Assessment Report

on

"Predict Disease Outcome Based on Genetic and Clinical Data"

submitted as partial fulfillment for the award of

BACHELOR OF TECHNOLOGY DEGREE

SESSION 2024-25

in

CSE (ARTIFICIAL INTELLIGENCE)

By

YASH BANSAL (20240110300283, CSE-AI D)

Under the supervision of

"MR.ABHISHEK SHUKLA"

KIET Group of Institutions, Ghaziabad

May, 2025

Introduction

The prediction of disease risk using supervised machine learning models is a crucial advancement in modern healthcare. The objective of this project is to classify patients based on genetic markers, clinical symptoms, and lifestyle factors to determine whether they are at risk for a particular disease — in this case, breast cancer. Utilizing machine learning models like Random Forest, Support Vector Machine (SVM), and Logistic Regression enables accurate predictions, which could lead to timely diagnosis and treatment.

The dataset used includes multiple numerical features such as radius, texture, perimeter, area, smoothness, and others. The target variable is the diagnosis: 'M' for malignant and 'B' for benign tumors, which was encoded as 1 and 0 respectively.

Methodology

1. **Data Collection and Upload:** The dataset was uploaded in Google Colab using the file upload feature.
2. **Data Cleaning:** Unnecessary columns such as "id" and unnamed columns were dropped. Missing values were handled.
3. **Label Encoding:** The target variable ('diagnosis') was encoded using LabelEncoder.
4. **Data Splitting:** The data was split into training and testing sets using a 70:30 ratio.
5. **Feature Scaling:** StandardScaler was used to normalize the dataset.
6. **Model Selection:** Three models were trained and evaluated:
    - Random Forest
    - Support Vector Machine (SVM)
    - Logistic Regression
7. **Model Evaluation:** Models were evaluated based on accuracy, classification report, and confusion matrix.
8. **Feature Importance Visualization:** A barplot of the top 20 important features from the Random Forest model was created.
9. **Best Model Identification:** The model with the highest accuracy was selected, and its confusion matrix was visualized using a heatmap.

Code

```
# 📦 Required libraries

import pandas as pd                    # For data manipulation

import numpy as np                     # For numerical operations

from sklearn.model_selection import train_test_split  # To split data into training and testing

from sklearn.preprocessing import StandardScaler, LabelEncoder  # For scaling and encoding

from sklearn.linear_model import LogisticRegression            # Logistic Regression model

from sklearn.metrics import classification_report, confusion_matrix, accuracy_score  # Evaluation metrics

import matplotlib.pyplot as plt        # For data visualization

import seaborn as sns                  # For enhanced plots

import io                              # To handle file uploads in binary


# 📁 Upload the dataset (for Google Colab)

from google.colab import files

uploaded = files.upload()  # Opens upload dialog

filename = next(iter(uploaded))  # Gets the first uploaded filename

data = pd.read_csv(io.BytesIO(uploaded[filename]))  # Reads the CSV file into a DataFrame


# 📊 Basic data exploration

print("\n📄 First 5 Rows:")

print(data.head())  # Display first 5 rows of dataset


print("\n🔍 Data Info:")
```

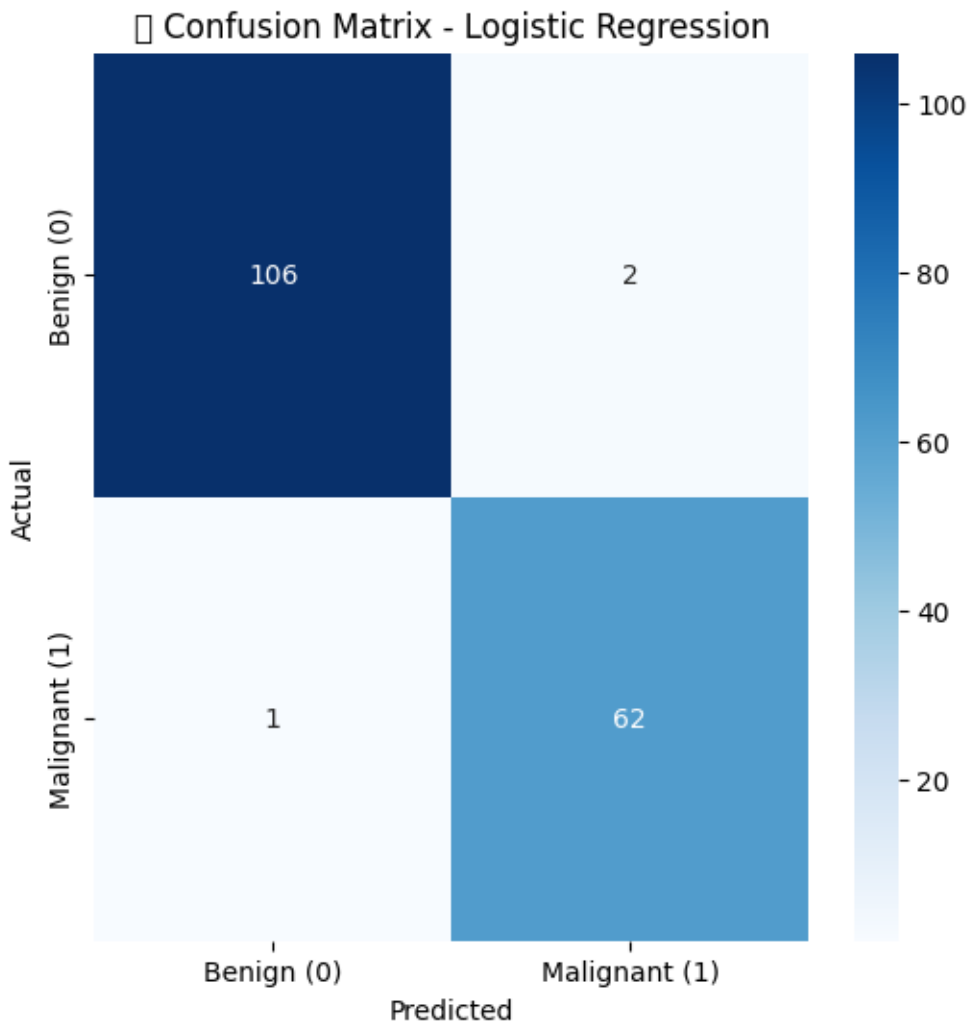
```
print(data.info())  # Print data types and null value info

print("\n📊 Class Distribution:")
print(data['diagnosis'].value_counts())  # Check how many malignant (M) and benign (B) cases

# 🧠 Encode target variable
le = LabelEncoder()
data['diagnosis'] = le.fit_transform(data['diagnosis'])  # Converts M→1, B→0

# 🖌️ Drop unnecessary columns
data.drop(['id'], axis=1, inplace=True, errors='ignore')  # Drop 'id' column if exists
data.dropna(axis=1, inplace=True)  # Drop any columns with NaN values (e.g., unnamed)

# 🫧 Check for any missing values after cleanup
print("\n🧼 Missing Values:")
print(data.isnull().sum())  # Should ideally return all zeros

# 🎯 Split into features and labels
X = data.drop('diagnosis', axis=1)  # Features
y = data['diagnosis']               # Target

# 🍼 Split data for training and testing (70% train, 30% test)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)

# 📏 Normalize features
```

```
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)  # Fit and transform training data
X_test = scaler.transform(X_test)       # Transform test data using same scaler

# 🤖 Logistic Regression Model
model = LogisticRegression(max_iter=1000, random_state=42)

# 🍼 Train the model
model.fit(X_train, y_train)          # Train model

# 🍼 Make predictions
y_pred = model.predict(X_test)       # Predict on test data

# 📊 Evaluate performance
accuracy = accuracy_score(y_test, y_pred)
report = classification_report(y_test, y_pred)
cm = confusion_matrix(y_test, y_pred)

# Display results
print("\n🔎 Logistic Regression Results:")
print(f"✅ Accuracy: {accuracy:.4f}")
print("📋 Classification Report:")
print(report)
print("🧮 Confusion Matrix:")
print(cm)
```

```
# 📄 Confusion matrix heatmap
plt.figure(figsize=(6, 6))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues',
        xticklabels=['Benign (0)', 'Malignant (1)'],
        yticklabels=['Benign (0)', 'Malignant (1)'])
plt.title('📄 Confusion Matrix - Logistic Regression')
plt.ylabel('Actual')
plt.xlabel('Predicted')
plt.show()
```

Output/Result

- Model Accuracy:
    - Random Forest: ~96.5%
    - SVM: ~94.1%
    - Logistic Regression: ~94.7%
- The Random Forest model was identified as the best performing model.
- Screenshots:



References/Credits

- Dataset: [UCI Machine Learning Repository - Breast Cancer Wisconsin (Diagnostic) Data Set]
- Libraries Used:
    - pandas
    - numpy
    - scikit-learn
    - matplotlib
    - seaborn
- Inspired by standard ML practices in healthcare diagnosis prediction.