SHRI VAISHNAV VIDYAPEETH VISHWAVIDHYALAYA , INDORE

SHRI VAISHANV INSTITUTE OF TECHNOLOGY ,INDORE

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

JUL – DEC 2022

PROJECT SYNOPSIS

ON

LUNG CANCER DETECTION USING ML

For the subject

Introduction to Data Science [BTIBM505]

Submitted By:

Gautam Jaiswal [20100BTCSAII07160]
Ojas Khatavkar [20100BTCSAII07178]
Sophiya saleema Qureshi [20100BTCSAII07190]
Kamakshi sharma [20100BTCSAII07170]
III year /Vth Semester
Section D (AI)

Under the guidance of:

Mr.Om Kant sharma

Lung cancer detection

Lung cancer is a potentially lethal illness. Cancer detection continues to be a challenge for medical professionals. The true cause of cancer and its complete treatment have still not been discovered. Cancer that is caught early enough can be treated. Image processing methods such as noise reduction, feature extraction, identification of damaged regions, and maybe a comparison with data on the medical history of lung cancer are used to locate portions of the lung that have been impacted by cancer.

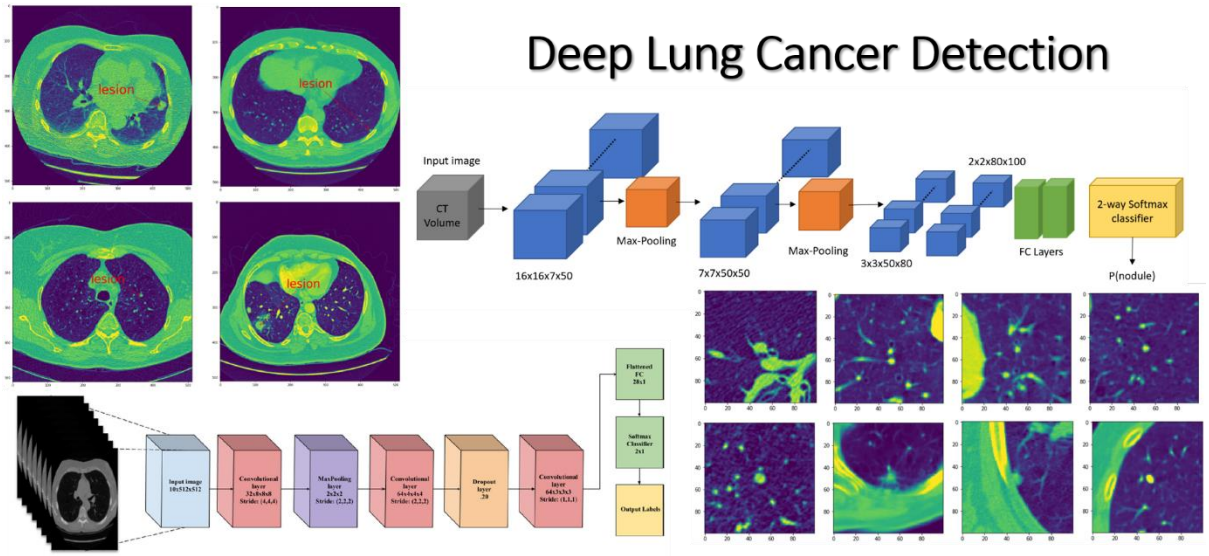
This research shows an accurate classification and prediction of lung cancer using technology that is enabled by machine learning and image processing.

Introduction

One of the most lethal types of the disease, lung cancer, is responsible for the passing away of about one million people every year. The current state of affairs in the world of medicine makes it absolutely essential to perform lung nodule identification on chest CT scans. This is due to the fact that lung nodules are becoming increasingly common. As a direct result of this, the deployment of CAD systems is required in order to accomplish the objective of early lung cancer identification.

When doing a CT scan, sophisticated X-ray equipment is utilized in order to capture images of the human body from a number of different angles. Following this, the images are fed into a computer, which processes them in such a way as to produce a cross-sectional view of the internal organs and tissues of the body.

If lung cancer is detected at an early stage, the American Cancer Society estimates that a patient has a 47 percent chance of surviving the disease. It is quite unlikely that X-ray pictures may accidentally reveal lung cancer in its earlier stages. It is famously difficult to detect lesions that are round and have a diameter of 510 millimetres or less.



Problem

Lung cancer detection using machine learning is done by classification.

Lung cancer is the major cause of cancer-related death in this generation, and it is expected to remain so for the foreseeable future. It is feasible to treat lung cancer if the symptoms of the disease are detected early. It is possible to construct a sustainable prototype model for the treatment of lung cancer using the current developments in computational intelligence without negatively impacting the environment. Because it will reduce the number of resources squandered as well as the amount of work necessary to complete manual tasks, it will save both time and money. To optimise the process of detection from the lung cancer dataset, a machine learning model based on support vector machines (SVMs),linear Regression,knn, decision tree , random forest was used. Using an these classifiers, lung cancer patients are classified based on their symptoms at the same time as the Python programming language is utilised to further the model implementation. The effectiveness of our SVM model was evaluated in terms of several different criteria. Several cancer datasets from the University of California, Irvine, library were utilised to evaluate the evaluated model. As a result of the favourable findings of this research, smart cities will be able to deliver better healthcare to their citizens. Patients with lung cancer can obtain real-time treatment in a cost-effective manner with the least amount of effort and latency from any location and at any time. The proposed model was compared with the existing SVM and SMOTE methods. The proposed method gets a 98.8% of accuracy rate when comparing the existing methods.

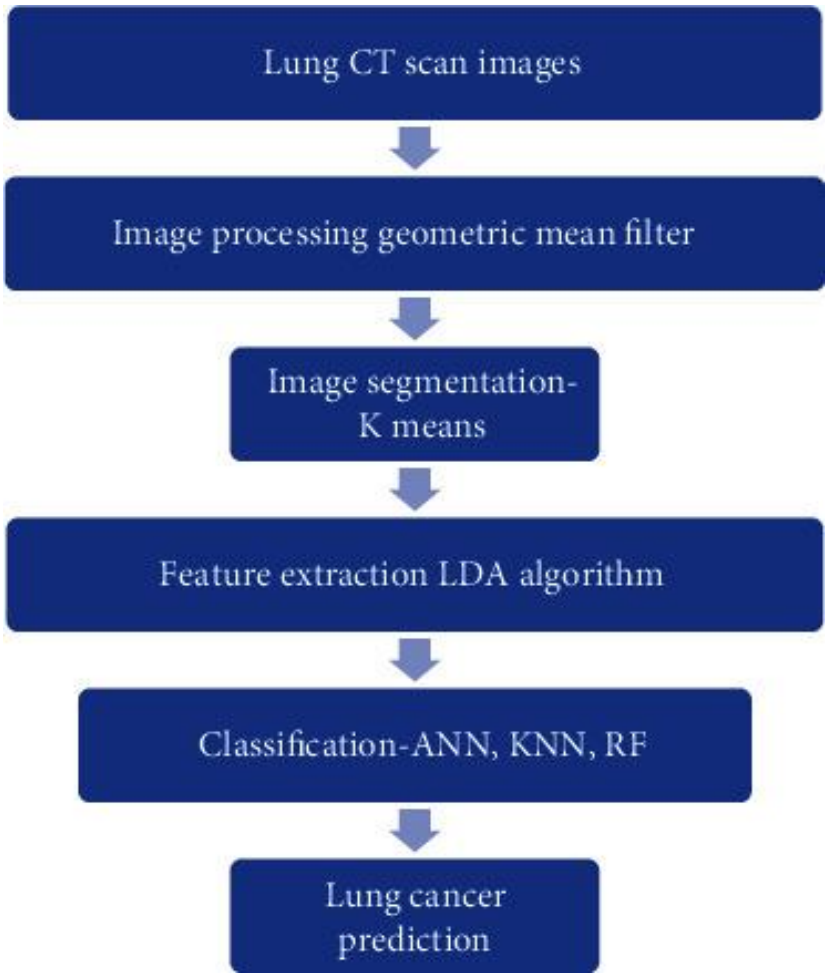
In order to detect areas of the lung that have been affected by cancer, image processing techniques such as noise reduction, feature extraction, identification of damaged regions, and maybe a comparison with data on the medical history of lung cancer are utilized. The majority of the time, digital image processing makes use of a diverse

set of methods to merge a number of distinct aspects of a picture into a single coherent entity. This research takes an innovative technique in order to zero down on a particular aspect of the overall lung image. The split region may be seen in a variety of ways, including from different viewpoints and when illuminated in different ways. When utilizing this method, one of the key benefits is the ability to differentiate between portions of a picture that have been impacted by cancer and sections that have not been affected by cancer.

The methodology section presents accurate classification and prediction of lung cancer using machine learning and image processing-enabled technology. First, images are acquired. Then, images are pre-processed using the geometric mean filter. This results in improving image quality. Then, images are segmented using the *K*-means algorithm. This segmentation helps in the identification of the region of interest. Then, machine learning classification techniques are applied. The result section contains details related to the dataset and results achieved by various techniques.

Methodology

This section shows an accurate classification and prediction of lung cancer using technology that is enabled by machine learning and image processing. To begin, photos need to be gathered. After that, a geometric mean filter is used to perform pre-processing on the images. This ultimately leads to an improvement in image quality. After that, the *K*-means method is used to segment the images. The identification of the region of interest is facilitated by this segmentation. After that, categorization strategies based on machine learning are utilized. The below figure illustrates the classification and prediction of lung cancer utilizing technology that enables machine learning and image processing.



The pre-processing of images plays a significant role in the proper classification of photographs of illnesses. CT scans provide images with a broad variety of artefacts, including noise, which may be seen in these scans. These artefacts may be removed by using image filtering methods. A geometric mean filter is applied to the input pictures in an effort to decrease the amount of noise.

The method of segmentation is used in the process of medical image processing. The basic role of a picture is to differentiate between components that are beneficial and those that are harmful. As a consequence of this, it separates a picture into distinct pieces based on the degree to which each component is similar to its surrounding components. This effect may be achieved by manipulating the intensity as well as the texture. An area of interest that has been segmented may be utilized as a diagnostic tool to quickly get information that is pertinent to the issue at hand.

When it comes to the process of segmenting medical pictures, the technique known as *K*-means clustering is the one that is used most often. During the clustering process, the picture is divided into a number of different groups, also known as clusters, which do not overlap with one another. These clusters are not connected to one another in any way. In this picture, there are a few distinct clusters that can be noticed. Every one of them has its own one-of-a-kind collection of reference points to which each pixel is assigned. To divide the available data into *k* separate groups, the *K*-means clustering algorithm divides the available information based on *k* reference points.

DATASET

Here in these project we have used some factors(conditions) and symptoms on which we are predicting weather a person have lung cancer or not .

Following are the factors which we have used as a dataset in training our models .

```
df1=pd.read_csv("lung_cancer.csv")
df1.columns
```

```
Index(['GENDER', 'AGE', 'SMOKING', 'YELLOW_FINGERS', 'ANXIETY',
       'PEER_PRESSURE', 'CHRONIC DISEASE', 'FATIGUE ', 'ALLERGY ', 'WHEEZING',
       'ALCOHOL CONSUMING', 'COUGHING', 'SHORTNESS OF BREATH',
       'SWALLOWING DIFFICULTY', 'CHEST PAIN', 'LUNG_CANCER'],
      dtype='object')
```

HERE
2 denotes YES
1 denotes NO

	GENDER	AGE	SMOKING	YELLOW_FINGERS	ANXIETY	PEER_PRESSURE	CHRONIC DISEASE
0	M	69	1	2	2	1	1
1	M	74	2	1	1	1	2
2	F	59	1	1	1	2	1
3	M	63	2	2	2	1	1
4	F	63	1	2	1	1	1

FATIGUE	ALLERGY	WHEEZING	ALCOHOL CONSUMING	COUGHING	SHORTNESS OF BREATH	SWALLOWING DIFFICULTY	CHEST PAIN	LUNG_CANCER
2	1	2	2	2	2	2	2	YES
2	2	1	1	1	2	2	2	YES
2	1	2	1	2	2	1	2	NO
1	1	1	2	1	1	2	2	NO
1	1	2	1	2	2	1	1	NO

Result

A dataset of 310 different patients was used in the experimental study. Images are pre-processed using the geometric mean filter. This results in improving image quality. Then, images are segmented using different classification algorithm. This segmentation helps in the identification of the region of interest. Then, machine learning classification techniques are applied.

For performance comparison, three parameters, accuracy, sensitivity, and specificity, are used:

$$\text{Accuracy}=TP+TNTP+TN+FP+FN,$$

$$\text{Sensitivity}=TPTP+FN,$$

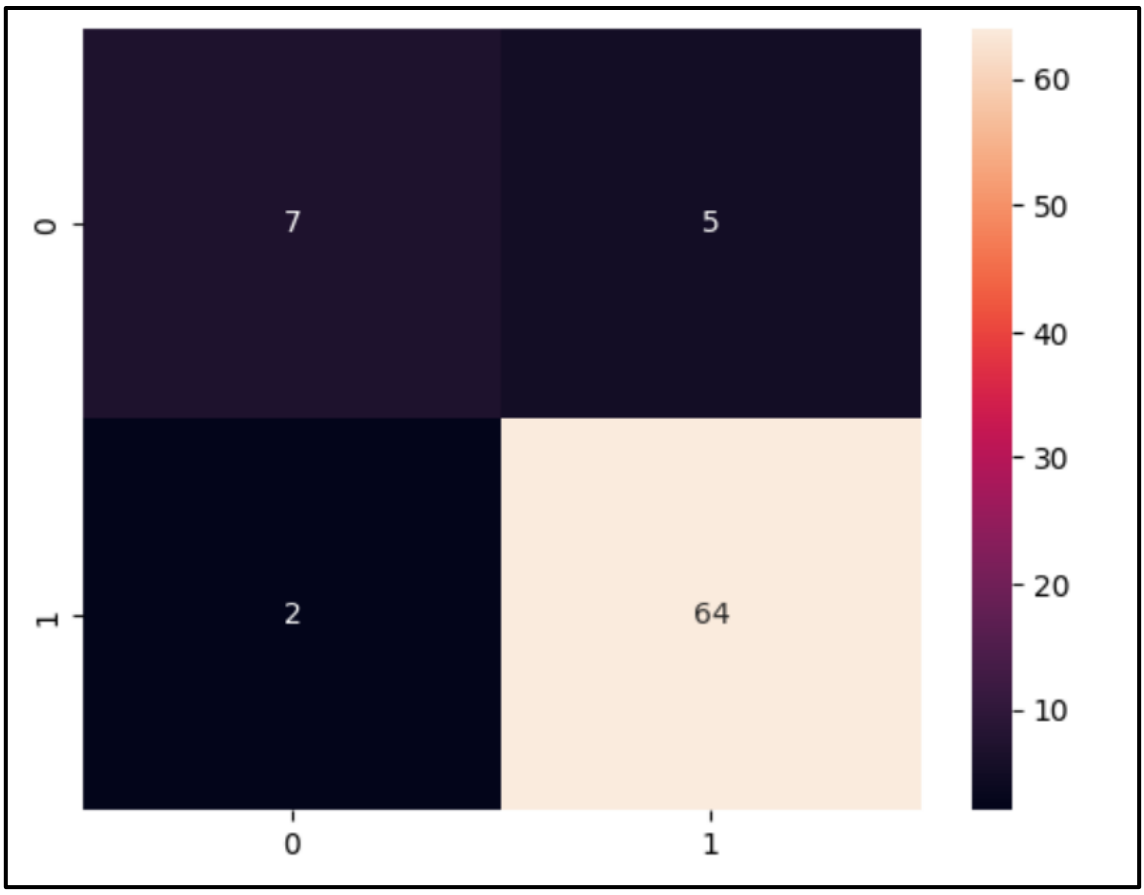
$$\text{Specificity}=TNTN+FP,$$

where TP is true positive, TN is true negative, FP is false positive, and FN is false negative.

Confusion matrix and classification report of various machine learning algorithms.

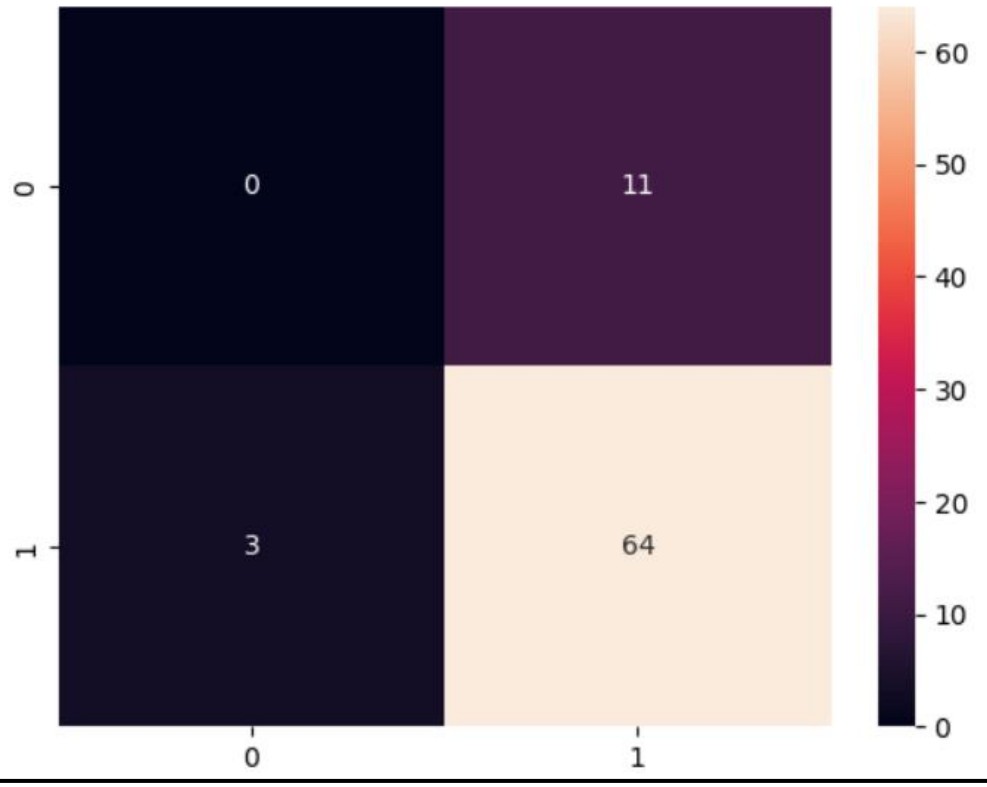
1. Linear regression

```
first 5 actual_y: [1 1 1 0 1] predicted y: [0.78383047 0.87888029 0.83485786 0.761
62371 0.79791579]
Model_score - 0.053811716223445605
```



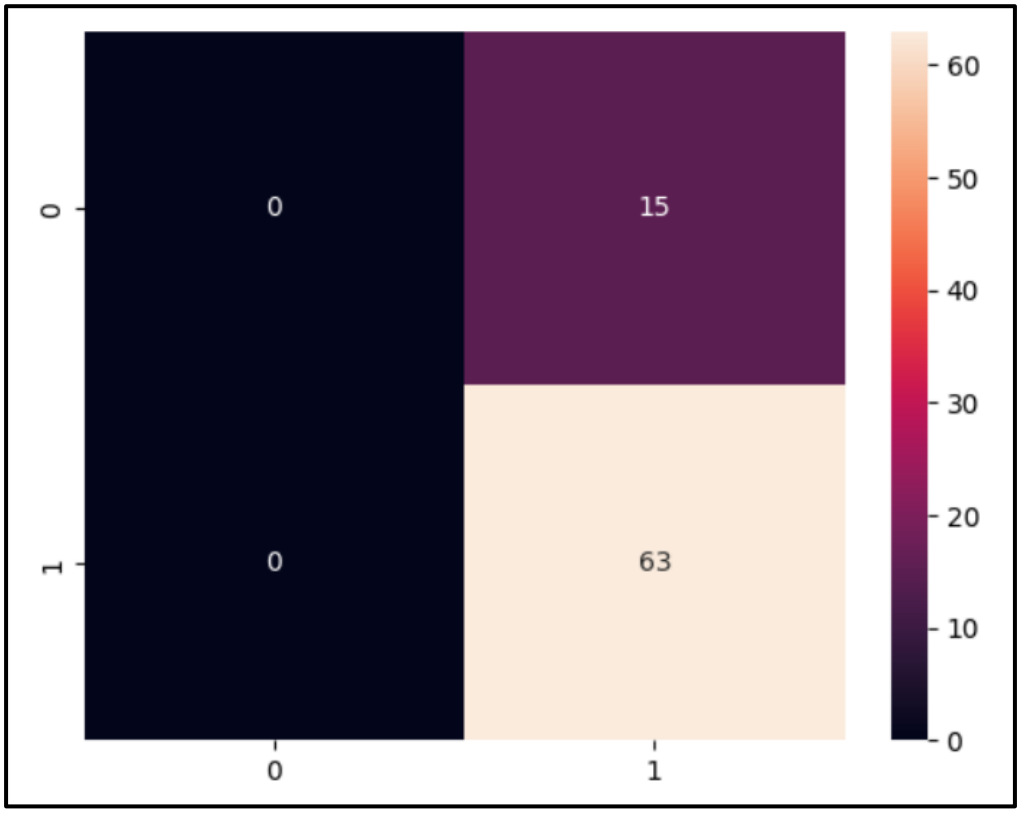
2. Logistic Regression

```
first 5 actual_y: [1 1 0 1 1] predicted y: [1 1 1 1 1]
confusion_matrix
[[ 0 11]
 [ 3 64]]
classification_report
              precision    recall  f1-score   support

           0       0.00      0.00      0.00        11
           1       0.85      0.96      0.90        67

    accuracy                           0.82        78
   macro avg       0.43      0.48      0.45        78
weighted avg       0.73      0.82      0.77        78

accuracy_score- 0.8205128205128205
Model_score - 0.8205128205128205
```



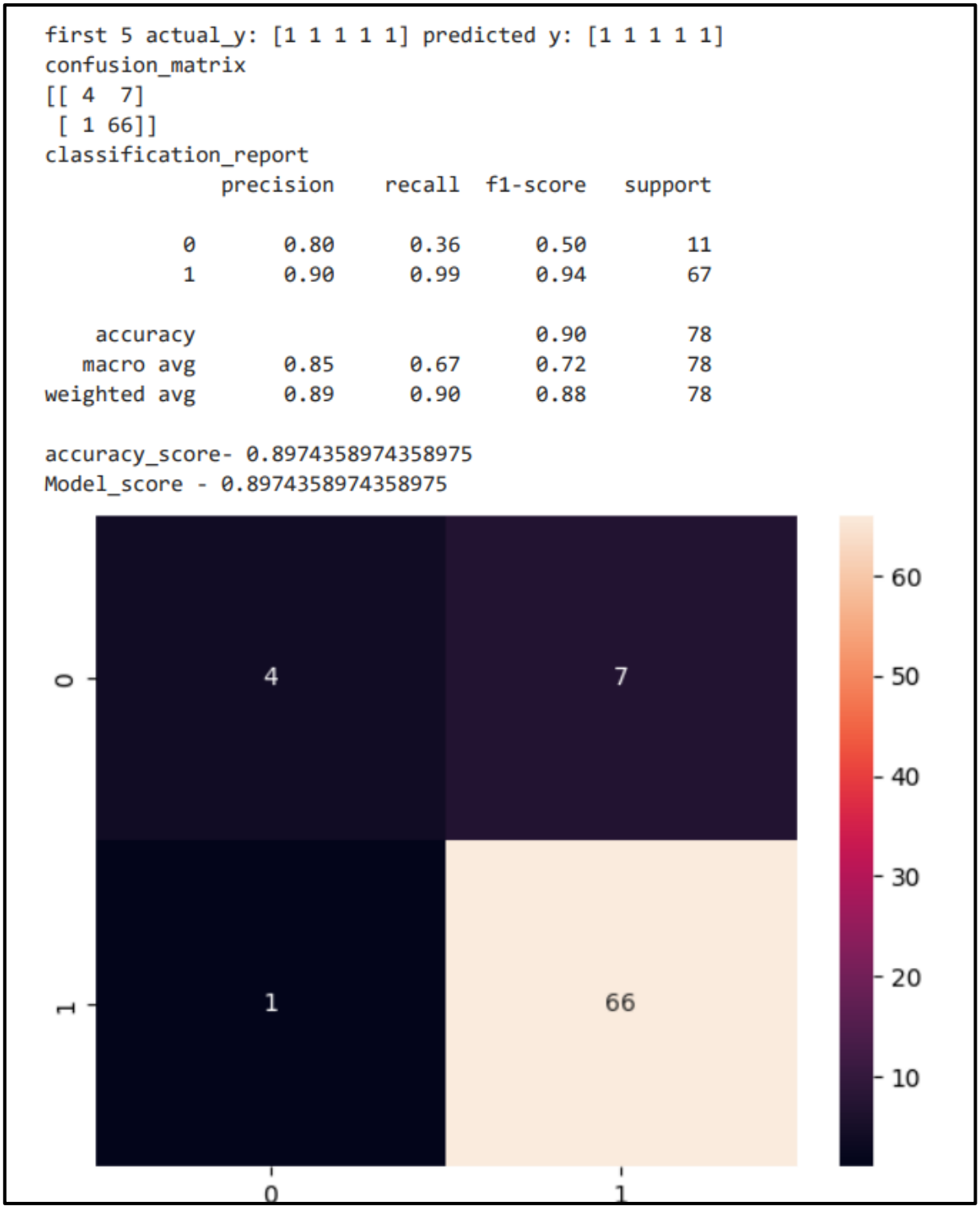
3. K Nearest Classifier

```
first 5 actual_y: [1 1 1 0 0] predicted y: [1 1 1 1 1]
confusion_matrix
[[ 0 15]
 [ 0 63]]
classification_report
              precision    recall  f1-score   support

           0       0.00      0.00      0.00        15
           1       0.81      1.00      0.89        63

    accuracy                           0.81        78
   macro avg       0.40      0.50      0.45        78
weighted avg       0.65      0.81      0.72        78

accuracy_score- 0.8076923076923077
Model_score - 0.8076923076923077
```



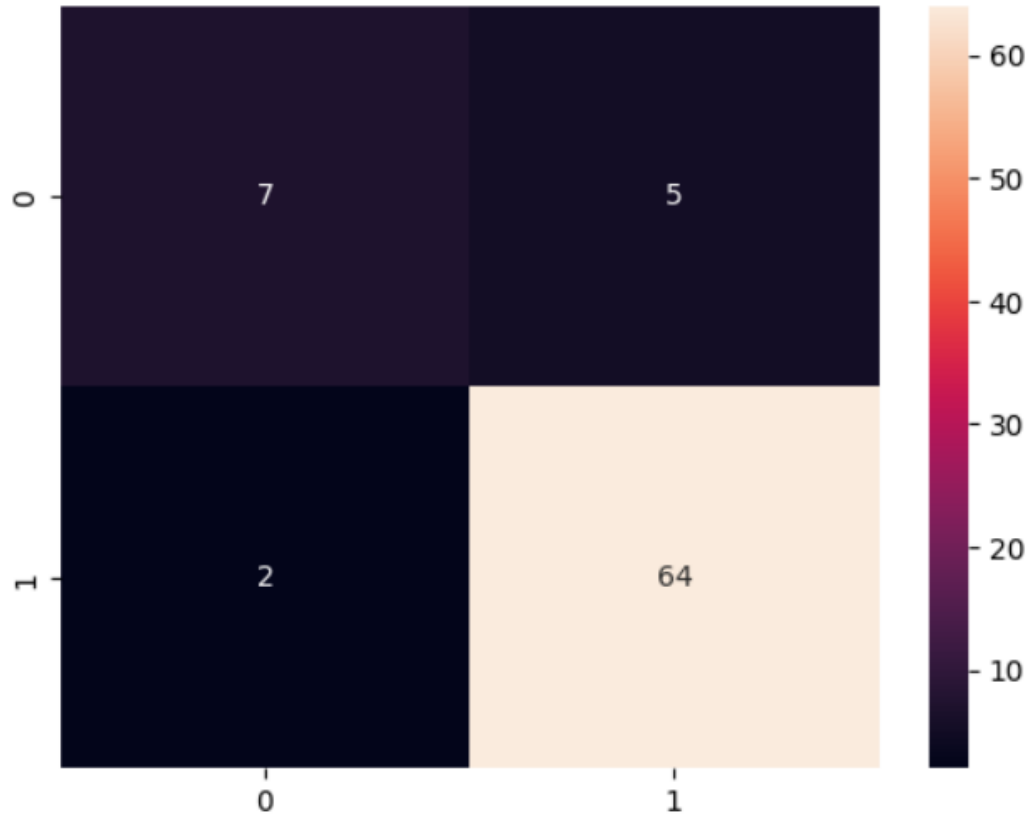
4. Decision Tree Classifier

```
first 5 actual_y: [1 1 1 1 1] predicted y: [1 1 1 1 1]
confusion_matrix
[[ 4  7]
 [ 1 66]]
classification_report
              precision    recall  f1-score   support

           0       0.80      0.36      0.50        11
           1       0.90      0.99      0.94        67

    accuracy                           0.90        78
   macro avg       0.85      0.67      0.72        78
weighted avg       0.89      0.90      0.88        78

accuracy_score- 0.8974358974358975
Model_score - 0.8974358974358975
```



5. Random Forest Classifier

```
first 5 actual_y: [0 0 1 1 0] predicted y: [1 0 1 1 0]
confusion_matrix
[[ 7  5]
 [ 2 64]]
classification_report
              precision    recall  f1-score   support

           0       0.78      0.58      0.67        12
           1       0.93      0.97      0.95        66

    accuracy                           0.91        78
   macro avg       0.85      0.78      0.81        78
weighted avg       0.90      0.91      0.90        78

accuracy_score- 0.9102564102564102
Model_score - 0.9102564102564102
```



Conclusion

Lung cancer is one of the deadliest types of the disease, claiming the lives of approximately one million people each year. Given the current state of affairs in medicine, it is critical that lung nodule identification be performed on chest CT scans. As a result, the use of CAD systems is crucial for the early detection of lung cancer. Image processing is a necessary activity that is employed in a wide range of economic domains. It is used in X-ray imaging of the lungs to find areas of the body that have developed malignant growths. Image processing techniques such as noise reduction, feature extraction, identification of damaged regions, and maybe comparison with data on the medical history of lung cancer are used to locate sections of the lung that have been affected by cancer.

This study demonstrates accurate lung cancer classification and prediction using technologies enabled by machine learning and image processing. To begin, photographs must be collected. Following that, the images are pre-processed using a geometric mean filter. This eventually leads to an increase in image quality. The *K*-means approach is then used to segment the images. This segmentation makes it easier to identify the region of interest. Following that, machine learning-based categorization algorithms are used.

Random Forest predicts lung cancer with more accuracy. This research will help to increase the accuracy of lung cancer detection systems that use strong classification and prediction techniques. This study brings cutting-edge images based on machine learning techniques for implementation purposes.

accuracy of all the model for the given problem is given below

1. linear Regression= 0.053811716223445605
2. Logistic regresssion=0.8205128205128205
3. k nearest neighbour=0.8076923076923077
4. decision tree classifier=0.8974358974358975
5. random forest classifier=0.9102564102564102

From all the above model the accuracy of the **Random Forest** is maximum hence we will use random forest algorithm to detect the lung cancer.

```
first 5 actual_y: [0 0 1 1 0] predicted y: [1 0 1 1 0]
confusion_matrix
[[ 7  5]
 [ 2 64]]
classification_report
              precision    recall  f1-score   support

           0       0.78      0.58      0.67        12
           1       0.93      0.97      0.95        66

    accuracy                           0.91        78
   macro avg       0.85      0.78      0.81        78
weighted avg       0.90      0.91      0.90        78

accuracy_score- 0.9102564102564102
Model_score - 0.9102564102564102
```

Lung cancer prediction using random forest.

LUNG CANCER PREDICTION

SUBMITTED BY:

GAUTAM JAISWAL
SOPHIYA SALEEMA QURESHI
KAMAKSHI SHARMA
OJAS KHATAVKAR

INTRODUCTION

Lung cancer is the one of the leading cause of cancer deaths in both women and men. Manifestation of Lung cancer in the body of the patient reveals through early symptoms in most of the cases.Treatment and prognosis depend on the histological type of cancer, the stage (degree of spread), and the patient's performance status. Possible treatments include surgery, chemotherapy , and radiotherapy Survival depends on stage, overall health , and other factors, but overall only 14% of people diagnosed with lung cancer survive five years after the diagnosis.

OBJECTIVE

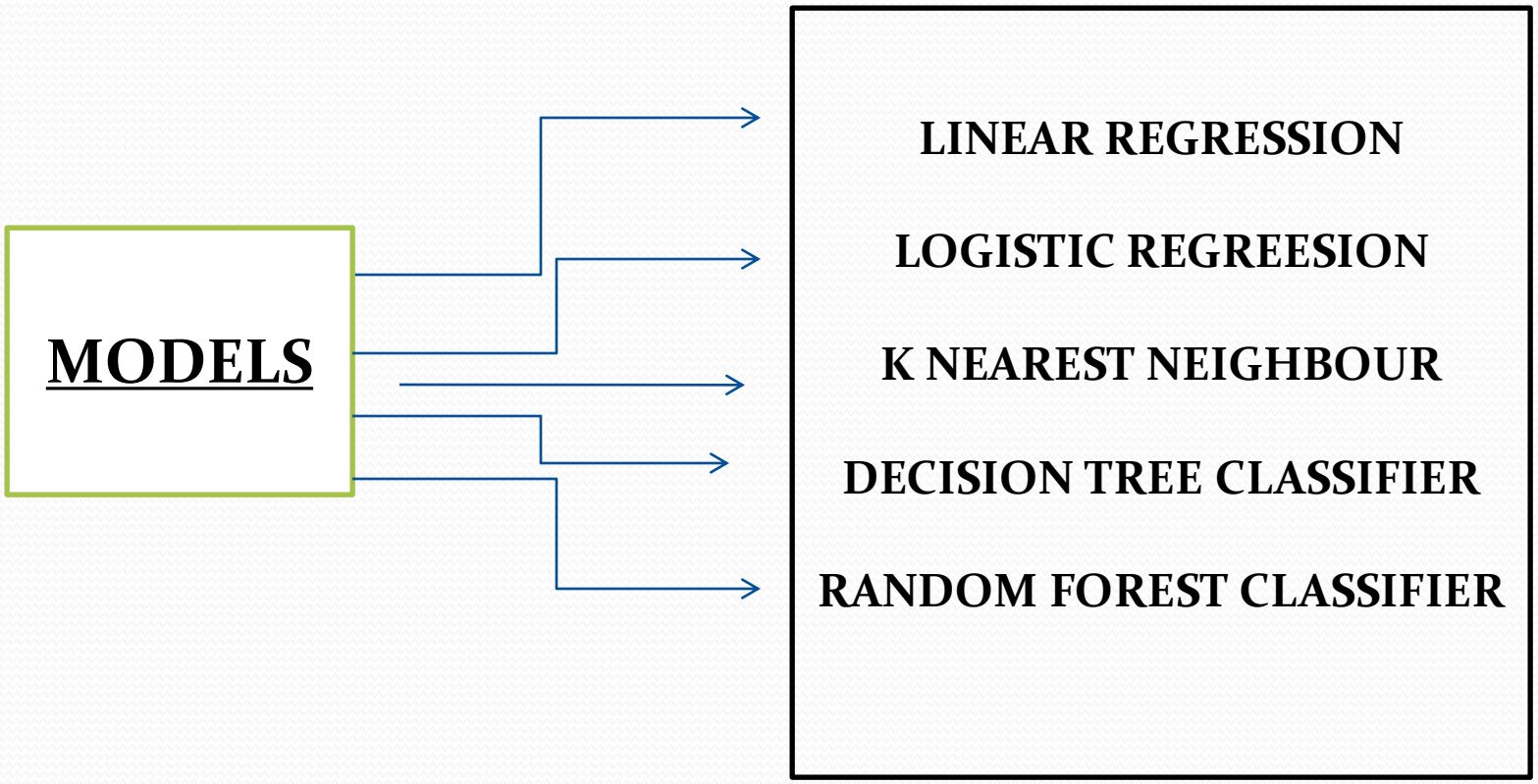
- Implement various machine learning algorithm to Detect Lung Cancer of person based on various symptoms and habits (using the gathered dataset from various source) and find out which algorithm is best(highest accuracy)for the given problem.

Symptoms of Lung Cancer

The following are the generic lung cancer symptoms

- i. A cough that does not go away and gets worse over time
- ii. Coughing up blood (heamoptysis) or bloody mucus.
- iii. Chest, shoulder, or back pain that doesn't go away and often is made worse by deep Hoarseness
- iv. Weight loss and loss of appetite
- v. Increase in volume of sputum.

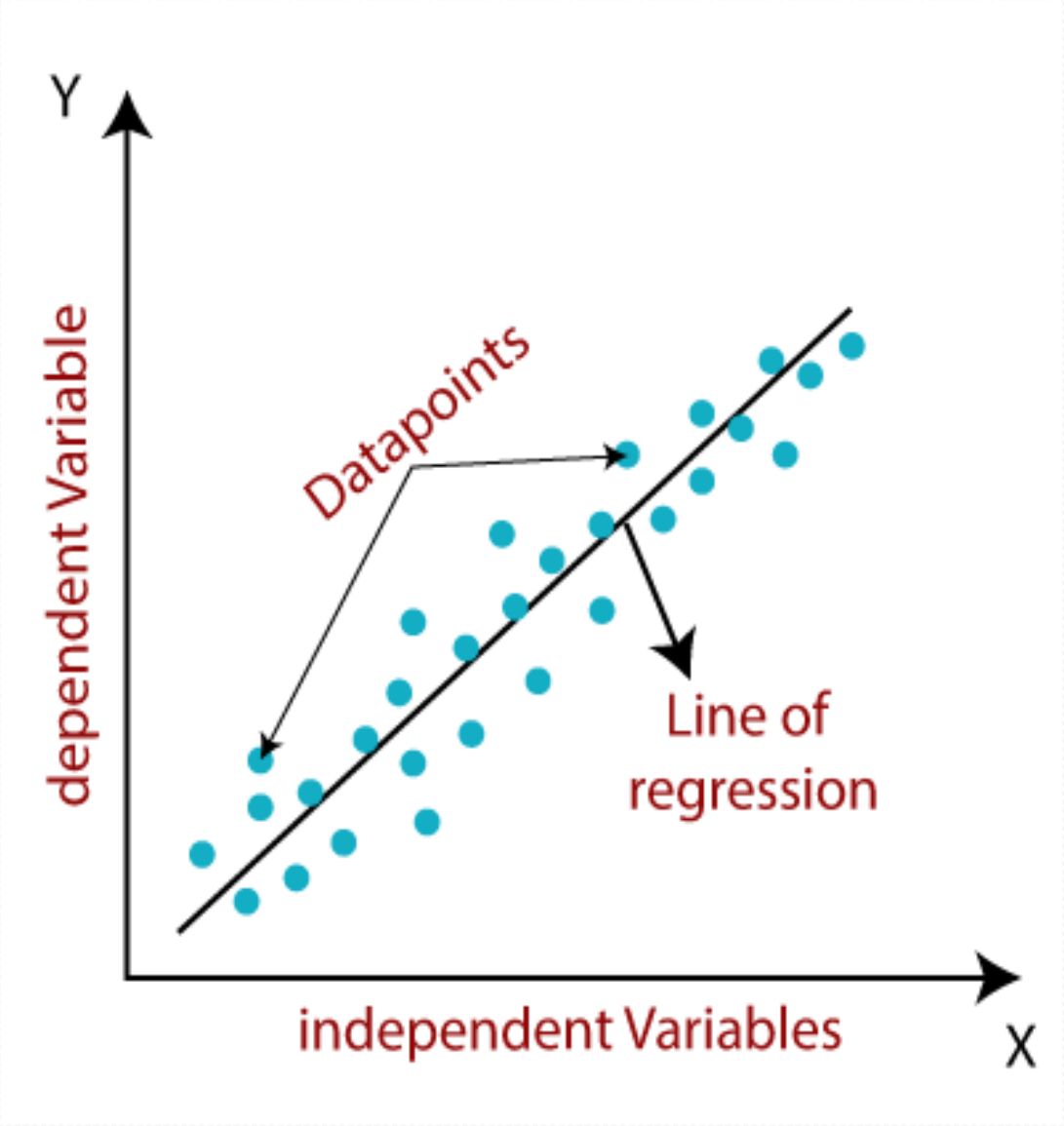
METHODOLODY



LINAER REGRESSION

_Linear regression is one of the easiest and most popular Machine Learning algorithms. It is a statistical method that is used for predictive analysis. Linear regression makes predictions for continuous/real or numeric variables such as sales, salary, age, product price, etc.

Linear regression algorithm shows a linear relationship between a dependent (y) and one or more independent (y) variables, hence called as linear regression. Since linear regression shows the linear relationship, which means it finds how the value of the dependent variable is changing according to the value of the independent variable.

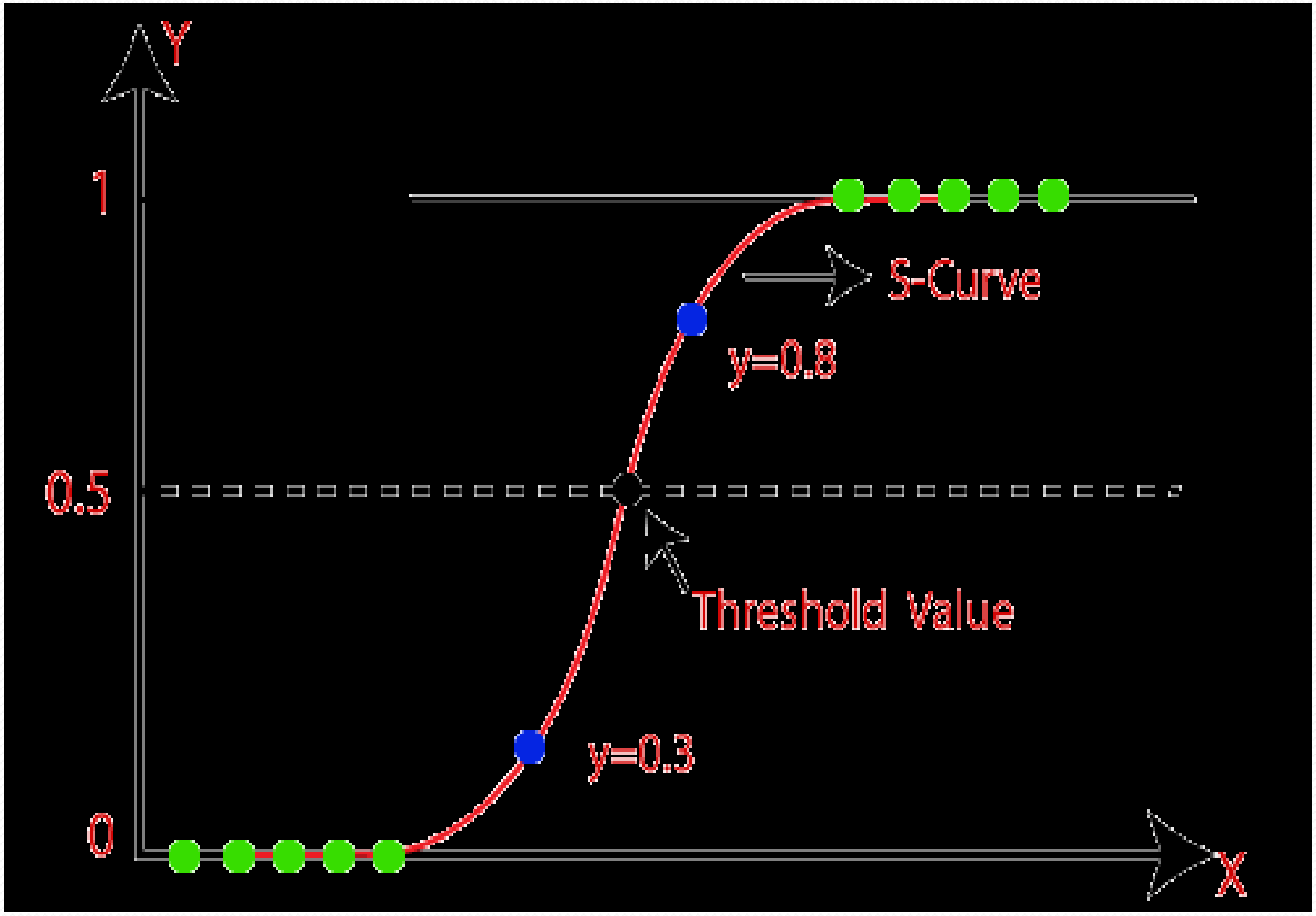

Y
dependent Variable
Datapoints
Line of
regression
independent Variables
X

LOGISTIC REGRESSION

It is a supervised learning classification technique that forecasts the likelihood of a target variable. There will only be a choice between two classes. Data can be coded as either one or yes, representing success, or as 0 or no, representing failure.

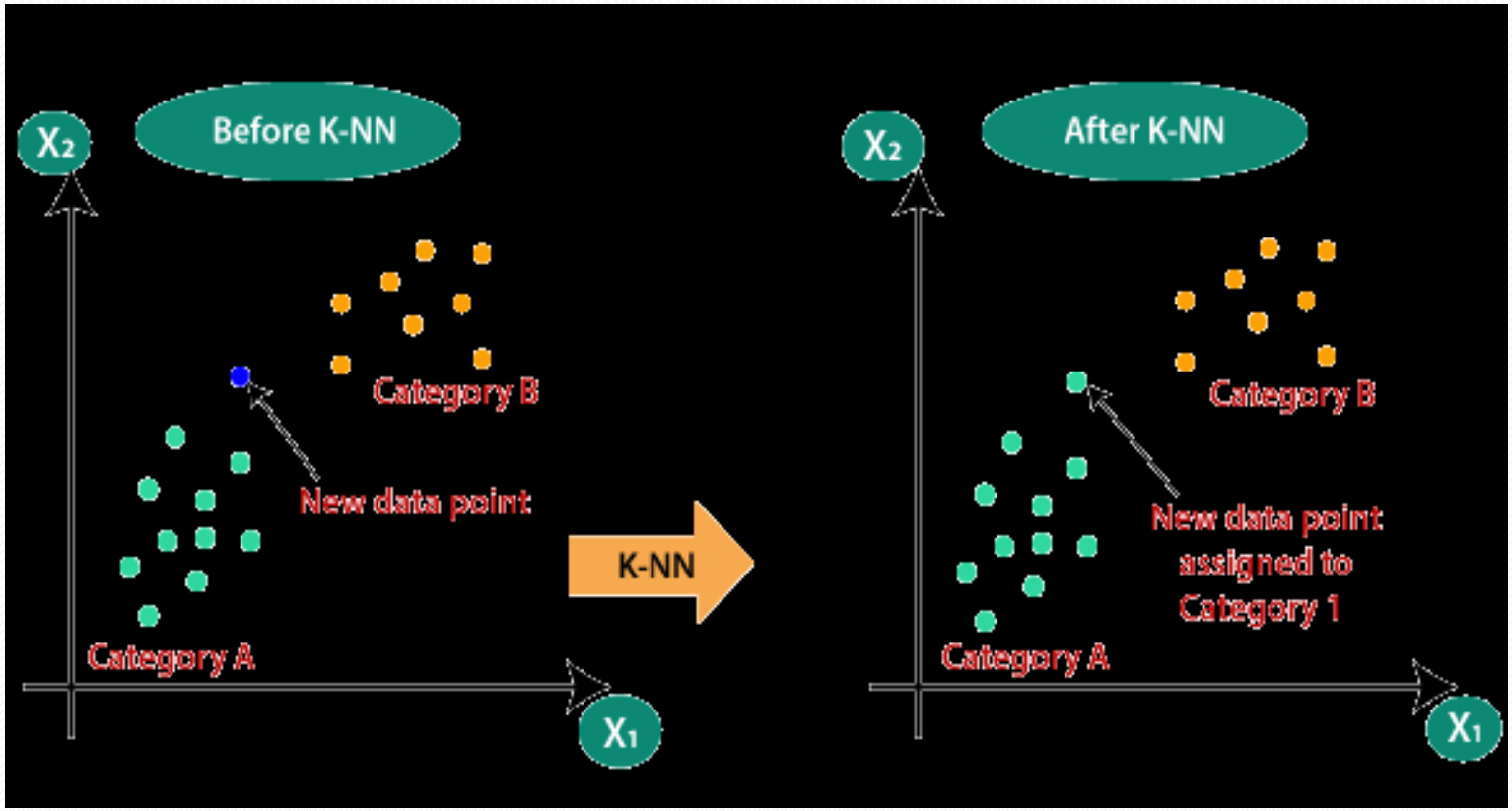
The dependent variable can be predicted most effectively using logistic regression. When the forecast is categorical, such as true or false, yes or no, or a 0 or 1, you can use it. A logistic regression technique can be used to determine whether or not an email is a spam.

Logistic regression, instead of fitting a regression line, we fit an "S" shaped logistic function, which predicts two maximum values (0 or 1).


Y
1
0.5
0
S-Curve
y=0.8
Threshold Value
y=0.3
X

K NEAREST NEIGHBOUR(KNN)

- K-NN algorithm assumes the similarity between the new case/data and available cases and put the new case into the category that is most similar to the available categories.This algorithm stores all the available data and classifies a new data point based on the similarity. This means when new data appears then it can be easily classified into a well suite category by using K- NN algorithm.
- It is also called a **lazy learner algorithm** because it does not learn from the training set immediately instead it stores the dataset and at the time of classification, it performs an action on the dataset.
- KNN algorithm at the training phase just stores the dataset and when it gets new data, then it classifies that data into a category that is much similar to the new data.


Before K-NN
X2
Category B
New data point
Category A
X1
K-NN
After K-NN
X2
Category B
New data point
assigned to
Category 1
Category A
X1

DECISION TREE CLASSIFIER

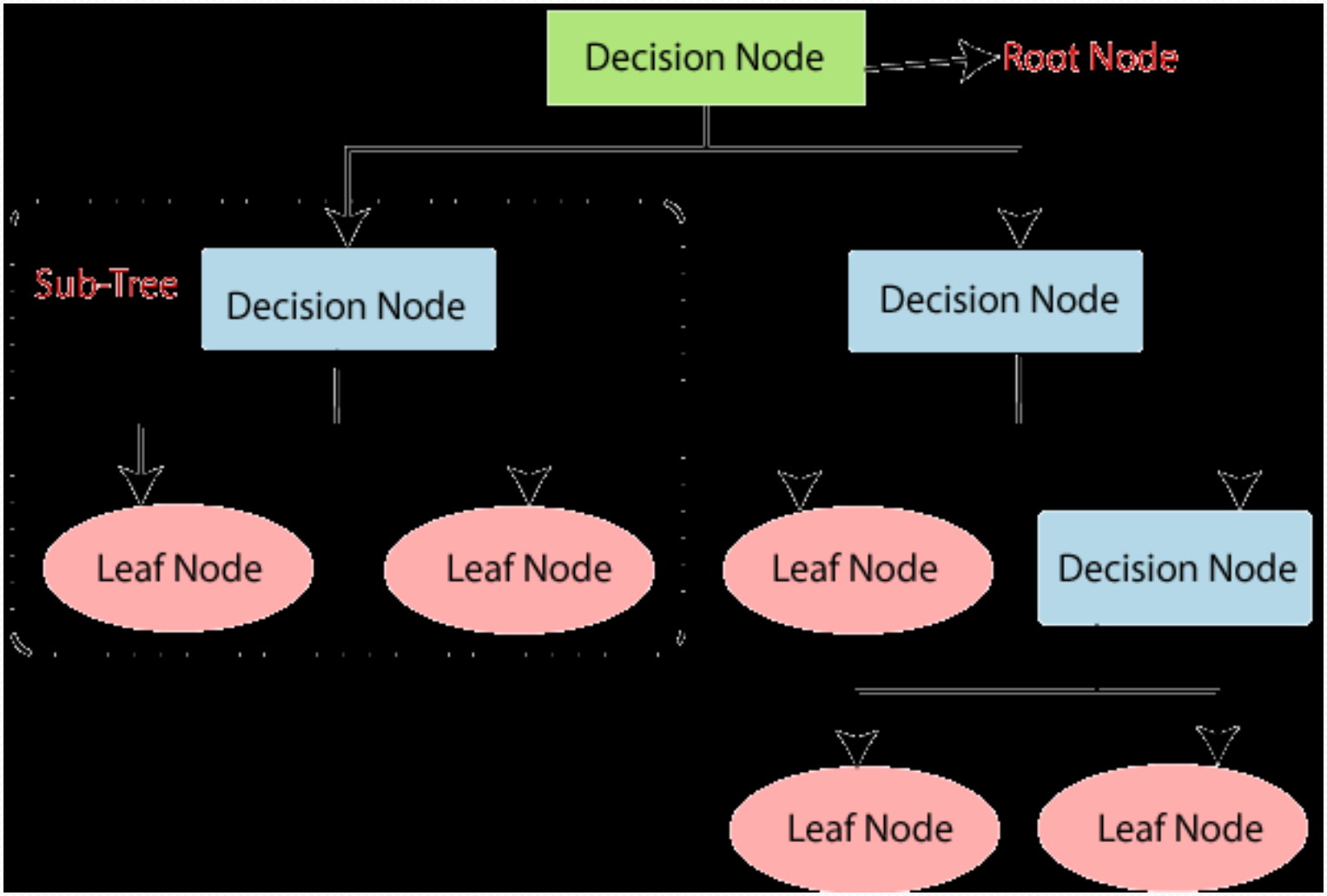
Decision Tree is a Supervised learning technique that can be used for both classification and Regression problems, but mostly it is preferred for solving Classification problems.

It is a graphical representation for getting all the possible solutions to a problem/decision based on given conditions.

It is a tree-structured classifier, where internal nodes represent the features of a dataset, branches represent the decision rules and each leaf node represents the outcome.

In a Decision tree, there are two nodes, which are the Decision Node and Leaf Node. Decision nodes are used to make any decision and have multiple branches, whereas Leaf nodes are the output of those decisions and do not contain any further branches.

The decisions or the test are performed on the basis of features of the given dataset.

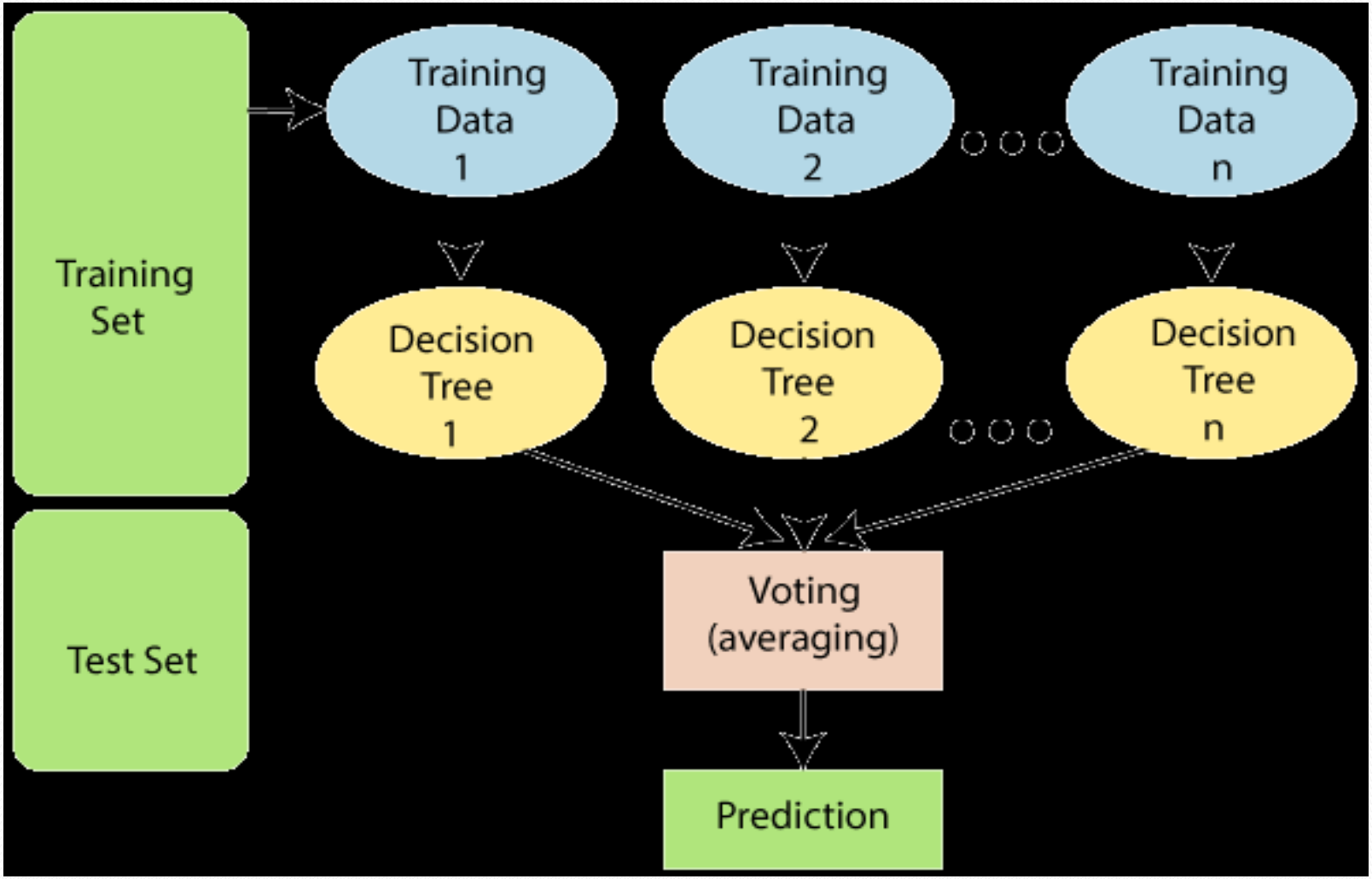

Decision Node
Root Node
Sub-Tree
Decision Node
Decision Node
Leaf Node
Leaf Node
Leaf Node
Decision Node
Leaf Node
Leaf Node

RANDOM FOREST CLASSIFIER

Random Forest is a popular machine learning algorithm that belongs to the supervised learning technique. It can be used for both Classification and Regression problems in ML. It is based on the concept of ensemble learning, which is a process of *combining multiple classifiers to solve a complex problem and to improve the performance of the model.*

Random Forest is a classifier that contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset.

Instead of relying on one decision tree, the random forest takes the prediction from each tree and based on the majority votes of predictions, and it predicts the final output.


Training Set
Test Set
Training Data 1
Training Data 2
Training Data n
Decision Tree 1
Decision Tree 2
Decision Tree n
Voting (averaging)
Prediction

STEPS

- 1. import library
- 2. read dataset
- 3. data analysis
- 4. Label encoding of the categorical columns
- 5. defining x and y
- 6. splitting x and y into train and test data
- 7. import the model
- 8. train the model with x_train and y_train
- 9. predict with x_test and got predicted y
- 10. evaluation with confusion matrix, classification report , accuracy score.

IMPLEMENTATION

Lung Cancer Detection

About Dataset The effectiveness of the cancer prediction system helps people to know their cancer risk wi a low cost and it also helps the people to take the appropriate decision based on their cancer risk status. The data is collected from the website online lung cancer prediction system.

In [17]:
```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
df1=pd.read_csv("lung_cancer.csv")
df1.columns
```

Out[17]:
```
Index(['GENDER', 'AGE', 'SMOKING', 'YELLOW_FINGERS', 'ANXIETY',
       'PEER_PRESSURE', 'CHRONIC DISEASE', 'FATIGUE ', 'ALLERGY ', 'WHEEZING',
       'ALCOHOL CONSUMING', 'COUGHING', 'SHORTNESS OF BREATH',
       'SWALLOWING DIFFICULTY', 'CHEST PAIN', 'LUNG_CANCER'],
      dtype='object')
```

In [27]:
```
df.head(2)
```

Out[27]:

	GENDER	AGE	SMOKING	YELLOW_FINGERS	ANXIETY	PEER_PRESSURE	CHRONIC DISEASE	FATIGUE
0	M	69	1	2	2	1	1	2
1	M	74	2	1	1	1	2	2

linear Regression

In [26]:
```
from sklearn.preprocessing import LabelEncoder
lb=LabelEncoder()
df1['LUNG_CANCER']=lb.fit_transform (df1[ 'LUNG_CANCER'])
df1['GENDER']=lb.fit_transform (df1[ 'GENDER'])
x=df1[[ 'GENDER', 'AGE', 'SMOKING', 'YELLOW_FINGERS', 'ANXIETY','PEER_PRESSURE', 'C
y=df1[ 'LUNG_CANCER']
from sklearn.model_selection import train_test_split
x_train,x_test,y_train, y_test=train_test_split(x,y,test_size=.25)
from sklearn.linear_model import LinearRegression
lr=LinearRegression()
lr.fit(x_train, y_train)
y_pred=lr.predict(x_test)
print (f'first 5 actual_y: {y_test.values[:5]} predicted y: {y_pred[:5]}')

print('Model_score -',lr.score (x_test,y_test))
sns.heatmap(a, annot=True)
plt.show()
```

```
first 5 actual_y: [1 1 1 0 1] predicted y: [0.78383047 0.87888029 0.83485786 0.761
62371 0.79791579]
Model_score - 0.053811716223445605
```

logistic regression

In [18]:

```
from sklearn.preprocessing import LabelEncoder
lb=LabelEncoder()
df1['LUNG_CANCER']=lb.fit_transform (df1[ 'LUNG_CANCER'])
df1['GENDER']=lb.fit_transform (df1[ 'GENDER'])
x=df1[[ 'GENDER', 'AGE', 'SMOKING', 'YELLOW_FINGERS', 'ANXIETY','PEER_PRESSURE', 'C
y=df1[ 'LUNG_CANCER']
from sklearn.model_selection import train_test_split
x_train,x_test,y_train, y_test=train_test_split(x,y,test_size=.25)
from sklearn.linear_model import LogisticRegression
lr=LogisticRegression()
lr.fit(x_train, y_train)
y_pred=lr.predict(x_test)
print (f'first 5 actual_y: {y_test.values[:5]} predicted y: {y_pred[:5]}')
from sklearn.metrics import confusion_matrix, classification_report, accuracy_score
print('confusion_matrix')
a=confusion_matrix (y_test,y_pred)
print(a)
print('classification_report')
print(classification_report (y_test,y_pred))
print('accuracy_score-', accuracy_score (y_test,y_pred))
print('Model_score -',lr.score (x_test,y_test))
sns.heatmap(a, annot=True)
plt.show()
```

k nearest classifier

```
from sklearn.preprocessing import LabelEncoder
lb=LabelEncoder()
df1['LUNG_CANCER']=lb.fit_transform (df1[ 'LUNG_CANCER'])
df1['GENDER']=lb.fit_transform (df1[ 'GENDER'])
x=df1[[ 'GENDER', 'AGE', 'SMOKING', 'YELLOW_FINGERS', 'ANXIETY','PEER_PRESSURE', '(
```

nvert/html/ds project lung.ipynb?download=false

ds project lung

```
y=df1[ 'LUNG_CANCER']
from sklearn.model_selection import train_test_split
x_train,x_test,y_train, y_test=train_test_split(x,y,test_size=.25)
#from sklearn.linear_model import LogisticRegression
#lr=LogisticRegression()
from sklearn.neighbors import KNeighborsClassifier
lr=KNeighborsClassifier()
lr.fit(x_train, y_train)
y_pred=lr.predict(x_test)
print (f'first 5 actual_y: {y_test.values[:5]} predicted y: {y_pred[:5]}')
from sklearn.metrics import confusion_matrix, classification_report, accuracy_score
print('confusion_matrix')
a=confusion_matrix (y_test,y_pred)
print(a)
print('classification_report')
print(classification_report (y_test,y_pred))
print('accuracy_score-', accuracy_score (y_test,y_pred))
print('Model_score -',lr.score (x_test,y_test))
sns.heatmap(a, annot=True)
plt.show()
```

decision tree classifier

In [20]:

```
from sklearn.preprocessing import LabelEncoder
lb=LabelEncoder()
df1['LUNG_CANCER']=lb.fit_transform (df1[ 'LUNG_CANCER'])
df1['GENDER']=lb.fit_transform (df1[ 'GENDER'])
x=df1[[ 'GENDER', 'AGE', 'SMOKING', 'YELLOW_FINGERS', 'ANXIETY','PEER_PRESSURE', 'C
y=df1[ 'LUNG_CANCER']
from sklearn.model_selection import train_test_split
x_train,x_test,y_train, y_test=train_test_split(x,y,test_size=.25)
from sklearn.tree import DecisionTreeClassifier
lr=DecisionTreeClassifier()

lr.fit(x_train, y_train)
y_pred=lr.predict(x_test)
print (f'first 5 actual_y: {y_test.values[:5]} predicted y: {y_pred[:5]}')
from sklearn.metrics import confusion_matrix, classification_report, accuracy_score
print('confusion_matrix')
a=confusion_matrix (y_test,y_pred)
print(a)
print('classification_report')
print(classification_report (y_test,y_pred))
print('accuracy_score-', accuracy_score (y_test,y_pred))
print('Model_score -',lr.score (x_test,y_test))
sns.heatmap(a, annot=True)
plt.show()
```

Random forest classifier

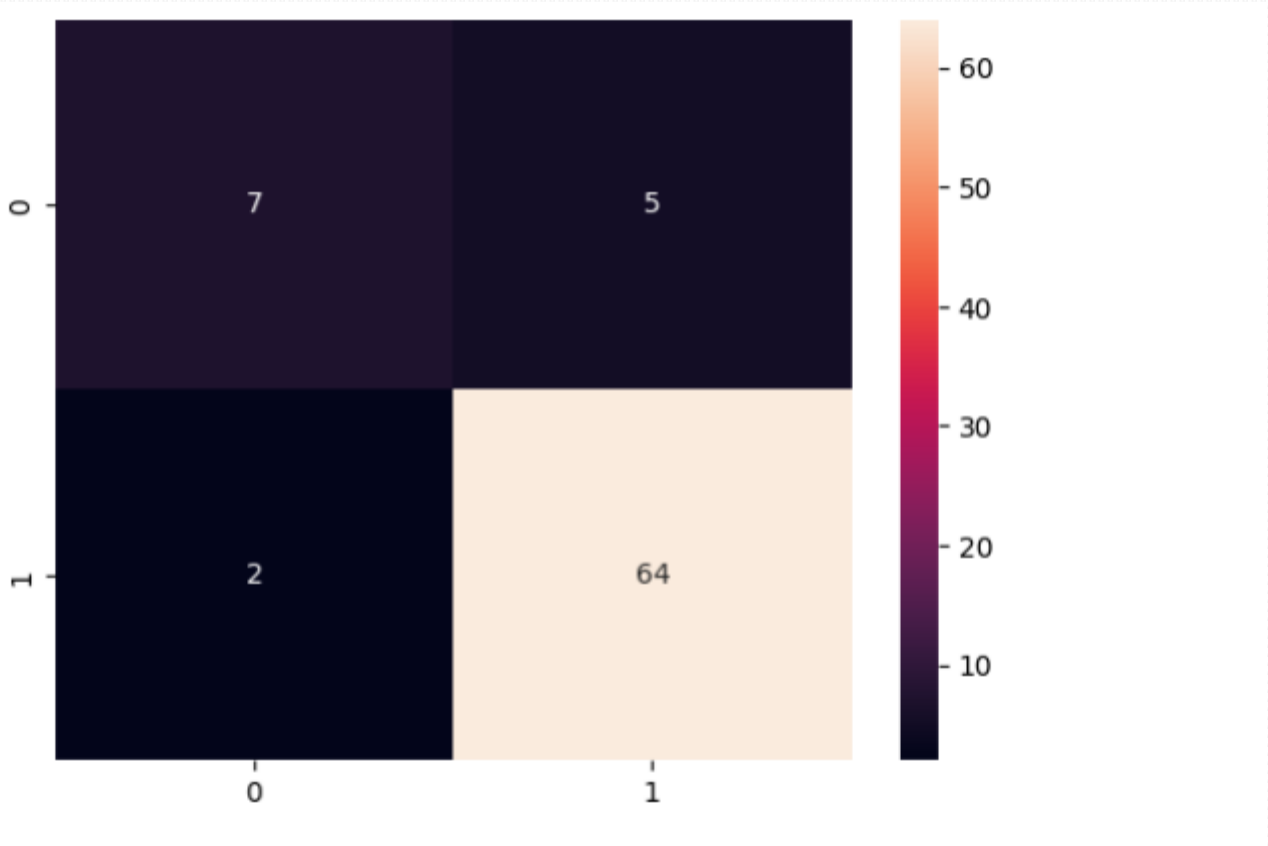
In [22]:
```
from sklearn.preprocessing import LabelEncoder
lb=LabelEncoder()
df1['LUNG_CANCER']=lb.fit_transform (df1[ 'LUNG_CANCER'])
df1['GENDER']=lb.fit_transform (df1[ 'GENDER'])
x=df1[[ 'GENDER', 'AGE', 'SMOKING', 'YELLOW_FINGERS', 'ANXIETY','PEER_PRESSURE', 'C
y=df1[ 'LUNG_CANCER']
from sklearn.model_selection import train_test_split
x_train,x_test,y_train, y_test=train_test_split(x,y,test_size=.25)
from sklearn.ensemble import RandomForestClassifier
lr=RandomForestClassifier()

lr.fit(x_train, y_train)
y_pred=lr.predict(x_test)
print (f'first 5 actual_y: {y_test.values[:5]} predicted y: {y_pred[:5]}')
from sklearn.metrics import confusion_matrix, classification_report, accuracy_score
print('confusion_matrix')
a=confusion_matrix (y_test,y_pred)
print(a)
print('classification_report')
print(classification_report (y_test,y_pred))
```

```
print('accuracy_score-', accuracy_score (y_test,y_pred))
print('Model_score -',lr.score (x_test,y_test))
sns.heatmap(a, annot=True)
plt.show()
```

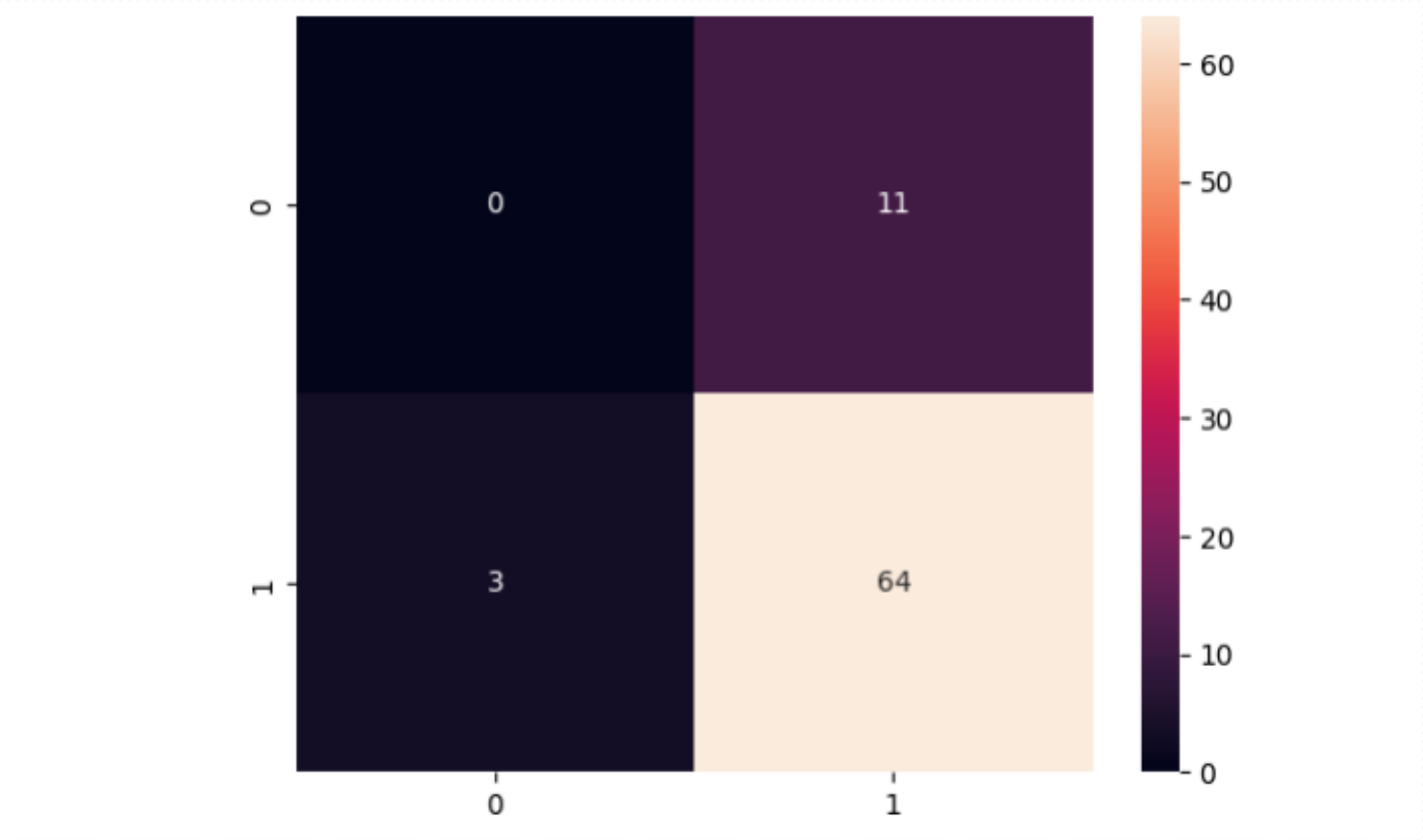
```
first 5 actual_y: [0 0 1 1 0] predicted y: [1 0 1 1 0]
confusion_matrix
[[ 7  5]
 [ 2 64]]
classification_report
              precision    recall  f1-score   support

           0       0.78      0.58      0.67        12
           1       0.93      0.97      0.95        66

    accuracy                           0.91        78
   macro avg       0.85      0.78      0.81        78
weighted avg       0.90      0.91      0.90        78

accuracy_score- 0.9102564102564102
Model_score - 0.9102564102564102
```

RESULT

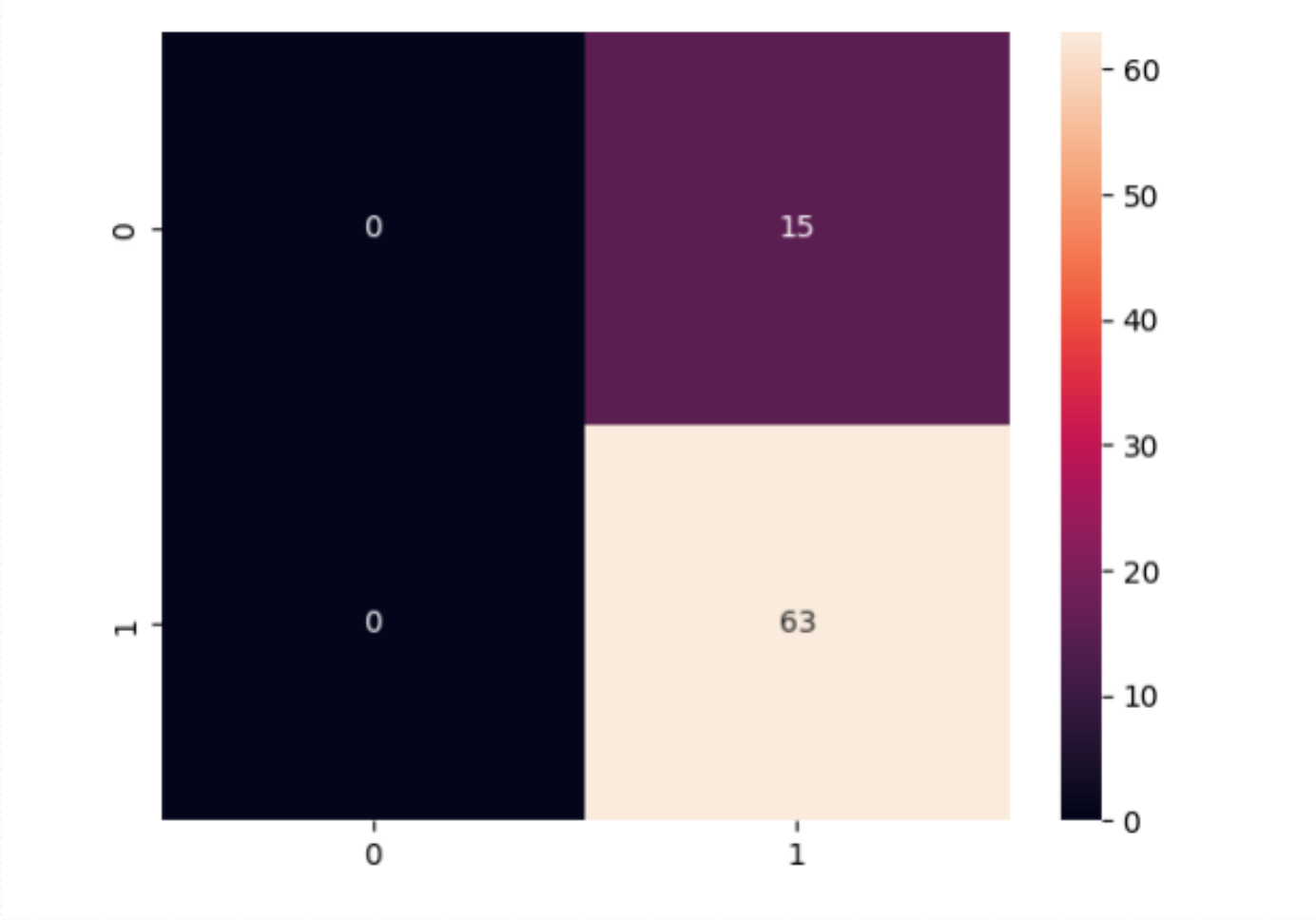
LINEAR REGRESSION:



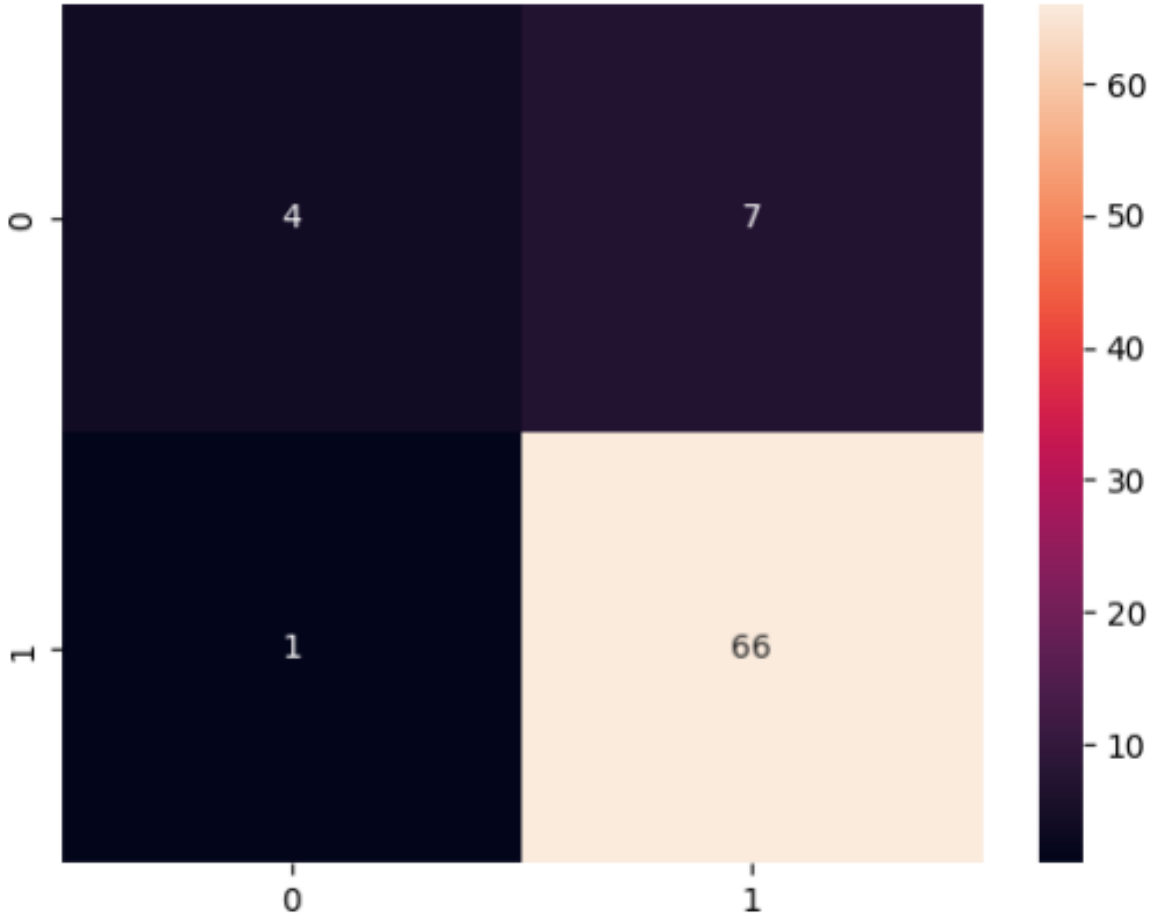
LOGISTIC REGRESSION



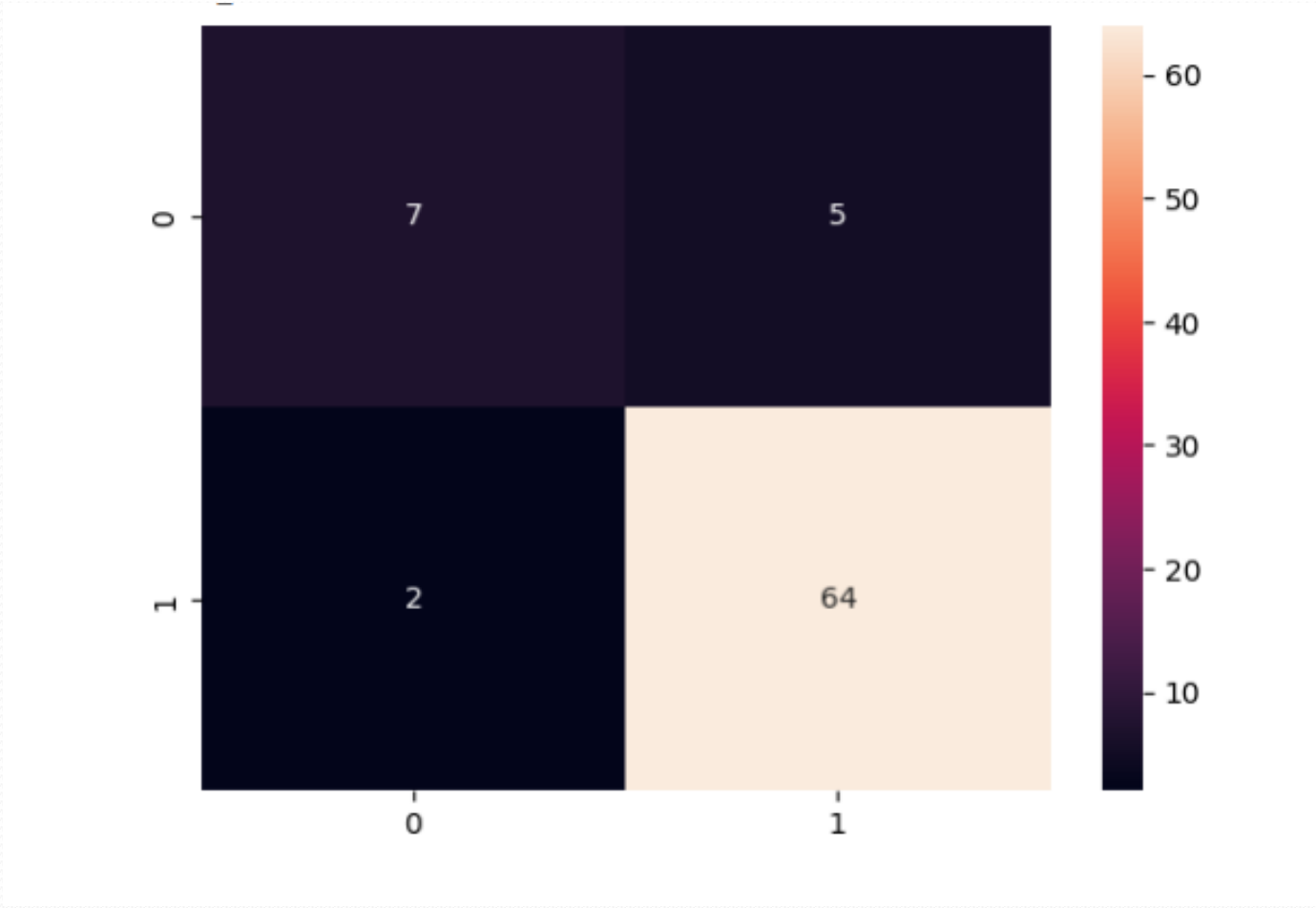
K NEAREST NEIGHBOUR(KNN)



DECISION TREE CLASSIFIER

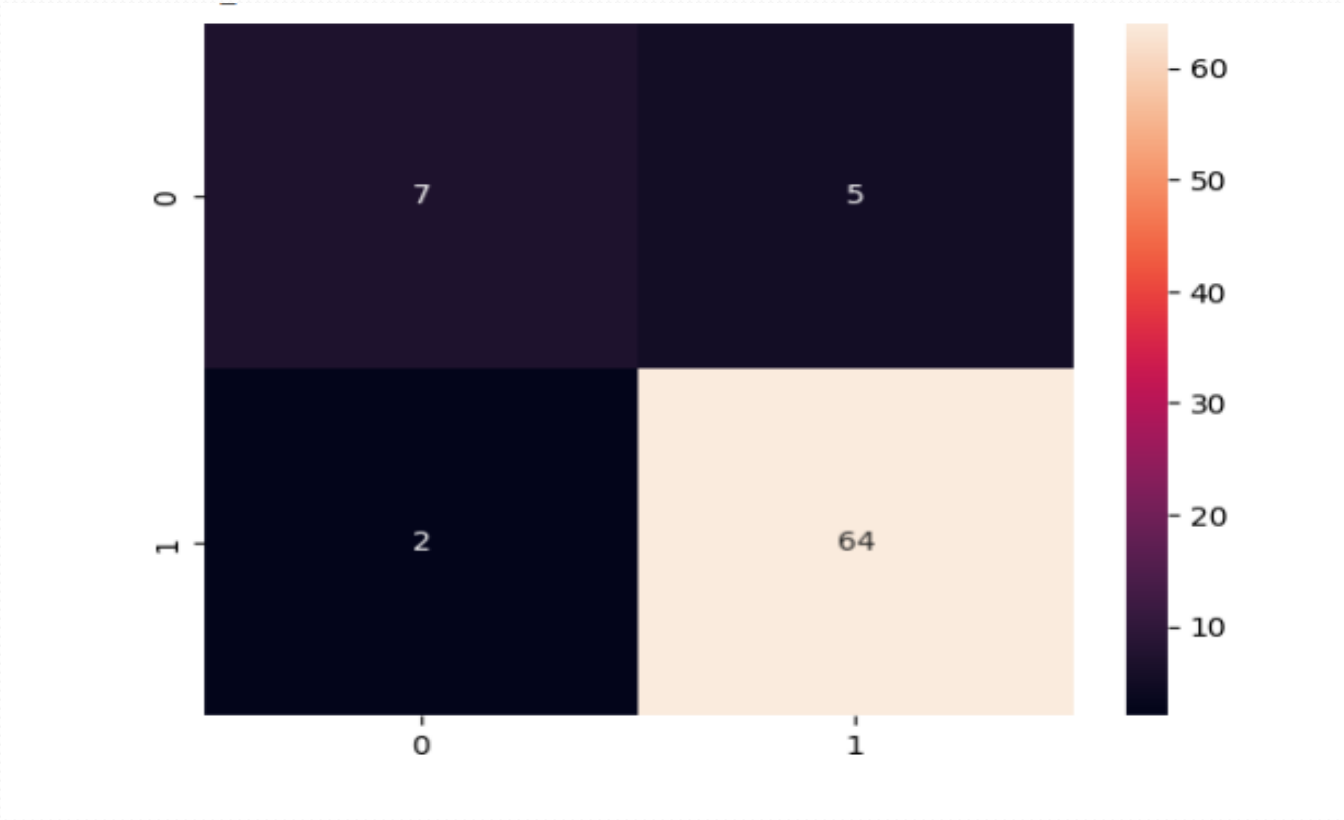


RANDOM FOREST CLASSIFIER



CONCLUSION:

Here,Random Forest Classifier predict lung cancer with more accuracy. This result will help us to increase the accuracy of lung cancer prediction systems that use strong classification and prediction techniques.



Lung Cancer Detection

INTRODUCTION TO DATA SCIENCE(BTIBM505)

PROJECT:LUNG CANCER DETECTION

MADE BY:

GAUTAM JAISWAL(20100BTCSAII07160)

OJAS KHATAVKAR (20100BTCSAII07178)

SOPHIYA SALEEMA QURESHI (20100BTCSAII07190)

KAMAKSHI SHARMA (20100BTCSAII07170)

SEC-D (AI) 3RD YEAR 5TH SEM

SESSION 2022-23

About Dataset The effectiveness of the cancer prediction system helps people to know their cancer risk wi a low cost and it also helps the people to take the appropriate decision based on their cancer risk status. The data is collected from the website online lung cancer prediction system.

In [35]:
```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
```

In [36]:
```
df1=pd.read_csv("lung_cancer.csv")
df1.columns
```

Out[36]:
```
Index(['GENDER', 'AGE', 'SMOKING', 'YELLOW_FINGERS', 'ANXIETY',
       'PEER_PRESSURE', 'CHRONIC DISEASE', 'FATIGUE ', 'ALLERGY ', 'WHEEZING',
       'ALCOHOL CONSUMING', 'COUGHING', 'SHORTNESS OF BREATH',
       'SWALLOWING DIFFICULTY', 'CHEST PAIN', 'LUNG_CANCER'],
      dtype='object')
```

In [37]:
```
print("HERE BEFORE LABEL ENCODING")
print("2 denotes 'YES, person have cancer'\n'1 denotes NO,person does not have canc
df1.head()
```

```
HERE BEFORE LABEL ENCODING
2 denotes 'YES, person have cancer'
'1 denotes NO,person does not have cancer
```

Out[37]:

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

In [38]:
```
df1.describe()
```

Out[38]:

	AGE	SMOKING	YELLOW_FINGERS	ANXIETY	PEER_PRESSURE	CHRONIC DISEASE	FATI
count	309.000000	309.000000	309.000000	309.000000	309.000000	309.000000	309.00
mean	62.673139	1.563107	1.569579	1.498382	1.501618	1.504854	1.67
std	8.210301	0.496806	0.495938	0.500808	0.500808	0.500787	0.46
min	21.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.00
25%	57.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.00
50%	62.000000	2.000000	2.000000	1.000000	2.000000	2.000000	2.00
75%	69.000000	2.000000	2.000000	2.000000	2.000000	2.000000	2.00
max	87.000000	2.000000	2.000000	2.000000	2.000000	2.000000	2.00

In [39]:
```
df1["LUNG_CANCER"].unique()
```

Out[39]:
```
array(['YES', 'NO'], dtype=object)
```

In [40]:
```
from sklearn.preprocessing import LabelEncoder
lb=LabelEncoder()
df1['LUNG_CANCER']=lb.fit_transform (df1[ 'LUNG_CANCER'])
df1['GENDER']=lb.fit_transform (df1[ 'GENDER'])
x=df1[[ 'GENDER', 'AGE', 'SMOKING', 'YELLOW_FINGERS', 'ANXIETY','PEER_PRESSURE', 'C
       'ALCOHOL CONSUMING', 'COUGHING', 'SHORTNESS OF BREATH',
       'SWALLOWING DIFFICULTY', 'CHEST PAIN']]
y=df1[ 'LUNG_CANCER']
from sklearn.model_selection import train_test_split
x_train,x_test,y_train, y_test=train_test_split(x,y,test_size=.25)
```

In [41]:
```
print("HERE AFTER LABEL ENCODING")
print("1 denotes 'YES, person have cancer'\n'0 denotes NO,person does not have canc
df1.head()
```

```
HERE AFTER LABEL ENCODING
1 denotes 'YES, person have cancer'
'0 denotes NO,person does not have cancer
```

Out[41]:

	GENDER	AGE	SMOKING	YELLOW_FINGERS	ANXIETY	PEER_PRESSURE	CHRONIC DISEASE	FATIGUE
0	1	69	1	2	2	1	1	2
1	1	74	2	1	1	1	2	2
2	0	59	1	1	1	2	1	2
3	1	63	2	2	2	1	1	1
4	0	63	1	2	1	1	1	1

linear Regression

In [42]:
```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train, y_test=train_test_split(x,y,test_size=.25)
from sklearn.linear_model import LinearRegression
lr=LinearRegression()
lr.fit(x_train, y_train)
y_pred=lr.predict(x_test)
print("1 denotes 'YES, person have cancer'\n'0 denotes NO,person does not have canc
print (f'first 5 actual_y: {y_test.values[:5]} predicted y: {y_pred[:5]}')

print('Model_score :',lr.score (x_test,y_test))
linearReg_accuracy=lr.score(x_test,y_test)
#sns.heatmap(a, annot=True)
#plt.show()
```

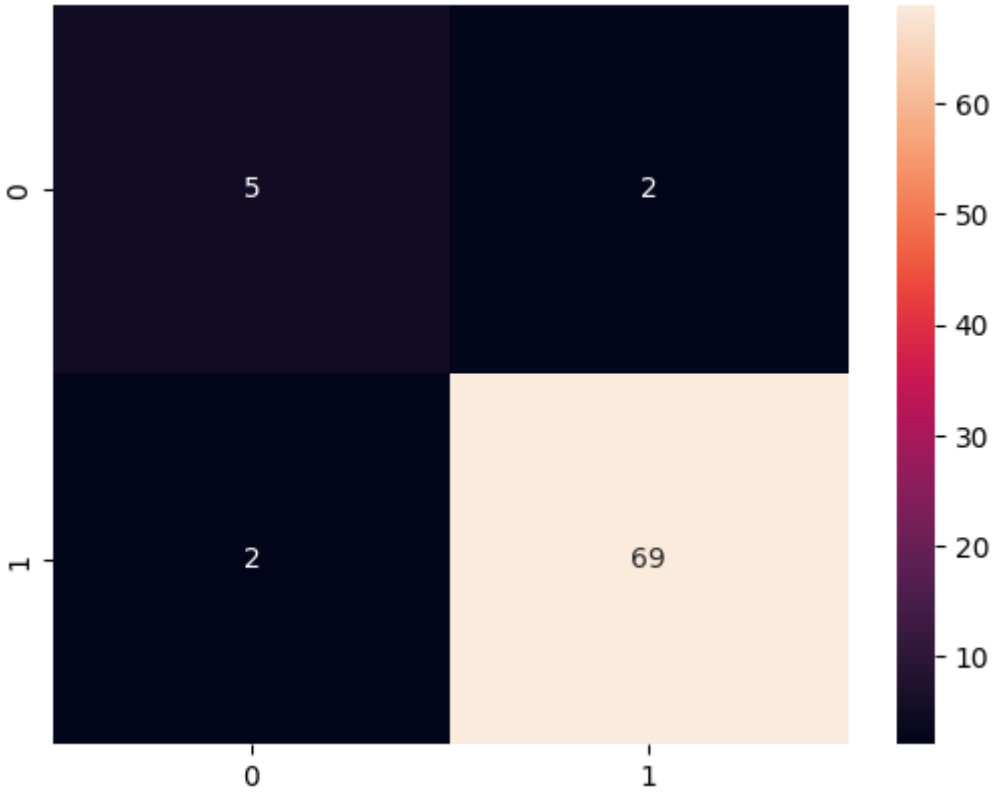
```
1 denotes 'YES, person have cancer'
'0 denotes NO,person does not have cancer
first 5 actual_y: [1 1 1 1 1] predicted y: [0.90192815 1.19227659 1.26731771 0.985
90128 0.99406915]
Model_score : 0.2079084725452871
```

logistic regression

In [43]:
```
from sklearn.linear_model import LogisticRegression
lr=LogisticRegression()
lr.fit(x_train, y_train)
y_pred=lr.predict(x_test)
print("1 denotes 'YES, person have cancer'\n'0 denotes NO,person does not have canc
print (f'first 5 actual_y: {y_test.values[:5]} predicted y: {y_pred[:5]}')
from sklearn.metrics import confusion_matrix, classification_report, accuracy_score
print('confusion_matrix')
a=confusion_matrix (y_test,y_pred)
print(a)
print('classification_report')
print(classification_report (y_test,y_pred))
print('accuracy_score-', accuracy_score (y_test,y_pred))
print('Model_score -',lr.score (x_test,y_test))
logisticReg_accuracy=accuracy_score(y_test,y_pred)
sns.heatmap(a, annot=True)
plt.show()
```

```
1 denotes 'YES, person have cancer'
'0 denotes NO,person does not have cancer
first 5 actual_y: [1 1 1 1 1] predicted y: [1 1 1 1 1]
confusion_matrix
[[ 5  2]
 [ 2 69]]
classification_report
              precision    recall  f1-score   support

           0       0.71      0.71      0.71         7
           1       0.97      0.97      0.97        71

    accuracy                           0.95        78
   macro avg       0.84      0.84      0.84        78
weighted avg       0.95      0.95      0.95        78

accuracy_score- 0.9487179487179487
Model_score - 0.9487179487179487
```

```
C:\Users\Hp\AppData\Local\Programs\Python\Python310\lib\site-packages\sklearn\linear_model\_logistic.py:444: ConvergenceWarning: 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(
```

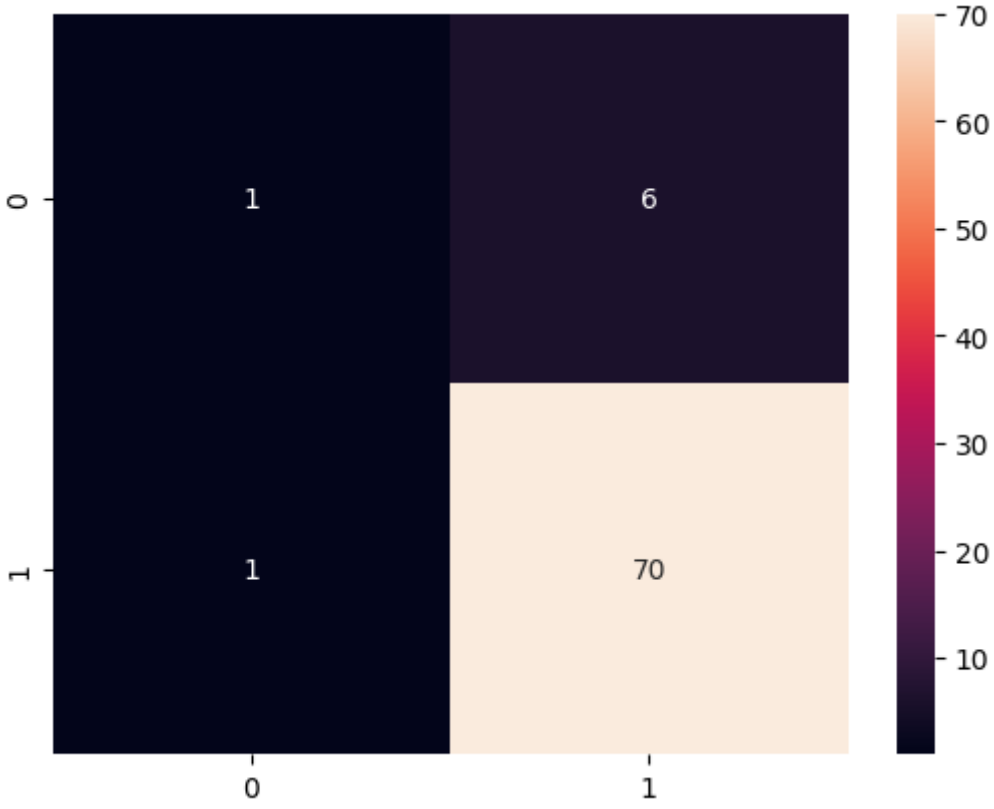


k nearest classifier

In [44]:
```
from sklearn.neighbors import KNeighborsClassifier
lr=KNeighborsClassifier()
lr.fit(x_train, y_train)
```

```
y_pred=lr.predict(x_test)
print("1 denotes 'YES, person have cancer'\n'0 denotes NO,person does not have canc
print (f'first 5 actual_y: {y_test.values[:5]} predicted y: {y_pred[:5]}')
from sklearn.metrics import confusion_matrix, classification_report, accuracy_score
print('confusion_matrix')
a=confusion_matrix (y_test,y_pred)
print(a)
print('classification_report')
print(classification_report (y_test,y_pred))
print('accuracy_score-', accuracy_score (y_test,y_pred))
print('Model_score -',lr.score (x_test,y_test))
sns.heatmap(a, annot=True)
plt.show()
```

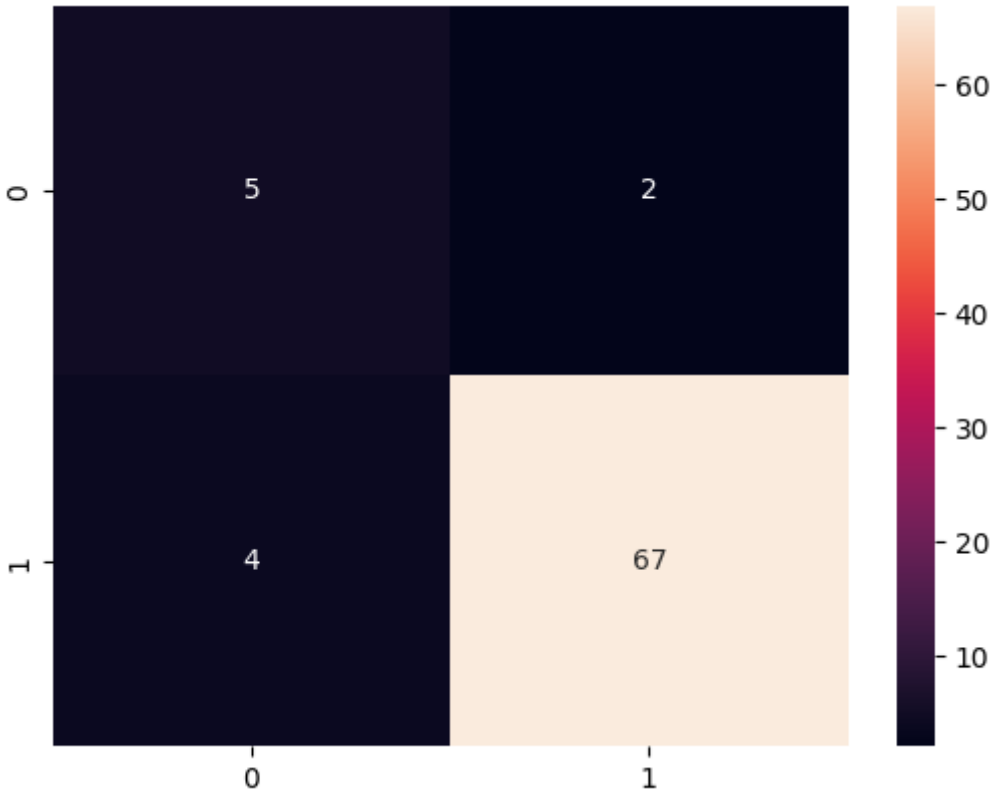
```
1 denotes 'YES, person have cancer'
'0 denotes NO,person does not have cancer
first 5 actual_y: [1 1 1 1 1] predicted y: [1 1 1 1 1]
confusion_matrix
[[ 1  6]
 [ 1 70]]
classification_report
              precision    recall  f1-score   support

           0       0.50      0.14      0.22         7
           1       0.92      0.99      0.95        71

    accuracy                           0.91        78
   macro avg       0.71      0.56      0.59        78
weighted avg       0.88      0.91      0.89        78

accuracy_score- 0.9102564102564102
Model_score - 0.9102564102564102
```



decision tree classifier

In [45]:
```
from sklearn.tree import DecisionTreeClassifier
lr=DecisionTreeClassifier()

lr.fit(x_train, y_train)
y_pred=lr.predict(x_test)
print("1 denotes 'YES, person have cancer'\n'0 denotes NO,person does not have canc
print (f'first 5 actual_y: {y_test.values[:5]} predicted y: {y_pred[:5]}')
from sklearn.metrics import confusion_matrix, classification_report, accuracy_score
print('confusion_matrix')
a=confusion_matrix (y_test,y_pred)
print(a)
print('classification_report')
print(classification_report (y_test,y_pred))
print('accuracy_score-', accuracy_score (y_test,y_pred))
print('Model_score -',lr.score (x_test,y_test))
sns.heatmap(a, annot=True)
plt.show()
```

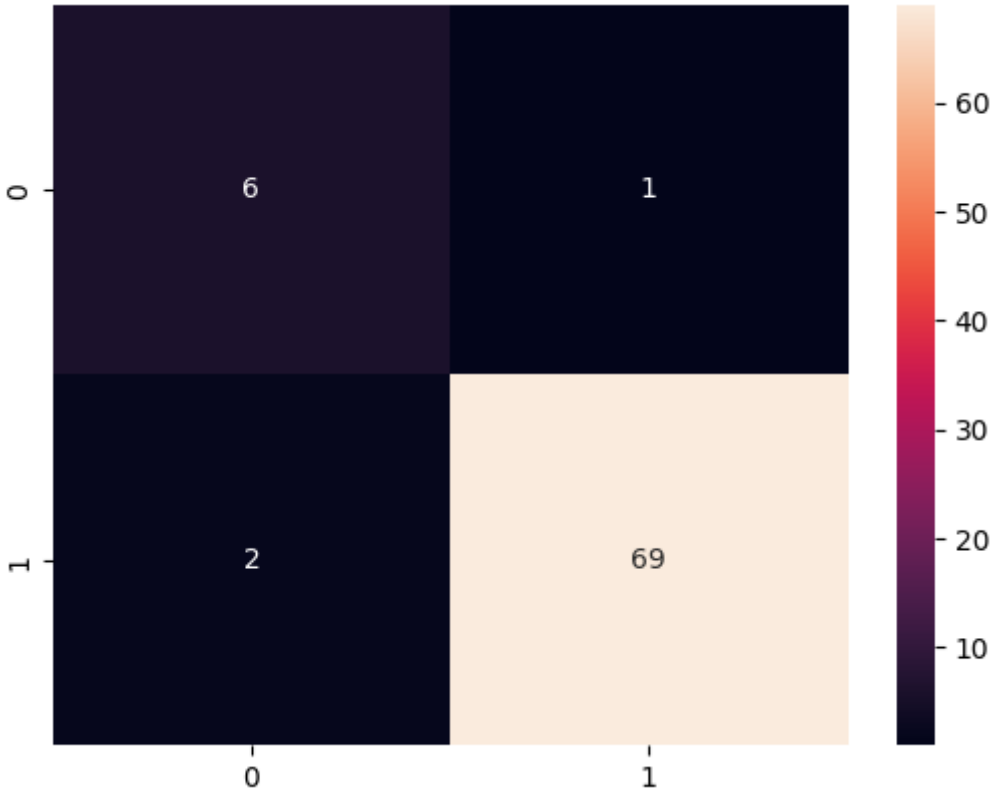
```
1 denotes 'YES, person have cancer'
'0 denotes NO,person does not have cancer
first 5 actual_y: [1 1 1 1 1] predicted y: [1 1 1 1 1]
confusion_matrix
[[ 5  2]
 [ 4 67]]
classification_report
              precision    recall  f1-score   support

           0       0.56      0.71      0.63         7
           1       0.97      0.94      0.96        71

    accuracy                           0.92        78
   macro avg       0.76      0.83      0.79        78
weighted avg       0.93      0.92      0.93        78

accuracy_score- 0.9230769230769231
Model_score - 0.9230769230769231
```



Random forest

In [46]:
```
from sklearn.ensemble import RandomForestClassifier
lr=RandomForestClassifier()
lr.fit(x_train, y_train)
y_pred=lr.predict(x_test)
print("1 denotes 'YES, person have cancer'\n'0 denotes NO,person does not have canc
print (f'first 5 actual_y: {y_test.values[:5]} predicted y: {y_pred[:5]}')
from sklearn.metrics import confusion_matrix, classification_report, accuracy_score
print('confusion_matrix')
a=confusion_matrix (y_test,y_pred)
print(a)
print('classification_report')
print(classification_report (y_test,y_pred))
print('accuracy_score-', accuracy_score (y_test,y_pred))
print('Model_score -',lr.score (x_test,y_test))
sns.heatmap(a, annot=True)
plt.show()
```

```
1 denotes 'YES, person have cancer'
'0 denotes NO,person does not have cancer
first 5 actual_y: [1 1 1 1 1] predicted y: [1 1 1 1 1]
confusion_matrix
[[ 6  1]
 [ 2 69]]
classification_report
              precision    recall  f1-score   support

           0       0.75      0.86      0.80         7
           1       0.99      0.97      0.98        71

    accuracy                           0.96        78
   macro avg       0.87      0.91      0.89        78
weighted avg       0.96      0.96      0.96        78

accuracy_score- 0.9615384615384616
Model_score - 0.9615384615384616
```



accuracy of all the model for the given problem is given below

1. linear Regression= 0.2079084725452871
2. Logistic regresssion=0.9487179487179487
3. k nearest neighbour=0.9102564102564102
4. decision tree classifier=0.9230769230769231
5. random forest classifier=0.9615384615384616

from all of the above model the accuracy of the random forest is maximum hence we will use random forest algorithm to detect the lung cancer

In [ ]:

In [ ]: