Machine Learning Approach to Assessing Assumptions of Normality

Oklahoma State University

Benedict Kongyir

February 4, 2025

Introduction

The application of machine learning (ML) to statistical challenges has emerged as a transformative paradigm, offering solutions that often surpass classical statistical methods in accuracy and adaptability (Simić, 2021). Among these challenges, the assessment of normality–a cornerstone assumption for parametric statistical tests–remains critical. Violations of normality can undermine the validity of inferential procedures such as t-tests and ANOVA, which assume that data are drawn from Gaussian populations (Wilson and Engel, 1990).

Traditional methods for testing normality fall into two categories: *informal* graphical techniques (e.g., Q-Q plots) and *formal* hypothesis tests (e.g., Shapiro-Wilk). While intuitive, graphical methods lack objectivity, whereas formal tests suffer from sample-size sensitivity. For instance, small samples reduce the power to detect non-normality, while large samples may flag trivial deviations (Razali and Wah, 2011). This limitation has spurred efforts to develop robust, sample-agnostic alternatives.

A pivotal shift occurred in the 1990s when Wilson and Engel (1990) pioneered the use of artificial neural networks (ANNs) for normality testing. Their approach mapped samples to feature vectors comprising skewness, kurtosis, Shapiro-Wilk statistics, and entropy measures, achieving performance comparable to established tests like Lilliefors. However, their ANN exhibited reduced sensitivity to true normal samples and was restricted to fixed sample sizes (e.g., $n = 30$), limiting generalizability. Subsequent work by Sigut et al. (2006) expanded the feature set to include Fisher-transformed correlation coefficients and tested models across varying sample sizes ($n = 10$–300). While their ANN outperformed classical tests, its efficacy diminished when applied to sample sizes outside the training range.

Recent advances by Simić (2021) introduced a descriptor-based neural network (DBNN) incorporating quartiles, standard deviation, and extremal values alongside traditional statistics. The DBNN demonstrated robust type I error control and competitive power relative to Shapiro-Wilk and Anderson-Darling tests for $n = 10$–100. However, it struggled with symmetric, short-tailed distributions, highlighting lingering gaps in ML-driven normality assessment.

This evolving landscape underscores the need for a unified framework that integrates diverse statistical descriptors with modern ML algorithms to improve generalizability across sample sizes and distributional alternatives.

Proposed Method

We frame normality testing as a **binary classification task**, where samples belong to one of two classes:

- C_N: Drawn from a normal distribution
- $C_{N'}$: Drawn from a non-normal distribution

Feature Engineering

To capture multi-faceted distributional characteristics, each sample is represented by a feature vector including:

- **Shape descriptors**: Skewness, kurtosis
- **Goodness-of-fit statistics**: Shapiro-Wilk, Anderson-Darling, Jarque-Bera
- **Nonparametric metrics**: Lilliefors, Cramér-von Mises
- **Dispersion measures**: Coefficient of variation (CV), range, Gini coefficient
- **Ancillary features**: Sample size, energy (sum of squared values), outlier count

Normalization of Numeric Features

To maintain the relative importance of each predictor and ensure more balanced and effective results, we used the minimax normalization technique to rescale numeric predictors, preventing any single feature from dominating due to its scale.

Model Architecture

Six ML classifiers are trained to discriminate C_N from $C_{N'}$:

- Logistic Regression (baseline interpretability)
- Random Forest (ensemble decision trees)
- Artificial Neural Network (feedforward architecture)
- Gradient Boosting Machines (sequential tree boosting)
- Support Vector Machines (kernel-based separation)
- K-Nearest Neighbors (instance-based learning)

Variable Importance

Variable importance analysis is integral to ML workflows, serving two key purposes:

- **Feature Selection**: Identifying the most influential predictors to improve model accuracy while reducing complexity.
- **Interpretability**: Revealing key patterns that differentiate normal from abnormal samples, making model decisions more transparent and aligning them with domain expertise.

We use the mean decrease in the Gini impurity which measures how frequently a feature splits nodes to improve class separation to calculate the variable importance from the Random Forest model.

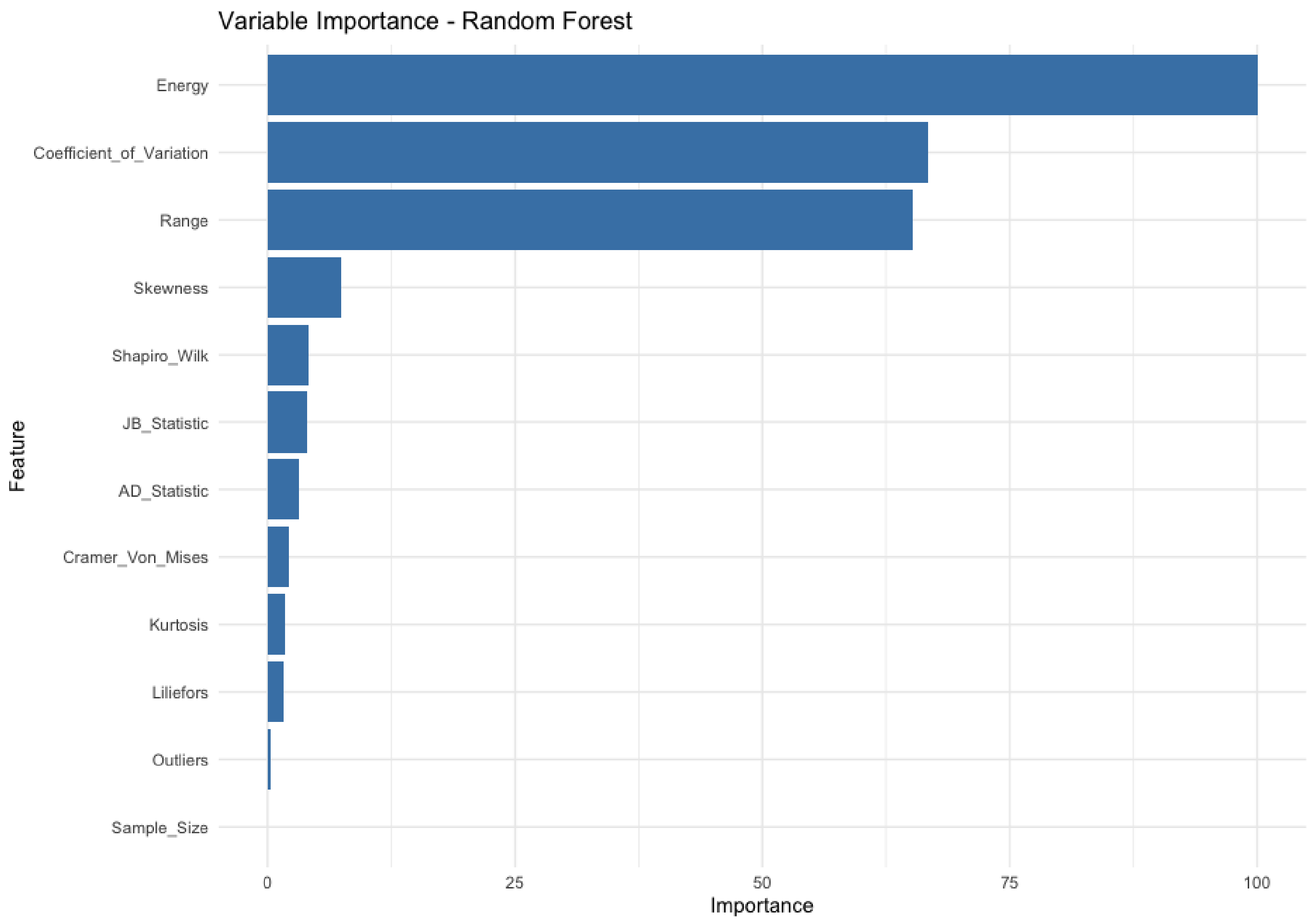

Figure 1: *Showing variable importance of the various features.*

Receiver Operating Characteristics(ROC) Curve

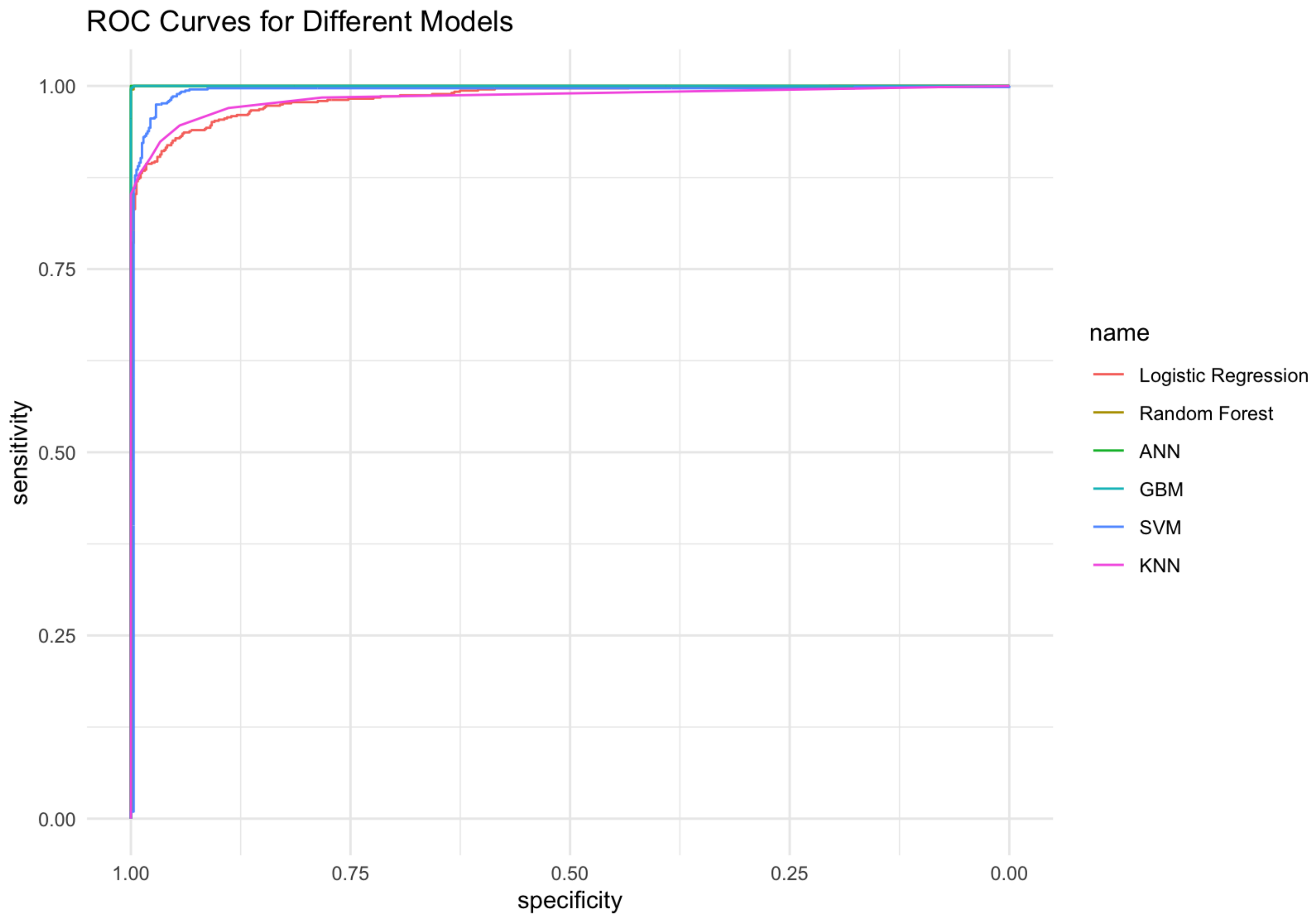

Figure 2: *Showing Receiver Operating Characteristics(ROC) Curve for each model.*

0.1 Model Comparisons

Model	Accuracy(%)	Sensitivity(%)	Specificity(%)	AUC(%)
Logistic Regression	93.81	91.59	96.03	98.40
Random Forest	99.69	99.37	100	100
Artificial Neural Network	99.84	99.68	100	100
Gradient Boosting Trees	99.84	99.68	100	100
Support Vector Machines	94.52	90.32	98.73	99.15
K-Nearest Neighbor	93.97	90.16	97.78	98.39

Table 1: Comparing Models Using Accuracy Rates, Precision, Sensitivity, Specificity and AUC

Appendix

Algorithms

The following lays out the concept and processes for generating sample data, preparing the data, and building, and testing the various models to validate them.

Algorithm for ML Models for Normality

Algorithm 1 Data Generation Algorithm

```
Require: Sample size $n$, number of samples $N$, distribution type dist, label label
Ensure: Dataset with statistical features and labels
 1: for $i = 1$ to $N$ do
 2:     Generate $n$ samples from the specified distribution dist
 3:     Calculate statistical features:
 4:         Skewness, kurtosis, Jarque-Bera statistic, Anderson-Darling statistic
 5:         Zero-crossing rate, Gini coefficient, number of outliers
 6:         Shapiro-Wilk statistic, Liliefors statistic, Cramer-von Mises statistic
 7:         Sample size, range, coefficient of variation, and energy
 8:     Assign the label label to the data
 9: end for
10: Combine all generated data into a single dataset
11: return Dataset containing statistical features and labels
```

2. Data Preparation for Machine Learning Models

Algorithm 2 Data Preparation Algorithm

```
Require: Dataset
Ensure: Training and testing datasets with normalized features
1: Shuffle the dataset to randomize sample order
2: Split the data into training (70%) and testing (30%) sets
3: for each numeric feature in the dataset do
4:     Apply min-max normalization: $x' = \frac{x-\min(x)}{\max(x)-\min(x)}$
5: end for
6: Ensure the Label column is treated as a categorical variable
7: return Training and testing datasets
```

3. Model Building and Testing Step

```
Algorithm 3 Model Building and Testing Algorithm
Require: Training and testing datasets
Ensure: Trained models and performance metrics
 1: Define models: Logistic Regression, Random Forest, ANN, GBM, SVM, KNN
 2: for each model do
 3:     Train the model using the training dataset
 4:     Predict labels for the test dataset
 5:     Evaluate the model using a confusion matrix
 6: end for
 7: Calculate performance metrics: accuracy, precision, recall, F1-score
 8: return Trained models, predictions, and performance metrics
```

4. Validation Step

```
Algorithm 4 Validation Algorithm
Require: Validation dataset, trained models
Ensure: Performance metrics on the validation dataset
 1: Generate validation dataset with distributions not used in training
 2: Calculate statistical features as in the data generation step
 3: Shuffle the validation dataset
 4: Normalize the validation dataset using min-max normalization
 5: for each trained model do
 6:     Predict labels for the validation dataset
 7:     Evaluate predictions using a confusion matrix
 8: end for
 9: Calculate performance metrics: accuracy, precision, recall, F1-score
10: return Performance metrics on the validation dataset
```

5. Plotting and Visualization

```
Algorithm 5 Plotting and Visualization Algorithm
Require: Trained models, test data
Ensure: ROC curves and variable importance plots
 1: Plot ROC curves for all models:
 2:     Compute true positive rate (TPR) and false positive rate (FPR)
 3:     Plot the ROC curve and compute AUC
 4: For Random Forest model:
 5:     Calculate variable importance scores
 6:     Plot variable importance to visualize feature contributions
 7: return ROC curves and variable importance plots
```

References

Nik AK Razali and Yap Bee Wah. Power comparisons of shapiro-wilk, kolmogorov-smirnov, lilliefors and anderson-darling tests. *Journal of Statistical Modeling and Analytics*, 2(1): 21–33, 2011.

J Sigut, J Piñeiro, J Estévez, and P Toledo. A neural network approach to normality testing. *Intelligent Data Analysis*, 10(6):509–519, 2006.

Miloš Simić. Testing for normality with neural networks. *Neural Computing and Applications*, 33(23):16279–16313, 2021.

Paul R Wilson and Alejandro B Engel. Testing for normality using neural networks. In *[1990] Proceedings. First International Symposium on Uncertainty Modeling and Analysis*, pages 700–704. IEEE, 1990.