Python for Computer Science and Data Science 2 (CSE 3652)

**Major Assignment-3: MACHINE LEARNING- CLASSIFICATION, REGRESSION AND CLUSTERING**

**Overview**

Imagine you’ve been hired by *FashionX*, a fast-growing online fashion retailer struggling to organize its expanding product catalog. The company needs an automated system to classify fashion items into categories like T-shirts, Dresses, Shoes, etc., based on images.

As a data scientist, your task is to develop a machine learning model to classify 28x28 pixel images from the Fashion *MNIST* dataset into 10 fashion categories. You’ll apply algorithms like K-Nearest Neighbors (KNN) and Support Vector Machines (SVM) for classification. Additionally, you’ll use t-SNE to visualize the data, uncovering patterns and clusters in the high-dimensional space.

This project will help FashionX scale its operations by efficiently automating product categorization, making it easier for customers to find what they’re looking for.

**Tasks**

**Task 1: Data Preprocessing**

* Download the Fashion MNIST dataset using TensorFlow. You can load it directly using the following code:

import tensorflow as tf

from tensorflow.keras.datasets import fashion\_mnist

# Load the dataset

(train\_images, train\_labels), (test\_images, test\_labels) = \

fashion\_mnist.load\_data()

* The dataset consists of 60,000 training images and 10,000 test images of fashion products, with 10 distinct categories. Each image is 28x28 pixels.
* Perform the following preprocessing steps:

1. Normalize the images (values should be between 0 and 1).
2. Flatten the 28x28 pixel images into 1D arrays (784 pixels per image).
3. Handle any missing or incorrect values (if any).

* Split the dataset into training and test sets.

**Task 2: K-Nearest Neighbors (KNN) Classification**

* Implement the K-Nearest Neighbors (KNN) algorithm using the preprocessed dataset.
* Experiment with different values of *k* (e.g., *k* = 3, *k* = 5, *k* = 7) to see how the model performance changes.
* Evaluate the model performance using accuracy on the test dataset.
* Provide a comparison of different *k* values and the impact on accuracy.

**Task 3: Support Vector Machine (SVM) Classification**

* Train a Support Vector Machine (SVM) classifier on the same preprocessed dataset.
* Experiment with different kernels (linear, polynomial, radial basis function) and hyperparameters such as C.
* Evaluate the model performance using accuracy on the test dataset.
* Compare the SVM performance with that of the KNN model.

**Task 4: Data Visualization with t-SNE**

* Use the t-SNE technique to reduce the dimensionality of the data from 784 features to 2 or 3 dimensions.
* Visualize the 2D or 3D representation of the Fashion MNIST dataset and observe the clustering of different fashion categories.
* Analyze the plot to identify how well the categories are separated, and discuss the results.

**Task 5: Model Evaluation and Reporting**

* Evaluate the performance of both the KNN and SVM models using accuracy, precision, recall, F1- score, and confusion matrix.
* Discuss which model performs better and why, based on the evaluation metrics.
* Write a report summarizing the approach used in each task, the results obtained, and insights derived from the visualizations.

**Code:**

# Task 1: Data Preprocessing

import tensorflow as tf

from sklearn.model\_selection import train\_test\_split

import numpy as np

# Load dataset

(train\_images, train\_labels), (test\_images, test\_labels) = tf.keras.datasets.fashion\_mnist.load\_data()

# Normalize

train\_images = train\_images / 255.0

test\_images = test\_images / 255.0

# Flatten

X\_train = train\_images.reshape(-1, 784)

X\_test = test\_images.reshape(-1, 784)

# Check for NaNs

assert not np.isnan(X\_train).any() and not np.isnan(X\_test).any()

# Labels are already in numeric form (0-9)

y\_train, y\_test = train\_labels, test\_labels

# Task 2: K-Nearest Neighbors (KNN) Classification

from sklearn.neighbors import KNeighborsClassifier

from sklearn.metrics import accuracy\_score

for k in [3, 5, 7]:

knn = KNeighborsClassifier(n\_neighbors=k)

knn.fit(X\_train, y\_train)

y\_pred = knn.predict(X\_test)

acc = accuracy\_score(y\_test, y\_pred)

print(f"K={k}, Accuracy={acc:.4f}")

# Task 3: Support Vector Machine (SVM) Classification

from sklearn.svm import SVC

kernels = ['linear', 'poly', 'rbf']

for kernel in kernels:

svm = SVC(kernel=kernel, C=1.0)

svm.fit(X\_train[:10000], y\_train[:10000]) # limit for performance

y\_pred = svm.predict(X\_test[:2000])

acc = accuracy\_score(y\_test[:2000], y\_pred)

print(f"SVM Kernel={kernel}, Accuracy={acc:.4f}")

# Task 4: Data Visualization with t-SNE

from sklearn.manifold import TSNE

import matplotlib.pyplot as plt

tsne = TSNE(n\_components=2, random\_state=42, perplexity=30)

X\_embedded = tsne.fit\_transform(X\_test[:1000])

y\_sample = y\_test[:1000]

plt.figure(figsize=(10, 8))

scatter = plt.scatter(X\_embedded[:, 0], X\_embedded[:, 1], c=y\_sample, cmap='tab10', s=10)

plt.title('t-SNE visualization of Fashion MNIST')

plt.colorbar(scatter, ticks=range(10))

plt.show()

# Task 5: Model Evaluation and Reporting

from sklearn.metrics import classification\_report, confusion\_matrix

# Use best KNN (e.g., K=5)

knn = KNeighborsClassifier(n\_neighbors=5)

knn.fit(X\_train, y\_train)

knn\_preds = knn.predict(X\_test)

# Use best SVM (e.g., RBF kernel)

svm = SVC(kernel='rbf')

svm.fit(X\_train[:10000], y\_train[:10000])

svm\_preds = svm.predict(X\_test[:2000])

# Evaluation

print("KNN Classification Report:")

print(classification\_report(y\_test, knn\_preds))

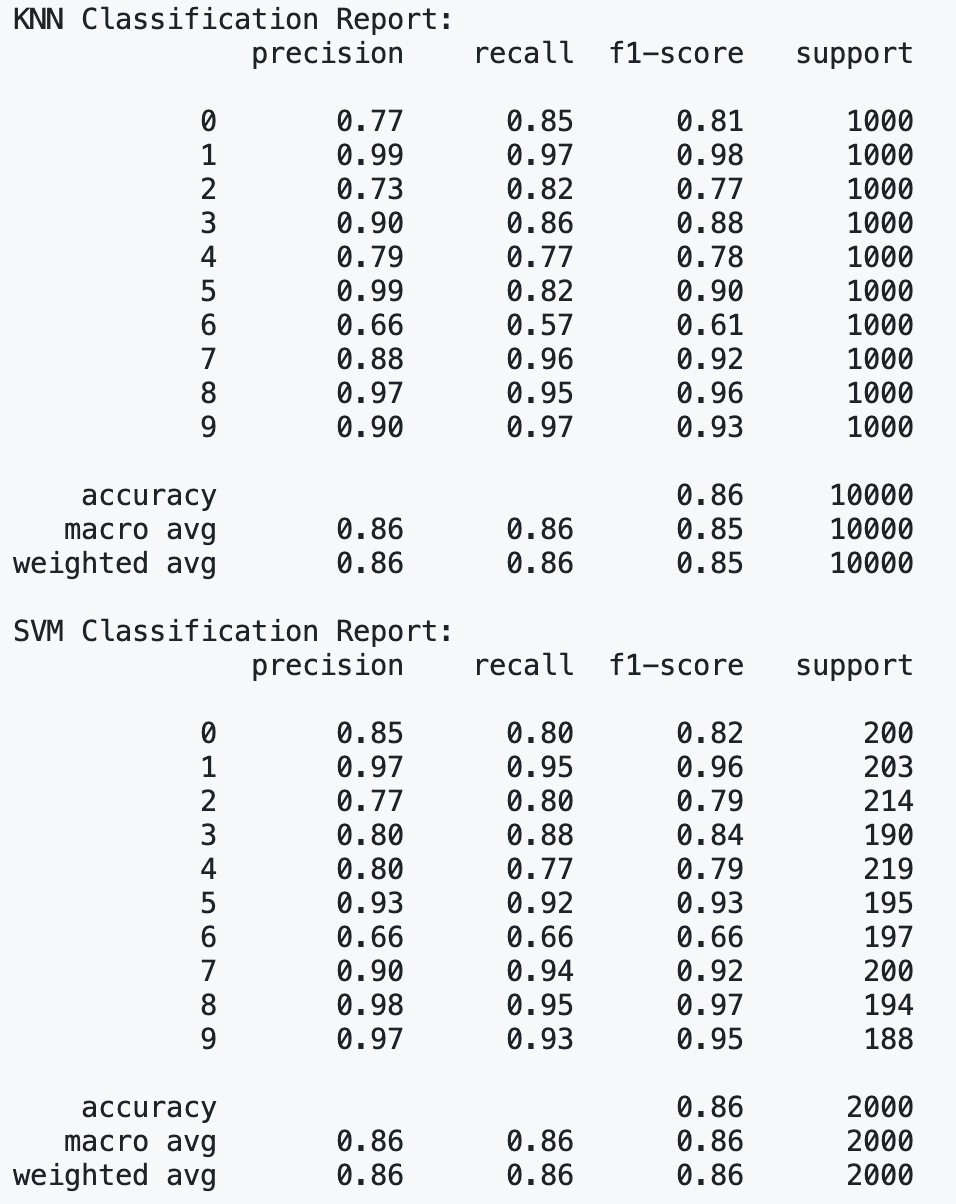
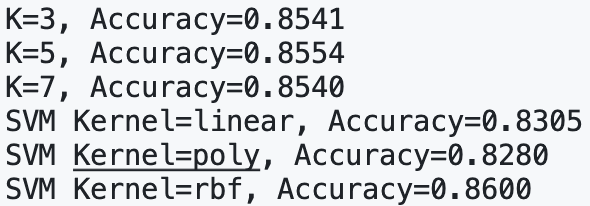
print("SVM Classification Report:")

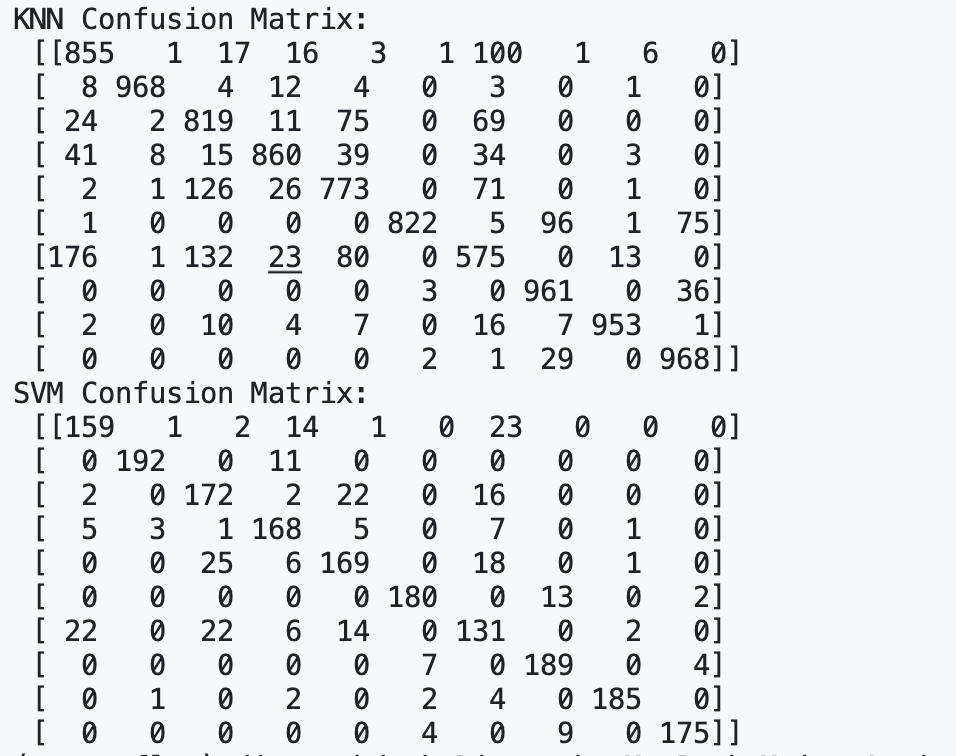
print(classification\_report(y\_test[:2000], svm\_preds))

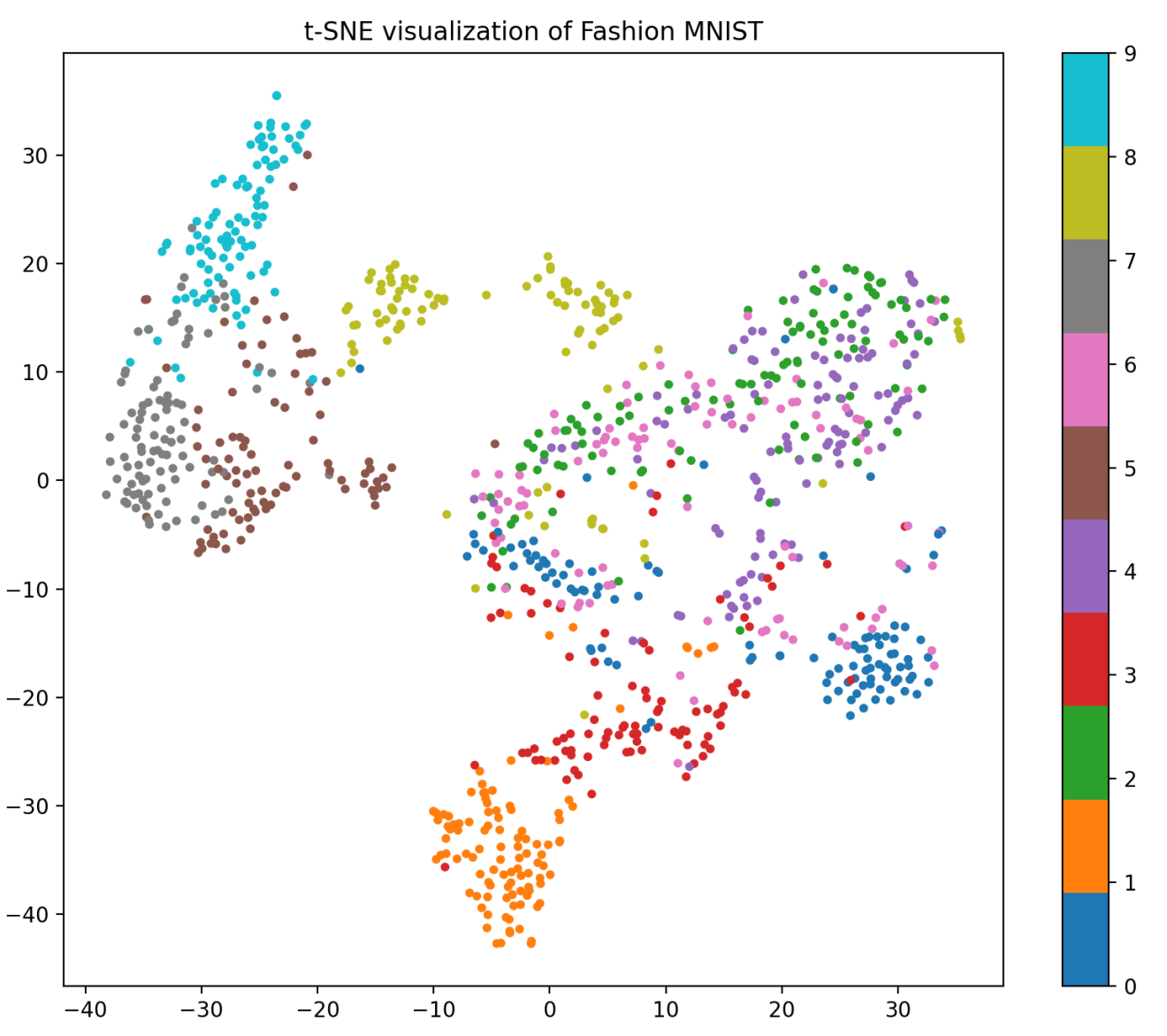
print("KNN Confusion Matrix:\n", confusion\_matrix(y\_test, knn\_preds))

print("SVM Confusion Matrix:\n", confusion\_matrix(y\_test[:2000], svm\_preds))

**Output:**

****

****

****