Problem Statement:-

We are working on data from Thoracic surgery of primary lung cancer from this we are going to find accuracy of patient will die or survive from after 1yr year of surgery. for that we are going to usee Random Forest algorithm to find accuracy of survival.

Random Forest

Random Forest is a machine learning algorithm that is commonly used for classification, regression, and feature selection tasks. It is an ensemble learning method that combines multiple decision trees to improve the accuracy and generalization of the model.

We are using sklearn to import RandomForest . scikit-learn sklearn is a popular Python library for machine learning. It provides a wide variety of machine learning algorithms and tools for data preprocessing, model selection, and evaluation.

sklearn includes a range of supervised and unsupervised learning algorithms, including linear regression, logistic regression, support vector machines (SVMs), k-nearest neighbors (KNN), decision trees, random forests, gradient boosting, clustering algorithms, and many more.

```python
#import libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
```

```python
#inserting data set
from google.colab import drive
drive.mount('/content/drive')
data = pd.read_csv('/content/drive/MyDrive/kgce lms/expt7/expt7.csv')
#printing dataset
data
```

```
Drive already mounted at /content/drive; to attempt to forcibly
remount, call drive.mount("/content/drive", force_remount=True).
```

```
      DGN  PRE4  PRE5  PRE6 PRE7 PRE8 PRE9 PRE10 PRE11 PRE14 PRE17
PRE19  \
0    DGN2  2.88  2.16  PRZ1    F    F    F     T     T  OC14     F
F
1    DGN3  3.40  1.88  PRZ0    F    F    F     F     F  OC12     F
F
2    DGN3  2.76  2.08  PRZ1    F    F    F     T     F  OC11     F
F
```

```
3    DGN3  3.68  3.04  PRZ0    F    F    F     F     F  OC11     F
F
4    DGN3  2.44  0.96  PRZ2    F    T    F     T     T  OC11     F
F
..    ...   ...   ...   ...  ...  ...  ...   ...   ...   ...   ...   .
..
465  DGN2  3.88  2.12  PRZ1    F    F    F     T     F  OC13     F
F
466  DGN3  3.76  3.12  PRZ0    F    F    F     F     F  OC11     F
F
467  DGN3  3.04  2.08  PRZ1    F    F    F     T     F  OC13     F
F
468  DGN3  1.96  1.68  PRZ1    F    F    F     T     T  OC12     F
F
469  DGN3  4.72  3.56  PRZ0    F    F    F     F     F  OC12     F
F

    PRE25 PRE30 PRE32  AGE Risk1Yr
0       F     T     F   60       F
1       F     T     F   51       F
2       F     T     F   59       F
3       F     F     F   54       F
4       F     T     F   73       T
..    ...   ...   ...  ...     ...
465     F     T     F   63       F
466     F     T     F   61       F
467     F     F     F   52       F
468     F     T     F   79       F
469     F     T     F   51       F

[470 rows x 17 columns]
```

```python
#import LabelEndcoder from sklearn.preprocessing
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()

data['DGN'] = le.fit_transform(data['DGN'])

# data['PRE4'] = le.fit_transform(data['PRE4'])

# data['PRE5'] = le.fit_transform(data['PRE5'])

data['PRE6'] = le.fit_transform(data['PRE6'])

data['PRE7'] = le.fit_transform(data['PRE7'])

data['PRE8'] = le.fit_transform(data['PRE8'])

data['PRE9'] = le.fit_transform(data['PRE9'])
```

```python
data['PRE10'] = le.fit_transform(data['PRE10'])

data['PRE11'] = le.fit_transform(data['PRE11'])

data['PRE14'] = le.fit_transform(data['PRE14'])

data['PRE17'] = le.fit_transform(data['PRE17'])

data['PRE19'] = le.fit_transform(data['PRE19'])

data['PRE25'] = le.fit_transform(data['PRE25'])

data['PRE30'] = le.fit_transform(data['PRE30'])

data['PRE32'] = le.fit_transform(data['PRE32'])

data['Risk1Yr'] = le.fit_transform(data['Risk1Yr'])
```

```python
data
```

```
     DGN  PRE4  PRE5  PRE6  PRE7  PRE8  PRE9  PRE10  PRE11  PRE14  
PRE17  \
0      1  2.88  2.16     1     0     0     0      1      1      3  
0
1      2  3.40  1.88     0     0     0     0      0      0      1  
0
2      2  2.76  2.08     1     0     0     0      1      0      0  
0
3      2  3.68  3.04     0     0     0     0      0      0      0  
0
4      2  2.44  0.96     2     0     1     0      1      1      0  
0
..   ...   ...   ...   ...   ...   ...   ...    ...    ...    ...    .
..
465    1  3.88  2.12     1     0     0     0      1      0      2  
0
466    2  3.76  3.12     0     0     0     0      0      0      0  
0
467    2  3.04  2.08     1     0     0     0      1      0      2  
0
468    2  1.96  1.68     1     0     0     0      1      1      1  
0
469    2  4.72  3.56     0     0     0     0      0      0      1  
0

     PRE19  PRE25  PRE30  PRE32  AGE  Risk1Yr
0        0      0      1      0   60        0
1        0      0      1      0   51        0
2        0      0      1      0   59        0
3        0      0      0      0   54        0
```

```
4         0      0      1      0   73        1
..      ...    ...    ...    ...  ...      ...
465       0      0      1      0   63        0
466       0      0      1      0   61        0
467       0      0      0      0   52        0
468       0      0      1      0   79        0
469       0      0      1      0   51        0

[470 rows x 17 columns]
```

```
#allocing data / bindind data to variable for prediction
y = data['Risk1Yr']
x = data.drop(['Risk1Yr'], axis = 1)
```

```
#printing x
x
```

```
     DGN  PRE4  PRE5  PRE6  PRE7  PRE8  PRE9  PRE10  PRE11  PRE14
PRE17  \
0      1  2.88  2.16     1     0     0     0      1      1      3
0
1      2  3.40  1.88     0     0     0     0      0      0      1
0
2      2  2.76  2.08     1     0     0     0      1      0      0
0
3      2  3.68  3.04     0     0     0     0      0      0      0
0
4      2  2.44  0.96     2     0     1     0      1      1      0
0
..   ...   ...   ...   ...   ...   ...   ...    ...    ...    ...    .
..
465    1  3.88  2.12     1     0     0     0      1      0      2
0
466    2  3.76  3.12     0     0     0     0      0      0      0
0
467    2  3.04  2.08     1     0     0     0      1      0      2
0
468    2  1.96  1.68     1     0     0     0      1      1      1
0
469    2  4.72  3.56     0     0     0     0      0      0      1
0

     PRE19  PRE25  PRE30  PRE32  AGE
0        0      0      1      0   60
1        0      0      1      0   51
2        0      0      1      0   59
3        0      0      0      0   54
4        0      0      1      0   73
..     ...    ...    ...    ...  ...
465      0      0      1      0   63
466      0      0      1      0   61
```

```
467      0      0      0      0   52
468      0      0      1      0   79
469      0      0      1      0   51

[470 rows x 16 columns]
```

```
#printing y
y
```

```
0      0
1      0
2      0
3      0
4      1
      ..
465    0
466    0
467    0
468    0
469    0
Name: Risk1Yr, Length: 470, dtype: int64
```

```
#Import the train_test_split from the sklearn
#train_test_split is a method used in machine learning to split a 
dataset into two subsets
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size = 
0.3)
print(x_train.shape)
print(x_test.shape)
print(y_train.shape)
print(y_test.shape)
```

```
(329, 16)
(141, 16)
(329,)
(141,)
```

```
#Import the RandomForestClassifier from the sklearn
from sklearn.ensemble import RandomForestClassifier
model=RandomForestClassifier()
model.fit(x_train,y_train)
```

```
RandomForestClassifier()
```

```
y_predict=model.predict(x_test)
y_predict
```

```
array([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, 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, 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, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0,
       0, 0, 0, 0, 0, 0, 0, 0, 0])
```

```python
#Import the accuracy_score from the sklearn
from sklearn.metrics import accuracy_score
print('Accuracy of survival after one year of surgery is')
accuracy_score(y_test,y_predict)*100
```

```
Accuracy of survival after one year of surgery is
```

```
85.1063829787234
```