In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

In [2]:
```
data = pd.read_csv(r"C:\Users\ASUS\Downloads\archive (2)\survey lung cancer.csv")
```

In [3]:
```
data.head()
```

Out[3]:

	GENDER	AGE	SMOKING	YELLOW_FINGERS	ANXIETY	PEER_PRESSURE	CHRONIC DISEASE	FATIGUE	ALLERGY	WHEEZING	AL CONS
0	M	69	1	2	2	1	1	2	1	2	
1	M	74	2	1	1	1	2	2	2	1	
2	F	59	1	1	1	2	1	2	1	2	
3	M	63	2	2	2	1	1	1	1	1	
4	F	63	1	2	1	1	1	1	1	2	

In [4]:
```
data.tail()
```

Out[4]:

	GENDER	AGE	SMOKING	YELLOW_FINGERS	ANXIETY	PEER_PRESSURE	CHRONIC DISEASE	FATIGUE	ALLERGY	WHEEZING	CO
304	F	56	1	1	1	2	2	2	1	1	
305	M	70	2	1	1	1	1	2	2	2	
306	M	58	2	1	1	1	1	1	2	2	
307	M	67	2	1	2	1	1	2	2	1	
308	M	62	1	1	1	2	1	2	2	2	

In [5]:
```
data.sample(20)
```

Out[5]:

	GENDER	AGE	SMOKING	YELLOW_FINGERS	ANXIETY	PEER_PRESSURE	CHRONIC DISEASE	FATIGUE	ALLERGY	WHEEZING	CO
28	F	53	2	2	2	1	2	1	1	2	
185	M	77	2	1	1	1	1	1	2	2	
163	M	68	2	1	1	2	2	2	2	2	
257	M	64	2	1	1	1	1	1	2	2	
283	M	60	1	2	2	1	1	2	1	2	
30	F	57	2	2	1	1	1	1	1	1	
92	M	52	2	1	1	1	2	2	2	2	
200	F	63	1	1	1	2	1	1	1	2	
50	F	56	1	1	1	2	2	2	2	2	
210	M	54	2	1	1	1	1	1	2	2	
247	M	67	1	2	1	1	1	2	1	2	
229	M	57	1	1	1	1	2	1	2	1	
157	F	57	2	2	1	2	1	1	1	1	
219	M	70	1	1	1	1	2	2	2	1	
111	M	61	2	2	2	1	1	2	2	1	
98	M	64	1	2	2	2	1	2	2	1	
75	M	58	2	1	1	1	1	1	2	2	
106	F	61	2	2	2	2	2	2	1	1	
139	M	63	1	2	1	1	1	2	1	2	
40	M	68	2	1	2	1	1	2	2	1	

In [6]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 309 entries, 0 to 308
Data columns (total 16 columns):
 #   Column                 Non-Null Count  Dtype
---  ------                 --------------  -----
 0   GENDER                 309 non-null    object
 1   AGE                    309 non-null    int64
 2   SMOKING                309 non-null    int64
 3   YELLOW_FINGERS         309 non-null    int64
 4   ANXIETY                309 non-null    int64
 5   PEER_PRESSURE          309 non-null    int64
 6   CHRONIC DISEASE        309 non-null    int64
 7   FATIGUE                309 non-null    int64
 8   ALLERGY                309 non-null    int64
 9   WHEEZING               309 non-null    int64
 10  ALCOHOL CONSUMING      309 non-null    int64
 11  COUGHING               309 non-null    int64
 12  SHORTNESS OF BREATH    309 non-null    int64
 13  SWALLOWING DIFFICULTY  309 non-null    int64
 14  CHEST PAIN             309 non-null    int64
 15  LUNG_CANCER            309 non-null    object
dtypes: int64(14), object(2)
memory usage: 38.8+ KB
```

In [7]:
```
data.describe()
```

Out[7]:

	AGE	SMOKING	YELLOW_FINGERS	ANXIETY	PEER_PRESSURE	CHRONIC DISEASE	FATIGUE	ALLERGY	WHEEZING
count	309.000000	309.000000	309.000000	309.000000	309.000000	309.000000	309.000000	309.000000	309.000000
mean	62.673139	1.563107	1.569579	1.498382	1.501618	1.504854	1.673139	1.556634	1.556634
std	8.210301	0.496806	0.495938	0.500808	0.500808	0.500787	0.469827	0.497588	0.497588
min	21.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000
25%	57.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000
50%	62.000000	2.000000	2.000000	1.000000	2.000000	2.000000	2.000000	2.000000	2.000000
75%	69.000000	2.000000	2.000000	2.000000	2.000000	2.000000	2.000000	2.000000	2.000000
max	87.000000	2.000000	2.000000	2.000000	2.000000	2.000000	2.000000	2.000000	2.000000

In [8]:
```
data.shape
```

Out[8]:
```
(309, 16)
```

In [9]:
```
data.size
```

Out[9]:
```
4944
```

In [10]:
```
data.isnull().sum()
```

Out[10]:
```
GENDER                   0
AGE                      0
SMOKING                  0
YELLOW_FINGERS           0
ANXIETY                  0
PEER_PRESSURE            0
CHRONIC DISEASE          0
FATIGUE                  0
ALLERGY                  0
WHEEZING                 0
ALCOHOL CONSUMING        0
COUGHING                 0
SHORTNESS OF BREATH      0
SWALLOWING DIFFICULTY    0
CHEST PAIN               0
LUNG_CANCER              0
dtype: int64
```

In [11]:
```
data.GENDER.value_counts()
```

Out[11]:
```
GENDER
M    162
F    147
Name: count, dtype: int64
```

In [12]:
```
data.LUNG_CANCER.value_counts()
```

Out[12]:
```
LUNG_CANCER
YES    270
NO      39
Name: count, dtype: int64
```

In [13]:
```
No = data[data.LUNG_CANCER == "NO"]
Yes = data[data.LUNG_CANCER == "YES"]
```

In [14]:
```
print(No.shape)
print(Yes.shape)
```

```
(39, 16)
(270, 16)
```

In [15]:
```
Yes_sample = Yes.sample(n=39)
Yes_sample.shape
```

Out[15]:
```
(39, 16)
```

In [16]:
```
newdata = pd.concat([Yes_sample, No], axis=0)
```

In [17]:
```
newdata.head()
```

Out[17]:

	GENDER	AGE	SMOKING	YELLOW_FINGERS	ANXIETY	PEER_PRESSURE	CHRONIC DISEASE	FATIGUE	ALLERGY	WHEEZING	CO
112	F	68	1	1	1	2	1	2	1	2	
185	M	77	2	1	1	1	1	1	2	2	
285	F	58	2	2	2	2	1	2	1	1	
36	M	60	1	2	1	1	2	1	1	2	
264	M	70	2	1	1	1	1	2	1	2	

In [18]:
```
newdata.sample(15)
```

Out[18]:

	GENDER	AGE	SMOKING	YELLOW_FINGERS	ANXIETY	PEER_PRESSURE	CHRONIC DISEASE	FATIGUE	ALLERGY	WHEEZING	CO
303	M	51	1	2	1	1	2	2	2	2	
264	M	70	2	1	1	1	1	2	1	2	
49	M	60	1	1	2	2	2	1	1	1	
208	M	67	1	2	2	2	1	2	2	1	
150	M	67	2	1	2	1	1	2	2	1	
156	M	47	2	2	1	1	2	1	1	1	
270	F	70	2	1	1	1	1	2	1	1	
229	M	57	1	1	1	1	2	1	2	1	
231	M	64	2	2	2	2	1	2	2	1	
154	F	64	2	2	1	2	2	1	1	1	
2	F	59	1	1	1	2	1	2	1	2	
202	M	74	2	1	1	1	2	2	2	2	
4	F	63	1	2	1	1	1	1	1	2	
159	M	68	1	1	2	2	2	1	1	1	
32	M	56	2	2	2	1	1	1	1	1	

In [19]:
```
newdata.LUNG_CANCER.value_counts()
```

Out[19]:
```
LUNG_CANCER
YES    39
NO     39
Name: count, dtype: int64
```

In [20]:
```
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
newdata.GENDER = le.fit_transform(newdata.GENDER)
newdata.head()
```

Out[20]:

	GENDER	AGE	SMOKING	YELLOW_FINGERS	ANXIETY	PEER_PRESSURE	CHRONIC DISEASE	FATIGUE	ALLERGY	WHEEZING	CO
112	0	68	1	1	1	2	1	2	1	2	
185	1	77	2	1	1	1	1	1	2	2	
285	0	58	2	2	2	2	1	2	1	1	
36	1	60	1	2	1	1	2	1	1	2	
264	1	70	2	1	1	1	1	2	1	2	

In [21]:
```
'''
few informations
M = 1
F = 0
from Labelencoder
'''
```

Out[21]:
```
'\nfew informations\nM = 1\nF = 0\nfrom Labelencoder\n'
```

In [22]:
```
X=newdata.drop("LUNG_CANCER" , axis="columns")
X.head()
```

Out[22]:

	GENDER	AGE	SMOKING	YELLOW_FINGERS	ANXIETY	PEER_PRESSURE	CHRONIC DISEASE	FATIGUE	ALLERGY	WHEEZING	CO
112	0	68	1	1	1	2	1	2	1	2	
185	1	77	2	1	1	1	1	1	2	2	
285	0	58	2	2	2	2	1	2	1	1	
36	1	60	1	2	1	1	2	1	1	2	
264	1	70	2	1	1	1	1	2	1	2	

In [23]:
```
X.shape
```

Out[23]:
```
(78, 15)
```

In [24]:
```
Y=newdata['LUNG_CANCER'].values
```

In [25]:
```
Y
```

Out[25]:
```
array(['YES', 'YES', 'YES', 'YES', 'YES', 'YES', 'YES', 'YES', 'YES',
       'YES', 'YES', 'YES', 'YES', 'YES', 'YES', 'YES', 'YES', 'YES',
       'YES', 'YES', 'YES', 'YES', 'YES', 'YES', 'YES', 'YES', 'YES',
       'YES', 'YES', 'YES', 'YES', 'YES', 'YES', 'YES', 'YES', 'YES',
       'YES', 'YES', 'YES', 'NO', 'NO', 'NO', 'NO', 'NO', 'NO', 'NO',
       'NO', 'NO', 'NO', 'NO', 'NO', 'NO', 'NO', 'NO', 'NO', 'NO', 'NO',
       'NO', 'NO', 'NO', 'NO', 'NO', 'NO', 'NO', 'NO', 'NO', 'NO', 'NO',
       'NO', 'NO', 'NO', 'NO', 'NO', 'NO', 'NO', 'NO', 'NO', 'NO'],
      dtype=object)
```

In [26]:
```
print(Y)
```

```
['YES' 'YES' 'YES' 'YES' 'YES' 'YES' 'YES' 'YES' 'YES' 'YES' 'YES' 'YES'
 'YES' 'YES' 'YES' 'YES' 'YES' 'YES' 'YES' 'YES' 'YES' 'YES' 'YES' 'YES'
 'YES' 'YES' 'YES' 'YES' 'YES' 'YES' 'YES' 'YES' 'YES' 'YES' 'YES' 'YES'
 'YES' 'YES' 'YES' 'NO' 'NO' 'NO' 'NO' 'NO' 'NO' 'NO' 'NO' 'NO' 'NO' 'NO'
 'NO' 'NO' 'NO' 'NO' 'NO' 'NO' 'NO' 'NO' 'NO' 'NO' 'NO' 'NO' 'NO' 'NO' 'NO'
 'NO' 'NO' 'NO' 'NO' 'NO' 'NO' 'NO' 'NO' 'NO' 'NO' 'NO' 'NO' 'NO' 'NO' 'NO']
```

In [27]:
```
Y.shape
```

Out[27]:
```
(78,)
```

In [28]:
```
from sklearn.preprocessing import LabelEncoder
le2 = LabelEncoder()
Y = le2.fit_transform(Y)
```

In [29]:
```
Y
```

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

In [30]:
```
'''
few informations
```

```
YES = 1
NO = 0
from labelencoder
'''
```

Out[30]:
```
'\nfew informations\nYES = 1\nNO = 0\nfrom labelencoder\n'
```

In [31]:
```
from sklearn.model_selection import train_test_split
X_train , X_test , y_train , y_test = train_test_split(X , Y , test_size= 0.1)
```

In [32]:
```
len(X_train)
```

Out[32]:
```
70
```

In [33]:
```
len(X_test)
```

Out[33]:
```
8
```

In [34]:
```
len(y_train)
```

Out[34]:
```
70
```

In [35]:
```
len(y_test)
```

Out[35]:
```
8
```

In [36]:
```
X_train.shape
```

Out[36]:
```
(70, 15)
```

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

Out[37]:
```
(70,)
```

In [38]:
```
X_train.head()
```

Out[38]:

	GENDER	AGE	SMOKING	YELLOW_FINGERS	ANXIETY	PEER_PRESSURE	CHRONIC DISEASE	FATIGUE	ALLERGY	WHEEZING	CO
146	1	51	1	2	1	1	2	2	2	2	
141	1	62	2	1	2	1	1	2	1	2	
267	1	60	2	2	2	2	2	1	2	1	
36	1	60	1	2	1	1	2	1	1	2	
34	1	59	1	2	2	1	1	1	1	1	

In [39]:
```
y_train
```

Out[39]:
```
array([1, 1, 1, 1, 0, 1, 1, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 1, 0,
       0, 1, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0, 1, 1, 0, 1, 1, 0, 0, 0, 0,
       0, 1, 1, 1, 1, 0, 0, 1, 1, 0, 1, 1, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1,
       1, 1, 0, 1])
```

In [40]:
```
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
```

In [41]:
```
lrmodel = LogisticRegression()
lrmodel.fit(X_train , y_train)
```

```
C:\Users\ASUS\AppData\Roaming\Python\Python311\site-packages\sklearn\linear_model\_logistic.py:460: ConvergenceW
arning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
```

Out[41]:
```
▾ LogisticRegression
LogisticRegression()
```

In [42]:
```
lrmodel.predict(X_test)
```

Out[42]:
```
array([0, 0, 1, 1, 1, 1, 1, 1])
```

In [43]:
```
from sklearn.metrics import accuracy_score
X_train_prediction = lrmodel.predict(X_train)
training_data_accuracy = accuracy_score(X_train_prediction , y_train)
```

In [44]:
```
training_data_accuracy
```

Out[44]:
```
0.9428571428571428
```

In [45]:
```
from sklearn.metrics import accuracy_score
X_test_prediction = lrmodel.predict(X_test)
testing_data_accuracy = accuracy_score(X_test_prediction , y_test)
```

In [46]:
```
testing_data_accuracy
```

Out[46]:
```
0.875
```

In [47]:
```
dtmodel = DecisionTreeClassifier(random_state=42)
dtmodel.fit(X_train, y_train)
```

Out[47]:
```
DecisionTreeClassifier(random_state=42)
```

In [48]:
```
from sklearn.metrics import accuracy_score
X_train_prediction = dtmodel.predict(X_train)
training_data_accuracy = accuracy_score(X_train_prediction , y_train)
```

In [49]:
```
training_data_accuracy
```

Out[49]:
```
1.0
```

In [50]:
```
from sklearn.metrics import accuracy_score
X_test_prediction = dtmodel.predict(X_test)
testing_data_accuracy = accuracy_score(X_test_prediction , y_test)
```

In [51]:
```
testing_data_accuracy
```

Out[51]:
```
0.875
```

In [53]:
```
rfmodel = RandomForestClassifier(n_estimators=100, random_state=42)
rfmodel.fit(X_train, y_train)
```

Out[53]:
```
RandomForestClassifier(random_state=42)
```

In [54]:
```
from sklearn.metrics import accuracy_score
X_train_prediction = rfmodel.predict(X_train)
training_data_accuracy = accuracy_score(X_train_prediction , y_train)
```

In [55]:
```
training_data_accuracy
```

Out[55]:
```
1.0
```

In [56]:
```
from sklearn.metrics import accuracy_score
X_test_prediction = rfmodel.predict(X_test)
testing_data_accuracy = accuracy_score(X_test_prediction , y_test)
```

In [57]:
```
testing_data_accuracy
```

Out[57]:
```
0.875
```

In [58]:
```
from sklearn.svm import SVC
svcmodel = SVC(kernel='linear')  # You can also use 'rbf', 'poly', etc.
svcmodel.fit(X_train, y_train)
```

Out[58]:
```
SVC(kernel='linear')
```

In [59]:
```
from sklearn.metrics import accuracy_score
X_train_prediction = svcmodel.predict(X_train)
training_data_accuracy = accuracy_score(X_train_prediction , y_train)
```

In [60]:
```
training_data_accuracy
```

Out[60]:
```
0.9142857142857143
```

In [61]:
```
from sklearn.metrics import accuracy_score
X_test_prediction = svcmodel.predict(X_test)
testing_data_accuracy = accuracy_score(X_test_prediction , y_test)
```

In [62]:
```
testing_data_accuracy
```

Out[62]:
```
0.875
```

In [63]:
```
'''
Gender: M(male), F(female)
Age: Age of the patient
Smoking: YES=2 , NO=1.
Yellow fingers: YES=2 , NO=1.
Anxiety: YES=2 , NO=1.
Peer_pressure: YES=2 , NO=1.
Chronic Disease: YES=2 , NO=1.
Fatigue: YES=2 , NO=1.
Allergy: YES=2 , NO=1.
Wheezing: YES=2 , NO=1.
Alcohol: YES=2 , NO=1.
Coughing: YES=2 , NO=1.
Shortness of Breath: YES=2 , NO=1.
Swallowing Difficulty: YES=2 , NO=1.
Chest pain: YES=2 , NO=1.
'''
```

Out[63]:
```
'\nGender: M(male), F(female)\nAge: Age of the patient\nSmoking: YES=2 , NO=1.\nYellow fingers: YES=2 , NO=1.\n
Anxiety: YES=2 , NO=1.\nPeer_pressure: YES=2 , NO=1.\nChronic Disease: YES=2 , NO=1.\nFatigue: YES=2 , NO=1.\nA
llergy: YES=2 , NO=1.\nWheezing: YES=2 , NO=1.\nAlcohol: YES=2 , NO=1.\nCoughing: YES=2 , NO=1.\nShortness of B
reath: YES=2 , NO=1.\nSwallowing Difficulty: YES=2 , NO=1.\nChest pain: YES=2 , NO=1.\n'
```

In [64]:
```
svcmodel.predict([[1,69,1,2,2,1,1,2,1,2,2,2,2,2,2]])
```

```
C:\Users\ASUS\AppData\Roaming\Python\Python311\site-packages\sklearn\base.py:465: UserWarning: X does not have v
alid feature names, but SVC was fitted with feature names
  warnings.warn(
```

Out[64]:
```
array([1])
```

In [65]:
```
input_data = (1,69,1,2,2,1,1,2,1,2,2,2,2,2,2)
input_data_as_numpy_array = np.asarray(input_data)
input_data_reshaped = input_data_as_numpy_array.reshape(1 , -1)
prediction = rfmodel.predict(input_data_reshaped)
if prediction == [0]:
    print("The person does not have Lung Cancer.")
else:
    print("The person have Lung Cancer.")
```

```
The person have Lung Cancer.
```

```
C:\Users\ASUS\AppData\Roaming\Python\Python311\site-packages\sklearn\base.py:465: UserWarning: X does not have v
alid feature names, but RandomForestClassifier was fitted with feature names
  warnings.warn(
```

In [66]:
```
import pickle
with open("Lung_cancer_model_pickle" , "wb") as file:
     pickle.dump(svcmodel , file)
```

In [67]:
```
import numpy as np
import pickle
with open("Lung_cancer_model_pickle" , "rb") as file:
     svcmodel = pickle.load(file)
```

In [68]:
```
input_data = (1,69,1,2,2,1,1,2,1,2,2,2,2,2,2)
input_data_as_numpy_array = np.asarray(input_data)
input_data_reshaped = input_data_as_numpy_array.reshape(1 , -1)
prediction = rfmodel.predict(input_data_reshaped)
if prediction == [0]:
    print("The person does not have Lung Cancer.")
else:
    print("The person have Lung Cancer.")
```

```
The person have Lung Cancer.
```

```
C:\Users\ASUS\AppData\Roaming\Python\Python311\site-packages\sklearn\base.py:465: UserWarning: X does not have v
alid feature names, but RandomForestClassifier was fitted with feature names
  warnings.warn(
```

In [69]:
```
#Gender(0 for Female, 1 for Male)
gender = int(input(""))
#Age in years
age = int(input(""))
```

```
#Smoking: YES=2 , NO=1
smoking = int(input(""))
#Yellow_fingers: YES=2 , NO=1
yellow_fingers = int(input(""))
#Anxiety: YES=2 , NO=1
anxiety = int(input(""))
#Peer_pressure: YES=2 , NO=1
peer_pressure = int(input(""))
#Chronic_Disease: YES=2 , NO=1
chronic_Disease = int(input(""))
#Fatigue: YES=2 , NO=1
fatigue = int(input(""))
#Allergy: YES=2 , NO=1
allergy = int(input(""))
#Wheezing: YES=2 , NO=1
wheezing = int(input(""))
#Alcohol: YES=2 , NO=1
alcohol = int(input(""))
#Coughing: YES=2 , NO=1
coughing = int(input(""))
#Shortness_of_Breath: YES=2 , NO=1
shortness_of_Breath = int(input(""))
#Swallowing_Difficulty: YES=2 , NO=1
swallowing_Difficulty = int(input(""))
#Chest_pain: YES=2 , NO=1
chest_pain = int(input(""))
```

In [70]:

```
input_data = (gender, age, smoking, yellow_fingers, anxiety, peer_pressure, chronic_Disease, fatigue, allergy, v
input_data_as_numpy_array = np.asarray(input_data)
input_data_reshaped = input_data_as_numpy_array.reshape(1 , -1)
prediction = rfmodel.predict(input_data_reshaped)
if prediction == [0]:
    print("The person does not have Lung Cancer.")
else:
    print("The person have Lung Cancer.")
```

```
The person have Lung Cancer.
C:\Users\ASUS\AppData\Roaming\Python\Python311\site-packages\sklearn\base.py:465: UserWarning: X does not have v
alid feature names, but RandomForestClassifier was fitted with feature names
  warnings.warn(
```

In [ ]: