Breast Cancer Detection

December 3, 2024

Anthony Adungo Machine Learning Project.

This project leverages machine learning to develop a predictive model for early detection of breast cancer. Using a dataset of medical attributes such as tumor size, texture, and cell features, the model classifies whether a tumor is benign or malignant. The goal is to enhance diagnostic precision, reduce invasive procedures, and improve patient outcomes through data-driven insights.

The dataset used can be found at https://www.kaggle.com/datasets/uciml/breast-cancer-wisconsin-data

1 Part 1: Data Pre-processing

I) Importing the dataset and exploring its properties.

[1]:
```
#Importing all the necessary libraries.
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
```

[2]:
```
df = pd.read_csv('breast cancer kaggle.csv')
df.sample(5)
```

[2]:
```
           id diagnosis  radius_mean  texture_mean  perimeter_mean  area_mean  \
125     86561         B        13.85         17.21           88.44      588.7
64   85922302         M        12.68         23.84           82.69      499.0
272   8910988         M        21.75         20.99          147.30     1491.0
321    894618         M        20.16         19.66          131.10     1274.0
99     862548         M        14.42         19.77           94.48      642.5

     smoothness_mean  compactness_mean  concavity_mean  concave points_mean  \
125          0.08785           0.06136         0.01420              0.01141
64           0.11220           0.12620         0.11280              0.06873
272          0.09401           0.19610         0.21950              0.10880
321          0.08020           0.08564         0.11550              0.07726
99           0.09752           0.11410         0.09388              0.05839
```

```
     …  texture_worst  perimeter_worst  area_worst  smoothness_worst  \
125  …          23.58            100.3       725.9            0.1157
64   …          33.47            111.8       888.3            0.1851
272  …          28.18            195.9      2384.0            0.1272
321  …          23.03            150.2      1657.0            0.1054
99   …          30.86            109.5       826.4            0.1431

     compactness_worst  concavity_worst  concave points_worst  symmetry_worst  \
125             0.1350          0.08115               0.05104          0.2364
64              0.4061          0.40240               0.17160          0.3383
272             0.4725          0.58070               0.18410          0.2833
321             0.1537          0.26060               0.14250          0.3055
99              0.3026          0.31940               0.15650          0.2718

     fractal_dimension_worst  Unnamed: 32
125                  0.07182          NaN
64                   0.10310          NaN
272                  0.08858          NaN
321                  0.05933          NaN
99                   0.09353          NaN

[5 rows x 33 columns]
```

[3]:
```
df.shape
```

[3]:
```
(569, 33)
```

[4]:
```
df.columns
```

[4]:
```
Index(['id', 'diagnosis', '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'],
      dtype='object')
```

[5]:
```
df.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)
memory usage: 146.8+ KB
```

[6]:
```
#Finding the statistical summary of the dataset.
df.describe()
```

[6]:

	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]
```

[7]:
```python
print(df.dtypes)
```

```
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
```

[8]:
```python
#Finding the categorical variables.
df.select_dtypes(include='object').columns
```

[8]:
```
Index(['diagnosis'], dtype='object')
```

[9]:
```
'''There is only one column with a categorical variable, that is the diagnosis␣
↪column.'''
```

[9]:
```
'There is only one column with a categorical variable, that is the diagnosis
column.'
```

[10]:
```
len(df.select_dtypes(include='object').columns)
```

[10]:
```
1
```

[11]:
```
df.select_dtypes(include=['float64','int64']).columns
```

[11]:
```
Index(['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'],
      dtype='object')
```

[12]:
```
len(df.select_dtypes(include=['float64','int64']).columns)
```

[12]:
```
32
```

II) Dealing with missing values.

[13]:
```
missing_data = df.isnull()
print(missing_data.head())
for column in missing_data.columns.values.tolist():
    print(column)
    print(missing_data[column].value_counts())
    print(" ")
```

```
      id  diagnosis  radius_mean  texture_mean  perimeter_mean  area_mean  \
0  False      False        False         False           False      False
1  False      False        False         False           False      False
2  False      False        False         False           False      False
3  False      False        False         False           False      False
4  False      False        False         False           False      False

   smoothness_mean  compactness_mean  concavity_mean  concave points_mean  \
0            False             False           False                False
1            False             False           False                False
2            False             False           False                False
3            False             False           False                False
4            False             False           False                False
```

```
   …  texture_worst  perimeter_worst  area_worst  smoothness_worst  \
0  …          False            False       False             False
1  …          False            False       False             False
2  …          False            False       False             False
3  …          False            False       False             False
4  …          False            False       False             False

   compactness_worst  concavity_worst  concave points_worst  symmetry_worst  \
0              False            False                 False           False
1              False            False                 False           False
2              False            False                 False           False
3              False            False                 False           False
4              False            False                 False           False

   fractal_dimension_worst  Unnamed: 32
0                    False         True
1                    False         True
2                    False         True
3                    False         True
4                    False         True

[5 rows x 33 columns]
id
id
False    569
Name: count, dtype: int64

diagnosis
diagnosis
False    569
Name: count, dtype: int64

radius_mean
radius_mean
False    569
Name: count, dtype: int64

texture_mean
texture_mean
False    569
Name: count, dtype: int64

perimeter_mean
perimeter_mean
False    569
Name: count, dtype: int64
```

```
area_mean
area_mean
False    569
Name: count, dtype: int64

smoothness_mean
smoothness_mean
False    569
Name: count, dtype: int64

compactness_mean
compactness_mean
False    569
Name: count, dtype: int64

concavity_mean
concavity_mean
False    569
Name: count, dtype: int64

concave points_mean
concave points_mean
False    569
Name: count, dtype: int64

symmetry_mean
symmetry_mean
False    569
Name: count, dtype: int64

fractal_dimension_mean
fractal_dimension_mean
False    569
Name: count, dtype: int64

radius_se
radius_se
False    569
Name: count, dtype: int64

texture_se
texture_se
False    569
Name: count, dtype: int64

perimeter_se
perimeter_se
False    569
```

```
Name: count, dtype: int64

area_se
area_se
False    569
Name: count, dtype: int64

smoothness_se
smoothness_se
False    569
Name: count, dtype: int64

compactness_se
compactness_se
False    569
Name: count, dtype: int64

concavity_se
concavity_se
False    569
Name: count, dtype: int64

concave points_se
concave points_se
False    569
Name: count, dtype: int64

symmetry_se
symmetry_se
False    569
Name: count, dtype: int64

fractal_dimension_se
fractal_dimension_se
False    569
Name: count, dtype: int64

radius_worst
radius_worst
False    569
Name: count, dtype: int64

texture_worst
texture_worst
False    569
Name: count, dtype: int64

perimeter_worst
```

```
perimeter_worst
False    569
Name: count, dtype: int64

area_worst
area_worst
False    569
Name: count, dtype: int64

smoothness_worst
smoothness_worst
False    569
Name: count, dtype: int64

compactness_worst
compactness_worst
False    569
Name: count, dtype: int64

concavity_worst
concavity_worst
False    569
Name: count, dtype: int64

concave points_worst
concave points_worst
False    569
Name: count, dtype: int64

symmetry_worst
symmetry_worst
False    569
Name: count, dtype: int64

fractal_dimension_worst
fractal_dimension_worst
False    569
Name: count, dtype: int64

Unnamed: 32
Unnamed: 32
True    569
Name: count, dtype: int64
```

[14]:
```
df.isnull().values.sum()
```

[14]: 569

[15]:
```
df.columns[df.isnull().any()]
```

[15]: Index(['Unnamed: 32'], dtype='object')

[16]:
```
'''There is one column with null values.'''
```

[16]: 'There is one column with null values.'

[17]:
```
#Dropping the column.
df = df.drop(columns='Unnamed: 32')
```

[18]:
```
df.shape
```

[18]: (569, 32)

III) Dealing with categorical data.

[19]:
```
df.select_dtypes(include='object').columns
```

[19]: Index(['diagnosis'], dtype='object')

[20]:
```
df['diagnosis'].unique()
```

[20]: array(['M', 'B'], dtype=object)

[21]:
```
'''There are only two unique values, malignant and benign.'''
```

[21]: 'There are only two unique values, malignant and benign.'

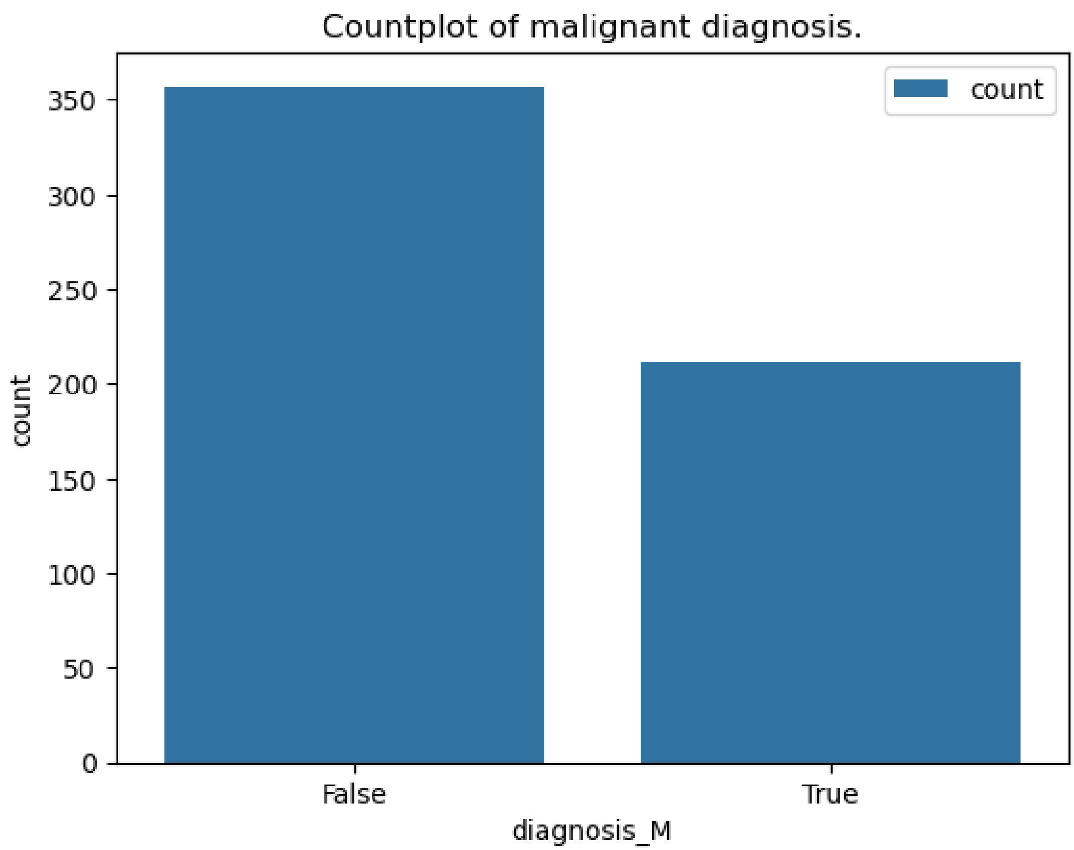
[22]:
```
# One hot encoding
dataset = pd.get_dummies(data = df, drop_first=True)
dataset.head()
```

[22]:
```
         id  radius_mean  texture_mean  perimeter_mean  area_mean  \
0    842302        17.99         10.38          122.80     1001.0
1    842517        20.57         17.77          132.90     1326.0
2  84300903        19.69         21.25          130.00     1203.0
3  84348301        11.42         20.38           77.58      386.1
4  84358402        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
```

```
   symmetry_mean  ...  texture_worst  perimeter_worst  area_worst  \
0         0.2419  ...          17.33           184.60      2019.0
1         0.1812  ...          23.41           158.80      1956.0
2         0.2069  ...          25.53           152.50      1709.0
3         0.2597  ...          26.50            98.87       567.7
4         0.1809  ...          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  diagnosis_M
0          0.4601                  0.11890         True
1          0.2750                  0.08902         True
2          0.3613                  0.08758         True
3          0.6638                  0.17300         True
4          0.2364                  0.07678         True

[5 rows x 32 columns]
```

IV) Visualization.

[23]:
```
sns.countplot(dataset,x='diagnosis_M', label='count')
plt.title('Countplot of malignant diagnosis.')
plt.show()
```



[24]:
```
#Count of benign values.
(dataset.diagnosis_M==0).sum()
```

[24]: 357

[25]:
```
#Count of malignant values.
(dataset.diagnosis_M==1).sum()
```

[25]: 212

V) Correlation matrix and heatmap.

[26]:
```
dataset_2 = dataset.drop(columns= 'diagnosis_M')
```

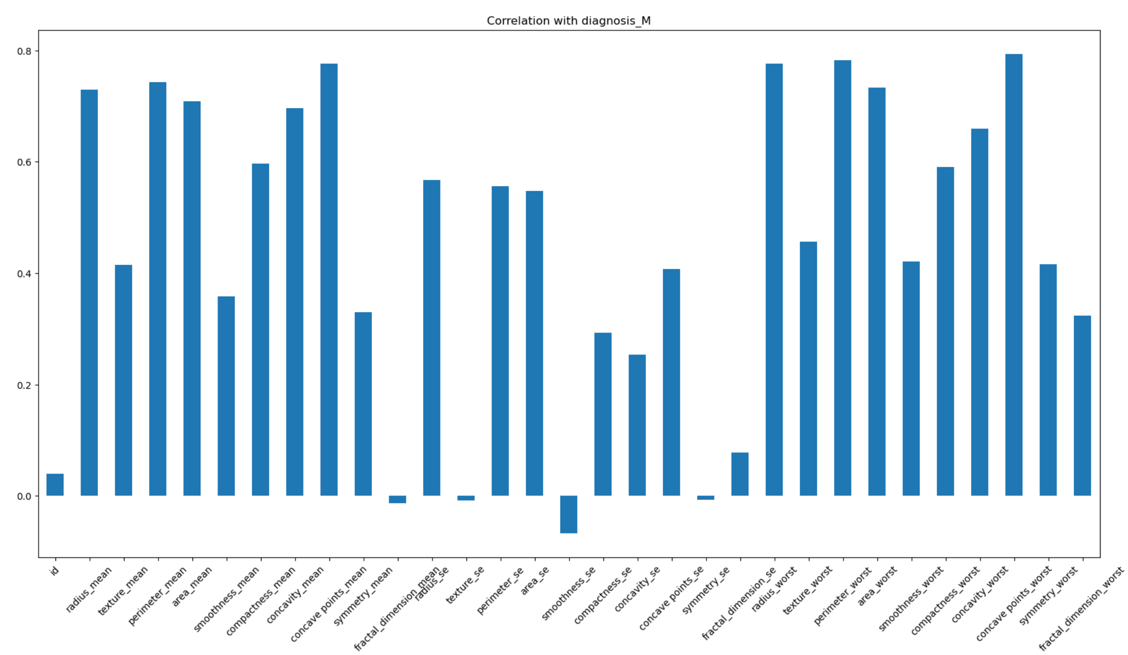
[27]:
```
dataset_2.head(2)
```

[27]:
```
       id  radius_mean  texture_mean  perimeter_mean  area_mean  \
0  842302        17.99         10.38           122.8     1001.0
1  842517        20.57         17.77           132.9     1326.0

   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

   symmetry_mean  …  radius_worst  texture_worst  perimeter_worst  \
0         0.2419  …         25.38          17.33            184.6
1         0.1812  …         24.99          23.41            158.8

   area_worst  smoothness_worst  compactness_worst  concavity_worst  \
0      2019.0            0.1622             0.6656           0.7119
1      1956.0            0.1238             0.1866           0.2416

   concave points_worst  symmetry_worst  fractal_dimension_worst
0                0.2654          0.4601                  0.11890
1                0.1860          0.2750                  0.08902

[2 rows x 31 columns]
```

[28]:
```python
dataset_2.corrwith(dataset['diagnosis_M']).plot.bar(
    figsize=(20,10),
    title = 'Correlation with diagnosis_M',
    rot = 45)
```

[28]:
```
<Axes: title={'center': 'Correlation with diagnosis_M'}>
```



[29]:
```python
corr = dataset.corr()
```

[30]:

```
corr
```

[30]:

	id	radius_mean	texture_mean	perimeter_mean
id	1.000000	0.074626	0.099770	0.073159
radius_mean	0.074626	1.000000	0.323782	0.997855
texture_mean	0.099770	0.323782	1.000000	0.329533
perimeter_mean	0.073159	0.997855	0.329533	1.000000
area_mean	0.096893	0.987357	0.321086	0.986507
smoothness_mean	-0.012968	0.170581	-0.023389	0.207278
compactness_mean	0.000096	0.506124	0.236702	0.556936
concavity_mean	0.050080	0.676764	0.302418	0.716136
concave points_mean	0.044158	0.822529	0.293464	0.850977
symmetry_mean	-0.022114	0.147741	0.071401	0.183027
fractal_dimension_mean	-0.052511	-0.311631	-0.076437	-0.261477
radius_se	0.143048	0.679090	0.275869	0.691765
texture_se	-0.007526	-0.097317	0.386358	-0.086761
perimeter_se	0.137331	0.674172	0.281673	0.693135
area_se	0.177742	0.735864	0.259845	0.744983
smoothness_se	0.096781	-0.222600	0.006614	-0.202694
compactness_se	0.033961	0.206000	0.191975	0.250744
concavity_se	0.055239	0.194204	0.143293	0.228082
concave points_se	0.078768	0.376169	0.163851	0.407217
symmetry_se	-0.017306	-0.104321	0.009127	-0.081629
fractal_dimension_se	0.025725	-0.042641	0.054458	-0.005523
radius_worst	0.082405	0.969539	0.352573	0.969476
texture_worst	0.064720	0.297008	0.912045	0.303038
perimeter_worst	0.079986	0.965137	0.358040	0.970387
area_worst	0.107187	0.941082	0.343546	0.941550
smoothness_worst	0.010338	0.119616	0.077503	0.150549
compactness_worst	-0.002968	0.413463	0.277830	0.455774
concavity_worst	0.023203	0.526911	0.301025	0.563879
concave points_worst	0.035174	0.744214	0.295316	0.771241
symmetry_worst	-0.044224	0.163953	0.105008	0.189115
fractal_dimension_worst	-0.029866	0.007066	0.119205	0.051019
diagnosis_M	0.039769	0.730029	0.415185	0.742636

	area_mean	smoothness_mean	compactness_mean
id	0.096893	-0.012968	0.000096
radius_mean	0.987357	0.170581	0.506124
texture_mean	0.321086	-0.023389	0.236702
perimeter_mean	0.986507	0.207278	0.556936
area_mean	1.000000	0.177028	0.498502
smoothness_mean	0.177028	1.000000	0.659123
compactness_mean	0.498502	0.659123	1.000000
concavity_mean	0.685983	0.521984	0.883121
concave points_mean	0.823269	0.553695	0.831135
symmetry_mean	0.151293	0.557775	0.602641

fractal_dimension_mean	-0.283110	0.584792	0.565369
radius_se	0.732562	0.301467	0.497473
texture_se	-0.066280	0.068406	0.046205
perimeter_se	0.726628	0.296092	0.548905
area_se	0.800086	0.246552	0.455653
smoothness_se	-0.166777	0.332375	0.135299
compactness_se	0.212583	0.318943	0.738722
concavity_se	0.207660	0.248396	0.570517
concave points_se	0.372320	0.380676	0.642262
symmetry_se	-0.072497	0.200774	0.229977
fractal_dimension_se	-0.019887	0.283607	0.507318
radius_worst	0.962746	0.213120	0.535315
texture_worst	0.287489	0.036072	0.248133
perimeter_worst	0.959120	0.238853	0.590210
area_worst	0.959213	0.206718	0.509604
smoothness_worst	0.123523	0.805324	0.565541
compactness_worst	0.390410	0.472468	0.865809
concavity_worst	0.512606	0.434926	0.816275
concave points_worst	0.722017	0.503053	0.815573
symmetry_worst	0.143570	0.394309	0.510223
fractal_dimension_worst	0.003738	0.499316	0.687382
diagnosis_M	0.708984	0.358560	0.596534

	concavity_mean	concave points_mean	symmetry_mean	\
id	0.050080	0.044158	-0.022114	
radius_mean	0.676764	0.822529	0.147741	
texture_mean	0.302418	0.293464	0.071401	
perimeter_mean	0.716136	0.850977	0.183027	
area_mean	0.685983	0.823269	0.151293	
smoothness_mean	0.521984	0.553695	0.557775	
compactness_mean	0.883121	0.831135	0.602641	
concavity_mean	1.000000	0.921391	0.500667	
concave points_mean	0.921391	1.000000	0.462497	
symmetry_mean	0.500667	0.462497	1.000000	
fractal_dimension_mean	0.336783	0.166917	0.479921	
radius_se	0.631925	0.698050	0.303379	
texture_se	0.076218	0.021480	0.128053	
perimeter_se	0.660391	0.710650	0.313893	
area_se	0.617427	0.690299	0.223970	
smoothness_se	0.098564	0.027653	0.187321	
compactness_se	0.670279	0.490424	0.421659	
concavity_se	0.691270	0.439167	0.342627	
concave points_se	0.683260	0.615634	0.393298	
symmetry_se	0.178009	0.095351	0.449137	
fractal_dimension_se	0.449301	0.257584	0.331786	
radius_worst	0.688236	0.830318	0.185728	
texture_worst	0.299879	0.292752	0.090651	

perimeter_worst	0.729565	0.855923	0.219169
area_worst	0.675987	0.809630	0.177193
smoothness_worst	0.448822	0.452753	0.426675
compactness_worst	0.754968	0.667454	0.473200
concavity_worst	0.884103	0.752399	0.433721
concave points_worst	0.861323	0.910155	0.430297
symmetry_worst	0.409464	0.375744	0.699826
fractal_dimension_worst	0.514930	0.368661	0.438413
diagnosis_M	0.696360	0.776614	0.330499

	…	texture_worst	perimeter_worst	area_worst	\
id	…	0.064720	0.079986	0.107187	
radius_mean	…	0.297008	0.965137	0.941082	
texture_mean	…	0.912045	0.358040	0.343546	
perimeter_mean	…	0.303038	0.970387	0.941550	
area_mean	…	0.287489	0.959120	0.959213	
smoothness_mean	…	0.036072	0.238853	0.206718	
compactness_mean	…	0.248133	0.590210	0.509604	
concavity_mean	…	0.299879	0.729565	0.675987	
concave points_mean	…	0.292752	0.855923	0.809630	
symmetry_mean	…	0.090651	0.219169	0.177193	
fractal_dimension_mean	…	-0.051269	-0.205151	-0.231854	
radius_se	…	0.194799	0.719684	0.751548	
texture_se	…	0.409003	-0.102242	-0.083195	
perimeter_se	…	0.200371	0.721031	0.730713	
area_se	…	0.196497	0.761213	0.811408	
smoothness_se	…	-0.074743	-0.217304	-0.182195	
compactness_se	…	0.143003	0.260516	0.199371	
concavity_se	…	0.100241	0.226680	0.188353	
concave points_se	…	0.086741	0.394999	0.342271	
symmetry_se	…	-0.077473	-0.103753	-0.110343	
fractal_dimension_se	…	-0.003195	-0.001000	-0.022736	
radius_worst	…	0.359921	0.993708	0.984015	
texture_worst	…	1.000000	0.365098	0.345842	
perimeter_worst	…	0.365098	1.000000	0.977578	
area_worst	…	0.345842	0.977578	1.000000	
smoothness_worst	…	0.225429	0.236775	0.209145	
compactness_worst	…	0.360832	0.529408	0.438296	
concavity_worst	…	0.368366	0.618344	0.543331	
concave points_worst	…	0.359755	0.816322	0.747419	
symmetry_worst	…	0.233027	0.269493	0.209146	
fractal_dimension_worst	…	0.219122	0.138957	0.079647	
diagnosis_M	…	0.456903	0.782914	0.733825	

	smoothness_worst	compactness_worst	concavity_worst	\
id	0.010338	-0.002968	0.023203	
radius_mean	0.119616	0.413463	0.526911	

```
texture_mean                  0.077503         0.277830         0.301025
perimeter_mean                0.150549         0.455774         0.563879
area_mean                     0.123523         0.390410         0.512606
smoothness_mean               0.805324         0.472468         0.434926
compactness_mean              0.565541         0.865809         0.816275
concavity_mean                0.448822         0.754968         0.884103
concave points_mean           0.452753         0.667454         0.752399
symmetry_mean                 0.426675         0.473200         0.433721
fractal_dimension_mean        0.504942         0.458798         0.346234
radius_se                     0.141919         0.287103         0.380585
texture_se                   -0.073658        -0.092439        -0.068956
perimeter_se                  0.130054         0.341919         0.418899
area_se                       0.125389         0.283257         0.385100
smoothness_se                 0.314457        -0.055558        -0.058298
compactness_se                0.227394         0.678780         0.639147
concavity_se                  0.168481         0.484858         0.662564
concave points_se             0.215351         0.452888         0.549592
symmetry_se                  -0.012662         0.060255         0.037119
fractal_dimension_se          0.170568         0.390159         0.379975
radius_worst                  0.216574         0.475820         0.573975
texture_worst                 0.225429         0.360832         0.368366
perimeter_worst               0.236775         0.529408         0.618344
area_worst                    0.209145         0.438296         0.543331
smoothness_worst              1.000000         0.568187         0.518523
compactness_worst             0.568187         1.000000         0.892261
concavity_worst               0.518523         0.892261         1.000000
concave points_worst          0.547691         0.801080         0.855434
symmetry_worst                0.493838         0.614441         0.532520
fractal_dimension_worst       0.617624         0.810455         0.686511
diagnosis_M                   0.421465         0.590998         0.659610

                         concave points_worst  symmetry_worst  \
id                                   0.035174       -0.044224
radius_mean                          0.744214        0.163953
texture_mean                         0.295316        0.105008
perimeter_mean                       0.771241        0.189115
area_mean                            0.722017        0.143570
smoothness_mean                      0.503053        0.394309
compactness_mean                     0.815573        0.510223
concavity_mean                       0.861323        0.409464
concave points_mean                  0.910155        0.375744
symmetry_mean                        0.430297        0.699826
fractal_dimension_mean               0.175325        0.334019
radius_se                            0.531062        0.094543
texture_se                          -0.119638       -0.128215
perimeter_se                         0.554897        0.109930
area_se                              0.538166        0.074126
```

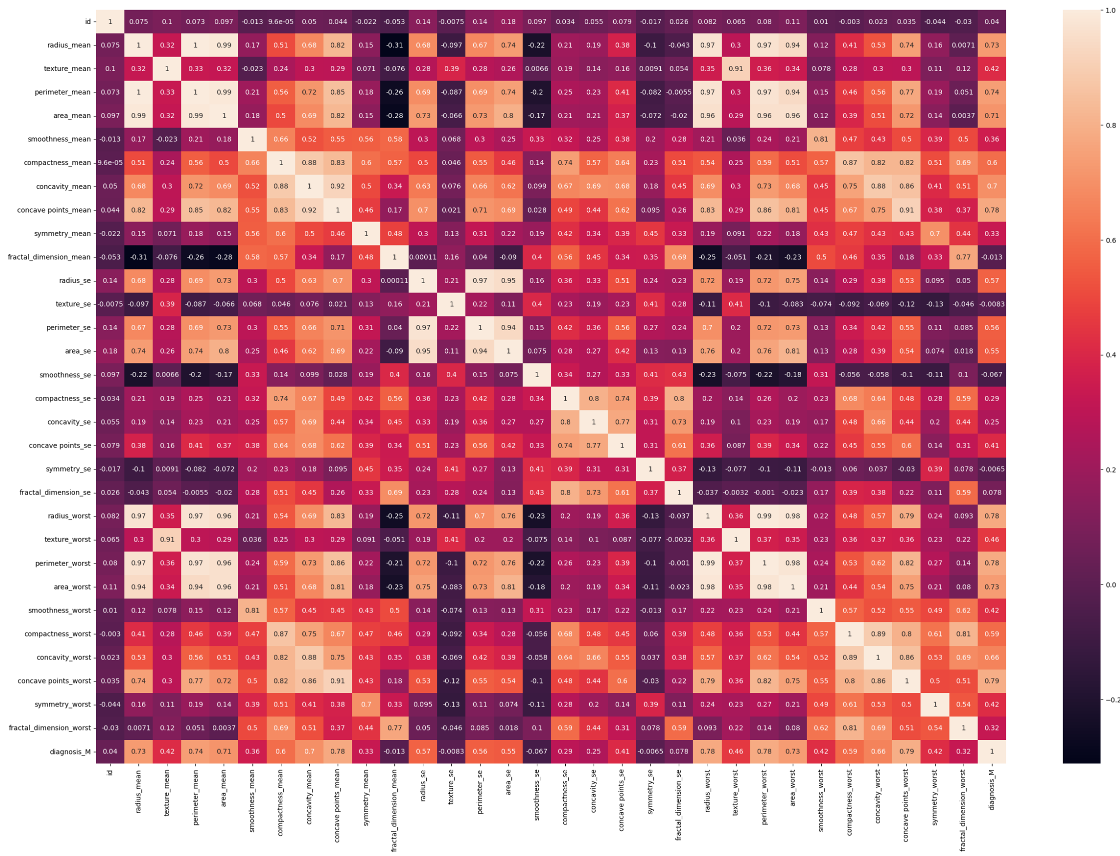
smoothness_se	-0.102007	-0.107342
compactness_se	0.483208	0.277878
concavity_se	0.440472	0.197788
concave points_se	0.602450	0.143116
symmetry_se	-0.030413	0.389402
fractal_dimension_se	0.215204	0.111094
radius_worst	0.787424	0.243529
texture_worst	0.359755	0.233027
perimeter_worst	0.816322	0.269493
area_worst	0.747419	0.209146
smoothness_worst	0.547691	0.493838
compactness_worst	0.801080	0.614441
concavity_worst	0.855434	0.532520
concave points_worst	1.000000	0.502528
symmetry_worst	0.502528	1.000000
fractal_dimension_worst	0.511114	0.537848
diagnosis_M	0.793566	0.416294

	fractal_dimension_worst	diagnosis_M
id	-0.029866	0.039769
radius_mean	0.007066	0.730029
texture_mean	0.119205	0.415185
perimeter_mean	0.051019	0.742636
area_mean	0.003738	0.708984
smoothness_mean	0.499316	0.358560
compactness_mean	0.687382	0.596534
concavity_mean	0.514930	0.696360
concave points_mean	0.368661	0.776614
symmetry_mean	0.438413	0.330499
fractal_dimension_mean	0.767297	-0.012838
radius_se	0.049559	0.567134
texture_se	-0.045655	-0.008303
perimeter_se	0.085433	0.556141
area_se	0.017539	0.548236
smoothness_se	0.101480	-0.067016
compactness_se	0.590973	0.292999
concavity_se	0.439329	0.253730
concave points_se	0.310655	0.408042
symmetry_se	0.078079	-0.006522
fractal_dimension_se	0.591328	0.077972
radius_worst	0.093492	0.776454
texture_worst	0.219122	0.456903
perimeter_worst	0.138957	0.782914
area_worst	0.079647	0.733825
smoothness_worst	0.617624	0.421465
compactness_worst	0.810455	0.590998
concavity_worst	0.686511	0.659610

```
concave points_worst                          0.511114     0.793566
symmetry_worst                                0.537848     0.416294
fractal_dimension_worst                       1.000000     0.323872
diagnosis_M                                   0.323872     1.000000

[32 rows x 32 columns]
```

[31]:
```python
plt.figure(figsize=(30,20))
sns.heatmap(corr, annot= True)
```

[31]:
```
<Axes: >
```



VI) Splitting the dataset into train and test sets.

[32]:
```python
dataset.head()
```

[32]:
```
         id  radius_mean  texture_mean  perimeter_mean  area_mean  \
0    842302        17.99         10.38          122.80     1001.0
1    842517        20.57         17.77          132.90     1326.0
2  84300903        19.69         21.25          130.00     1203.0
3  84348301        11.42         20.38           77.58      386.1
```

```
4  84358402        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

   symmetry_mean  …  texture_worst  perimeter_worst  area_worst  \
0         0.2419  …          17.33           184.60      2019.0
1         0.1812  …          23.41           158.80      1956.0
2         0.2069  …          25.53           152.50      1709.0
3         0.2597  …          26.50            98.87       567.7
4         0.1809  …          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  diagnosis_M
0          0.4601                  0.11890         True
1          0.2750                  0.08902         True
2          0.3613                  0.08758         True
3          0.6638                  0.17300         True
4          0.2364                  0.07678         True

[5 rows x 32 columns]
```

[33]:
```python
#Matrix of features.
X = dataset.iloc[:,1:-1].values
```

[34]:
```python
X.shape
```

[34]: (569, 30)

[35]:
```python
#Dependent variable.
y = dataset.iloc[:,-1].values
```

[36]:
```python
y.shape
```

[36]: (569,)

[37]:
```python
from sklearn.model_selection import train_test_split
```

[38]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
↪random_state=101)
```

[39]:
```
X_train.shape
```

[39]:
```
(455, 30)
```

[40]:
```
X_test
```

[40]:
```
array([[1.236e+01, 1.854e+01, 7.901e+01, …, 8.442e-02, 2.983e-01,
         7.185e-02],
        [1.404e+01, 1.598e+01, 8.978e+01, …, 7.453e-02, 2.725e-01,
         7.234e-02],
        [1.291e+01, 1.633e+01, 8.253e+01, …, 8.235e-02, 3.024e-01,
         6.949e-02],
        …,
        [1.128e+01, 1.339e+01, 7.300e+01, …, 8.611e-02, 2.102e-01,
         6.784e-02],
        [1.487e+01, 2.021e+01, 9.612e+01, …, 1.017e-01, 2.369e-01,
         6.599e-02],
        [1.822e+01, 1.887e+01, 1.187e+02, …, 1.776e-01, 2.812e-01,
         8.198e-02]])
```

[41]:
```
y_train.shape
```

[41]:
```
(455,)
```

[42]:
```
y_test
```

[42]:
```
array([False, False, False,  True, False, False, False,  True, False,
       False,  True, False, False, False,  True, False, False, False,
        True,  True, False, False, False, False,  True, False,  True,
       False,  True,  True, False,  True, False,  True, False, False,
        True,  True,  True,  True,  True, False, False, False, False,
       False,  True, False,  True, False,  True, False, False,  True,
       False, False,  True,  True, False, False,  True,  True, False,
       False,  True, False, False,  True,  True, False,  True, False,
       False, False,  True,  True, False,  True,  True, False, False,
       False, False, False, False, False,  True, False,  True,  True,
       False,  True,  True, False, False, False, False, False,  True,
        True,  True, False, False, False, False, False, False, False,
       False, False, False, False, False,  True])
```

VII) Feature scaling.

[43]:
```
from sklearn.preprocessing import StandardScaler
```

[44]:
```
scaler = StandardScaler()
```

[45]:
```
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

[46]:
```
X_train
```

[46]:
```
array([[ 0.02090193,  0.28562106,  0.01889271, …,  0.28708398,
        -0.59963793, -0.32285831],
       [-0.53400124, -1.40599342, -0.51656117, …, -0.50392051,
         0.88583908,  0.43518026],
       [-0.2551693 , -0.43868901, -0.3137073 , …, -0.99299632,
        -0.22946024, -0.68461207],
       …,
       [ 0.53715513,  0.08001046,  0.48846929, …,  0.56225563,
        -0.41534346, -1.1291307 ],
       [ 1.28254744,  0.49590463,  1.24364806, …,  1.36496632,
         1.21947563,  0.77994255],
       [-0.11437297, -0.1466285 , -0.12892951, …,  0.20346828,
        -0.09918311,  0.32007679]])
```

[47]:
```
X_test
```

[47]:
```
array([[-0.48706913, -0.17933928, -0.519373  , …, -0.46515324,
         0.11370878, -0.69173752],
       [-0.02326947, -0.7774792 , -0.08675198, …, -0.61550946,
        -0.29618755, -0.66488004],
       [-0.33522996, -0.69570226, -0.37797783, …, -0.49662314,
         0.17884735, -0.82109189],
       …,
       [-0.78522606, -1.38262858, -0.76078919, …, -0.43946041,
        -1.28597597, -0.91153034],
       [ 0.20586965,  0.21085357,  0.16792001, …, -0.20244792,
        -0.86178093, -1.01293101],
       [ 1.13070827, -0.1022353 ,  1.07493791, …,  0.95144869,
        -0.15796669, -0.13650031]])
```

2 Part 2: Building the models.

I) Logistic Regression.

[48]:
```
from sklearn.linear_model import LogisticRegression
```

[49]:
```
model_lr = LogisticRegression(random_state=0)
```

[50]:
```
model_lr.fit(X_train, y_train)
```

[50]:
```
LogisticRegression(random_state=0)
```

[51]:
```
y_pred = model_lr.predict(X_test)
```

[52]:
```
from sklearn.metrics import accuracy_score, confusion_matrix, f1_score,␣
↪precision_score, recall_score
```

[53]:
```
acc = accuracy_score(y_test, y_pred)
f1 = f1_score(y_test, y_pred)
prec = precision_score(y_test, y_pred)
rec = recall_score(y_test, y_pred)
```

[54]:
```
results = pd.DataFrame([['Logistic Regression', acc, f1, prec, rec]],
                       columns= ['Model', 'Accuracy', 'F1 Score', 'Precision',␣
↪'Recall'])
```

[55]:
```
results
```

[55]:

	Model	Accuracy	F1 Score	Precision	Recall
0	Logistic Regression	0.991228	0.987952	1.0	0.97619

[56]:
```
cm = confusion_matrix(y_test, y_pred)
print(cm)
```

```
[[72  0]
 [ 1 41]]
```

Cross Validation.

[57]:
```
from sklearn.model_selection import cross_val_score
```

[58]:
```
accuracies = cross_val_score(estimator=model_lr, X=X_train,y = y_train, cv=10)
```

[59]:
```
print("Accuracy is {:.2f}%".format(accuracies.mean()*100))
print("Standard deviation is {:.2f}%".format(accuracies.std()*100))
```

```
Accuracy is 97.16%
Standard deviation is 2.18%
```

II) Random Forest Classifier

[60]:
```
from sklearn.ensemble import RandomForestClassifier
```

[61]:
```
classifier_rf = RandomForestClassifier(random_state=0)
classifier_rf.fit(X_train, y_train)
```

[61]:
```
RandomForestClassifier(random_state=0)
```

[62]:
```
y_pred = classifier_rf.predict(X_test)
```

[63]:
```
acc = accuracy_score(y_test, y_pred)
f1 = f1_score(y_test, y_pred)
```

```
prec = precision_score(y_test, y_pred)
rec = recall_score(y_test, y_pred)
```

[64]:
```
model_results = pd.DataFrame([['Random Forest', acc, f1, prec, rec]],
                        columns= ['Model', 'Accuracy', 'F1 Score', 'Precision',␣
 ↪'Recall'])
```

[65]:
```
model_results
```

[65]:

	Model	Accuracy	F1 Score	Precision	Recall
0	Random Forest	0.973684	0.963855	0.97561	0.952381

[66]:
```
cm = confusion_matrix(y_test, y_pred)
print(cm)
```

```
[[71  1]
 [ 2 40]]
```

Cross Validation.

[67]:
```
accuracies = cross_val_score(estimator=classifier_rf, X=X_train,y = y_train,␣
 ↪cv=10)
print("Accuracy is {:.2f}%".format(accuracies.mean()*100))
print("Standard deviation is {:.2f}%".format(accuracies.std()*100))
```

```
Accuracy is 95.63%
Standard deviation is 3.23%
```

[68]:
```
'''Based on accuracy, Logistic Regression is the best model.'''
```

[68]: 'Based on accuracy, Logistic Regression is the best model.'

3 Part 3: Using Randomized Search to find the best parameters.(Logistic Regression)

[69]:
```
from sklearn.model_selection import RandomizedSearchCV
```

[70]:
```
parameters = {'penalty':['l1', 'l2', 'elasticnet', 'none'],
              'C':[0.25,0.5, 0.75, 1, 1.25, 1.5, 1.75, 2.0],
              'solver':['newton-cg','lbfgs','liblinear', 'sag', 'saga']}
```

[71]:
```
parameters
```

[71]:
```
{'penalty': ['l1', 'l2', 'elasticnet', 'none'],
 'C': [0.25, 0.5, 0.75, 1, 1.25, 1.5, 1.75, 2.0],
 'solver': ['newton-cg', 'lbfgs', 'liblinear', 'sag', 'saga']}
```

[72]:
```
random_search = RandomizedSearchCV(estimator=model_lr, param_distributions=␣
↪parameters, n_iter=10, scoring='roc_auc',n_jobs=-1, cv=10, verbose=3)
```

[73]:
```
random_search.fit(X_train, y_train)
```

```
Fitting 10 folds for each of 10 candidates, totalling 100 fits
```

[73]:
```
RandomizedSearchCV(cv=10, estimator=LogisticRegression(random_state=0),
                   n_jobs=-1,
                   param_distributions={'C': [0.25, 0.5, 0.75, 1, 1.25, 1.5,
                                              1.75, 2.0],
                                        'penalty': ['l1', 'l2', 'elasticnet',
                                                    'none'],
                                        'solver': ['newton-cg', 'lbfgs',
                                                   'liblinear', 'sag',
                                                   'saga']},
                   scoring='roc_auc', verbose=3)
```

[74]:
```
random_search.best_estimator_
```

[74]:
```
LogisticRegression(C=1.5, random_state=0, solver='saga')
```

[75]:
```
random_search.best_score_
```

[75]:
```
0.9939075630252102
```

[76]:
```
random_search.best_params_
```

[76]:
```
{'solver': 'saga', 'penalty': 'l2', 'C': 1.5}
```

4 Part 4: Final Model.

[77]:
```
model = LogisticRegression(C=1.
↪25,class_weight=None,dual=False,fit_intercept=True,intercept_scaling=1,l1_ratio=None,␣
↪max_iter=100,multi_class='auto',n_jobs=None,penalty='l1',random_state=0,solver='saga',tol=0
↪0001,verbose=0,warm_start=False)
model.fit(X_train,y_train)
```

[77]:
```
LogisticRegression(C=1.25, penalty='l1', random_state=0, solver='saga')
```

[78]:
```
y_pred = model.predict(X_test)
```

[79]:
```
acc = accuracy_score(y_test, y_pred)
f1 = f1_score(y_test, y_pred)
prec = precision_score(y_test, y_pred)
rec = recall_score(y_test, y_pred)
```

```
final_model_results = pd.DataFrame([['Final Logistic Regression', acc, f1,␣
↪prec, rec]],
                        columns= ['Model', 'Accuracy', 'F1 Score', 'Precision',␣
↪'Recall'])
final_model_results
```

[79]:

	Model	Accuracy	F1 Score	Precision	Recall
0	Final Logistic Regression	0.991228	0.987952	1.0	0.97619

[80]:
```
accuracies = cross_val_score(estimator=model, X=X_train,y = y_train, cv=10)
print("Accuracy is {:.2f}%".format(accuracies.mean()*100))
print("Standard deviation is {:.2f}%".format(accuracies.std()*100))
```

```
Accuracy is 97.16%
Standard deviation is 2.18%
```

5 Part 5: Predicting a single observation.

[81]:
```
dataset.head()
```

[81]:

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

	symmetry_mean	…	texture_worst	perimeter_worst	area_worst	\
0	0.2419	…	17.33	184.60	2019.0	
1	0.1812	…	23.41	158.80	1956.0	
2	0.2069	…	25.53	152.50	1709.0	
3	0.2597	…	26.50	98.87	567.7	
4	0.1809	…	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  diagnosis_M
0          0.4601                  0.11890         True
1          0.2750                  0.08902         True
2          0.3613                  0.08758         True
3          0.6638                  0.17300         True
4          0.2364                  0.07678         True

[5 rows x 32 columns]
```

[82]:

```
single_obsv = [[17.99, 10.38, 122.80, 1001.0, 0.11840, 0.27760,        0.
↪3001,        0.14710, 0.2419, 0.07871, 1.0950, 0.9053, 8.589, 153.40, 0.
↪006399, 0.04904,        0.05373, 0.01587, 0.03003, 0.006193, 25.38,
17.33, 184.60, 2019.0, 0.1622, 0.6656, 0.7119, 0.2654, 0.4601, 0.11890]]
```

[83]:

```
single_obsv
```

[83]:

```
[[17.99,
  10.38,
  122.8,
  1001.0,
  0.1184,
  0.2776,
  0.3001,
  0.1471,
  0.2419,
  0.07871,
  1.095,
  0.9053,
  8.589,
  153.4,
  0.006399,
  0.04904,
  0.05373,
  0.01587,
  0.03003,
  0.006193,
  25.38,
  17.33,
  184.6,
  2019.0,
  0.1622,
  0.6656,
  0.7119,
  0.2654,
  0.4601,
  0.1189]]
```

[84]:
```
model.predict(scaler.transform(single_obsv))
```

[84]:
```
array([ True])
```

[85]:
```
'''The cancer is malignant.'''
```

[85]:
```
'The cancer is malignant.'
```