**Loan Prediction Model Documentation**

**Overview**

This Python script builds and evaluates several machine learning models to predict loan default using the Lending Club loan dataset. The goal is to classify whether a loan is likely to be a bad loan (bad\_loan = 1) or not (bad\_loan = 0) based on various borrower and loan-related features.

**Libraries Used**

* numpy: For numerical computations.
* pandas: For data manipulation and analysis.
* sklearn: For machine learning models, preprocessing, and evaluation metrics.
* seaborn and matplotlib.pyplot: For data visualization.
* xgboost: For the XGBoost classifier.

**Dataset**

The dataset (lending\_club\_loan\_dataset.csv) contains 20,000 entries and 15 columns, including features like grade, annual\_inc, dti, purpose, term, and bad\_loan indicating loan status.

**Data Preprocessing**

1. **Handling Missing Values**:
   * Imputed missing values in home\_ownership using the mode.
   * Imputed missing values in dti using the mean.
   * Removed last\_major\_derog\_none column due to high null values.
2. **Feature Engineering**:
   * Converted term values to lowercase for consistency.
   * Applied one-hot encoding to categorical variables (grade, home\_ownership, purpose, term).
3. **Feature Scaling**:
   * Standardized numerical features (annual\_inc, emp\_length\_num, dti, revol\_util, total\_rec\_late\_fee, od\_ratio) using StandardScaler.

**Model Building and Evaluation**

**Logistic Regression**

* **Parameters**: max\_iter=1000, solver='lbfgs', class\_weight='balanced'.
* **Results**:
  + Accuracy: 0.64
  + Precision, Recall, F1-score: Detailed for both classes (0 and 1).
  + Confusion Matrix: True positives, true negatives, false positives, false negatives.

**K-Nearest Neighbors (KNN)**

* **Parameters**: n\_neighbors=100.
* **Results**:
  + Accuracy: 0.80
  + Precision, Recall, F1-score: Detailed for both classes (0 and 1).
  + Confusion Matrix: True positives, true negatives, false positives, false negatives.

**Support Vector Machine (SVM)**

* **Parameters**: class\_weight='balanced', probability=True.
* **Results**:
  + Accuracy: 0.64
  + Precision, Recall, F1-score: Detailed for both classes (0 and 1).
  + Confusion Matrix: True positives, true negatives, false positives, false negatives.

**Decision Tree Classifier**

* **Parameters**: max\_depth=4.
* **Results**:
  + Accuracy: 0.80
  + Precision, Recall, F1-score: Detailed for both classes (0 and 1).
  + Confusion Matrix: True positives, true negatives, false positives, false negatives.

**Random Forest Classifier**

* **Parameters**: n\_estimators=1000, max\_depth=3.
* **Results**:
  + Accuracy: 0.80
  + Precision, Recall, F1-score: Detailed for both classes (0 and 1).
  + Confusion Matrix: True positives, true negatives, false positives, false negatives.

**XGBoost Classifier**

* **Parameters**: n\_estimators=5000, max\_depth=4, learning\_rate=0.3.
* **Results**:
  + Accuracy: 0.75
  + Precision, Recall, F1-score: Detailed for both classes (0 and 1).
  + Confusion Matrix: True positives, true negatives, false positives, false negatives.

**Conclusion**

Among the models evaluated, K-Nearest Neighbors (KNN) achieved the highest accuracy of 0.80, while Logistic Regression and SVM both yielded an accuracy of 0.64. Further tuning and feature engineering could potentially improve the performance of the models.