Project Steps

This document outlines the necessary steps involved in a machine learning project, explaining the importance of each step, and compares the accuracies between different models. The steps include Data Loading, Preprocessing and Splitting, Model Definition and Setup, SMOTE and Pipeline Creation, Hyperparameter Tuning, Voting Classifier Evaluation, and Visualization of Model Performance.

**1. Data Loading:**

* **What it involves:** Importing necessary libraries and loading the dataset into the project environment. This step is crucial as it sets the foundation for all subsequent data handling and analysis.
* **Why it's used:** To ensure that all necessary tools and the dataset are available for data manipulation, visualization, and model building.

**2. Data Preprocessing and Splitting:**

* **What it involves:** Cleaning the data to remove or impute missing values, encoding categorical variables, normalizing or standardizing numerical values, and splitting the dataset into training and testing sets.
* **Why it's used:** Clean and well-prepared data improves model accuracy and performance. Splitting the data allows for the evaluation of the model on unseen data, giving a better estimate of real-world performance.

**3. Defining ML Models and Setup Pipelines:**

* **What it involves:** Setting up different machine learning models such as Logistic Regression, SVC, Random Forest, and Gradient Boosting within pipelines for standardized preprocessing steps.
* **Why it's used:** Pipelines simplify the process of coding and ensure that all steps in the data transformation and model training process are executed in the correct sequence. This also aids in model comparison and selection.

**4. SMOTE and Pipeline Creation:**

* **What it involves:** Addressing class imbalance in the dataset using SMOTE (Synthetic Minority Over-sampling Technique) and creating pipelines to evaluate model performance.
* **Why it's used:** To ensure that models do not become biased towards the majority class, improving their ability to predict minority class instances accurately.

**5. Hyperparameter Tuning:**

* **What it involves:** Utilizing GridSearchCV and RandomizedSearchCV for finding the optimal model parameters for models like Random Forest, Logistic Regression, Gradient Boosting, and SVC.
* **Why it's used:** Optimal hyperparameters can significantly improve model performance by tuning it more closely to the specific dataset characteristics.

**6. Voting Classifier Evaluation:**

* **What it involves:** Combining different models into a voting classifier to make predictions based on the majority vote or average predicted probabilities of the models for improved prediction accuracy.
* **Why it's used:** This ensemble method leverages the strengths of individual models to achieve higher accuracy than any single model could on its own.

**7. Visualization of Model Performance:**

* **What it involves:** Using tools like confusion matrices to visualize and compare the performance of different models.
* **Why it's used:** Visualization aids in the intuitive understanding of model performance, highlighting strengths and weaknesses in predictions, such as the balance between sensitivity and specificity.

**Performance metrics:**

Below is the detailed performance metrics for several machine learning models, including their accuracy scores and classification reports. Here is a summary of how each model performed:

1. **Gradient Boosting:**
   * Best score: 0.7532
   * Accuracy: 0.76
   * Performance: Good balance between precision and recall across classes.
2. **SVC (Support Vector Classifier):**
   * Best score: 0.7494
   * Accuracy: 0.75
   * Performance: Similar precision and recall to Gradient Boosting but slightly lower in accuracy.
3. **Random Forest:**
   * Best score: 0.7505
   * Accuracy: 0.75
   * Performance: Comparable to SVC, with a slight variation in precision and recall across classes.
4. **Logistic Regression:**
   * Best score: 0.7473
   * Accuracy: 0.75
   * Performance: Consistent with other models, showing balanced precision and recall.
5. **Voting Classifier:**
   * Accuracy: 0.88
   * Performance: Significantly higher accuracy than individual models, indicating effective combination of model predictions to improve overall performance.

In summary, the Voting Classifier outperforms individual models significantly, with an accuracy of 0.88, showcasing the benefit of combining multiple models to achieve higher predictive accuracy. Among the individual models, Gradient Boosting leads with the highest accuracy (0.7532), followed closely by Random Forest (0.7505), SVC (0.7494), and Logistic Regression (0.7473). Each model shows a good balance of precision and recall, indicating their effectiveness in handling both classes of the dataset.

Github link: <https://github.com/RavadaKavya/ML-Based-Diabetes-Prediction-from-BRFSS-Health-Indicators>.