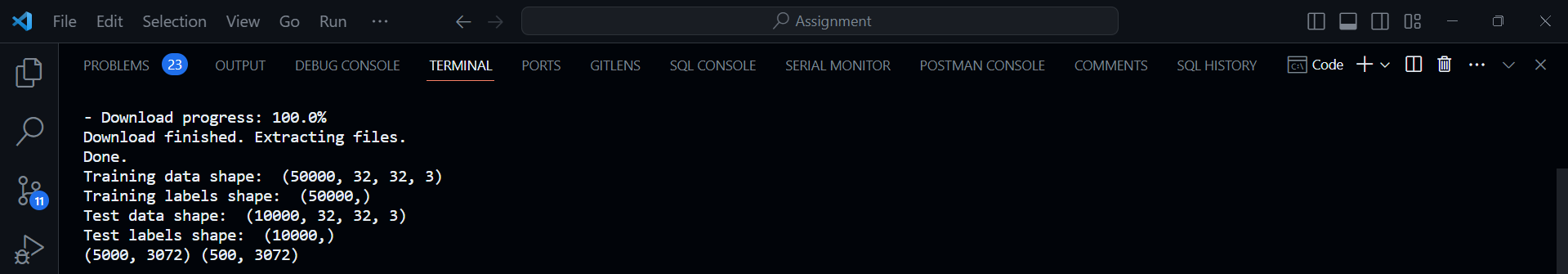
Assignment Report

Assignment 1 (KNN)

Fahim Wayez

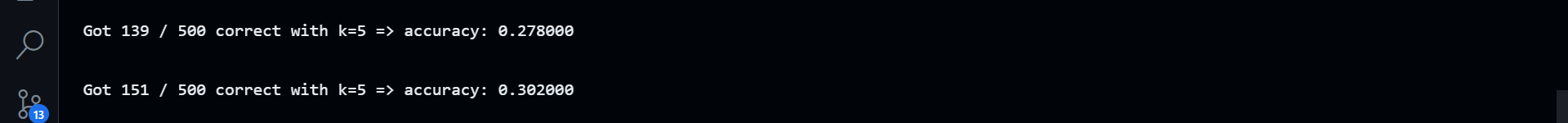
21-44499-1 | Section B

The performance of the K-Nearest Neighbors (KNN) algorithm on the CIFAR-10 dataset has been analyzed in this assignment. Upon loading the data, we observed that the training set contains 50,000 examples while the test set contains 10,000 examples (50000, 32, 32, 3 indicates that the dataset consists of 50,000 images, each of size 32x32 pixels with 3 color channels). Als0, there are indications that there are 50,000 corresponding labels for the training images and 10,000 corresponding labels for the test images. To prevent memory errors, a smaller subset of the data is chosen for training and testing, 5,000 training samples and 500 test samples. The data is flattened so that each row contains all the pixel values of an example. This results in Xtrain of shape (5000,3072) and Xtest of shape (500,3072).



Code Segment : Training, Testing, Subsampling, and Reshaping dataset

The KNearestNeighbor class is initialized and trained with the training data. The predict method is used to compute distances using both Euclidean and Manhattan metrics and predict labels for the test data using k=5. The accuracy for both metrics is printed. After initial performance of the KNN with a fixed K value of 5, the accuracy for was approximately 27.8% and 30.2% using Euclidean and Manhattan distance metrics respectively.



Code Segment : Initial accuracy with both the distance metrics

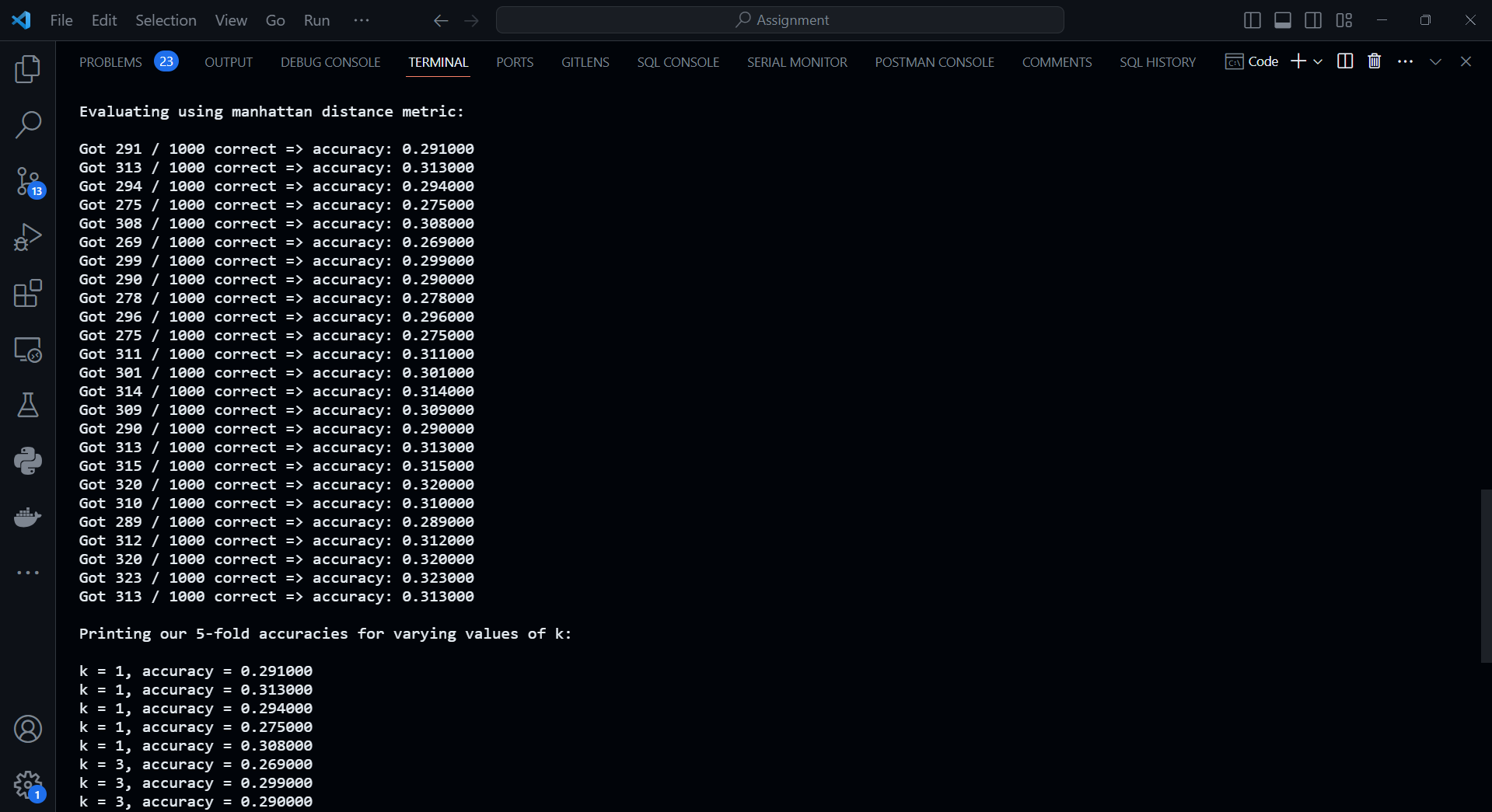
To find the optimal value of k, 5-fold Cross-Validation was performed. This involves splitting the training data into 5 folds, training the model on 4 folds, and validating it on the remaining fold. This process was repeated for different values of k (1,3,4,8,10) and recorded the accuracies. The k value with the highest average accuracy across all folds is selected as the optimal k value. Using the optimal k value obtained from Cross-Validation, the model was evaluated on the test set. For both Euclidean and Manhattan distance metrics, the Cross-Validation determines that k=10 yields the highest average accuracy. This optimal k value helps prevent overfitting (when the model performs well on the training data but poorly on the test data) and underfitting (when the model is too simple to capture the underlying structure of the data).

A screenshot of a computer

Description automatically generatedA black background with a black square

Description automatically generated with medium confidence

Code Segment : 5-fold validation using different k values with Euclidean Distance Metric

A black background with a black square

Description automatically generated with medium confidence

Code Segment : 5-fold validation using different k values with Manhattan Distance Metric

A scatter plot and error bar plot of the Cross-Validation accuracies for different k values are shown.

A screen shot of a graph

Description automatically generated

Figure : Scatter plot of 5-fold accuracy using Euclidean Distance Metric

A graph with lines and numbers

Description automatically generated

Figure : Cross-Validation plot of 5-fold accuracy using Euclidean Distance Metric

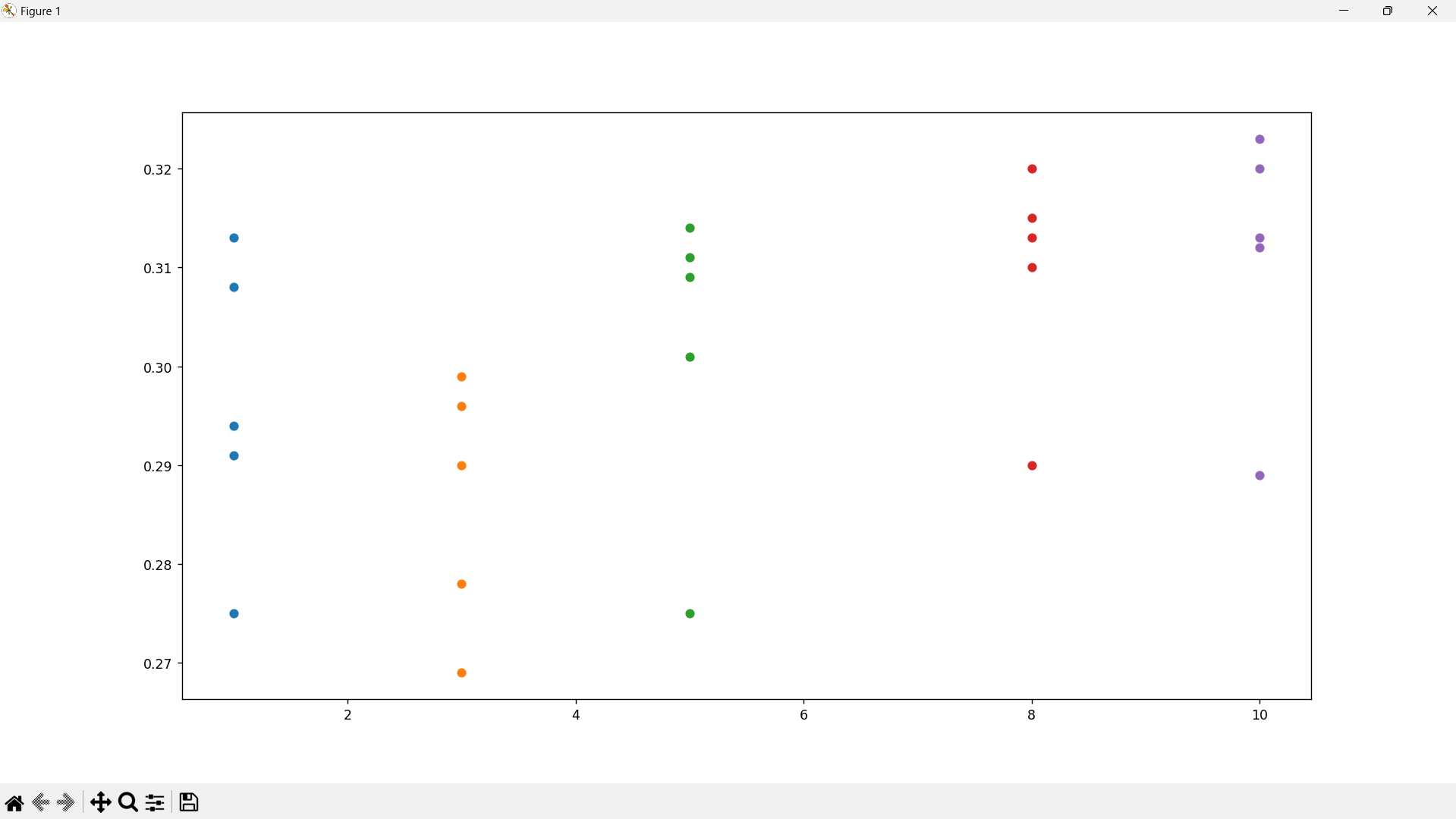


Figure : Scatter plot of 5-fold accuracy using Manhattan Distance Metric

A graph with a line

Description automatically generated

Figure : Cross-Validation plot of 5-fold accuracy using Manhattan Distance Metric

Using this optimal k value, the KNN classifier, for both the distance metrics, achieved the accuracy of approximately 56% on the test dataset. A higher accuracy indicates that the model is better able to generalize to unseen data.

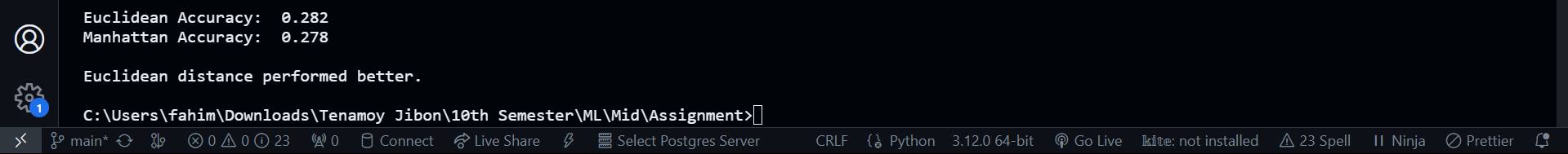


Code Segment : Accuracy using optimal k value for Euclidean Distance Metric



Code Segment : Accuracy using optimal k value for Manhattan Distance Metric

Both the accuracy scores are printed, and comparison says that for this time, Euclidean distance performed well.



Code Segment : Comparison of accuracy of the two Distance Metrics

**Discussion:** Cross-Validation is crucial for selecting the optimal hyperparameter such as k in KNN. This is necessary because choosing an appropriate k in the KNN algorithm can significantly impact the model’s performance. It helps in assessing the model’s generalization performance and prevents overfitting by evaluating on multiple subsets of the training data. It provides a more reliable estimate of the model’s performance than simply using a single train-test split. In this case, Cross-Validation allowed to find the value of k that led to the best performance on unseen data.

Having an optimal value of k improves the model’s ability to capture the underlying patterns in the data. With an appropriate k, the model balances bias and variance, leading to better fitting and generalization. A smaller k value e.g., 1 or 3 tends to overfit the training data because it is overly sensitive to noise. A larger k value e.g., 20 tends to underfit the data because it considers more neighbors, potentially leading to smoother decision boundaries but may ignore local patterns in the data.

While KNN provides a simple and interpretable approach to classification, it may not be the most suitable for complex datasets like CIFAR-10. KNN’s performance heavily relies on the choice of k, and it can be computationally expensive, especially with large datasets. More sophisticated models such as Neural Networks (NN) or Convolutional Neural Networks (CNN) often outperform KNN on image classification tasks. These models can automatically learn complex patterns and hierarchical features from raw pixel values, resulting in higher accuracies, making them more suitable for image classification tasks like CIFAR-10. In this case, the KNN classifier achieves a respectable accuracy of around 56% on the CIFAR-10 dataset, while with proper hyperparameter tuning and architecture design, NNs or CNNs could have potentially achieved higher accuracies. NNs and CNNs are scalable and can leverage GPUs for faster training, making them more practical for large-scale datasets. CNN, in particular, have been shown to achieve state-of-the-art performance on CIFAR-10 and similar datasets, often surpassing the accuracy of traditional machine learning like KNN. Additionally, deep learning models like CNNs can automatically learn features from raw pixel data, whereas KNN relies on handcrafted features representations and may struggle with high-dimensional data. However, it is worth nothing that CNNs typically require more computational resources and longer training times compared to KNN, which can be a consideration depending on the available resources and the specific requirements of the application.