1. Import Necessary Libraries

In [1]:
```
from sklearn.datasets import load_breast_cancer

import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import AdaBoostClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score,confusion_matrix

import warnings
warnings.filterwarnings('ignore')
```

2. Imoprt Data

In [2]:
```
cancer_data = load_breast_cancer()
cancer_data
```

Out[2]:
```
{'data': array([[1.799e+01, 1.038e+01, 1.228e+02, ..., 2.654e-01, 4.601e-01,
         1.189e-01],
        [2.057e+01, 1.777e+01, 1.329e+02, ..., 1.860e-01, 2.750e-01,
         8.902e-02],
        [1.969e+01, 2.125e+01, 1.300e+02, ..., 2.430e-01, 3.613e-01,
         8.758e-02],
        ...,
        [1.660e+01, 2.808e+01, 1.083e+02, ..., 1.418e-01, 2.218e-01,
         7.820e-02],
        [2.060e+01, 2.933e+01, 1.401e+02, ..., 2.650e-01, 4.087e-01,
         1.240e-01],
        [7.760e+00, 2.454e+01, 4.792e+01, ..., 0.000e+00, 2.871e-01,
         7.039e-02]]),
 'target': array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1,
1, 1,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0,
        0, 0, 1, 0, 1, 1, 1, 1, 1, 0, 0, 1, 0, 0, 1, 1, 1, 1, 0, 1, 0, 0,
        1, 1, 1, 1, 0, 1, 0, 0, 1, 0, 1, 0, 0, 1, 1, 1, 0, 0, 1, 0, 0, 0,
        1, 1, 1, 0, 1, 1, 0, 0, 1, 1, 1, 0, 0, 1, 1, 1, 1, 0, 1, 1, 0, 1,
        1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 0, 0, 1, 1, 1, 0, 0, 1, 0, 1, 0,
        0, 1, 0, 0, 1, 1, 0, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1,
        1, 1, 0, 1, 1, 1, 1, 0, 0, 1, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 1,
        1, 0, 1, 1, 0, 0, 0, 1, 0, 1, 0, 1, 1, 1, 0, 1, 1, 0, 0, 1, 0, 0,
        0, 0, 1, 0, 0, 0, 1, 0, 1, 0, 1, 1, 0, 1, 0, 0, 0, 0, 1, 1, 0, 0,
        1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 0, 1, 1, 0, 1, 1, 0, 0, 1, 0, 1, 1,
        1, 1, 0, 1, 1, 1, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1, 1, 0, 1, 1, 0, 1, 0, 0, 1, 1,
        1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 0, 1, 0, 1, 1, 1, 1, 1,
        1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 0, 1, 1, 1, 1, 0, 0,
        0, 1, 1, 1, 1, 0, 1, 0, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0,
        0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 0, 0, 0, 1, 0, 0,
        1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 0, 0, 1, 1,
        1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 0,
        1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 0, 0, 1, 0, 1, 1, 1, 1,
        1, 0, 1, 1, 0, 1, 0, 1, 1, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0,
        1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1,
        1, 1, 1, 0, 1, 0, 1, 1, 0, 1, 1, 1, 1, 1, 0, 0, 1, 0, 1, 0, 1, 1,
        1, 1, 1, 0, 1, 1, 0, 1, 0, 1, 0, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1,
        1, 1, 1, 1, 1, 0, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
        1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 1]),
 'frame': None,
 'target_names': array(['malignant', 'benign'], dtype='<U9'),
 'DESCR': '.. _breast_cancer_dataset:\n\nBreast cancer wisconsin (diagnostic) d
ataset\n--------------------------------------------\n\n**Data Set Characterist
ics:**\n\n    :Number of Instances: 569\n\n    :Number of Attributes: 30 numeri
c, predictive attributes and the class\n\n    :Attribute Information:\n
- radius (mean of distances from center to points on the perimeter)\n        -
texture (standard deviation of gray-scale values)\n        - perimeter\n
- area\n        - smoothness (local variation in radius lengths)\n        - com
pactness (perimeter^2 / area - 1.0)\n        - concavity (severity of concave p
ortions of the contour)\n        - concave points (number of concave portions o
f the contour)\n        - symmetry\n        - fractal dimension ("coastline app
roximation" - 1)\n\n        The mean, standard error, and "worst" or largest (m
ean of the three\n        worst/largest values) of these features were computed
```

```
for each image,\n        resulting in 30 features.  For instance, field 0 is Me
an Radius, field\n        10 is Radius SE, field 20 is Worst Radius.\n\n
- class:\n                - WDBC-Malignant\n                - WDBC-Benign\n\n
:Summary Statistics:\n\n    ===================================== ====== ======
\n                                           Min    Max\n    ==================
=================== ====== ======\n    radius (mean):                        6.
981  28.11\n    texture (mean):                       9.71   39.28\n    perimet
er (mean):                     43.79  188.5\n    area (mean):
143.5  2501.0\n    smoothness (mean):                   0.053  0.163\n    comp
actness (mean):                 0.019  0.345\n    concavity (mean):
0.0    0.427\n    concave points (mean):              0.0    0.201\n    symme
try (mean):                    0.106  0.304\n    fractal dimension (mean):
0.05   0.097\n    radius (standard error):            0.112  2.873\n    textu
re (standard error):           0.36   4.885\n    perimeter (standard error):
0.757  21.98\n    area (standard error):              6.802  542.2\n    smoot
hness (standard error):        0.002  0.031\n    compactness (standard erro
r):          0.002  0.135\n    concavity (standard error):            0.0    0.39
6\n    concave points (standard error):     0.0    0.053\n    symmetry (standa
rd error):           0.008  0.079\n    fractal dimension (standard error):
0.001  0.03\n    radius (worst):                     7.93   36.04\n    textur
e (worst):                     12.02  49.54\n    perimeter (worst):
50.41  251.2\n    area (worst):                       185.2  4254.0\n    smoo
thness (worst):                 0.071  0.223\n    compactness (worst):
0.027  1.058\n    concavity (worst):                  0.0    1.252\n    conca
ve points (worst):             0.0    0.291\n    symmetry (worst):
0.156  0.664\n    fractal dimension (worst):          0.055  0.208\n    =====
================================ ====== ======\n\n    :Missing Attribute Value
s: None\n\n    :Class Distribution: 212 - Malignant, 357 - Benign\n\n    :Creat
or:  Dr. William H. Wolberg, W. Nick Street, Olvi L. Mangasarian\n\n    :Donor:
Nick Street\n\n    :Date: November, 1995\n\nThis is a copy of UCI ML Breast Can
cer Wisconsin (Diagnostic) datasets.\nhttps://goo.gl/U2Uwz2\n\nFeatures are com
puted from a digitized image of a fine needle\naspirate (FNA) of a breast mass.
They describe\ncharacteristics of the cell nuclei present in the image.\n\nSepa
rating plane described above was obtained using\nMultisurface Method-Tree (MSM-
T) [K. P. Bennett, "Decision Tree\nConstruction Via Linear Programming." Procee
dings of the 4th\nMidwest Artificial Intelligence and Cognitive Science Societ
y,\npp. 97-101, 1992], a classification method which uses linear\nprogramming t
o construct a decision tree.  Relevant features\nwere selected using an exhaust
ive search in the space of 1-4\nfeatures and 1-3 separating planes.\n\nThe actu
al linear program used to obtain the separating plane\nin the 3-dimensional spa
ce is that described in:\n[K. P. Bennett and O. L. Mangasarian: "Robust Linear
\nProgramming Discrimination of Two Linearly Inseparable Sets",\nOptimization M
ethods and Software 1, 1992, 23-34].\n\nThis database is also available through
the UW CS ftp server:\n\nftp ftp.cs.wisc.edu\ncd math-prog/cpo-dataset/machine-
learn/WDBC/\n\n.. topic:: References\n\n   - W.N. Street, W.H. Wolberg and O.L.
Mangasarian. Nuclear feature extraction \n     for breast tumor diagnosis. IS&
T/SPIE 1993 International Symposium on \n     Electronic Imaging: Science and T
echnology, volume 1905, pages 861-870,\n     San Jose, CA, 1993.\n   - O.L. Man
gasarian, W.N. Street and W.H. Wolberg. Breast cancer diagnosis and \n     prog
nosis via linear programming. Operations Research, 43(4), pages 570-577, \n
July-August 1995.\n   - W.H. Wolberg, W.N. Street, and O.L. Mangasarian. Machin
e learning techniques\n     to diagnose breast cancer from fine-needle aspirate
s. Cancer Letters 77 (1994) \n     163-171.',
 'feature_names': array(['mean radius', 'mean texture', 'mean perimeter', 'mean
area',
        'mean smoothness', 'mean compactness', 'mean concavity',
        'mean concave points', 'mean symmetry', 'mean fractal dimension',
        'radius error', 'texture error', 'perimeter error', 'area error',
        'smoothness error', 'compactness error', 'concavity error',
        'concave points error', 'symmetry error',
        'fractal dimension error', 'worst radius', 'worst texture',
        'worst perimeter', 'worst area', 'worst smoothness',
        'worst compactness', 'worst concavity', 'worst concave points',
        'worst symmetry', 'worst fractal dimension'], dtype='<U23'),
 'filename': 'C:\\Users\\admin\\anaconda3\\lib\\site-packages\\sklearn\\dataset
s\\data\\breast_cancer.csv'}
```

In [5]:

```
cancer_data_df = pd.DataFrame(data=cancer_data.data, columns=cancer_data.feature_
cancer_data_df['Target'] = cancer_data.target
cancer_data_df
```

Out[5]:

	mean radius	mean texture	mean perimeter	mean area	mean smoothness	mean compactness	mean concavity	mean concave points	mean symmetry
0	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.30010	0.14710	0.2419
1	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.08690	0.07017	0.1812
2	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.19740	0.12790	0.2069
3	11.42	20.38	77.58	386.1	0.14250	0.28390	0.24140	0.10520	0.2597
4	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.19800	0.10430	0.1809
...	...	...	...	...	...	...	...	...	...
564	21.56	22.39	142.00	1479.0	0.11100	0.11590	0.24390	0.13890	0.1726
565	20.13	28.25	131.20	1261.0	0.09780	0.10340	0.14400	0.09791	0.1752
566	16.60	28.08	108.30	858.1	0.08455	0.10230	0.09251	0.05302	0.1590
567	20.60	29.33	140.10	1265.0	0.11780	0.27700	0.35140	0.15200	0.2397
568	7.76	24.54	47.92	181.0	0.05263	0.04362	0.00000	0.00000	0.1587

569 rows × 31 columns

3. Data Understanding

In [6]:

```
cancer_data_df.shape
```

Out[6]:

```
(569, 31)
```

In [7]:
```
cancer_data_df.isnull().sum()
```

Out[7]:
```
mean radius                0
mean texture               0
mean perimeter             0
mean area                  0
mean smoothness            0
mean compactness           0
mean concavity             0
mean concave points        0
mean symmetry              0
mean fractal dimension     0
radius error               0
texture error              0
perimeter error            0
area error                 0
smoothness error           0
compactness error          0
concavity error            0
concave points error       0
symmetry error             0
fractal dimension error    0
worst radius               0
worst texture              0
worst perimeter            0
worst area                 0
worst smoothness           0
worst compactness          0
worst concavity            0
worst concave points       0
worst symmetry             0
worst fractal dimension    0
Target                     0
dtype: int64
```

In [8]:
```
cancer_data_df.dtypes
```

Out[8]:
```
mean radius                float64
mean texture               float64
mean perimeter             float64
mean area                  float64
mean smoothness            float64
mean compactness           float64
mean concavity             float64
mean concave points        float64
mean symmetry              float64
mean fractal dimension     float64
radius error               float64
texture error              float64
perimeter error            float64
area error                 float64
smoothness error           float64
compactness error          float64
concavity error            float64
concave points error       float64
symmetry error             float64
fractal dimension error    float64
worst radius               float64
worst texture              float64
worst perimeter            float64
worst area                 float64
worst smoothness           float64
worst compactness          float64
worst concavity            float64
worst concave points       float64
worst symmetry             float64
worst fractal dimension    float64
Target                       int32
dtype: object
```

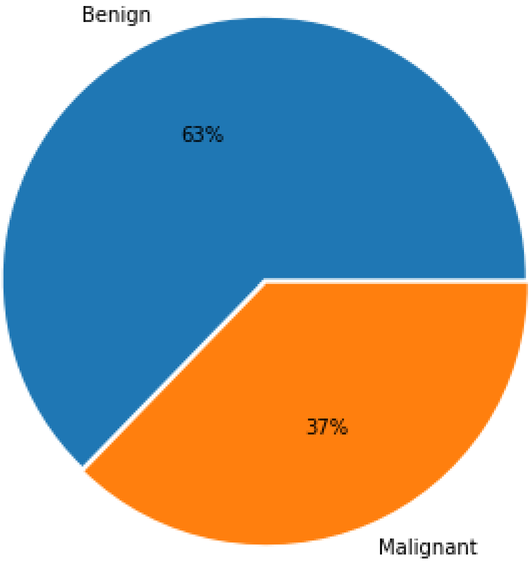
4. Model Building

In [9]:
```
X = cancer_data_df.drop(labels='Target',axis = 1)
y = cancer_data_df[['Target']]
```

In [11]:
```
y.value_counts() # 1: Malignant, 0: Benign
```

Out[11]:
```
Target
1         357
0         212
dtype: int64
```

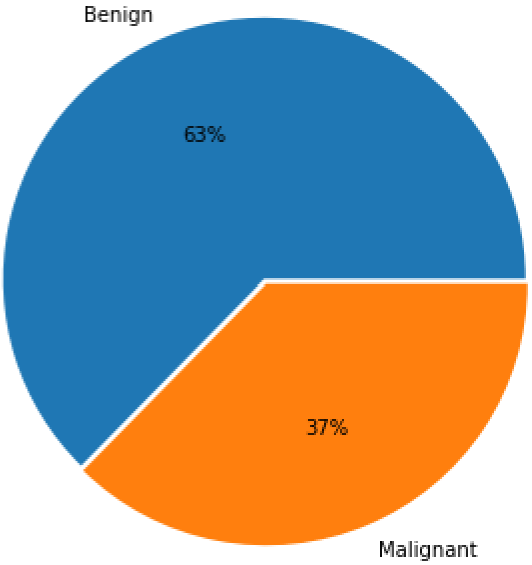
In [25]:
```
from matplotlib import pyplot as plt
plt.figure(figsize=(6,6))
plt.pie(x = y.value_counts(),labels=['Benign','Malignant'], explode=[0.02,0],autc
plt.show()
```



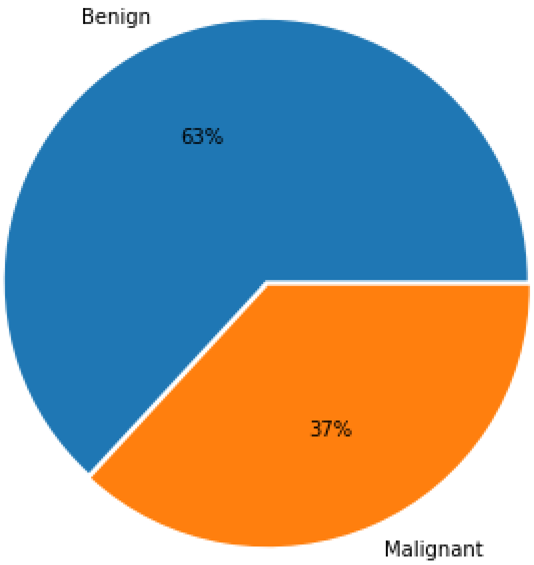
In [34]:
```
X_train,X_test,y_train,y_test = train_test_split(X,y,test_size=0.20,random_state=
```

In [21]:
```
#Stratify will be used by the train_test_split() function to ensure that both the
#examples in each class that is present in the provided “y” array
```

```
In [35]: from matplotlib import pyplot as plt      # Train Data
         plt.figure(figsize=(6,6))
         plt.pie(x = y_train.value_counts(),labels=['Benign','Malignant'], explode=[0.02,0
         plt.show()
```



In [36]:
```
from matplotlib import pyplot as plt        #Test Data
plt.figure(figsize=(6,6))
plt.pie(x = y_test.value_counts(),labels=['Benign','Malignant'], explode=[0.02,0
plt.show()
```



In [37]:
```
X_train.shape,y_train.shape
```

Out[37]:
```
((455, 30), (455, 1))
```

In [38]:
```
X_test.shape,y_test.shape
```

Out[38]:
```
((114, 30), (114, 1))
```

5. Model Training

In [42]:
```
adb_classifier = AdaBoostClassifier()
adb_classifier.fit(X_train,y_train)
```

Out[42]:
```
AdaBoostClassifier()
```

6. Model Testing || 7. Model Evaluation

Training Data

In [44]:

```
y_pred_train = adb_classifier.predict(X_train)
print(accuracy_score(y_train,y_pred_train))
print(confusion_matrix(y_train,y_pred_train))
```

```
1.0
[[170   0]
 [  0 285]]
```

Test Data

In [45]:

```
y_pred_test = adb_classifier.predict(X_test)
print(accuracy_score(y_test,y_pred_test))
print(confusion_matrix(y_test,y_pred_test))
```

```
0.9824561403508771
[[41  1]
 [ 1 71]]
```

---||--

How to Handle Imbalance Dataset and decrease FP/FN??

In [47]:

```
dt_model = DecisionTreeClassifier(max_depth=3,class_weight={0:1.5,1:1})
dt_model.fit(X_train,y_train)
```

Out[47]:

```
DecisionTreeClassifier(class_weight={0: 1.5, 1: 1}, max_depth=3)
```

6. Model Testing || 7. Model Evaluation

Train data

In [48]:

```
y_pred_train = dt_model.predict(X_train)
print(accuracy_score(y_train,y_pred_train))
print(confusion_matrix(y_train,y_pred_train))
```

```
0.978021978021978
[[164   6]
 [  4 281]]
```

```
In [49]: from sklearn.metrics import classification_report
         print(classification_report(y_train,y_pred_train))
```

```
              precision    recall  f1-score   support

           0       0.98      0.96      0.97       170
           1       0.98      0.99      0.98       285

    accuracy                           0.98       455
   macro avg       0.98      0.98      0.98       455
weighted avg       0.98      0.98      0.98       455
```

========================THE END==========================