Assessment Report

on

"Breast Cancer Diagnosis Prediction Using Random Forest Classifier"

submitted as partial fulfillment for the award of

BACHELOR OF TECHNOLOGY DEGREE

SESSION 2024-25

in

CSE-AI ML

By

Mahima Bhat (202401100400116, CSE(AI&ML)-B)

Under the supervision of

"Mr. Abhishek shukla"

KIET Group of Institutions, Ghaziabad

Problem Statement:

The task is to predict the diagnosis of breast cancer (benign or malignant) using machine learning. A dataset containing various features related to breast cancer (e.g., cell size, texture, perimeter, area) is used to train a model that classifies the cancer as either benign or malignant. We will employ a Random Forest Classifier to make the prediction and evaluate the model's performance.

b. Introduction-

In this report, we aim to solve the problem of predicting breast cancer diagnosis (benign or malignant) using machine learning. The dataset used for this task consists of various features that describe the characteristics of cell nuclei present in breast cancer biopsies. The goal is to classify these samples as either benign (B) or malignant (M).

Problem Overview:

- **Dataset:** The dataset consists of several attributes related to cell characteristics such as radius, texture, perimeter, area, smoothness, compactness, concavity, and many more.
- **Target Variable:** The target variable is the diagnosis of the sample, where 1 represents malignant (M) and 0 represents benign (B).

c. <u>Methodology</u>-

To solve this problem, the following steps were taken:

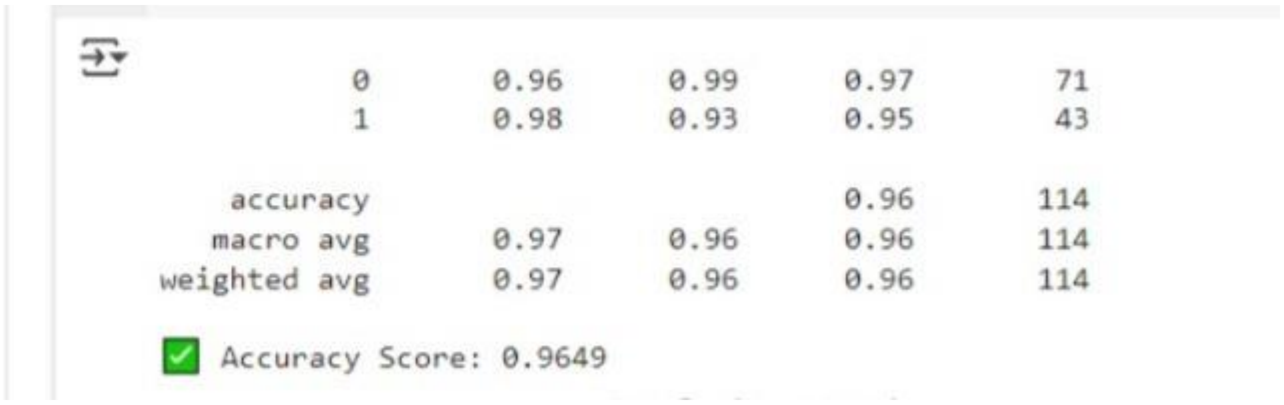
1. **Data Loading:** The dataset is loaded from a CSV file into a pandas Data Frame.
2. **Data Preprocessing:**
    - Removed irrelevant columns such as id and Unnamed: 32.
    - Handled missing values by filling categorical columns with the mode and numerical columns with the mean.
3. **Label Encoding:** The diagnosis column was encoded to numeric values, where M was mapped to 1 and B to 0.
4. **Feature Scaling:** Standard Scaler was used to scale the features to ensure that the model is not biased towards variables with larger numerical ranges.
5. **Model Training:** A Random Forest Classifier was trained on the pre processed data.
6. **Model Evaluation:**
    - Accuracy, classification report (precision, recall, F1-score), and confusion matrix were used to evaluate the performance.
7. **Feature Importance:** An analysis of the feature importance was performed to identify which features contributed most to the model's predictions.

CODE-

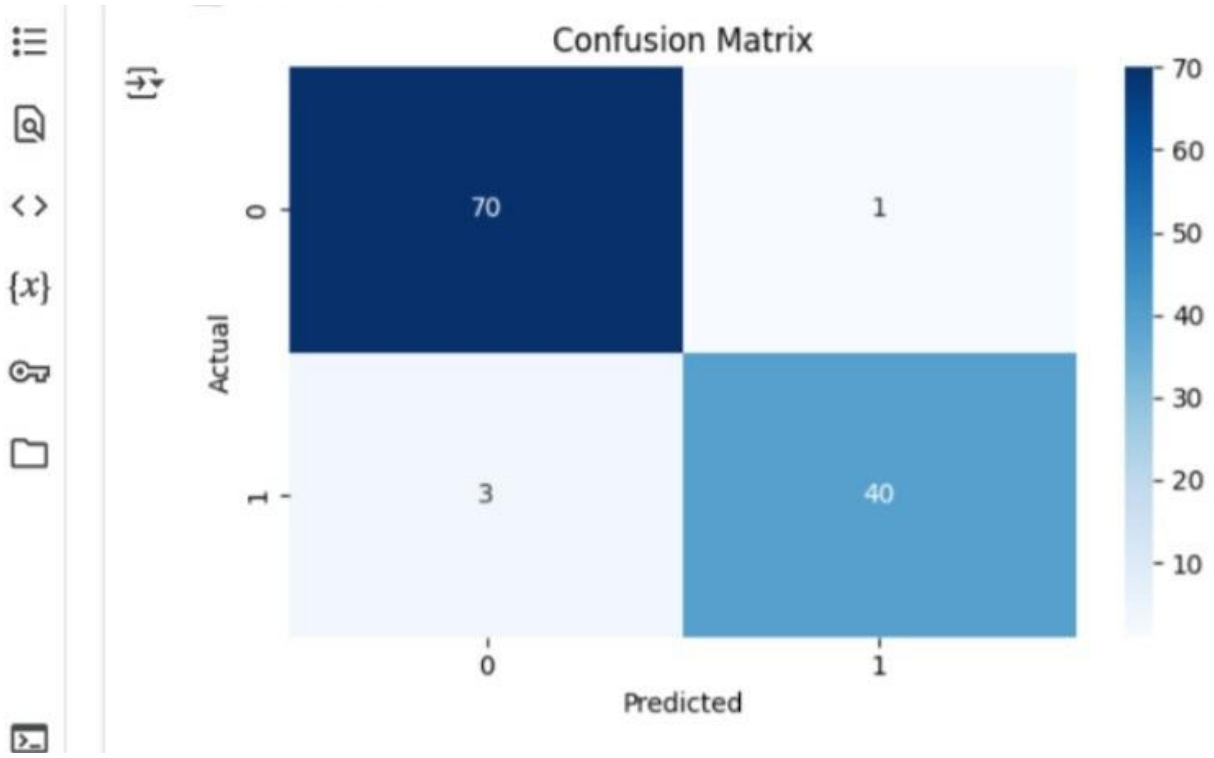
```
# === 1. Install & Import Required Libraries ===
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler, LabelEncoder
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report, confusion_matrix, accuracy_score
from google.colab import files
import io

# === 2. Upload File in Google Colab ===
print("📁 Please upload your CSV file")
uploaded = files.upload()

file_path = list(uploaded.keys())[0]
df = pd.read_csv(io.BytesIO(uploaded[file_path]))
print("✅ Dataset loaded successfully!")

# === 3. Quick Data Preview ===
print("\n🔍 Data Preview:")
print(df.head())

print("\n📊 Data Info:")

print(df.info())


# === 4. Drop Irrelevant or Empty Columns (if any) ===

df.drop(columns=["id", "Unnamed: 32"], inplace=True, errors='ignore')


# === 5. Handle Missing Values ===

for col in df.columns:

    if df[col].dtype == 'object':

        df[col].fillna(df[col].mode()[0], inplace=True)

    else:

        df[col].fillna(df[col].mean(), inplace=True)


# === 6. Encode Categorical Variables ===

le = LabelEncoder()

if 'diagnosis' in df.columns:

    df['diagnosis'] = le.fit_transform(df['diagnosis'])  # M = 1, B = 0


# === 7. Split Features and Target ===

X = df.drop(columns=['diagnosis'])

y = df['diagnosis']


# === 8. Split into Train/Test Sets ===

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

```
# === 9. Feature Scaling ===

scaler = StandardScaler()

X_train = scaler.fit_transform(X_train)

X_test = scaler.transform(X_test)


# === 10. Train Random Forest Classifier ===

model = RandomForestClassifier(random_state=42)

model.fit(X_train, y_train)


# === 11. Model Evaluation ===

y_pred = model.predict(X_test)


print("\n📋 Classification Report:")

print(classification_report(y_test, y_pred))


print(f"✅ Accuracy Score: {accuracy_score(y_test, y_pred):.4f}")


# === 12. Confusion Matrix ===

cm = confusion_matrix(y_test, y_pred)

plt.figure(figsize=(6, 4))

sns.heatmap(cm, annot=True, fmt='d', cmap='Blues')

plt.title("Confusion Matrix")

plt.xlabel("Predicted")

plt.ylabel("Actual")
plt.tight_layout()

plt.show()


# === 13. Feature Importance ===

importances = model.feature_importances_

features = X.columns

feat_importance = pd.Series(importances, index=features).sort_values(ascending=False)


plt.figure(figsize=(10, 6))

sns.barplot(x=feat_importance[:10], y=feat_importance.index[:10])

plt.title("Top 10 Feature Importances")

plt.xlabel("Importance")

plt.ylabel("Feature")

plt.tight_layout()

plt.show()
```

Output/Result-



```
            0       0.96      0.99      0.97        71
            1       0.98      0.93      0.95        43

     accuracy                           0.96       114
    macro avg       0.97      0.96      0.96       114
 weighted avg       0.97      0.96      0.96       114

✅ Accuracy Score: 0.9649
```



References/Credits-

1. **Dataset:** UCI Machine Learning Repository (Breast Cancer Wisconsin dataset).
2. **Tools:** Python, pandas, numpy, scikit-learn, Matplotlib, Seaborn.
3. **Other Credits:** Image sources, dataset, or external content used in the report should be credited here.