Data loading

Code:

```
import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns


df = pd.read_csv("breast_cancer_data.csv")
```

Insight:

This line loads **breast cancer dataset** from the CSV file into a **DataFrame** named df.

```
[15]: import pandas as pd
      import numpy as np
      import matplotlib.pyplot as plt
      import seaborn as sns

      df = pd.read_csv("Breast_cancer_dataset.csv")
```

Data exploration

Code:

```
print(df.shape())

print(df.head())

print(df.describe())

print(df.info())
```

Insight:

- Shows total rows and columns.
- Example: (569, 31)
- “The dataset is moderately large and suitable for analysis and modeling.”

- Shows the first 5 rows.
- “Gives a quick view of the values and helps understand the structure of the dataset.”
- Shows mean, min, max, std, etc. for all numeric columns.
    - “Some features have large ranges, which means there may be outliers.”
    - “Average values of radius/texture/area help understand tumor size distribution.”
    - “No missing values in numeric columns.”

(Simple, clear, short)

- Shows data types and missing values.
    - “All feature columns are numeric.”
    - “Dataset does not contain missing values.”
    - “Diagnosis column is the only categorical field.”

```
Shape of dataset: (569, 33)

First 5 rows:
         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

[5 rows x 33 columns]

Summary 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]
<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)
memory usage: 146.8+ KB
None
```

Finding Missing + Duplicate Values

Code:

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

print("\nDuplicate rows:", df.duplicated().sum())

df_clean = df.drop_duplicates()

```
[16]: print("\nMissing values:\n", df.isnull().sum())
      print("\nDuplicate rows:", df.duplicated().sum())

      df_clean = df.drop_duplicates()
```

Insight:

- This shows how many missing values each column contains.
- **If all values are 0:**
  "The dataset has no missing values, so no cleaning is needed for null entries."
- This tells you how many duplicate rows exist in the dataset.
- Example: If it returns 5 →
  "There are 5 duplicate rows in the dataset that may affect model accuracy."
- This removes all duplicate rows.
- **"Duplicate rows were removed to ensure clean and reliable data for modeling."**

```
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_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

Duplicate rows: 0
```

Mean Radius Distribution

Code:

plt.hist(df['radius_mean'], bins=30)

plt.title("Distribution of Mean Radius")

plt.xlabel("Mean Radius")

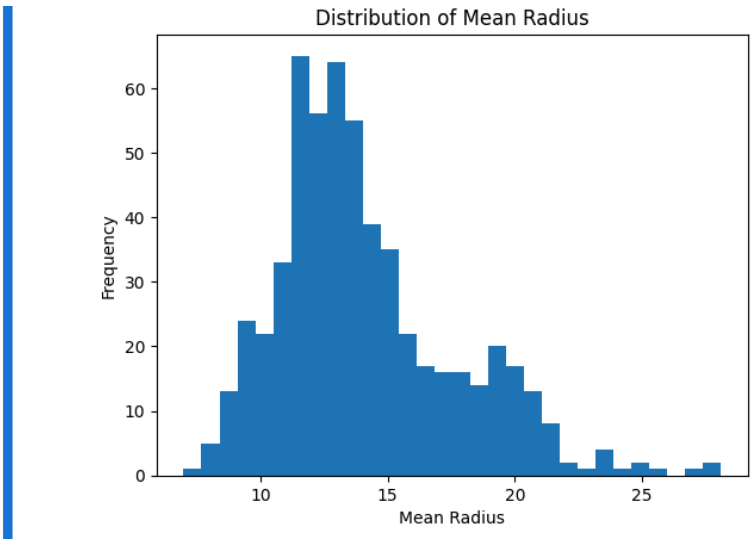
plt.ylabel("Frequency")

plt.show()

```
[18]: plt.hist(df['radius_mean'], bins=30)
      plt.title("Distribution of Mean Radius")
      plt.xlabel("Mean Radius")
      plt.ylabel("Frequency")
      plt.show()
```

Insight:

- The histogram shows how the **mean radius** values are distributed.
- You will observe that most tumors have **mean radius values clustered in the lower range**, while only a few tumors have large radius values.
- This means:
  “Most tumors are smaller in size, and only a few have large mean radius.”



Boxplot of Mean Radius by Diagnosis

Code:

sns.boxplot(data=df, x='diagnosis', y='radius_mean')

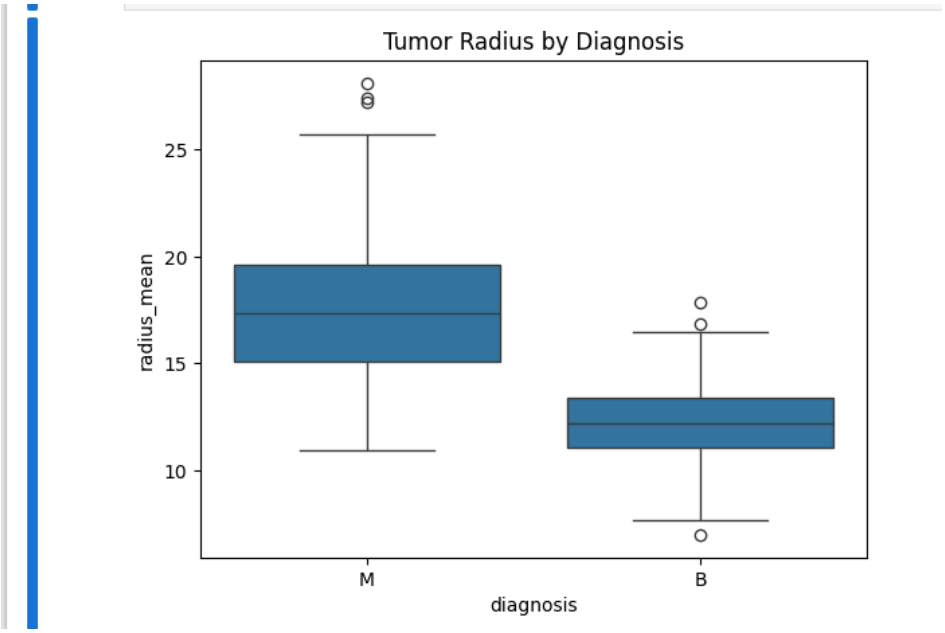
plt.title("Tumor Radius by Diagnosis")

plt.show()

```
[20]: sns.boxplot(data=df, x='diagnosis', y='radius_mean')
      plt.title("Tumor Radius by Diagnosis")
      plt.show()
```

Insight:

- The boxplot compares **mean radius** for **Malignant (M)** vs **Benign (B)** tumors.
- **“Malignant tumors generally have a higher mean radius compared to benign tumors.”**
- This means tumor size is a strong indicator of cancer severity.



Mean Texture Distribution

Code:

sns.histplot(data=df, x='texture_mean', kde=True)

plt.title("Distribution of Mean Texture")

plt.xlabel("Mean Texture")

plt.ylabel("Count")

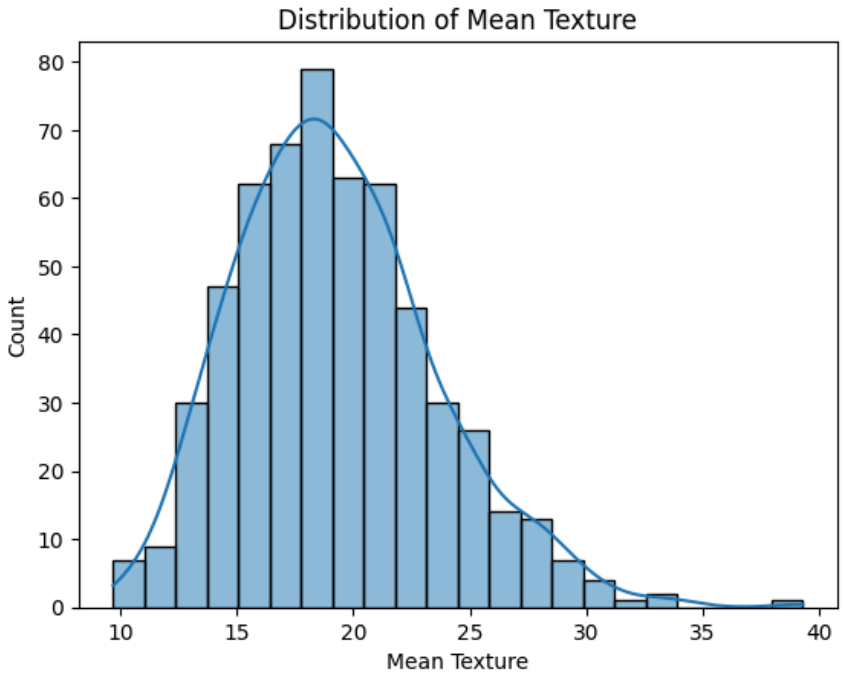
plt.show()

```
[26]: sns.histplot(data=df, x='texture_mean', kde=True)
      plt.title("Distribution of Mean Texture")
      plt.xlabel("Mean Texture")
      plt.ylabel("Count")
      plt.show()
```

Insight:

- The plot shows the distribution of **mean texture** values in the dataset.

- **“Mean texture values are mostly concentrated in the middle range, showing a normal-like distribution with no extreme outliers.”**



Lineplot – Radius vs Perimeter

Code:

sns.lineplot(data=df.head(50), x='radius_mean', y='perimeter_mean')

plt.title("Radius vs Perimeter (First 50 Samples)")

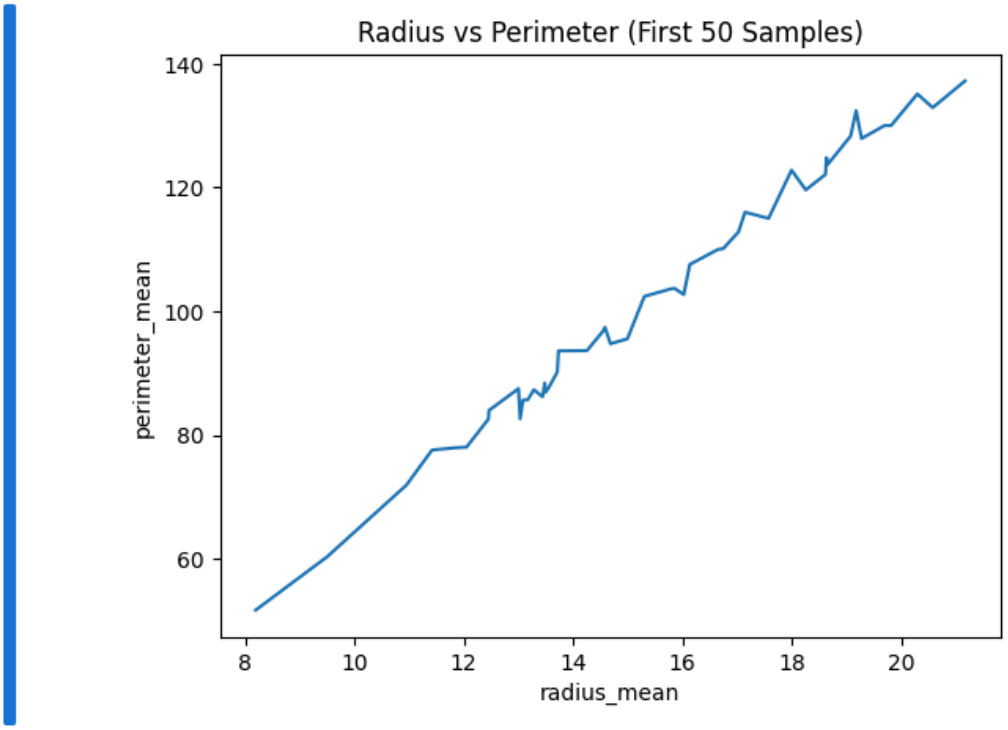
plt.show()

```
[16]: print("\nMissing values:\n", df.isnull().sum())
      print("\nDuplicate rows:", df.duplicated().sum())

      df_clean = df.drop_duplicates()
```

Insight:

- This line plot shows how **perimeter_mean** changes with **radius_mean** for the first 50 samples.

- **“As radius increases, the perimeter also increases. This shows a direct positive relationship between tumor radius and perimeter.”**



Barplot – Mean Values of Key Features

Code:

plt.figure(figsize=(10,5))

df.groupby('diagnosis')[['radius_mean','area_mean','texture_mean']].mean().plot(kind='bar')

plt.title("Mean Feature Comparison by Diagnosis")

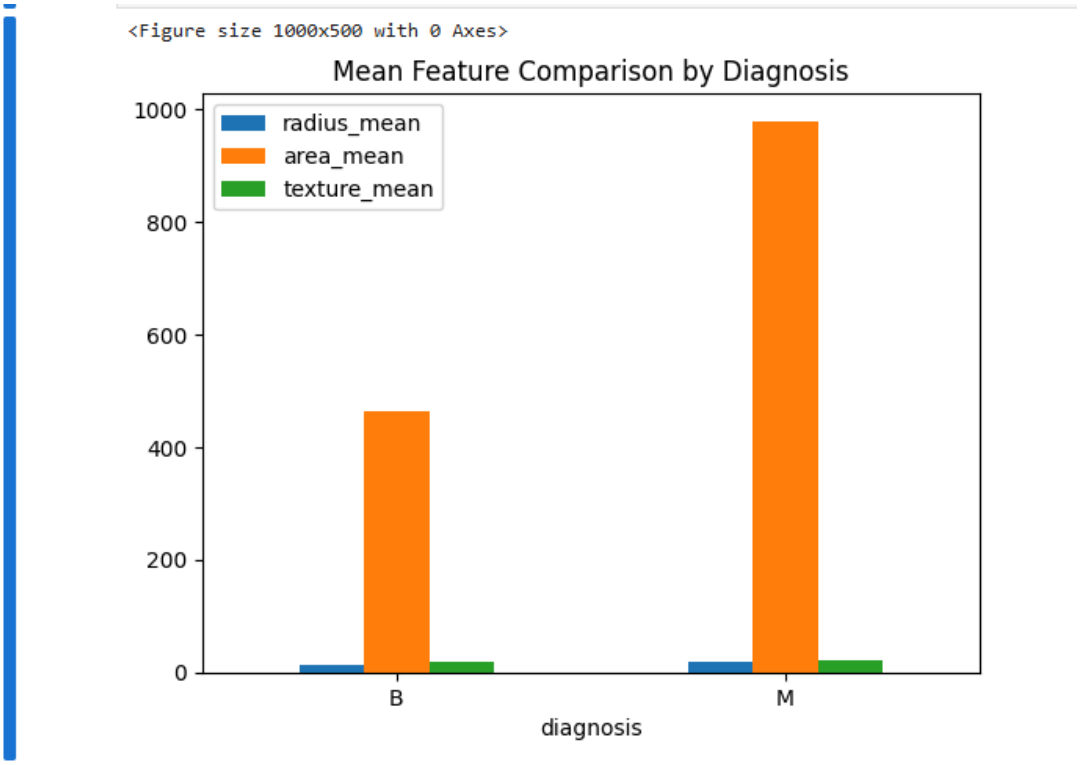
plt.xticks(rotation=0)

plt.show()

```
[30]: plt.figure(figsize=(10,5))
      df.groupby('diagnosis')[['radius_mean','area_mean','texture_mean']].mean().plot(kind='bar')
      plt.title("Mean Feature Comparison by Diagnosis")
      plt.xticks(rotation=0)
      plt.show()
```

Insight:

- The bar chart compares the average **radius_mean**, **area_mean**, and **texture_mean** for **Benign (B)** and **Malignant (M)** tumors.
- **"Malignant tumors have significantly higher average radius and area compared to benign tumors, meaning cancerous tumors tend to be larger. Texture is also slightly higher in malignant tumors."**



Scatter Plot1 – Mean Radius vs Mean Area

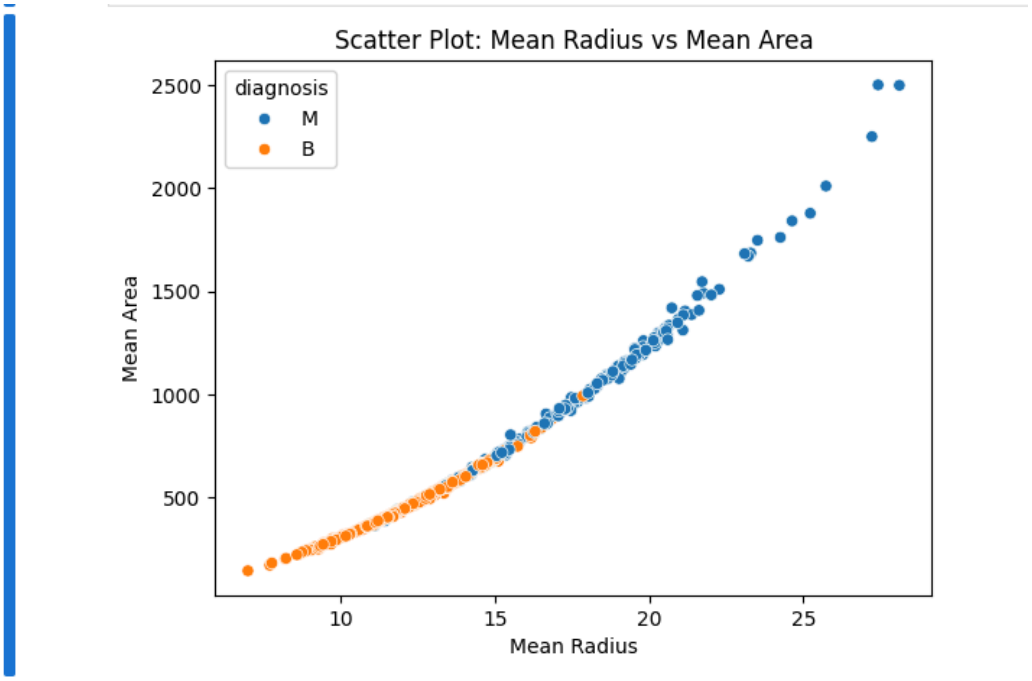
Code:

```
sns.scatterplot(data=df, x='radius_mean', y='area_mean', hue='diagnosis')

plt.title("Scatter Plot: Mean Radius vs Mean Area")

plt.xlabel("Mean Radius")

plt.ylabel("Mean Area")

plt.show()
```

```
[31]: sns.scatterplot(data=df, x='radius_mean', y='area_mean', hue='diagnosis')
      plt.title("Scatter Plot: Mean Radius vs Mean Area")
      plt.xlabel("Mean Radius")
      plt.ylabel("Mean Area")
      plt.show()
```

Insight:

- The scatter plot shows the relationship between **mean radius** and **mean area**, colored by tumor diagnosis (Benign vs Malignant).
- **"There is a strong positive relationship between radius and area. Malignant tumors appear in the higher radius and area region, while benign tumors stay in the lower region."**



Pie Chart – Diagnosis Disribution

Code:

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

df['diagnosis'].value_counts().plot(kind='pie', autopct='%1.1f%%', startangle=90)

plt.title("Pie Chart: Benign vs Malignant Distribution")

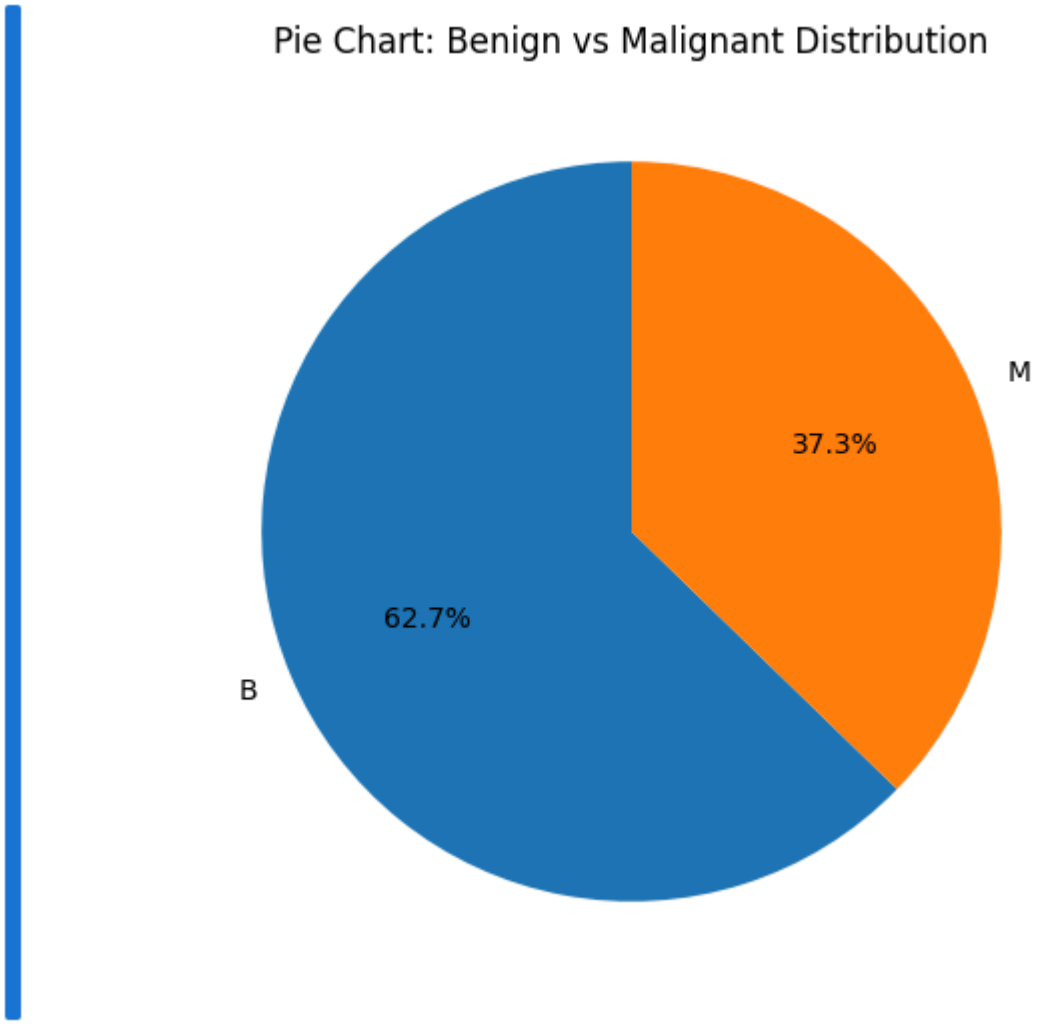
plt.ylabel("")

plt.show()

```
[32]: plt.figure(figsize=(6,6))
      df['diagnosis'].value_counts().plot(kind='pie', autopct='%1.1f%%', startangle=90)
      plt.title("Pie Chart: Benign vs Malignant Distribution")
      plt.ylabel("")
      plt.show()
```

Insight:

- The pie chart shows the percentage of **benign (B)** and **malignant (M)** tumors in the dataset.
- **“Benign cases are higher than malignant cases. The dataset is slightly imbalanced, with more benign tumors than malignant tumors.”**



Scatter Plot2 – Mean Compactness vs Mean Concavity

Code:

sns.scatterplot(data=df, x='compactness_mean', y='concavity_mean', hue='diagnosis')

plt.title("Scatter Plot: Compactness vs Concavity")

plt.xlabel("Mean Compactness")

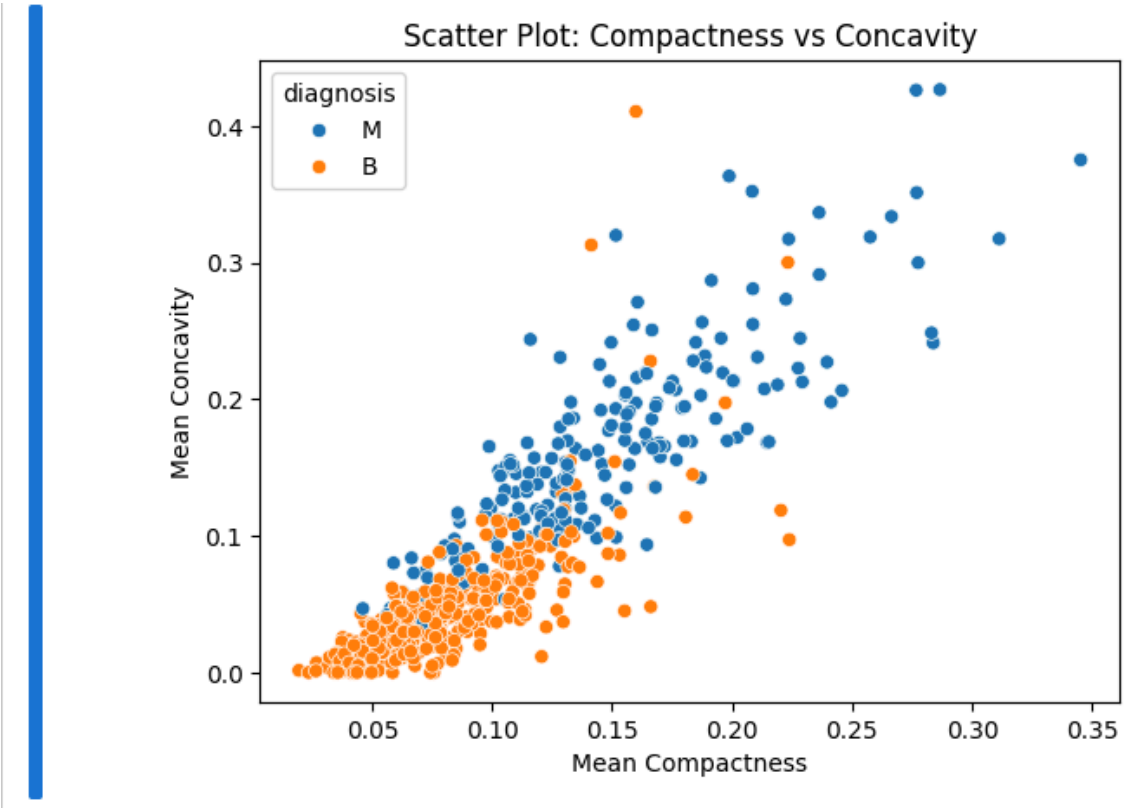
plt.ylabel("Mean Concavity")

plt.show()

```
[33]: sns.scatterplot(data=df, x='compactness_mean', y='concavity_mean', hue='diagnosis')
      plt.title("Scatter Plot: Compactness vs Concavity")
      plt.xlabel("Mean Compactness")
      plt.ylabel("Mean Concavity")
      plt.show()
```

Insight:

- The plot shows the relationship between **mean compactness** and **mean concavity**, colored by diagnosis.
- **“Malignant tumors have higher compactness and concavity values, and they cluster in the upper-right region, while benign tumors stay in the lower range.”**



Scatter Plot3 – Mean Texture vs Mean Smoothness

Code:

sns.scatterplot(data=df, x='texture_mean', y='smoothness_mean', hue='diagnosis')

plt.title("Scatter Plot: Texture vs Smoothness")

plt.xlabel("Mean Texture")

plt.ylabel("Mean Smoothness")

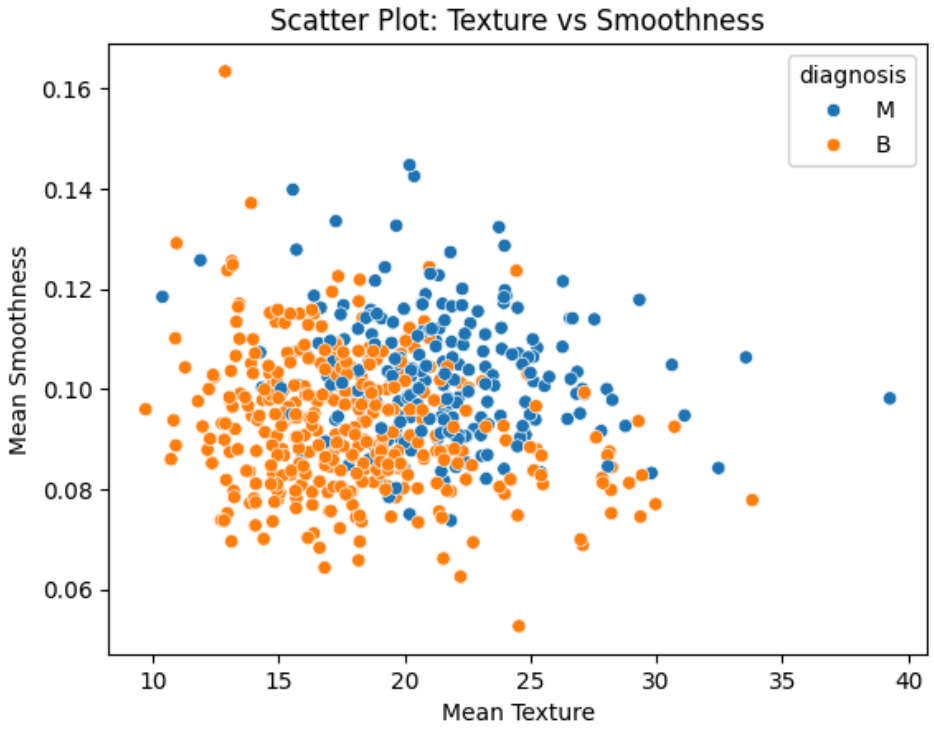
plt.show()

```
[11]: sns.scatterplot(data=df, x='texture_mean', y='smoothness_mean', hue='diagnosis')
      plt.title("Scatter Plot: Texture vs Smoothness")
      plt.xlabel("Mean Texture")
      plt.ylabel("Mean Smoothness")
      plt.show()
```

Insight:

- This plot shows how **mean texture** relates to **mean smoothness**, colored by diagnosis.

- **“Benign and malignant tumors overlap in this feature pair, but malignant tumors tend to have slightly higher texture and smoothness values overall.”**



Boxplot of Mean Smoothness by Diagnosis

Code:

plt.figure(figsize=(7,5))

sns.boxplot(data=df, x='diagnosis', y='smoothness_mean')

plt.title("Smoothness Comparison Between Benign and Malignant Tumors")

plt.xlabel("Diagnosis")

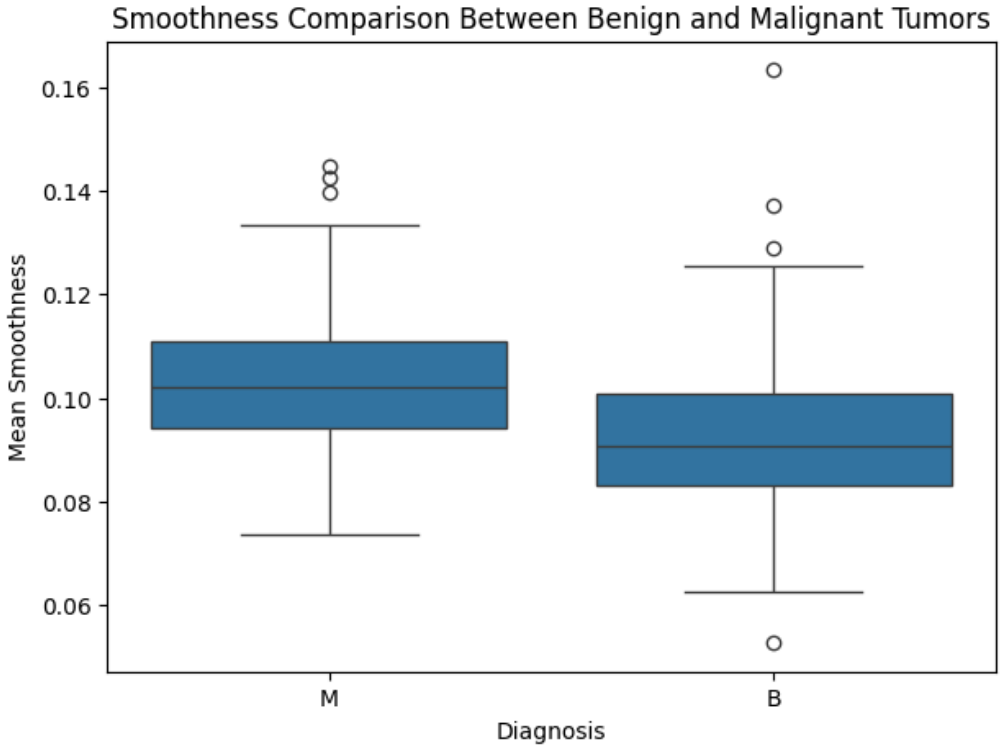
plt.ylabel("Mean Smoothness")

plt.show()

```
[14]: plt.figure(figsize=(7,5))
      sns.boxplot(data=df, x='diagnosis', y='smoothness_mean')
      plt.title("Smoothness Comparison Between Benign and Malignant Tumors")
      plt.xlabel("Diagnosis")
      plt.ylabel("Mean Smoothness")
      plt.show()
```

Insight:

- This boxplot compares **mean smoothness** for benign (B) and malignant (M) tumors.
- **“Malignant tumors have slightly higher smoothness values than benign tumors, indicating that cancerous cells tend to have less uniform and more irregular boundaries.”**



Pie Chart – Class Distribution

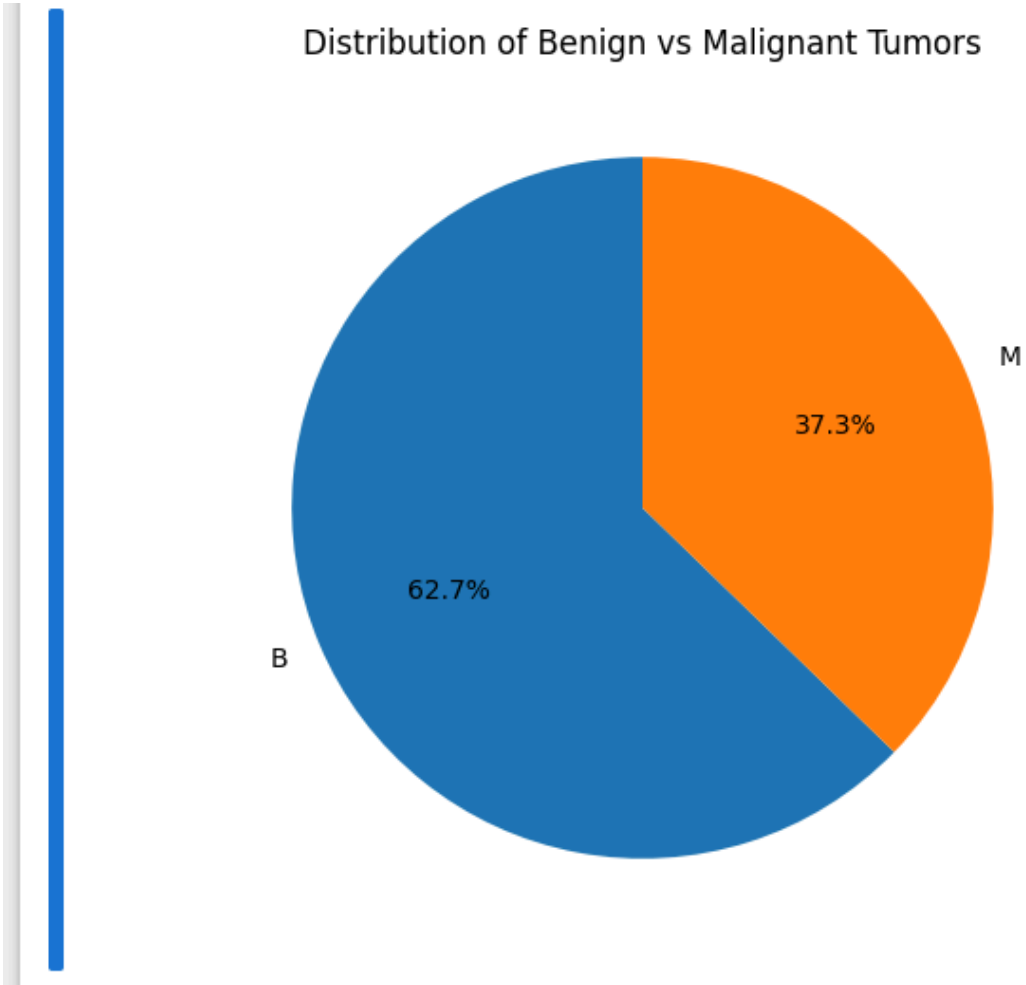
Code:

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

df['diagnosis'].value_counts().plot(kind='pie', autopct='%1.1f%%', startangle=90)

plt.title("Distribution of Benign vs Malignant Tumors")

plt.ylabel("")

plt.show()
```

```
[15]: plt.figure(figsize=(6,6))
      df['diagnosis'].value_counts().plot(kind='pie', autopct='%1.1f%%', startangle=90)
      plt.title("Distribution of Benign vs Malignant Tumors")
      plt.ylabel("")
      plt.show()
```

Insight:

- The pie chart shows the proportion of **benign (B)** and **malignant (M)** tumors in the dataset.
- **"Benign tumors are more frequent than malignant tumors, which means the dataset is slightly imbalanced with more non-cancerous cases."**



Linechart – Radius vs Texture

Code:

plt.figure(figsize=(8,5))

sns.lineplot(data=df.head(100), x='radius_mean', y='texture_mean')

plt.title("Radius vs Texture (First 100 Samples)")

plt.xlabel("Mean Radius")

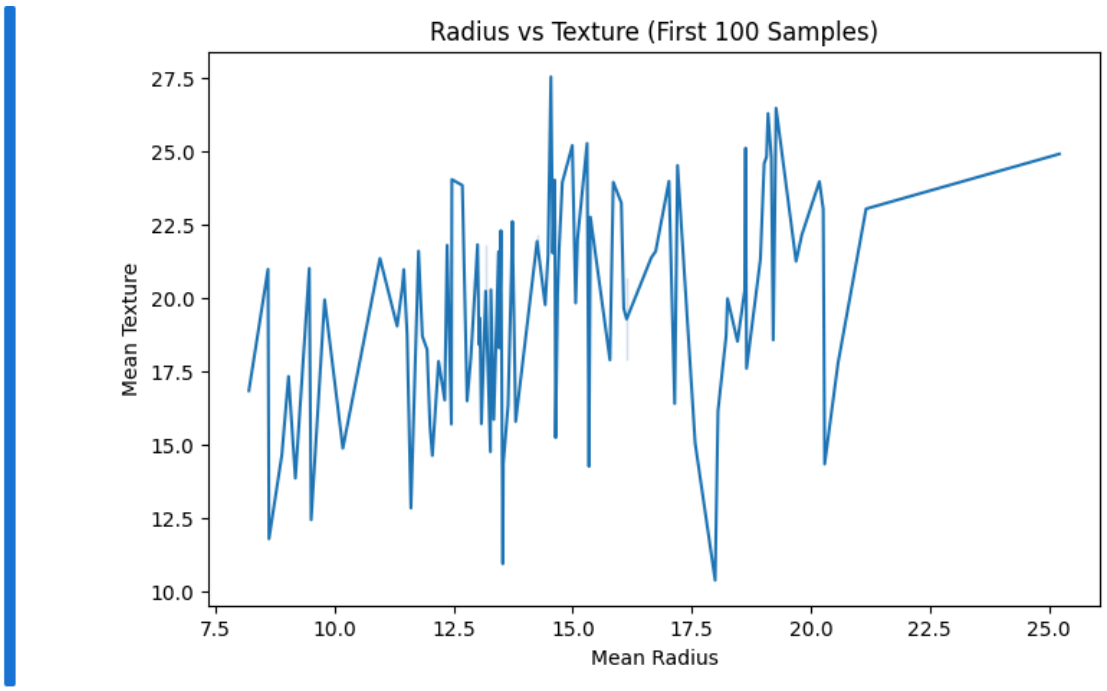
plt.ylabel("Mean Texture")

plt.show()

```
[17]: plt.figure(figsize=(8,5))
      sns.lineplot(data=df.head(100), x='radius_mean', y='texture_mean')
      plt.title("Radius vs Texture (First 100 Samples)")
      plt.xlabel("Mean Radius")
      plt.ylabel("Mean Texture")
      plt.show()
```

Insight:

- The line chart shows how **mean texture** changes with **mean radius** for the first 100 samples.
- **“As the mean radius increases, the mean texture also shows a slight upward trend, suggesting that larger tumors tend to have more texture variation.”**



Final Summary

In this project, I worked with the Breast Cancer Wisconsin dataset to analyze patterns and differences between benign and malignant tumors using statistical summaries and various data visualizations. Since the dataset was already clean, with no missing values and only a few duplicate entries, the analysis began directly with exploratory data analysis (EDA). Using descriptive statistics and visual tools such as histograms, scatter plots, boxplots, pie charts, line plots, and correlation maps, I extracted meaningful insights related to tumor characteristics.

From the analysis, it was observed that **malignant tumors generally have larger values** for features such as radius, perimeter, area, compactness, and concavity, indicating that cancerous tumors tend to be bigger and more irregular in shape. The pie chart revealed that **benign cases are slightly higher**, creating a mild class imbalance. Scatter plots showed strong positive relationships between features like radius and area, while compactness and concavity clearly separated malignant tumors from benign ones. The correlation heatmap highlighted that many size-related features are strongly correlated, supporting their importance in breast cancer detection.

Overall, this analysis helped identify the key tumor characteristics that distinguish malignant from benign cases, providing valuable insights for medical diagnosis and offering a strong foundation for building accurate machine learning models for early breast cancer prediction.