In [1]:

```
import pandas as pd
df = pd.read_csv('Breast_cancer_data.csv')
df
```

Out[1]:

	mean_radius	mean_texture	mean_perimeter	mean_area	mean_smoothness	diagnosis
0	17.99	10.38	122.80	1001.0	0.11840	0
1	20.57	17.77	132.90	1326.0	0.08474	0
2	19.69	21.25	130.00	1203.0	0.10960	0
3	11.42	20.38	77.58	386.1	0.14250	0
4	20.29	14.34	135.10	1297.0	0.10030	0
...	...	...	...	...	...	...
564	21.56	22.39	142.00	1479.0	0.11100	0
565	20.13	28.25	131.20	1261.0	0.09780	0
566	16.60	28.08	108.30	858.1	0.08455	0
567	20.60	29.33	140.10	1265.0	0.11780	0
568	7.76	24.54	47.92	181.0	0.05263	1

569 rows × 6 columns

In [2]:

```
df['diagnosis'].unique()
```

Out[2]:

```
array([0, 1], dtype=int64)
```

In [4]:

```
df.isnull().sum()
```

Out[4]:

```
mean_radius        0
mean_texture       0
mean_perimeter     0
mean_area          0
mean_smoothness    0
diagnosis          0
dtype: int64
```

In [12]:
```
### split data set into dependent and independent features
X = df.iloc[:,:-1]
y = df.iloc[:,-1]
print(x)
print(y)
```

```
     mean_radius  mean_texture  mean_perimeter  mean_area  mean_smoothness
0          17.99         10.38          122.80     1001.0          0.11840
1          20.57         17.77          132.90     1326.0          0.08474
2          19.69         21.25          130.00     1203.0          0.10960
3          11.42         20.38           77.58      386.1          0.14250
4          20.29         14.34          135.10     1297.0          0.10030
..           ...           ...             ...        ...              ...
564        21.56         22.39          142.00     1479.0          0.11100
565        20.13         28.25          131.20     1261.0          0.09780
566        16.60         28.08          108.30      858.1          0.08455
567        20.60         29.33          140.10     1265.0          0.11780
568         7.76         24.54           47.92      181.0          0.05263

[569 rows x 5 columns]
0      0
1      0
2      0
3      0
4      0
      ..
564    0
565    0
566    0
567    0
568    1
Name: diagnosis, Length: 569, dtype: int64
```

In [13]:
```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.33, rand
```

In [19]:
```
### to find best combination of parameters ,cv = cross validation
```

In [15]:
```
from sklearn.linear_model import LogisticRegression
Classifier = LogisticRegression()
```

In [18]:
```
from sklearn.model_selection import GridSearchCV
parameter = {'penalty':['l1','l2','elasticnet'],'C':[1,2,3,4,5,6,10,20,30,40,5
```

In [22]:
```
classifier_regressor = GridSearchCV(Classifier,parameter,scoring = 'accuracy',
classifier_regressor
```

Out[22]:
```
▸ GridSearchCV
▸ estimator: LogisticRegression
    ▸ LogisticRegression
```

In [23]:
```
classifier_regressor.fit(X_train,y_train)
```

```
  n_iter_i = _check_optimize_result(
C:\Users\peddi\AppData\Local\Programs\Python\Python311\Lib\site-packages\s
klearn\model_selection\_validation.py:425: FitFailedWarning:
330 fits failed out of a total of 495.
The score on these train-test partitions for these parameters will be set
to nan.
If these failures are not expected, you can try to debug them by setting e
rror_score='raise'.

Below are more details about the failures:
---------------------------------------------------------------------------
------
165 fits failed with the following error:
Traceback (most recent call last):
  File "C:\Users\peddi\AppData\Local\Programs\Python\Python311\Lib\site-pa
ckages\sklearn\model_selection\_validation.py", line 732, in _fit_and_scor
e
    estimator.fit(X_train, y_train, **fit_params)
  File "C:\Users\peddi\AppData\Local\Programs\Python\Python311\Lib\site-pa
ckages\sklearn\base.py", line 1151, in wrapper
```

In [24]:
```
print(classifier_regressor.best_params_)
```

```
{'C': 5, 'max_iter': 100, 'penalty': 'l2'}
```

In [25]:
```
print(classifier_regressor.best_score_)
```

```
0.9055023923444976
```

In [29]:
```
#prediction
y_pred = classifier_regressor.predict(X_test)
y_pred
```

Out[29]:
```
array([1, 0, 0, 1, 1, 0, 0, 0, 1, 1, 1, 0, 1, 0, 1, 0, 1, 1, 1, 0, 1, 1,
       0, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 0, 0, 1, 0, 1, 1,
       1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 1, 1, 1, 0, 1, 1, 1, 0, 0, 1, 1, 1,
       0, 0, 1, 1, 0, 0, 1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0,
       1, 1, 1, 1, 0, 1, 1, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 1, 1, 0, 1, 1,
       0, 1, 0, 0, 1, 0, 1, 1, 1, 0, 0, 1, 1, 0, 1, 0, 0, 1, 1, 0, 0, 0,
       1, 1, 1, 0, 1, 1, 1, 0, 1, 0, 1, 1, 0, 1, 0, 0, 0, 1, 1, 1, 1, 1,
       1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1,
       0, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 1], dtype=int64)
```

In [31]:
```
#accuracy score
from sklearn.metrics import accuracy_score,classification_report
score = accuracy_score(y_pred,y_test)
score
```

Out[31]:
```
0.9414893617021277
```

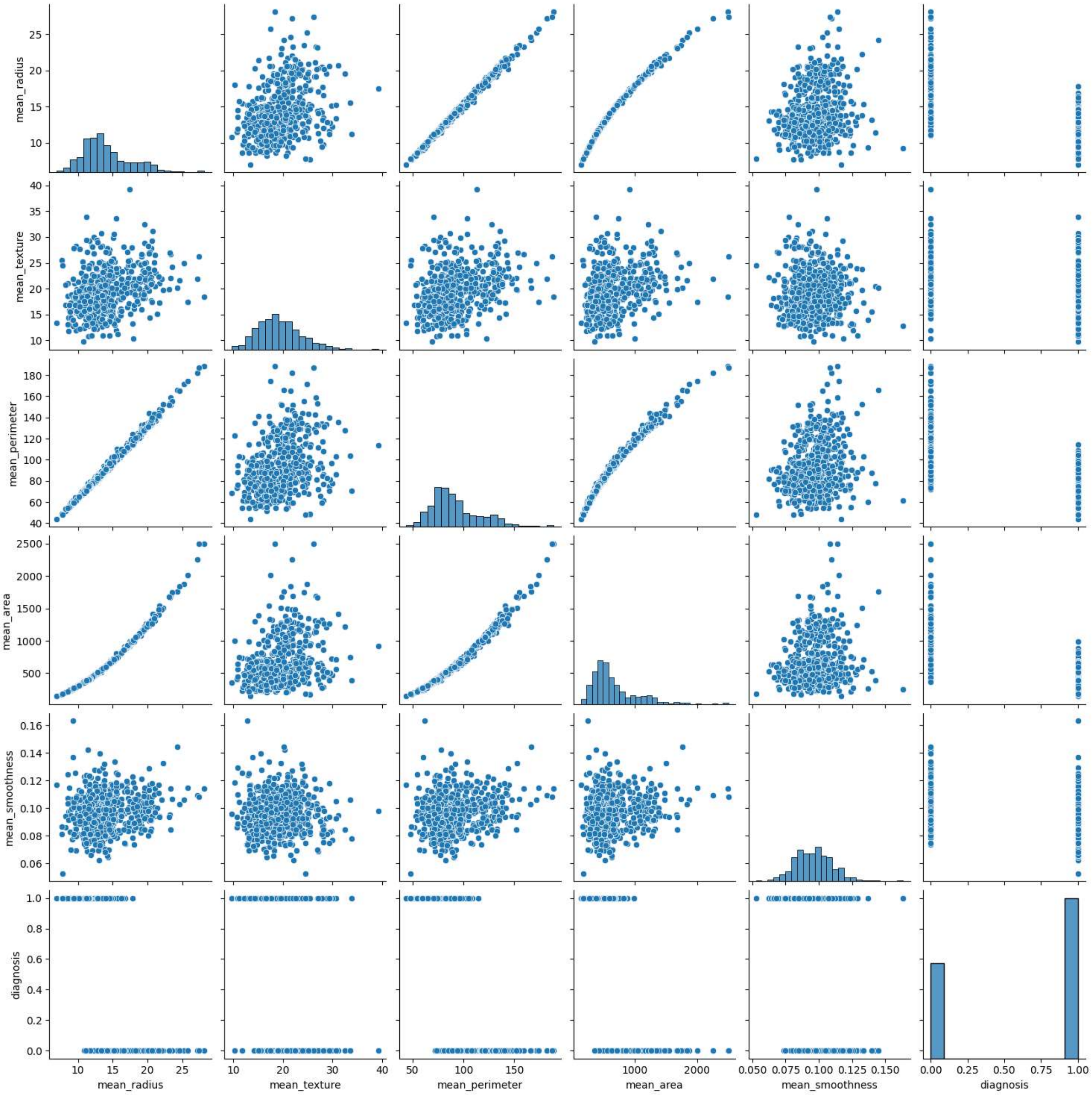
In [34]:
```
print(classification_report(y_pred,y_test))
```

```
              precision    recall  f1-score   support

           0       0.94      0.90      0.92        70
           1       0.94      0.97      0.95       118

    accuracy                           0.94       188
   macro avg       0.94      0.93      0.94       188
weighted avg       0.94      0.94      0.94       188
```

In [35]:
```python
import seaborn as sns
sns.pairplot(df)
```

```
C:\Users\peddi\AppData\Local\Programs\Python\Python311\Lib\site-packages\seab
orn\axisgrid.py:118: UserWarning: The figure layout has changed to tight
  self._figure.tight_layout(*args, **kwargs)
```

Out[35]:
```
<seaborn.axisgrid.PairGrid at 0x23ca06b6190>
```



In [36]:

```
df.corr()
```

Out[36]:

	mean_radius	mean_texture	mean_perimeter	mean_area	mean_smoothness
mean_radius	1.000000	0.323782	0.997855	0.987357	0.170581
mean_texture	0.323782	1.000000	0.329533	0.321086	-0.023389
mean_perimeter	0.997855	0.329533	1.000000	0.986507	0.207278
mean_area	0.987357	0.321086	0.986507	1.000000	0.177028
mean_smoothness	0.170581	-0.023389	0.207278	0.177028	1.000000
diagnosis	-0.730029	-0.415185	-0.742636	-0.708984	-0.358560

In [42]:

```
print("Enter mean_radius = ")
mean_radius = float(input())
print("Enter mean_texture = ")
mean_texture = float(input())
print("Enter mean_perimeter = ")
mean_perimeter = float(input())
print("Enter mean_area = ")
mean_area = float(input())
print("Enter mean_smoothness = ")
mean_smoothness = float(input())

# Make predictions based on user inputs
user_input = [[mean_radius, mean_texture, mean_perimeter, mean_area, mean_smoo
prediction = classifier_regressor.predict(user_input)

# Display the prediction
print("The predicted diagnosis is:", prediction[0])

if(prediction[0]):
    print("No cancer")
else:
    print("Cancer Detected")
```

```
Enter mean_radius =
17
Enter mean_texture =
10
Enter mean_perimeter =
120
Enter mean_area =
10001
Enter mean_smoothness =
0.118
The predicted diagnosis is: 0
Cancer Detected

C:\Users\peddi\AppData\Local\Programs\Python\Python311\Lib\site-packages\skle
arn\base.py:464: UserWarning: X does not have valid feature names, but Logist
icRegression was fitted with feature names
  warnings.warn(
```

In [40]:

```
# example input from data set
# 17.99    10.38    122.8    1001.0    0.11840    output = 0
# 7.76     24.54    47.92    181.0     0.05263    output = 1
```

In [ ]:

```
```