Breast Cancer Prediction Using Random Forest Classifier

Overview

This program builds and evaluates a machine-learning model to predict breast cancer diagnosis (malignant or benign) using the Random Forest Classifier. The program:

1. Prepares and visualizes the data.
2. Trains the model on a dataset.
3. Evaluates the model's performance.
4. Saves the trained model.
5. Demonstrates how to use the model to predict the diagnosis for new input data.

Dependencies

The following Python libraries are required to run this code:

- `pandas`: For data manipulation.
- `numpy`: For numerical computations.
- `matplotlib` and `seaborn`: For data visualization.
- `plotly.express`: For interactive visualizations.
- `scikit-learn`: For machine learning model training, evaluation, and preprocessing.
- `joblib`: For saving and loading the trained model.

Code Workflow

1. Load and Explore Dataset

- The dataset is loaded using `pd.read_csv('data.csv')`.
- Initial exploration is done using:
  - `df.head()`: Displays the first few rows.
  - `df.info()`: Displays column types and non-null counts.
  - `df.describe()`: Provides summary statistics.

2. Handle Missing Data

- Checks for missing values using `df.isnull().sum()`.
- Visualizes missing values using a heatmap (`sns.heatmap`).

3. Data Visualization

- The distribution of diagnosis (malignant or benign) is visualized using a count plot (`sns.countplot`).
- Correlation between features is displayed using an interactive heatmap (`plotly.express.imshow`).

- A scatter matrix visualizes relationships among specific features (`plotly.express.scatter_matrix`).

4. Data Preprocessing

- Converts the `diagnosis` column from categorical to numerical format (`M -> 1`, `B -> 0`).
- Drops unnecessary columns (`id`).
- Splits the dataset into features (`X`) and target (`y`).
- Normalizes features using `StandardScaler`.

5. Train-Test Split

- Splits the data into training (70%) and testing (30%) sets using `train_test_split`.

6. Model Training

- A `RandomForestClassifier` is initialized and trained using the training data (`rf_model.fit`).

7. Model Evaluation

- Predictions are made on the test data (`rf_model.predict`).
- Performance metrics are calculated:
  - **Accuracy**: `accuracy_score`
  - **Confusion Matrix**: Visualized using `sns.heatmap`.
  - **Classification Report**: Includes precision, recall, and F1-score.
- Feature importance is visualized using a bar plot (`plotly.express.bar`).
- Cross-validation accuracy is computed using `cross_val_score`.

8. Save the Trained Model

- The trained model is saved as a `.pkl` file using `joblib.dump`.

9. Predict New Data

- Demonstrates how to use the trained model for prediction:
  - Input data is reshaped and scaled using the same scaler used during training.
  - The trained model predicts whether the input represents malignant (1) or benign (0) cancer.

Input and Output

Inputs

- **Training Data**: CSV file (`data.csv`) with breast cancer features and diagnosis labels.
- **New Input Data**: A tuple containing 30 feature values.

Outputs

- **Model Metrics**:
  - Accuracy score.
  - Confusion matrix.

    - Classification report.
    - Cross-validation accuracy.
    - Feature importance plot.
- **Prediction**:
    - Prints `Malignant` if the model predicts 1.
    - Prints `Benign` if the model predicts 0.

Example Input for Prediction

```
input_data = (15.34, 14.26, 102.5, 704.4, 0.1073, 0.2135, 0.2077, 0.09756,
0.2521, 0.07032, 0.4388, 0.7096, 3.384, 44.91, 0.006789, 0.05328, 0.06446,
0.02252, 0.03672, 0.004394, 18.07, 19.08, 125.1, 980.9, 0.139, 0.5954, 0.6305,
0.2393, 0.4667, 0.09946)
```

Limitations

1. The model assumes that the input features for prediction are preprocessed (e.g., scaled) in the same way as the training data.
2. Any deviation in feature order or scaling might lead to incorrect predictions.

How to Use

1. Prepare the dataset as `data.csv` in the correct format.
2. Run the script to train the model and save it.
3. Use the saved model to make predictions on new data.

Conclusion

This program provides a robust pipeline to preprocess data, train a Random Forest Classifier, evaluate its performance, and use the trained model for breast cancer diagnosis.