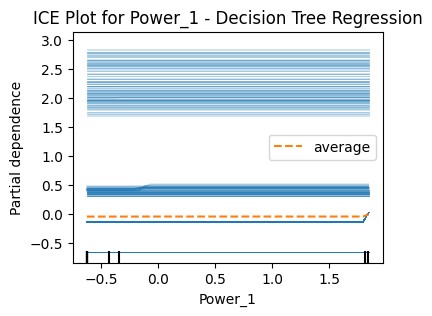
The ICE plots were generated using a high grid resolution to ensure smooth and accurate representations of the feature's impact on the model's predictions. Each feature was plotted separately, and the results were examined to understand how different models interpret the influence of specific features on the target variable.

By analyzing these ICE plots, we can gain insights into the consistency and variability of the model's predictions concerning individual feature values. This method helps in identifying interactions and non-linear relationships that may not be evident from PDPs alone, thus enhancing the interpretability of the models.

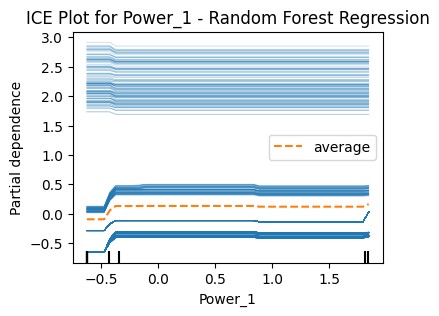
### **Linear Regression ICE Plot for Power\_1 attribute**



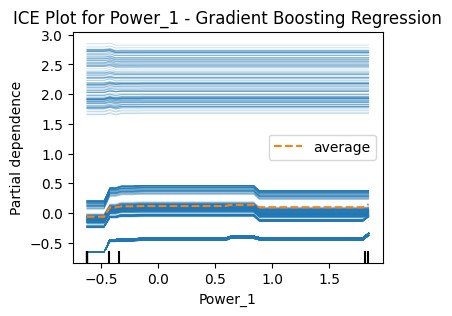
### **Decision Tree ICE Plot for Power\_1 attribute**



### **Random Forest ICE Plot for Power\_1 attribute**



### **Gradient Boosting ICE Plot for Power\_1 attribute**



# **Conclusion**

In this analysis, the influence of crucial factors such as Power, Amplified Spontaneous Emission (ASE), Nonlinear Interference (NLI), Number of Spans, and Total Distance on the Global Signal-to-Noise Ratio (GSNR) in optical communication networks was thoroughly examined. A variety of models, including Linear Regression, Random Forest, Gradient Boosting, and Decision Trees, were evaluated using interpretability techniques such as Partial Dependence Plots (PDPs), SHAP values, LIME, and Individual Conditional Expectation (ICE) plots. These tools provided valuable insights into the behavior and performance of each model in relation to the features impacting GSNR.

## **Key Takeaways**

### **Power**

* Across most models, an increase in power typically resulted in an increase in GSNR.
* The relationship between power and GSNR was sometimes nonlinear, with higher power levels eventually leading to signal degradation due to nonlinear effects like NLI.

### **Amplified Spontaneous Emission (ASE)**

* ASE emerged as a significant factor affecting GSNR across different models.
* Higher ASE levels consistently degraded GSNR, especially in Random Forest and Gradient Boosting models, where ASE was identified as one of the most influential features.

### **Nonlinear Interference (NLI)**

* NLI played a crucial role, particularly in scenarios involving higher power levels or longer distances.
* Higher NLI levels generally led to a reduction in GSNR, which aligns with the theoretical understanding that nonlinear effects can deteriorate signal quality.

### **Number of Spans**

* The impact of the number of spans on GSNR varied across models.
* Generally, an increase in spans led to a reduction in GSNR due to the cumulative effects of noise and nonlinearities. However, in some models like Random Forest, a mid-range number of spans resulted in higher GSNR, suggesting that a balance between signal regeneration and noise introduction might be optimal.

### **Total Distance**

* Longer distances typically reduced GSNR due to the accumulation of attenuation, ASE, and NLI.
* In certain models, a more complex relationship was observed, where mid-range distances offered better GSNR compared to very short or very long distances.

## **Insights from Model Interpretations**

* **SHAP Values**: Provided a clear ranking of feature importance, highlighting that Power, ASE, and NLI were the most critical factors affecting GSNR. ASE often exhibited the strongest negative impact on GSNR.
* **PDP and ICE Plots**: Revealed detailed insights into feature interactions and confirmed the nonlinear relationships between features like Power and NLI. These plots underscored the importance of optimizing these features for better network performance.
* LIME Explanations: Showed that ASE's impact could vary depending on the specific model, indicating complex interactions between ASE, Power, and NLI across different modeling approaches.

## **Final Conclusion**

The GSNR in optical networks is highly sensitive to the interaction of multiple factors, making it crucial to manage these elements carefully to achieve optimal performance. The analysis demonstrates that:

* Power levels need to be carefully balanced. While increasing power generally improves GSNR, it must be done cautiously to avoid excessive nonlinear effects that can degrade signal quality.
* ASE (Amplified Spontaneous Emission) should be minimized through efficient amplification strategies, as it consistently showed a significant negative impact on GSNR across different models.
* NLI (Nonlinear Interference) must be managed effectively, especially in scenarios involving higher power or longer transmission distances. NLI was identified as a critical factor that can substantially reduce GSNR if not properly controlled.

The use of advanced interpretability techniques such as SHAP values, PDPs, ICE plots, and LIME provided deep insights into how these factors interact and affect GSNR. These insights are invaluable for guiding the optimization of optical network performance, ensuring that networks are designed to balance signal strength, noise reduction, and nonlinear effects effectively. The findings from this analysis offer a roadmap for enhancing the reliability and efficiency of optical communication networks by carefully tuning these key parameters.

**THE END**