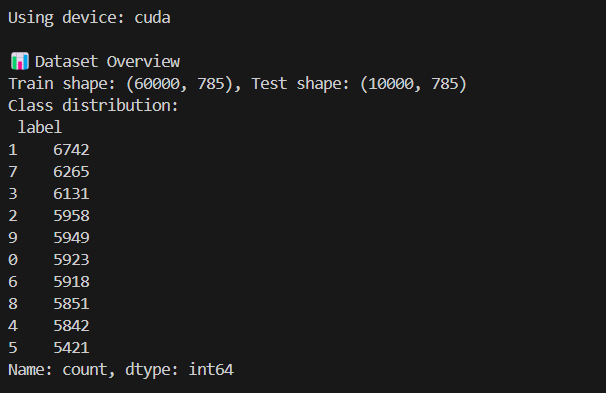
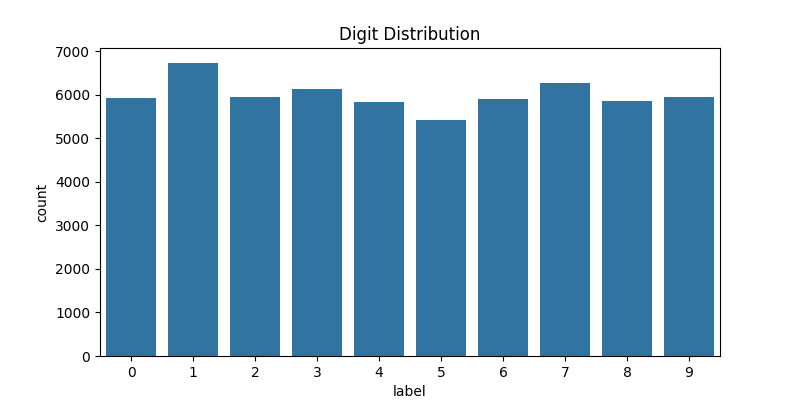
**MNIST CODE OUTPUTS**

I am using cuda for parallal processing in python

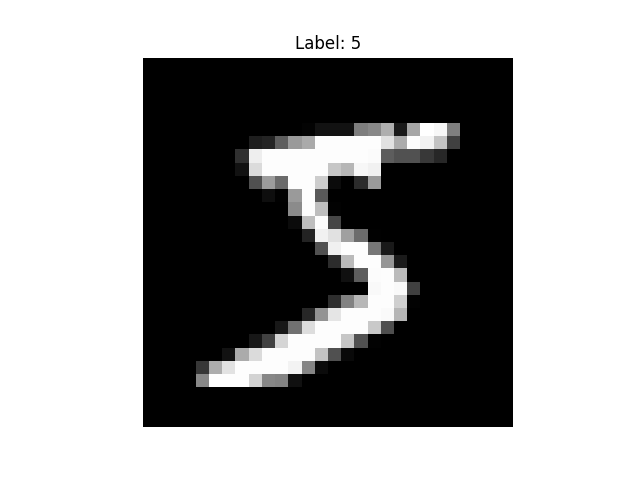


**EDA**

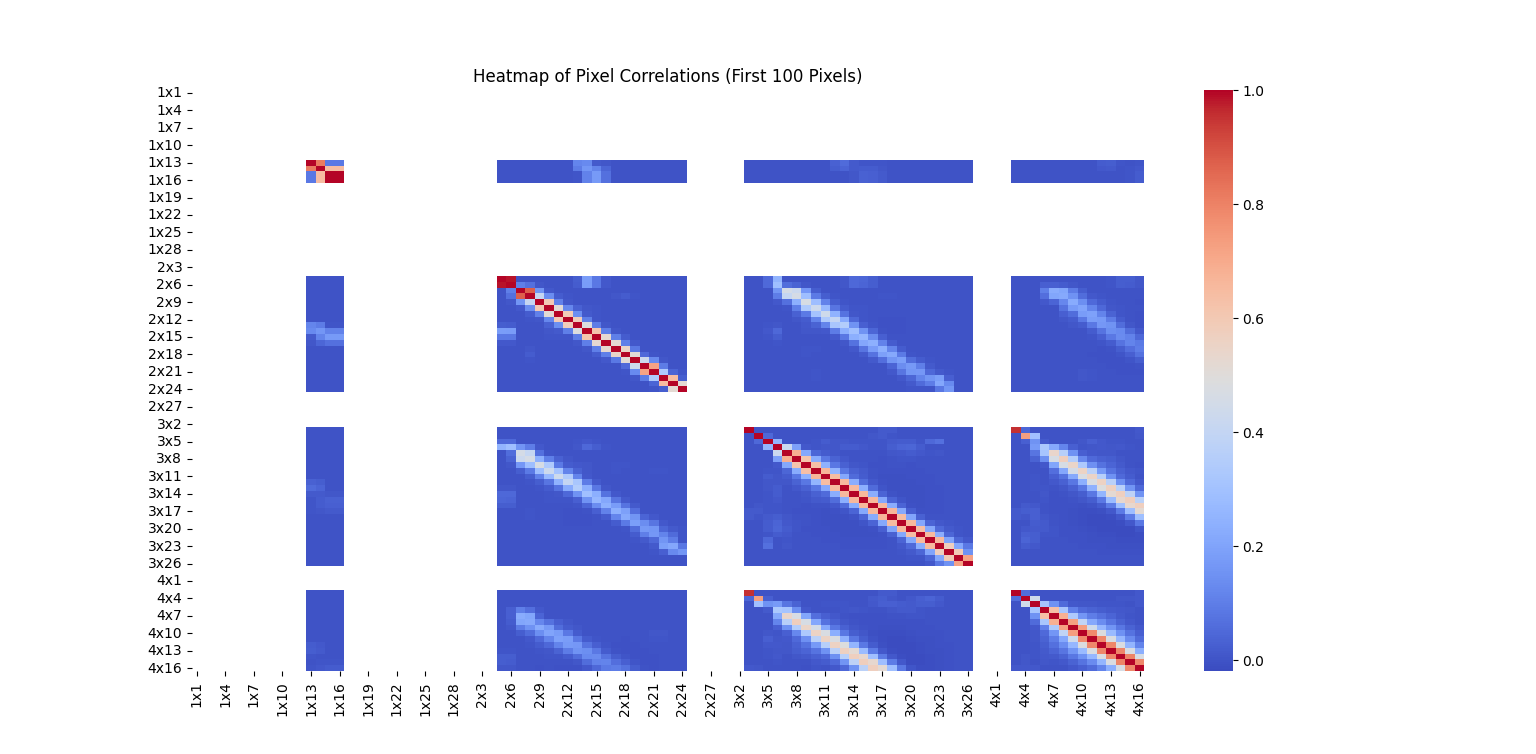
DIGIT DISTRIBUTION AS BAR GRAPH ,DIGITS AND THEIR FREQUENCY ARE GIVEN



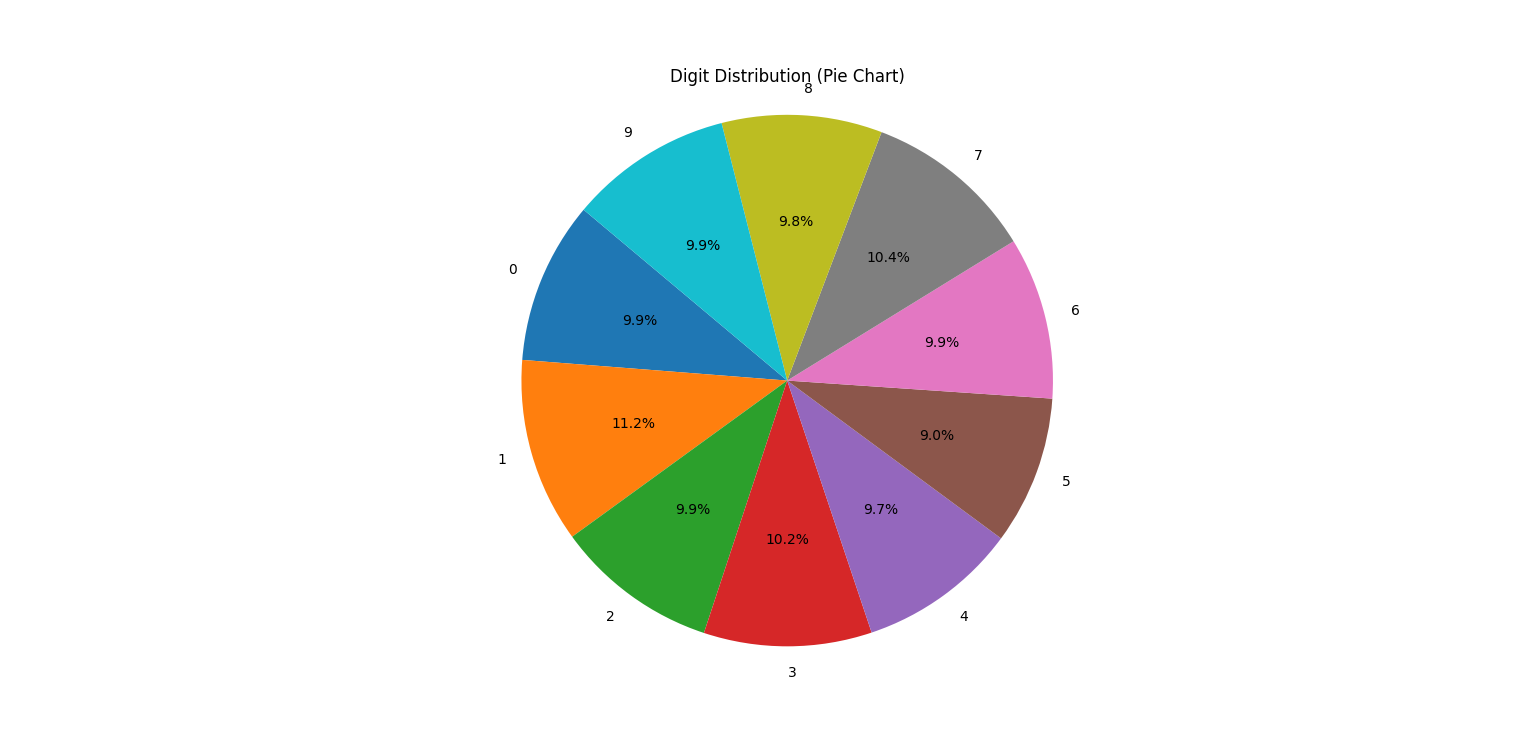
SAMPLE ELEMENT/OBJECT IN DATASET IS GIVEN BELOW



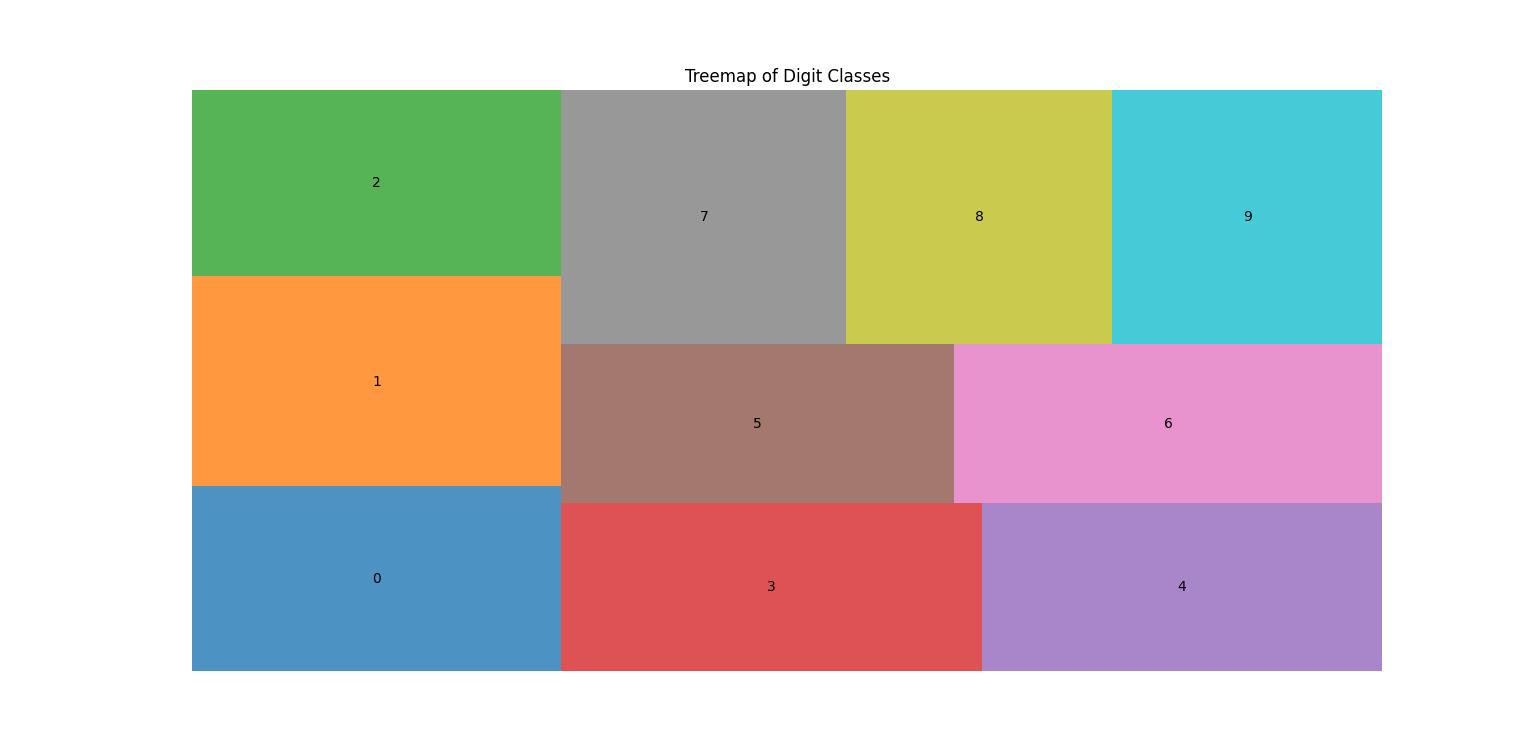
HEATMAP OF PIXEL CORRELATION IS GIVEN



Digit distribution as pie graph

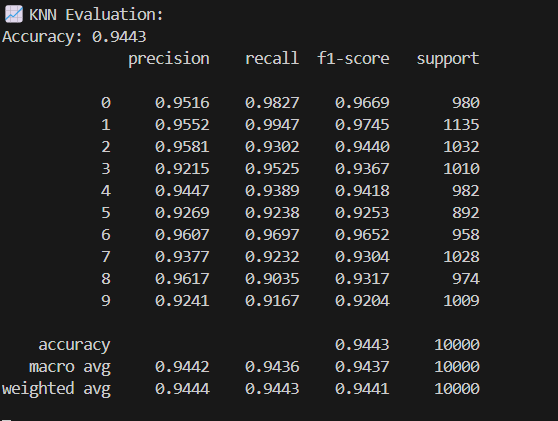


TREE MAP OF DIGIT CLASSES

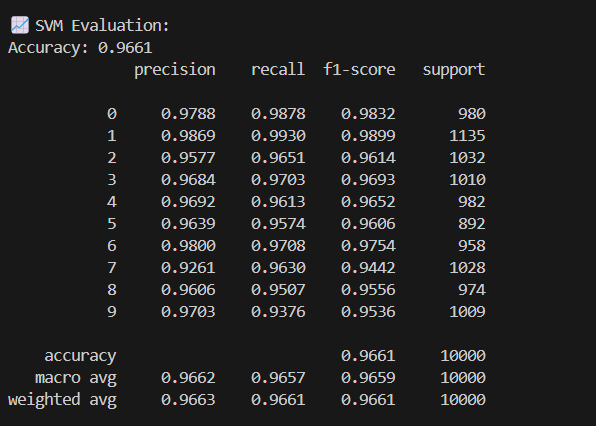


ACCURACY OF ML MODELS

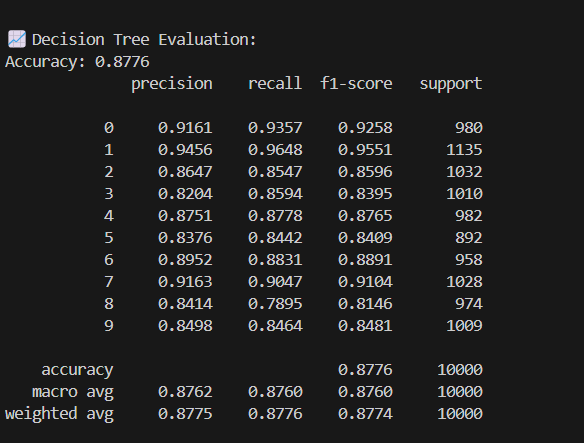
KNN



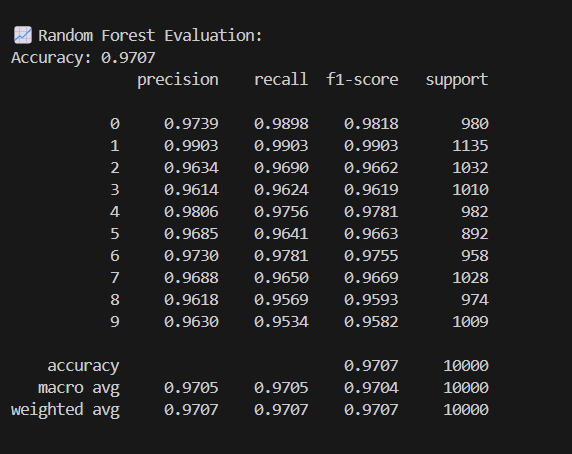
SVMS



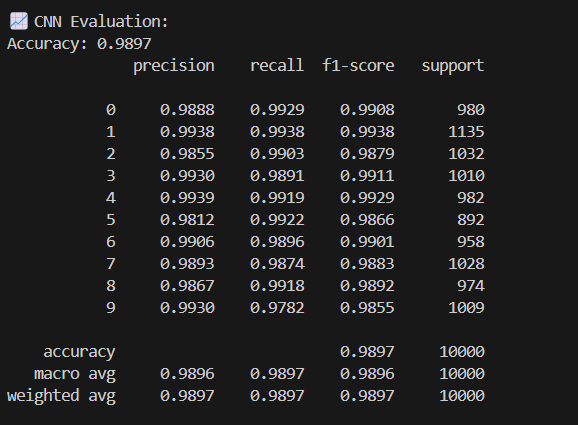
DECISION TREES



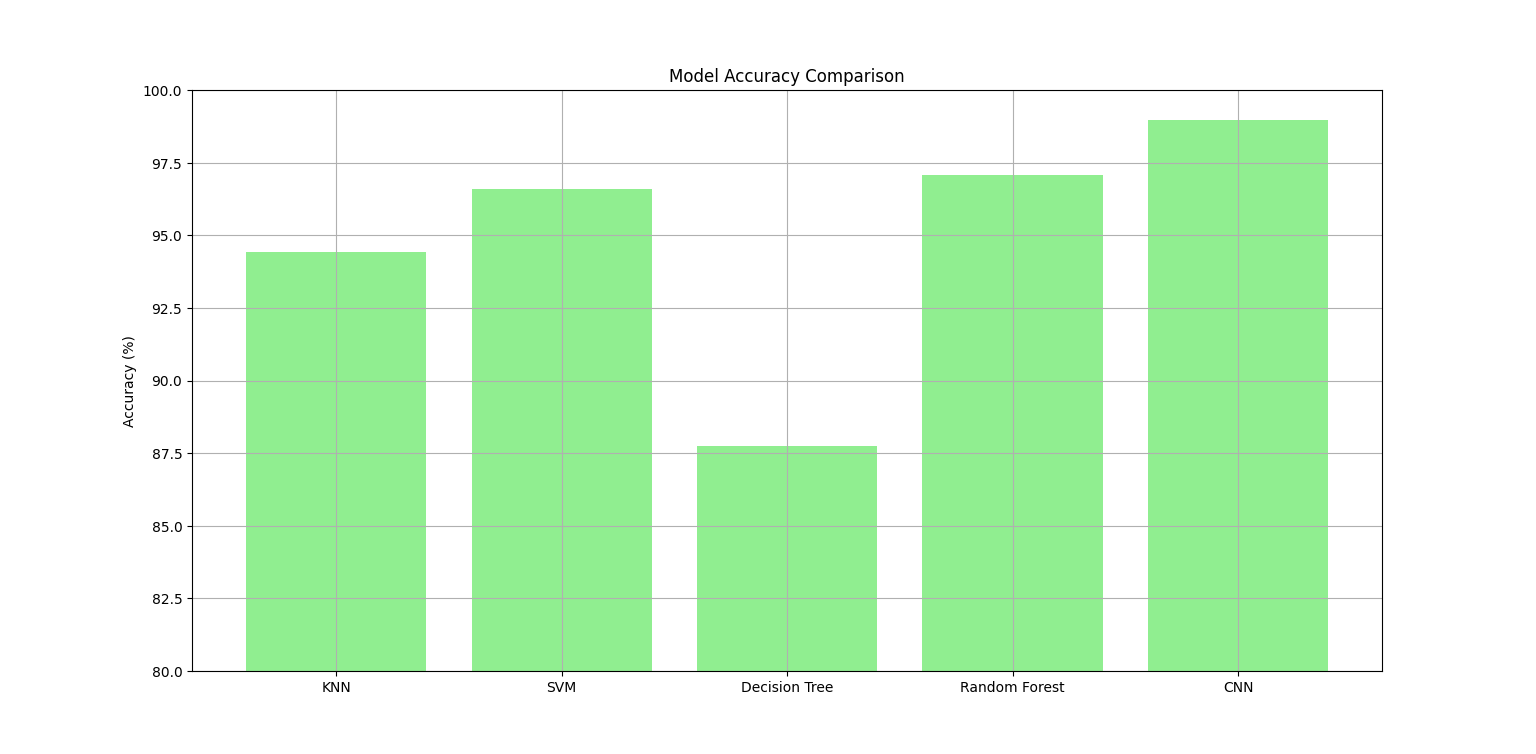
RANDOM FOREST



CNN MODEL



OVERALL COMPARISION OF ACCURACY



| **Model** | **Accuracy (%)** | **Summary** |
| --- | --- | --- |
| **KNN** | 94.43% | Good baseline. Fast to implement but slower on large datasets. Struggles slightly with similar digits. |
| **SVM** | 96.61% | Strong performer. Handles complex data well but slower to train. High precision and recall overall. |
| **Decision Tree** | 87.76% | Fast and interpretable but overfits and underperforms. Struggles with confusing digits. |
| **Random Forest** | 97.07% | High accuracy. Combats overfitting. Very consistent across all digits. Great traditional ML model. |
| **CNN (PyTorch)** | **98.97%** ✅ | Best performance. Learns image features automatically. Ideal for image tasks like MNIST. |

CNN outperforms all traditional models on MNIST with **almost 99% accuracy**, making it the best choice for handwritten digit recognition.