```
# Input2
Kumar.K.R.Devose
```

```
# Import libraries
import pandas as pd
from sklearn.datasets import load_breast_cancer
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report, accuracy_score,confusion_matrix
from sklearn.tree import DecisionTree
from sklearn.tree import DecisionTreeClassifier, plot_tree
from sklearn.ensemble import RandomForestClassifier
from sklearn.tree import *
from sklearn.preprocessing import StandardScaler
```

```
# Load the breast cancer dataset
cancer = load_breast_cancer()
```

```
df = pd.DataFrame(cancer.data, columns=cancer.feature_names)
df['target'] = cancer.target
```

```
# Display the first few rows of the dataset
print(df.head())
```

```
   mean radius  mean texture  mean perimeter  mean area  mean smoothness  \
0        17.99         10.38          122.80     1001.0          0.11840
1        20.57         17.77          132.90     1326.0          0.08474
2        19.69         21.25          130.00     1203.0          0.10960
3        11.42         20.38           77.58      386.1          0.14250
4        20.29         14.34          135.10     1297.0          0.10030

   mean compactness  mean concavity  mean concave points  mean symmetry  \
0           0.27760          0.3001              0.14710         0.2419
1           0.07864          0.0869              0.07017         0.1812
2           0.15990          0.1974              0.12790         0.2069
3           0.28390          0.2414              0.10520         0.2597
4           0.13278          0.1980              0.10043         0.1809

   mean fractal dimension  ...  worst texture  worst perimeter  \
0                 0.07871  ...          17.33           184.60
1                 0.05667  ...          23.41           158.80
2                 0.05999  ...          25.53           152.50
3                 0.09744  ...          26.50            98.87
4                 0.05883  ...          16.67           152.20

   worst area  worst smoothness  worst compactness  worst concavity  \
0      2019.0            0.1622             0.6656           0.7119
1      1956.0            0.1238             0.1866           0.2416
2      1709.0            0.1444             0.4245           0.4504
3       567.7            0.2098             0.8663           0.6869
4      1575.0            0.1374             0.2050           0.4000

   worst concave points  worst symmetry  worst fractal dimension  target
0                0.2654          0.4601                  0.11890       0
1                0.186           0.275                   0.08902       0
2                0.2430          0.3613                  0.08758       0
3                0.2575          0.6638                  0.17300       0
4                0.1625          0.2364                  0.07678       0

[5 rows x 31 columns]
```

```
print(df.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 569 entries, 0 to 568
Data columns (total 31 columns):
 #   Column                   Non-Null Count  Dtype
---  ------                   --------------  -----
 0   mean radius              569 non-null    float64
 1   mean texture             569 non-null    float64
 2   mean perimeter           569 non-null    float64
 3   mean area                569 non-null    float64
 4   mean smoothness          569 non-null    float64
 5   mean compactness         569 non-null    float64
 6   mean concavity           569 non-null    float64
 7   mean concave points      569 non-null    float64
 8   mean symmetry            569 non-null    float64
 9   mean fractal dimension   569 non-null    float64
 10  radius error             569 non-null    float64
 11  texture error            569 non-null    float64
 12  perimeter error          569 non-null    float64
 13  area error               569 non-null    float64
 14  smoothness error         569 non-null    float64
 15  compactness error        569 non-null    float64
 16  concavity error          569 non-null    float64
 17  concave points error     569 non-null    float64
 18  symmetry error           569 non-null    float64
 19  fractal dimension error  569 non-null    float64
 20  worst radius             569 non-null    float64
 21  worst texture            569 non-null    float64
 22  worst perimeter          569 non-null    float64
 23  worst area               569 non-null    float64
 24  worst smoothness         569 non-null    float64
 25  worst compactness        569 non-null    float64
 26  worst concavity          569 non-null    float64
 27  worst concave points     569 non-null    float64
 28  worst symmetry           569 non-null    float64
 29  worst fractal dimension  569 non-null    float64
 30  target                   569 non-null    int64
dtypes: float64(30), int64(1)
memory usage: 138.1 KB
None
```

```
print(df.describe())
```

```
       mean radius  mean texture  mean perimeter    mean area  \
count   569.000000    569.000000      569.000000   569.000000
mean     14.127292     19.289649       91.969033   654.889104
std       3.524049      4.301036       24.298981   351.914129
min       6.981000      9.710000       43.790000   143.500000
25%      11.700000     16.170000       75.170000   420.300000
50%      13.370000     18.842000       86.240000   551.100000
75%      15.780000     21.080000      104.100000   782.700000
max      28.110000     39.280000      188.500000  2501.000000

       mean smoothness  mean compactness  mean concavity  mean concave points  \
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          0.019300        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

       mean symmetry  mean fractal dimension  ...  worst texture  \
count     569.000000              569.000000  ...     569.000000
mean       0.181162                0.062798  ...      25.677223
std        0.027414                0.007060  ...       6.146258
min        0.106000                0.049960  ...      12.020000
25%        0.161900                0.057700  ...      21.080000
50%        0.179200                0.061540  ...      25.410000
75%        0.195700                0.066120  ...      29.720000
max        0.304000                0.097440  ...      49.540000

       worst perimeter  worst area  worst smoothness  worst compactness  \
count       569.000000  569.000000        569.000000         569.000000
mean        107.261213  880.583128          0.132369           0.254265
std          33.602542  569.356993          0.022832           0.157336
min          50.410000  185.200000          0.071170           0.027290
25%          84.110000  515.300000          0.116600           0.147200
50%          97.660000  686.500000          0.131300           0.211900
75%         125.400000  1084.000000         0.146000           0.339100
max         251.200000  4254.000000         0.223400           1.058000

       worst concavity  worst concave points  worst symmetry  \
count       569.000000            569.000000      569.000000
mean          0.272188              0.114606        0.290076
std           0.208624              0.065732        0.061867
min           0.000000              0.000000        0.156500
25%           0.114500              0.064930        0.250400
50%           0.226700              0.099930        0.282200
75%           0.382900              0.161400        0.317900
max           1.252000              0.291000        0.663800

       worst fractal dimension  target
count               569.000000  569.000
mean                  0.083946    0.627
std                   0.018061    0.484
min                   0.055040    0.000
25%                   0.071460    0.000
50%                   0.080040    1.000
75%                   0.092080    1.000
max                   0.207500    1.000

[8 rows x 31 columns]
```

```
df.isnull().sum()
```

```
0x1200)
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
```

```
# Split the data into features (X) and target (y)
X = df.drop(columns=['target'])
y = df['target']
```

```
# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
X_train.shape,X_test.shape
```

```
((455, 30), (114, 30))
```

```
# Standardize the features
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)
```

```
# Convert the scaled arrays back to DataFrames
X_train_scaled_df = pd.DataFrame(X_train_scaled, columns=X_train.columns)
X_test_scaled_df = pd.DataFrame(X_test_scaled, columns=X_test.columns)

# Now you can use the head() method
print(X_train_scaled_df.head())
print(X_test_scaled_df.head())
```

```
   mean radius  mean texture  mean perimeter  mean area  mean smoothness  \
0    -1.446753     -0.435316       -1.352985  -1.139118         0.799573
1     0.209291     -1.305831        0.203879   0.070100        -0.826962
2     1.500402     -1.134922        1.391225   1.363224        -1.100204
3    -0.991797     -1.416222       -0.962367  -0.908544        -0.868398
4     -1.11798     -1.018398       -1.120967  -0.969042        -1.090511

   mean compactness  mean concavity  mean concave points  mean symmetry  \
0          0.136157        0.035294             0.092732       0.465088
1          1.428275        2.011212             2.107603       1.137964
2         -0.367255       -0.365918            -0.004587       1.583032
3          0.054151        0.043468             0.115732       0.123920
4          -0.43962       -0.69151             -0.044904       1.204336

   mean fractal dimension  ...  worst radius  worst texture  worst perimeter  \
0               -0.248419  ...     -1.232861      -0.470595        -1.247958
1               1.594306  ...      2.172314       1.311279         2.303117
2               0.773693  ...      1.444482       2.165657         1.444482
3               0.014041  ...     -0.820247      -1.085536        -0.873872
4               0.968513  ...     -1.081129      -1.334616        -1.117138

   worst area  worst smoothness  worst compactness  worst concavity  \
0   -0.973968          0.722004           1.180732         0.973819
1    2.232222          1.274827           3.294808         2.074370
2    1.091227          1.190431           3.548904         3.044911
3   -0.742947          0.798624           -0.779202        -0.774950
4   -0.908169          0.173475           -0.980395        -1.205149

   worst concave points  worst symmetry  worst fractal dimension
0              1.523267        1.239207                 0.161992
1              2.888457        1.001202                 2.057830
2              2.879239        0.587903                 1.857994
3              0.980747        0.548677                 0.828422
4              1.151502        1.033544                 0.365311

[5 rows x 30 columns]
   mean radius  mean texture  mean perimeter  mean area  mean smoothness  \
0    -0.404487     -0.117399       -0.443211  -0.486403         0.200850
1    -0.307416     -1.489656       -0.297177  -0.364775        -0.786125
2    -0.380904     -0.905230       -0.404245  -0.445604         0.967526
3    -0.406317     -0.551105       -0.420670  -0.526153         0.994290
4     -0.72953     -1.115147       -0.745201  -0.705251         1.204512

   mean compactness  mean concavity  mean concave points  mean symmetry  \
0          0.196004        0.343010             0.362775       0.201369
1         -0.621214        0.149637             0.201482       0.352133
2          0.358414        0.715992             0.957221       0.477904
3          0.543389        0.037708             0.475450       0.773241
4          0.150698        0.271202             0.587608       0.615142

   mean fractal dimension  ...  worst radius  worst texture  worst perimeter  \
0               0.425732  ...     -0.401964       0.251548        -0.418768
1               0.890834  ...     -1.114703      -1.069475        -1.090064
2               0.212575  ...      0.236653       0.173235         0.228082
3               0.643943  ...      0.474630       0.570630         0.515922
4               0.601436  ...      0.107330       0.913225         0.132877

   worst area  worst smoothness  worst compactness  worst concavity  \
0   -0.351059          0.146753          -0.035708         0.202482
1   -0.835536         -0.048138          -0.629893        -0.686243
2    0.078348          0.351157          0.295316         0.543032
3    0.381672          0.635928          0.284837         0.375607
4   -0.726243          0.128718          -0.284407        -0.053677

   worst concave points  worst symmetry  worst fractal dimension
0             0.226238        0.268578                 0.254775
1            -0.746811       -0.067042                 0.291861
2             0.548391        0.035337                 0.620634
3             0.835762        0.546332                 0.547691
4             0.098564        -0.031126                 0.187799

[5 rows x 30 columns]
```

```
# Initialize and train a RandomForestClassifier
clf = RandomForestClassifier(random_state=42)
clf.fit(X_train_scaled, y_train)
```

```
RandomForestClassifier(random_state=42)
```

```
# Make predictions on the test set
y_pred = clf.predict(X_test_scaled)
y_pred
```

```
array([1, 0, 0, 1, 1, 0, 0, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 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,
       1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 0, 0, 1, 1, 1, 0, 0,
       1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1])
```

```
print("y_pred",y_pred)
print("y_test",y_test)
```

```
y_pred [1 0 0 1 1 0 0 0 1 1 1 0 1 1 1 1 1 1 0 1 1 1 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 1 0 1 1 1 1 1 1 1 1 1 1 1 0 0 0 1 0 0 1 1 1 0 0 1 0 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1
 1 1 0]
y_test 1    1
0    0
1    0
1    1
0    1
Name: target, dtype: int64
```

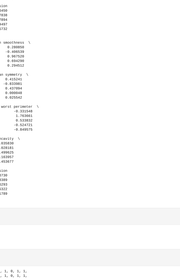
```
# Evaluate the model
print("Accuracy", accuracy_score(y_test, y_pred))
print("Classification Report:\n", classification_report(y_test, y_pred))
```

```
Accuracy 0.9649122807017544
Classification Report:
               precision    recall  f1-score   support

           0       0.98      0.93      0.95        43
           1       0.96      0.99      0.97        71

    accuracy                           0.96       114
   macro avg       0.97      0.96      0.96       114
weighted avg       0.97      0.96      0.96       114
```

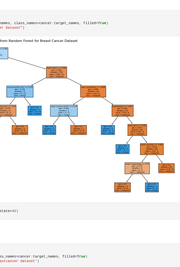
```
# Visualize one of the trees from the forest
plt.figure(figsize=(20,10))
plot_tree(clf[0], filled=True, feature_names=cancer.feature_names, class_names=cancer.target_names, filled=True)
plt.title("Decision Tree From Random Forest for Breast Cancer Dataset")
plt.show()
```




```
# Train Decision Tree Classifier with Gini criterion
clf_gini = DecisionTreeClassifier(criterion='gini', random_state=42)
clf_gini.fit(X_train_scaled, y_train)
```

```
DecisionTreeClassifier(random_state=42)
```

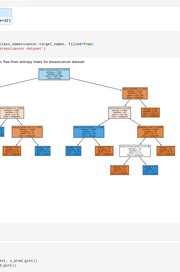
```
# Visualize the decision tree with Gini criterion
plt.figure(figsize=(20,10))
plot_tree(clf_gini, filled=True, feature_names=cancer.feature_names, class_names=cancer.target_names, filled=True)
plt.title("Creating a Decision Tree From Gini Index for breastcancer dataset")
plt.show()
```




```
# Train Decision Tree Classifier with entropy criterion
clf_entropy = DecisionTreeClassifier(criterion='entropy', random_state=42)
clf_entropy.fit(X_train_scaled, y_train)
```

```
DecisionTreeClassifier(criterion='entropy', random_state=42)
```

```
# Visualize the decision tree with entropy criterion
plt.figure(figsize=(20,10))
plot_tree(clf_entropy, feature_names=cancer.feature_names, class_names=cancer.target_names, filled=True)
plt.title("Creating a Decision Tree From entropy Index for breastcancer dataset")
plt.show()
```




```
# Make predictions
y_pred_gini = clf_gini.predict(X_test_scaled)
y_pred_entropy = clf_entropy.predict(X_test_scaled)
```

```
# Evaluate the models
print("Decision Tree with Gini Index")
print("Accuracy", accuracy_score(y_test, y_pred_gini))
print("Classification Report:\n", classification_report(y_test, y_pred_gini))
print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred_gini))

print("\nDecision Tree with Entropy")
print("Accuracy", accuracy_score(y_test, y_pred_entropy))
print("Classification Report:\n", classification_report(y_test, y_pred_entropy))
print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred_entropy))
```

```
Decision Tree with Gini Index
Accuracy 0.9473684210526315
Classification Report:
               precision    recall  f1-score   support

           0       0.93      0.93      0.93        43
           1       0.96      0.96      0.96        71

    accuracy                           0.95       114
   macro avg       0.94      0.94      0.94       114
weighted avg       0.95      0.95      0.95       114

Confusion Matrix:
 [[40 3]
 [ 3 68]]

Decision Tree with Entropy
Accuracy 0.9473684210526315
Classification Report:
               precision    recall  f1-score   support

           0       0.97      0.86      0.91        43
           1       0.93      0.99      0.96        71

    accuracy                           0.95       114
   macro avg       0.95      0.93      0.94       114
weighted avg       0.95      0.95      0.95       114

Confusion Matrix:
 [[37 6]
 [ 1 70]]
```

```
# ANSWER:TO AND CONCLUSIONS
```

```
# 10 Is this product, a bank or insurance company product?
Based on the model, the product performance was tested by the model accuracy on both the cases were > 0.9798
1 In both the cases the training set accuracy and testing set accuracy score are almost same
2 The confusion matrix and classification report yields very good model performance
```