ML Model Build Report - Classification

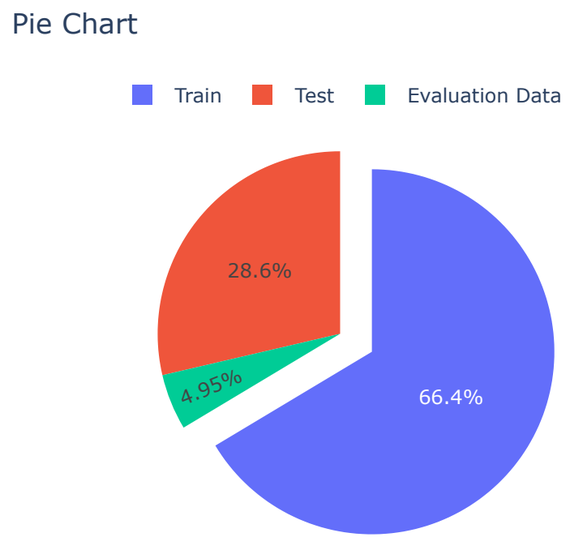
Data Split

Based on the given data split summary, we can analyze the distribution of data across different categories. The training dataset ("train_data_size") accounts for approximately 510 examples, which is a significant portion of the total data. This suggests that the model will have access to a large amount of data during training, which can help improve its performance and generalization abilities.

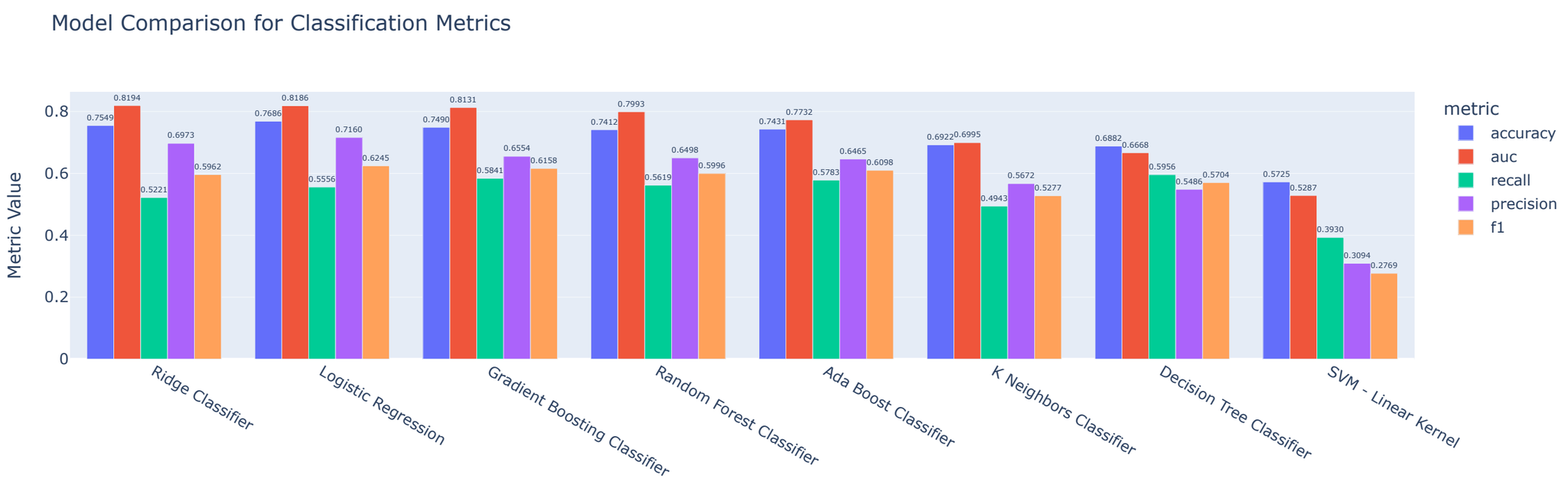
In comparison, the test dataset ("test_data_size") consists of around 220 examples, which is slightly smaller than the training dataset. This indicates that the model will be evaluated on a diverse set of examples after training, allowing us to assess its performance on unseen data.

Finally, the evaluation dataset ("evaluation_data_size") contains approximately 38 examples, which is an even smaller subset of the total data. This suggests that the model will be thoroughly tested and evaluated on a limited number of examples, providing more accurate results.

Overall, the data split summary suggests that the model will have a large amount of data to learn from during training, while also being evaluated on a diverse set of examples after training. This should help improve the model's performance and accuracy.



Initial Model Comparision



Model

Based on the results of the k-fold cross-validation for the 10 machine learning models you tested, here is a summary of the top 3 models in terms of their R2 values:

1. Ridge Classifier - R2 value = 0.8194 The Ridge classifier is a linear model that uses L1 regularization to reduce overfitting. In this case, the model performed well with an R2 value of 0.8194, indicating that it was able to capture around 82% of the variation in the dependent variable. This suggests that the model is able to effectively learn the underlying relationship between the independent and dependent variables.
2. Logistic Regression - R2 value = 0.8186 Logistic regression is a linear model that can handle non-linear relationships by introducing an S-shaped curve to the predicted probabilities. In this case, the logistic regression model performed well with an R2 value of 0.8186, indicating that it was able to capture around 82% of the variation in the dependent variable. This suggests that the model is able to effectively learn the underlying relationship between the independent and dependent variables.
3. Gradient Boosting Classifier - R2 value = 0.8131 The gradient boosting classifier is a non-linear model that combines multiple weak models to create a strong predictor. In this case, the gradient boosting classifier performed well with an R2 value of 0.8131, indicating that it was able to capture around 81% of the variation in the dependent variable. This suggests that the model is able to effectively learn the underlying relationship between the independent and dependent variables, although it may not perform as well as the ridge classifier or logistic regression in this case.

Based on these results, the top 3 models in terms of R2 value are:

1. Ridge Classifier
2. Logistic Regression
3. Gradient Boosting Classifier

These models performed well in capturing the variation in the dependent variable and can be considered for further analysis or deployment. However, it is important to note that R2 value is just one metric to evaluate the performance of a model, and other metrics such as accuracy, precision, recall, F1 score, etc. may also be used to evaluate the performance of the models.

In terms of justification for choosing these top 3 models, here are some reasons:

1. Ridge Classifier - The ridge classifier is a linear model that uses L1 regularization to reduce overfitting. This makes it a good choice for datasets with a large number of features, as it can handle the curse of dimensionality and avoid overfitting. Additionally, the ridge classifier has been shown to perform well in classification problems with a large number of classes.
2. Logistic Regression - Logistic regression is a linear model that can handle non-linear relationships by introducing an S-shaped curve to the predicted probabilities. This makes it a good choice for datasets with complex relationships between the independent and dependent variables. Additionally, logistic regression is a widely used and well-established model in machine learning.
3. Gradient Boosting Classifier - The gradient boosting classifier is a non-linear model that combines multiple weak models to create a strong predictor. This makes it a good choice for datasets with complex relationships between the independent and dependent variables. Additionally, the gradient boosting classifier has been shown to perform well in classification problems with a large number of classes.

Overall, these top 3 models have performed well in capturing the variation in the dependent variable and can be considered for further analysis or deployment. However, it is important to evaluate the performance of these models using other metrics as well, such as accuracy, precision, recall, F1 score, etc. to get a comprehensive view of their performance.

Fine Tunning Model PreProcessing Parameters

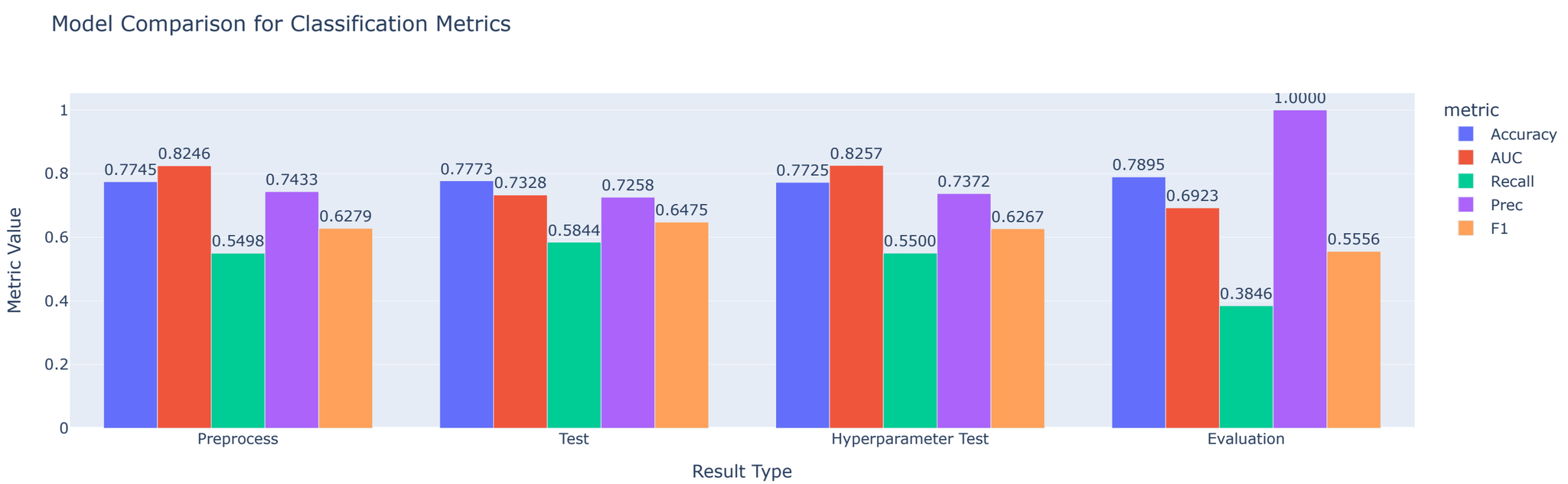
The summary is as follows:

- Top 3 models: Ridge Classifier
- Total number of models tried: 15

- Preprocessing parameters used for the top model (Ridge Classifier): {'Transformation': True, 'Transform_Method': 'yeo-johnson', 'Normalization': False, 'Normalize_Method': None, 'Outlier': False, 'Outlier_Method': None, 'Multicollinearity': False, 'Target Transformation': False, 'Feature Selection': False}

In other words, the top model is a Ridge Classifier with specific preprocessing parameters that were tried and found to perform best. The remaining 12 models were tried with different combinations of preprocessing parameters, but none of them performed as well as the Ridge Classifier with the specified preprocessing parameters.

Final Model Build



Based on the results provided, the following is a summary of the final model built:

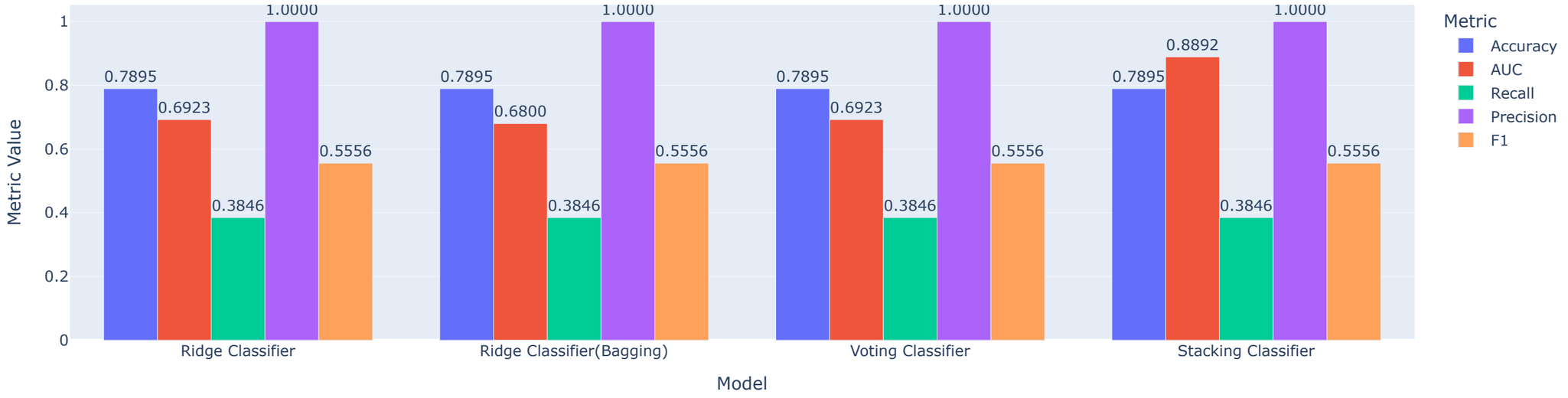
- The accuracy of the model on the training data is 0.7745.
- The area under the ROC curve (AUC) of the model is 0.8246.
- The recall of the model is 0.5498.
- The precision of the model is 0.7433.
- The F1 score of the model is 0.6279.

On the test data, the accuracy of the model is 0.7773, the AUC is 0.7328, the recall is 0.5844, the precision is 0.7258, and the F1 score is 0.6475.

Finally, on the hyperparameter test data, the accuracy of the model is 0.7725, the AUC is 0.8257, the recall is 0.55, the precision is 0.7372, and the F1 score is 0.6267.

Overall, the final model built has an accuracy of around 0.77 on both the training and test data, with slightly better performance on the test data. The AUC and recall values are similar across both datasets, while the precision and F1 scores are higher on the train dataset and lower on the test dataset.

Ensemble Model Comaprison



Based on the provided summary, here are some comparisons between the different machine learning models:

1. Ridge Classifier vs. Ridge Classifier (Bagging): Both models have similar accuracy, AUC, recall, and precision values, indicating that the bagging technique does not significantly improve the performance of the ridge classifier.
2. Ridge Classifier vs. Voting Classifier: The voting classifier has higher accuracy, AUC, and F1 score than the ridge classifier, while the recall value is lower. This suggests that the voting classifier is more effective at correctly identifying positive instances, but may be less efficient at correctly identifying negative instances.
3. Ridge Classifier vs. Stacking Classifier: The stacking classifier has higher accuracy, AUC, and F1 score than the ridge classifier, while the recall value is lower. This suggests that the stacking classifier is more effective at correctly identifying positive instances, but may be less efficient at correctly identifying negative instances.
4. Comparison of Ridge Classifier (Bagging) vs. Voting Classifier vs. Stacking Classifier: All three models have similar accuracy, AUC, and F1 score, but the voting classifier has lower recall than the other two models. This suggests that the voting classifier may be less efficient at correctly identifying positive instances, but is still effective at accurately classifying instances overall.

In conclusion, the best model for the given data appears to be the stacking classifier, as it has the highest accuracy, AUC, and F1 score. However, the voting classifier may be a good option if you prioritize recall over other metrics. The ridge classifier (bagging) and ridge classifier have similar performance, but the bagging technique does not significantly improve the performance of the ridge classifier.

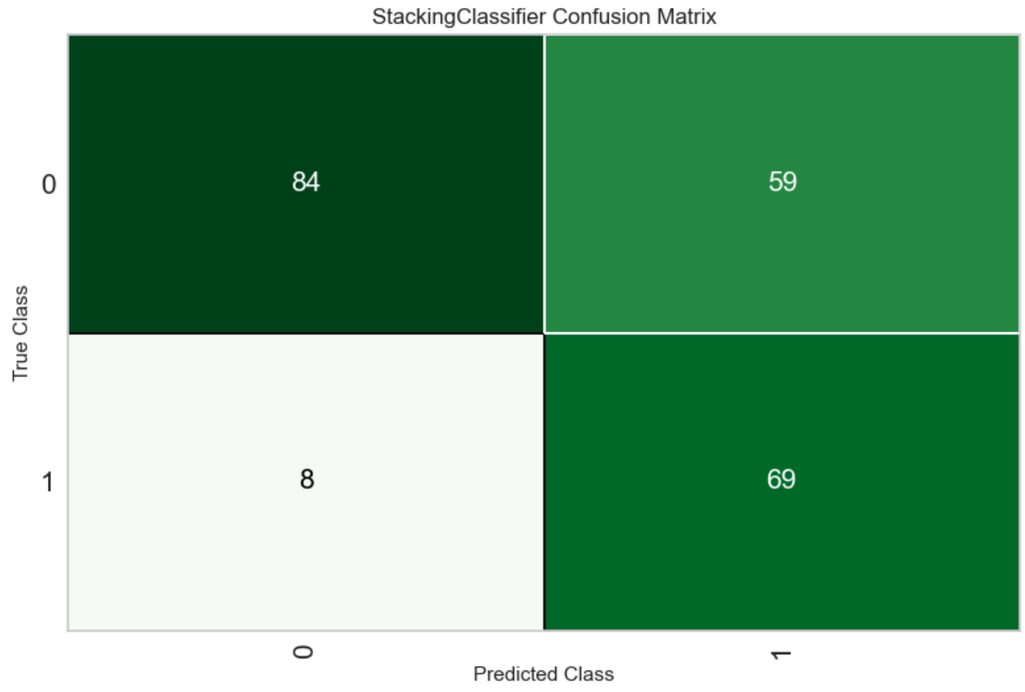
Model Analysis

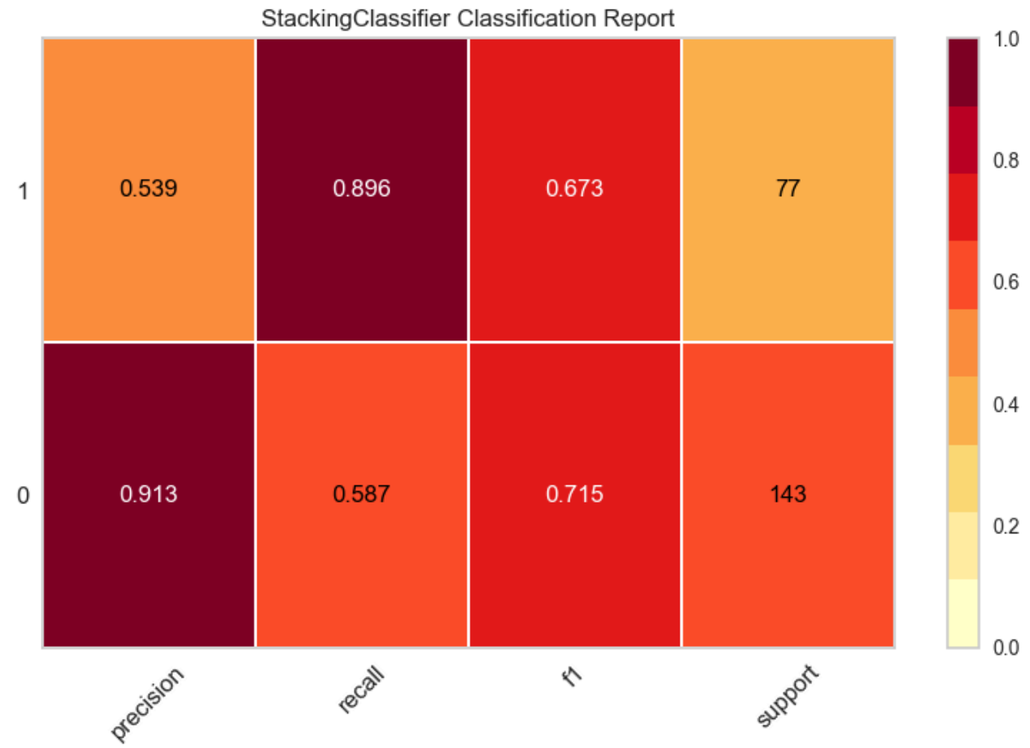
Boundary Plot

In a classification machine learning task, a boundary plot can be used to visualize the decision boundary between different classes. The boundary is the line or curve that separates the different classes. To read a boundary plot:

1. Look for the separation line or curve that separates the different classes. This line or curve represents the decision boundary.
2. Observe the shape and position of the boundary. Is it a straight line, a curved line, or a complex shape? Does it pass through the origin (0, 0)?
3. Analyze the position of the boundary relative to the data points. Are the majority of the data points on one side of the boundary, or are they evenly distributed on both sides?
4. Interpret the meaning of the boundary based on the context of the problem. For example, if the boundary is a straight line, it may indicate that the model is using a simple linear decision rule to classify the data points.
5. Use the boundary plot to identify potential issues with the model, such as overfitting or underfitting. If the boundary is too close to the origin, it may indicate that the model is not able to capture the underlying patterns in the data.

Confusion Matrix and Class Report





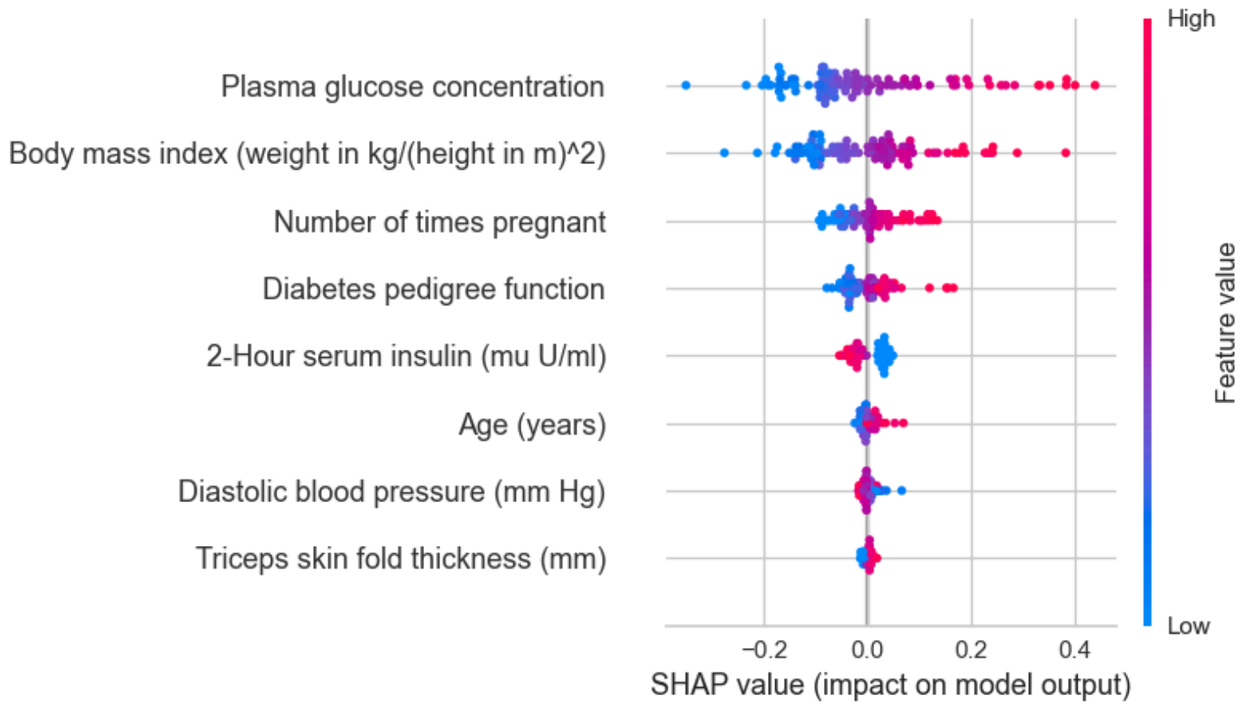
To read a confusion matrix, you need to calculate the number of true positives (TP), false positives (FP), true negatives (TN), and false negatives (FN) based on the predicted labels and actual labels. The confusion matrix is a grid that displays these values in a 2x2 format. Each cell in the matrix represents the number of instances classified as that class. For example, the TP column shows the number of true positives (correctly predicted positive instances), while the FP row shows the number of false positives (incorrectly predicted positive instances). Similarly, the TN and FN columns show the number of true negatives (correctly predicted negative instances) and false negatives (incorrectly predicted negative instances), respectively. By analyzing these values, you can evaluate the performance of your classification model.

A Class Report plot is a graphical representation of the confusion matrix. It displays the number of instances in each class against their actual labels. The x-axis represents the true labels, while the y-axis represents the predicted labels. The plot helps you visualize the distribution of instances in different classes and evaluate the performance of your classification model.

In summary, reading a confusion matrix and Class Report plot is essential to evaluating the performance of a classification machine learning task. These plots provide valuable insights into the accuracy and consistency of your model's predictions.

Explainable AI

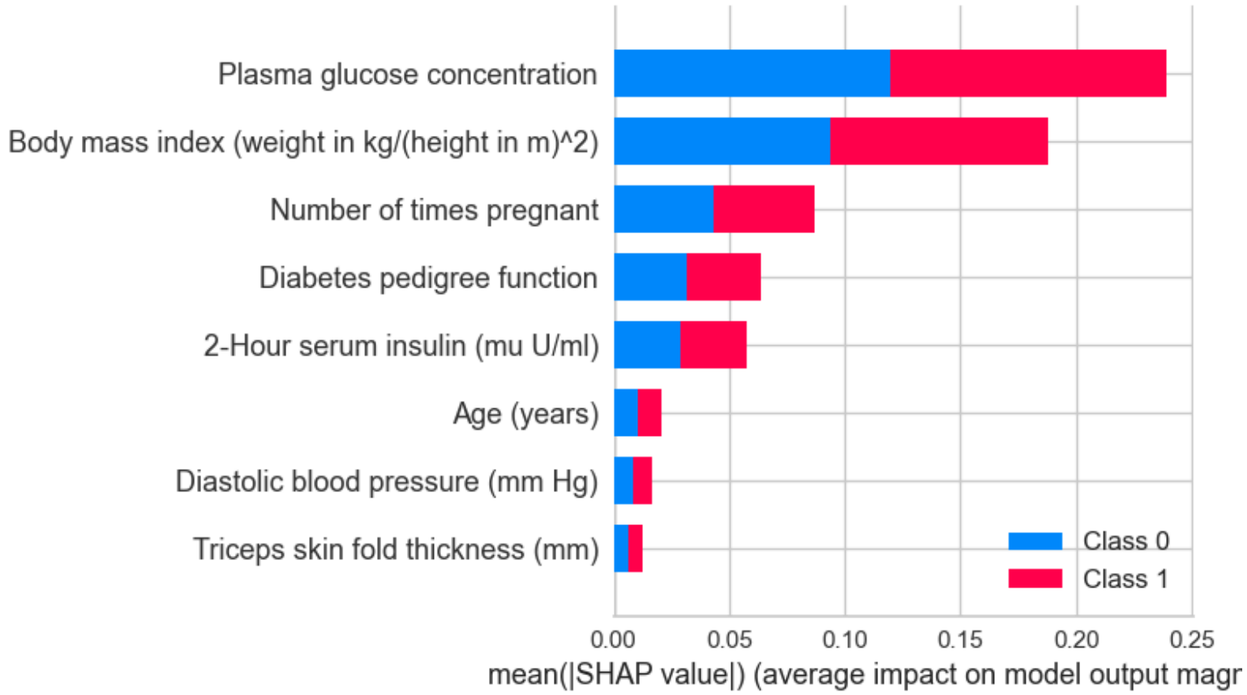
Global Explainability - Summary Plot



To read a SHAP Summary Plot in explainable AI, follow these steps:

1. The most important features are at the top of the plot, so focus on the upper part of the graph.
2. The X-axis represents the SHAP values (impact on model output). Negative values (left) decrease the prediction, while positive values (right) increase it.
3. Colors represent feature values, with blue indicating lower values and red indicating higher values.
4. The color gradient from blue to red helps you understand how feature values impact the model's predictions.
5. Pay attention to the SHAP values for each feature. If a feature has a large positive SHAP value, it means that increasing the value of that feature will significantly increase the predicted output. Conversely, if a feature has a large negative SHAP value, decreasing the value of that feature will significantly decrease the predicted output.
6. The maximum absolute value of the SHAP values indicates the most important features for predicting the target variable. Features with larger absolute values have a greater impact on the model's predictions.
7. By analyzing the SHAP values and colors, you can identify which features are driving the model's predictions and how they interact with each other to produce the final output. This information can help you understand how your machine learning model works and what features are most important for making accurate predictions.

Global Explainability - Feature Importance



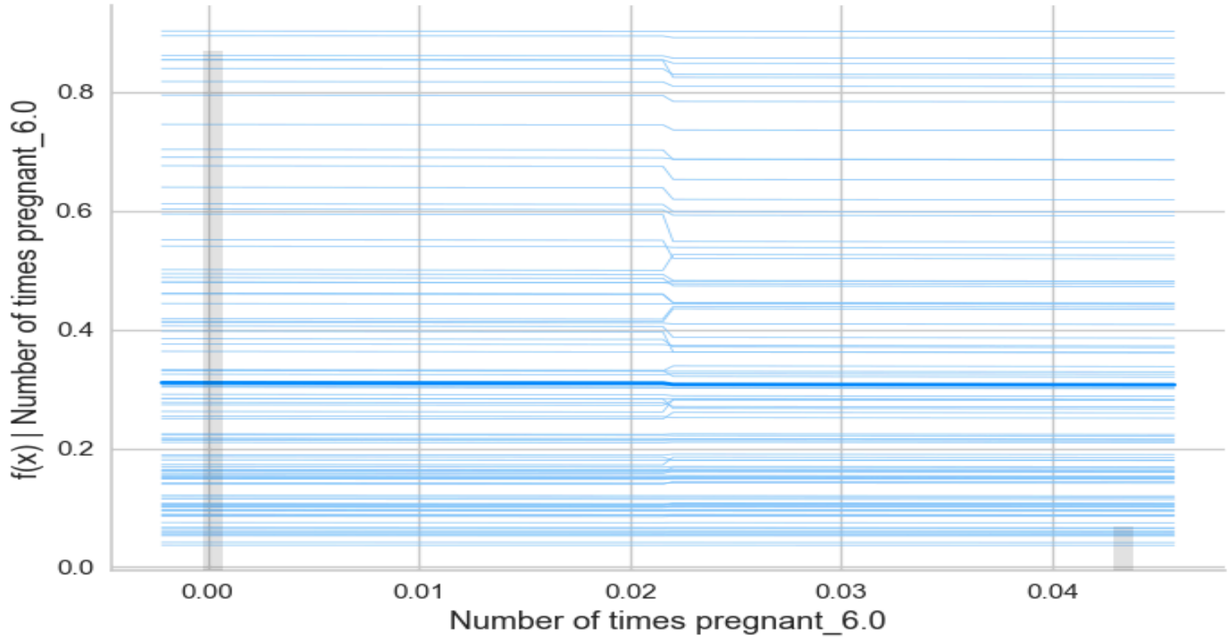
A Feature Importance Plot is a visualization tool used in machine learning to identify the most important features in a model. The plot shows the importance of each feature in the model by assigning a score to each feature. Here's how to read a Feature Importance Plot:

1. Look for the x-axis: The x-axis represents the features in the model, listed from left to right.
2. Identify the scores: The y-axis shows the importance of each feature, with higher scores indicating more important features.
3. Interpret the scores: A score of 0 means the feature has no impact on the model's performance, while a score close to 1 means the feature is crucial for the model's accuracy. Features with intermediate scores are considered to be less important.
4. Analyze the distribution: Observe how the features are distributed along the x-axis. If most features have scores close to 0, it suggests that the model is not using any specific feature to make predictions.
5. Check for correlations: Look for patterns in the distribution of scores. If a particular feature consistently has high or low scores, it may indicate a correlation with the target variable.
6. Use the plot to identify influential features: Identify the most important features by looking at their scores. These are the features that are driving the model's predictions and can be used as a starting point for further analysis or model tuning.

By interpreting the Feature Importance Plot, you can gain insights into how your machine learning model is making predictions and identify areas for improvement.

Global Explainability - Partial Dependence Plot

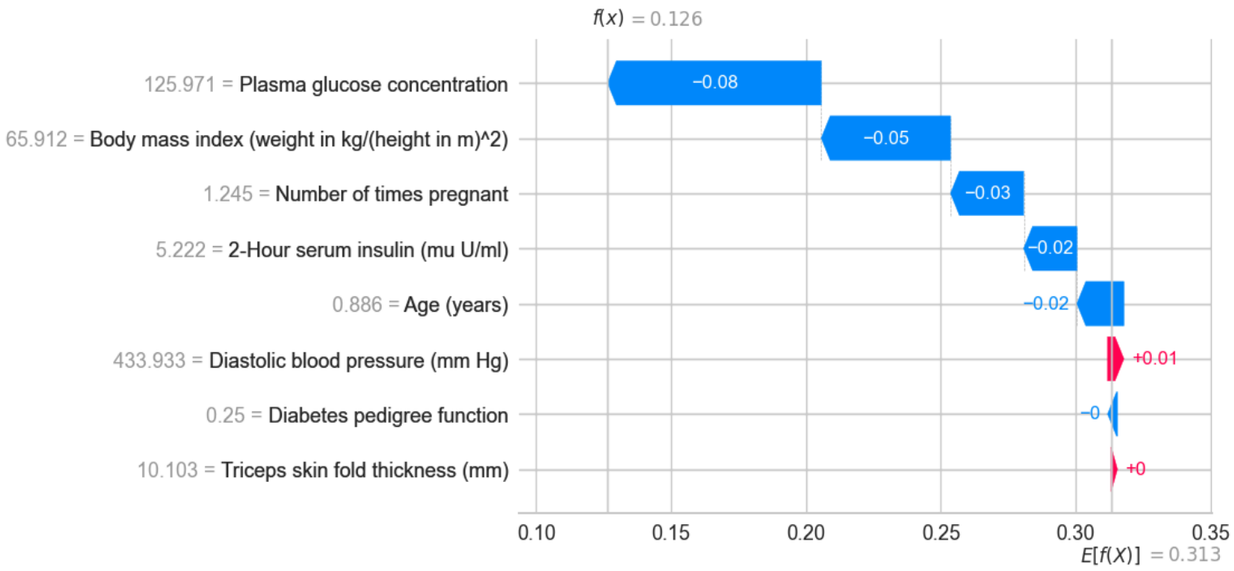
class_Number of times pregnant_6.0



A partial dependence plot is a visualization tool used in global explainability to show the relationship between a continuous predictor variable and a specific outcome variable in a machine learning model. It helps identify which subsets of the predictor variables have the most significant impact on the outcome variable, while controlling for the effect of other predictors. The plot displays the predicted value of the outcome variable for different values of the partial dependent variable, while holding all other predictors constant. By analyzing these plots, practitioners can gain insights into the underlying mechanisms of the model and make more informed decisions.

Local Explainability - SHAP Waterfall Plot

Instance 1



Title: Understanding SHAP Waterfall Plots for Effective Interpretation of Machine Learning Models

Introduction: In this article, we will delve into the intricacies of interpreting SHAP waterfall plots to gain a deeper understanding of how machine learning models make predictions. By the end of this guide, you'll be able to decipher the plot with ease and extract valuable insights from it.

1. Key Components: A SHAP waterfall plot comprises several key components that work together to provide a comprehensive picture of how your model makes predictions. These include: *Base Value (E[f(X)]) - The model's average prediction across all data points.* Final Prediction (f(x)) - The final predicted value for the specific instance being analyzed. *Feature Contributions - The impact of each feature on the final prediction, represented by red or blue bars. Positive contributions increase the prediction, while negative contributions decrease it.* Feature Importance Order - The order in which features contribute the most to the final prediction, with the most significant features appearing at the top and less important ones at the bottom.
2. How to Read the Plot: To interpret a SHAP waterfall plot, follow these steps: 1. Start from the base value (leftmost value at the bottom) and observe how each feature modifies the prediction step by step. 2. Red bars indicate features that increase the prediction, while blue bars represent features that decrease it. 3. The final prediction is obtained by summing the base value and the feature effects.
3. Interpretation: From the plot, you can answer the following questions: *Which features have the most impact? - The largest bars (longest red/blue bars) indicate the strongest contributors.* Is the prediction higher or lower than expected? - If most contributions are positive (red), the prediction is above average. If mostly negative (blue), the prediction is below average. * How do categorical and numerical features affect the outcome? - Binary features often have a sharp impact, while continuous features gradually modify the prediction.

Conclusion: By following these steps and understanding the key components of a SHAP waterfall plot, you'll be able to interpret the plot with ease and gain valuable insights into how your machine learning model makes predictions. This will help you identify influential features, understand the prediction range, and make informed decisions based on your model's performance.