Assessment Report

on

"Predict Disease Outcome Based on Genetic and Clinical Data"

submitted as partial fulfilment for the award of

BACHELOR OF TECHNOLOGY DEGREE

SESSION 2024-25

in

Name of discipline

By

Prajesh Singh Meena (202401100400136)

Under the supervision of

"Abhishek Shukla"

KIET Group of Institutions, Ghaziabad

Affiliated to

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

(Formerly UPTU)

May, 2025

1. Problem Statement

In the era of data science and machine learning, data visualization is a crucial step in understanding patterns and making informed decisions. This project involves uploading a real-world dataset, analysing it using data visualization techniques, and identifying relationships between different numerical variables through a correlation heatmap. The objective is to gain insights into the data structure and highlight which features are strongly correlated.

We will use Python as our primary programming language due to its simplicity and powerful data analysis libraries such as pandas, matplotlib, and seaborn. The goal is to process the dataset from scratch, clean it if necessary, generate histograms to visualize distributions, and finally, produce a heatmap that reflects the correlations between features.

2. Introduction

With the explosion of data in every field, making sense of that data is more important than ever. Raw numbers are difficult to interpret, especially in large datasets. This is where data visualization steps in. Visualization tools help to quickly grasp patterns, trends, and outliers that might otherwise go unnoticed. In machine learning and data analysis, visualizations not only enhance our understanding but also guide decision-making about which features to use or remove.

In this project, we are working with a dataset (most likely the Breast Cancer Wisconsin dataset, based on the structure), which contains measurements from cell nuclei present in breast cancer biopsies. Each feature corresponds to a property like radius, texture, perimeter, and so on. Our task is to load this dataset, explore it visually, and understand which features are most closely related to each other.

We will create histograms to observe the distribution of each feature and use a heatmap to display the correlation between every pair of numerical features. This is especially useful in identifying multicollinearity, selecting features for modeling, or simply understanding the dataset better.

3. Methodology

conduct this analysis, the following approach was followed:

1. **Dataset Upload**
   Using Google Colab's files.upload() feature, we uploaded the CSV file containing our dataset. This is a convenient method for handling files when working in cloud-based notebooks.

2. **Data Loading and Exploration**
   Once the file was uploaded, we used pandas to load the dataset into a DataFrame. We then printed the first few rows using head() and examined its structure using info(). This allowed us to check for any missing values, non-numeric columns, or unnecessary fields like an ID column.

3. **Data Cleaning (if necessary)**
   In some cases, datasets contain columns that aren't useful for analysis, such as unnamed columns or IDs. These were excluded from the correlation analysis by filtering out columns whose names contained 'id' or 'Unnamed'.

4. **Histogram Plotting**
   A histogram was generated for each numerical feature using DataFrame.hist() with matplotlib. These visualizations helped us understand the spread, skewness, and possible outliers in the data.

5. **Correlation Matrix and Heatmap**
   To analyze the relationships between features, we computed the correlation matrix using DataFrame.corr() on numeric columns. We visualized this using a seaborn.heatmap() with annotations, color coding, and proper label formatting to make it visually informative and accessible.

6. **Final Adjustments**
   The heatmap was formatted with appropriate sizing (figsize), font adjustments (annot_kws), and layout fixes (tight_layout()) to ensure readability even for datasets with many features.

4. Code

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from google.colab import files

def upload_dataset():
    uploaded = files.upload()
    if uploaded:
        file_name = next(iter(uploaded))
        return file_name
    return None

def load_and_display_data(file_path):
    data = pd.read_csv(file_path)
    print("Dataset preview:")
    print(data.head())
    print("\nDataset info:")
    data.info()
    return data

def visualize_data(data):
    data.hist(bins=20, figsize=(12, 10))
    plt.tight_layout()
    plt.show()

    numeric_data = data.select_dtypes(include=['number'])
    numeric_data = numeric_data.loc[:, ~numeric_data.columns.str.contains('^Unnamed|id', case=False)]
    correlation_matrix = numeric_data.corr()
```

```
plt.figure(figsize=(16, 14))
    sns.heatmap(
        correlation_matrix,
        annot=True,
        cmap="coolwarm",
        fmt=".2f",
        square=True,
        annot_kws={"size": 8}
    )
    plt.xticks(rotation=45, ha='right', fontsize=10)
    plt.yticks(rotation=0, fontsize=10)
    plt.title("Correlation Heatmap", fontsize=16, pad=20)
    plt.tight_layout()
    plt.show()

def main():
    file_path = upload_dataset()
    if not file_path:
        return
    data = load_and_display_data(file_path)
    visualize_data(data)

main()
```

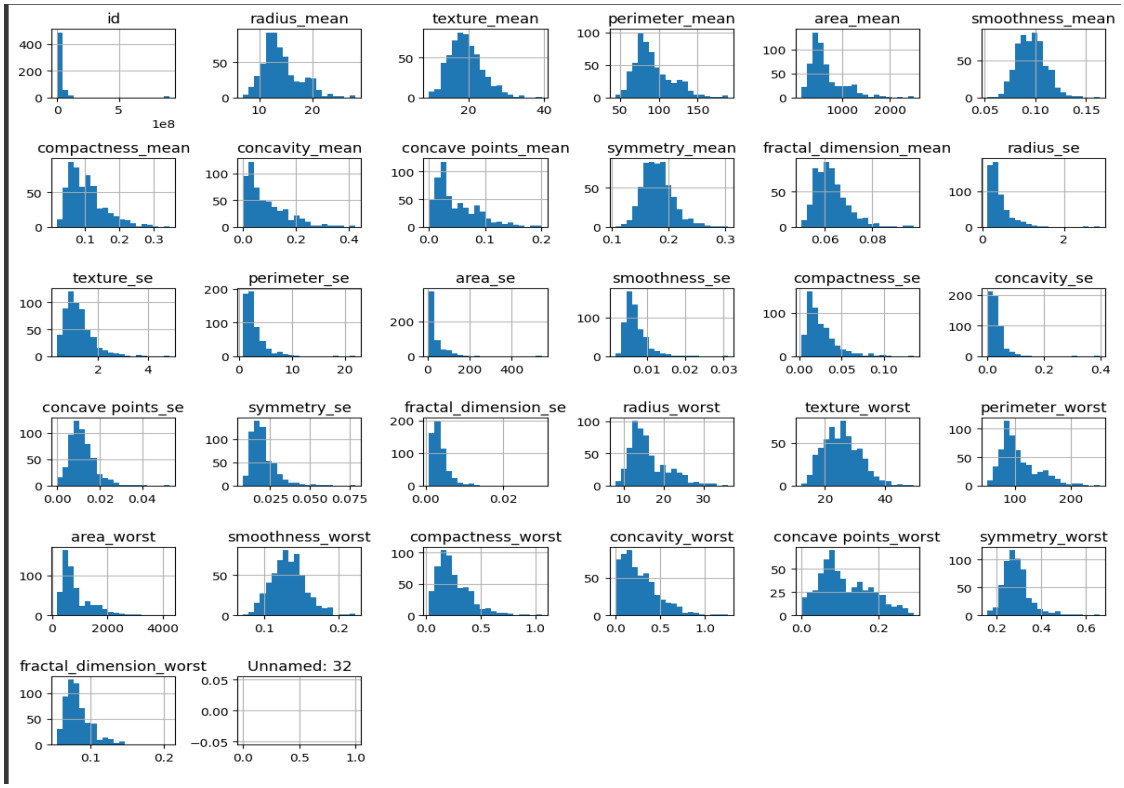
5. Output / Result

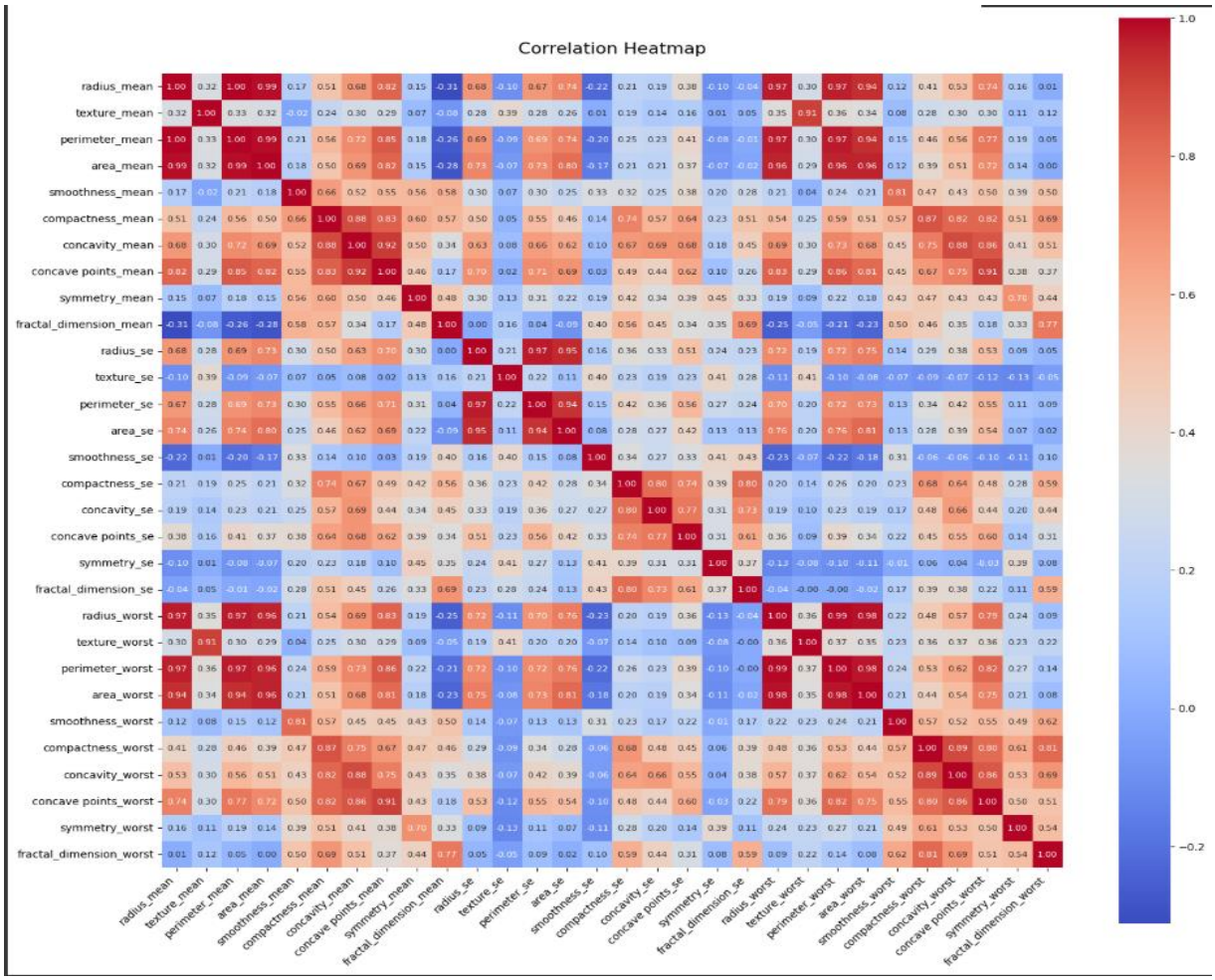
Histogram Output

```
Choose Files  Predict Dis…ical Data.csv
• Predict Disease Outcome Based on Genetic and Clinical Data.csv(text/csv) - 125204 bytes, last modified: 4/18/2025 - 100% done
Saving Predict Disease Outcome Based on Genetic and Clinical Data.csv to Predict Disease Outcome Based on Genetic and Clinical Data (5).csv
Dataset preview:
        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

   ...  texture_worst  perimeter_worst  area_worst  smoothness_worst  \
0  ...          17.33           184.60      2019.0            0.1622
1  ...          23.41           158.80      1956.0            0.1238
2  ...          25.53           152.50      1709.0            0.1444
3  ...          26.50            98.87       567.7            0.2098
4  ...          16.67           152.20      1575.0            0.1374

   compactness_worst  concavity_worst  concave points_worst  symmetry_worst  \
0             0.6656           0.7119                0.2654          0.4601
1             0.1866           0.2416                0.1860          0.2750
2             0.4245           0.4504                0.2430          0.3613
3             0.8663           0.6869                0.2575          0.6638
4             0.2050           0.4000                0.1625          0.2364

   fractal_dimension_worst  Unnamed: 32
0                  0.11890          NaN
1                  0.08902          NaN
2                  0.08758          NaN
3                  0.17300          NaN
4                  0.07678          NaN
```

```
Dataset info:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 569 entries, 0 to 568
Data columns (total 33 columns):
 #    Column                   Non-Null Count   Dtype
---   ------                   --------------   -----
 0    id                       569 non-null     int64
 1    diagnosis                569 non-null     object
 2    radius_mean              569 non-null     float64
 3    texture_mean             569 non-null     float64
 4    perimeter_mean           569 non-null     float64
 5    area_mean                569 non-null     float64
 6    smoothness_mean          569 non-null     float64
 7    compactness_mean         569 non-null     float64
 8    concavity_mean           569 non-null     float64
 9    concave points_mean      569 non-null     float64
 10   symmetry_mean            569 non-null     float64
 11   fractal_dimension_mean   569 non-null     float64
 12   radius_se                569 non-null     float64
 13   texture_se               569 non-null     float64
 14   perimeter_se             569 non-null     float64
 15   area_se                  569 non-null     float64
 16   smoothness_se            569 non-null     float64
 17   compactness_se           569 non-null     float64
 18   concavity_se             569 non-null     float64
 19   concave points_se        569 non-null     float64
 20   symmetry_se              569 non-null     float64
 21   fractal_dimension_se     569 non-null     float64
 22   radius_worst             569 non-null     float64
 23   texture_worst            569 non-null     float64
 24   perimeter_worst          569 non-null     float64
 25   area_worst               569 non-null     float64
 26   smoothness_worst         569 non-null     float64
 27   compactness_worst        569 non-null     float64
 28   concavity_worst          569 non-null     float64
 29   concave points_worst     569 non-null     float64
 30   symmetry_worst           569 non-null     float64
 31   fractal_dimension_worst  569 non-null     float64
 32   Unnamed: 32              0 non-null       float64
dtypes: float64(31), int64(1), object(1)
```

Correlation Heatmap Output





6. References / Credits

- **Pandas Library Documentation**
  https://pandas.pydata.org/

- **Matplotlib Library Documentation**
  https://matplotlib.org/

- **Seaborn Library Documentation**
  https://seaborn.pydata.org/

- **Dataset**: Breast Cancer Wisconsin (Diagnostic) Data Set
  Source: UCI Machine Learning Repository
  https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+(Diagnostic)

- **Platform Used**: Google Colab
  https://colab.research.google.com/