Introduction

The project aims to classify the quality of wine as 'good' or 'bad' using the K-Nearest Neighbors (KNN) algorithm. This report details the creation of the KNN model, data preprocessing, feature selection, model training, prediction, and evaluation.

Part A: Model Code

Functions Implemented:

- Euclidean Distance: Function developed to compute the distance between pairs of samples.
- Manhattan Distance: Another function was written to calculate a different norm of distance, useful for varied geometries in the data.
- Accuracy and Generalization Error: Developed to gauge model performance and its ability to generalize to unseen data.
- Precision, Recall, and F1 Score: Separate functions created for these metrics provide a nuanced view of the model's performance, especially in imbalanced datasets.
- Confusion Matrix: Enables understanding of the true positive, false positive, false negative, and true negative predictions.
- ROC Curve: A function to plot sensitivity vs. 1-specificity, providing insight into the model's threshold tuning.
- AUC for ROC: Computes the area under the ROC curve, summarizing the model's performance across thresholds.
- Precision-Recall Curve: Offers a view of the trade-off between precision and recall for different thresholds.
- KNN_Classifier Class: The class encompasses fit, predict, and additional methods to implement the KNN algorithm, considering neighbor counts, distance weighting, and other parameters.

Part B: Data Processing

The dataset was loaded, and the target variable 'quality' was transformed into a binary classification problem by thresholding. Descriptive statistics provided an overview of the features, while data shuffling ensured randomness before partitioning.

Redundancy and Feature Selection:

Pair plots indicated no extreme redundancy, thus all features were initially retained.

Partitioning:

The partition function split the data into training and testing sets, adhering to a specified ratio.

Part C: Model Evaluation

Initial Naive Performance:
Running the KNN with n_neighbors=5 yielded perfect accuracy (1.0) but an F1 score of 0, indicating a potential class imbalance or predictive deficiency in capturing the positive class.

Impact of Standardization:
Post-standardization, the model maintained the same accuracy, suggesting no immediate benefit from scaling. However, since the F1 score remained at 0, further investigation into standardization's effect on the minority class prediction is recommended.

Distance Weighting Evaluation:
Testing with inverse distance weighting did not yield a different outcome, suggesting either an implementation error or a data-specific issue.

Model Performance and Configurations

The table evaluating k, distance metrics, and weights across all combinations displayed consistent results: a high accuracy of 1.0 and an F1 score of 0, raising concerns about the model's efficacy in capturing the minority class.

Analysis and Recommendations

Model Analysis: The discrepancy between accuracy and F1 score is troubling and indicates potential issues such as data imbalance, model misconfiguration, or misinterpretation of the results.
Standardization Reassessment: Despite no immediate observed benefits, standardization typically benefits distance-based algorithms and should be reconsidered after resolving the F1 score issue.
Review of Implementation: Given the uniform results across multiple configurations, there is a likelihood of an error in the implementation of the model or the calculation of the F1 score.
Further Investigation: A deeper analysis of the data distribution, especially the balance between classes, should be performed. Additionally, cross-validation techniques might provide a more robust evaluation framework.
Advanced Diagnostics: The usage of additional metrics and visualizations such as the ROC curve and precision-recall curve could provide more insight into the model's performance nuances.

Conclusion

The consistent perfect accuracy and zero F1 score across various model configurations suggest a potential flaw in the model implementation or evaluation methods. The findings underscore the necessity to review the model's code, reconsider feature scaling, and delve into a more granular

analysis of the performance metrics. Future work should focus on identifying and addressing the root cause of the observed discrepancies to enhance the model's predictive capabilities.