IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Date of publication July 11, 2023

Digital Object Identifier TBD

Exploration of Decision Trees for classification

PRATIGYA PAUDEL[1], SUSHANK GHIMIRE[1]
[1]Institute of Engineering, Thapathali Campus, Bagmati 44600 Nepal (e-mail: pratigyapaudel0@gmail.com)
Corresponding author: Pratigya Paudel (e-mail: pratigyapaudell0@gmail.com).

"This work was completed as a part of a college practical for Data Mining (CT725)."

ABSTRACT Decision tress fall under the class of machine learning algorithms. Decision trees are a popular and powerful machine learning algorithm used for classification and regression tasks. They are intuitive, easy to understand, and offer great interpretability, making them widely adopted in various domains. This paper focuses on the binary classification of breast cancer toy dataset to determine whether the given attributes lead to the cancer being malignant or benign. Different results are obtained while classifying the dataset based on information gain, classification error and Gini index.

INDEX TERMS Decision Tree, Supervised Machine Learning, Gini Index

I. INTRODUCTION

DECISION TREE is a popular non-parametric supervised learning technique utilized for both classification and regression tasks. It employs a hierarchical structure resembling a tree to make predictions by utilizing simple decision rules derived from the input features. This versatile algorithm is capable of handling various types of problems in machine learning. Decision trees can be thought of as an approximation method that segments data into distinct regions, resembling conditional control statements in an algorithm.

The breast cancer dataset is a well-known and widely used benchmark in machine learning for classification tasks related to breast cancer diagnosis. It contains various attributes describing breast mass characteristics, including demographic and clinical information, and serves as the basis for developing predictive models. The dataset is widely used for testing and ranking the performance of classification algorithms. The various attributes in the dataset leads to the result of the particular case being benign or malignant. The even distribution of benign and malignant cases in the dataset makes it perfect to be classified using a decision tree. This lab focuses on the classification of the dataset using decision tree, using different indices.

II. METHODOLOGY

A. THEORY

Decision tree works by recursively splitting the data based on the attribute values. For the given number of features, the feature with the highest information gain, classification error or the Gini index is kept as the root node and the subsequent highest information gain features are added as nodes. Traversing down the tree-like structure brings about an end with a leaf, which is determined by a decision boundary. The leaf node is the target classification for the given set of attributes.

Decision trees offer a high level of interpretability. The decision rules learned by the tree can be easily understood and visualized, making them suitable for domains where transparency and explainability are important, such as healthcare or finance. Using the relevant features, the data can be analyzed and categorized to set clear decision boundaries for making the decision tree nodes.

The classification report is a common tool used to evaluate the performance of a classification model. It provides a detailed summary of various metrics that assess how well the model has performed in classifying different classes or categories. The report is typically organized in a tabular format, with rows representing each class and columns representing different evaluation metrics. The most commonly included metrics in a classification report are:

- Precision: Precision is the ratio of true positive predictions to the total number of positive predictions. It measures the accuracy of the model in correctly identifying positive instances.
- Recall: Recall, also known as sensitivity or true positive rate, is the ratio of true positive predictions to the total number of actual positive instances. It measures the model's ability to correctly identify positive instances.
- F1-score: The F1-score is the harmonic mean of precision and recall. It provides a balanced measure of the model's performance, taking into account both precision and recall.

- Support: Support refers to the number of actual occurrences of each class in the dataset. It represents the number of samples belonging to a particular class.
- Accuracy: Accuracy is the overall correctness of the model's predictions. It measures the proportion of correctly classified instances out of the total number of in
- Macro average: The macro average is the average of the performance metrics across all classes. It treats each class equally, regardless of their support.
- Weighted average: The weighted average is the average of the performance metrics, weighted by the support of each class. It provides a more accurate representation when classes have imbalanced support.

B. MATHEMATICAL FORMULAE

1) Dataset Split based on Information Gain

Entropy is an information theory metric that measures the impurity or uncertainty in a group of observations.
For N classes, entropy is calculated as:

$$H(S) = -\sum_{i=1}^{n} p_i \log_2(p_i) \tag{1}$$

For each attribute, the weighted average entropy is calculated after the split as:

$$H(S,A) = \sum_{v \in \text{values}(A)} \frac{|S_v|}{|S|} \cdot H(S_v) \tag{2}$$

where:

- $H(S,A)$ represents the weighted average entropy after the split using attribute A.
- v represents each unique value in the set of values of attribute A.
- $|S_v|$ denotes the number of instances in the dataset S that have the attribute value v.
- $|S|$ denotes the total number of instances in the dataset S.
- $H(S_v)$ denotes the entropy of the subset of instances in the dataset S that have the attribute value v.

Finally, the information gain from the split is determined as:

$$IG(S,A) = H(S) - H(S,A) \tag{3}$$

2) Dataset Split based on Gini Index

Gini index can be used for classifying the dataset. The gini index of a node is calculated as:

$$\text{Gini}(S) = 1 - \sum_{i=1}^{n} p_i^2 \tag{4}$$

where:

- $\text{Gini}(S)$ represents the Gini index of a node or dataset S.
- p_i is the proportion of samples in the node or dataset that belong to class i.
- The summation is taken over all n classes.

The formula for splitting based on Gini Index is:

$$\text{Gini Split}(S,A) = \sum_{v \in \text{values}(A)} \left(\frac{|S_v|}{|S|}\right) \cdot \text{Gini Impurity}(S_v) \tag{5}$$

where:

- $\text{Gini Split}(S,A)$ represents the Gini index of the split of dataset S based on attribute A.
- v represents each unique value in the set of values of attribute A.
- $|S_v|$ denotes the number of instances in the dataset S that have the attribute value v.
- $|S|$ denotes the total number of instances in the dataset S.
- $\text{Gini Impurity}(S_v)$ represents the Gini impurity of the subset of instances in the dataset S that have the attribute value v.

C. SYSTEM BLOCK DIAGRAM

The system workflow is as shown below:

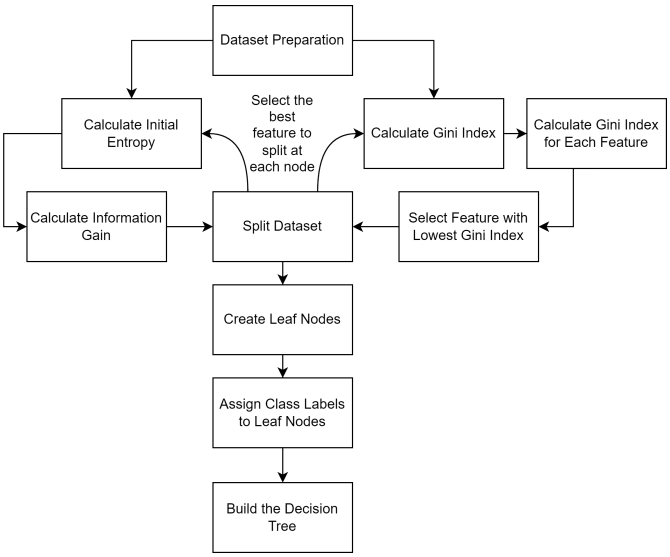

FIGURE 1. Building a Decision Tree

D. INSTRUMENTATION TOOLS

The entirety of the process is done using Python. Google Colab, short for Google Colaboratory, is an online platform provided by Google for running and sharing Jupyter notebook environments and it was used for all of the coding. Google colab provides a number of built-in functions for data analysis. The process of making the decision trees has been carried out using a number of available functions within the scikit-learn library. Initially, the dataset as a whole is loaded using the load-dataset function. The dataset is visualized using pandas. Then, the process of building a Decision Tree Classifier is done using the readily available functions within scikit-learn. The results are then visualized using different visualization tools like Seaborn and matplotlib.

TABLE 1. Breast-Cancer Dataset

Attribute	Sample 1 (Benign)	Sample 2 (Malignant)
Radius Mean	12.34	15.78
Texture Mean	23.54	34.67
Perimeter Mean	78.78	120.34
Area Mean	532.0	1189.0
Smoothness Mean	0.0865	0.1021
Compactness Mean	0.0703	0.2778
Concavity Mean	0.0346	0.3309
Concave Points Mean	0.0239	0.1974
Symmetry Mean	0.1724	0.2972
Fractal Dimension Mean	0.0625	0.0908
Radius SE	0.2012	0.6097
Texture SE	0.4505	1.132
Perimeter SE	1.501	4.001
Area SE	13.76	49.03
Smoothness SE	0.0059	0.0104
Compactness SE	0.0088	0.0366
Concavity SE	0.0197	0.2025
Concave Points SE	0.0052	0.0346
Symmetry SE	0.0184	0.0372
Fractal Dimension SE	0.0030	0.0054
Radius Worst	15.04	21.54
Texture Worst	29.5	45.34
Perimeter Worst	97.67	147.7
Area Worst	686.5	1552.0
Smoothness Worst	0.1023	0.1438
Compactness Worst	0.0926	0.7094
Concavity Worst	0.0622	0.7869
Concave Points Worst	0.0393	0.3184
Symmetry Worst	0.1727	0.3587
Fractal Dimension Worst	0.0587	0.0926

E. WORKING PRINCIPLE

1) Dataset Collection

The toy dataset for breast cancer is loaded from the scikit-learn library. The feature names, the target classes and the attribtues are then loaded to a Pandas dataframe and the values are visualized. The dataset is then split into training and validation dataset in the ratio of 67:33.

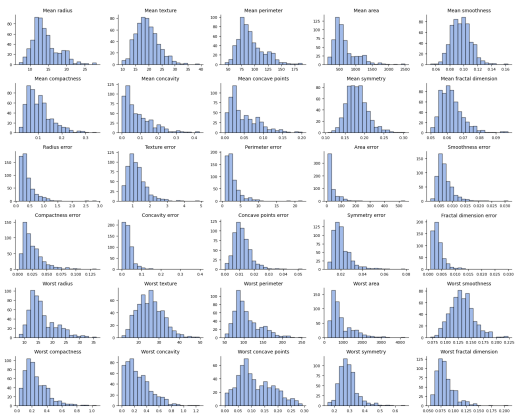

FIGURE 2. Histogram of Attribute values

2) Decision Tree Classifier Training

A decision tree object is initialized using the scikit-library for dataset split using both gini index and the information gain using entropy. The object instance is then trained on the training dataset. Using the trained classifier, the validation set is then loaded into it to predict the values. The results obtained from the classifier is compared to the actual labels of the validation dataset.

3) Data Analysis and Visualization

The decision tree made from training the classifier is visualized using different libraries. A confusion matrix is made with the prediction and actual values from the validation dataset.

III. RESULTS

The breast-cancer dataset consists of 30 trainable attributes, as shown in Table 1

A. DECISION TREE USING ENTROPY

We made a decision tree classifier to classify between the two target classes of breast cancer dataset . Three decision trees were made using the information gain from entropy. The first decision tree was allowed its maximum depth but the other two were limited to the depth of 3 and 2.

The decision tree as shown in Figure 4 shows a tree structure with no limitations on the depth. So, the maximum depth as seen from the tree is 6. The blue fill on the nodes represent the benign targets and the ones in orange represent the malignant cases. The colour of the nodes correspond with the likelihood of a given set of attributes being the actual target class. The white node in the tree represents a neutral node with an equal number of distribution for both the malignant and the benign cases.

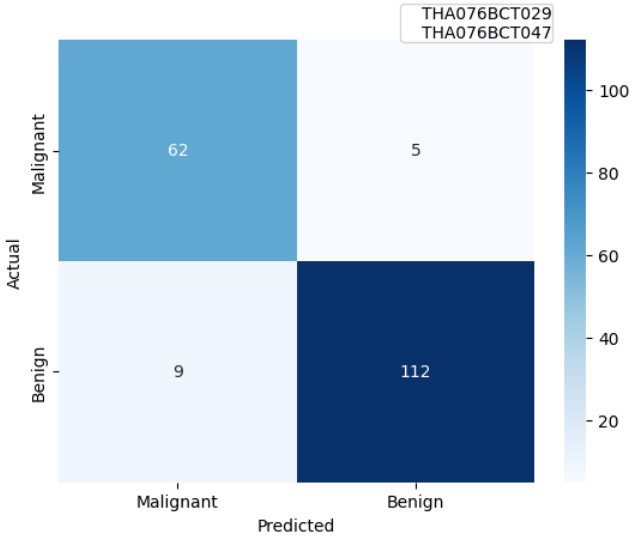

FIGURE 3. Confusion matrix for entropy based classifier

The confusion matrix in Figure 3 shows high density of data falling under the principal diagonal indicating a high number of validation dataset being classified correctly. There still are a few cases of malignant being labeled as benign and vice versa.

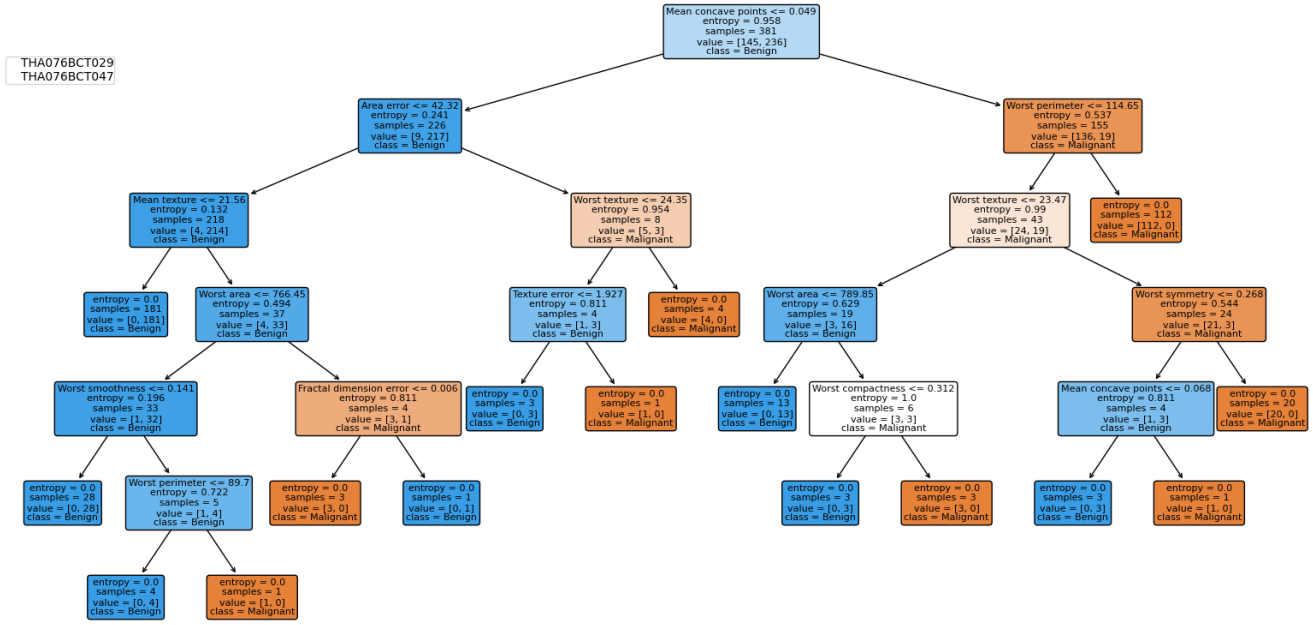

FIGURE 4. Decision Tree using Information Gain

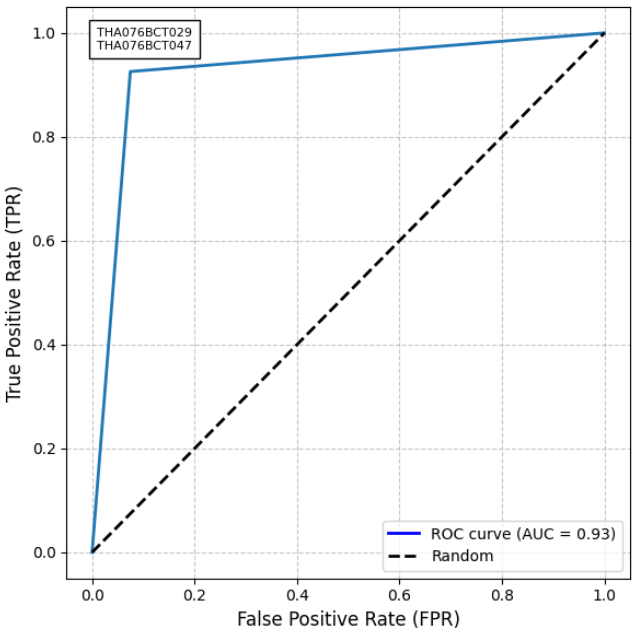

FIGURE 5. ROC plot for entropy based classifier

The Area Under the ROC Curve (Figure 5) summarizes the overall model performance. 0.93 refers to a good model performance.

The classification report shows high performance across the board for both the target classes using different metrics as in Table 2. The precision score for Benign class is much higher compared to the Malignant class. However, the complete opposite is seen with Recall scores for both.

TABLE 2. Classification Report using entropy based classifier

Class	Precision	Recall	F1-Score	Support
Malignant	0.79	0.90	0.84	67
Benign	0.94	0.87	0.90	121
Accuracy			0.88	188
Macro Avg	0.86	0.88	0.87	188
Weighted Avg	0.88	0.88	0.88	188

The decision tree when limited to the depth of 3 shows a much simpler structure as in Figure 6. Majority of the nodes in this tree classify the node as benign. 4 of the 7 leaf nodes are classified as benign cases. The simpler table leads to a confusion matrix that is very similar to Figure 3 but this one given as Figure 7 shows lesser number of classification along the principal diagonal for the malignant ones than the previous decision tree classifier.

The classification report for the decision tree limited to the maximum depth of 3 shows better overall results across the board, compared to when the report is made on the unconstrained decision tree.

TABLE 3. Classification Report using entropy based classifier(max-depth:3)

Class	Precision	Recall	F1-Score	Support
Malignant	0.90	0.90	0.90	67
Benign	0.94	0.94	0.94	121
Accuracy			0.93	188
Macro Avg	0.92	0.92	0.92	188
Weighted Avg	0.93	0.93	0.93	188

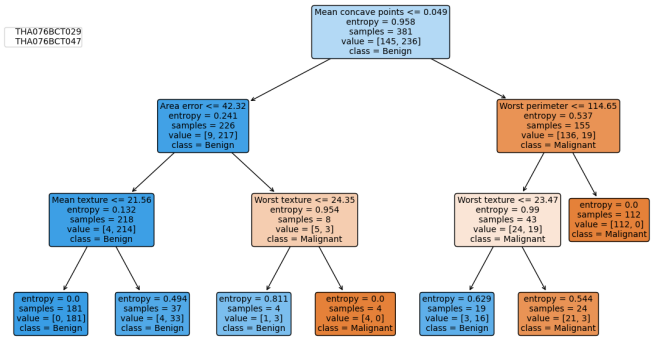

FIGURE 6. Decision Tree using Information Gain(max-depth=3)

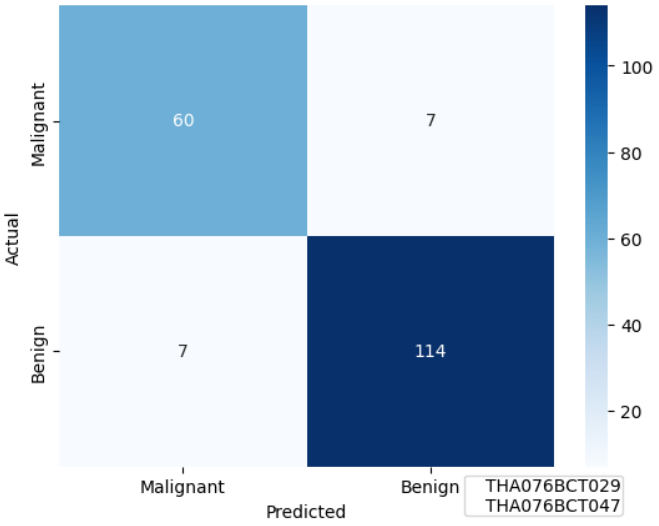

FIGURE 7. Confusion matrix for entropy based classifier(max-depth=3)

TABLE 4. Classification Report using entropy based classifier(max-depth:2)

Class	Precision	Recall	F1-Score	Support
Metrics	Precision	Recall	F1-Score	Support
Malignant	0.79	0.90	0.84	67
Benign	0.94	0.87	0.90	121
Accuracy			0.88	188
Macro Avg	0.86	0.88	0.87	188
Weighted Avg	0.88	0.88	0.88	188

The classification report for the decision tree, when the nodes are permitted to the maximum depth of 2 can be seen as in Table 4. The performance in this is degraded, compared to the results obtained with the maximum depth of 3. The decision tree is further simplified as in Figure 8 when the depth of the tree is fixed to a maximum value of 2. Contrary to the decision trees before, there is a surplus number of leaf nodes that end with an malignant case on the leaf node. Even with the lower number of leaf nodes for benign target, the confusion matrix in Figure 9 shows a high number of benign targets being correctly classified. This does however see a major drop from the numbers seen on the previous confusion matrices. The malignant targets are able to maintain their consistency with the previously seen results with a drop of 2 on the correctly predicted labels compared to the largest decision tree in Figure 4

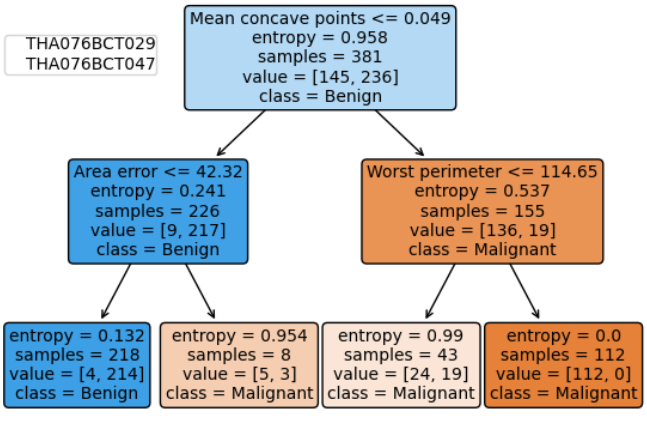

FIGURE 8. Decision Tree using Information Gain(max-depth=2)

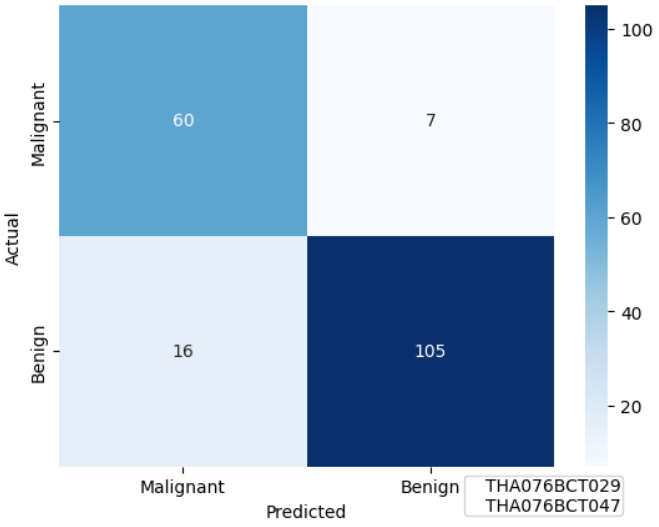

FIGURE 9. Confusion matrix for entropy based classifier(max-depth=2)

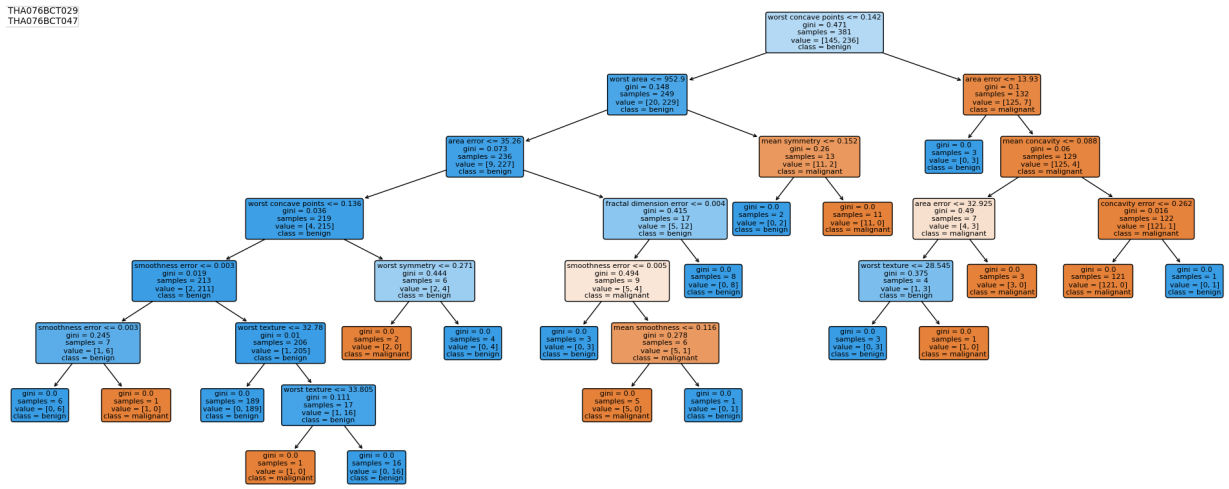

FIGURE 10. Confusion matrix for entropy based classifier

B. DECISION TREE USING GINI INDEX

The process of building a decision tree using Gini Index is similar to that of the process of building a tree using Entropy but the splitting criteria is specified using Gini Index. The decision trees made using Gini Index are of similar setups.

Figure 10 shows a decision tree classifier made by splitting the dataset using Gini Index. The tree is a larger structure than the one obtained from Information Gain. The maximum depth of the tree here is seen to be 7. There are a lot more leaf nodes that end with the target of benign than that of malignant, which is also in accordance with the volume of the distribution of the dataset in terms of the data targets.

The confusion matrix(Figure 11) for the classification results using the gini index shows the highest number of malignant classes being labelled correctly. While the benign classes have a high number as well, it still falls short behind the highest classified values.

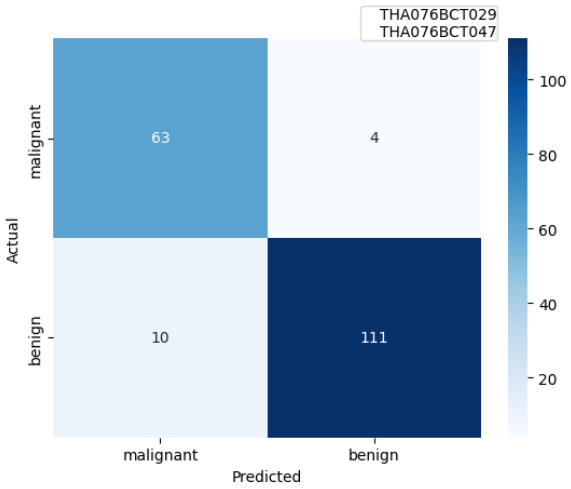

FIGURE 11. Confusion matrix for entropy based classifier

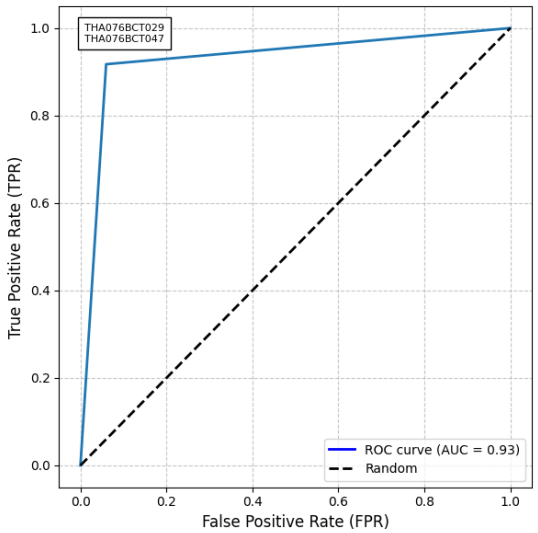

FIGURE 12. Confusion matrix for entropy based classifier

Figure 12 shows a similar ROC plot to Figure 5. The plot is over the same value, denoting similar performance across the classifiers.

The different values obtained from the different metrics from the classification report in Table 5 show a better performance than the classification reports that were seen before this.

TABLE 5. Classification Report using Gini Index

Class	Precision	Recall	F1-Score	Support
Metrics	Precision	Recall	F1-Score	Support
Malignant	0.86	0.94	0.90	67
Benign	0.97	0.92	0.94	121
Accuracy			0.93	188
Macro Avg	0.91	0.93	0.92	188
Weighted Avg	0.93	0.93	0.93	188

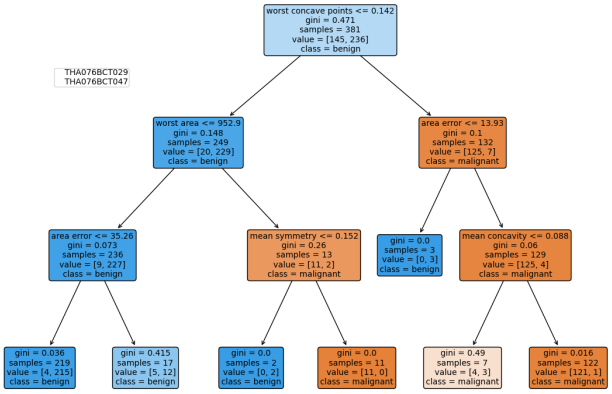

FIGURE 13. Decision Tree using Gini Index(max-depth=3)

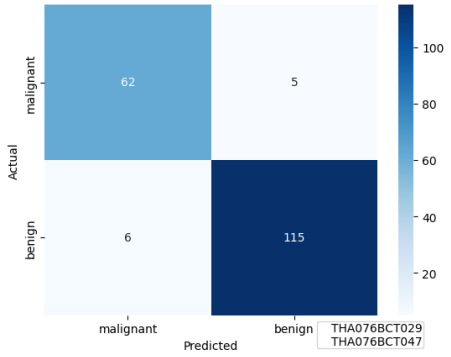

FIGURE 14. Confusion matrix for Gini Index Classifier(max-depth=3)

The decision tree limited to the maximum depth of 3 in Figure 13 is much simpler than the larger original Figure 10. The smaller tree is comparable to Figure 6 which had 7 leaf nodes, 4 of which belonged to the class Benign. The corresponding confusion matrix in Figure 14 shows the best performing decision tree in terms of correctly classifying the two target classes. The decision tree shows superior performance of the decision tree compared to the one featured in Figure 10.

Verifying the results from the classification report in Table 7, it is evident that the decision tree has been able to classify the dataset better than the unconstrained tree. With the decrease in the number of nodes, the decision tree classifier with maximum depth of 2 ends with equal proportion of benign and malignant classes. The pattern learnt from the dataset is similar.

The results from the classifier is portrayed in a confusion matrix as shown in Figure 16

The Table 7 shows different metrics and their tested quantity. Using the differnet scores, performance metrics are calculated. The given classification report displays how the decision classifier is very accurate.

TABLE 6. Classification Report using Gini Index(max-depth=3)

Class	Precision	Recall	F1-Score	Support
Malignant	0.91	0.93	0.92	67
Benign	0.96	0.95	0.95	121
Accuracy			0.94	188
Macro Avg	0.94	0.94	0.94	188
Weighted Avg	0.94	0.94	0.94	188

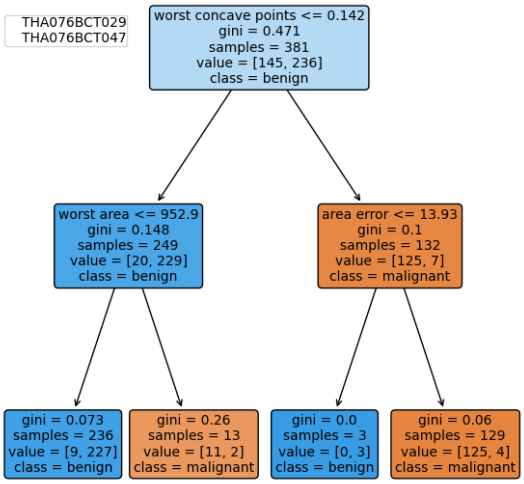

FIGURE 15. Confusion matrix for entropy based classifier

TABLE 7. Classification Report using Gini Index(max-depth=3)

Class	Precision	Recall	F1-Score	Support
Malignant	0.91	0.94	0.93	67
Benign	0.97	0.95	0.96	121
Accuracy			0.95	188
Macro Avg	0.94	0.95	0.94	188
Weighted Avg	0.95	0.95	0.95	188

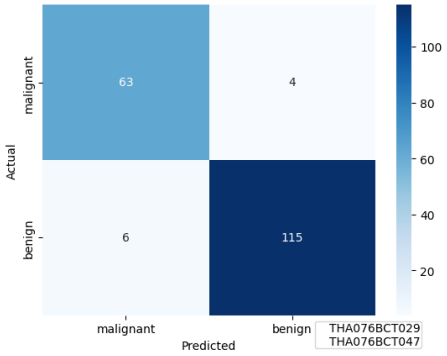

FIGURE 16. Confusion matrix for entropy based classifier

IV. DISCUSSION AND ANALYSIS

The previous sections demonstrated the successful usage of a Decision Tree Classifier for the breast cancer dataset. By applying the Decision Tree Classifier to the dataset, we were able to effectively classify the instances into different classes. The decision tree model reduced the complexity of the dataset by creating a hierarchical structure of decision rules based on the features. The resulting decision tree displayed a clear separation between different classes, with the nodes representing decision points and the leaf nodes representing the final class predictions.

One notable observation from the Decision Tree Classifier results is that the performance of the model is directly related to the depth of the tree. Deeper trees tend to have higher accuracy in distinguishing between the classes. However, it's important to note that excessively deep trees can lead to overfitting, capturing noise or irrelevant features in the dataset. Therefore, it's crucial to find the right balance by selecting an optimal depth for the decision tree that maximizes performance without overfitting. A clear example is set from the confusion matrices of the decision trees Figure 10, compared to the one with lower number of nodes, Figure 15.

Additionally, the Decision Tree Classifier allows for feature importance analysis. By examining the top-splitting nodes and their associated features, we can gain insights into the most important features for classifying the breast cancer instances. This information can be valuable for further analysis and understanding of the underlying factors contributing to the classification.

Decision Tree Classifier provides a powerful tool for classifying breast cancer instances based on the available features. It simplifies the decision-making process by creating a hierarchical structure of decision rules. By selecting an appropriate tree depth and considering feature importance, we can obtain accurate and interpretable results for classifying breast cancer cases using the decision tree model.

V. CONCLUSION

This lab was conducted in accordance with the principles of using a Decision Tree Classifier. The application of the Decision Tree Classifier on the breast cancer dataset demonstrates its effectiveness in this specific context. The main objective of using a Decision Tree Classifier, which is classification, was successfully achieved by accurately representing the decision rules derived from the breast cancer dataset. By considering various features and selecting the appropriate decision rules, the Decision Tree Classifier effectively captured the patterns and relationships within the breast cancer dataset, leading to accurate classification results.

The analysis reveals that the performance of the Decision Tree Classifier is influenced by several factors, including the depth of the tree and the selection of features specific to the breast cancer dataset. Deeper trees tend to provide higher accuracy in classifying breast cancer instances. However, it is important to balance the depth to avoid overfitting and capture relevant features only. The selection of features, determined by the decision rules, plays a crucial role in accurately predicting the class labels for breast cancer cases. By considering the most informative features specific to breast cancer, the Decision Tree Classifier can effectively distinguish between benign and malignant instances. The results obtained from the Decision Tree Classifier highlight its capability to handle breast cancer classification tasks and provide interpretable models. By examining the decision rules and their associated features, valuable insights can be gained regarding the factors influencing the classification outcomes for breast cancer cases. Additionally, the Decision Tree Classifier allows for feature importance analysis, providing an understanding of the most influential features specific to breast cancer in the classification process.

The utilization of the Decision Tree Classifier proves its efficacy in classifying breast cancer instances. By considering the depth, feature selection, and interpreting the decision rules specific to breast cancer, accurate and interpretable classification results can be achieved. The Decision Tree Classifier offers a valuable approach to gain insights into the underlying patterns and relationships within the breast cancer dataset, making it a powerful tool for breast cancer classification tasks.

While the Decision Tree Classifier is a powerful tool for classifying breast cancer instances, it has a few limitations that should be considered. One of the demerits is its tendency to overfit the training data, especially when the tree is allowed to grow to a deep level.

VI. REFERENCES

- David Bowser-Chao and Debra L. Dzialo. "Comparison of the use of binary decision trees and neural networks in top-quark detection." *Physical Review D*, vol. 47, no. 5, pp. 1900–1905, Mar. 1993. doi: 10.1103/physrevd.47.1900. Available at: https://doi.org/10.1103%2Fphysrevd.47.1900.
- P. M. Murphy and M. J. Pazzani. "Exploring the Decision Forest: An Empirical Investigation of Occam's Razor in Decision Tree Induction." 1994. arXiv preprint cs/9403101. Available at: https://arxiv.org/abs/cs/9403101.

PRATIGYA PAUDEL is a fourth year student, studying computer engineering under IOE, Thapathali Campus. She has been involved in a lot of machine learning projects and has a keen eye for data analysis and AI related stuff. With the enthusiasm for Artificial Intelligence (AI), she is driven by the potential of AI to transform industries and tackle complex challenges. Her academic journey has equipped her with a strong foundation in AI concepts, including machine learning and data analysis. She possesses a relentless curiosity and is always eager to explore the latest advancements in AI. Her goal is to apply her knowledge and make a meaningful contribution in the field.

SUSHANK GHIMIRE is a fourth year student, studying computer engineering under IOE, Thapathali Campus. He possesses a lot of interest, working with data. His educational path has provided him with a solid understanding of AI concepts, encompassing machine learning and data analysis. He possesses an unwavering curiosity and is constantly eager to delve into the latest advancements in AI. His objective is to leverage his knowledge and expertise to create a significant impact in the field.

APPENDIX

A. DECISION TREE USING ENTROPY

```
import matplotlib.pyplot as plt
import pandas as pd
from sklearn.datasets import load_breast_cancer
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn import tree
from sklearn.metrics import confusion_matrix
import seaborn as sns
from sklearn.metrics import classification_report

# Load the breast cancer dataset
data = load_breast_cancer()
# Get the feature names
feature_names = [x.capitalize() for x in data.feature_names]
# Get the class names
class_names = [x.capitalize() for x in data.target_names]

# Get the feature data (X)
X = data.data
target = data.target
# Convert feature data to a pandas DataFrame
data = pd.DataFrame(X)
target = pd.DataFrame(target)

#print first 5 data
data.head()

# Create the decision tree classifier object
classifier = DecisionTreeClassifier(criterion = 'entropy', random_state = 0)

#split data
x_train,x_test,y_train,y_test=train_test_split(data, target, test_size = 0.33, random_state

# Concatenate the features and target labels for training data
training_data = pd.concat([x_train,y_train],axis = 1)
# Concatenate the features and target labels for testing data
testing_data = pd.concat([x_test,y_test],axis = 1)

# Train the decision tree classifier
train=classifier.fit(x_train, y_train)
# Predict the target labels for the testing data using the trained classifier
Pred=classifier.predict(x_test)
print("prediction",Pred)

# Generate the confusion matrix
conf = confusion_matrix(y_test, Pred)
# Set the colormap for the heatmap
cmap = 'Blues'
# Create a heatmap of the confusion matrix
sns.heatmap(conf, annot=True,cmap=cmap, xticklabels=class_names, yticklabels=class_names,fr
plt.xlabel('Predicted')
plt.ylabel('Actual')
legend_handles = [
```

```
    plt.Line2D([], [], color='black', marker='o', markersize=10, label='THA076BCT029\nTHA0
]
plt.legend(handles=legend_handles, loc='upper left', bbox_to_anchor=(0.7, 1.1), ncol=len(le
plt.show

import numpy as np

target_names = ['malignant', 'benign']
report = classification_report(y_test, Pred, target_names=target_names)

# Format precision, recall, and f1-score values to two decimal places
report = report.replace('avg / total', 'avg/total')
report = report.replace('\n\n', '\n')

print("Classification Report:")
print(report)




fig, ax = plt.subplots(figsize=(20, 10))
# Plot the decision tree with custom styling
tree.plot_tree(train, ax=ax, filled=True, rounded=True, feature_names=feature_names, class_
# Customize the plot aesthetics
ax.set_xlabel("Features", fontsize=14)
ax.set_ylabel("Breast Cancer Class", fontsize=14)
# Adjust the arrow properties
for arrow in ax.get_xticklines():
  arrow.set_markersize(5)
# Adjust the spacing between subplots if needed
plt.subplots_adjust(wspace=0.5, hspace=0.5)
legend_handles = [
    plt.Line2D([], [], color='black', marker='o', markersize=10, label='THA076BCT029\nTHA0
]
plt.legend(handles=legend_handles, loc='upper left', bbox_to_anchor=(0, 0.9), ncol=len(lege
# Display the plot
plt.show()


#For max depth = 3
classifier = DecisionTreeClassifier(criterion = 'entropy', random_state = 0,max_depth = 3)
train=classifier.fit(x_train, y_train)
train=classifier.fit(x_train, y_train)
Pred=classifier.predict(x_test)
print("prediction",Pred)

conf = confusion_matrix(y_test, Pred)
cmap = 'Blues'
sns.heatmap(conf, annot=True,cmap=cmap,fmt='d',xticklabels=class_names, yticklabels=class_n
plt.xlabel('Predicted')
plt.ylabel('Actual')
legend_handles = [
    plt.Line2D([], [], color='black', marker='o', markersize=10, label='THA076BCT029\nTHA0
]
plt.legend(handles=legend_handles, loc='upper left', bbox_to_anchor=(0.8, 0), ncol=len(lege
plt.show
```

```
import numpy as np

target_names = ['malignant', 'benign']
report = classification_report(y_test, Pred, target_names=target_names)

# Format precision, recall, and f1-score values to two decimal places
report = report.replace('avg / total', 'avg/total')
report = report.replace('\n\n', '\n')

print("Classification Report:")
print(report)

fig, ax = plt.subplots(figsize=(14, 8))
# Plot the decision tree with custom styling
tree.plot_tree(train, ax=ax, filled=True, rounded=True, feature_names=feature_names, class_
# Customize the plot aesthetics
ax.set_xlabel("Features", fontsize=14)
ax.set_ylabel("Breast Cancer Class", fontsize=14)
# Adjust the arrow properties
for arrow in ax.get_xticklines():
  arrow.set_markersize(5)
# Adjust the spacing between subplots if needed
plt.subplots_adjust(wspace=0.5, hspace=0.5)
legend_handles = [
    plt.Line2D([], [], color='black', marker='o', markersize=10, label='THA076BCT029\nTHA0
]
plt.legend(handles=legend_handles, loc='upper left', bbox_to_anchor=(0, 0.9), ncol=len(lege
# Display the plot
plt.show()

#For max depth 2
classifier = DecisionTreeClassifier(criterion = 'entropy', random_state = 0,max_depth = 2)
train=classifier.fit(x_train, y_train)

train=classifier.fit(x_train, y_train)
Pred=classifier.predict(x_test)
print("prediction",Pred)

conf = confusion_matrix(y_test, Pred)
cmap = 'Blues'
sns.heatmap(conf, annot=True,cmap=cmap, fmt='d',xticklabels=class_names, yticklabels=class_
plt.xlabel('Predicted')
plt.ylabel('Actual')
legend_handles = [
    plt.Line2D([], [], color='black', marker='o', markersize=10, label='THA076BCT029\nTHA0
]
plt.legend(handles=legend_handles, loc='upper left', bbox_to_anchor=(0.8, 0), ncol=len(lege
plt.show

import numpy as np

target_names = ['malignant', 'benign']
report = classification_report(y_test, Pred, target_names=target_names)

# Format precision, recall, and f1-score values to two decimal places
```

```
report = report.replace('avg / total', 'avg/total')
report = report.replace('\n\n', '\n')

print("Classification Report:")
print(report)

fig, ax = plt.subplots(figsize=(7, 5))
# Plot the decision tree with custom styling
tree.plot_tree(train, ax=ax, filled=True, rounded=True, feature_names=feature_names, class_
# Customize the plot aesthetics
ax.set_xlabel("Features", fontsize=14)
ax.set_ylabel("Breast Cancer Class", fontsize=14)
# Adjust the arrow properties
for arrow in ax.get_xticklines():
  arrow.set_markersize(5)
# Adjust the spacing between subplots if needed
plt.subplots_adjust(wspace=0.5, hspace=0.5)
legend_handles = [
    plt.Line2D([], [], color='black', marker='o', markersize=10, label='THA076BCT029\nTHA0
]
plt.legend(handles=legend_handles, loc='upper left', bbox_to_anchor=(0, 0.9), ncol=len(lege
# Display the plot
plt.show()
```

B. DECISION TREE USING GINI INDEX

```
import matplotlib.pyplot as plt
import pandas as pd
from sklearn.datasets import load_breast_cancer
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn import tree
from sklearn.metrics import confusion_matrix
import seaborn as sns
from sklearn.metrics import classification_report

# Load the breast cancer dataset
data = load_breast_cancer()
# Get the feature names
feature_names = [x.capitalize() for x in data.feature_names]
# Get the class names
class_names = [x.capitalize() for x in data.target_names]


# Get the feature data (X)
X = data.data
target = data.target
# Convert feature data to a pandas DataFrame
data = pd.DataFrame(X)
target = pd.DataFrame(target)

#print first 5 data
data.head()

# Create the decision tree classifier object
```

```
classifier = DecisionTreeClassifier(criterion = 'gini', random_state = 0)

#split data
x_train,x_test,y_train,y_test=train_test_split(data, target, test_size = 0.33, random_state

# Concatenate the features and target labels for training data
training_data = pd.concat([x_train,y_train],axis = 1)
# Concatenate the features and target labels for testing data
testing_data = pd.concat([x_test,y_test],axis = 1)

# Train the decision tree classifier
train=classifier.fit(x_train, y_train)
# Predict the target labels for the testing data using the trained classifier
Pred=classifier.predict(x_test)
print("prediction",Pred)

# Generate the confusion matrix
conf = confusion_matrix(y_test, Pred)
# Set the colormap for the heatmap
cmap = 'Blues'
# Create a heatmap of the confusion matrix
sns.heatmap(conf, annot=True,cmap=cmap, xticklabels=class_names, yticklabels=class_names,fr
plt.xlabel('Predicted')
plt.ylabel('Actual')
legend_handles = [
    plt.Line2D([], [], color='black', marker='o', markersize=10, label='THA076BCT029\nTHA0
]
plt.legend(handles=legend_handles, loc='upper left', bbox_to_anchor=(0.7, 1.1), ncol=len(le
plt.show

import numpy as np

target_names = ['malignant', 'benign']
report = classification_report(y_test, Pred, target_names=target_names)

# Format precision, recall, and f1-score values to two decimal places
report = report.replace('avg / total', 'avg/total')
report = report.replace('\n\n', '\n')

print("Classification Report:")
print(report)


fig, ax = plt.subplots(figsize=(25, 10))
# Plot the decision tree with custom styling
tree.plot_tree(train, ax=ax, filled=True, rounded=True, feature_names=feature_names, class_
# Customize the plot aesthetics
ax.set_xlabel("Features", fontsize=14)
ax.set_ylabel("Breast Cancer Class", fontsize=14)
# Adjust the arrow properties
for arrow in ax.get_xticklines():
  arrow.set_markersize(5)
# Adjust the spacing between subplots if needed
plt.subplots_adjust(wspace=0.5, hspace=0.5)
legend_handles = [
    plt.Line2D([], [], color='black', marker='o', markersize=10, label='THA076BCT029\nTHA0
```

```
]
plt.legend(handles=legend_handles, loc='upper left', bbox_to_anchor=(0, 1), ncol=len(legend
# Display the plot
plt.show()


#For max depth = 3
classifier = DecisionTreeClassifier(criterion = 'gini', random_state = 0,max_depth = 3)
train=classifier.fit(x_train, y_train)
train=classifier.fit(x_train, y_train)
Pred=classifier.predict(x_test)
print("prediction",Pred)

conf = confusion_matrix(y_test, Pred)
cmap = 'Blues'
sns.heatmap(conf, annot=True,cmap=cmap,fmt='d',xticklabels=class_names, yticklabels=class_n
plt.xlabel('Predicted')
plt.ylabel('Actual')
legend_handles = [
    plt.Line2D([], [], color='black', marker='o', markersize=10, label='THA076BCT029\nTHA07
]
plt.legend(handles=legend_handles, loc='upper left', bbox_to_anchor=(0.8, 0), ncol=len(lege
plt.show

import numpy as np

target_names = ['malignant', 'benign']
report = classification_report(y_test, Pred, target_names=target_names)

# Format precision, recall, and f1-score values to two decimal places
report = report.replace('avg / total', 'avg/total')
report = report.replace('\n\n', '\n')

print("Classification Report:")
print(report)

fig, ax = plt.subplots(figsize=(14, 8))
# Plot the decision tree with custom styling
tree.plot_tree(train, ax=ax, filled=True, rounded=True, feature_names=feature_names, class_
# Customize the plot aesthetics
ax.set_xlabel("Features", fontsize=14)
ax.set_ylabel("Breast Cancer Class", fontsize=14)
# Adjust the arrow properties
for arrow in ax.get_xticklines():
  arrow.set_markersize(5)
# Adjust the spacing between subplots if needed
plt.subplots_adjust(wspace=0.5, hspace=0.5)
legend_handles = [
    plt.Line2D([], [], color='black', marker='o', markersize=10, label='THA076BCT029\nTHA07
]
plt.legend(handles=legend_handles, loc='upper left', bbox_to_anchor=(0, 0.9), ncol=len(lege
# Display the plot
plt.show()


#max depth 2
```

```
classifier = DecisionTreeClassifier(criterion = 'gini', random_state = 0,max_depth = 2)
train=classifier.fit(x_train, y_train)

train=classifier.fit(x_train, y_train)
Pred=classifier.predict(x_test)
print("prediction",Pred)

conf = confusion_matrix(y_test, Pred)
cmap = 'Blues'
sns.heatmap(conf, annot=True,cmap=cmap, fmt='d',xticklabels=class_names, yticklabels=class_
plt.xlabel('Predicted')
plt.ylabel('Actual')
legend_handles = [
    plt.Line2D([], [], color='black', marker='o', markersize=10, label='THA076BCT029\nTHA0
]
plt.legend(handles=legend_handles, loc='upper left', bbox_to_anchor=(0.8, 0), ncol=len(lege
plt.show

import numpy as np

target_names = ['malignant', 'benign']
report = classification_report(y_test, Pred, target_names=target_names)

# Format precision, recall, and f1-score values to two decimal places
report = report.replace('avg / total', 'avg/total')
report = report.replace('\n\n', '\n')

print("Classification Report:")
print(report)

fig, ax = plt.subplots(figsize=(7, 5))
# Plot the decision tree with custom styling
tree.plot_tree(train, ax=ax, filled=True, rounded=True, feature_names=feature_names, class_
# Customize the plot aesthetics
ax.set_xlabel("Features", fontsize=14)
ax.set_ylabel("Breast Cancer Class", fontsize=14)
# Adjust the arrow properties
for arrow in ax.get_xticklines():
  arrow.set_markersize(5)
# Adjust the spacing between subplots if needed
plt.subplots_adjust(wspace=0.5, hspace=0.5)
legend_handles = [
    plt.Line2D([], [], color='black', marker='o', markersize=10, label='THA076BCT029\nTHA0
]
plt.legend(handles=legend_handles, loc='upper left', bbox_to_anchor=(0, 0.9), ncol=len(lege
# Display the plot
plt.show()
```