VNU-HCM UNIVERSITY OF SCIENCE

FACULTY OF INFORMATION TECHNOLOGY

FUNDAMENTALS OF ARTIFICIAL INTELLIGENCE

PROJECT 2 - DECISION TREE

INSTRUCTORS:
Dr Nguyễn Tiến Huy
Mr Nguyễn Trần Duy Minh
Mr Nguyễn Thanh Tình

MEMBERS:
Lê Nguyên Thảo - 23127118
Đỗ Đăng Nhật Tiến - 23127269
Nguyễn Thanh Owen - 23127447
Trần Minh Quang - 23127464

Table of contents

1 Implementing Decision Tree Classifier For Each Datasets 5
1.1 Model Setup and Training 5
1.2 Experimental Procedure 5

2 Analysis of the Heart Disease dataset. 6
2.1 Data Preparation 6
2.1.1 Dataset Overview 6
2.1.2 Missing Values Handling 7
2.1.3 Dataset Splitting 7
2.2 Performance Evaluation of the Decision Tree Model 7
2.2.1 Evaluation Metrics: Confusion Matrix and Classification Report . . . 7
2.2.2 Insights 8
2.3 Depth and Accuracy of Decision Trees 9
2.3.1 Experimental Setup 9
2.3.2 Results 9
2.3.3 Insights 9
2.3.4 Conclusion 10

3 Analysis of the Palmer Penguins dataset. 11
3.1 Data Preparation 11
3.1.1 Dataset Overview 11
3.1.2 Handling Missing Values 11
3.1.3 Encoding Categorical Features 11
3.1.4 Dataset Splitting 12
3.2 Performance Evaluation of the Decision Tree Model 12
3.2.1 Evaluation Metrics: Confusion Matrix and Classification Report . . . 12
3.2.2 Insights 13
3.3 Depth and Accuracy of Decision Trees 14
3.3.1 Experimental Setup 14
3.3.2 Results 14
3.3.3 Insights 14
3.3.4 Conclusion 15

4 Analysis of the Car Evaluation dataset. 15
4.1 Data Preparation 15
4.1.1 Dataset Overview 15
4.1.2 Encoding Categorical Features 16
4.1.3 Dataset Splitting 16
4.2 Performance Evaluation of the Decision Tree Model 17
4.2.1 Evaluation Metrics: Confusion Matrix and Classification Report . . . 17
4.2.2 Insights 17
4.3 Depth and Accuracy of Decision Trees 18
4.3.1 Experimental Setup 18
4.3.2 Results 18

4.3.3 Insights 18
4.3.4 Conclusion 19

5 Comparative analysis of all three datasets. 19

6 Reference 20

Member information

Full Name	Student ID	Email
Lê Nguyên Thảo	23127118	lnthao23@clc.fitus.edu.vn
Đỗ Đăng Nhật Tiến	23127269	ddntien23@clc.fitus.edu.vn
Nguyễn Thanh Owen	23127447	ntowen23@clc.fitus.edu.vn
Trần Minh Quang	23127464	tmquang23@clc.fitus.edu.vn

Table 1: List of member

Task and Completion

Student ID	Name	Task	Percent
23127118	Lê Nguyên Thảo	Report	100%
23127269	Đỗ Đăng Nhật Tiến	Python Notebooks For The UCI Car Evaluation Dataset	100%
23127447	Nguyễn Thanh Owen	Python Notebooks For The UCI Palmer Penguins Dataset	100%
23127464	Trần Minh Quang	Python Notebooks For The UCI Heart Disease Dataset	100%

Table 2: Assignment

1 Implementing Decision Tree Classifier For Each Datasets

1.1 Model Setup and Training

The decision tree classification method is employed to construct a model capable of predicting the species of penguins based on their morphological traits and geographic origin.

Key parameters used during model training:

- **Splitter criterion:** The model uses the Gini index to measure the purity at each node. This helps determine which feature best splits the data at each step.
- **Tree depth:** The model is trained with various tree depths, including: unlimited depth (None) and specific limits such as 2, 3, 4, 5, 6, and 7. This allows analysis of how model complexity affects performance.
- **Random state:** A fixed parameter is used to ensure reproducibility of results across different runs. This is essential for fair comparison of models.

1.2 Experimental Procedure

For each data ratio, the model training process follows these steps:

1. **Data splitting:** The dataset is divided into training and testing sets using stratified sampling to preserve species distribution across both subsets.
2. **Model training:** The *Decision Tree Classifier* is trained on the training set.
3. **Prediction:** The model predicts labels for the samples in the test set.
4. **Performance evaluation:** The model's performance is evaluated using metrics such as accuracy, confusion matrix, and classification report.
5. **Depth comparison:** Accuracy is recorded and compared across different tree depths to determine the optimal model complexity.

2 Analysis of the Heart Disease dataset.

2.1 Data Preparation

2.1.1 Dataset Overview

In this project, we use the Heart Disease dataset from the UCI Machine Learning Repository (ID: 45). This dataset is widely applied in medical research to build predictive models for heart disease based on various clinical and biological indicators.

Initially, the dataset contained a label column named `num`, with values ranging from 0 to 4: 0 indicated no heart disease, while values from 1 to 4 indicated increasing severity of the disease.

To adapt the dataset for binary classification, the `num` column was renamed to `target` and recoded: 0 remained 0 (no disease), and any value greater than 0 was converted to 1 (disease present).

Finally, the features and the new `target` column were combined into a single table and saved as `heart_disease_binary.csv`, ready for subsequent analysis and modeling.

Column Name	**Data Type**	**Description**
`age`	Integer	Age of the patient in years.
`sex`	Integer	Gender of the patient (0 = female, 1 = male).
`cp`	Integer	Chest pain type (1–4, with different clinical meanings).
`trestbps`	Integer	Resting blood pressure (in mm Hg).
`chol`	Integer	Serum cholesterol level (in mg/dl).
`fbs`	Integer	Fasting blood sugar $>$ 120 mg/dl (1 = true, 0 = false).
`restecg`	Integer	Resting electrocardiographic results (0–2).
`thalach`	Integer	Maximum heart rate achieved.
`exang`	Integer	Exercise-induced angina (1 = yes, 0 = no).
`oldpeak`	Float	ST depression induced by exercise relative to rest.
`slope`	Integer	Slope of the peak exercise ST segment (1–3).
`ca`	Float	Number of major vessels (0–3) colored by fluoroscopy. May contain missing values.
`thal`	Float	Thalassemia status (3 = normal, 6 = fixed defect, 7 = reversible defect). May contain missing values.
`target`	Integer	Heart disease diagnosis (1 = disease, 0 = no disease). Target variable.

2.1.2 Missing Values Handling

During data inspection, we identified missing values in two columns: `ca` and `thal`. Specifically, there are 4 missing entries in `ca` and 2 missing entries in `thal`.

Since the number of missing values is relatively small compared to the total number of records (303 samples), we chose to remove rows containing missing values. This simple yet effective approach helps ensure that subsequent analysis and model training are based on complete and reliable data.

2.1.3 Dataset Splitting

After preprocessing, the dataset was divided into training and testing subsets using various train-test ratios:

- 40% training – 60% testing
- 60% training – 40% testing
- 80% training – 20% testing
- 90% training – 10% testing

Applying different split ratios allows us to evaluate how the size of the training data impacts model performance. Typically, larger training sets help improve learning, while sufficient testing data ensures reliable evaluation.

To maintain consistency, **stratified sampling** was used during the splitting process. This technique preserves the original distribution of heart disease cases across both training and testing sets, preventing class imbalance and ensuring fair model assessment.

2.2 Performance Evaluation of the Decision Tree Model

2.2.1 Evaluation Metrics: Confusion Matrix and Classification Report

After completing the training phase with different train-test split ratios and various tree depths, the Decision Tree classifiers were evaluated based on their predictive performance on the corresponding test sets.

For each model, predictions were made on the test data and compared against the true labels. Two primary evaluation tools were employed:

- **Classification Report:** This report summarizes important metrics:
  - **Precision:** The proportion of correctly predicted positive samples among all samples predicted as positive, calculated as:

$$\text{Precision} = \frac{TP}{TP + FP}$$

- **Recall:** The proportion of correctly predicted positive samples among all actual positive samples, calculated as:

$$\text{Recall} = \frac{TP}{TP + FN}$$

- **F1-score:** The harmonic mean of precision and recall, balancing the two:

$$\text{F1-Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

- **Accuracy:** The proportion of all correctly classified samples:

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

- **Confusion Matrix:** The confusion matrix records:

  - **True Positive (TP):** Correct predictions of heart disease presence.
  - **True Negative (TN):** Correct predictions of heart disease absence.
  - **False Positive (FP):** Incorrect predictions of disease in healthy individuals.
  - **False Negative (FN):** Incorrect predictions of health in diseased individuals.

  In the confusion matrix:

  - Rows represent the actual class labels (ground truth).
  - Columns represent the predicted class labels.

Confusion matrices were visualized using heatmaps to facilitate the identification of error patterns and misclassification tendencies.

This comprehensive evaluation allows for an in-depth analysis of each model's performance, providing insights into the trade-off between detecting true cases and minimizing false alarms, and helping to determine optimal model configurations.

2.2.2 Insights

The evaluation results highlight several important factors that influence the performance of decision tree classifiers:

- **Trade-off between Precision and Recall:** A model with high precision makes fewer mistakes in predicting healthy patients as sick, but it may miss some real cases of disease. On the other hand, a model with high recall can detect more true patients but might incorrectly classify some healthy individuals. Choosing between precision and recall depends on whether avoiding missed cases or reducing false alarms is more important.

- **Effect of Training Data Size:** Training with more data (such as 80% or 90% of the dataset) improves the model's ability to learn patterns and make accurate predictions.

- **Impact of Class Imbalance:** Although stratified sampling was used, the model still predicted slightly better for the majority class. Minor class imbalance can affect results, and rebalancing techniques may help improve fairness.

2.3 Depth and Accuracy of Decision Trees

2.3.1 Experimental Setup

To systematically assess the impact of tree depth, the Decision Tree Classifier was trained and evaluated using various `max_depth` settings:

- Depth values tested: **None**, **2**, **3**, **4**, **5**, **6**, and **7**.

For each depth:

- Stratified sampling was employed during data splitting to maintain class distribution.
- Model performance was measured based on the accuracy obtained on the test set.

2.3.2 Results

The table below summarizes the test set accuracy achieved at each depth level:

Max Depth	Test Set Accuracy (%)
None	76.67
2	71.67
3	81.57
4	76.67
5	76.67
6	73.33
7	76.67

2.3.3 Insights

- **Optimal tree depth:** The maximum depth of 3 achieved the highest test set accuracy at 81.57%. This indicates that moderately deep trees are most effective in capturing important patterns in the dataset without overfitting.

- **Impact of shallow trees:** A shallow tree with depth 2 resulted in the lowest accuracy (71.67%), suggesting that the model was too simple to model the underlying relationships effectively, leading to underfitting.
- **Impact of deeper trees:** Trees with greater depth values, such as 6, also exhibited reduced accuracy (73.33%), implying that overfitting may start to occur when the tree becomes too complex.
- **Stability across moderate depths:** Depths set at None, 4, 5, and 7 all produced similar accuracies around 76.67%. This stability indicates that once the tree reaches a certain complexity threshold, additional depth does not significantly improve performance.

2.3.4 Conclusion

Although test set accuracy was relatively stable around 76.67% across most tree depths, the model achieved a noticeably higher accuracy of 81.57% at depth 3. This suggests that a moderately deep tree captures important patterns better than very shallow or overly deep trees. Depth 2 resulted in the lowest accuracy (71.67%), indicating underfitting, while deeper trees like depth 6 showed signs of reduced generalization. Overall, selecting an appropriate depth, such as 3, significantly improves model performance without unnecessary complexity.

3 Analysis of the Palmer Penguins dataset.

3.1 Data Preparation

3.1.1 Dataset Overview

The dataset used in this project is the **Palmer Penguins** dataset, a widely-used alternative to the classic Iris dataset. It contains biological measurements of three penguin species: *Adelie*, *Chinstrap*, and *Gentoo*, collected from the *Palmer Archipelago* in Antarctica.

Column Name	**Data Type**	**Description**
`species`	String	Penguin species: *Adelie*, *Chinstrap*, or *Gentoo*. This is the target label.
`island`	String	Island where the penguin was found: *Biscoe*, *Dream*, or *Torgersen*.
`bill_length_mm`	Float	Bill length in millimeters. A key morphological feature.
`bill_depth_mm`	Float	Bill depth in millimeters. Used together with bill length for classification.
`flipper_length_mm`	Integer	Flipper length in millimeters. Correlates with species and size.
`body_mass_g`	Integer	Body mass in grams. Related to species and gender.
`sex`	String	Gender of the penguin: *Male* or *Female*.
`year`	Integer	Year of data collection: 2007, 2008, or 2009.

3.1.2 Handling Missing Values

The original dataset contains some missing entries, particularly in the **sex**, **bill_length_mm**, and **body_mass_g** columns. These missing values can negatively impact model accuracy and consistency.

3.1.3 Encoding Categorical Features

Machine learning algorithms require numerical input. Therefore, categorical text columns were transformed into numeric values using label encoding.

Encoding mapping:

- `species`: Adelie → 0, Chinstrap → 1, Gentoo → 2

- `island`: Biscoe → 0, Dream → 1, Torgersen → 2
- `sex`: Female → 0, Male → 1

  This encoding helps models process and learn from the data effectively.

3.1.4 Dataset Splitting

After preprocessing, the dataset was split into training and testing subsets using several train-test ratios:

- 40% training – 60% testing
- 60% training – 40% testing
- 80% training – 20% testing
- 90% training – 10% testing

Using multiple split ratios allows us to observe how training data size affects model performance.

To ensure fairness, **stratified sampling** was applied. This preserves the proportion of each penguin species in both training and testing sets, avoiding class imbalance.

3.2 Performance Evaluation of the Decision Tree Model

3.2.1 Evaluation Metrics: Confusion Matrix and Classification Report

To assess the performance of the Decision Tree classifier in identifying penguin species, two primary evaluation tools were employed:

- **Confusion Matrix:** This provides a tabular summary of the model's predictions, showing how many instances were correctly or incorrectly classified for each class. It is particularly useful for detecting patterns of misclassification.

  Example (Classification Report for 40% Training Set)

  - Adelie penguins: 78 were correctly classified as Adelie, while 10 were misclassified as Chinstrap.
  - Chinstrap penguins: All 41 were correctly classified without any errors.
  - Gentoo penguins: 70 were correctly classified as Gentoo, and 1 was misclassified as Chinstrap.

- **Classification Report:** This includes key metrics for each species:

  - *Precision* – the ratio of correctly predicted positive observations to total predicted positives.

- *Recall* – the ratio of correctly predicted positives to all actual positives.
- *F1-score* – the harmonic mean of precision and recall.

Example (Classification Report for 40% Training Set)

- Adelie shows perfect **precision** (1.00) but a slightly lower **recall** (0.89), meaning that while most Adelie predictions are correct, the model occasionally misses some actual Adelie samples.
- Chinstrap has a **recall** of 1.00, indicating that all actual Chinstrap samples were correctly identified. However, its **precision** is lower at 0.79, suggesting that other species are sometimes misclassified as Chinstrap.
- Gentoo performs the best overall, with near-perfect **precision** and **recall** (1.00 and 0.99), meaning the model is highly confident and accurate in recognizing this species.

3.2.2 Insights

- **Precision and Recall Behavior**:
  - **Precision** remained high across all splits, especially for Adelie and Gentoo, indicating few false positive predictions.
  - **Recall** values were generally lower, suggesting that the model missed some true cases, particularly for Adelie in the 40% training split and for Gentoo in the 90% training ratios.
- **Impact of Confusion between Classes**:
  - The **Confusion Matrix** showed that most misclassifications happened between Adelie and Chinstrap species.
  - Gentoo penguins were classified almost perfectly across all splits, as confirmed by both the Classification Report and Confusion Matrix.
  - This indicates that Gentoo has more **distinctive features**, making it easier for the model to recognize.
- **Effect of Training Size on Accuracy**:
  - As shown in the Classification Report, overall **accuracy increased** from 94% to 97% when moving from 40% to 60% training data.
  - However, when increasing the training size to 80% and 90%, the **accuracy improvement plateaued**, suggesting diminishing returns.
  - This indicates that after a certain point, adding more training data provides limited additional improvement.

3.3 Depth and Accuracy of Decision Trees

3.3.1 Experimental Setup

To systematically assess the impact of tree depth, the Decision Tree Classifier was trained and evaluated using various `max_depth` settings:

- Depth values tested: **None**, **2**, **3**, **4**, **5**, **6**, and **7**.

For each depth:

- Stratified sampling was employed during data splitting to maintain class distribution.
- Model performance was measured based on the accuracy obtained on the test set.

3.3.2 Results

The table below summarizes the test set accuracy achieved at each depth level:

Max Depth	**Test Set Accuracy (%)**
None	94.03
2	94.03
3	95.52
4	94.03
5	94.03
6	94.03
7	94.03

3.3.3 Insights

- **Impact of Tree Depth:**
  - The model achieved the highest test set accuracy (95.52%) when the tree depth was limited to 3.
  - For other depths (None, 2, 4, 5, 6, and 7), the accuracy remained stable around 94.03%, suggesting that increasing or decreasing tree depth beyond 3 did not bring further improvement.
- **Impact of Shallow Trees:**

  - At depth 2, the model achieved the same accuracy (94.03%) as deeper trees, suggesting that even shallow trees were sufficient to capture the main patterns of the dataset.

- **Impact of Deeper Trees:**
  - For deeper trees (depths 4, 5, 6, 7, and None), the test accuracy remained stable at 94.03%, showing that increasing depth beyond 3 did not bring additional benefits.

3.3.4 Conclusion

The experimental results show that limiting the Decision Tree's depth to 3 achieved the highest test accuracy (95.52%). However, the model exhibited stable and high performance (94.03%) across other tested depths, indicating that tree complexity had a limited impact on overall effectiveness.

4 Analysis of the Car Evaluation dataset.

4.1 Data Preparation

4.1.1 Dataset Overview

The additional dataset is the Car Evaluation dataset, which is a classical dataset frequently employed in classification problems within the field of machine learning. This dataset provides an evaluation of the acceptability of different car models based on several features such as buying price, maintenance cost, number of doors, capacity, luggage boot size, and safety level.

Column Name	**Data Type**	**Description**
`buying`	String	Buying price of the car: vhigh, high, med, low.
`maint`	String	Maintenance cost: vhigh, high, med, low.
`door`	String	Number of doors: 2, 3, 4, 5more.
`persons`	String	Capacity in terms of persons to carry: 2, 4, more.
`lug_boot`	String	Size of luggage boot: small, med, big.
`safety`	String	Safety rating: low, med, high.
`class`	String	Car acceptability class (target label): unacc, acc, good, vgood.

The **class** column is the target for prediction, representing the acceptability of the car.

4.1.2 Encoding Categorical Features

Since all features in the dataset are of a **string nature**, it is necessary to transform them into a numerical format to enable machine learning models (specifically decision trees) to process the data effectively.

The encoding technique used for this transformation is **Label Encoding**, which maps each distinct category to a unique integer value.

Encoding mapping:

- buying: vhigh → 3, high → 2, med → 1, low → 0
- maint: vhigh → 3, high → 2, med → 1, low → 0
- doors: 2 → 0, 3 → 1, 4 → 2, 5more → 3
- persons: 2 → 0, 4 → 1, more → 2
- lug_boot: small → 0, med → 1, big → 2
- safety: low → 0, med → 1, high → 2
- class (target): unacc → 0, acc → 1, good → 2, vgood → 3

4.1.3 Dataset Splitting

After preprocessing, the dataset was divided into training and testing subsets using various **train-test ratios**:

- 40% training – 60% testing
- 60% training – 40% testing
- 80% training – 20% testing
- 90% training – 10% testing

Applying different split ratios allows us to evaluate how the size of the training data impacts model performance. Typically, larger training sets help improve learning, while sufficient testing data ensures reliable evaluation.

To maintain consistency, **stratified sampling** was applied during splitting to ensure the proportions of car acceptability classes were preserved in both the training and testing sets.

4.2 Performance Evaluation of the Decision Tree Model

4.2.1 Evaluation Metrics: Confusion Matrix and Classification Report

To thoroughly assess the classification performance of the model, two key evaluation tools were employed: **Confusion Matrix** and **Classification Report.**

- **Classification Report:** The Classification Report offers essential metrics such as Precision, Recall, and F1-Score for each class individually, giving a deeper insight into the model's strengths and weaknesses.

- **Confusion Matrix:** The Confusion Matrix provides a detailed visualization of the number of correctly and incorrectly classified instances for each class, helping to pinpoint where the model tends to make mistakes.

Key observations from the experiments: Instances belonging to the unacc class were predicted with extremely high precision and recall, indicating the model's strong capability to identify unacceptable cars.

For the acc class, although the precision was relatively good, the recall was slightly lower, suggesting that some acceptable cars were misclassified into other categories.

The good and vgood classes were more prone to mutual misclassification, particularly when the decision tree had a shallow depth, which is understandable as these two classes share many similar attributes.

4.2.2 Insights

The evaluation results from the Car Evaluation dataset reveal several key observations regarding decision tree classifier performance:

- **Trade-off between Precision and Recall:** Across different train-test splits (40%-60%, 60%-40%, 80%-20%, and 90%-10%), precision and recall values fluctuated slightly. In splits with less training data (e.g., 40%-60%), precision tended to decrease, indicating a higher likelihood of **misclassification**. As the training set size increased (e.g., 80%-20%, 90%-10%), both precision and recall improved, suggesting that the model became better at correctly identifying car evaluation categories.

- **Effect of Training Data Size:** Expanding the training set size consistently enhanced model performance. Models trained with 80% or 90% of the data outperformed those trained with 40% or 60%, confirming that a larger and more diverse training set helps the decision tree capture underlying patterns more effectively and improves overall prediction stability.

- **Impact of Class Imbalance:** Despite applying stratified sampling, slight class imbalances persisted. The model achieved better performance for dominant classes like "acc" and "good", while rarer classes such as "vgood" showed lower classification rates. Incorporating resampling or other class balancing strategies could further improve fairness and accuracy across all categories.

4.3 Depth and Accuracy of Decision Trees

4.3.1 Experimental Setup

To systematically assess the performance of the Decision Tree Classifier on the Car Evaluation dataset, several experiments were conducted:

- Depth values tested: **None**, **2**, **3**, **4**, **5**, **6**, and **7**.

For each depth:

- **Stratified sampling** was employed during data splitting to preserve the class distribution across training and test sets.
- The model was trained using the specified *max_ depth*, and performance was assessed based on the accuracy obtained on the test set.
- Different train-test split ratios (60%-40%, 40%-60%, 80%-20%, and 90%-10%) were also explored to study the effect of training data size in combination with tree depth.

4.3.2 Results

The table below summarizes the test set accuracy achieved at each depth level:

Max Depth	**Test Set Accuracy (%)**
None	97.98
2	77.75
3	77.75
4	81.50
5	85.26
6	89.02
7	92.49

4.3.3 Insights

- **Optimal tree depth:** Setting max_depth=None (allowing the tree to grow fully) achieved the highest test set accuracy at 97.98%. This suggests that, for the Car Evaluation dataset, a fully grown decision tree was highly effective at capturing all relevant patterns without significant overfitting.

- **Impact of shallow trees:** A shallow tree with depth 2 & 3 yielded the lowest test accuracy (77.75%), indicating that the model was too simple to capture the complex decision boundaries among different car evaluation categories, resulting in underfitting.
- **Impact of deeper trees:** Trees with moderate to deep depths, such as 6 (89.02% accuracy) and 7 (92.49% accuracy), performed better than shallow trees. However, the slight decrease compared to the fully grown tree indicates that limiting depth, even moderately, may slightly restrict model performance.
- **Stability across moderate depths:** Trees with depths of 4, 5, 6, and 7 exhibited steadily increasing accuracies from 81.50% to 92.49%. This trend suggests that deeper trees generally perform better on this dataset, but the most significant gains occur once the tree reaches a moderate to high depth.

4.3.4 Conclusion

Although test set accuracy improved steadily with increased tree depth, the model achieved the highest performance of 97.98% when no maximum depth constraint was applied. Among limited-depth models, a depth of 7 yielded strong accuracy (92.49%), suggesting that deeper trees are more effective at capturing the complex structure of the Car Evaluation dataset. Very shallow trees, such as depths 2 and 3 (both achieving only 77.75% accuracy), exhibited clear underfitting, failing to model essential relationships. Overall, allowing greater depth significantly enhances model performance.

5 Comparative analysis of all three datasets.

Criteria	**Heart Disease**	**Penguins**	**Car Evaluation**
Data Type	Numerical	Numerical + Categorical	Categorical
Data Distribution	Slightly skewed	Normal (numerical); Imbalanced (categorical)	Discrete, fairly even
Target Variable	0: No, 1: Yes	Adelie, Gentoo, Chinstrap	unacc, acc, good, vgood
Target Balance	Balanced (55%-45%)	Slight imbalance	Severe imbalance
Missing Values	6 (numerical)	19 (mixed types)	None
Scaling/Encoding	Scaling needed	Scaling + Encoding	Encoding only

Table 3: Comparative Analysis of Heart Disease, Penguins, and Car Evaluation Datasets

6 Reference

1.Hướng dẫn cài đặt phần mềm Graphviz
2.Heart Disease Detection in Python - Machine Learning Project
3.Classifying Penguin Species using Machine Learning Algorithms
4. What is A Confusion Matrix in Machine Learning?
5.Compute Classification Report and Confusion Matrix in Python
6.Decision Tree