Machine Learning Model Analysis Report

1. Introduction

This report provides a comprehensive overview of the data analysis and model development process, including exploratory data analysis (EDA), data preprocessing, model training, evaluation, and recommendations. The primary goal is to enhance the predictive performance of the models and achieve an R^2 score of at least 0.8.

The analysis follows a structured approach:

1. **Exploratory Data Analysis (EDA)** to understand the dataset.
2. **Data Cleaning and Preprocessing** to prepare the dataset for modeling.
3. **Dimensionality Reduction using PCA** to improve efficiency.
4. **Model Training & Evaluation** using various machine learning algorithms.
5. **Findings & Recommendations** for further improvements.
6. **Appendix** containing the complete code.

2. Exploratory Data Analysis (EDA)

Key Findings:

- **Data Inspection:** The dataset was loaded, and initial observations were made regarding its structure, data types, and missing values.
- **Statistical Summary:** Measures such as mean, median, standard deviation, and quartiles were analyzed to understand feature distributions.
- **Missing Values:** Several features had missing values, requiring imputation techniques.
- **Outliers:** Box plots and histograms revealed the presence of outliers in numerical features.
- **Correlation Analysis:** A heatmap was used to analyze feature correlations, identifying redundant features.
- **Feature Distributions:** Histograms and density plots were used to check the normality of numerical features.

3. Data Cleaning and Preprocessing

Steps Taken:

1. **Handling Missing Values:**
    - Mean/median imputation for numerical variables.
    - Mode imputation for categorical variables.

    - Dropping columns with excessive missing values.
2. **Outlier Treatment:**
    - Winsorization and transformation techniques were applied to reduce outlier effects.
    - Log transformations were used for highly skewed distributions.
3. **Feature Scaling:**
    - Standardization (z-score normalization) for numerical features.
    - Min-Max scaling where appropriate.
4. **Categorical Encoding:**
    - One-hot encoding and label encoding were applied to categorical features.
5. **Dimensionality Reduction with PCA:**
    - PCA was performed to reduce feature dimensions while retaining most variance.
    - The optimal number of components was selected based on explained variance.

4. Model Training & Evaluation

Models Trained:

We trained and evaluated four machine learning models to predict the target variable.

1. **Random Forest:**
    - An ensemble learning method that combines multiple decision trees.
    - Hyperparameters tuned: number of estimators, max depth, and minimum samples split.
2. **Support Vector Machine (SVM):**
    - A powerful model for classification and regression.
    - Used RBF kernel with optimized hyperparameters.
3. **Naïve Bayes:**
    - A probabilistic classifier based on Bayes' theorem.
    - Worked best with categorical features after transformation.
4. **K-Nearest Neighbors (KNN):**
    - A distance-based classification model.
    - Optimal K value selected using cross-validation.

Performance Evaluation:

- **Metrics Used:**

    - R^2 Score (primary evaluation metric)
    - Mean Absolute Error (MAE)
    - Mean Squared Error (MSE)
    - Precision, Recall, and F1-score for classification
- **Hyperparameter tuning:** GridSearchCV was used to find the best hyperparameters.
- **Best Model:** Random Forest achieved the highest R^2 score, nearing the target of 0.8.

5. Findings & Recommendations

Observations:

- **Random Forest performed best** with optimized hyperparameters, achieving an R^2 close to 0.8.
- **PCA improved computation efficiency** but led to slight loss of predictive accuracy.
- **Feature scaling had a significant impact** on models like SVM and KNN.
- **Hyperparameter tuning significantly improved model performance.**

Recommendations:

1. **Further Hyperparameter Tuning:**
    - Conduct more iterations using Bayesian Optimization or Randomized Search.
2. **Ensemble Learning:**
    - Combine multiple models (e.g., Random Forest + SVM) to improve generalization.
3. **Feature Engineering:**
    - Create new meaningful features from existing ones.
    - Use domain knowledge to refine feature selection.
4. **Model Deployment:**
    - Deploy the best-performing model using FastAPI.
    - Integrate a monitoring system to track real-world performance.