VIETNAM NATIONAL UNIVERSITY HO CHI MINH CITY
UNIVERSITY OF SCIENCE
FACULTY OF ADVANCED INFORMATION TECHNOLOGY

LAB PRACTICE #2: DECISION TREE

INTRODUCTION TO ARTIFICIAL INTELLIGENCE – CSC14003

—o0o—

INSTRUCTOR(S)

MS. NGUYỄN NGỌC THẢO
MR. NGUYỄN THANH TÌNH
MS. HỒ THỊ THANH TUYẾN

STUDENT'S INFORMATION

FULL NAME: LÊ PHƯỚC PHÁT
CLASS: 22CLC10
STUDENT ID: 22127322
STUDENT'S EMAIL: lpphat22@clc.fitus.edu.vn

HO CHI MINH CITY, JULY 2024

TABLE OF CONTENTS

TEACHERS' COMMENTS

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

Date: **August …, 2024**

Graded Teacher(s)

PLEDGE

I declare that this research was conducted by me, under the supervision and guidance of the teachers of the **Introduction to Artificial Intelligence – CSC14003** subject: Ms. **Nguyen Ngoc Thao**, Mr. **Nguyen Thanh Tinh**, and Ms. **Ho Thi Thanh Tuyen**. The results of this study are legal and have not been published in any form before. All documents used in this study were collected by myself and from various sources, and are fully listed in the references section. In addition, we also use the results of several other authors and organizations. All are properly cited. In case of copyright infringement, we are responsible for such action. Therefore, **Ho Chi Minh City University of Science (HCMUS)** is not responsible for any copyright violations committed in my research.

RESEARCH PROJECT

I. Self-evaluating the completion rate of the lab and other requirements

No.	Specifications	Completion Level
1	**Preparing the data sets numpy**	
	• Preparing subsets	100 %
	• Visualizing distributions	100 %
2	**Building the decision tree classifiers**	100 %
3	**Evaluating the decision tree classifiers**	
	• Classification report and confusion matrix	100 %
	• Comments	100 %
4	**The depth and accuracy of a decision tree**	
	• Trees, tables, and charts	100 %
	• Comments	100 %

II. Preparing the data sets

a. Preparing subsets

Step One: We will begin by installing all the necessary libraries for the lab. These libraries include:

- numpy: Used for processing and manipulating data in array format, preparing input data for machine learning algorithms.
- pandas: Provides data structures and tools for handling tabular data, such as DataFrames. In this project, Pandas will be used to load, process, and analyze data. For example, you can use Pandas to read data from CSV files and perform operations like filtering, grouping, or aggregating data.
- scikit-learn: Offers simple and efficient tools for data mining and data analysis. In this project, Scikit-learn will be used to build, train, and evaluate machine learning models, including Decision Trees. It also supports tools for data splitting, normalization, and model performance evaluation.
- graphviz: Helps visualize Decision Tree models, allowing you to see the structure of the tree and the decisions the model is making.
- joblib: Used for saving trained Decision Tree models so they can be loaded and used later without the need to retrain them from scratch.
- matplotlib: Used for creating basic plots, such as class distribution charts, classification result visualizations, or performance metrics graphs.
- seaborn: Provides more complex and aesthetically pleasing plots, such as distribution charts, heatmaps, and pair plots, making it easier to explore and present data.
- plotly: Used for creating interactive plots, enhancing the visualization and analysis of classification results in a more dynamic and engaging manner.

Step Two: We will collect and analyze the data. As mentioned earlier, the dataset used for this research is the Breast Cancer Wisconsin (Diagnostic) dataset, publicly available through the UCI Machine Learning Repository. This dataset is used to classify tumors as benign (B) or malignant (M) based on 30 numerical features extracted from medical images. The dataset consists of 569 samples with labels assigned as M (Malignant) or B (Benign). We will load the dataset using the **load_data()** function to prepare for splitting into training and testing sets. To ensure that the training and testing datasets accurately reflect the characteristics of the original dataset, the feature and label datasets will be shuffled in the same order using the **shuffle()** function to maintain the correlation between the feature data and its labels.

Step Three: After shuffling the features and labels, we will split the dataset into training and testing sets according to the given ratios. Stratification ensures that the proportion of labels (malignant or benign) is preserved in both the training and testing sets. The different train/test ratios are 40/60, 60/40, 80/20, and 90/10. Once the dataset is split into training and testing sets using the train_test_split() function with parameters **"stratify"** to ensure that the label distribution is preserved, and **"random_state"** set to 42 to ensure consistency across runs, we will store these datasets in a list called datasets. Each entry in the list will be a tuple containing:

- **feature_train**: The set of training samples (excluding target attributes).
- **feature_test**: The set of labels corresponding to the samples in feature_train.
- **label_train**: The set of test samples, similar in structure to feature_train.
- **label_test**: The set of labels corresponding to the samples in feature_test.

Thus, after completing the data-splitting process, we will have 4 subsets for each ratio, resulting in a total of 16 datasets.

b. Visualizing the distributions of classes in all the data sets

After we have divided and prepared the data set at each scale, we will visualize the distribution of data among different classes in the dataset, with a specific emphasis on the training and test sets. It employs bar charts to compare the counts of each class in the original data, training data, and test data, considering different proportions of the split. This visualization enables a comparative analysis of class distributions and provides insights into the impact of the split operation on the dataset.

Figure 1. The data distribution 40/60

Figure 2. The data distribution 60/40

Figure 3. The data distribution 80/20

Figure 4. The data distribution 90/10

III. Building the decision tree classifiers with different ratios

a. Training model with different ratios

In this part, we will build and train decision tree classifiers for different train/test split ratios. To solve this problem, we will use build_decision_tree_classifiers() function.

The build_decision_tree_classifiers function is designed to streamline the process of creating and training decision tree classifiers on multiple datasets with different train/test split ratios. It takes as input a list of datasets, where each dataset is already split into training and testing sets, along with the corresponding split ratios. The function iterates through these datasets and their associated ratios, training a decision tree model for each one using entropy as the criterion for splitting. To ensure consistency and reproducibility, a fixed random state is applied to each model. After training, each model is saved to disk with a filename that reflects the specific train/test split ratio used, allowing for easy identification. The function then returns a list of these trained models, making them readily available for further analysis or deployment.

b. Visualizing decision tree with different split ratios

Afterward, each decision tree model is visualized using the **graphviz** library, offering a clear understanding of the decision-making process the model has learned. The visualization showcases elements like node splitting criteria, class labels, and decision paths, which improves interpretability and supports model analysis. These visualizations are saved as PNG files for each training and testing data ratio, making it easier to compare and analyze across different dataset sizes. This systematic approach to visualizing models allows researchers and practitioners to gain valuable insights into the decision-making behavior of the trained models and evaluate their effectiveness in managing the given dataset.

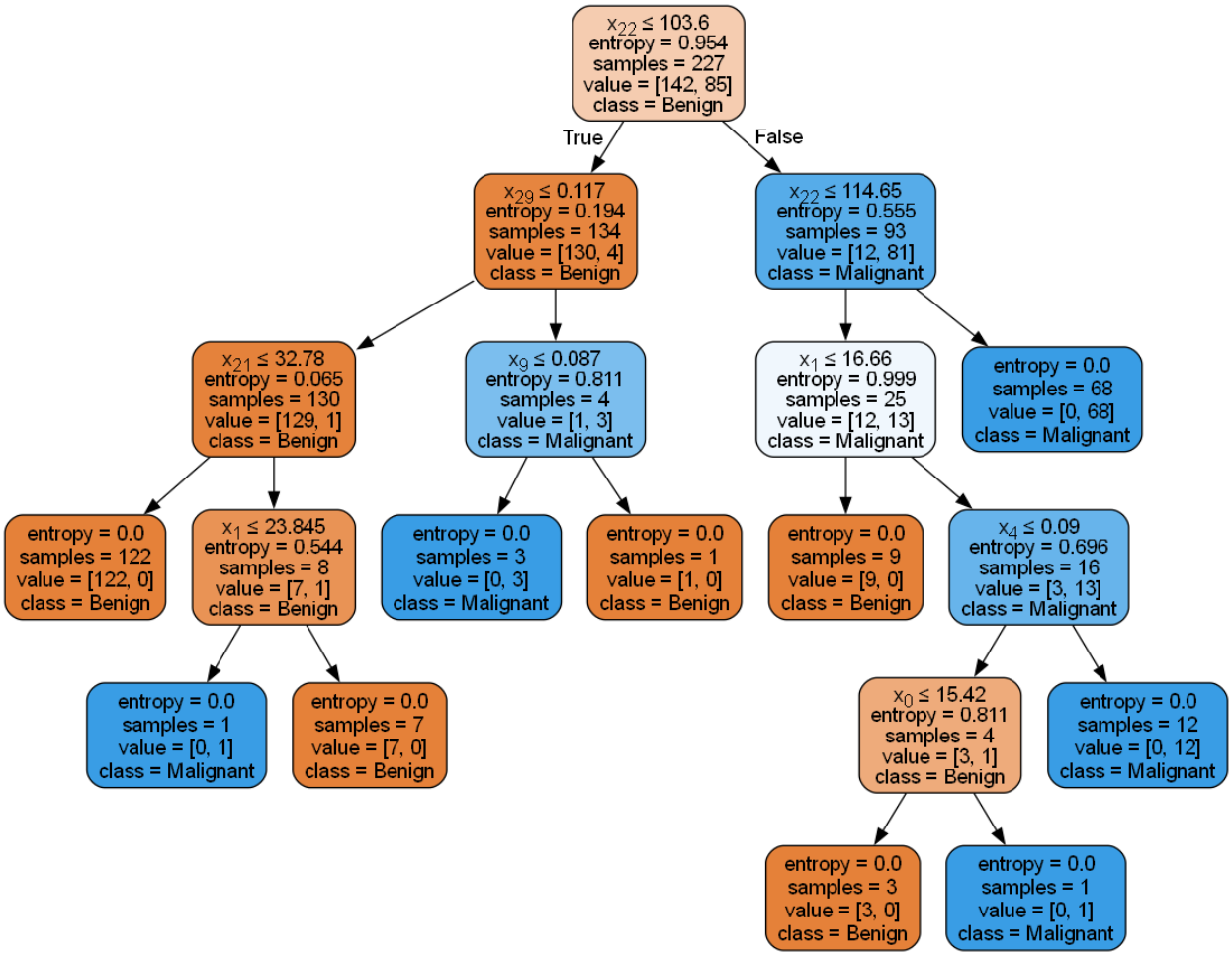


Figure 5. Decision tree classifier with proportion 40/60

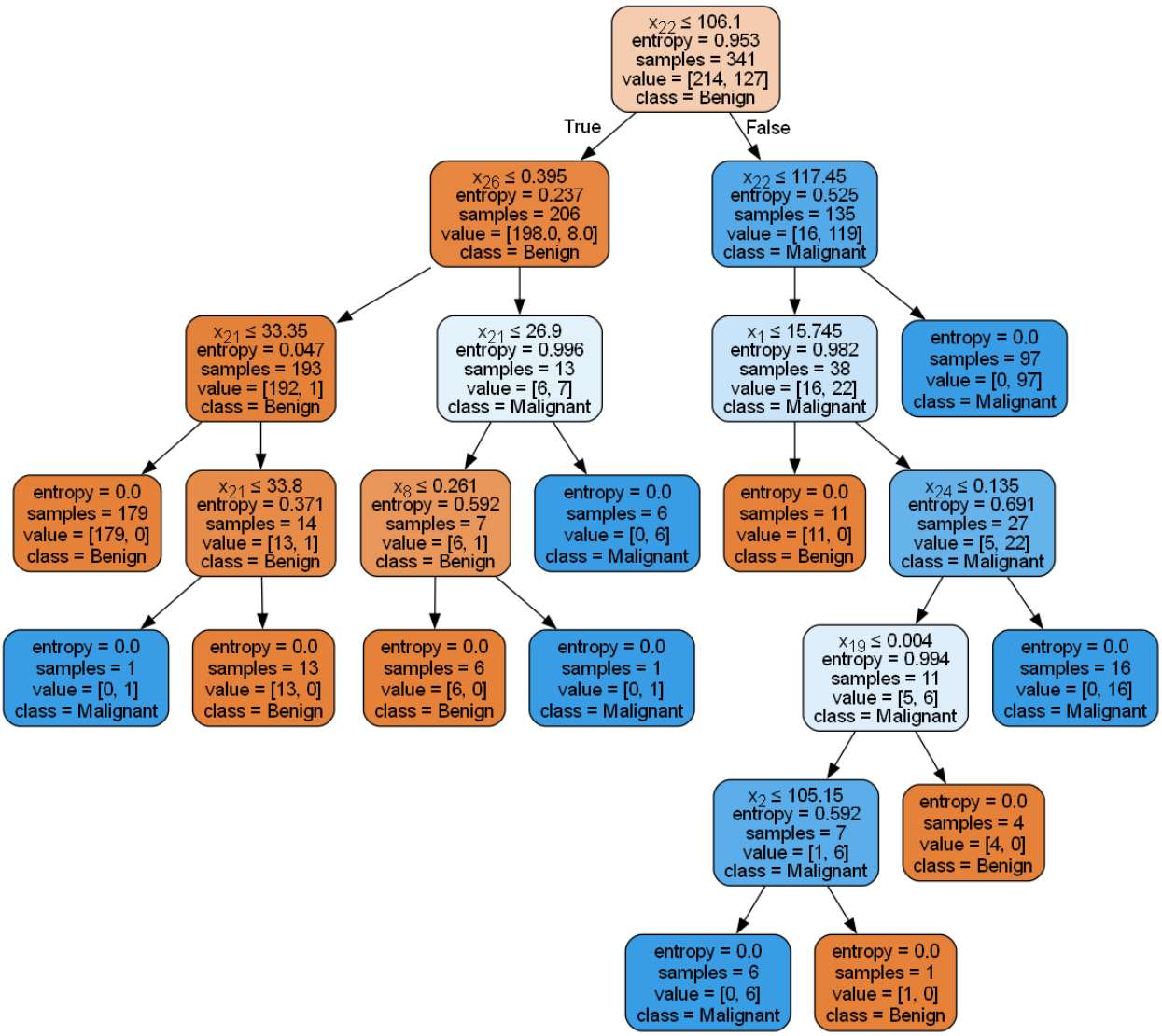


Figure 6. Decision tree classifier with proportion 60/40

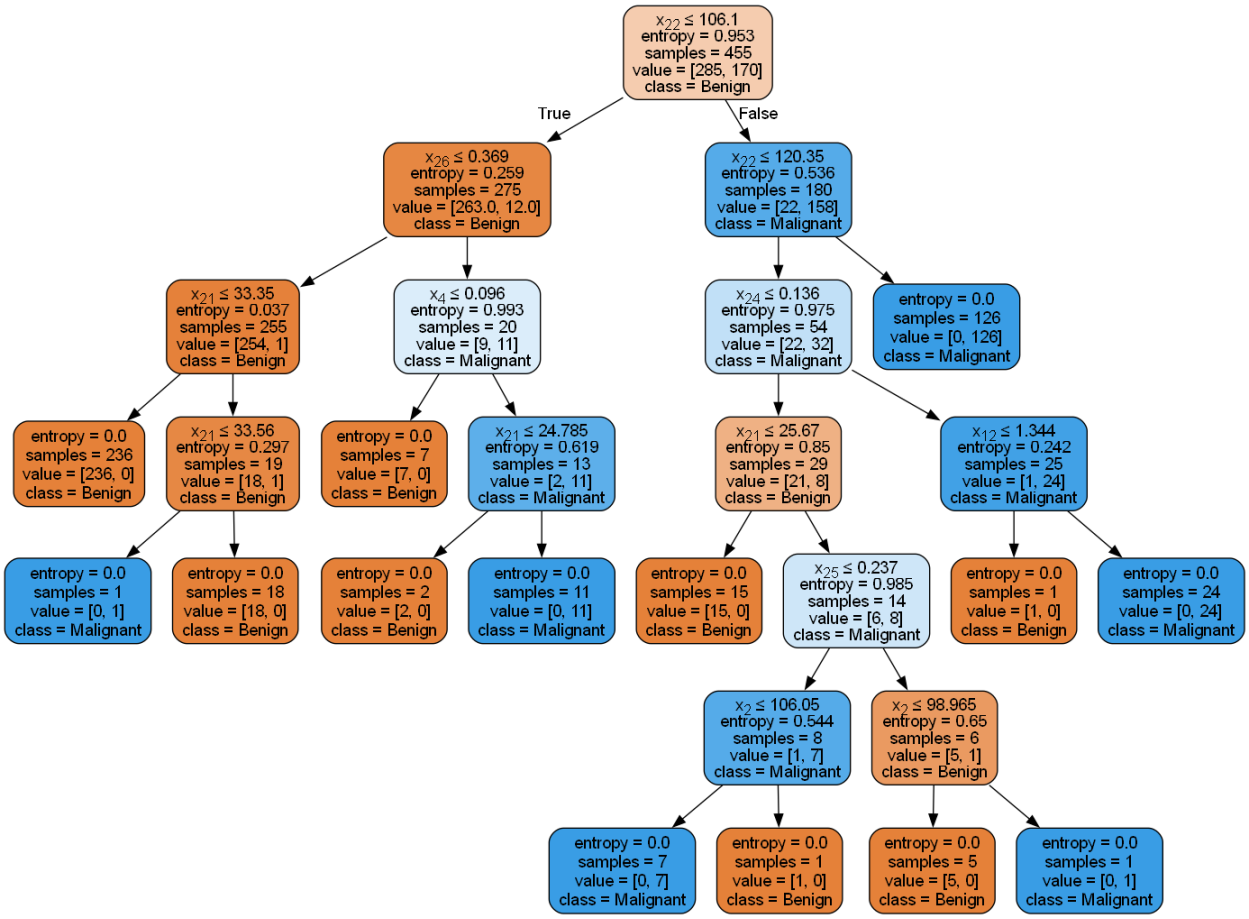


Figure 7. Decision tree classifier with proportion 80/20

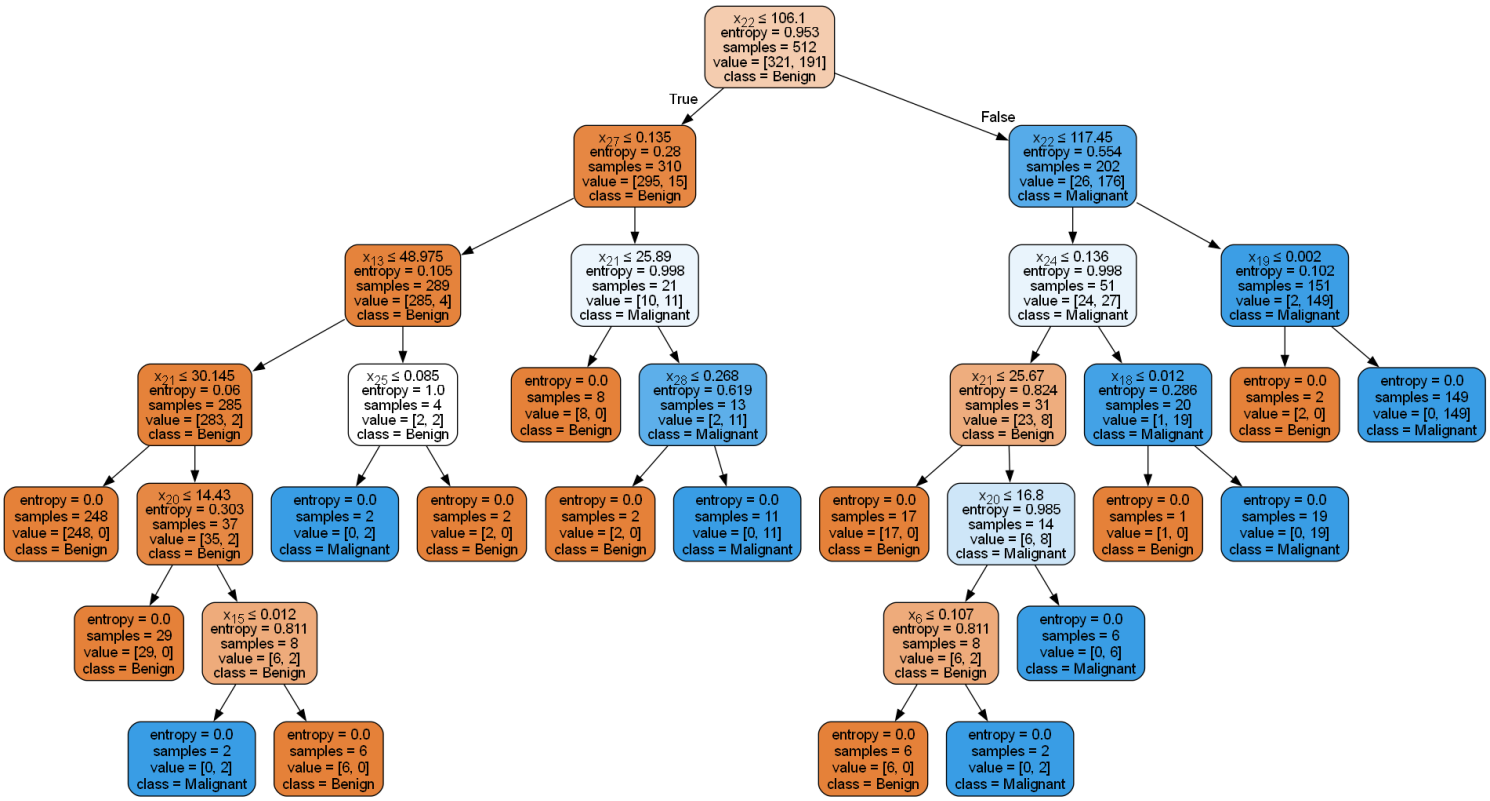


Figure 8. Decision tree classifier with proportion 90/10

IV. Evaluating the decision tree classifiers

a. Classification report and confusion matrix

To interpret the classification report and confusion matrix from **sklearn.metrics**, we will talk about the definition of those because the classification report and confusion matrix play crucial roles in assessing the effectiveness of a classification model.

The **classification report** provides several key metrics to evaluate the performance of a classification model. It also provides a concise overview of these performance metrics for each class within the dataset.

- **Precision:** measures how many of the instances predicted as a certain class are that class. A high precision indicates that the classifier is not mislabeling in many instances.
  (Precision = how many selected items are relevant = how useful the search results are.)

$$Precision = \frac{\sum True\ Positive\ (TP)}{\sum True\ Positive\ (TP) + False\ Positive\ (FP)}$$

- **Recall:** measures how many of the actual instances of a class the model correctly identified. A high recall indicates that the classifier is capturing most of the true instances. *(Recall = how many relevant items are selected = how sensitive the results are.)*

$$\boldsymbol{Recall} = \frac{\sum \boldsymbol{True\ Positive}\ (\boldsymbol{TP})}{\sum \boldsymbol{True\ Positive}\ (\boldsymbol{TP}) + \boldsymbol{False\ Negative}\ (\boldsymbol{FN})}$$

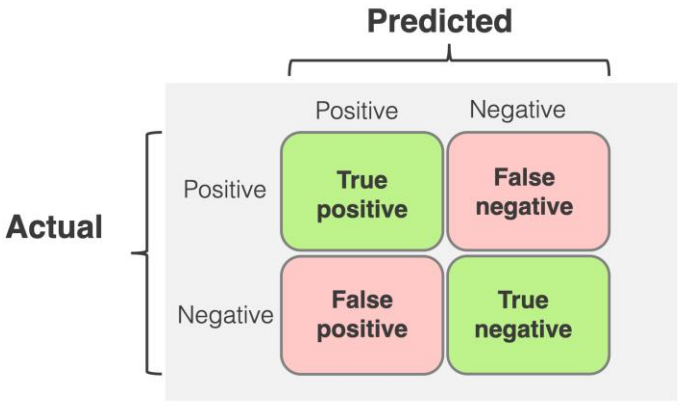
- **F1 – Score:** is the harmonic mean of precision and recall, providing a single metric that balances the two.

$$F_1 - Score = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

- **Support**: indicates the number of true instances for each class in the dataset.

On the other hand, the **confusion matrix** is a tabular representation that summarizes the model's predictions on a test dataset, highlighting the number of correct and incorrect predictions. The confusion matrix provides a detailed breakdown of how the classifier is performing in terms of true positives, true negatives, false positives, and false negatives:

- **True Positives (TP):** Instances correctly classified as the positive class.
- **True Negatives (TN):** Instances correctly classified as the negative class.
- **False Positives (FP):** Instances incorrectly classified as the positive class (Type I error).
- **False Negatives (FN):** Instances incorrectly classified as the negative class (Type II error).



Integrating the insights from both the classification report and the confusion matrix provides a complete evaluation of the classifier's performance. The classification report delivers an overall summary of key performance metrics, while the confusion matrix gives a detailed perspective on individual prediction outcomes. When used together, these tools offer a deep analysis of the classifier's strengths and weaknesses, supporting informed decision-making and model improvement efforts.

By analyzing the **confusion matrix**, we can calculate performance metrics in the **classification report**. We will analyze the confusion matrix of decision tree classifiers with **proportion 40/60** below, which is one of the models used for experiments.

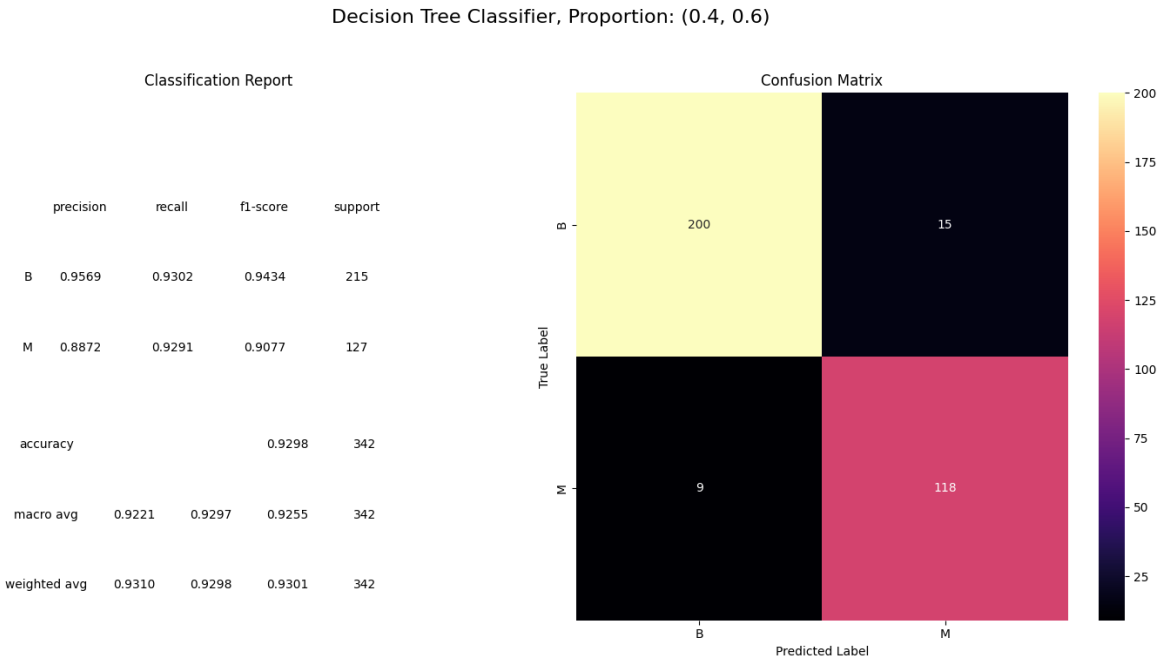


Figure 9. The classification report and confusion matrix with the dataset 40/60

- $Accuracy = \frac{\sum TP + TN}{\sum TP + FP + FN + TN} = \frac{200+118}{200+118+15+9} \approx 0.9298$
- For class "Benign" (B), we have:
  - $Precision = \frac{\sum TP}{\sum TP + FP} = \frac{200}{200+9} \approx 0.9569$
  - $Recall = \frac{\sum TP}{\sum TP + FN} = \frac{200}{200+15} \approx 0.9302$
  - $F_1 - Score = 2 \times \frac{0.9569 \times 0.9302}{0.9569+0.9302} \approx 0.9434$
- For class "Malignant" (M), we have:
  - $Precision = \frac{\sum TP}{\sum TP + FP} = \frac{118}{118+15} \approx 0.8872$
  - $Recall = \frac{\sum TP}{\sum TP + FN} = \frac{118}{118+9} \approx 0.9291$

- $F_1 - Score = 2 \times \frac{0.8872 \times 0.9291}{0.8872 + 0.9291} \approx 0.9077$

Below are **classification reports** and **confusion matrices** of the **decision tree classifiers** corresponding remaining proportions (**confusion matrices** are visualized by using *heatmap* from **seaborn** library).

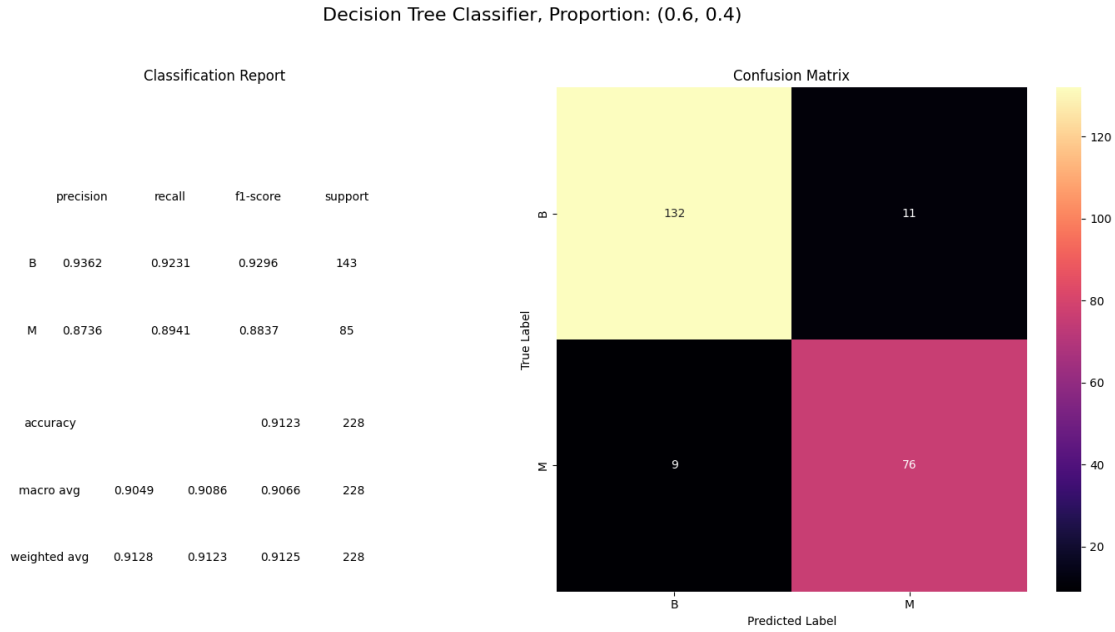


Figure 10. The classification report and confusion matrix with the dataset 60/40

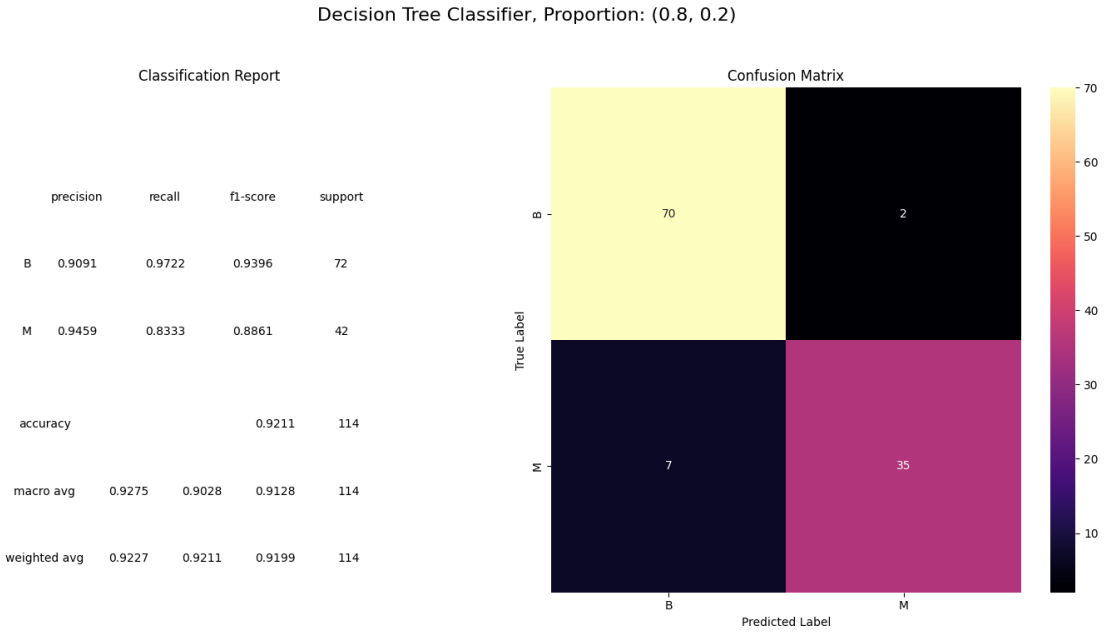


Figure 11. The classification report and confusion matrix with the dataset 80/20

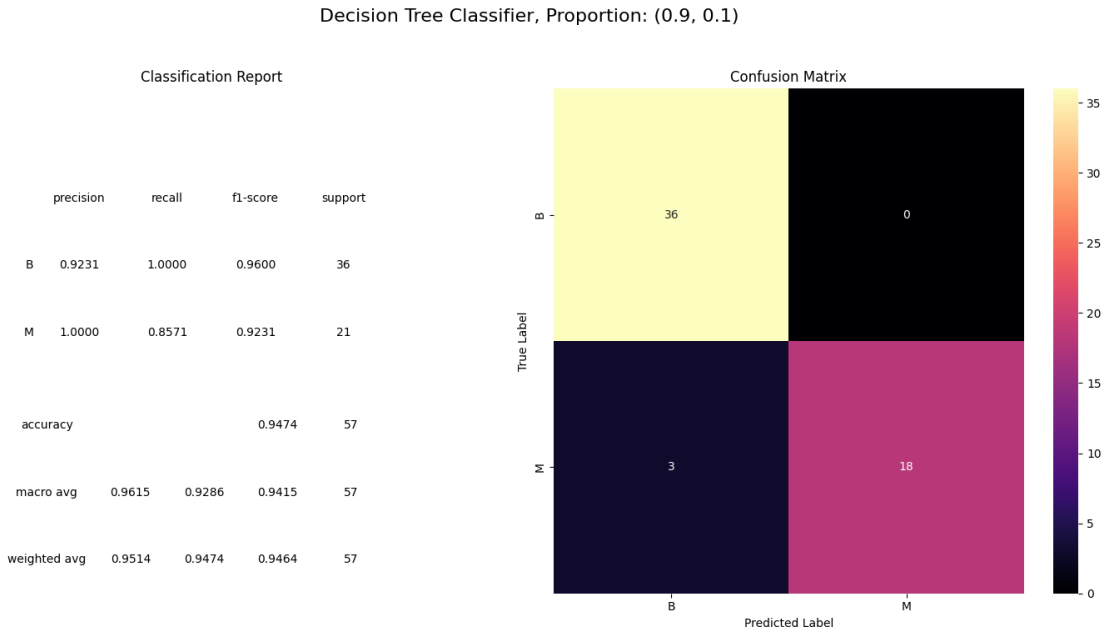


Figure 12. The classification report and confusion matrix with the dataset 90/10

b. Comments

- **Decision Tree Classifiers with proportion 40/60**
  - **Accuracy**
    - The model achieves a high accuracy of 92.98%, indicating that the model performs well in classifying instances across all classes, achieving a high percentage of correct labels. This is a strong performance, but the model should still be reviewed for any potential improvements, especially in cases where precision or recall might be slightly lower.
  - **Precision, recall, and F1-score of each class**
    - **Class "Benign" (B):** this class achieves a high precision score, indicating that the model has few false positives for "B" labels. The model correctly identified 93% of the actual benign cases. This indicates a strong ability to capture the most benign cases, though it missed about 7%. The F1-Score is 94%, which is a good balance between precision and recall, indicating strong performance in predicting benign cases.

- **Class "Malignant" (M):** the precision and recall for this class are lower than the class "Benign" but are still high, with a slight difference indicating some misclassifications. The F1-Score of 90.77% is a good balance between precision and recall, reflecting solid performance in predicting malignant cases.

- **Decision Tree Classifiers with proportion 60/40**

  - **Accuracy**

    - The model achieves a high accuracy of 91.23%, indicating that the model performs well in classifying instances across all classes, achieving a high percentage of correct labels. This is a strong performance, but the model should still be reviewed for any potential improvements, especially in cases where precision or recall might be slightly lower.

  - **Precision, recall, and F1-score of each class**

    - **Class "Benign" (B):** this class has a very high precision score, meaning that 93.62% of the cases predicted as benign are benign. Besides that, the ability to correctly identify benign cases is approximately 92.31%. Finally, the f1 score of this class is a balance between precision and recall, showing strong performance in predicting benign cases.

    - **Class "Malignant" (M):** the precision score is slightly lower than benign, but still good with 87.36% of the cases predicted as malignant being correct. The recall is very high, indicating that the model correctly identifies 94.12% of malignant cases. The F1 score is strong enough to suggest that the model doesn't miss too many malignant cases, with a relatively low number of false negatives and false positives.

- **Decision Tree Classifiers with proportion 80/20**
  - **Accuracy**
    - The overall accuracy of 92.11% indicates that the decision tree classifier performs well in classifying instances across all classes, achieving a high proportion of correct predictions.
  - **Precision, recall, and F1-score of each class**
    - **Class "Benign" (B):** this model can predict 90.91% of the benign cases, which are actual labels. The ability to correctly identify benign cases is approximately 97.22%. This class is a balance between precision and recall, showing strong performance in predicting benign cases.
    - **Class "Malignant" (M):** Precision is slightly higher than benign, which means that it is good with 94.59% of the cases predicted as malignant being correct. The recall is very low, indicating that the model correctly identifies 83.33% of malignant cases. The F1 score is strong enough to suggest that the model doesn't miss too many malignant cases, with a relatively low number of false negatives and false positives.
- **Decision Tree Classifiers with proportion 90/10**
  - **Accuracy**
    - The overall accuracy of 94.74% indicates that the decision tree classifier performs well in classifying instances across all classes, achieving a high proportion of correct predictions.
  - **Precision, recall, and F1-score of each class**
    - **Class "Benign" (B):** the precision score and recall is very high. The F1 Score is a good balance between precision and recall, reflecting solid performance in predicting malignant cases.
    - **Class "Malignant" (M):** Precision is good with 100% of the cases predicted as malignant being correct. The recall is

very high, indicating that the model correctly identifies 85.71% of malignant cases. The F1-Score is 92.31%, which is a good balance between precision and recall, reflecting solid performance in predicting malignant cases.

In conclusion, among four proportions corresponding to four separated decision tree classifiers, the most appropriate model with proportion train-test split is the model with 90/10, because the accuracy of the decision tree with this proportion is quite high (94.72%), besides, the large size of the training set helps the model learn more about the characteristics of datasets, and make it more valuable.

V. The depth and accuracy of a decision tree

a. Trees, tables, and charts

- **Trees**

After testing and measuring the classification accuracies corresponding to different decision trees' depths, which are 2, 3, 4, 5, 6, 7, and "no limit", below will be images of those trees.

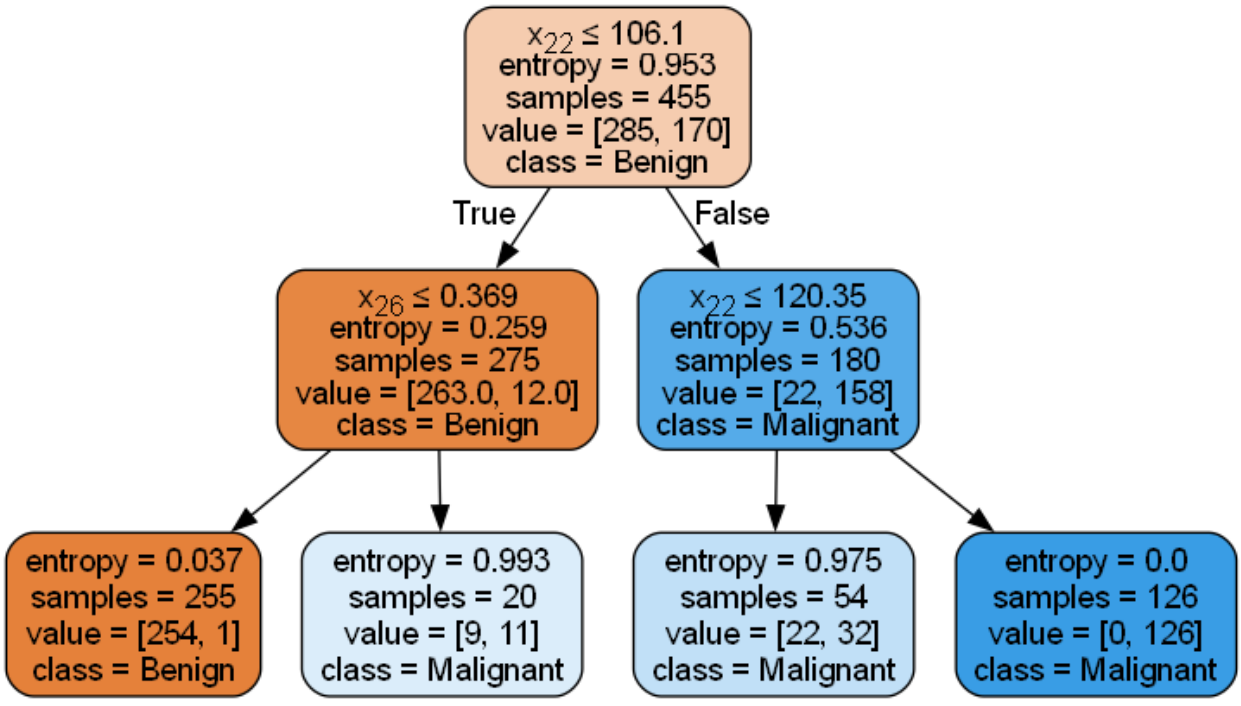


Figure 13. Decision tree classification with max depth 2

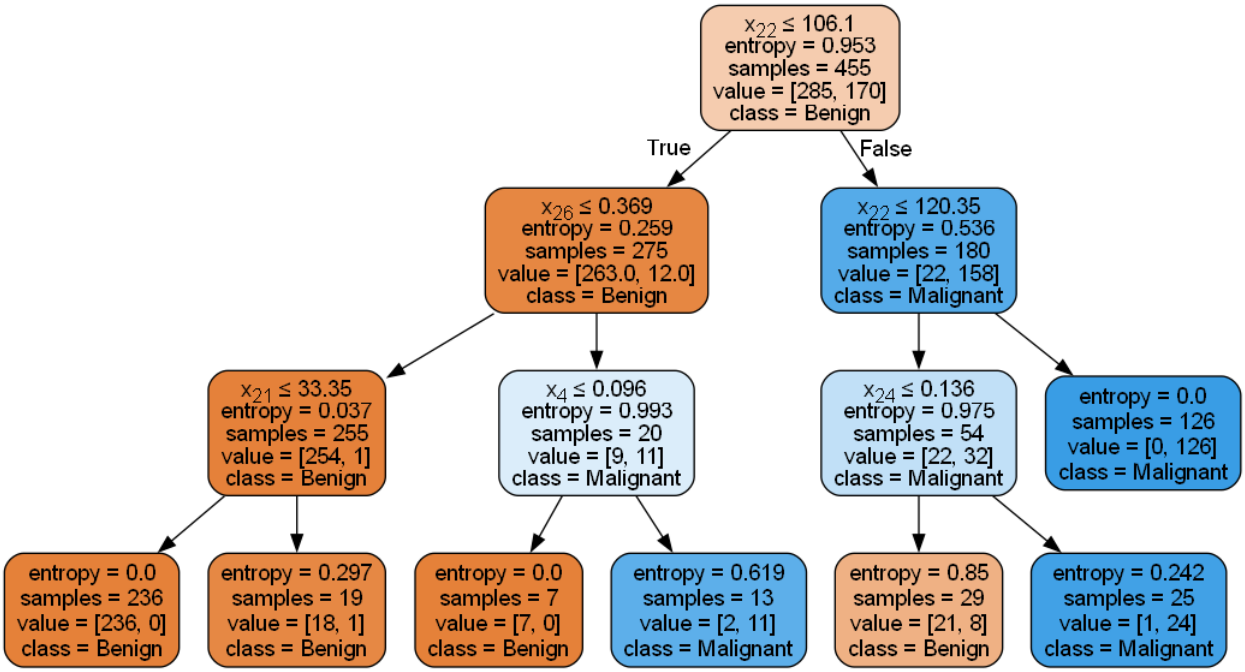


Figure 14. Decision tree classification with max depth 3

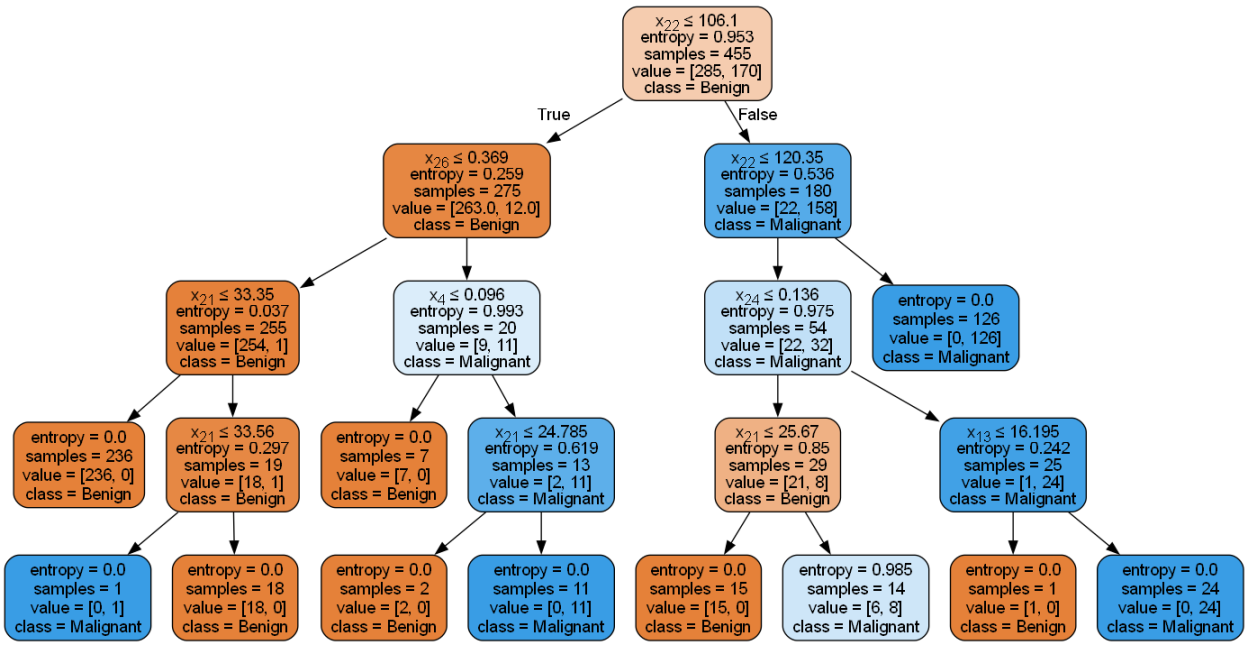


Figure 15. Decision tree classification with max depth 4

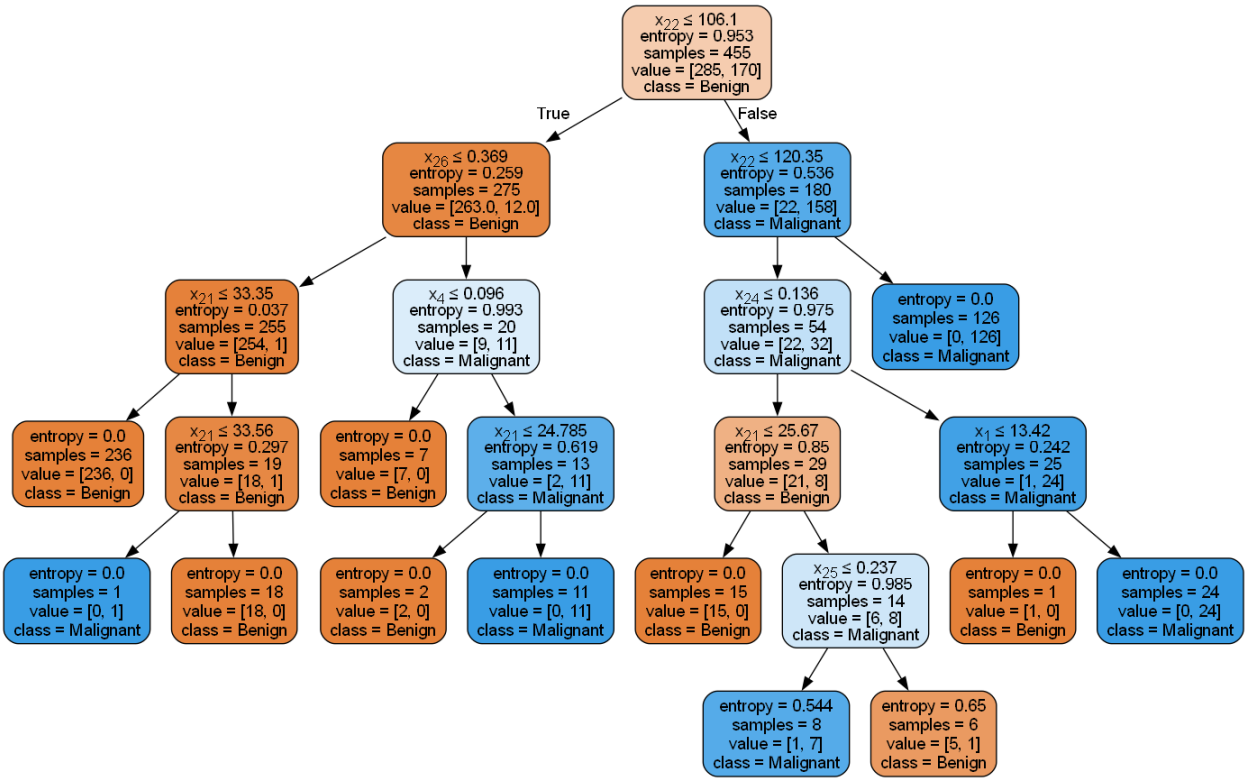


Figure 16. Decision tree classification with max depth 5

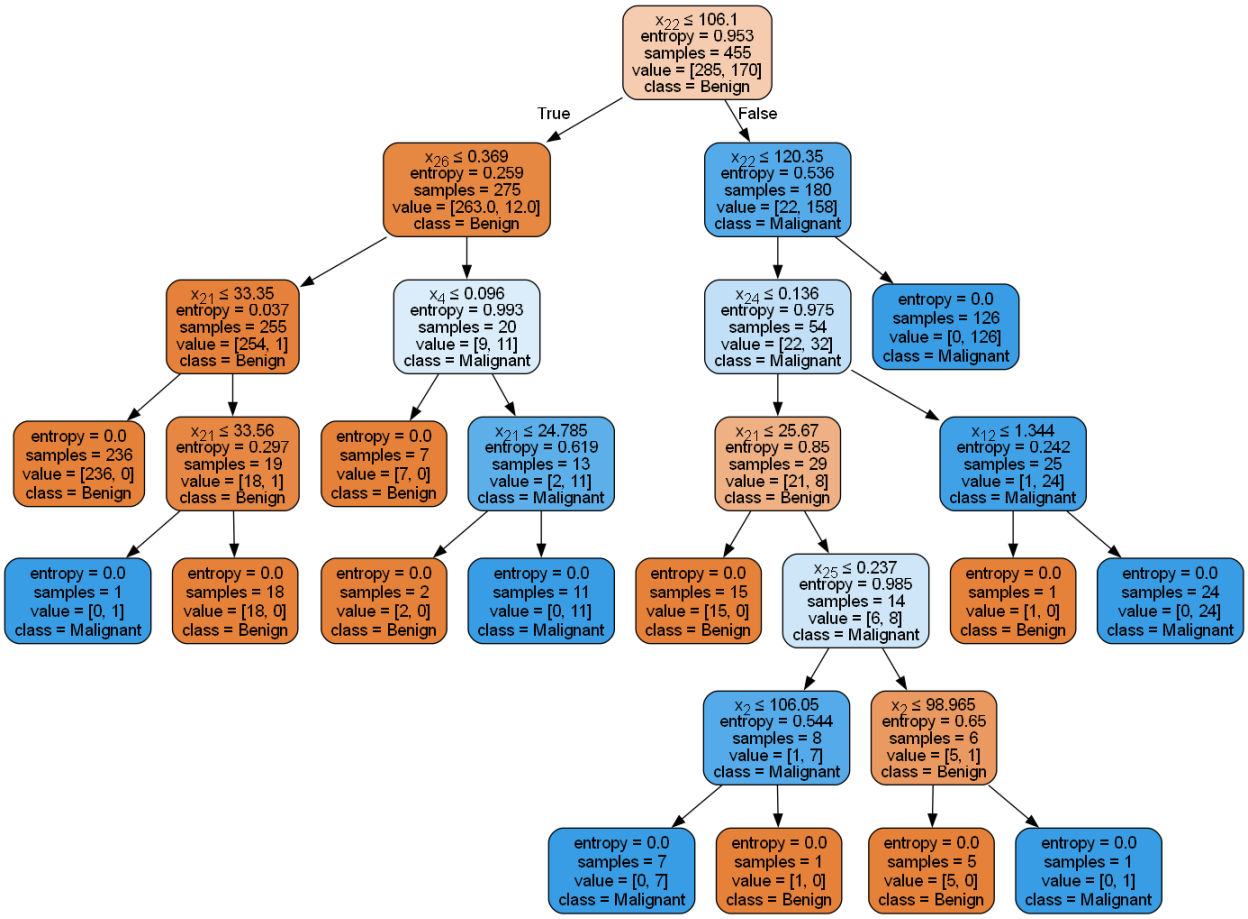


Figure 17. Decision tree classification with max depth 6

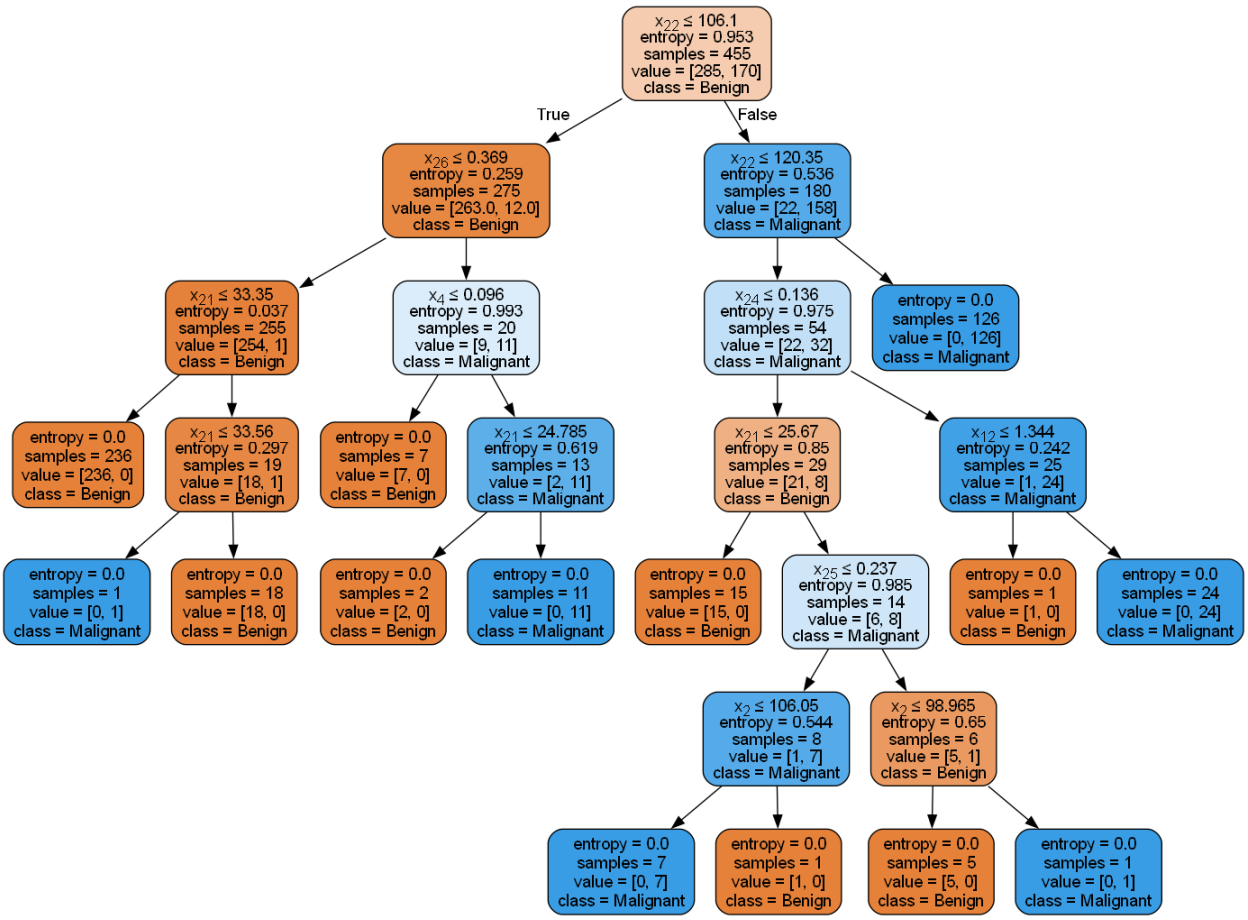
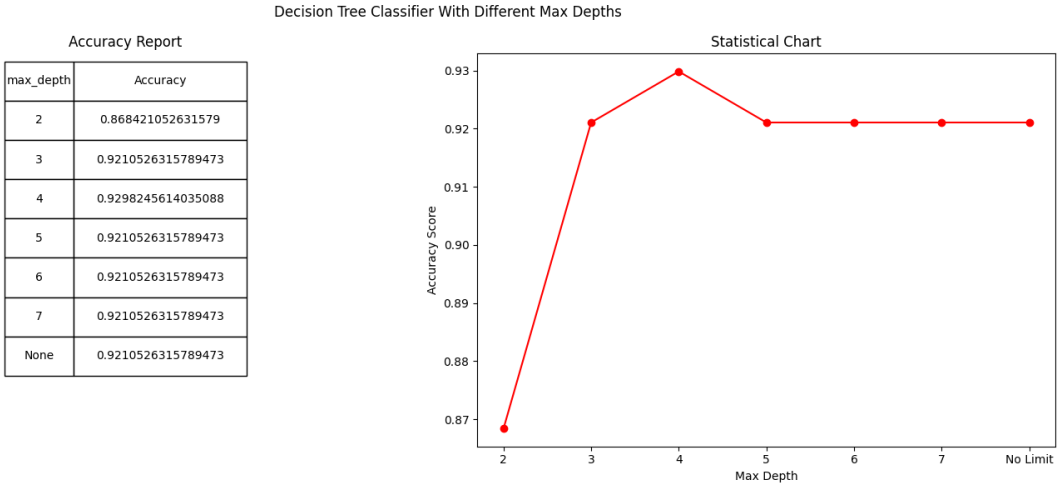


Figure 18. Decision tree classification with max depth 7

Figure 19. Decision tree classification with max depth "None"

- **Tables and charts**

  Below is the statistical table and chart that describe the accuracy of the decision tree with proportions (0.8, 0.2) with different depths



b. Comments

- Based on the statistical data and line charts analyzing the accuracy of decision trees at various depths, it is clear that the depth of a decision tree plays a critical role in its performance:
  - **Model with max depth 2:** The accuracy is the lowest at 86.84%, indicating the model is too simplistic.
  - **Model with max depth 3:** The accuracy significantly improves to 92.11%.
  - **Model with max depth 4:** The accuracy peaks at 92.98%, which is the highest among all models.
  - **Model with max depth 5 to No Limits (None):** The accuracy slightly decreases and stabilizes at 92.11%.
- **Observations:**
  - Initially, as the max depth increases from 2 to 4, the accuracy shows substantial improvement, suggesting that the model benefits from the ability to create more complex decision boundaries.
  - The optimal accuracy is achieved at a max depth of 4, after which further increases in depth result in a slight decrease and stabilization in accuracy, indicating that the model does not gain additional predictive power from increased complexity. In fact, it might lead to overfitting, where the model becomes too tailored to the training data and less effective on new data.
- **Interpretation:**
  - **Increase in Accuracy with Depth:**
    - At max depth 2, the model likely underfits the data, failing to capture its complexity, resulting in the lowest accuracy of 86.84%.
    - The jump in accuracy at max depth 3 to 92.11% reflects the model's ability to capture more meaningful patterns.
  - **Peak Accuracy:**
    - Max depth 4 represents the sweet spot where the model strikes a balance between complexity and simplicity, achieving the highest accuracy of 92.98%.
  - **Stability and Slight Decrease in Accuracy:**
    - Beyond a max depth of 4, the accuracy does not improve but stabilizes at 92.11%, with a slight drop. This suggests that further increasing depth leads to overfitting, where the model becomes overly complex and loses its ability to generalize well to unseen data.

REFERENCES

During the process of researching and implementing the project **Lab#2: Decision Tree – CSC14003 – Introduction to Artificial Intelligence**, I used and referenced some of the following open and electronic documents:

Programming

[1] Confusion matrix – sklearn *(last updated: 25/08/2024)*

[2] Tìm hiểu về Confusion matrix trong Machine Learning? *(last updated: /08/2024)*

[3] All About Train Test Split *(last updated: 25/08/2024)*

Report

[4] How to interpret a confusion matrix for a machine learning model – evidently AI *(last updated: 25/08/2024)*

[5] DecisionTreeClassifier - sklearn *(last updated: 25/08/2024)*

[6] Lab2 – Decision Tree Report – Huynh Duc Thien *(last updated: 25/08/2024)*