Breast Cancer Classification Using Random Forest: Code Explanation

Github Link:

https://github.com/ganeshvannam/Breast-Cancer-Classification-using-Random-Forest

Loading & Exploring the Dataset:

- **Loading the Data**: I used sklearn.datasets.load_breast_cancer() to load the breast cancer dataset, which contains 30 features representing tumor characteristics.
- **Feature and Labels**: I printed the feature names and target labels. The target labels represent the tumor classification as either **'malignant'** or **'benign'.**
- **X and y**: X contains the features, and y contains the target labels.

Splitting the Data:

- The dataset was split into training and testing sets using train_test_split(), with 80% allocated for training and 20% for testing. I applied **stratified sampling** using stratify=y to preserve the class distribution of malignant and benign samples across both sets**.** This ensures that both the training and testing sets are representative of the original dataset.

Cross-Validation and Hyperparameter Tuning:

- I used **StratifiedKFold** for 5-fold cross-validation to ensure balanced class distribution in each fold. Hyperparameter tuning was performed using **GridSearchCV**, which explores different combinations of parameters like:
    - n_estimators: Number of trees in the forest.
    - max_depth: Maximum depth of each tree to control overfitting.
    - min_samples_split: Minimum number of samples required to split a node.
- **GridSearchCV** selects the best combination of these hyperparameters based on accuracy, optimizing the model's performance.

Model Training:

- After obtaining the best hyperparameters from **GridSearchCV**, I trained the **Random Forest classifier** on the training data (X_train, y_train)**.** The model was fitted using the optimized parameters, ensuring that it learns from the data with the most effective settings. The trained model can then be evaluated on the test data to assess its generalization.

Evaluating the Model:

- I used the trained model to predict labels for the test dataset. The model’s performance was evaluated using several metrics:
    - **Accuracy**: Measures the proportion of correct predictions.
    - **Precision**: Evaluates the model’s ability to correctly identify malignant samples (true positives).
    - **Recall(Sensitivity)**: Measures the model’s ability to identify all actual malignant samples.
    - **F1-score**: Balances precision and recall for a more comprehensive evaluation.
    - **Specificity:** measures the model’s ability to correctly identify benign samples, focusing on how many benign cases were accurately predicted. It helps us understand how well the model avoids false positives
    - **Confusion Matrix**: Provides a detailed breakdown of true positives, false positives, true negatives, and false negatives.
- Finally, I printed the output of these metrics to assess the model's performance.