Ganforge Assignment 3

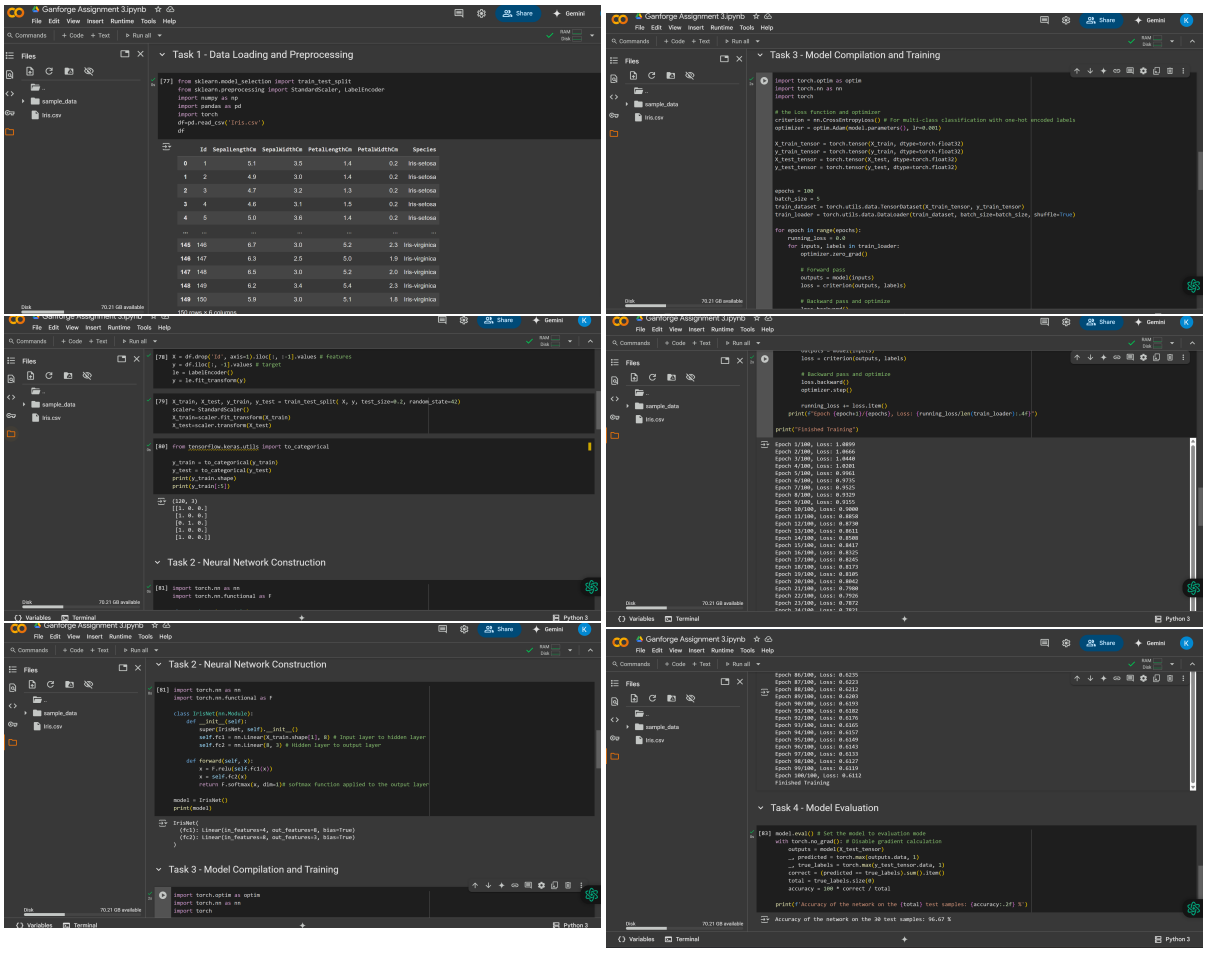
Kartikey Agarwal

June 2025

Insights

Task 1

- The Iris dataset contains 150 samples, each representing a flower from one of three species: Iris-setosa, Iris-versicolor, and Iris-virginica under four continuous features: sepal length, sepal width, petal length, and petal width.
- The dataset is clean and well-curated, with no missing values. Each feature column contains 150 non-null values, and the data types are appropriate for analysis (float for features, string/categorical for species)
- The dataset is split into training and test sets (80% training, 20% testing) as given in the problem statement and then normalization is done using `StandardScaler` to standardize the features. This step makes sure all input features have similar scales, which helps the neural network learn better.
- **One-Hot Encoding:** Then I converted the target labels into a one-hot encoded format using `to_categorical`. This is needed for multi-class classification with neural networks.
- The model has:
  - An input layer for the 4 features,
  - One hidden layer with 8 neurons and ReLU activation,
  - An output layer with 3 neurons and softmax activation for predicting the species.
- The model uses a **softmax** activation function in the last layer to help the network decide which flower species is most likely by giving a probability for each class. Then the model was finally compiled and evaluated using Adam Optimizer and Cross Entropy Loss.



Task 2

Overview

In this task, I implemented an image similarity search tool using a pre-trained deep learning model (ResNet18) and Spotify's Annoy library for approximate nearest neighbor search. The system compares dog and cat images based on visual similarity by extracting feature vectors and indexing them efficiently.

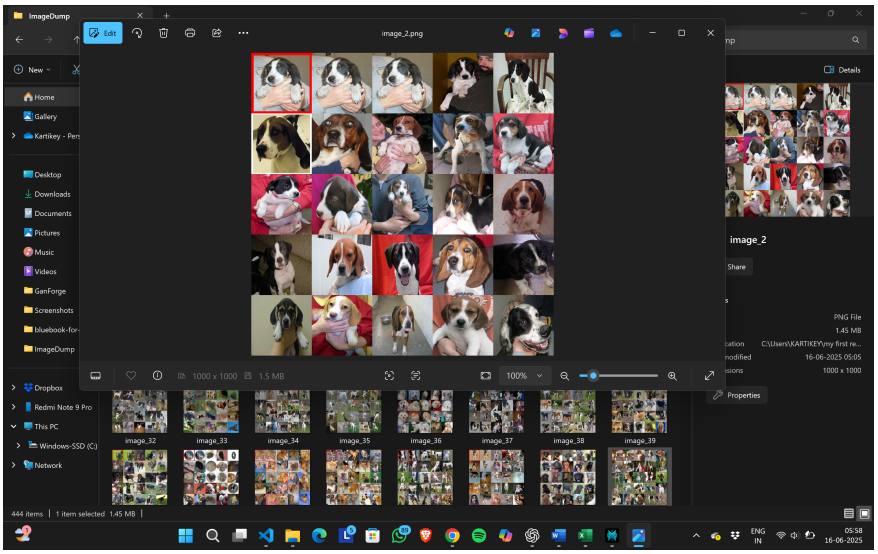
How Feature Detection Works

Feature detection means finding the important parts or patterns in an image that help tell it apart from other images. I used a model called ResNet18, which has already been trained to understand many kinds of images. Instead of comparing images pixel by pixel, the model looks for patterns—like edges,

shapes, or textures—that stay similar even when the image changes a little. The model works through layers, where the early layers detect simple things like lines or corners, and the deeper layers learn more detailed parts like a dog's face or body shape. I removed the last part of the model that normally gives a label, and instead took the values from the penultimate layer. These values are stored in a list of 512 numbers, called a feature vector, which describes what the image looks like in a smart way. If two images are similar, their feature vectors will also be similar, and that helps us find images that look alike.

Annoy for Image Similarity Search

The `Annoy` (Approximate Nearest Neighbors Oh Yeah) library builds a tree-based index to perform fast similarity searches over high-dimensional vectors. We use Annoy to find the top 24 similar images to a given query image using angular distance as the similarity metric.



Github Repo: https://github.com/Kari3020/GanForge