

**Faculty of Engineering & Technology**

**Electrical & Computer Engineering Department**

**Artificial intelligent**

**ENCS 3340**

**Project 2**

**Comparative Study of Image Classification Using Decision Tree, Naive Bayes, and Feedforward Neural Networks**

**Prepared by:**

Name: Naeem Mrabie Number:1210787

Partner Name: Tariq Zaid Number: 1212974

**Instructor:** Dr.Samah Alaydi

**Section:**3

**Date:** 30/6/2025

# **Abstract:**

This project evaluates three machine learning methods—Decision Tree, Naive Bayes, and Neural Network—for classifying animal images. We use a set of 780 pictures sorted into six groups: bird, cat, cow, dog, lion, and panda. Each picture is resized to (64×64) pixels, converted into a single row of pixel values, and split into 624(80%) images for training and 156(20%) for testing. Naive Bayes offers a quick, baseline approach by assuming pixel values is independent. Decision Trees build simple rules that clearly show how each image is classified based on key pixels. The Neural Network MLPClassifier explores deeper patterns by learning hidden representations across one or two layers. We assess each model using accuracy, precision, recall, F1-score, and confusion matrices to understand both overall performance and specific class strengths or weaknesses. Our findings highlight trade-offs: Naive Bayes is fastest but less precise; Decision Trees are easy to interpret yet can overfit; Neural Networks achieve the highest accuracy at the cost of longer training. We used python programming language, Kaggle for collecting images.

# **Introduction:**

Image classification means teaching a computer to recognize what’s in a picture like Whether an image shows a cat, dog, cow, bird, lion, and panda. This project compares three popular ways to do this:

* Naïve Bayes classifier: a fast basic method that uses probability.
* Decision Tree classifier: makes decision by following simple (like Entropy) rules, so it easy to understand.
* Neural Network: A more advance technique that can find complex patterns.

The objective is to analyze each method’s strengths and limitations, particularly for high -dimensional data such as images.

# **Dataset Description and preprocessing**

## **Dataset:**

We created a dataset with 780 images, sorted into six animal classes: bird, cat, dog, cow, lion, and panda. Each class has about 130 images. For each animal, 104 images are used to train the model and 26 images are used to test it. This gives enough variety to properly evaluate how well each model works.

## **Preprocessing steps:**

* **Resizing**: All images were resized to 64x64 pixels. By resizing all images to the same dimensions, the model learns from consistent input features.
* **Flattening**: Each image is converted to a 1D array of pixel values (length:4096).
* **Splitting**: Data split into 80% training (624 images) and 20% testing (156 images).
* **CSV File Organization:** All image file paths and their corresponding class labels were stored in a CSV file with two columns: image path and label.
* **Normalization**: pixel values scaled to [0,1] range for neural network training.

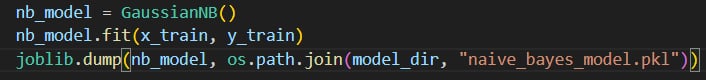
# **Model Implementation:**

## **Naïve Bayes Classifier:**

The Naïve Bayes approach treats each pixel as an independent feature. For each class, it estimates thee mean and variance of each pixel across the training set, and classifies new image by calculating the probability of observing their pixel values given in each class.

Key properties:

* Fast training /prediction
* Assume pixel independence
* Implementation: used GaussianNB from scikit learn.



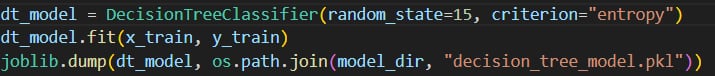
**Figure 1: implementation Naive Bayes classifier**

## **Decision Tree Classifier:**

The decision Tree learns a hierarchy of rules based on pixel intensities, splitting the data to maximize class separation at each node, the Decision tree used for training Recursive divide and concur approach.

Key properties:

* Highly interpretable
* Can overfit high dimensional data
* Implementation: Used DecisionTreeClassifier from scikit -learn.



**Figure 2: implementation Decision Tree**

## **Neural Network (MLP Classifier):**

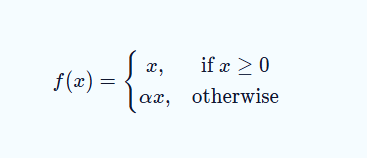
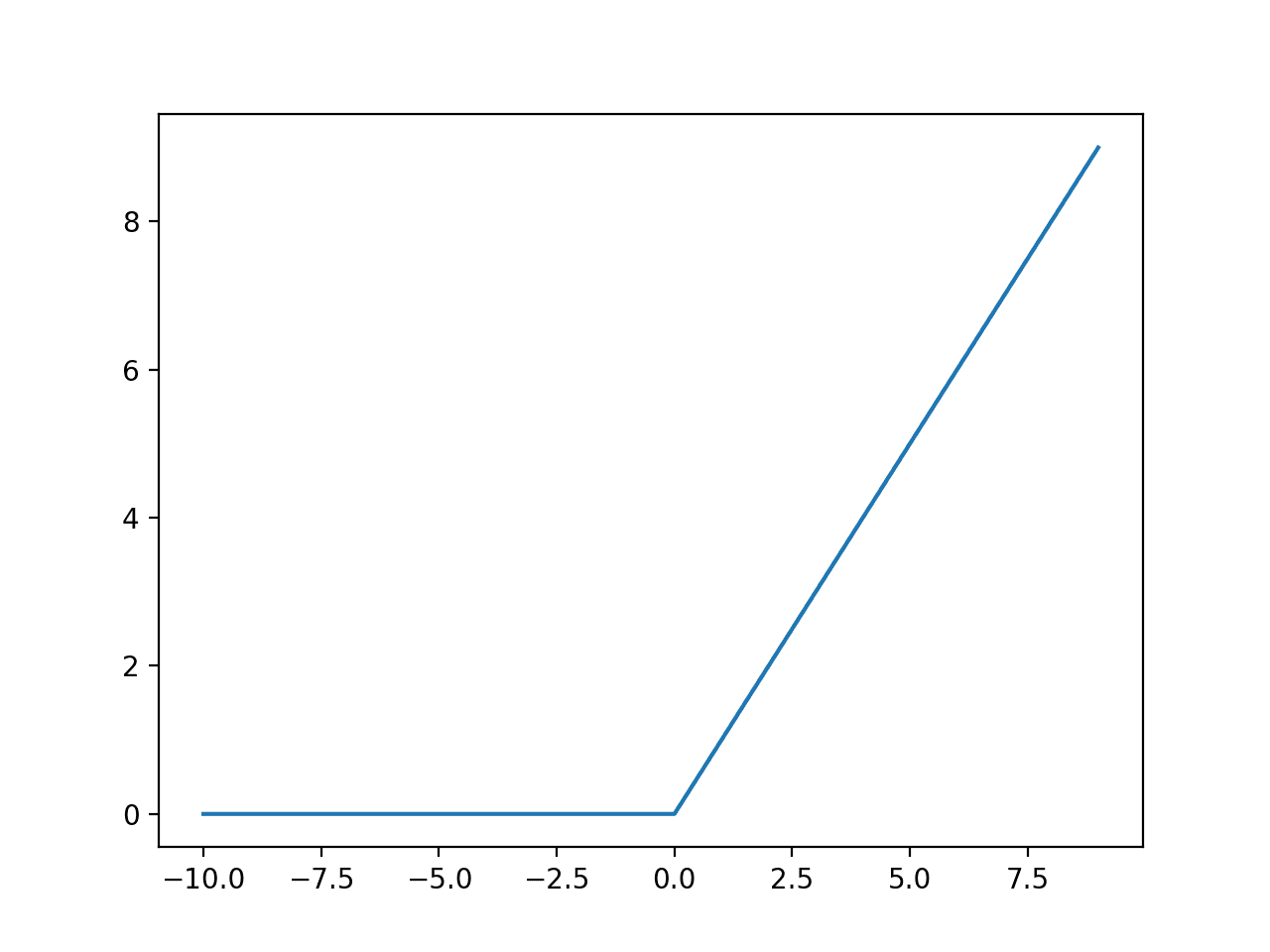
A Feedforward Network (Multi – layer Perceptron, or MLP) can model complex, non – linear relationships between pixel data and class label by learning deep representation.

Key properties:

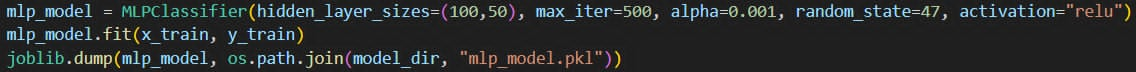
* Can achieve higher accuracy compared to simpler models
* Requires more training time and careful parameter tuning.

Implementation details:

* Used MLPClassfier from scikit learn
* Architecture: in the first step we using one hidden layer size = 50 neuron, and then we update to a two hidden layer in the first layer=100 neuron and in the second layer 50 neuron for better efficiency.
* Learning rate: 0.001
* Maximum iteration: 500
* Random state: 47 (for reproducibility)
* Activation Function: Relu (Rectifier Linear unit) for negative value small multiplying factor and for positive value the value of input.



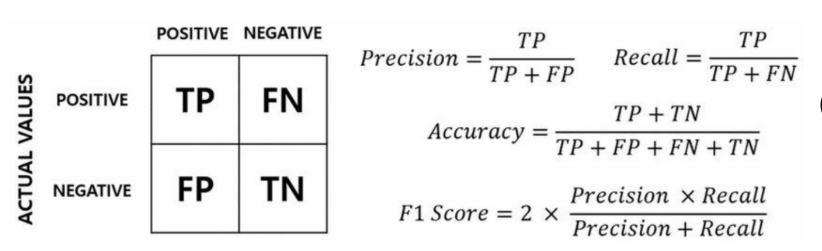
**Figure 3: Rectifier linear unit activation function plot using MATLAB and how it is exactly work**



**Figure 4: implementation Neural Network (MLP classifier)**

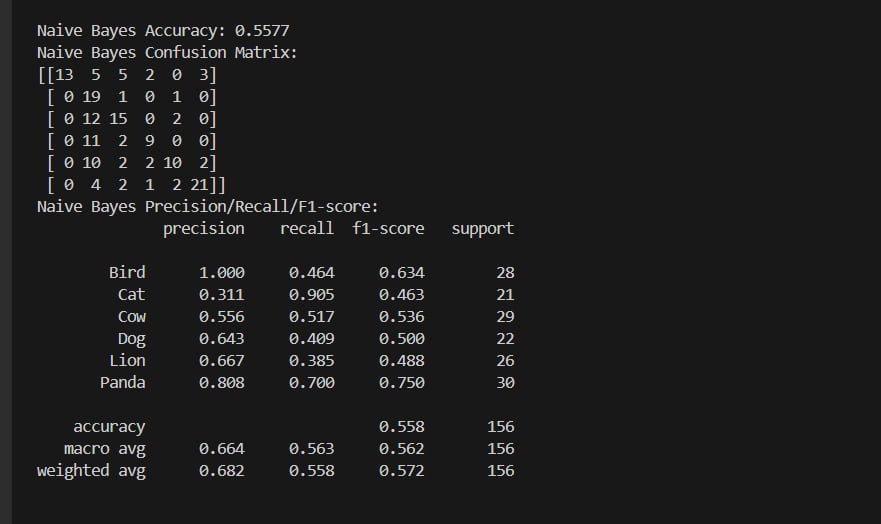
# **Evaluation Metrics:**

1. **Accuracy:** is the number of correctly classified positive and negative examples divide by the total number of examples in the test set. Is not suitable in some applications.
2. **Precision:** is the number of correctly classified positive examples divide by the total number of examples that are classified as positive.
3. **Recall:** is the number of correctly classified positive examples divide by the total number of actual positive example in the test set.
4. **F1-Score:** is the harmonic mean of precision and recall**.**
5. **Confusion Matrix:** a table (2-D Array) that summarize the performance of classification model by showing how many predictions were correct and incorrect, broken down by class.



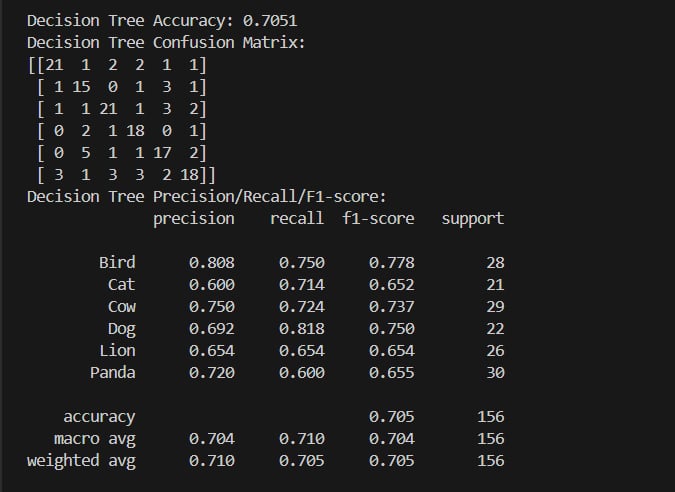
**Figure 5: Evaluation Metric Summary**

## **Accuracy, precision, Recall, f1-score (Naïve Bayes)**



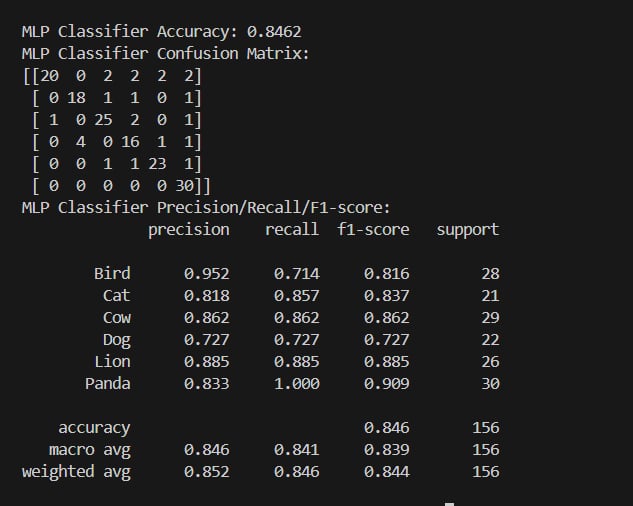
**Figure 6: Detailed Evaluation Metrics (Naive Bayes)**

## **Accuracy, precision, Recall, f1-score (Decision Tree)**



**Figure 7: Detailed Evaluation Metrics (Decision Tree)**

## **Accuracy, precision, Recall, f1-score (Neural Network MLP Classifier)**



**Figure 8: Detailed Evaluation Metrics (MLP Classifier)**

# **Discussion:**

## **Naïve Bayes**

Naïve bayes was the fastest and easiest to use. It trained almost instantly on the whole dataset, but its simplicity led to lower accuracy. It is not inherently designed for image processing, as it assumes feature independence and is best suited for structured data such as text classification, spam detection, and medical diagnosis. However, it can be applied to image classification tasks when images are transformed into low-dimensional, statistical feature vectors such as color histograms. While it is fast, simple, and effective in many cases, Naïve Bayes tends to perform better in recall than precision, meaning it is more likely to correctly identify all positive cases, but may misclassify some negatives as positives—especially in imbalanced datasets.

## **Decision Tree**

Decision Trees were still fast and easy to understand. However, its accuracy stopped improving with more data. It can also overfit and pick up noise if not pruned. It is not inherently designed for processing raw image data, but they can be effectively used for image classification when images are first transformed into compact feature vectors such as color histograms. While they are highly interpretable and capable of handling non-linear decision boundaries, Decision Trees tend to overfit on high-dimensional inputs like raw pixel data. They are commonly used in structured classification tasks, such as medical diagnosis and decision support systems. In terms of performance, Decision Trees often achieve high precision but may suffer in recall if the model overfits or if class imbalance exists, meaning they are better at correctly predicting positive cases than capturing all actual positives.

## **5.3 Neural Network**

Neural Networks took the most time and effort to tune. They gave the best accuracy by learning complex patterns. The downside is they’re harder to interpret. They also need more computing power and time to train. it is a fully connected feed-forward neural network employed for a broad range of classification and regression tasks—including fraud detection, speech recognition, and image classification when images are reduced to structured feature vectors (e.g., color histograms, HOG descriptors). Because each neuron in an MLP is linked to every input feature, the model can capture complex, non-linear relationships that simpler algorithms (e.g., Naïve Bayes, Decision Trees) cannot. In evaluation, two key metrics are precision (the proportion of predicted positives that are correct) and recall (the proportion of actual positives that are successfully retrieved). MLPs typically provide a balanced trade-off between these metrics—outperforming Naïve Bayes in precision (fewer false positives) and improving recall over plain Decision Trees (fewer false negatives) when properly tuned and trained on sufficient data.

## 

|  |  |
| --- | --- |
| Library | Used for |
| OS | |  | | --- | | Handles file and directory operations (creating folders, joining paths, renaming files) |  |  | | --- | |  | |
| csv | |  | | --- | | Reads from and writes to CSV files to store image paths and labels. |  |  | | --- | |  | |
| Cv2 | |  | | --- | | Processes images: reading, resizing, color conversion, and feature extraction (histograms). |  |  | | --- | |  | |
| numpy | |  | | --- | | Performs efficient numerical operations and handles arrays for machine learning input. |  |  | | --- | |  | |
| joblib | |  | | --- | | Saves and loads trained models (serialization) without retraining. |  |  | | --- | |  | |
| |  | | --- | | matplotlib.pyplot |  |  | | --- | |  | | |  | | --- | | Visualizes the decision tree using plotting functions. |  |  | | --- | |  | |
| |  | | --- | | train\_test\_split |  |  | | --- | |  | | |  | | --- | | Splits the dataset into training and testing sets for model evaluation. |  |  | | --- | |  | |
| accuracy\_score, confusion\_matrix, classification\_report | |  | | --- | | Evaluates model performance using accuracy and classification metrics. |  |  | | --- | |  | |
| |  | | --- | | GaussianNB |  |  | | --- | |  | | |  | | --- | |  |  |  | | --- | | A probabilistic Naïve Bayes classifier used as a simple baseline model. | |
| |  | | --- | | DecisionTreeClassifier |  |  | | --- | |  | | A rule-based classifier that builds a decision tree from the training data. |
| |  | | --- | | plot\_tree |  |  | | --- | |  | | |  | | --- | | Visualizes the structure of the decision tree model. |  |  | | --- | |  | |
| |  | | --- | | MLPClassifier |  |  | | --- | |  | | Implements a neural network to learn complex non-linear relationships in the data. |