An 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

Taranjeet Singh Bagga

By

Taranjeet Singh Bagga (202401100300263)

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

INTRODUCTION

Predicting disease outcomes based on genetic and clinical data is a critical task in personalized medicine and healthcare. With advancements in data collection, large datasets containing patient demographics, clinical measurements, and genetic markers are now available. The goal of this project is to build a machine learning model that can accurately predict whether a patient is likely to develop a specific disease or not, based on these features. Automating this prediction process can assist healthcare professionals in early diagnosis and treatment planning. This project simplifies the workflow by requiring only a CSV file as input, enabling quick and efficient model training.

METHODOLOGY

The approach taken in this analysis is focused on exploratory data analysis (EDA), which is an essential first step in understanding and preparing data for machine learning. The process starts by loading the dataset and inspecting its structure, including data types, shape (number of rows and columns), missing values, and summary statistics such as mean, standard deviation, and percentiles. This helps identify any issues like missing or incorrect data early on. The next step involves analyzing the target variable, diagnosis, to check how balanced the classes are—this is important because an imbalanced dataset can affect model performance. The correlation matrix is then plotted to explore relationships between numerical features and identify highly correlated pairs, which might indicate redundancy. A set of important features is visualized using both distribution plots (to see how values differ across diagnosis categories) and box plots (to compare statistical summaries like median and range across classes). Finally, a scatter plot between two selected features is used to observe whether the data shows any clear separation between the diagnosis classes. Altogether, this approach helps uncover insights, detect patterns, and guide the next steps in model building by highlighting which features are most informative.

CODE

```
import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns


# Load the dataset

df = pd.read_csv('3. Predict Disease Outcome Based on Genetic and Clinical Data.csv')


# Explore the dataset

print("Data types:\n", df.dtypes)

print("\nShape:", df.shape)

print("\nMissing values:\n", df.isnull().sum())

print("\nDescriptive statistics:\n", df.describe())


# Analyze the 'diagnosis' feature
```

```
print("\nDiagnosis distribution:\n", df['diagnosis'].value_counts())

sns.countplot(x='diagnosis', data=df)

plt.title('Distribution of Diagnosis')

plt.show()


# Correlation analysis (Corrected)

numeric_features = df.select_dtypes(include=['number'])  # Select only numeric features

plt.figure(figsize=(12, 8))

sns.heatmap(numeric_features.corr(), annot=False, cmap='coolwarm')  # Use numeric_features for correlation

plt.title('Correlation Matrix')

plt.show()


# Visualize key variables

features = ['radius_mean', 'texture_mean', 'perimeter_mean', 'area_mean', 'smoothness_mean']

for feature in features:

    plt.figure()

    sns.histplot(data=df, x=feature, hue='diagnosis', kde=True)

    plt.title(f'Distribution of {feature}')
```

```
    plt.show(


    )


    plt.figure()
    sns.boxplot(x='diagnosis', y=feature, data=df)
    plt.title(f'Box Plot of {feature} by Diagnosis')
    plt.show()


plt.figure(figsize=(8, 6))
sns.scatterplot(x='radius_mean', y='texture_mean', hue='diagnosis',
data=df)
plt.title('Radius Mean vs. Texture Mean')
plt.show()
```

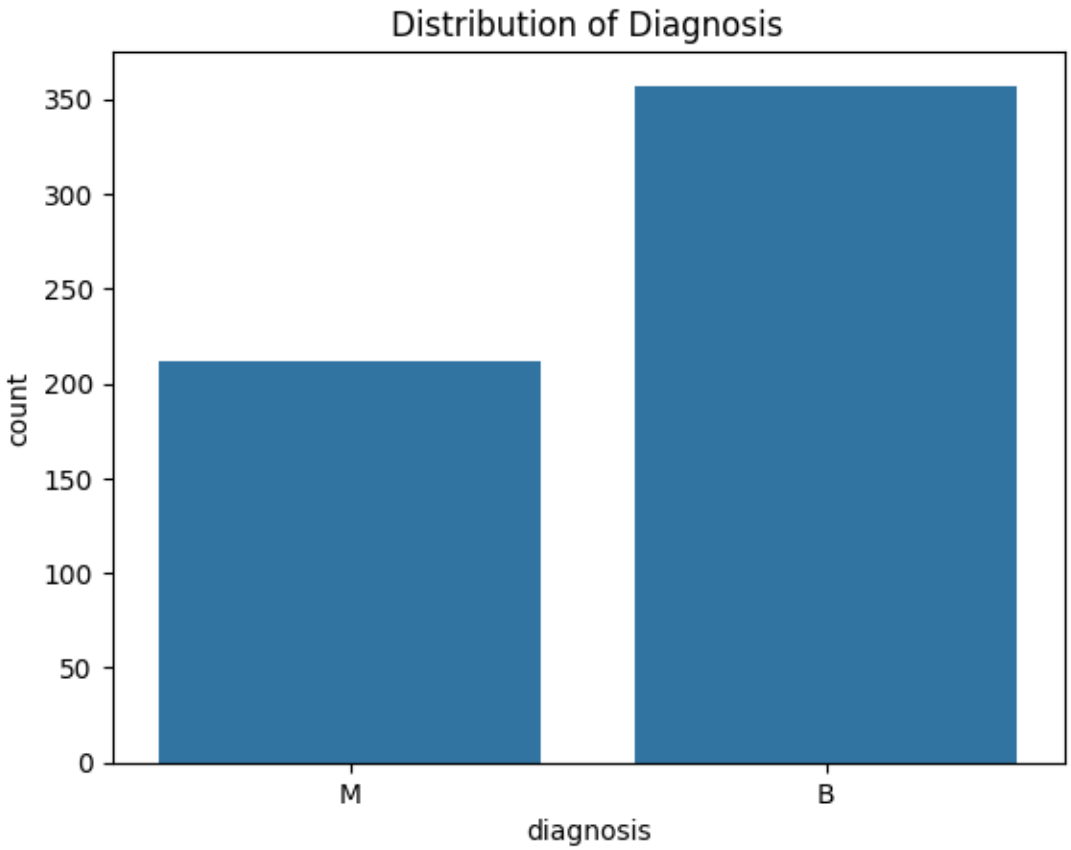
OUTPUT

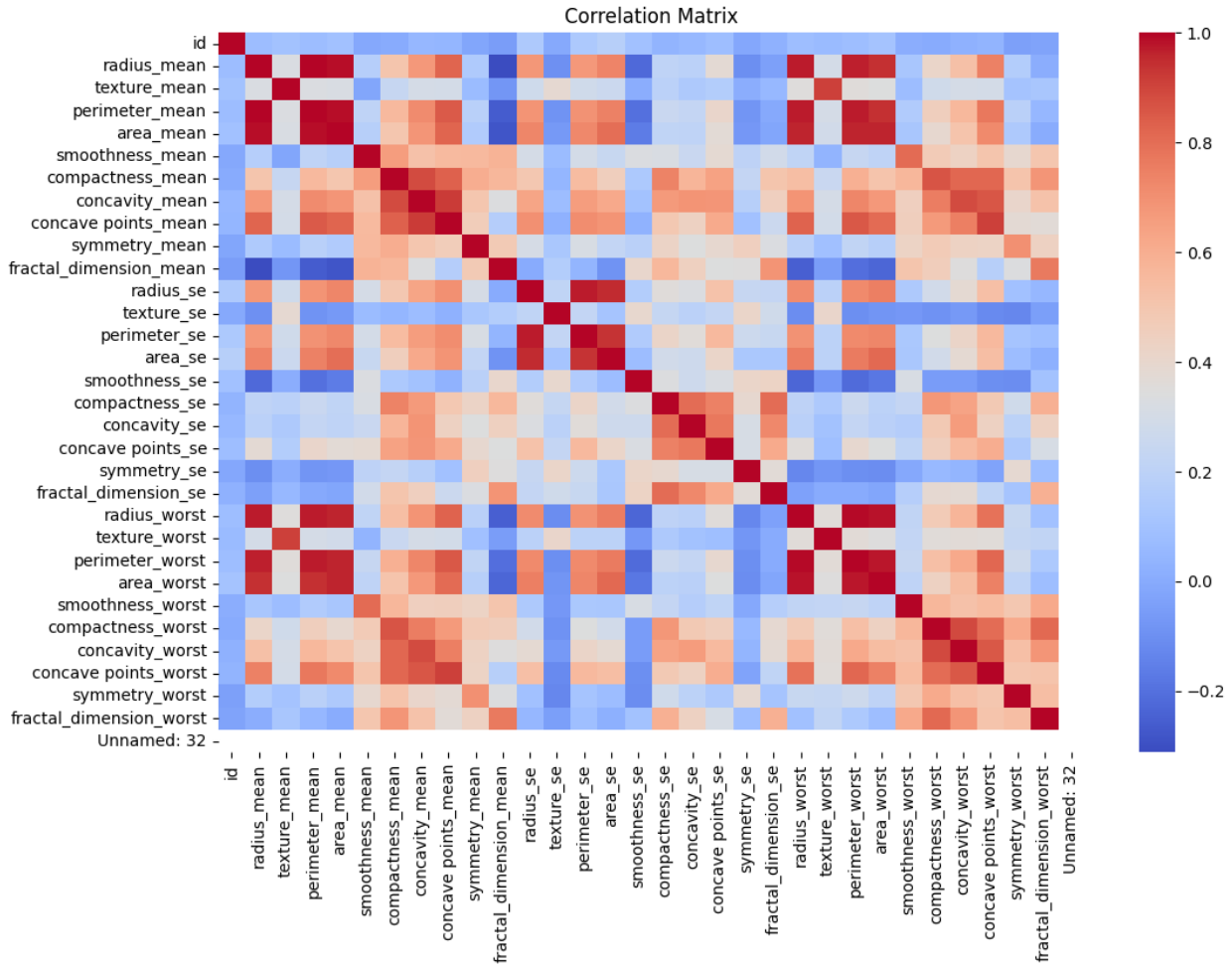
```
Data types:
 id                         int64
diagnosis                  object
radius_mean               float64
texture_mean              float64
perimeter_mean            float64
area_mean                 float64
smoothness_mean           float64
compactness_mean          float64
concavity_mean            float64
concave points_mean       float64
symmetry_mean             float64
fractal_dimension_mean    float64
radius_se                 float64
texture_se                float64
perimeter_se              float64
area_se                   float64
smoothness_se             float64
compactness_se            float64
concavity_se              float64
concave points_se         float64
symmetry_se               float64
fractal_dimension_se      float64
radius_worst              float64
texture_worst             float64
perimeter_worst           float64
area_worst                float64
smoothness_worst          float64
compactness_worst         float64
concavity_worst           float64
concave points_worst      float64
symmetry_worst            float64
fractal_dimension_worst   float64
Unnamed: 32               float64
dtype: object
```

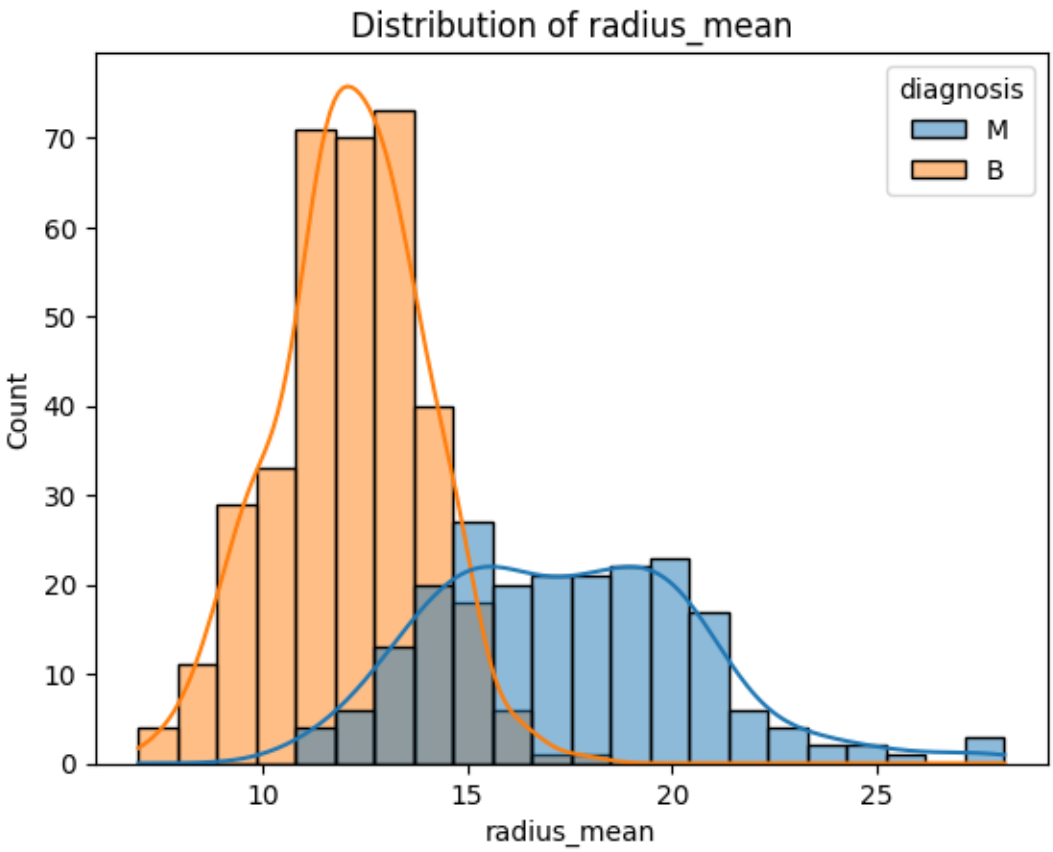
```
Shape: (569, 33)

Missing values:
 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
Unnamed: 32               569
dtype: int64
```

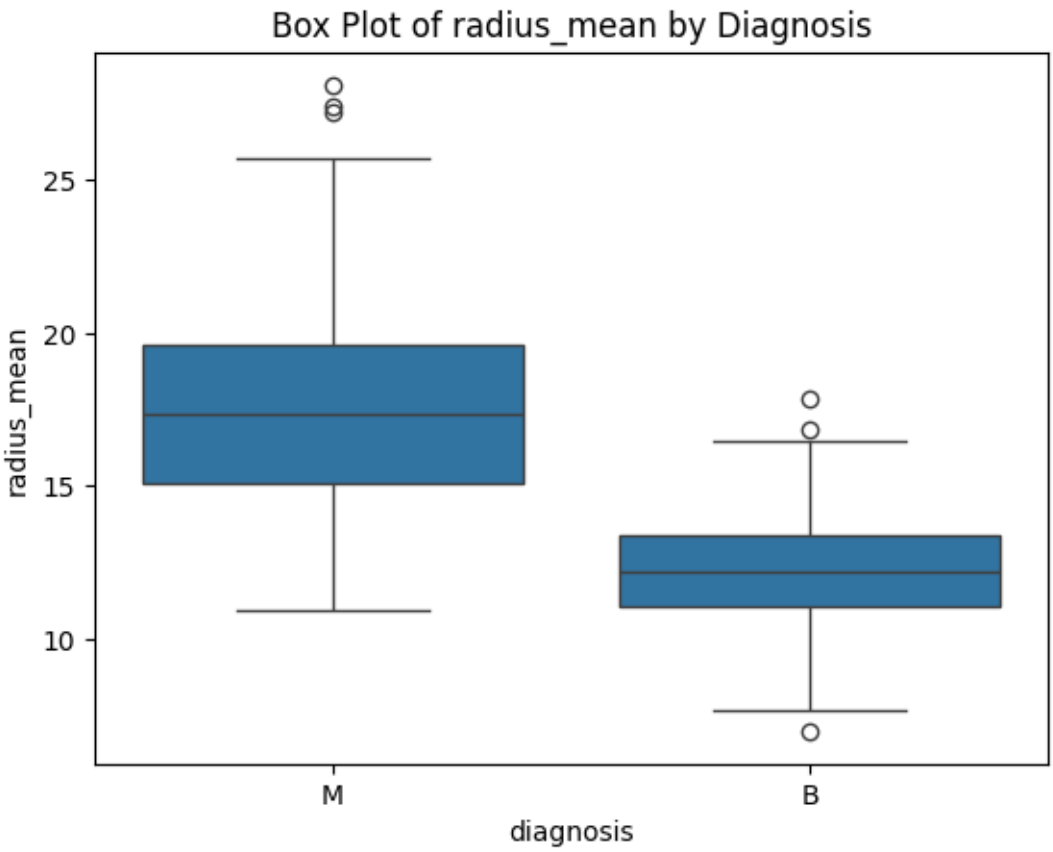
```
Descriptive statistics:
                 id  radius_mean  texture_mean  perimeter_mean    area_mean  \
count  5.690000e+02   569.000000    569.000000      569.000000   569.000000
mean   3.037183e+07    14.127292     19.289649       91.969033   654.889104
std    1.250206e+08     3.524049      4.301036       24.298981   351.914129
min    8.670000e+03     6.981000      9.710000       43.790000   143.500000
25%    8.692180e+05    11.700000     16.170000       75.170000   420.300000
50%    9.060240e+05    13.370000     18.840000       86.240000   551.100000
75%    8.813129e+06    15.780000     21.800000      104.100000   782.700000
max    9.113205e+08    28.110000     39.280000      188.500000  2501.000000

       smoothness_mean  compactness_mean  concavity_mean  concave points_mean  \
count       569.000000        569.000000      569.000000           569.000000
mean          0.096360          0.104341        0.088799             0.048919
std           0.014064          0.052813        0.079720             0.038803
min           0.052630          0.019380        0.000000             0.000000
25%           0.086370          0.064920        0.029560             0.020310
50%           0.095870          0.092630        0.061540             0.033500
75%           0.105300          0.130400        0.130700             0.074000
max           0.163400          0.345400        0.426800             0.201200

       symmetry_mean  ...  texture_worst  perimeter_worst   area_worst  \
count     569.000000  ...     569.000000       569.000000   569.000000
mean        0.181162  ...      25.677223       107.261213   880.583128
std         0.027414  ...       6.146258        33.602542   569.356993
min         0.106000  ...      12.020000        50.410000   185.200000
25%         0.161900  ...      21.080000        84.110000   515.300000
50%         0.179200  ...      25.410000        97.660000   686.500000
75%         0.195700  ...      29.720000       125.400000  1084.000000
max         0.304000  ...      49.540000       251.200000  4254.000000

       smoothness_worst  compactness_worst  concavity_worst  \
count        569.000000         569.000000       569.000000
mean           0.132369           0.254265         0.272188
std            0.022832           0.157336         0.208624
min            0.071170           0.027290         0.000000
25%            0.116600           0.147200         0.114500
50%            0.131300           0.211900         0.226700
75%            0.146000           0.339100         0.382900
max            0.222600           1.058000         1.252000

       concave points_worst  symmetry_worst  fractal_dimension_worst  \
count            569.000000      569.000000               569.000000
mean               0.114606        0.290076                 0.083946
std                0.065732        0.061867                 0.018061
min                0.000000        0.156500                 0.055040
25%                0.064930        0.250400                 0.071460
50%                0.099930        0.282200                 0.080040
75%                0.161400        0.317900                 0.092080
max                0.291000        0.663800                 0.207500
```

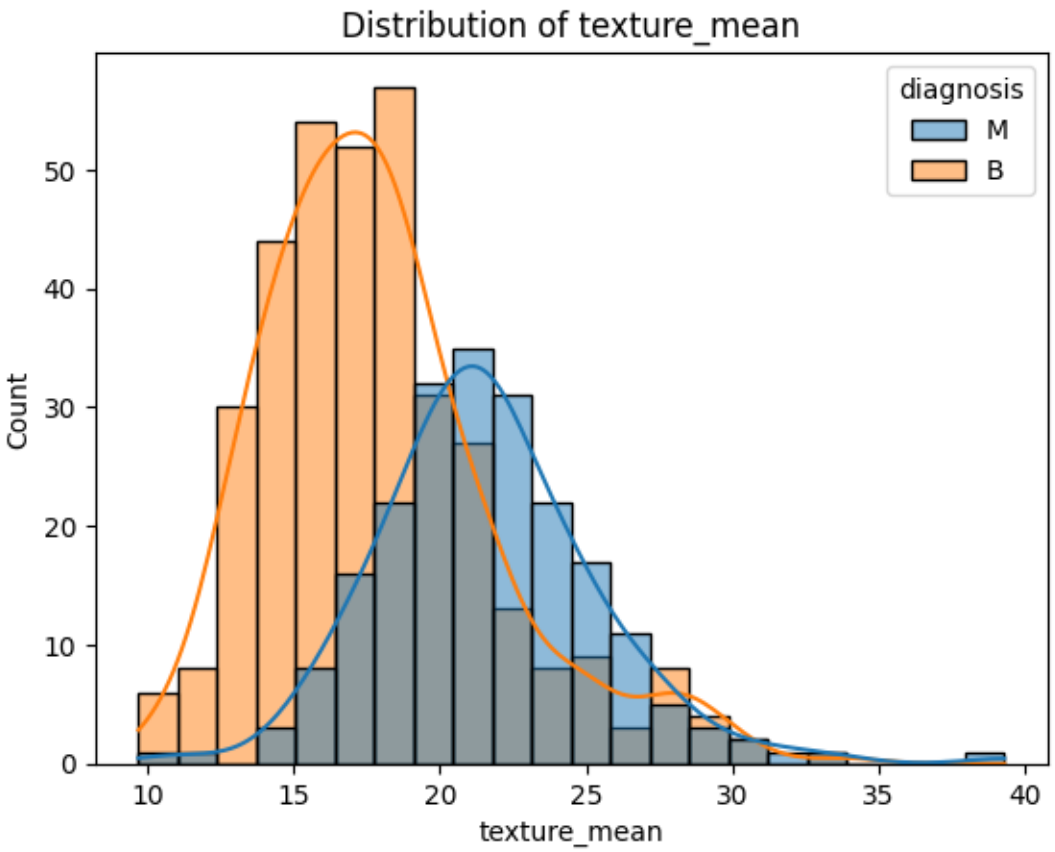
```
       Unnamed: 32
count          0.0
mean           NaN
std            NaN
min            NaN
25%            NaN
50%            NaN
75%            NaN
max            NaN

[8 rows x 32 columns]

Diagnosis distribution:
 diagnosis
B    357
M    212
Name: count, dtype: int64
```

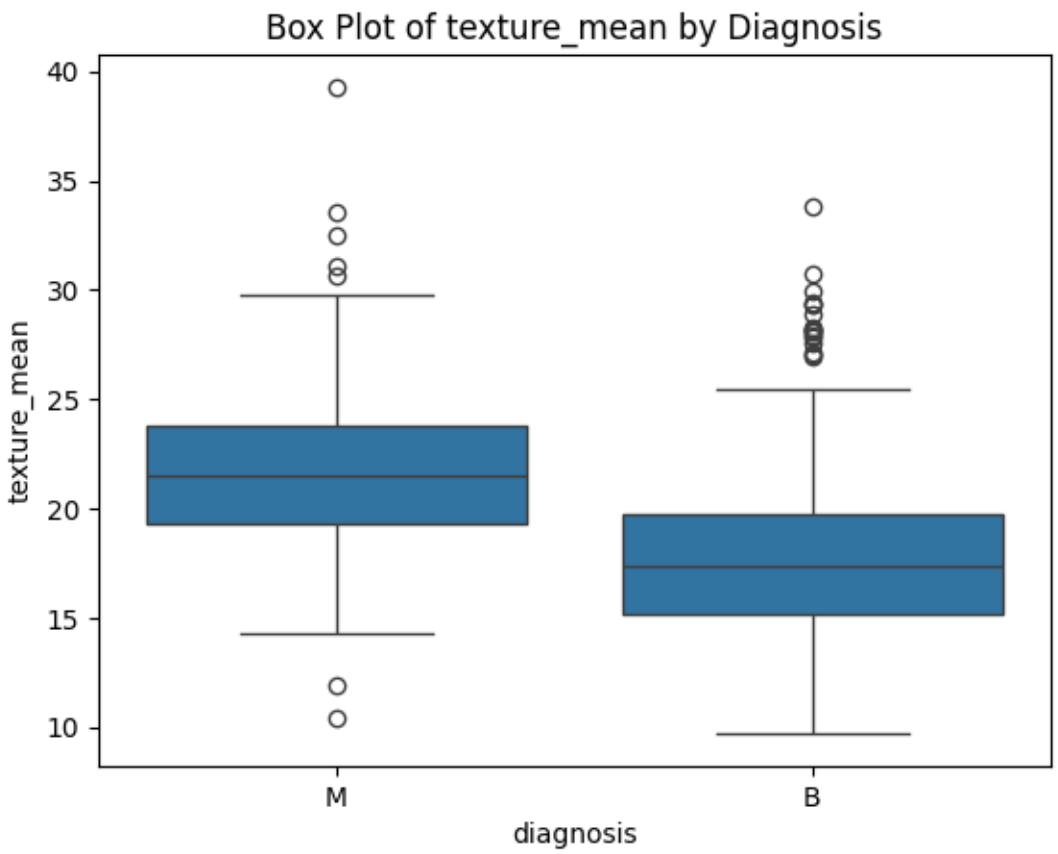


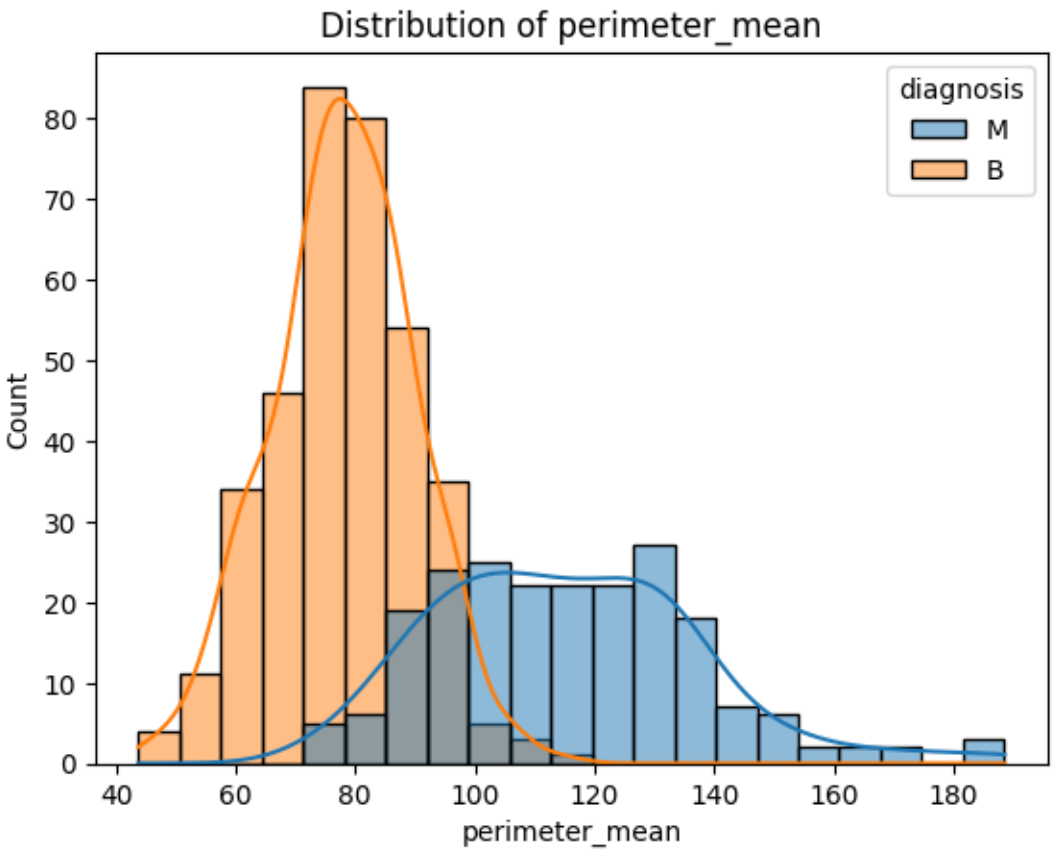

Correlation Matrix
id
radius_mean
texture_mean
perimeter_mean
area_mean
smoothness_mean
compactness_mean
concavity_mean
concave points_mean
symmetry_mean
fractal_dimension_mean
radius_se
texture_se
perimeter_se
area_se
smoothness_se
compactness_se
concavity_se
concave points_se
symmetry_se
fractal_dimension_se
radius_worst
texture_worst
perimeter_worst
area_worst
smoothness_worst
compactness_worst
concavity_worst
concave points_worst
symmetry_worst
fractal_dimension_worst
Unnamed: 32
1.0
0.8
0.6
0.4
0.2
0.0
−0.2

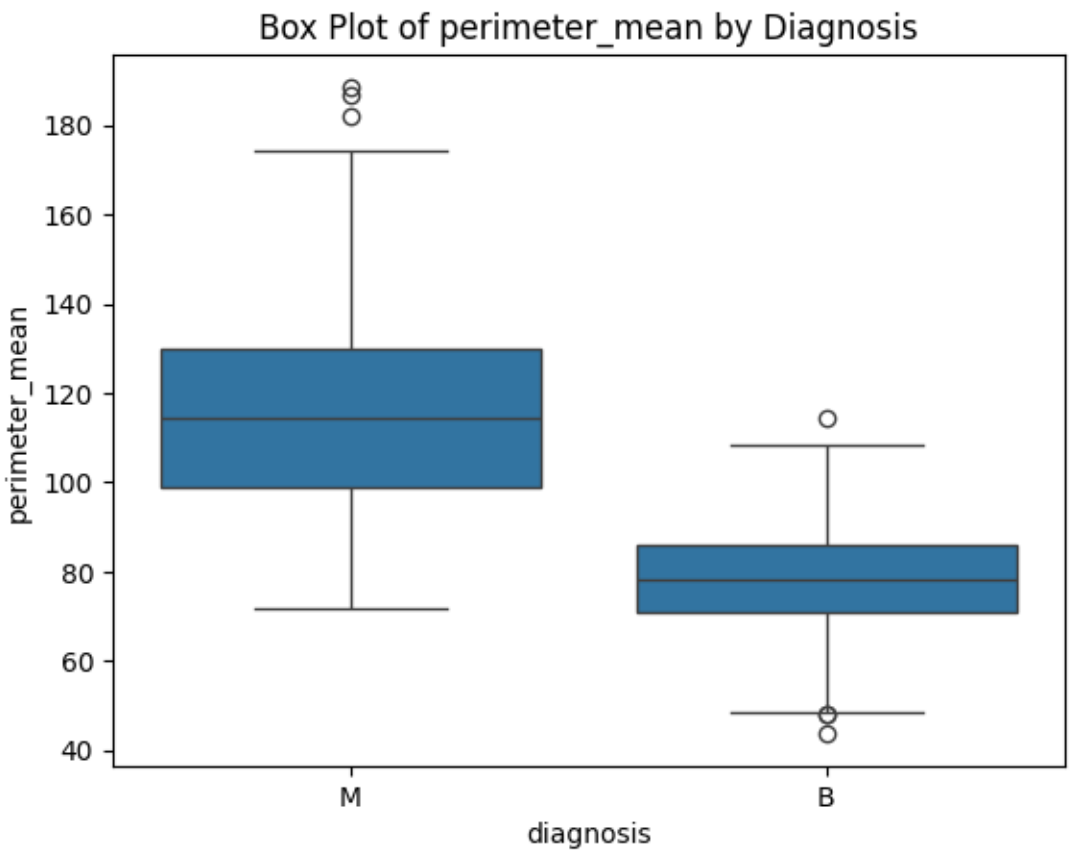

Distribution of radius_mean
diagnosis
M
B
Count
radius_mean
0
10
20
30
40
50
60
70
10
15
20
25

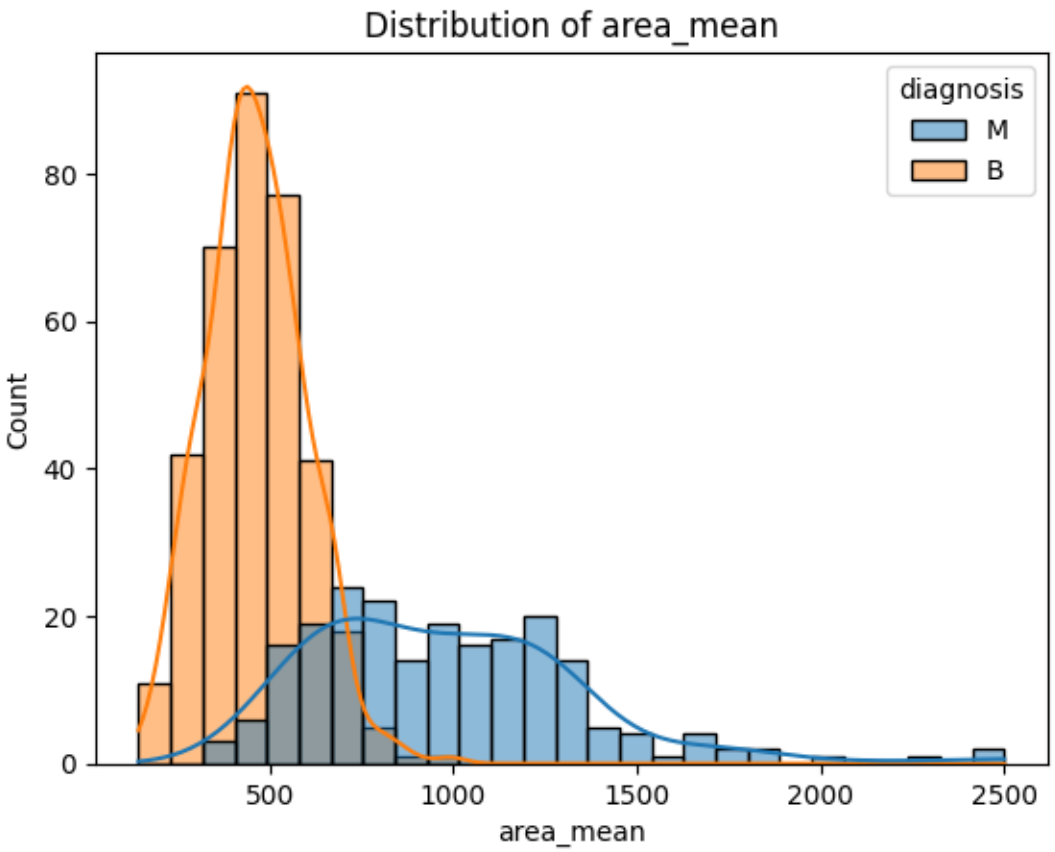

Box Plot of radius_mean by Diagnosis
radius_mean
25
20
15
10
M
B
diagnosis

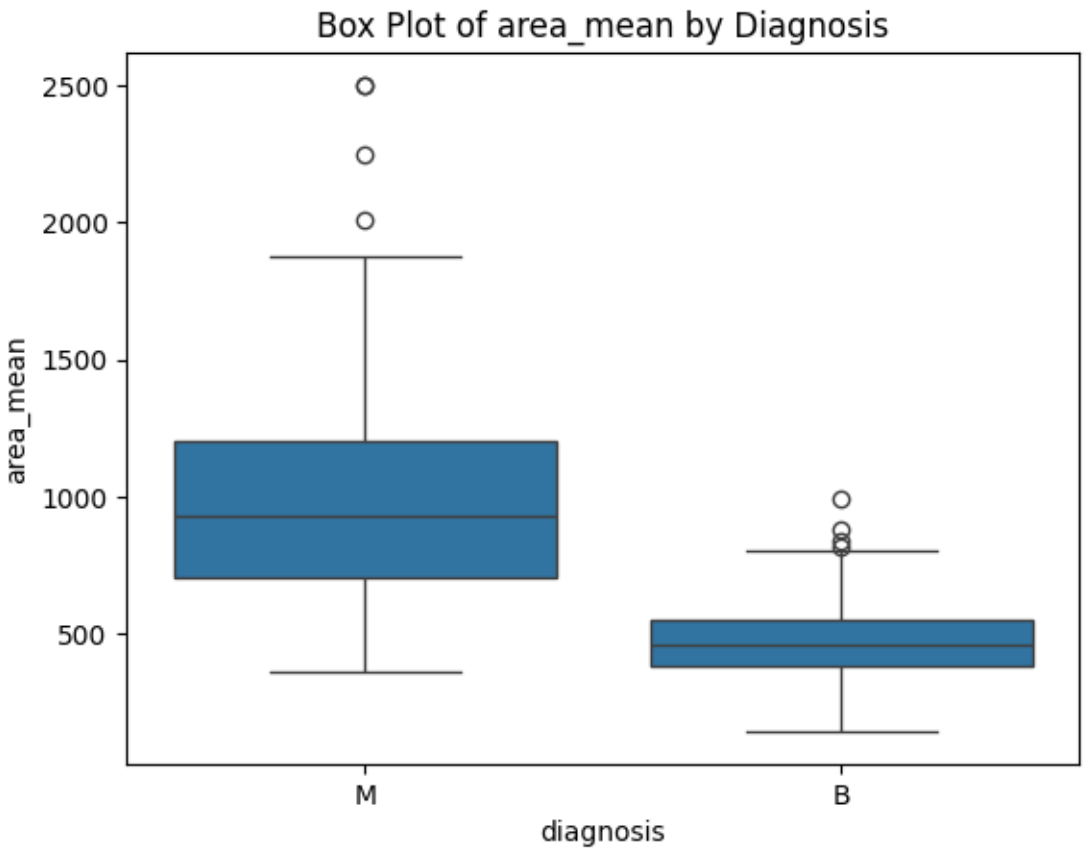

Distribution of texture_mean
diagnosis
M
B
Count
texture_mean
0
10
20
30
40
50
10
15
20
25
30
35
40

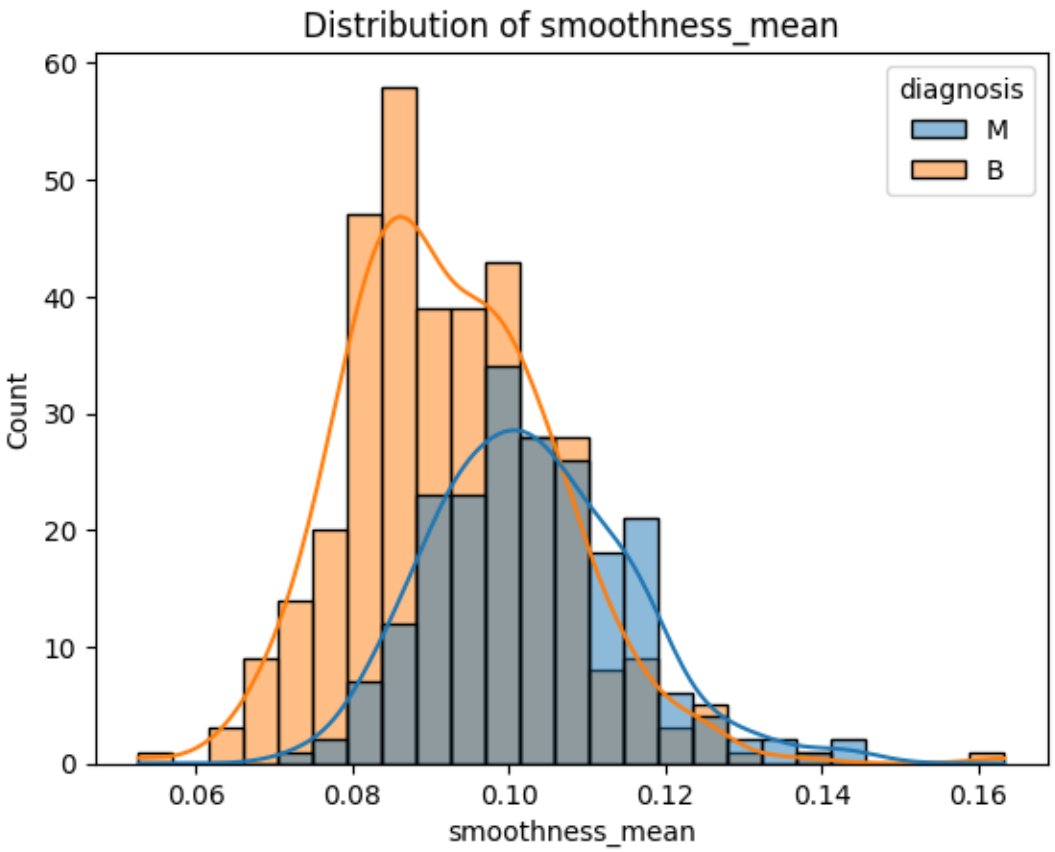

Box Plot of texture_mean by Diagnosis
texture_mean
40
35
30
25
20
15
10
M
B
diagnosis

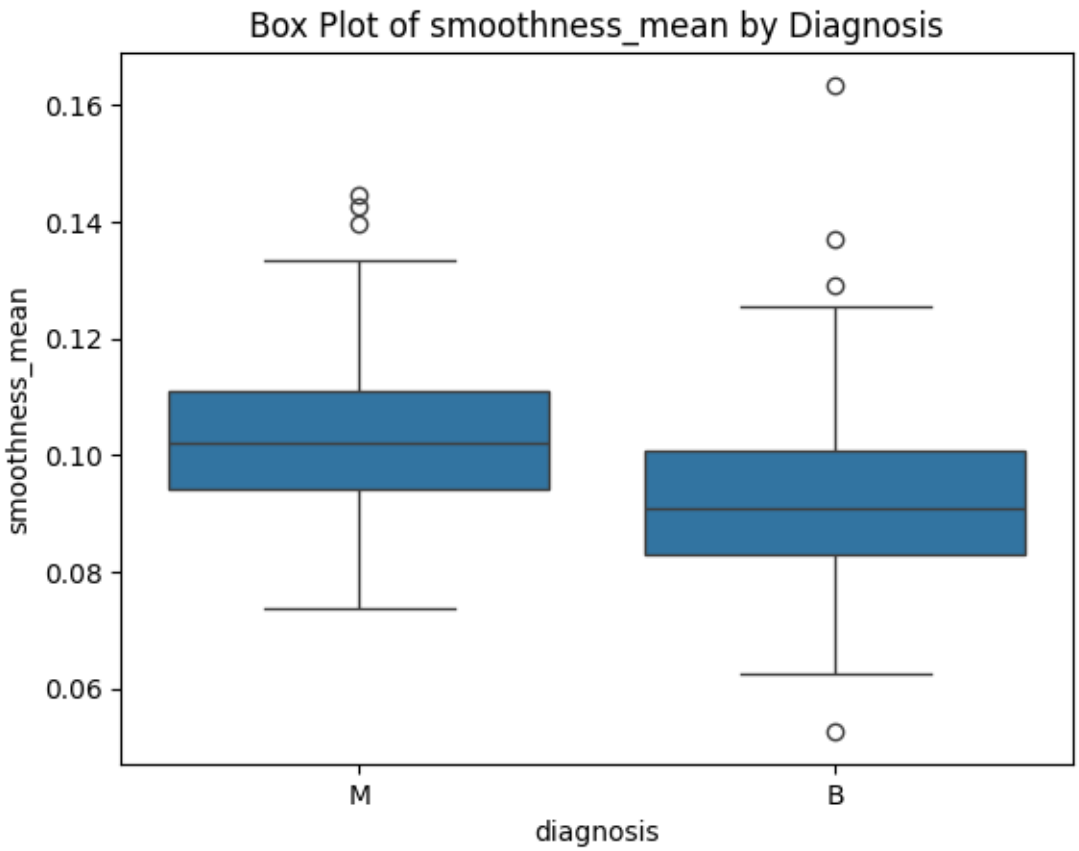

Distribution of perimeter_mean
diagnosis
M
B
Count
perimeter_mean
0
10
20
30
40
50
60
70
80
40
60
80
100
120
140
160
180

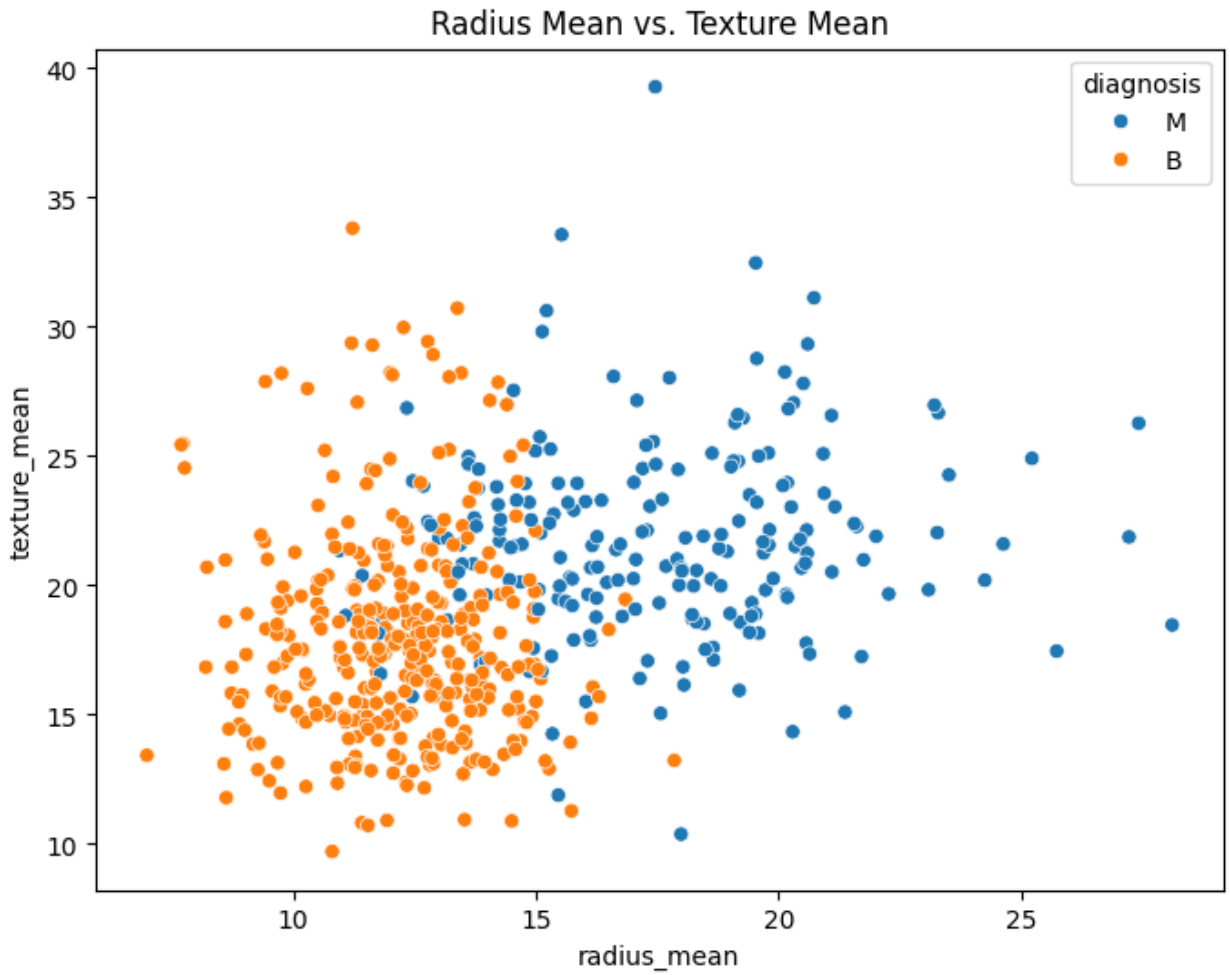

Box Plot of perimeter_mean by Diagnosis
perimeter_mean
180
160
140
120
100
80
60
40
M
B
diagnosis


Distribution of area_mean
diagnosis
M
B
Count
0
20
40
60
80
500
1000
1500
2000
2500
area_mean


Box Plot of area_mean by Diagnosis
2500
2000
1500
1000
500
area_mean
M
B
diagnosis


Distribution of smoothness_mean
diagnosis
M
B
Count
smoothness_mean
0
10
20
30
40
50
60
0.06
0.08
0.10
0.12
0.14
0.16


Box Plot of smoothness_mean by Diagnosis
smoothness_mean
0.16
0.14
0.12
0.10
0.08
0.06
M
B
diagnosis


Radius Mean vs. Texture Mean
diagnosis
M
B
texture_mean
radius_mean
40
35
30
25
20
15
10
10
15
20
25

Credits

This analysis was carried out using Python and popular data analysis and visualization libraries, including **Pandas**, **Matplotlib**, and **Seaborn**. These tools played a crucial role in data handling, statistical exploration, and the creation of informative visualizations.

- **Dataset**: The dataset titled *"Predict Disease Outcome Based on Genetic and Clinical Data"* was utilized for exploratory data analysis.
- **Tools Used**:
  - **Pandas**: For data loading, cleaning, and basic exploration.
  - **Matplotlib & Seaborn**: For generating plots and visual insights.
- **Analysis & Visualization**: All data exploration, correlation analysis, and visualizations were performed using custom Python scripts developed as part of this project.

Special thanks to open-source contributors and the data science community for maintaining such powerful tools and resources that support meaningful data-driven insights.