**Letter Recognition using Machine Learning Models**

**Dataset Overview**

The dataset used in this project is the **Letter Recognition Dataset**, which consists of **20,000 instances** and **16 numerical features** derived from black-and-white character images. The goal is to classify each image as one of the **26 capital letters** of the English alphabet. These images were generated using **20 different fonts**, with each letter randomly distorted to create **unique variations**. The numerical features include **statistical moments and edge counts**, with values scaled between **0 and 15**.

The dataset has no missing values and is structured as follows:

* **Target Variable**: letter (categorical, 26 values from A to Z)
* **Features**: 16 integer values representing different geometric and statistical properties of the character images
* **Training/Test Split**: Typically, **16,000 instances** are used for training and **4,000 instances** for testing

A graph of blue bars

Description automatically generated

**Objective**

The primary goal of this project is to **compare multiple classification models** to determine which one performs best in recognizing handwritten capital letters. The models evaluated include:

1. **K-Nearest Neighbors (KNN)**
2. **Decision Tree Classifier**
3. **Random Forest Classifier**
4. **Naive Bayes Classifier**

**Methodology**

**1. Data Preprocessing**

* The dataset was loaded into a Pandas DataFrame.
* The letter column was set as the target variable (y), while the remaining columns formed the feature matrix (X).
* The data was split into **80% training** and **20% testing** sets using train\_test\_split.
* Features were standardized using StandardScaler to improve model performance.

**2. Results Visualization**

* **Feature importance graphs** were generated for each model using permutation importance.
* **Confusion matrices** were plotted to analyze model misclassification patterns.
* **A bar chart comparing accuracy scores** of all models was created to identify the best performer.

**3. Model Training & Evaluation**

Each classifier was trained using the training data and evaluated using the test set. The performance was assessed using **accuracy scores** and **confusion matrices**.

**K-Nearest Neighbors (KNN)**

* **Number of Neighbors**: 5
* **Accuracy Score**: Computed on test data
* **Feature Importance**: Assessed using permutation importance
* **Confusion Matrix**: Generated to visualize classification errors

A graph of a bar graph

Description automatically generated with medium confidence

A graph of numbers and symbols

Description automatically generated with medium confidence

**Decision Tree Classifier**

* **Model trained on entire dataset**
* **Accuracy Score**: Computed on test data
* **Feature Importance**: Evaluated
* **Confusion Matrix**: Generated

A graph with blue and white bars

Description automatically generated

A grid of numbers

Description automatically generated with medium confidence

**Random Forest Classifier**

* **Number of Estimators**: 100
* **Random State**: 42 (for reproducibility)
* **Accuracy Score**: Computed
* **Feature Importance**: Evaluated
* **Confusion Matrix**: Generated

A graph with blue and white bars

Description automatically generated

A graph of numbers and symbols

Description automatically generated with medium confidence

**Naive Bayes Classifier**

* **Gaussian Naive Bayes model** was applied
* **Accuracy Score**: Computed
* **Feature Importance**: Evaluated
* **Confusion Matrix**: Generated

A graph with blue bars

Description automatically generated with medium confidenceA blue and white grid with black numbers

Description automatically generated with medium confidence

A graph showing different colored squares

Description automatically generated

**Findings & Conclusion**

* **Random Forest achieved the highest accuracy**, followed by **KNN, Decision Tree, and Naive Bayes**.
* **Feature importance analysis** revealed that certain statistical properties contributed more to classification accuracy.
* **Confusion matrices** provided insights into which letters were misclassified most often and which were easily distinguishable.
* **Further improvements** could include tuning hyperparameters, using deep learning models, or experimenting with additional feature engineering techniques.

This project successfully demonstrated the **application of multiple machine learning models** in **letter recognition** and provided insights into their relative effectiveness for classification tasks.