*Investigate and Implement KNN Classifier*

1st Ankush Laxman Patil   
ankush.patil.stud@fra-uas.de

2nd Ayan Borthakur  
ayan.borthakur@stud.fra-uas.de

3rd Nasir Ishaq  
Nasir.ishaq@stud.fra-uas.de

*Abstract— Machine learning (ML) has grown over a wide range of sectors such as finance, education, communication, transportation, retail, and healthcare. In response to this, researchers have developed a variety of algorithms to analyze a large amount of data and derive quantifiable insights from that data. Subsequently, ML has a different range of algorithms to support text, image, audio, video, and numeric formats, and based on the application ML can predict the outcomes. Supervised learning, which is a type of machine learning, uses classification and regression to analyze the data. In general, there are several types of classifiers present under supervised machine learning, such as decision trees, support vector machines, naive Bayes, and k-nearest neighbors (kNN) designed under this section. In this paper, we have proposed the implementation of the most widely used k-nearest neighbor (kNN) machine learning algorithm used to predict the outcome variable or dependent variable based on the input variables or independent variables. KNN classifier can be used with HTM by incorporating it as a sub-module for classification tasks. Precisely, HTM models can learn to represent the input data in a high-dimensional feature space and then use the kNN classifier to classify the data based on the closest neighbors in this feature space. This can be useful in situations where the input data has a complex temporal structure and requires a more sophisticated approach to classification than simple threshold-based methods. So the developed kNN model is also integrated with the Neocortex API in order to get the input dataset from Hierarchical temporal memory (HTM). Designed kNN model gets a stream of sequences as input from HTM and then it classifies whether the predicated sequence has a match or mismatch with the input data sequence with 80% accuracy. Additionally, the design procedure, challenges, and enhancements to improve model accuracy are discussed in the paper.*

*Keywords—Machine Learning; Supervised Learning, K-Nearest Neighbors; Neocortex API; Hierarchical temporal memory.*

# Introduction

The aim of machine learning is to create statistical models that allow computers to learn from data without being to be explicitly programmed. More precisely, these statistical models and algorithms generate the "learn" method, which helps the computer in predictions about the future based on the trained data. It is also observed that some of the algorithms predict with better accuracy in the specific type of dataset. Supervised, unsupervised, and reinforcement learning are the three main types of machine learning based on the learning patterns of the model.

### The supervised machine learning algorithms learn from input variables and a pre-labeled dataset to predict an output variable. supervised learning can be used for a wide variety of applications, such as image and speech rec. cognition, natural language processing, fraud detection, and recommendation systems based on it regression and classifiers algorithm.

## Regression

Regression is a type of supervised learning which predicts a continuous numerical output for a given input data point. The input data is a set of features or attributes, but the output is a continuous value rather than a discrete class label. In regression, the algorithm learns from labeled training data to predict the value of a target variable based on the values of other variables. The best example of a regression algorithm could be trained to predict the price of a house based on features such as the number of bedrooms, location, age, and area.

## Classification

Classification is also a type of supervised learning where the goal is to predict a discrete class label for a given input data point. In classification, the algorithm learns from labeled training data to classify new, unseen data points into one of the predefined classes. For example, a classification algorithm could be trained to classify if the person is diabetic or non-diabetic based on certain input features. The algorithm would learn from a set of labeled characteristics of a large dataset and their corresponding classifications, and then it could predict whether an unknown person is diabetic or not.

In this paper, we have specifically focused on designing the k-nearest neighbor (kNN) classifier which comes under the supervised machine learning algorithm. It is mainly used for pattern recognition and classification problems. The basic idea is to develop the kNN classifier algorithm as a prototype and then integrate it with the *Neocortex API.* The focus of Neocortex API is to implement Hierarchical Temporal Memory Cortical Learning Algorithm. HTM is a type of machine learning model that is inspired by the structure and function of the neocortex in the brain. HTM models are designed to learn and recognize patterns in time-varying data, such as audio, video, and sensor data. So, the HTM will act as an input to the kNN classifier and this integration of kNN with Neocortix API can be useful in situations where the input data has a complex temporal structure and requires a more sophisticated approach to classification than traditional threshold-based methods.

# Literature Review

In this section, the kNN

we will understand the KNN in detail and understand the design procedure of different researchers.

The selection of K is an important parameter in the algorithm, and its value determines the complexity and accuracy of the classifier.

In a study by Zhang et al. (2019), they proposed an adaptive KNN algorithm based on the density of the data points. The algorithm dynamically adjusts the value of K based on the density of the data points in the local region. They showed that the adaptive KNN algorithm outperformed the traditional KNN algorithm on several benchmark datasets.

Zhang et al. (2019) proposed an adaptive KNN algorithm based on the density of the data points in their study. Based on the density of the data points in the immediate area, the algorithm dynamically modifies the value of K. On several benchmark datasets, they demonstrated that the adaptive KNN algorithm performed better than the conventional KNN technique.

In another study by Yu et al. (2020), they proposed a weighted KNN algorithm that assigns different weights to the K nearest neighbors based on their distance from the query point. The weights are computed using a Gaussian kernel function, and the classifier is trained using a cross-validation technique. They showed that the weighted KNN algorithm outperformed the traditional KNN algorithm on several benchmark datasets.

Liu et al. (2021) proposed a hybrid KNN algorithm that combines KNN with the random subspace method. The algorithm randomly selects a subset of features and applies KNN to the reduced feature space. The process is repeated multiple times, and the results are combined using an ensemble technique. They showed that the hybrid KNN algorithm outperformed both the traditional KNN algorithm and the random subspace method on several benchmark datasets.

In conclusion, the literature suggests that KNN can be enhanced by adapting the value of K based on the density of the data points, assigning different weights to the K nearest neighbors based on their distance, and combining KNN with other algorithms using ensemble techniques. These techniques can improve the accuracy and robustness of the KNN classifier for various classification problems.

# Theoretical background and parameters

The k-nearest-neighbor classification algorithm was first used in an unpublished US Air Force School of Aviation Proceedings of the International Conference on Intelligent Computing and Control Systems to execute characteristic analysis when clear parametric approximations of probability densities were unknown or difficult to determine. In the KNN classifier algorithm, there are several parameters that plays important role in algorithm designs, and those parameters are discussed below,

## Distance calculations

The KNN algorithm works by finding the k-nearest neighbors to a new input sample in the training dataset based on a distance metric. The distance metric can be Euclidean distance, Manhattan distance, or any other appropriate distance measure. Once the k-nearest neighbors are identified, the algorithm assigns the new sample to the class that is most common among its k-nearest neighbors. The most common distance calculations techniques used in kNN are described below:

### Euclidean Distance: Euclidean distance is the straight-line distance between two points in a Euclidean space. In other words, it is the distance between two points in a 2D or 3D space. The Euclidean distance between two points (a1, b1) and (a2, b2) can be calculated as follows:

distance = sqrt((a2 - a1)^2 + (b2 - b1)^2)

2. Manhattan Distance: Manhattan distance is the distance between two points measured along the axes at right angles. It is named after the grid-like layout of the streets in Manhattan. The Manhattan distance between two points (a1, b1) and (a2, b2) can be calculated as follows:

distance = |a2 - a1| + |b2 - b1|

## K value selection:

The value of K is a hyperparameter that needs to be selected before making predictions. The value of K controls the number of neighbors that are used to make predictions. A larger value of K results in a smoother decision boundary but may result in more misclassifications of the training data. A smaller value of K results in a more complex decision boundary but may result in overfitting the training data. The choice of K is typically made using a validation set or cross-validation. The algorithm is trained on a portion of the data, and the remaining data is used to validate the performance of the algorithm for different values of K. The value of K that results in the highest validation accuracy is selected as the final value of K.

## Voting principle:

In the KNN algorithm, the prediction for a new observation is made based on the class labels of its K nearest neighbors. The class with the most number of occurrences among the K nearest neighbors is selected as the predicted class for the new observation. This is known as the majority voting principle.

The majority voting principle is simple yet effective and has been shown to perform well in many applications. However, it is important to note that the choice of K can affect the voting outcome, and a larger value of K may result in a less confident prediction. Additionally, in the case of imbalanced data, where one class has significantly more instances than the other, the majority voting principle may result in biased predictions. In such cases, techniques such as weighted voting or distance-weighted voting can be used to balance the influence of each class.

Apart from these parameters Feature selection and Preprocessing techniques such as normalization, standardization, and scaling can be considered while designing the model and increasing its accuracy of the model.

# Theoretical background and parameters

In this section, the design and working methodology of the KNN classifier is explained in detail with the help of examples.

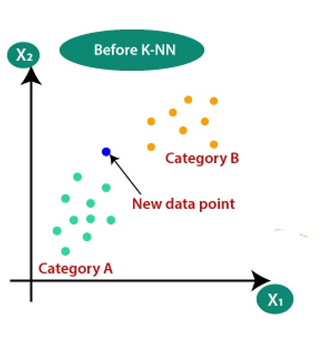


Figure. 1: Plotted KNN Data for prediction

As described in Figure. 1 the categorical data is plotted on X and Y axis and data has two primary categories i.e. Category A and Category B. The goal is to predict the new data points given in the blue column which do not belong to any category. This is a classic example of a classification-based problem and in order to predict the outcome we have to follow the below-mentioned steps,

Step-1: Select the number K of the neighbors for computation and it is recommended that select the odd value of K for better predictions. In this case, let's choose k=5. so, the model will select the 5 nearest neighbors.

Step-2: In the second step, the algorithm will calculate the Euclidean distance as shown in Figure 2 from all the 5 nearest neighbors from the new data points and store the distance in the distance table.

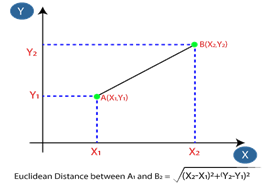


Figure. 2: Euclidean Distance Calculation

Step-3: The distance table will have all the values along with their known category label.

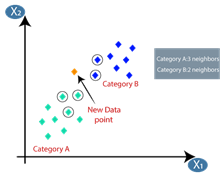


Figure. 3: Euclidean Distance Calculation

Step 4: In step 4, As Figure 3 shows the voting principle will calculate the vote for each category and then the highest number of vote categories is assigned to new data points. For example, the Category of the New Data Point is A and the final outcome is given in Figure.4 below,

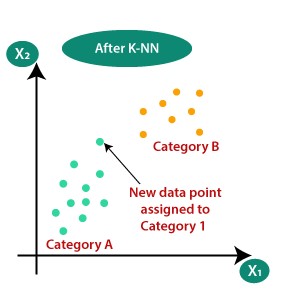


Figure. 4: Final KNN Classifier Outcome

As discussed in the above example the KNN Classifier is an effective machine-learning algorithm for classification problems and works well with all kinds of categorical dataset.

# Model Construction and Classification Process

# Results

# Conclusion

## Base model prototype with hard-coded sequnce

## The enchace version of KNN Classifer

1. Inout and training phase

TP AND Multisequnce learning things

1. Desigin mdel

Clases

## Base model prototype with hard-coded sequnce

## The enchace version of KNN Classifer

1. Inout and training phase

TP AND Multisequnce learning things

1. Desigin mdel

Clases

##### References

The template will number citations consecutively within brackets [1]. The sentence punctuation follows the bracket [2]. Refer simply to the reference number, as in [3]—do not use “Ref. [3]” or “reference [3]” except at the beginning of a sentence: “Reference [3] was the first ...”

Number footnotes separately in superscripts. Place the actual footnote at the bottom of the column in which it was cited. Do not put footnotes in the abstract or reference list. Use letters for table footnotes.

Unless there are six authors or more give all authors’ names; do not use “et al.”. Papers that have not been published, even if they have been submitted for publication, should be cited as “unpublished” [4]. Papers that have been accepted for publication should be cited as “in press” [5]. Capitalize only the first word in a paper title, except for proper nouns and element symbols.

For papers published in translation journals, please give the English citation first, followed by the original foreign-language citation [6].

1. G. Eason, B. Noble, and I. N. Sneddon, “On certain integrals of Lipschitz-Hankel type involving products of Bessel functions,” Phil. Trans. Roy. Soc. London, vol. A247, pp. 529–551, April 1955. *(references)*
2. J. Clerk Maxwell, A Treatise on Electricity and Magnetism, 3rd ed., vol. 2. Oxford: Clarendon, 1892, pp.68–73.
3. I. S. Jacobs and C. P. Bean, “Fine particles, thin films and exchange anisotropy,” in Magnetism, vol. III, G. T. Rado and H. Suhl, Eds. New York: Academic, 1963, pp. 271–350.
4. K. Elissa, “Title of paper if known,” unpublished.
5. R. Nicole, “Title of paper with only first word capitalized,” J. Name Stand. Abbrev., in press.
6. Y. Yorozu, M. Hirano, K. Oka, and Y. Tagawa, “Electron spectroscopy studies on magneto-optical media and plastic substrate interface,” IEEE Transl. J. Magn. Japan, vol. 2, pp. 740–741, August 1987 [Digests 9th Annual Conf. Magnetics Japan, p. 301, 1982].
7. M. Young, The Technical Writer’s Handbook. Mill Valley, CA: University Science, 1989.