Breast Cancer Detection Analysis

Aniakwa Nathan

August 1, 2025

Abstract

This document presents an analysis of the breast cancer dataset using machine learning techniques. The dataset, containing 569 entries and 31 features, was preprocessed, split into training, validation, and test sets, and used to train Logistic Regression, SVM, and Neural Network models. Hyperparameter tuning was performed, and the Neural Network model achieved perfect classification on the test set. Key findings and code implementations are detailed below.

1 Introduction

This analysis uses the breast cancer dataset available in scikit-learn to develop predictive models for cancer detection. The dataset is loaded and preprocessed, followed by model training and evaluation. The code and results are presented as extracted from a Jupyter notebook.

2 Dataset Loading and Preprocessing

The dataset is loaded using scikit-learn's `load_breast_cancer` function and converted into a pandas DataFrame. The features are scaled using `StandardScaler`.

```
import pandas as pd
from sklearn.datasets import load_breast_cancer

# Load the dataset
cancer = load_breast_cancer()

# Create a DataFrame
df = pd.DataFrame(data=cancer.data, columns=cancer.feature_names)
df['target'] = cancer.target

# Check for missing values
print(df.isnull().sum())

# Separate features (X) and target (y)
X = df.drop('target', axis=1)
y = df['target']

```

```python
# Scale numerical features
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
```

The output confirms no missing values in the dataset:

```
mean radius                 0
mean texture                0
...
worst fractal dimension     0
target                      0
dtype: int64
```

3 Data Splitting

The scaled data is split into training (398 samples), validation (85 samples), and test (86 samples) sets. The shapes are:

```
Shape of X_train: (398, 30)
Shape of X_val: (85, 30)
Shape of X_test: (86, 30)
Shape of y_test: (86,)
```

4 Model Training and Hyperparameter Tuning

Three models are trained: Logistic Regression, Support Vector Machine (SVM), and Neural Network. Hyperparameter tuning is performed using `GridSearchCV`.

4.1 Logistic Regression

```python
# Define parameter grid for Logistic Regression
lr_param_grid = {
    'C': [0.001, 0.01, 0.1, 1, 10, 100]
}
```

4.2 SVM

```python
# Define parameter grid for SVM
svm_param_grid = {
    'C': [0.1, 1, 10, 100],
    'gamma': ['scale', 'auto'],
    'kernel': ['linear', 'rbf']
}

# Initialize GridSearchCV for SVM
from sklearn.svm import SVC
```

```
svm_grid_search = GridSearchCV(SVC(), svm_param_grid, cv=5,
    scoring='f1')
svm_grid_search.fit(X_train, y_train)

# Print best parameters
print('Best parameters for SVM:', svm_grid_search.best_params_)
```

4.3 Neural Network

```
# Define parameter grid for Neural Network
nn_param_grid = {
    'hidden_layer_sizes': [(50,), (100,), (50, 50), (100, 50)],
    'activation': ['tanh', 'relu'],
    'solver': ['adam', 'sgd'],
    'alpha': [0.0001, 0.001, 0.01, 0.1],
    'learning_rate': ['constant', 'adaptive']
}

# Fit GridSearchCV
nn_grid_search.fit(X_train, y_train)
print('Best F1 score for Neural Network:',
    nn_grid_search.best_score_)
```

Output:

```
Best F1 score for Neural Network: 0.9860754681113963
```

5 Model Evaluation

The models are evaluated on the validation set using accuracy, precision, recall, F1-score, and AUC.

5.1 Logistic Regression Metrics

```
from sklearn.metrics import accuracy_score, precision_score,
    recall_score, f1_score, roc_auc_score

# Logistic Regression evaluation
accuracy = accuracy_score(y_val, y_val_pred)
precision = precision_score(y_val, y_val_pred)
recall = recall_score(y_val, y_val_pred)
f1 = f1_score(y_val, y_val_pred)
auc = roc_auc_score(y_val, model.predict_proba(X_val)[:, 1])

print('Logistic Regression Metrics:')
print(f'Accuracy: {accuracy:.4f}')
print(f'Precision: {precision:.4f}')
print(f'Recall: {recall:.4f}')
print(f'F1-score: {f1:.4f}')
```

```
print(f'AUC: {auc:.4f}')
```

5.2 SVM Metrics

```
svm_model_prob = SVC(probability=True)
svm_model_prob.fit(X_train, y_train)
svm_pred = svm_model_prob.predict(X_val)
svm_accuracy = accuracy_score(y_val, svm_pred)
svm_precision = precision_score(y_val, svm_pred)
svm_recall = recall_score(y_val, svm_pred)
svm_f1 = f1_score(y_val, svm_pred)
svm_auc = roc_auc_score(y_val,
   svm_model_prob.predict_proba(X_val)[:, 1])
```

5.3 Neural Network Test Set Evaluation

The Neural Network, identified as the best model, is evaluated on the test set:

```
test_model = best_nn_model
y_pred = test_model.predict(X_test)

test_accuracy = accuracy_score(y_test, y_pred)
test_precision = precision_score(y_test, y_pred)
test_recall = recall_score(y_test, y_pred)
test_f1 = f1_score(y_test, y_pred)
test_auc = roc_auc_score(y_test,
   test_model.predict_proba(X_test)[:, 1])

print('Neural Network Test Metrics:')
print(f'Accuracy: {test_accuracy:.4f}')
print(f'Precision: {test_precision:.4f}')
print(f'Recall: {test_recall:.4f}')
print(f'F1-score: {test_f1:.4f}')
print(f'AUC: {test_auc:.4f}')
```

Output:

```
Neural Network Test Metrics:
Accuracy: 1.0000
Precision: 1.0000
Recall: 1.0000
F1 score: 1.0000
AUC: 1.0000
```

6 Data Sample

A sample of the dataset is presented in Table 1.

Table 1: Sample of Breast Cancer Dataset

Index	Mean Radius	Mean Texture	Mean Perimeter	Mean Area	Mean Smoothness	Target
0	17.99	10.38	122.80	1001.0	0.11840	0
1	20.57	17.77	132.90	1326.0	0.08474	0
2	19.69	21.25	130.00	1203.0	0.10960	0
3	11.42	20.38	77.58	386.1	0.14250	0
4	20.29	14.34	135.10	1297.0	0.10030	0

7 Key Findings

The breast cancer dataset, containing 569 entries and 31 features, was successfully loaded and confirmed to have no missing values. The data was preprocessed by scaling features and splitting into training, validation, and test sets. The Neural Network model, after hyperparameter tuning, achieved perfect classification on the test set with an F1 score of 0.9861. While promising, further validation is needed to ensure generalization. The analysis provides insights into key features for breast cancer detection.