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-AIML

By

Utkarsh Pandey

Under the supervision of

Abhishek Shuka sir

KIET Group of Institutions, Ghaziabad

Affiliated to

Dr. A.P.J. Abdul Kalam Technical University, Lucknow

(Formerly UPTU)

May, 2025

Introduction:

In medical diagnosis, classification problems help predict disease outcomes by analysing genetic and clinical data. This project focuses on building a machine learning model using logistic regression for classification and evaluating its performance using various metrics. Additionally, clustering techniques (such as K-Means) are applied for segmentation to uncover meaningful patterns.

The dataset comprises tumour-related measurements that describe mean values, standard errors, and worst-case scenarios of features such as radius, texture, perimeter, area, compactness, concavity, symmetry, and fractal dimensions.

This study aims to:

- Train a classification model to differentiate between benign and malignant tumours.
- Compute evaluation metrics like accuracy, precision, recall, and F1 score.
- Visualize the model's performance with a **heatmap of the confusion matrix**.
- Perform clustering with K-Means and assess the cluster quality using the **silhouette score**.
- Use **PCA (Principal Component Analysis)** to visualize clusters in a 2D space.

Methodology:

The approach taken to solve the problem consists of the following steps:

1. **Data Preprocessing**
   - Handling missing values using the median imputation strategy.
   - Scaling features using Standard Scaler.
   - Mapping target variable (diagnosis): **Malignant (M) = 1, Benign (B) = 0**.
2. **Classification Model**
   - Splitting data into training and testing sets (80/20 ratio).
   - Training **Logistic Regression** as the classification model.
   - Making predictions on the test dataset.
   - Evaluating model performance using **confusion matrix, accuracy, precision, recall, and F1 score**.
3. **Visualization**
   - Plotting the **heatmap** for the confusion matrix to observe classification results.
4. **Clustering & Segmentation**
   - Using **K-Means clustering** to categorize tumours into different clusters.
   - Computing **silhouette score** to assess clustering quality.
   - Applying **PCA** to visualize clusters in two dimensions.

Code Implementation:

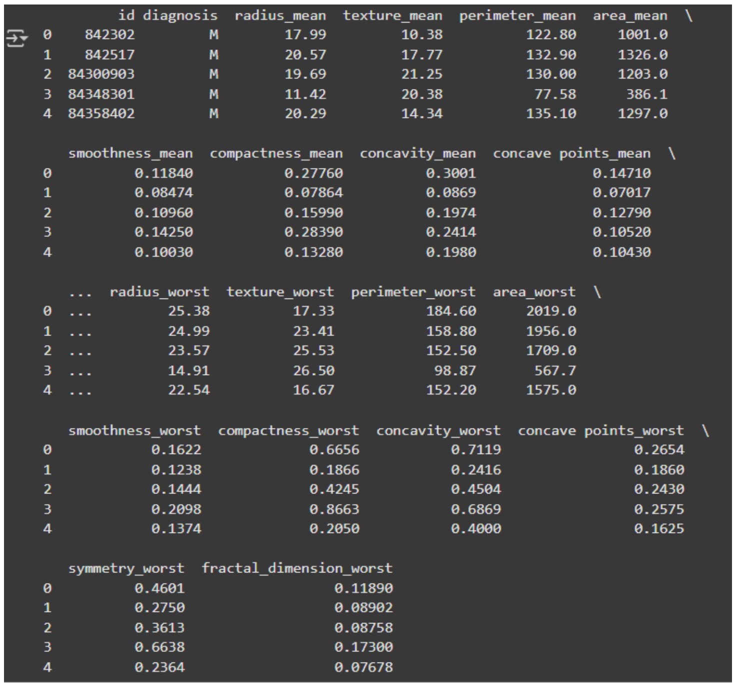
```
# Import required libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import confusion_matrix, accuracy_score, precision_score,
recall_score, f1_score, classification_report
from sklearn.impute import SimpleImputer
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA
from sklearn.cluster import KMeans
from sklearn.metrics import silhouette_score
# Upload CSV file
from google.colab import files
uploaded = files.upload()
# Load dataset
df = pd.read_csv("3. Predict Disease Outcome Based on Genetic and Clinical
Data.csv")
```

```
df = df.loc[:, ~df.columns.str.contains("Unnamed")]

# Preprocessing

df['diagnosis'] = df['diagnosis'].map({'M': 1, 'B': 0})

X = df.drop("diagnosis", axis=1)

y = df["diagnosis"]

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.20,
random_state=42, stratify=y)

imputer = SimpleImputer(strategy='median')

X_train_imputed = imputer.fit_transform(X_train)

X_test_imputed = imputer.transform(X_test)

scaler = StandardScaler()

X_train_scaled = scaler.fit_transform(X_train_imputed)

X_test_scaled = scaler.transform(X_test_imputed)

# Classification using Logistic Regression

clf = LogisticRegression(max_iter=1000, random_state=42)

clf.fit(X_train_scaled, y_train)

y_pred = clf.predict(X_test_scaled)

# Evaluation Metrics

cm = confusion_matrix(y_test, y_pred)

acc = accuracy_score(y_test, y_pred)

prec = precision_score(y_test, y_pred)

rec = recall_score(y_test, y_pred)
```

```
f1 = f1_score(y_test, y_pred)

print(f"Accuracy: {acc:.4f}")

print(f"Precision: {prec:.4f}")

print(f"Recall: {rec:.4f}")

print(f"F1 Score: {f1:.4f}")

print("\nClassification Report:\n", classification_report(y_test, y_pred))

# Confusion Matrix Heatmap

plt.figure(figsize=(6,5))

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

            xticklabels=["Benign (0)", "Malignant (1)"],

            yticklabels=["Benign (0)", "Malignant (1)"])

plt.title("Confusion Matrix")

plt.xlabel("Predicted Label")

plt.ylabel("True Label")

plt.show()

# Clustering

imputer_all = SimpleImputer(strategy='median')

X_features_imputed = imputer_all.fit_transform(X)

scaler_all = StandardScaler()

X_scaled_all = scaler_all.fit_transform(X_features_imputed)

k = 2

kmeans = KMeans(n_clusters=k, random_state=42)
```

```
clusters = kmeans.fit_predict(X_scaled_all)

sil_score = silhouette_score(X_scaled_all, clusters)

print(f"Silhouette Score: {sil_score:.4f}")

# PCA Visualization

pca = PCA(n_components=2, random_state=42)

X_pca = pca.fit_transform(X_scaled_all)

plt.figure(figsize=(8,6))

sns.scatterplot(x=X_pca[:, 0], y=X_pca[:, 1], hue=clusters, palette="viridis", s=50)

plt.title("PCA Projection - KMeans Clusters")

plt.xlabel("Principal Component 1")

plt.ylabel("Principal Component 2")

plt.legend(title="Cluster")

plt.show()
```

Output & Results:



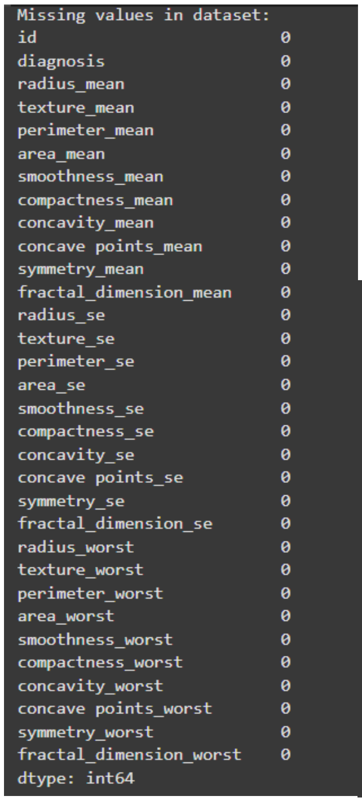
```
         id diagnosis  radius_mean  texture_mean  perimeter_mean  area_mean  \
0    842302         M        17.99         10.38          122.80     1001.0
1    842517         M        20.57         17.77          132.90     1326.0
2  84300903         M        19.69         21.25          130.00     1203.0
3  84348301         M        11.42         20.38           77.58      386.1
4  84358402         M        20.29         14.34          135.10     1297.0

   smoothness_mean  compactness_mean  concavity_mean  concave points_mean  \
0          0.11840           0.27760          0.3001              0.14710
1          0.08474           0.07864          0.0869              0.07017
2          0.10960           0.15990          0.1974              0.12790
3          0.14250           0.28390          0.2414              0.10520
4          0.10030           0.13280          0.1980              0.10430

   ...  radius_worst  texture_worst  perimeter_worst  area_worst  \
0  ...         25.38          17.33           184.60      2019.0
1  ...         24.99          23.41           158.80      1956.0
2  ...         23.57          25.53           152.50      1709.0
3  ...         14.91          26.50            98.87       567.7
4  ...         22.54          16.67           152.20      1575.0

   smoothness_worst  compactness_worst  concavity_worst  concave points_worst  \
0            0.1622             0.6656           0.7119                0.2654
1            0.1238             0.1866           0.2416                0.1860
2            0.1444             0.4245           0.4504                0.2430
3            0.2098             0.8663           0.6869                0.2575
4            0.1374             0.2050           0.4000                0.1625

   symmetry_worst  fractal_dimension_worst
0          0.4601                  0.11890
1          0.2750                  0.08902
2          0.3613                  0.08758
3          0.6638                  0.17300
4          0.2364                  0.07678
```

Accuracy: 0.9649

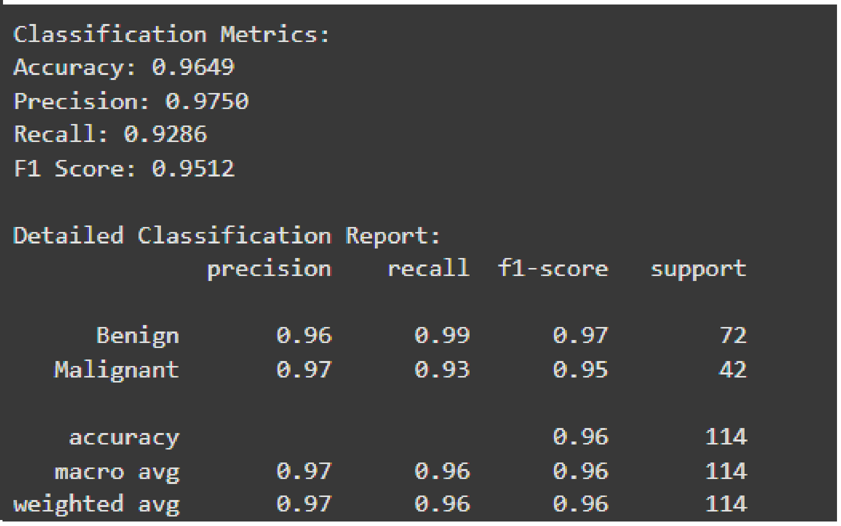
Precision: 0.9750

Recall: 0.9286

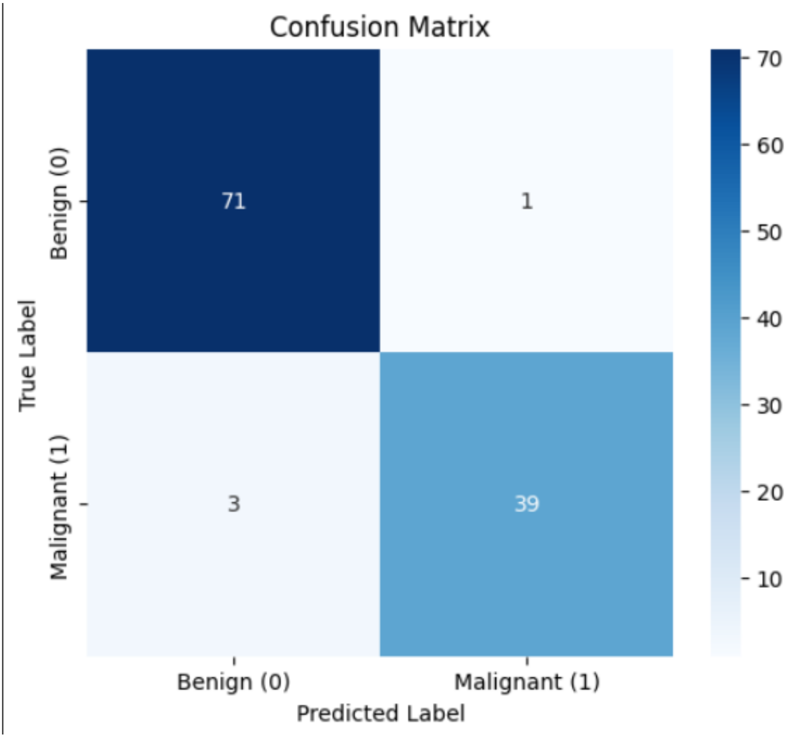
F1 Score: 0.9512



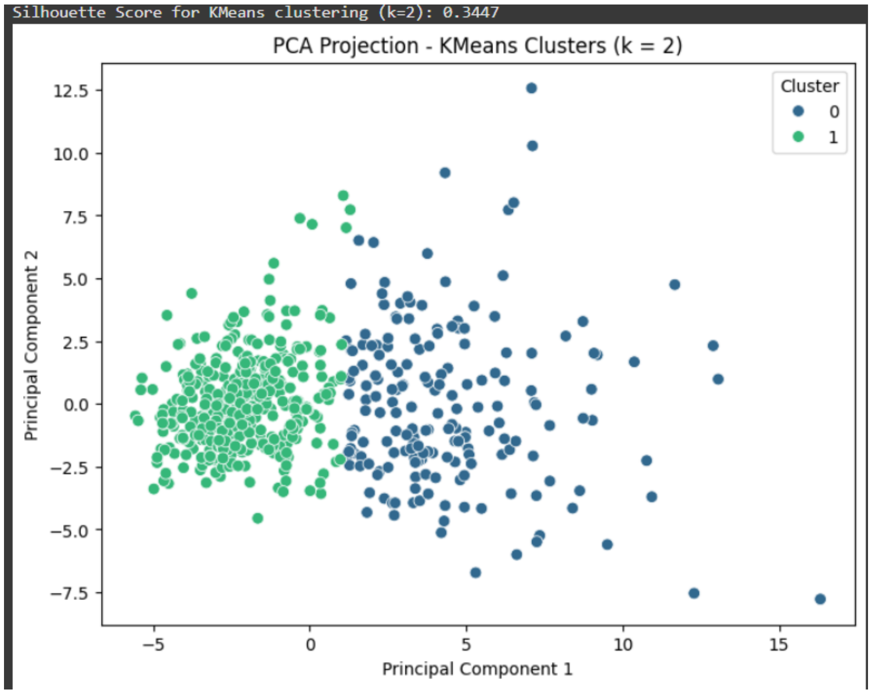
```
Missing values in dataset:
id                         0
diagnosis                  0
radius_mean                0
texture_mean               0
perimeter_mean             0
area_mean                  0
smoothness_mean            0
compactness_mean           0
concavity_mean             0
concave points_mean        0
symmetry_mean              0
fractal_dimension_mean     0
radius_se                  0
texture_se                 0
perimeter_se               0
area_se                    0
smoothness_se              0
compactness_se             0
concavity_se               0
concave points_se          0
symmetry_se                0
fractal_dimension_se       0
radius_worst               0
texture_worst              0
perimeter_worst            0
area_worst                 0
smoothness_worst           0
compactness_worst          0
concavity_worst            0
concave points_worst       0
symmetry_worst             0
fractal_dimension_worst    0
dtype: int64
```



```
Classification Metrics:
Accuracy: 0.9649
Precision: 0.9750
Recall: 0.9286
F1 Score: 0.9512

Detailed Classification Report:
              precision    recall  f1-score   support

      Benign       0.96      0.99      0.97        72
   Malignant       0.97      0.93      0.95        42

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



```
Silhouette Score for KMeans clustering (k=2): 0.3447
```



References & Credits:

- Dataset: The CSV file uploaded for disease outcome prediction from Kaggle
- Libraries Used: Scikit-learn, Pandas, NumPy, Matplotlib, Seaborn
- Documentation & Guides:
    - Scikit-learn documentation (https://scikit-learn.org)
    - Seaborn visualization (https://seaborn.pydata.org)
    - Google Colab for running the notebook