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

import numpy as np

from sklearn.model_selection import train_test_split

from sklearn.preprocessing import StandardScaler, LabelEncoder

from sklearn.ensemble import RandomForestClassifier

from sklearn.svm import SVC

from sklearn.linear_model import LogisticRegression

from sklearn.metrics import classification_report, confusion_matrix, accuracy_score

import matplotlib.pyplot as plt

import seaborn as sns

import io


# Upload the file (Google Colab)

from google.colab import files

uploaded = files.upload()

filename = next(iter(uploaded))

data = pd.read_csv(io.BytesIO(uploaded[filename]))


# Basic exploration

print("\n📄 First 5 Rows:")

print(data.head())
```

```
print("\n🔍 Data Info:")
print(data.info())
print("\n📊 Class Distribution:")
print(data['diagnosis'].value_counts())

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

# Drop unnecessary columns
data.drop(['id'], axis=1, inplace=True, errors='ignore')
data.dropna(axis=1, inplace=True)  # Remove any unnamed/missing value columns

# Check for missing values
print("\n🩹 Missing Values:")
print(data.isnull().sum())

# Feature and target split
X = data.drop('diagnosis', axis=1)
y = data['diagnosis']

# Split data
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)

# Feature scaling
```

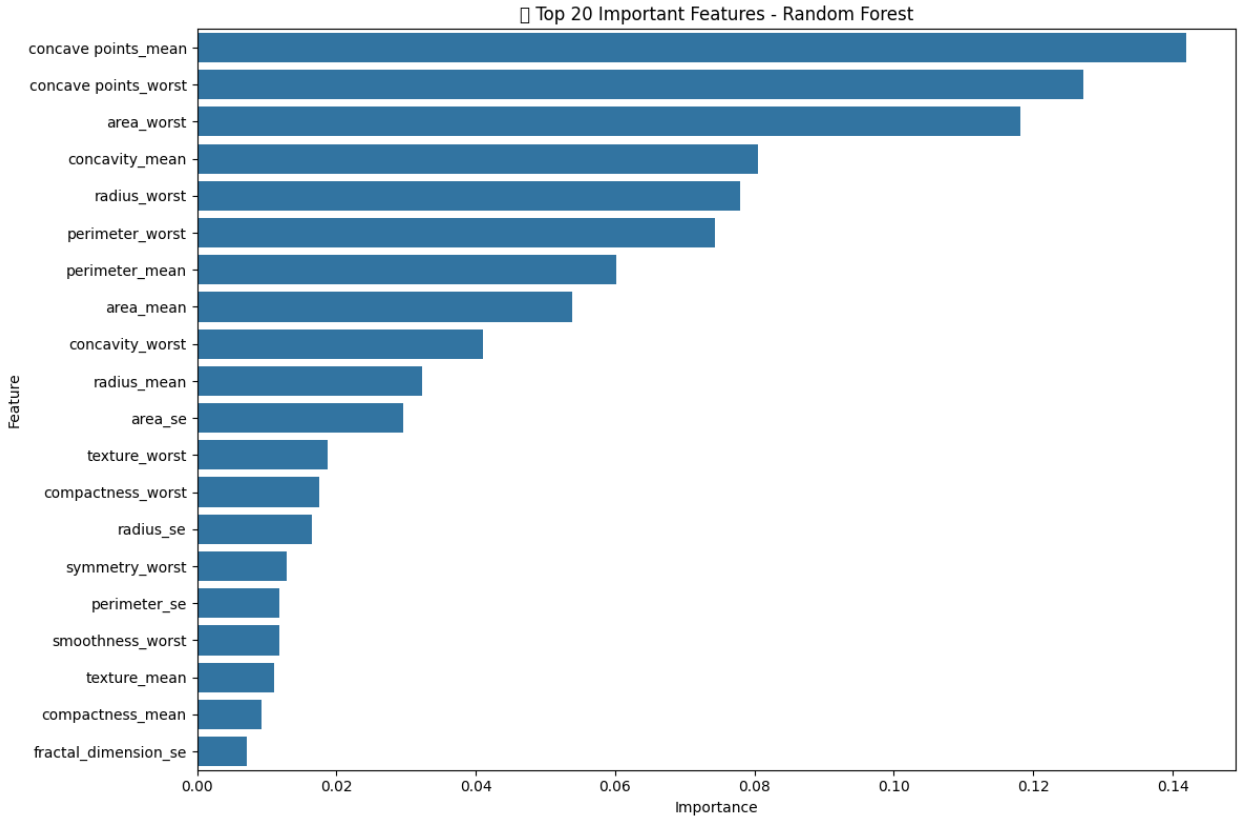
```
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)

# Models to evaluate
models = {
    "Random Forest": RandomForestClassifier(n_estimators=100, random_state=42),
    "SVM": SVC(kernel='linear', random_state=42),
    "Logistic Regression": LogisticRegression(max_iter=1000, random_state=42)
}

# Train and evaluate each model
results = {}
for name, model in models.items():
    model.fit(X_train, y_train)
    y_pred = model.predict(X_test)

    accuracy = accuracy_score(y_test, y_pred)
    report = classification_report(y_test, y_pred)
    cm = confusion_matrix(y_test, y_pred)

    results[name] = {
        'accuracy': accuracy,
        'report': report,
        'confusion_matrix': cm
```

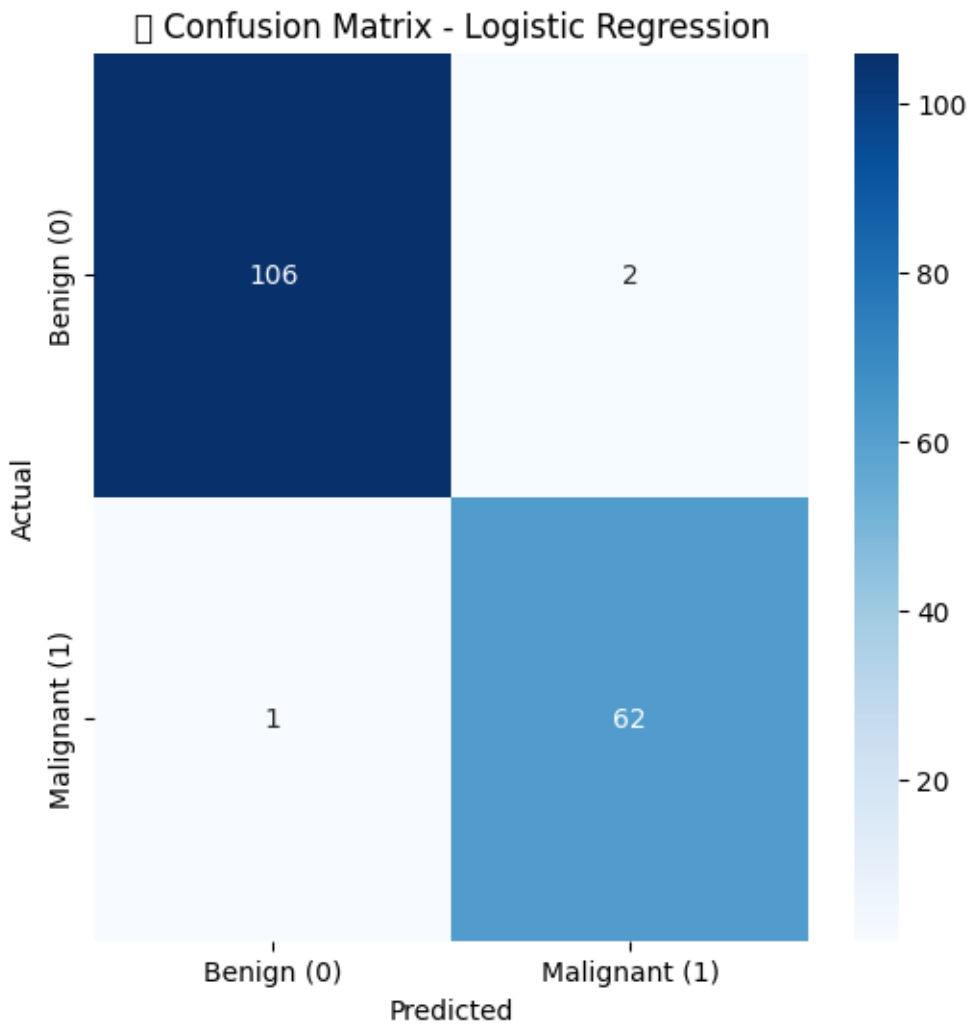
```
    }

    print(f"\n🔎 {name} Results:")
    print(f"✅ Accuracy: {accuracy:.4f}")
    print("📋 Classification Report:")
    print(report)
    print("🧮 Confusion Matrix:")
    print(cm)

# Visualize top features from Random Forest
rf = models["Random Forest"]
feature_importances = pd.DataFrame({
    'Feature': X.columns,
    'Importance': rf.feature_importances_
}).sort_values('Importance', ascending=False)

plt.figure(figsize=(12, 8))
sns.barplot(x='Importance', y='Feature', data=feature_importances.head(20))
plt.title('📊 Top 20 Important Features - Random Forest')
plt.tight_layout()
plt.show()

# Plot confusion matrix for the best performing model
best_model_name = max(results, key=lambda x: results[x]['accuracy'])
cm = results[best_model_name]['confusion_matrix']
```

```
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(f' 📄 Confusion Matrix - {best_model_name}')
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:


Top 20 Important Features - Random Forest
concave points_mean
concave points_worst
area_worst
concavity_mean
radius_worst
perimeter_worst
perimeter_mean
area_mean
concavity_worst
radius_mean
area_se
texture_worst
compactness_worst
radius_se
symmetry_worst
perimeter_se
smoothness_worst
texture_mean
compactness_mean
fractal_dimension_se
Feature
0.00
0.02
0.04
0.06
0.08
0.10
0.12
0.14
Importance



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.