Project 1: KNN

Team RosAlene: Rosato Sara, Khachmadi Arsalene

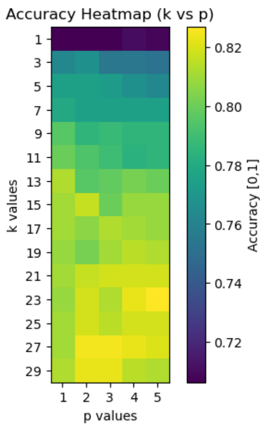
In this first project, we will explore the KNN algorithm and implement it from scratch.
KNN algorithm is a straightforward supervised machine learning approach that may be applied to both classification and regression tasks.

Task 1 and 2

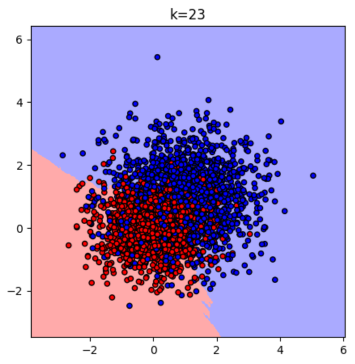
We started by implementing a KNN class that contains the following functions:

- *__init__()* to initialize the model, including empty initialization of X (input), y (output), and the parameter k (number of neighborhoods);
- *train()* in which we train the model by storing the training set to be used afterward in the prediction and in the distance calculation;
- *predict()* in which we predict the most frequent label among the K-nearest neighbors
- *minkowski_dist()* in which we calculate the distance between the training points and the test points.

In notebook *task_2*, after having uploaded the training and validation datasets, we performed hyper-parameter tuning to choose the best values for k and p, mainly chosen by the best accuracy value among those obtained by the combination of the two parameters. We plotted the accuracy heatmap as in Fig. (a) to have a more clear situation of the values obtained in the iterations.
In this way, with the dataset provided, ***best_k=23*** and ***best_p=5***, and we plotted the results as in Fig. (b).



(a) Heatmap



(b) Results with k=23, p=5

Task 4

“All problems due to high dimension may be subsumed under the heading ”the curse of dimensionality”. Since this is a curse, there is no need to feel discouraged about the possibility of obtaining significant results despite it.” Bellman used this term to describe the difficulty of finding an optimum in a high-dimensional space by exhaustive search.
The curse of dimensionality affects the KNN algorithm in many ways, especially in its performance and efficiency. When p (dimension) gets larger, we have the maximum difference, meaning that the greater the dimension, the farther the points will be.
In other words, in high-dimensional spaces, the volume of the spaces grows exponentially with the dimension and the points become increasingly sparse, the distance between points tends to become uniform. For this reason, KNN may require a large number of distance computations, which could eventually slow down the model.
This claim is supported by exercise 2.3, in which the CDF (cumulative distribution function) of the points depends on the volume of the ball which directly affects the probability distribution of the points within the space.
In this way, the notion of ”nearest neighbors” vanishes, since neighborhoods in high-dimensional spaces are no longer local.
The solutions of the two exercises of task 4 are given below.

ChatGPT

We used ChatGPT to better understand some theoretical aspects of the *predict()* function, to revise the usage of a few Python methods like bincount from the NumPy library, and to remember some concepts of Probability. It was useful in general since it gave us quick answers to precise questions, but some of them were wrong or incomplete. For this reason, we think that it is a valuable tool, but it must be combined with previous knowledge or at least some basis about the topics asked.