**Machine Learning – CSE 6363**

**Assignment 1 Report**

Documented by: Manan Arora (1002143328)

**Introduction:**

In this assignment, I have implemented a KNN algorithm from scratch, compared it with the Scikit-learn library’s implementation, and then evaluated its performance using K-fold cross-validation. To determine if there is a significant difference between the custom KNN and Scikit-learn KNN, I have done a Paired T-test. Based on the T-test, we will either reject the null hypothesis or accept it.

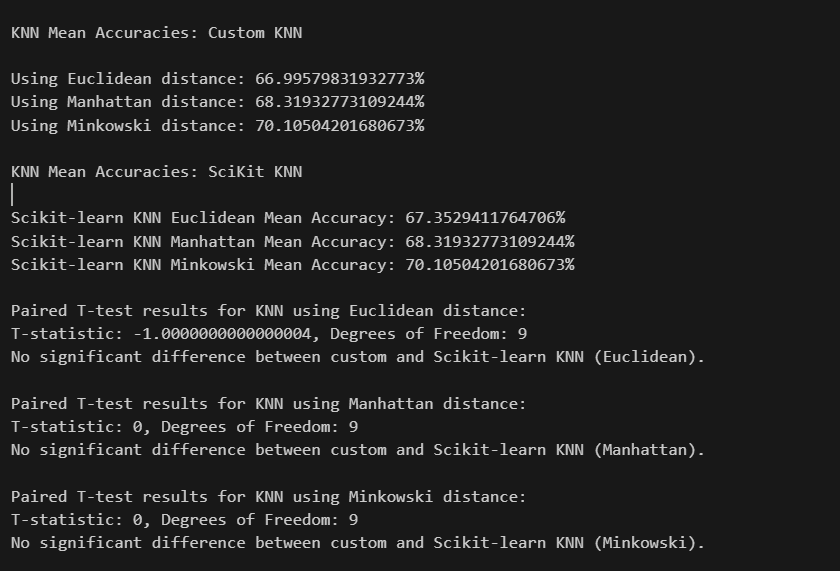
**Dataset 1: Breast cancer Dataset**

For the Breast cancer dataset, it includes features like, age, menopause, tumor-size, inv-nodes, node-caps, deg-malig, breast, breast-quad, irradiat. For the data processing steps, I have used One-hot encoding to transform the categorical features. Then, the feature values are scaled using StandardScaler. Additionally, target labels are mapped to 0 (no-recurrence-events) and 1 (recurrence-events).

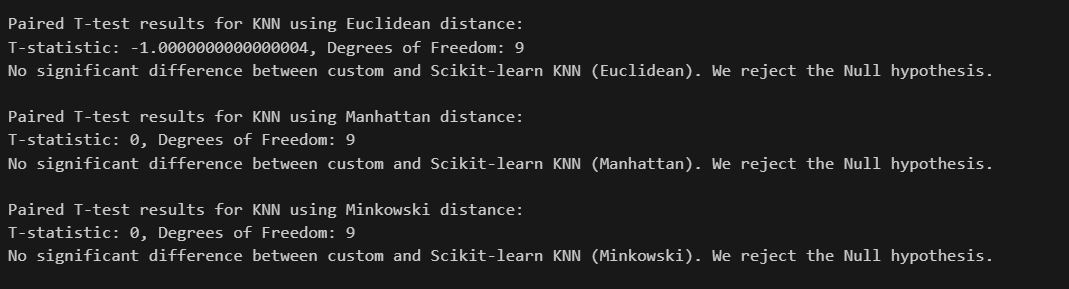
For the implementation of KNN from scratch with different distance metrics:

* Euclidean Distance: Measures the straight-line distance between two points
* Manhattan Distance: Calculate the sum of the absolute differences between the two points
* Minkowski Distance: Generalizes both Euclidean and Manhattan distances, with a parameter p

The k value is taken as 3 in the custom KNN class, and the prediction is calculated by using a max function of the nearest labels. We can modify the value of k and then check for which value of k the accuracy values come best. This custom KNN algorithm is compared to Scikit-learn implementation. To evaluate the performance of the custom KNN model and Scikit-learns model, I have used 10-fold cross-validation. In the custom cross validation function it splits the data into 10 folds and the model is trained on 9 folds and tested on the remaining one. This process repeats for all 10 folds, and the average accuracy is reported.



For the Paired T-test, I have tried to check if there is a significant difference between the custom KNN and Scikit-learn KNN. The null hypothesis was that there is no significant difference between the two implementations. In the result below, we can see that there is no significant difference between the custom implementation and the Scikit-learn KNN implementation for all the 3 distance metrics so we reject the Null hypothesis.



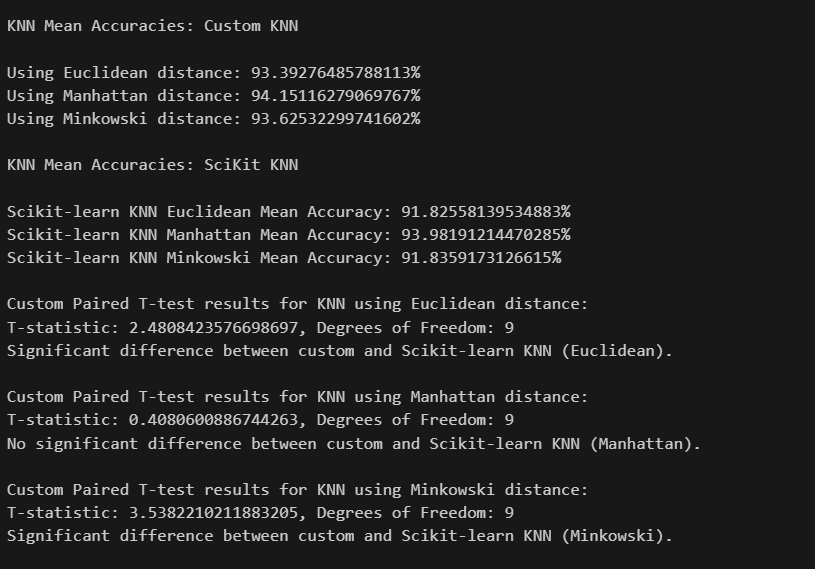
**Dataset 2: Car Evaluation Dataset**

For the Car Evaluation dataset, it includes features like, buying, maint, doors, persons, lug\_boot, safety. For the data processing steps, since all the features are categorical, they are numerically encoded using a custom mapping. Each unique feature value is assigned a unique integer to make them suitable. Additionally, target labels are encoded into numerical values as well.

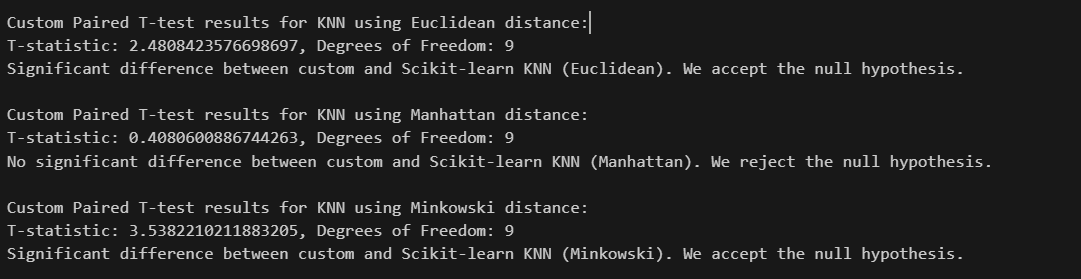
For the implementation of KNN from scratch with different distance metrics:

* Euclidean Distance: Measures the straight-line distance between two points
* Manhattan Distance: Calculate the sum of the absolute differences between the two points
* Minkowski Distance: Generalizes both Euclidean and Manhattan distances, with a parameter p

This custom KNN algorithm is compared to Scikit-learns implementation. To evaluate the performance of the custom KNN model and Scikit-learns model, I have used 10-fold cross-validation. In the custom cross validation function it splits the data into 10 folds and the model is trained on 9 folds and tested on the remaining one. This process repeats for all 10 folds, and the average accuracy is reported.



For the Paired T-test, I have tried to check if there is a significant difference between the custom KNN and Scikit-learn KNN. The null hypothesis was that there is no significant difference between the two implementations. In the result below, we can see that there is significant difference between the custom implementation and the Scikit-learn KNN implementation for all the 2 of the distance metrics (Euclidean and Minkowski) so we accept the Null hypothesis for those 2 distance metrics and reject the null hypothesis for Manhattan as there is no significant difference between the two implementation results.



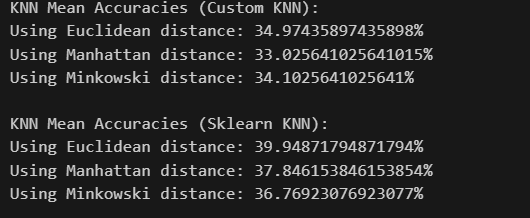
**Dataset 3: Hayes-Roth Dataset**

For the Hayes-Roth dataset, it includes features like, hobby, age, educational level, marital status. In data processing I am just loading the data and preparing the data by extracting features and storing them. As the dataset is quite simple, there is not much processing involved in this dataset.

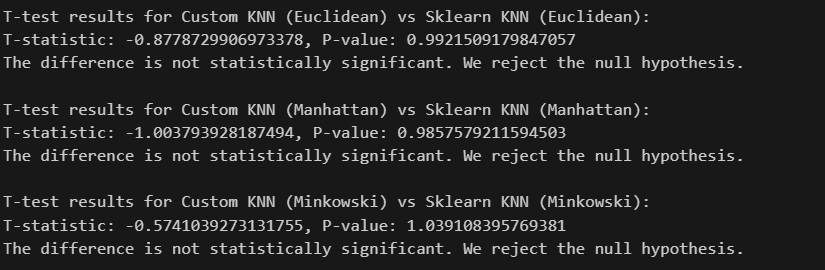
For the implementation of KNN from scratch with different distance metrics:

* Euclidean Distance: Measures the straight-line distance between two points
* Manhattan Distance: Calculate the sum of the absolute differences between the two points
* Minkowski Distance: Generalizes both Euclidean and Manhattan distances, with a parameter p

The k value is taken as 3 in the custom KNN class, and the prediction is calculated by using a max function of the nearest labels. We can modify the value of k and then check for which value of k the accuracy values come best. This custom KNN algorithm is compared to Scikit-learns implementation. To evaluate the performance of the custom KNN model and Scikit-learns model, I have used 10-fold cross-validation. In the custom cross validation function it splits the data into 10 folds and the model is trained on 9 folds and tested on the remaining one. This process repeats for all 10 folds, and the average accuracy is reported.



For the Paired T-test, I have tried to check if there is a significant difference between the custom KNN and Scikit-learn KNN. The null hypothesis was that there is no significant difference between the two implementations. In the result below, we can see that there is no significant difference between the custom implementation and the Scikit-learn KNN implementation for all the three distance metrics, so we reject the Null hypothesis.



**References:**

* [Develop k-Nearest Neighbors in Python From Scratch - MachineLearningMastery.com](https://machinelearningmastery.com/tutorial-to-implement-k-nearest-neighbors-in-python-from-scratch/)
* [A Gentle Introduction to k-fold Cross-Validation - MachineLearningMastery.com](https://machinelearningmastery.com/k-fold-cross-validation/)
* [Exploring KNN with Different Distance Metrics | by Abdullah Siddique | Dev Genius](https://blog.devgenius.io/exploring-knn-with-different-distance-metrics-85aea1e8299)
* [Exploring the Different Distance Metrics in K-Nearest Neighbors (KNN) | by kiarash shamaii | Medium](https://medium.com/@kiarash.shamaii/exploring-the-different-distance-metrics-in-k-nearest-neighbors-knn-d72cac010c2a)