Breast Cancer Classification – Machine Learning Workflow

Step 1: Problem Definition

Objective: Build a machine learning model to classify breast tumors as malignant or benign. This is a binary classification problem where the output label is 0 (malignant) or 1 (benign).

Step 2: Install & Import Dependencies

Required Python libraries:

- numpy
- pandas
- scikit-learn

Imports used:

```
from sklearn import datasets
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score
```

Step 3: Load Dataset

Dataset used: Breast Cancer dataset from scikit■learn.

```
breast_cancer_dataset = datasets.load_breast_cancer()
```

Step 4: Convert to DataFrame

Convert dataset to structured DataFrame:

```
data_frame = pd.DataFrame(breast_cancer_dataset.data,
columns=breast_cancer_dataset.feature_names)
data_frame['label'] = breast_cancer_dataset.target
```

Step 5: Exploratory Data Analysis (EDA)

Performed EDA steps:

- data_frame.head()
- data_frame.shape
- data_frame.info()
- data_frame.isnull().sum()
- data_frame.describe()
- data_frame['label'].value_counts()
- data_frame.groupby('label').mean()

Step 6: Feature/Target Split

X = data_frame.drop(columns='label', axis=1)
y = data_frame['label']

Step 7: Train■Test Split

Split for model evaluation:
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=2)

80% training | 20% testing | same split reproducible using random_state

Step 8: Model Selection

Model selected: Logistic Regression
model = LogisticRegression(max_iter=10000)

Step 9: Model Training

model.fit(X_train, y_train)
Model learns relationships between features and class labels.

Step 10: Model Evaluation

Training accuracy = accuracy_score(y_train, model.predict(X_train))
Testing accuracy = accuracy_score(y_test, model.predict(X_test))

Step 11: Predictive System

input_data = X_train.iloc[0].values.reshape(1,-1)
prediction = model.predict(input_data)
Mapping:
0 = Malignant
1 = Benign

Step 12: Future Improvements

• Apply StandardScaler
• Try SVM / Random Forest / Gradient Boosting
• Hyperparameter tuning: GridSearchCV / RandomizedSearchCV
• Save and export model for deployment

Model Pipeline Summary

1. Define objective
2. Load dataset
3. Build DataFrame
4. Perform EDA
5. Split into X and y
6. Train-test split
7. Train model
8. Evaluate
9. Predict sample
10. Improve and deploy

Authored By
SATYAM GAJJAR