



FACULTY OF ENGINEERING & TECHNOLOGY

Computer Science Department

Artificial Intelligence – ENCS3340

Project Report

Comparative Study of Image Classification

Prepared by:

Duha Imad 1220623

Leen Anas 1220380

Instructor: Yazan Abu Farha.

Date: 30/6/2025

Section: 4

Introduction

In this project, we look at how well different machine learning models can classify images into the correct categories. The three models we use are a Decision Tree, a Naive Bayes classifier, and a Multi-Layer Perceptron (MLP) neural network. Our main goal is to compare these models and see which one gives the best results on our image dataset.

We start by preparing the data. The images are sorted into folders for each class. To make them easier to work with, we turn all the images into grayscale and resize them to 64×64 pixels. This way, every image looks the same size and has only one color channel. After that, we flatten the image into a one-dimensional list of pixel values so that we can feed it into the models. Next, we split the data into two parts: a training set (80% of the dataset) and a testing set (20% of the dataset). The training set is used to help the models learn what each class looks like, and the testing set is used to check how well they do on new images they haven't seen before.

Finally, we train the three models and evaluate them by looking at their accuracy, as well as their confusion matrix and classification report. We also check the time it takes for each model to run. This lets us compare not only their accuracy but also how fast they are.

Dataset Description

For this project, The dataset contains three classes with one folder each: animals, flowers, and traffic signs. Each of these folders holds exactly 800 image, which gives us a good number of examples for training and testing.

Preprocessing the Dataset for Machine Learning

The first step involved loading the images from organized folders, each representing a different class. To simplify the data and focus on intensity rather than color, every image was converted to grayscale. Then, to ensure uniformity across the dataset, all images were resized to 64 by 64 pixels. Each resized grayscale image was flattened into a one-dimensional array, turning the 2D pixel grid into a long vector of 4096 values, which is suitable for feeding into machine learning models. Finally, the dataset was split into training and testing sets using an 80:20 ratio to train the models on most of the data and evaluate their performance on unseen images.

Models Implemented

- **Decision Tree**

A Decision Tree is a model that works like a flowchart, making decisions by asking simple questions based on the image features, such as pixel brightness. It splits the data into branches step by step to sort the images into different classes. We built and trained the decision tree using scikit-learn's `DecisionTreeClassifier()`, which automatically finds the best ways to divide the data to correctly classify the images. Decision Trees are easy to interpret and can handle complex decision boundaries, making them suitable for understanding which features (pixels) are most important.

- **Naive Bayes**

Naive Bayes is a model that uses probability to classify data based on Bayes' theorem. It assumes all features, like pixel values, are independent from each other, which is a simple but powerful assumption. Even with this idea, it works well, especially with lots of features like pixels. We used the `GaussianNB` version because it assumes the pixel values follow a normal (Gaussian) distribution, which fits well with continuous data like grayscale images.

Naive Bayes models are fast and effective, especially when features are independent, which often works well for image pixels.

- **Multi-Layer Perceptron (MLP)**

A Multi-Layer Perceptron is a type of feedforward neural network where data passes through layers of interconnected nodes. It learns to represent complex patterns by adjusting weights during training. For this project, we chose an MLP architecture with two hidden layers containing 256 and 128 neurons respectively, which allows the model to learn deeper feature representations. The training ran for up to 300 iterations (max_iter=300), giving the network enough time to converge and improve accuracy on the classification task.

Results

Below are the results for the decision tree showing the confusion matrix, and the classification report.

As we can see we got a good results with a reasonable amount of time.

```
=== Decision Tree Results ===
Confusion Matrix:
[[ 93  63  12]
 [ 54  77  24]
 [ 18  36 103]]

Classification Report:
              precision    recall  f1-score   support

     0       0.56         0.55         0.56         168
     1       0.44         0.50         0.47         155
     2       0.74         0.66         0.70         157

 accuracy          0.57
 macro avg         0.58         0.57         0.57         480
weighted avg         0.58         0.57         0.57         480

Execution Time for decision tree model: 7.20 seconds
```

Below are the results for the naïve bayes showing the confusion matrix, and the classification report.

We got good results with great time!

```
=== Naive Bayes Results ===
Confusion Matrix:
[[111  42  15]
 [ 39  72  44]
 [ 34  25  98]]

Classification Report:
              precision    recall  f1-score   support

     0       0.60       0.66       0.63       168
     1       0.52       0.46       0.49       155
     2       0.62       0.62       0.62       157

 accuracy          0.59          480
 macro avg       0.58       0.58       0.58       480
weighted avg       0.58       0.59       0.58       480

Execution Time for naive bayes model: 0.30 seconds
```

For reference, Class 1 is “animals” , 2 is “flowers” ,and 3 was for “traffic signs”.

Below are the results for the MLP classifier showing the confusion matrix, and the classification report.

We got great results with longer time than the previous two models.

```
=== MLP Classifier Results ===
[[131  29   8]
 [ 68  79   8]
 [ 33  24 100]]

Classification Report:

```

	precision	recall	f1-score	support
0	0.56	0.78	0.66	168
1	0.60	0.51	0.55	155
2	0.86	0.64	0.73	157
accuracy			0.65	480
macro avg	0.68	0.64	0.65	480
weighted avg	0.67	0.65	0.65	480

```
Execution Time for MLP model: 41.97 seconds
```

As we can see above, we built three models getting three different results, we analyzed their performance using standard classification metrics: **accuracy**, **precision**, **recall**, **F1-score**, and **execution time**.

Let's declare the meaning of each measure and compare the results of each model:

1. Precision

Precision measures how many of the predicted positive results are actually correct. It's important in cases where false positives are costly. In this project, a model with low precision for a class often incorrectly assigns many images to that class even if they don't belong to it. Now for the comparison

- We observed that **Naive Bayes** and **MLP** generally performed better in precision, especially for class 2, meaning they made fewer false predictions for that class.

2. Recall

Recall tells us how many actual positives were correctly identified. A high recall means few false negatives, which is useful when it's important not to miss correct predictions.

- MLP showed strong recall for some classes, indicating it was better at "catching" the correct instances and not missing them.
- In contrast, **Decision Tree and Naive Bayes** had lower recall for certain classes, meaning they often missed actual instances of that class.

3. F1-Score

F1-score balances precision and recall. it's a better than both when there's an uneven class distribution or when both false positives and false negatives matter.

- MLP showed the most balanced F1-scores, especially for the more visually varied classes, suggesting that it handled the classification task better overall. The Decision Tree, however, had more unbalanced F1-scores, meaning it was less consistent across classes.

4. Accuracy

Accuracy is the overall percentage of correct predictions. While it's easy to calculate, it doesn't reflect how well the model performs on each class, especially if some classes are harder than others, so we cannot depend on it totally.

- MLP had the highest overall accuracy, likely because it captures complex patterns better.
- Naive Bayes and Decision Tree were slightly behind, with the Decision Tree struggling a bit more with one of the classes.

5. Execution Time

Execution time is a measure of how long the model takes to run.

- **Naive Bayes** was extremely fast due to its simplicity, making it suitable for quick predictions.
- **Decision Tree** was slower, but still acceptable.
- **MLP**, though more accurate, was the slowest due to training complexity and deep layers.

Discussion

Strengths and weaknesses

As we saw in the comparison above, the MLP classifier overperformed the other two models in everything except for the time, it can be accepted since most of the time is for training and not testing.

Interesting observations

While working on the project we did some changes and performed hyperparameter tuning on some values and observed interesting results and changes

- At first, we had a smaller data set when we changed it to a larger one, we noticed a huge difference in the performance and that was actually expected.
- We had another data set classes like cars and the models mismatched it a lot with animals which is really interesting.
- We performed hyperparameter tuning on MLP classifier we add neurons to the layers from 128,64 to 256,128, we noticed a huge improvement! However, when we tried to make it even better by adding one more layer, we noticed that the accuracy is really high but the confusion matrix and other values are so bad which is a sign of Overfitting! So, we removed that additional layer

Also, the models seems to miss match the flowers with animals more than any other classes and we think that this is caused by the similarities in the colors (in nature)

Conclusion

In this project, we tested and compared three models: Naive Bayes, Decision Tree, and MLPClassifier. Each one had its own strengths. Naive Bayes was the fastest, Decision Tree was easier to understand, and the MLPClassifier gave the best results overall.

The MLP model showed better performance in terms of accuracy and how well it handled the images. Even though it took more time to train, it was more reliable, especially with difficult cases.

From this comparison, we learned that simpler models can still work well when time and resources are limited, but if we want better accuracy, more advanced models like MLP are a better choice. We also saw how tuning the model can make a big difference in the results.