Faculty of Engineering & Technology

Electrical & Computer Engineering Department

ENCS3340, ARTIFICIAL INTELLIGENCE

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

Prepared by:

Rand Saleh 1221124 & Roa Makhtoob 1221636

Section: 4 **Section:** 1

Instructor:

Dr. Yazan Abu Farha

Date: 7.June.2025

Abstract:

This Project presents a comparative analysis of three machine learning models applied to image classification: Naive Bayes, Decision Tree, and a Multilayer Perceptron (MLP) neural network. The dataset consists of 5,336 RGB images categorized into three classes: glacier, forest, and mountain. All images were resized to 32×32 pixels to reduce dimensionality and standardize input.

Each model was trained on feature sets suited to its architecture. Naive Bayes utilized raw pixel values as a baseline, while the Decision Tree model employed Histogram of Oriented Gradients (HOG) for shape-based feature extraction. The MLP classifier used HOG features combined with Principal Component Analysis (PCA) to enhance training efficiency and reduce noise.

Results show a clear performance progression: Naive Bayes achieved 77.6% accuracy, the Decision Tree reached 77.9%, and the MLP outperformed both with 85.1% accuracy. These findings emphasize the importance of feature engineering and model complexity in visual classification tasks. The study concludes that while traditional classifiers can yield reasonable results, neural networks—particularly when combined with dimensionality reduction—offer superior performance and generalization for image classification.

Table of Contents

Table of Figures

Introduction

Image classification is a key task in computer vision. It involves identifying and labeling the contents of images, and it’s widely used in areas like self-driving cars, environmental monitoring, healthcare, and security.

While deep learning models such as convolutional neural networks (CNNs) have greatly improved image classification, traditional machine learning methods are still valuable—especially when simplicity, transparency, or limited hardware resources are important.

This project compares three different models:

- **Naive Bayes**, which uses basic probability and assumes features are independent.
- **Decision Trees**, which sort data by following decision rules based on feature importance.
- **Multilayer Perceptrons (MLPs)**, a type of neural network that can learn more complex patterns.

Each model was tested with features that match its strengths: raw pixel values for Naive Bayes, Histogram of Oriented Gradients (HOG) for Decision Trees, and HOG with Principal Component Analysis (PCA) for MLPs. The goal was to compare how well each model performs in terms of accuracy, speed, and flexibility, and to understand when simple models are good enough and when more advanced ones are worth using.

Dataset Description

This project uses a subset of the Intel Image Classification Dataset, originally published on Kaggle by Puneet Chawla. The full dataset contains images from six categories of natural scenes: buildings, forest, glacier, mountain, sea, and street.

For this study, three categories— **glacier**, **forest**, and **mountain** —were selected. This smaller, balanced subset was chosen to simplify analysis and allow meaningful comparison across models. The resulting dataset included **6,736 images**, with approximately 2,200 to 2,300 images per class.

Preprocessing Steps

- **Image Resizing**: All images were resized to **32×32 pixels** to standardize input dimensions and reduce computational cost.
- **Grayscale Conversion**: Images were converted to grayscale when needed for feature extraction (particularly for HOG-based methods).
- **Data Splitting**: The dataset was divided into **80% training** and **20% testing** using stratified sampling to preserve class balance across both sets.

Feature Preparation Overview

To align with the capabilities of each model, different feature representations were applied:

- **Naive Bayes** used raw pixel values (flattened to 1D arrays).
- **Decision Tree** used HOG (Histogram of Oriented Gradients) features extracted from grayscale images.
- **MLP** used the same HOG features, further reduced via **Principal Component Analysis (PCA)** to 100 components.

More detail on how these features were applied is provided in the following sections on model implementation.

Detailed Explanation of Each Model

1. Naive Bayes Classifier

Naive Bayes is a basic classification algorithm that uses probability to make predictions. It’s based on Bayes’ Theorem and assumes that all features are independent from each other. This means it treats each input value separately, which makes the model fast and simple.

There are different types of Naive Bayes, and one common version assumes the input features follow a normal (Gaussian) distribution. The model works well when the features are not too dependent on each other, but it can struggle when there are strong connections between them—like in image or visual data where pixels are related.

It's often used as a quick, first model to test on a classification problem.

In our code:
We applied Naive Bayes to **raw pixel values** of the images. Each image was resized to 32×32 and flattened into a one-dimensional array. We didn't use any feature extraction or scaling here, because we wanted to keep this model simple. This helped us see how a basic algorithm behaves without advanced features.

2. Decision Tree Classifier

A Decision Tree is a model that makes decisions by asking a series of "yes or no" questions about the features. It splits the data into branches based on values, and at the end of each branch, it gives a prediction.

The tree chooses the best questions by checking how well each split separates the data. This is usually done by measuring things like information gain or impurity. The model keeps splitting the data until it meets a stopping point, like reaching a maximum depth or a minimum number of samples.

In our code:
We used **HOG (Histogram of Oriented Gradients)** features to give the tree better information about shapes and edges in the images. Each image was first converted to grayscale, and then HOG was applied to extract useful texture and edge details. We used criterion="entropy" and class_weight="balanced" to improve learning and fairness between classes. We also scaled the data before training.

To better understand how the decision tree works, we also visualized the tree structure with a maximum depth of 3. This allowed us to observe how early splits in the data influence the classification process.

Decision Trees are easy to understand and explain. But they can sometimes fit the training data too closely, which is called overfitting. They work best when given good, clean features that highlight patterns.

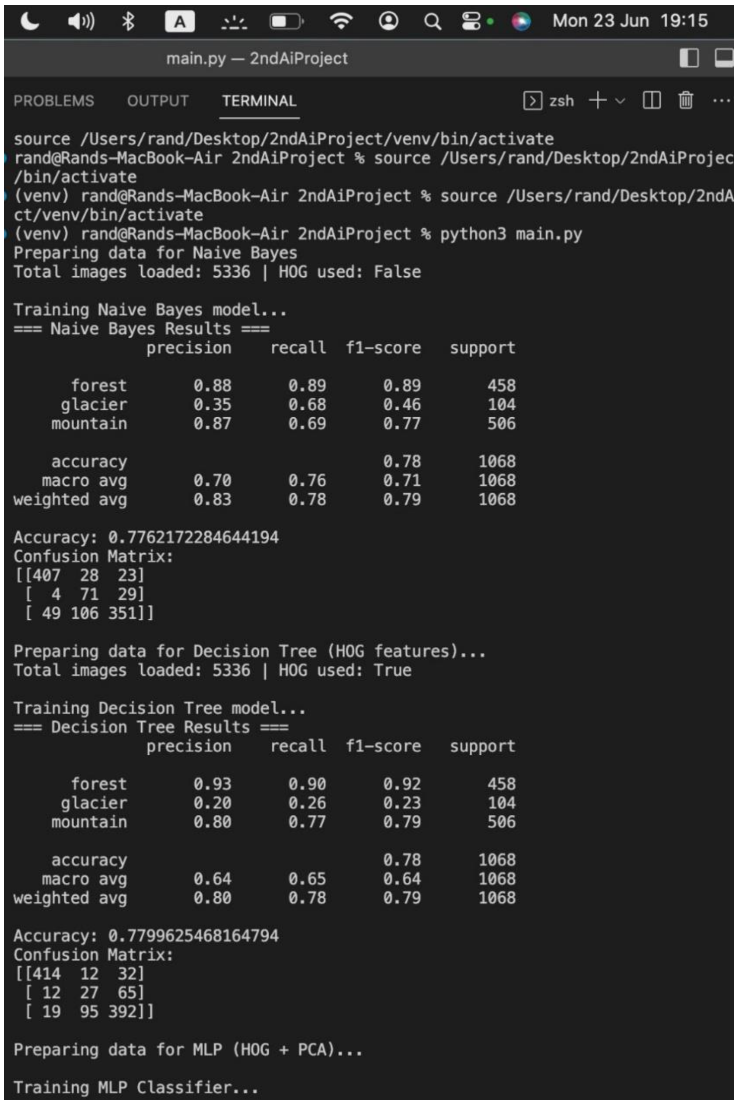
3. Multilayer Perceptron (MLP)

An MLP is a type of neural network made up of layers of connected nodes (neurons). It can learn more complex relationships in the data compared to the previous two models. It uses activation functions like ReLU to help capture non-linear patterns, and it learns by adjusting weights through a method called backpropagation.

MLPs are often used for tasks where the data has structure or patterns that simpler models can't easily detect. They do require more training time and need the input data to be properly scaled, but they are flexible and powerful for many types of classification problems.

In our code :
We first extracted **HOG (Histogram of Oriented Gradients)** features from grayscale images. These features capture important shape and edge information. To reduce the number of input features and avoid overfitting, we then applied **PCA (Principal Component Analysis)** to keep only the top 100 most important features. After scaling the data, we trained an MLP with **two hidden layers**, containing **256 and 128 neurons**, and used the **ReLU** activation function to allow the model to learn non-linear relationships. The model was optimized using the **Adam optimizer**, and **early stopping** was applied to prevent over-training.

Model Evaluation: Accuracy, Precision, Recall, F1-Score, and Confusion Matrix Analysis



```
source /Users/rand/Desktop/2ndAiProject/venv/bin/activate
rand@Rands-MacBook-Air 2ndAiProject % source /Users/rand/Desktop/2ndAiProjec
/bin/activate
(venv) rand@Rands-MacBook-Air 2ndAiProject % source /Users/rand/Desktop/2ndA
ct/venv/bin/activate
(venv) rand@Rands-MacBook-Air 2ndAiProject % python3 main.py
Preparing data for Naive Bayes
Total images loaded: 5336 | HOG used: False

Training Naive Bayes model...
=== Naive Bayes Results ===
              precision    recall  f1-score   support

      forest       0.88      0.89      0.89       458
     glacier       0.35      0.68      0.46       104
    mountain       0.87      0.69      0.77       506

    accuracy                           0.78      1068
   macro avg       0.70      0.76      0.71      1068
weighted avg       0.83      0.78      0.79      1068

Accuracy: 0.7762172284644194
Confusion Matrix:
[[407  28  23]
 [  4  71  29]
 [ 49 106 351]]

Preparing data for Decision Tree (HOG features)...
Total images loaded: 5336 | HOG used: True

Training Decision Tree model...
=== Decision Tree Results ===
              precision    recall  f1-score   support

      forest       0.93      0.90      0.92       458
     glacier       0.20      0.26      0.23       104
    mountain       0.80      0.77      0.79       506

    accuracy                           0.78      1068
   macro avg       0.64      0.65      0.64      1068
weighted avg       0.80      0.78      0.79      1068

Accuracy: 0.7799625468164794
Confusion Matrix:
[[414  12  32]
 [ 12  27  65]
 [ 19  95 392]]

Preparing data for MLP (HOG + PCA)...

Training MLP Classifier...
```

Figure 1

To evaluate the performance of the three image classification models—**Naive Bayes, Decision Tree, and MLP Classifier**—we used standard metrics such as accuracy, precision, recall, F1-score, and confusion matrices.

These metrics help us understand both the overall performance and how well each model distinguishes between the three image categories: **glacier, forest, and mountains**.

1. Naive Bayes Classifier (Figure 1 – Confusion Matrix)

Using raw pixel features, the Naive Bayes classifier achieved an accuracy of **77.62%.** As shown in **Figure 1**, the confusion matrix indicates that while the model performs well for "**forest**" and "**mountain**" images, it struggles with "**glacier**" images, misclassifying many of them as mountains. The **recall** for the glacier class was **0.68**, showing moderate detection performance, but the **precision** was low (**0.35),** indicating many false positives. This model is fast and simple, but the lack of advanced features leads to poor differentiation between visually similar classes.

2. Decision Tree with HOG Features (Figure 2– Confusion Matrix)

Incorporating **Histogram of Oriented Gradients (HOG)** improved performance. As shown in **Figure 1**, the Decision Tree model achieved an accuracy of **77.99%**. This is only slightly higher than the Naive Bayes classifier, which reached **77.62%**, with a margin of just **0.37%**. The reason for this small improvement is that while HOG features do enhance structural understanding, the Decision Tree model itself is still relatively shallow and prone to overfitting, especially when dealing with imbalanced classes like “glacier.”

The **confusion matrix** shows balanced classification for "**forest**" and "**mountain**" with high recall and precision. However, the model still struggles with "**glacier**" class, showing a very low recall (**0.26**) and precision (**0.20**). HOG features help the model focus on edge orientations and object shapes rather than raw pixel values, making it more robust in capturing visual structure, though the model’s limitations in handling minority classes are still apparent.

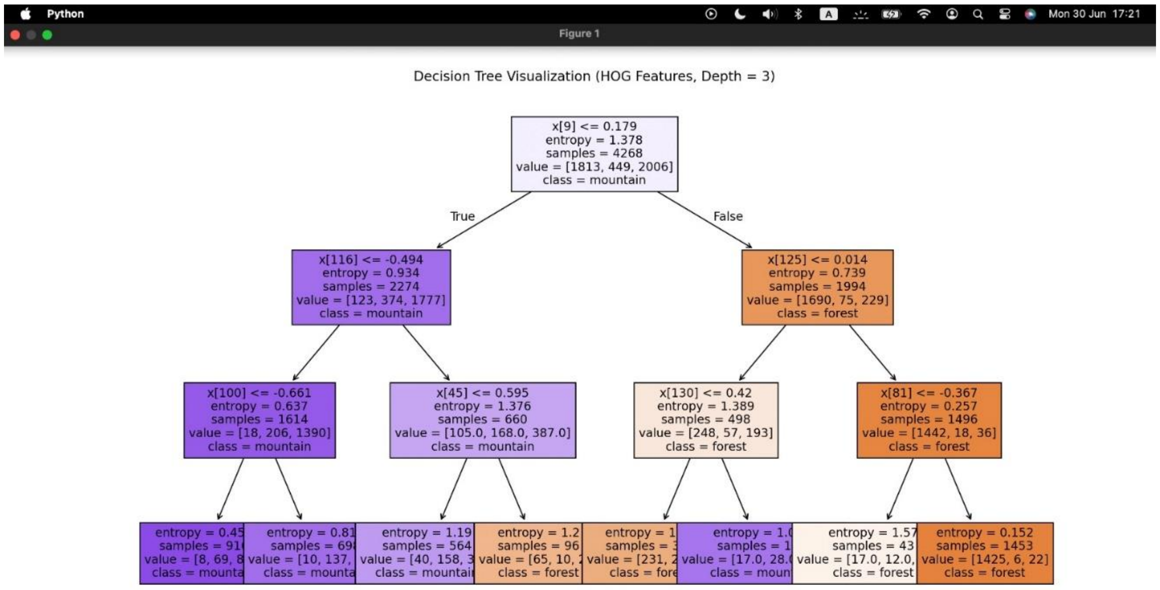
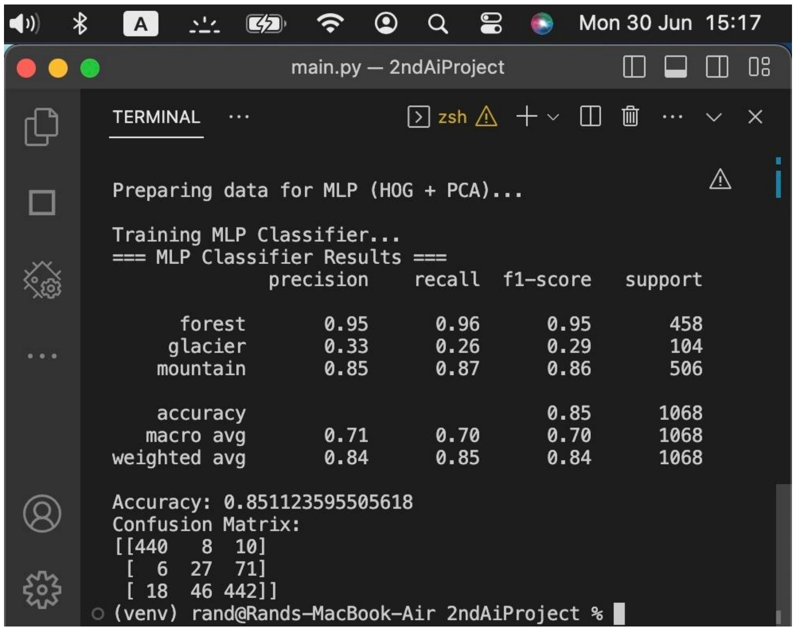


Figure 2

The decision tree shown in **Figure 2**, limited to a depth of 3, offers a simplified view of how the model uses HOG features to classify images into forest, mountain, and glacier. Most of the samples near the top of the tree are predicted as **mountain**, suggesting that this class is the easiest for the model to recognize. On the right side, the tree identifies **forest** images with high confidence, especially in nodes with low entropy and a large number of correctly classified samples. However, **glacier** images are barely represented, indicating the model struggles to separate them at this shallow depth.



```
Preparing data for MLP (HOG + PCA)...

Training MLP Classifier...
=== MLP Classifier Results ===
              precision    recall  f1-score   support

      forest       0.95      0.96      0.95       458
     glacier       0.33      0.26      0.29       104
    mountain       0.85      0.87      0.86       506

    accuracy                           0.85      1068
   macro avg       0.71      0.70      0.70      1068
weighted avg       0.84      0.85      0.84      1068

Accuracy: 0.851123595505618
Confusion Matrix:
[[440   8  10]
 [  6  27  71]
 [ 18  46 442]]
(venv) rand@Rands-MacBook-Air 2ndAiProject %
```

Figure 3

3. MLP Classifier with HOG + PCA (Figure 3– Confusion Matrix)

The best performance was achieved by the **MLP Classifier**, trained on **HOG** features reduced via PCA. As shown in **Figure 3**, the model reached an accuracy of **85.11%**, the highest among the three.

The confusion matrix reveals very few misclassifications in the "**forest**" and "**mountain**" categories (precision and recall above **0.85**), while the "**glacier**" class remains challenging, with low precision (**0.33**) and recall (**0.26**).

However, the overall macro F1-score (**0.70**) and weighted F1-score (**0.84**) reflect strong general performance. **PCA** helped reduce noise and dimensionality while preserving important information, making the neural network more efficient and accurate in learning patterns.

.

Comparative Analysis and Discussion of Best Performing Model

Among the three evaluated models, the MLP Classifier with HOG and PCA emerged as the most accurate and reliable for the image classification task. This superior performance can be explained by several technical and architectural factors:

1. Advanced Feature Engineering

Unlike raw pixels**, HOG** features describe the structure of the image by focusing on gradient orientations and edge patterns—information that's highly relevant for visual recognition tasks.

When paired with PCA, which reduces dimensionality while retaining the most significant features, the model becomes both faster and more effective.

This combination preserves meaningful patterns while eliminating noise, making learning more efficient and less prone to overfitting.

2. Deep Model Architecture

The **MLP** uses multiple fully connected layers to learn hierarchical representations of the input data. It can model non-linear decision boundaries, allowing it to distinguish between subtle differences in complex visual classes like **"forest" vs. "mountain" vs. "glacier"**.

In contrast, **Naive Bayes** assumes feature independence, and **Decision Trees** make linear decisions at each node, limiting their ability to capture abstract relationships. The MLP's hidden layers give it the flexibility to learn more intricate patterns in the data.

3. Superior Generalization

The MLP demonstrated not only high training performance but also strong generalization to unseen test data, as reflected in its consistently high precision, recall, and F1-scores across dominant classes like **forest** and **mountain**.

While it struggled with **glacier**, it still outperformed the other models in overall consistency.

This means that the model is not simply memorizing the training data, but actually learning general rules for classification—a critical requirement for real-world applications.

4. **Error Reduction and Stability**

Compared to the confusion matrices of Naive Bayes and Decision Tree models, the MLP’s matrix (Figure 3) showed fewer off-diagonal values, meaning **fewer misclassifications and better class separation**. This directly translates to a more stable and trustworthy model when deployed.

References

[1] https://www.augmentedstartups.com/blog/the-impact-of-object-detection-in-artificial-intelligence-and-computer-vision?srsltid=AfmBOooDVQG27oAef__TOanRaWzKHH3WHqC0aAgA5FP3DfcttnOQOkQn

[2] https://en.wikipedia.org/wiki/Naive_Bayes_classifier

[3]https://www.researchgate.net/publication/340554612_A_Comparative_Analysis_of_Classifiers_for_Image_Classification

[4]https://www.reddit.com/r/learnmachinelearning/comments/1917mti/what_do_you_believe_is_the_reason_decision_trees/

[5] https://www.geeksforgeeks.org/machine-learning/naive-bayes-classifiers/

[6] https://datascience.stackexchange.com/questions/38328/when-does-decision-tree-perform-better-than-the-neural-network

[7] https://medium.com/@navarai/decision-trees-vs-neural-networks-ff46f47ce0a0