Article

High-Level K-Nearest Neighbors (HLKNN): A Supervised Machine Learning Model for Classification Analysis

Elife Ozturk Kiyak [1] , **Bita Ghasemkhani** [2] **and Derya Birant** [3,*]

[1] Independent Researcher, Izmir 35390, Turkey; elife.ozturk@cs.deu.edu.tr
[2] Graduate School of Natural and Applied Sciences, Dokuz Eylul University, Izmir 35390, Turkey; bita.ghasemkhani@ogr.deu.edu.tr
[3] Department of Computer Engineering, Dokuz Eylul University, Izmir 35390, Turkey
[*] Correspondence: derya@cs.deu.edu.tr

Abstract: The k-nearest neighbors (KNN) algorithm has been widely used for classification analysis in machine learning. However, it suffers from noise samples that reduce its classification ability and therefore prediction accuracy. This article introduces the high-level k-nearest neighbors (HLKNN) method, a new technique for enhancing the k-nearest neighbors algorithm, which can effectively address the noise problem and contribute to improving the classification performance of KNN. Instead of only considering *k* neighbors of a given query instance, it also takes into account the neighbors of these neighbors. Experiments were conducted on 32 well-known popular datasets. The results showed that the proposed HLKNN method outperformed the standard KNN method with average accuracy values of 81.01% and 79.76%, respectively. In addition, the experiments demonstrated the superiority of HLKNN over previous KNN variants in terms of the accuracy metric in various datasets.

Keywords: machine learning; k-nearest neighbors; classification; supervised learning; artificial intelligence

1. Introduction

Machine learning (ML) is a technique to equip a system with the capability of predicting future events. It utilizes various algorithms to analyze historical data and learn from them with the intention of forecasting new outcomes for unseen data. Actually, it attempts to discover patterns in past experiences hidden in related data through the training process and generates a predictive model. Nowadays, machine learning plays notable roles in different fields to contribute to making effective decisions in organizations. It is worth emphasizing that ML is currently an essential concept to extend the artificial intelligence (AI) of smart systems by alleviating human interventions because of their limited brain and memory capacities. AI is a research area in computer science that aims to simulate human-like thoughts in machines applicable in many fields, including industry, health, security, transportation, energy, and commerce. Also, artificial intelligence paves the route for the advent of modern technologies struggling to automate tasks and act independently from individuals [1].

There is a wide range of applications of ML, such as anomaly detection to identify abnormal items, recommendation systems to offer items to users, clustering to group similar items, regression for predicting numerical values, and classification to categorize items. Bearing this in mind, ML applications rely on the extraction of patterns and regularities from the previous related data. Therefore, the reproducibility and reusability of ML-based models make them appropriate in many fields such as intelligent decision-making, fraud detection, image processing, financial analytics, facial recognition, Internet of Things (IoT), natural language processing, and user behavior analytics [2].

Based on the type of problems to be solved by ML concepts, there are various categories for learning, including reinforcement, unsupervised, supervised, and semisupervised learning. In reinforcement learning, the ML model relies on rewards during agent–environment interactions. Its purpose is to assess optimal behaviors that enhance the environment. It is widely used in the domains of robotics, gaming, advertising, and manufacturing. In unsupervised learning, the ML model attempts to learn from an unlabeled dataset, for instance, in anomaly detection, density estimation, and clustering tasks. The other type of machine learning algorithm is supervised learning, which only learns from labeled data to find out a suitable mapping function. It is commonly used in credit scoring, sentiment analysis, email filtering, and risk assessment [3]. Semisupervised learning is a combination of unsupervised and supervised learning techniques utilizing both unlabeled and labeled data in its learning process, including applications such as fraud detection, speech analysis, and text classification.

Classification is a type of supervised learning technique, which is extensively used to generate various ML models with the purpose of class label prediction in a related dataset. There are many classification methods proposed in machine learning literature for a broad range of applications. The following are the most well-known classification algorithms: naive Bayes, random forest, decision trees, support vector machines, and neural networks. Moreover, the k-nearest neighbors (KNN) algorithm is one of the most significant classification algorithms, which can be applied to regression tasks, too. KNN is an instance-based learning methodology, also called "lazy learning". It means that the training dataset is postponed to be completely processed until encountering a given test query for prediction via this algorithm. Actually, KNN can perform an exhaustive search in large dimensions. In the KNN classifier, the training set is supposed as an m-dimensional space of patterns for searching the k-closest tuples to a given test query. The closeness is described in terms of a specific distance metric, e.g., Euclidean distance. Additionally, the min–max normalization approach is applicable to balance the effect of various ranges before calculating the distance [4].

In this study, we improved the KNN algorithm due to its advantages such as effectiveness, easy implementation, and simplicity [5]. The benefits of KNN also include ease of interpretation and understanding of the results. Furthermore, it has the ability to predict both discrete and categorical variables; thus, it can be utilized for both regression and classification tasks. Another advantage of KNN is the independence of any data distribution since it uses local neighborhoods. Moreover, it can easily support incremental data since the training process is not needed for the newly arrived samples. The standard KNN algorithm has been proven to be an effective method in many studies [6].

Even though KNN is effective in many cases, it has a shortcoming that reduces its classification ability. Noisy data samples are a considerable limitation since they only search the nearest neighbors of a test instance, and these samples affect the result of the neighborhood concept, so all of these affect the accuracy. The proposed method in this paper decreases this limitation by enhancing the search process.

The main contributions and novel aspects of this study are summarized as follows:

i.   This paper proposes a machine learning method, named high-level k-nearest neighbors (HLKNN), based on the KNN algorithm.
ii.   It contributes to improving the KNN approach since the noisy samples inefficiently affect its classification ability.
iii.   Our work is original in that the HLKNN model was applied to 32 benchmark datasets to predict the related target values with the objective of model evaluation.
iv.   The experimental results revealed that HLKNN outperformed KNN on 26 out of 32 datasets in terms of classification accuracy.
v.   The presented method in this study was also compared with the previous KNN variants, including uniform kNN, weighted kNN, kNNGNN, k-MMS, k-MRS, k-NCN, NN, k-NFL, k-NNL, k-CNN, and k-TNN, and the superiority of HLKNN over its counterparts was approved on the same datasets.

The rest of this paper is organized as follows. The related works are discussed in Section 2. The proposed HLKNN method is explained in detail in Section 3. The experimental studies are presented in Section 4. Eventually, the conclusion and future works of the current study are described in Section 5.

2. Related Works

First, this section briefly explains the most relevant works to our study. After that, it describes the various types of KNN algorithms in the literature.

2.1. A Review of Related Literature

K-nearest neighbors (KNN) is a supervised machine learning method that can be utilized for both classification and regression tasks. KNN considers the similarity factor between new and available data to classify an object into predefined categories. KNN has been widely used in many fields such as industry [7–9], machine engineering [10], health [11–13], marketing [14], electrical engineering [15], security [16–18], manufacturing [19], energy [20–22], aerial [23], environment [24], geology [25,26], maritime [27,28], geographical information systems (GIS) [29], and transportation [30].

In the field of industry, the authors introduced a hybrid bag of features (HBoF) in [7] to classify the multiclass health conditions of spherical tanks. In their HBoF-based approach, they extracted vital features using the Boruta algorithm. These features were then fed into the multiclass KNN to distinguish normal and faulty conditions. This KNN-based method yielded high accuracy, surpassing other advanced techniques. In [8], the authors have reported that the KNN algorithm attained a great performance according to different evaluation metrics for predicting the compressive strength of concrete. Their model was highly recommended in the construction industry owing to the fact that it required fewer computational resources to be implemented. In another work [9], a KNN-based method was developed to efficiently detect faults by considering outliers on the basis of the elbow rule for industrial processes. Similarly, in machine engineering, the authors presented a KNN-based fault diagnosis approach for rotating machinery, which was validated through testing on the bearing datasets [10].

In the field of health, Salem et al. [11] investigated a KNN-based method for diabetes classification and prediction that can contribute to the early treatment of diabetes as one of the main chronic diseases. The outperformance of their method was evaluated in terms of various metrics and approved its applicability in the healthcare system for diabetes compared to other techniques. In another work [12], the researchers focused on an approach to early detect placental microcalcification by the KNN algorithm, which could lead to the improvement of maternal and fetal health monitoring during pregnancy. The results gained from real clinical datasets revealed the efficiency of the proposed model for pregnant women. In [13], a wearable device was introduced on the basis of embedded insole sensors and the KNN machine learning model for gait analysis from the perspective of controlling prostheses. The results of applying the KNN algorithm showed the success of the device in predicting various gait phases with high accuracy.

In marketing, Nguyen et al. [14] represented a beneficial recommendation system via KNN-based collaborative filtering, in which similar users according to their cognition parameters were effectively grouped to obtain more relevant recommendations in different e-commerce activities. In the field of electrical engineering, Corso et al. [15] developed a classification approach for the insulator contamination levels by the KNN algorithm with high accuracy to predict insulating features as predictive maintenance of power distribution networks. In the field of security, a KNN-based technique was proposed in [16] to classify botnet attacks in an IoT network. In addition, the forward feature selection was utilized in their technique to obtain improved accuracy and execution time in comparison to other benchmark methods. In [17], a novel security architecture was provided to effectively deal with forged and misrouted commands in an industrial IoT considering different technologies, namely, KNN, random substance learning (RSL), software-defined network

(SDN), and a blockchain-based integrity checking system (BICS). In another work [18], an intrusion detection system (IDS) was implemented for wireless sensor networks by employing KNN and arithmetic optimization algorithms in that an edge intelligence framework was utilized for denial-of-service attacks in WSNs.

In the field of manufacturing [19], Zheng et al. introduced a consensus model on the basis of KNN to classify cloud transactions regarding their priorities. In their model, different parameters, e.g., service level agreements (SLA), cloud service providers (CSP), cloud service consumers (CSC), and smart contract (SC) types, were utilized for distance calculation in the KNN algorithm. In the field of electrical energy [20], a short-term load forecasting technique was proposed by the weighted KNN algorithm to achieve high accuracy for fitting the model. Similarly, in another work [21], the authors focused on an effective KNN-based model for the failure diagnosis of wind power systems regarding particle swarm optimization. In [22], the researchers investigated seismic energy dissipation for rocking foundations in the seismic loading process by means of different supervised machine learning algorithms, including KNN, support vector regression (SVR), and decision tree regression (DTR). The k-fold cross-validation was applied to the mentioned algorithms and, according to the results, KNN and SVR outperformed DTR in terms of accuracy.

In the aerial field, Martínez-Clark et al. [23] represented a KNN-based method as a flock inspiration from nature for a group of unmanned aerial vehicles (UAVs). In their method, an optimal number of UAVs was obtained with regard to heading synchronization in drone implementation. In the environment field, a predictive model was developed in [24] for the construction and demolition waste rate forecast by means of KNN and principal component analysis (PCA). In the field of geology, the authors [25] reported that the KNN algorithm was utilized for a three-dimensional envision of the stratigraphic structure of porous media related to sedimentary formations. In a similar work in geology, Bullejos et al. [26] implemented a technique for the evaluation of KNN prediction confidence in that the three-dimensional model for the stratigraphic structure of porous media is approved. The results of their KNN-based method contributed to improving the predictability of groundwater investigations.

In the maritime field, the authors [27] introduced a novel trajectory model for offshore waters with the means of KNN and long short-term memory (LSTM) techniques for high-density and low-density trajectories, respectively. Their experimental results revealed that the mean square error is significantly decreased in comparison with the previous artificial-neural-network-based models. Another research [28] in this area employed a KNN-based approach. The authors utilized the PCA method for predictive ship maintenance, with a specific emphasis on propulsion engines. In the field of GIS, the authors [29] presented a predictive model with an ensemble KNN for typhoon tracks, which included supervised classification and regression applied on various samples of typhoons gaining experimental results with high accuracies and low running times. In the field of transportation [30], the effect of missed data was detected through the KNN classifier for the prediction of traffic flow on public roads. Their results show that the presented method was efficiently applicable for smart transportation in urban traffic control.

2.2. A Review of KNN Variants

In the literature, researchers have provided various types of KNN algorithms considering their importance among other machine-learning-based models. Thanks to the advantages of KNN, it is considered as one of the highly recommended benchmark algorithms in machine learning for different applications, as discussed in the previous section. One of the KNN variants is uniform kNN [31], in which a weighting scheme is considered based on the k value. It means that the distances between a data point and its k-nearest corresponding neighbors have the same impression on the prediction process. Another form of KNN is weighted kNN [31], in that the generated weights through a weighting function are assigned to the k-nearest neighbors of a data point. The idea behind this type of KNN is to provide greater weights for near data points and smaller weights for far data

points, e.g., through a reverse function of the distance between points. Another algorithm, entitled the kNN learning method based on a graph neural network (kNNGNN) [31], involves a weighting function in the KNN graph neural network for a specified data point to predict its class label.

Another type of KNN was introduced as k-min–max sum (k-MMS) [32] regarding the remoteness of every k neighbor from the previously chosen neighbors and the approachability of these k neighbors to the sample, simultaneously. The k-Min Ranking Sum (k-MRS) [32] is the other type of KNN algorithm, in that the neighbors are considered for a sample that is less than k-nearest points to that sample. The other KNN variant is the k-nearest centroid neighborhood (k-NCN) [33], in that the k-nearest centroid neighbors of a sample are found, and the majority of obtained votes are assigned to the class of that sample with the aim of solving their ties in a random manner. Similarly, the nearest neighbor rule (NN) [34] is another variant of KNN. In this classification method, the closest neighbors are allocated to a sample, and then the most probable class is determined via voting technique. Besides, various schemes are proposed for NN to improve its performance, e.g., sample weighting, geometric neighboring, and metric learning.

The other variant of KNN is the k-nearest feature line (k-NFL) [35], which supposes that there are linearly self-reliant points to assign for each class in a minimum manner. Then, the feature points of that class are inferred via a predefined feature plane to calculate the final class points concerning distance calculation. Furthermore, the k-nearest neighbor line (k-NNL) [36] was introduced based on the locally nearest neighborhood rule. In this algorithm, alleviation of the computational cost is considered in comparison with k-NFL because of computing just one feature line for every class. Other types of KNN were discussed in [37], namely, k-center nearest neighbor (k-CNN) and k-tunable nearest neighbor (k-TNN). In the k-CNN model, the remoteness of a data point sample for all feature lines is calculated. After, the voting technique is utilized to select the most probable class among the k-closest feature line class labels. In this model, the leave-one-out error estimator is used for the computation of the most efficient values for k. In the k-TNN model, sample data point distance from all feature lines is calculated through the TNN approach. Then, the k-closest feature lines are identified, and the voting method is applied to them to specify the most probable classes. Similarly, the leave-one-out error estimator is applicable in the k-TNN model to determine the best k values.

Our work is different from the previous studies in many respects. It presents a novel machine learning method, abbreviated as HLKNN. It proposes a different neighborhood searching mechanism. Instead of only considering k neighbors of a given query instance, it also takes into account the neighbors of these neighbors. It contributes to improving the accuracy of KNN and also other variants of the KNN algorithm in the literature.

3. Materials and Methods

3.1. The Proposed Method: High-Level K-Nearest Neighbors (HLKNN)

This section presents a comprehensive explanation of the High-Level K-Nearest Neighbors (HLKNN) method. Figure 1 presents the general structure of the proposed method. The first stage (the data collection stage) includes data discovery, data integration, and data augmentation. Data discovery involves gathering data from sources (i.e., databases) for analysis. Data integration refers to the process of combining datasets from different sources and provides the researcher with a unified view of these data. Data augmentation creates new sensible samples to increase the diversity and size of the training dataset while retaining its meaning. The next stage (the data preprocessing stage) consists of different steps such as data cleaning, data reduction, and data transformation. Data cleaning is performed by handling missing values and removing outlier observations. Data dimensionality reduction (i.e., feature selection) refers to using a method that identifies important variables and maps the samples from a high-dimensional space to a lower-dimensional space. Data transformation includes data smoothing, feature generation, and aggregation to convert the data into a form suitable for analysis. In the machine learning stage, a model

is created by the proposed HLKNN method for categorizing the given input as an output class. For a given k value, the class labels of k-nearest neighbors are determined according to a distance metric such as Euclidean distance, Manhattan distance, or Minkowski distance. Afterward, the class labels of k-nearest neighbors of neighbors are added to the list. After that, a majority voting mechanism is applied to predict the output. This mechanism finds the label that achieves the highest number of votes in the list. For example, the winner of the list [$c1$ $c1$ $c2$ $c2$ $c1$] is determined as the class label "$c1$". In the final stage, the categories of the test set samples are predicted, and evaluation metrics (i.e., accuracy, precision, recall, f-score) are used to assess the efficiency and robustness of the model.

Figure 1. The general structure of the proposed method.

Figure 2 illustrates an example of the proposed method. Assume that the number of neighbors is set to three ($k = 3$). Therefore, the classical KNN method will find the three closest neighbors of the given input sample x and select the square-gray class with the most votes. However, in the general view, it is expected to be classified the data point x as a blue-circle class. To solve this problem, our HLKNN method also considers the class labels of neighbors of these neighbors. In this case, the dominant class label in the list is determined as a blue circle. Thus, the proposed method handles the noisy observations in an efficient way during the classification task compared to the classical KNN method, which suffers from noise samples. The green dashed line denotes the low-level neighborhood, while the red dashed line indicates the high-level neighborhood.

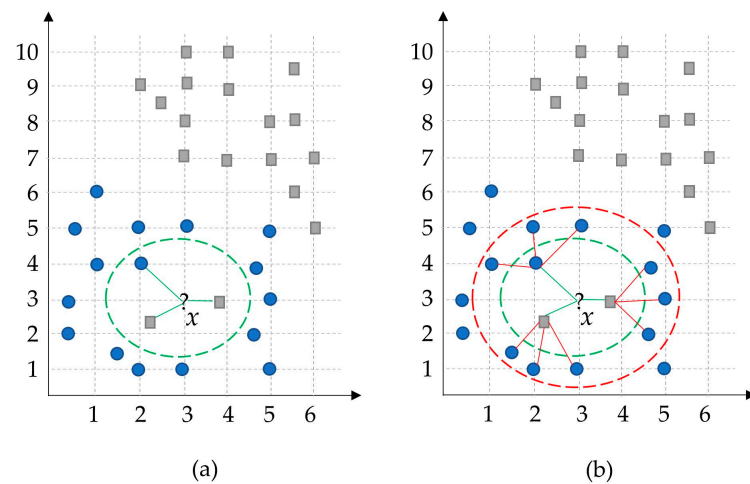**Figure 2.** A sample simulation of the presented HLKNN method. (**a**) low-level neighborhood; (**b**) high-level neighborhood. (?, circle, square, red and green dot boxes, and lines mean what are the neighbors of an instance point x, a specific class of instances, another specific class of instances, boundaries to show high-level neighborhood and low-level neighborhood, and belonging of neighbors to an instance point, respectively)

3.2. Formal Description

Assume that D is a dataset with n samples such that $D = \{(x_i, y_i)\}_{i=1}^n$. A data sample (x_i, y_i) consists of an m-dimensional input vector x_i and its corresponding class y_i. A class label y_i is an element of a finite number of r different labels such that $y_i \in Y = \{c_1, c_1, \ldots, c_r\}$. Suppose a number $k > 0$ is fixed and denotes the number of neighbors. A KNN method basically depends on a distance metric $dist : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$ such as the Euclidean distance measure. Given a query instance x, the neighborhood $N_k(x)$ in D is defined as the set of k samples $\overrightarrow{x}$, where $dist\left(x, \overrightarrow{x}\right)$ is smallest.

Let L be a lazy learning algorithm, which takes a dataset $D = \{(x_i, y_i)\}_{i=1}^n$ and a query instance x as input and returns a class label y as output. The model $M = L(D)$ is a function $M : X \to Y$ that maps a query input $x \in X$ to a class label $y \in Y$. The algorithm searches the k nearest points and selects the most frequent class among them.

For the classification, we propose a two-level KNN search to avoid noisy samples. First, the KNNs to x are computed for a query instance $x \in D$, so that the results are returned, building the set C. Next, the KNNs to each data sample s in C are computed and added in set C. The result of classification is determined by selecting the class label with the most votes.

Definition 1 (Low-Level Neighborhood). *For a given query instance x, $dist_k(x)$ is the distance of the kth-nearest neighbor of x. For a given positive number k, the low-level neighborhood of x, which is denoted by $N_k(x)$, contains every sample $\overrightarrow{x}$ whose distance from x is not greater than kth distance, such as*

$$N_k(x) = \left\{ \overrightarrow{x} \,\middle|\, \overrightarrow{x} \in D \text{ and } dist\left(x, \overrightarrow{x}\right) \leq dist_k(x) \right\} \tag{1}$$

Although the $dist_k(x)$ is well defined for a positive integer k, the cardinality of $N_k(x)$ can be greater than k. For instance, if there are 2 samples for 1-distance from test instance x and 1 sample for 2-distance from the instance, then, the cardinality of the low-level neighborhood will be 3 such that $|N_k(x)| = 3$, which is greater than the value of $k = 2$.

Definition 2 (High-Level Neighborhood). *The high-level neighborhood of a query instance x is the set $NN_k(s)$ for $s \in N_k(x)$ that contains k nearest neighbors of the neighbors of x.*

The pseudocode of the HLKNN method is given in Algorithm A1 in the Appendix A section of this paper. The steps of the algorithm are explained in detail as follows.

Step 1: The algorithm utilizes the KNN search procedure to find k-nearest neighbors $N_k(x)$ of an instance x from the test set D_{Test}.

Step 2: Class labels are inserted into the list, named C, for each neighbor sample s in the set of neighbors $N_k(x)$ (low-level neighborhood). $NN_k(s)$ refers to the high-level neighborhood, which means the set of k-nearest neighbors of the neighbors.

Step 3: After specifying the high-level neighbors, class labels of the new set are added to the list.

Step 4: A voting mechanism is applied to predict the class label of the query instance x.

Step 5: After producing an output for the instance x, the whole process is repeated for other data instances in the testing set. In the end, the list P contains all predicted labels of the testing set.

The time complexity of HLKNN is $O(n(k + k^2))$ since it calculates the distance from the query instance to each sample inside the dataset and determines $(k + k^2)$ nearest neighbors, where n is the number of instances in the dataset and k is the number of neighbors. For instance, for k from 1 to 7, the number of corresponding neighbors yielded 2, 6, 12, 20, 30, 42, and 56, respectively. According to our tests for various values of k, it was considered as 7 in the implementation of the HLKNN method in this study. Consequently, we assessed both KNN and HLKNN for 56 numbers of neighbors over 32 well-known datasets. We obtained 79.04% and 81.01% accuracy for the KNN and HLKNN methods on average, respectively. Although HLKNN involves additional processing time for handling neighbor–neighbor relationships, it has been developed to address the critical constraint of KNN, namely, noisy samples. Therefore, the emphasis is devoted to the reliability of the HLKNN method rather than execution time. The recent developments in high-performance computing enable us to take the edge off this limitation by permitting the parallel execution of the proposed approach.

4. Experimental Studies

This section presents the experiments performed to validate the effectiveness of the proposed method and discusses the main findings. The experimental studies were intended to put the outlined methodology into action, and benchmark datasets were used to validate the HLKNN method. By implementing the methodology, we aimed to provide insight into the contribution of the presented model.

The proposed method was implemented in the C# language using the Weka machine learning library [38,39]. To assess the performance, experiments were conducted by using the 10-fold cross-validation method. It should be noted here that we tested the HLKNN method by different values for the number of neighbors (known as k) and eventually determined the optimal value of 7.

To prove the efficiency of the proposed method, we applied it to 32 different widely used datasets. The validity of the proposed method was assessed by the various evaluation metrics, including accuracy, precision, recall, and f-score [40].

Accuracy refers to the number of correctly predicted data points to the total number of data points. In a more precise definition, it is calculated by dividing the sum of true positives and true negatives by the sum of true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN). A true positive indicates a data point correctly classified as true, while a true negative is a data point correctly classified as false. Conversely, a false positive refers to a data point incorrectly classified as true, and a false negative is a data point incorrectly classified as false. The formula for calculating accuracy is expressed as Equation (2).

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{2}$$

Precision is the proportion of relevant instances among all retrieved instances. In other words, it is determined by dividing the number of true positives by the total number of positive predictions (TP + FP), as represented in Equation (3).

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{3}$$

Recall, referred to as sensitivity, is the fraction of retrieved instances among all relevant instances. It means that it measures the ratio of true positives to all the observations in the actual class, as given in Equation (4).

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{4}$$

The f-score is a metric that combines the precision and recall of the model, and it is calculated as the harmonic mean of the model's precision and recall, as shown in Equation (5).

$$\text{f-score} = 2 * \frac{\text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}} \tag{5}$$

This section is divided into three subsections: dataset description, results, and comparison with KNN variants, in which specific aspects of our research are explained in detail.

4.1. Dataset Description

The effectiveness of the proposed method was demonstrated through experimental studies using a specific set of datasets. The study involved 32 different benchmark datasets broadly used in machine learning, each well known, and publicly available. Table 1 provides details on these datasets, including the number of attributes, instances, classes, and domains. These datasets include a number of instances ranging from 36 to 45,211, attributes ranging from 2 to 45, and classes ranging from 2 to 10. They have different types of values, including numerical, categorical, or mixed type.

Table 1. Dataset characteristics.

ID	Dataset Name	Number of Attributes	Number of Instances	Number of Classes	Missing	Domain
1	balance-scale	4	625	3	Yes	psychology
2	banana	2	5300	2	No	shape
3	bank-marketing	16	45,211	2	No	banking
4	banknote-authentication	4	1372	2	No	finance
5	blood-transfusion-service	4	748	2	No	health
6	climate-simulation-crashes	17	540	2	No	environment
7	credit-approval	15	690	2	Yes	finance
8	dermatology	33	366	6	Yes	health
9	ecoli	7	336	8	Yes	biology
10	flare2	10	1066	2	Yes	environment
11	habermans-survival	3	306	2	Yes	health
12	heart	13	270	2	Yes	health
13	heart-disease-processed-cleveland	13	303	5	No	health
14	heart-disease-reprocessed-hungarian	13	294	5	No	health
15	horse-colic-surgical	27	368	2	Yes	animal
16	indian-liver-patient	10	583	2	No	health
17	iris	4	150	3	No	life
18	led7digit	7	500	10	No	science
19	liver-disorders	6	345	2	Yes	health
20	monks2	6	601	2	No	science
21	pasture	22	36	3	No	environment

Table 1. *Cont.*

ID	Dataset Name	Number of Attributes	Number of Instances	Number of Classes	Missing	Domain
22	planning-relax	13	182	2	No	health
23	postoperative.patient.data	8	90	3	Yes	health
24	saheart	9	462	2	No	health
25	segment	19	2310	7	No	science
26	seismic-bumps	18	2584	2	No	geology
27	sponge	45	76	3	Yes	marine
28	thoracic-surgery	16	470	2	No	health
29	vertebra-column-2c	6	310	2	No	health
30	vertebra-column-3c	6	310	3	No	health
31	wholesale-region	8	440	3	No	business
32	wine	13	178	3	No	chemistry

The datasets were obtained from the UCI Machine Learning Repository [41], covering various domains such as health, psychology, banking, environment, biology, and so on. The utilization of diverse datasets from different domains showcases the independence of the proposed approach from any specific field.

4.2. Results

To demonstrate the effectiveness of the HLKNN algorithm, we compared it with the standard KNN algorithm through the 10-fold cross-validation technique in terms of accuracy over 32 well-known datasets, which are described in Table 1. Table 2 shows the comparison results. According to the results, the proposed HLKNN method performed equal to or better than the standard KNN algorithm for 26 out of 32 datasets, which are marked in bold. For example, HLKNN showed better performance than KNN in the habermans-survival dataset with accuracy values of 74.84% and 72.55%, respectively. In another instance, HLKNN (95.52%) demonstrated its superiority over KNN (94.94%) in the wine dataset. In addition, HLKNN outperformed KNN on average, with accuracy values of 81.01% and 79.76%, respectively.

Since the HLKNN method includes the neighbors of neighbors, it does not encounter the risk of overfitting for noisy nearest instances in the dataset, and therefore better results can be revealed in comparison with the KNN method, as shown in Table 2. Actually, HLKNN is proposed for conquering the noise problem in the datasets. To avoid overfitting, it should be noted here that the value of k can be optimally determined by considering the data size, i.e., a low k value for small datasets.

During the evaluation process, the 10-fold cross-validation technique was used in this study. This technique involves randomly dividing the data into 10 equal-sized and disjoint partitions. During each iteration, one partition is kept for the test process, whereas the remaining partitions are used for the training task. The procedure is repeated 10 times, each time with different training/testing sets, and the overall evaluation metrics were calculated as the average of all 10 runs. Table 3 shows the ability of the HLKNN classifier fold by fold. The final column in the table reports the average accuracy of the 10-fold results. For example, accuracy values in each fold are not less than 86% in the banana dataset. Accuracy values range between 86.06% and 90.49% in the bank-marketing dataset. For instance, there are small fluctuations in the accuracy rates of the folds for the seismic-bumps dataset.

Accuracy alone is insufficient to evaluate a classifier's performance. Therefore, besides accuracy, the KNN and HLKNN algorithms were also compared in terms of precision, recall, and f-score metrics. The formulas of the metrics are discussed in the experimental studies section. These metrics have values ranging from 0 to 1, where 1 is the best value. In other words, the performance improves as the value increases. The results obtained for each metric are shown in Figure 3, Figure 4, and Figure 5, respectively.

Table 2. Comparison of the KNN and HLKNN algorithms on 32 benchmark datasets in terms of classification accuracy (%) with equal and greater numbers in bold for each row.

Dataset Name	KNN	HLKNN
balance-scale	**89.44**	89.29
banana	**89.75**	89.40
bank-marketing	88.52	**88.72**
banknote-authentication	**99.85**	**99.85**
blood-transfusion-service	**77.14**	77.01
climate-simulation-crashes	91.30	**91.48**
credit-approval	**86.67**	**86.67**
dermatology	96.17	**96.44**
ecoli	**86.90**	86.03
flare2	81.99	**82.37**
habermans-survival	72.55	**74.84**
heart	81.97	**82.69**
heart-disease-processed-cleveland	59.74	**60.37**
heart-disease-reprocessed-hungarian	64.63	**66.69**
horse-colic-surgical	80.67	**81.33**
indian-liver-patient	67.58	**67.90**
iris	96.67	**97.33**
led7digit	73.60	**74.80**
liver-disorders	61.16	**62.61**
monks2	58.40	**65.41**
pasture	66.67	**70.00**
planning-relax	62.64	**70.73**
postoperative.patient.data	**71.11**	**71.11**
saheart	64.29	**69.48**
segment	**94.89**	94.28
seismic-bumps	93.27	**93.31**
sponge	**92.11**	91.96
thoracic-surgery	84.68	**84.89**
vertebra-column-2c	79.03	**80.65**
vertebra-column-3c	76.13	**77.42**
wholesale-region	67.95	**71.59**
wine	94.94	**95.52**
Mean	*79.76*	*81.01*

Table 3. Evaluation of HLKNN algorithm on 32 benchmark datasets in terms of classification accuracy (%) fold by fold (F stands for fold).

Dataset Name	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	Mean
balance-scale	90.48	84.13	80.95	88.89	95.24	87.10	98.39	90.32	87.10	90.32	89.29
banana	90.00	90.94	88.68	88.11	91.70	86.98	88.68	90.94	88.30	89.62	89.40
bank-marketing	90.07	88.94	87.61	89.16	87.61	89.38	86.06	90.49	89.16	88.72	88.72
banknote-authentication	100.00	100.00	100.00	100.00	99.27	100.00	99.27	100.00	100.00	100.00	99.85
blood-transfusion-service	74.67	76.00	81.33	76.00	80.00	74.67	76.00	76.00	77.03	78.38	77.01
climate-simulation-crashes	92.59	92.59	92.59	90.74	94.44	88.89	90.74	90.74	92.59	88.89	91.48
credit-approval	85.51	86.96	73.91	86.96	84.06	92.75	88.41	84.06	94.20	89.86	86.67
dermatology	94.59	97.30	100.00	97.30	97.30	94.59	100.00	97.22	97.22	88.89	96.44
ecoli	85.29	85.29	91.18	79.41	85.29	82.35	90.91	90.91	81.82	87.88	86.03
flare2	85.05	84.11	85.05	81.31	81.31	77.57	84.91	85.85	82.08	76.42	82.37
habermans-survival	74.19	83.87	80.65	74.19	70.97	64.52	80.00	80.00	66.67	73.33	74.84
heart	86.67	80.00	80.00	73.33	79.31	96.55	82.76	86.21	68.97	93.10	82.69

Table 3. *Cont.*

Dataset Name	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	Mean
heart-disease-processed-cleveland	64.52	64.52	61.29	53.33	60.00	43.33	56.67	76.67	56.67	66.67	60.37
heart-disease-reprocessed-hungarian	76.67	63.33	60.00	60.00	86.21	68.97	58.62	58.62	79.31	55.17	66.69
horse-colic-surgical	80.00	70.00	90.00	80.00	80.00	83.33	83.33	90.00	76.67	80.00	81.33
indian-liver-patient	67.80	69.49	77.97	72.41	60.34	65.52	74.14	62.07	55.17	74.14	67.90
iris	100.00	100.00	93.33	100.00	100.00	100.00	100.00	93.33	93.33	93.33	97.33
led7digit	78.00	70.00	84.00	70.00	70.00	68.00	76.00	78.00	84.00	70.00	74.80
liver-disorders	65.71	57.14	62.86	57.14	68.57	47.06	70.59	67.65	64.71	64.71	62.61
monks2	55.74	65.00	60.00	73.33	61.67	73.33	76.67	60.00	63.33	65.00	65.41
pasture	75.00	50.00	25.00	100.00	75.00	75.00	100.00	66.67	66.67	66.67	70.00
planning-relax	94.74	73.68	66.67	38.89	77.78	72.22	66.67	88.89	50.00	77.78	70.73
postoperative.patient.data	77.78	77.78	55.56	88.89	44.44	55.56	66.67	77.78	100.00	66.67	71.11
saheart	63.83	76.60	63.04	60.87	69.57	67.39	73.91	71.74	78.26	69.57	69.48
segment	91.77	96.10	93.94	91.34	94.37	94.81	95.24	96.10	95.24	93.94	94.28
seismic-bumps	93.05	91.12	95.37	93.44	91.86	94.19	92.25	93.80	93.80	94.19	93.31
sponge	100.00	100.00	100.00	75.00	100.00	87.50	85.71	85.71	85.71	100.00	91.96
thoracic-surgery	87.23	82.98	85.11	87.23	78.72	87.23	80.85	87.23	82.98	89.36	84.89
vertebra-column-2c	77.42	67.74	70.97	87.10	80.65	90.32	77.42	87.10	77.42	90.32	80.65
vertebra-column-3c	80.65	67.74	67.74	80.65	80.65	90.32	67.74	80.65	74.19	83.87	77.42
wholesale-region	68.18	75.00	77.27	65.91	70.45	59.09	81.82	77.27	65.91	75.00	71.59
wine	100.00	100.00	88.89	94.44	83.33	94.44	100.00	100.00	100.00	94.12	95.52

Figure 3 shows the comparison of the performances of algorithms in terms of precision. HLKNN achieved high precision values (above 0.95) for some datasets, including banknote-authentication, dermatology, iris, and wine. As can be seen from the figure, the outperformance of HLKNN over KNN was approved by the precision metric for 29 out of 32 datasets. This means that the HLKNN algorithm frequently tends to produce better results than the KNN algorithm. Moreover, notable differences between the results for the algorithms were observed for some datasets such as the pasture, planning-relax, and wholesale-region datasets. For example, HLKNN (0.9196) performed better than KNN (0.8483) in the sponge dataset with a remarkable difference in terms of precision metric. The biggest difference in the precision values of HLKNN and KNN was observed in the postoperative.patient.data dataset as 0.7111 and 0.5057, respectively. On average, HLKNN (0.8129) performed better than KNN (0.7611) with an improvement of over 5%.

Figure 4 shows the comparison of KNN and the proposed HLKNN method in terms of recall. The results from the figure revealed that HLKNN achieved equal or higher accuracy rates than KNN in 26 of 32 datasets. For example, in the saheart dataset, the recall value obtained by the HLKNN algorithm is closer to 1 than this obtained by the traditional KNN algorithm. As another instance, a notable difference between HLKNN and KNN was detected over the monks2 dataset in terms of recall metric. The biggest difference between the methods was observed on the planning-relax dataset with an improvement of over 8%. As can be seen, the best recall value (almost equal to 1) was obtained in the banknote-authentication dataset. HLKNN also achieved high recall values (above 0.95) for some datasets, including dermatology, iris, and wine.

Figure 5 shows the comparison of the performances of methods in terms of f-score. As can be seen, the outperformance of HLKNN over KNN was approved for 29 out of 32 datasets. For instance, HLKNN (0.9196) has better classification performance than KNN (0.8832) on the sponge dataset. Furthermore, it can be noted that there is a notable difference between the results for both algorithms in some datasets such as pasture, planning-relax, postoperative.patient.data, and wholesale-region. The best performance of the HLKNN method was achieved on the banknote-authentication dataset in terms of f-score value, which is almost equal to 1. On average, HLKNN (0.8111) performed better than KNN (0.7779) with an improvement of over 3%. According to the values of all metrics (accuracy,

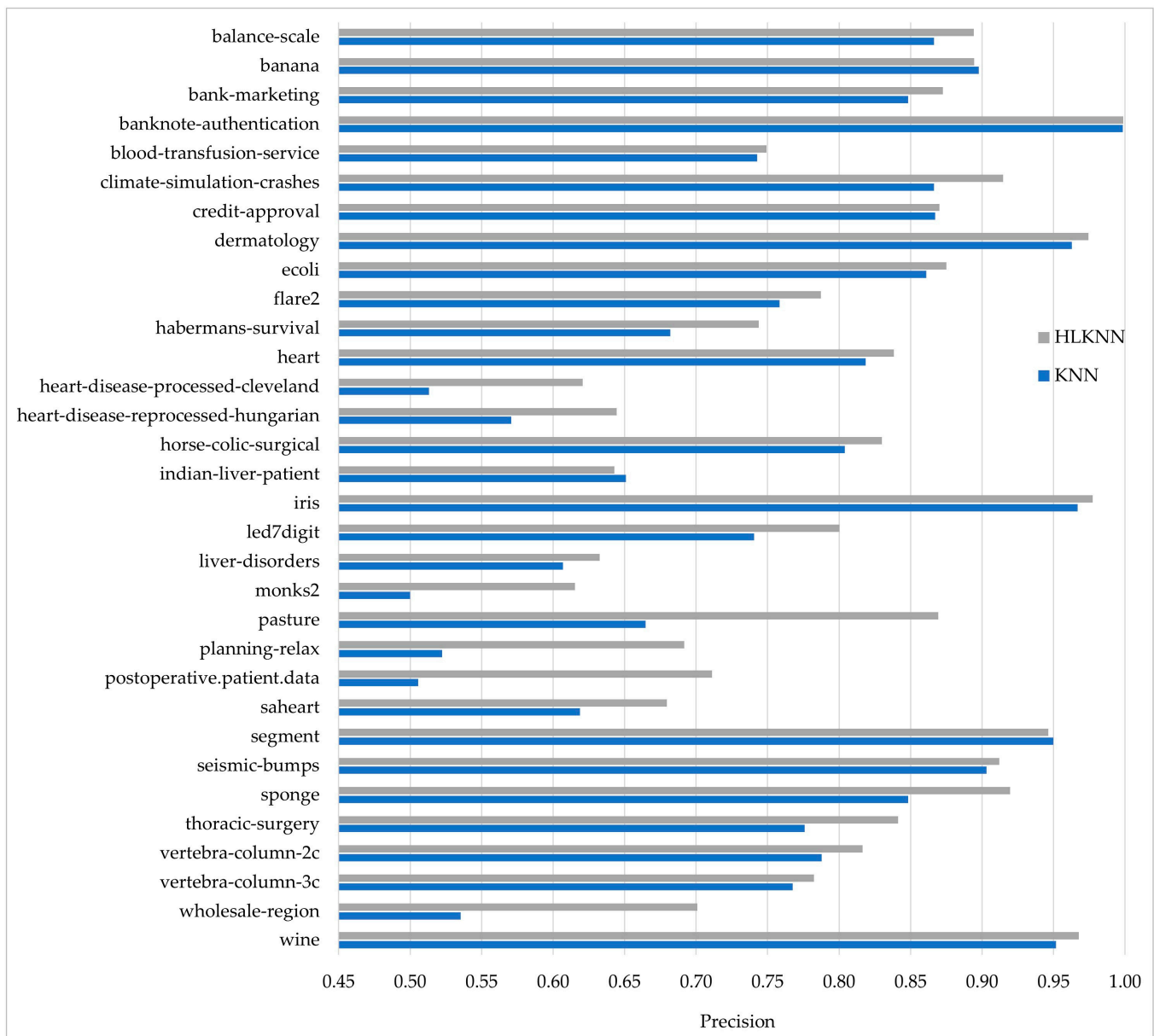precision, recall, f-score), it can be concluded that HLKNN frequently tends to produce better results than KNN.

Figure 3. Comparison of the KNN and HLKNN algorithms on 32 benchmark datasets in terms of precision metric.

4.3. Comparison with KNN Variants

In this section, the proposed HLKNN method was compared with the variants of KNN [31–37] in terms of classification performance. Table 4 shows the results reported in the previous studies on the same datasets. Since the results were directly taken from the referenced studies, a range of popular datasets were regarded, including balance-scale, dermatology, ecoli, habermans-survival, heart, iris, liver-disorders, segment, and wine. The circumstances for datasets (i.e., data preprocessing) were not altered, to ensure a fair comparison in the same conditions between this study and the referenced studies. According to the results, the HLKNN outperformed the mentioned variants of KNN in all the datasets except the liver-disorders dataset. For example, HLKNN (86.03%) showed its superiority over uniform kNN [31] (81.47%), weighted kNN [31] (82.97%), and kNNGNN [31] (81.51%) on the ecoli dataset. As the results show, the HLKNN method

acquired the highest accuracy of 97.33% for the iris dataset in comparison with other variants of KNN. In particular, the highest accuracy difference between HLKNN and the rest was observed in the wine dataset, where HLKNN increased the accuracy by over 26% against the k-MRS algorithm [32].
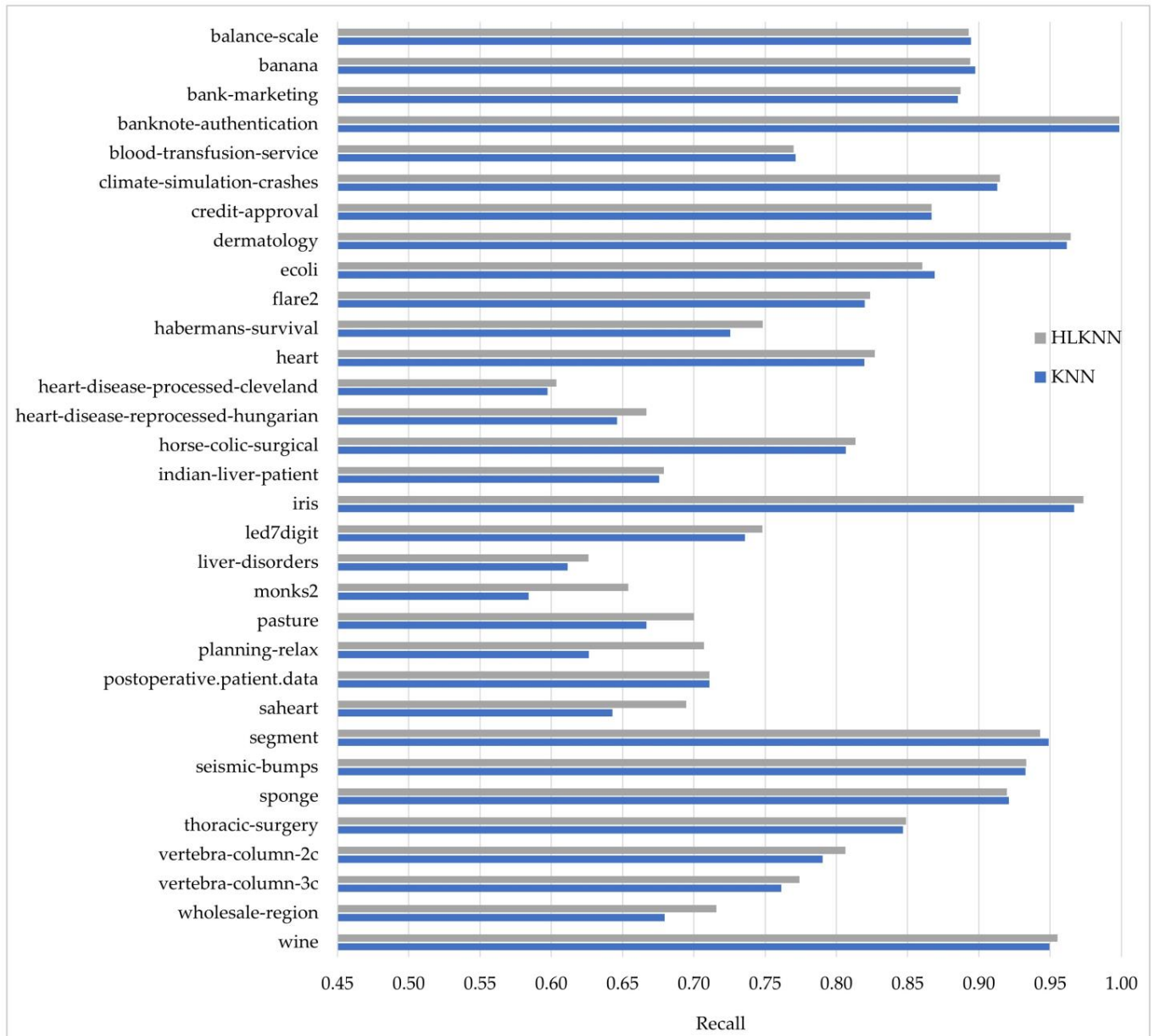

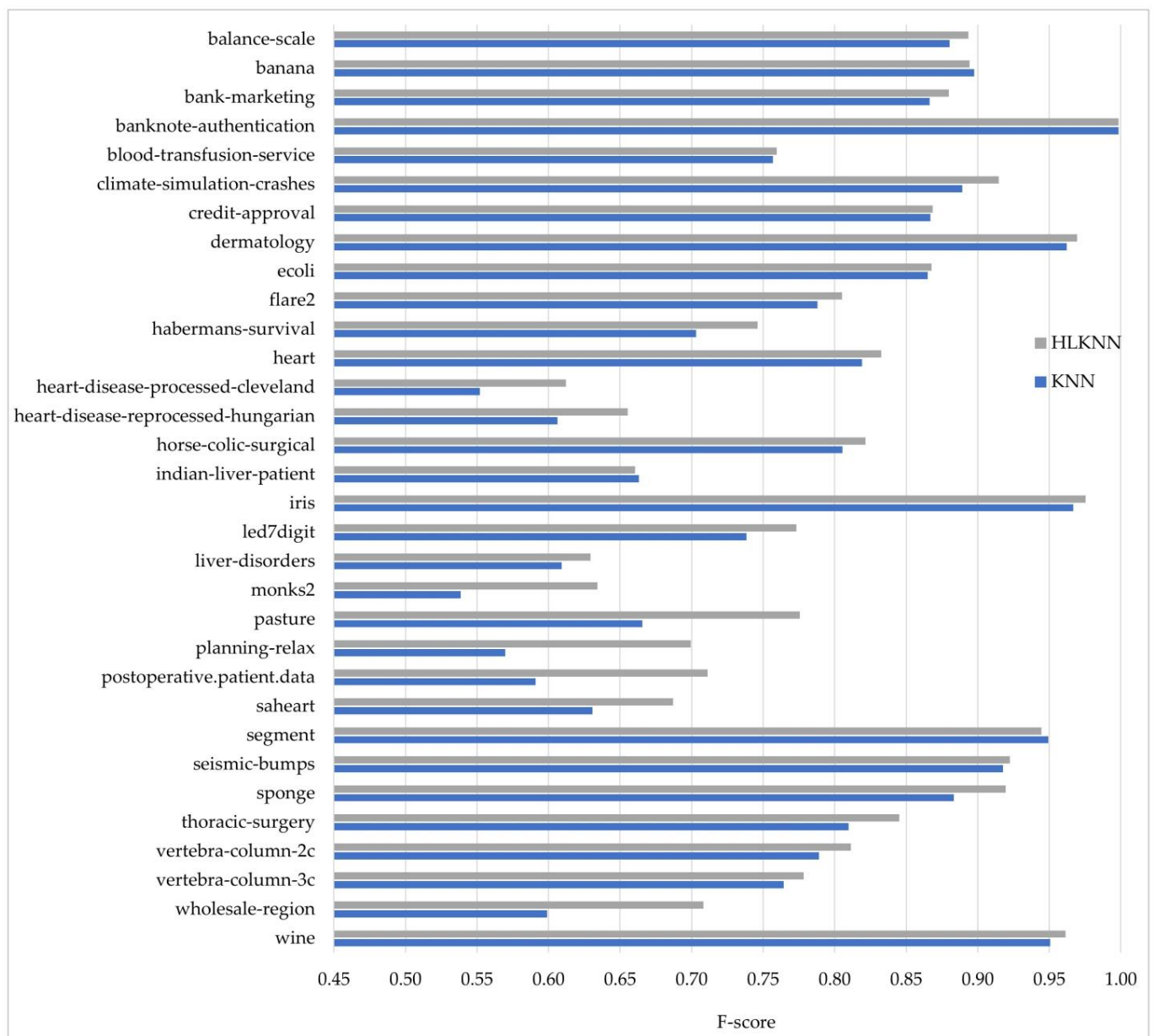**Figure 4.** Comparison of the KNN and HLKNN algorithms on 32 benchmark datasets in terms of recall metric.

Figure 5. Comparison of the KNN and HLKNN algorithms on 32 benchmark datasets in terms of f-score metric.

Table 4. Comparison of the proposed method and various KNN variants in terms of accuracy (%) with the greatest number in bold for each row.

Dataset Name	Uniform kNN [31]	Weighted kNN [31]	kNN GNN [31]	k-MMS [32]	k-MRS [32]	k-NCN [33]	NN [34]	k-NFL [35]	k-NNL [36]	k-CNN [37]	k-TNN [37]	HLKNN (Proposed)
balance-scale	85.47	85.64	86.73	-	-	-	-	-	-	-	-	**89.29**
dermatology	-	-	-	85.08	85.68	89.89	85.68	95.36	93.34	95.03	90.77	**96.44**
ecoli	81.47	82.97	81.51	-	-	-	-	-	-	-	-	**86.03**
habermans-survival	-	-	-	73.14	73.99	72.94	68.25	72.42	64.72	74.31	73.07	**74.84**
heart	79.19	79.07	78.74	63.36	64.29	67.39	59.67	67.79	64.29	66.47	64.75	**82.69**
iris	91.14	92.34	94.27	94.53	95.20	96.27	95.20	96.13	93.73	95.07	96.27	**97.33**
liver-disorders	-	-	-	66.15	66.32	**69.28**	61.63	67.36	59.83	63.48	66.09	62.61
segment	89.79	91.6	92.51	-	-	-	-	-	-	-	-	**94.28**
wine	89.65	91.39	94.78	69.63	68.75	71.22	71.57	85.84	83.36	77.28	69.86	**95.52**

5. Conclusions and Future Work

In this paper, we proposed a machine learning model, named high-level k-nearest neighbors (HLKNN), to be utilized in predictive analytics. This model contributes to improving the classification ability of the KNN algorithm that is widely employed in various fields of machine learning. Since the KNN is sensitive to irrelevant data, its accuracy is adversely affected by the quality of the training dataset. According to the experimental results, our developed model usually outperformed KNN on popular datasets with average accuracy values of 81.01% and 79.76%, respectively. HLKNN performed equal to or better than KNN on 26 out of 32 datasets. Besides, the HLKNN method was compared with different types of KNN algorithms, namely, uniform kNN, weighted kNN, kNNGNN, k-MMS, k-MRS, k-NCN, NN, k-NFL, k-NNL, k-CNN, and k-TNN [31–37], which approved the superiority of the HLKNN method over other variants in terms of accuracy metric in different datasets, including balance-scale, dermatology, ecoli, habermans-survival, heart, iris, liver-disorders, segment, and wine.

The main outcomes of this study are summarized as follows:

i. The experimental results revealed the effectiveness of HLKNN compared to the KNN algorithm for 26 out of 32 datasets in terms of classification accuracy.

ii. On average, the HLKNN outperformed KNN with accuracy values of 81.01% and 79.76%, respectively.

iii. In terms of the precision metric, HLKNN (0.8129) performed better than KNN (0.7611) with an improvement of over 5% on average.

iv. In terms of the recall metric, the results revealed that HLKNN achieved equal or higher accuracy rates than KNN in 26 of 32 datasets.

v. In terms of the f-score metric, HLKNN (0.8111) achieved better performance than KNN (0.7779) with an improvement of over 3% on average.

vi. The proposed model in this study was also compared with various KNN-based methods, including uniform kNN, weighted kNN, kNNGNN, k-MMS, k-MRS, k-NCN, NN, k-NFL, k-NNL, k-CNN, and k-TNN, and the superiority of HLKNN over its counterparts was approved.

It is noteworthy to mention that the HLKNN method has several potential limitations. It is presented for classification tasks, not regression. Additionally, HLKNN depends on the existence of suitable neighbors of neighbors, in contrast to the original KNN method, which relies on only neighbors of a data point. In addition, HLKNN has a longer runtime compared to the KNN algorithm since it involves additional processing time for handling neighbor–neighbor relationships. We measured the runtime of both KNN and HLKNN over 32 well-known datasets by utilizing an Intel® system with properties of Core™ i7-8650U CPU 1.90 GHz and 8.00 GB memory. On average, we obtained runtimes of 0.31002 s and 2.60809 s, respectively, for the KNN and HLKNN algorithms. However, in comparison to the advantages (handling the noise problem, improving accuracy and generalization ability) of HLKNN, the drawback of higher computational cost can be reasonable in principle. Furthermore, in our study, the emphasis is given to the reliability of the methodology rather than running time. Moreover, the recent developments in high-performance computing permit alleviating this constraint by allowing the parallel execution of the proposed method. The main advantages and limitations of HLKNN contribute to making informed decisions on the model selection and implementation.

In future work, the HLKNN method can be applied to different datasets (e.g., sentiment analysis, human activity recognition, and predictive maintenance datasets) with different numbers of instances generated via real-world events or synthetically produced by artificial methods with the aim of achieving improved accuracies in terms of various evaluation metrics for a wide range of machine learning applications. In addition, it is conceivable to extend the HLKNN for the regression task with regard to various approaches, involving modifications of the model architecture, output interpretation, and data handling techniques.

Abbreviations

The following abbreviations are used in this paper.

AI	Artificial intelligence
BICS	Blockchain-based integrity checking system
CSC	Cloud service consumers
CSP	Cloud service providers
DTR	Decision tree regression
GIS	Geographical information systems
HBoF	Hybrid bag of features
HLKNN	High-level k-nearest neighbors
IDS	Intrusion detection system
KNN	K-nearest neighbors
k-CNN	K-center nearest neighbor
k-MMS	K-min–max sum
k-MRS	k-Min ranking sum
k-NCN	K-nearest centroid neighborhood
k-NFL	K-nearest feature line
k-NNGNN	K-nearest neighbors learning method based on a graph neural network
k-NNL	K-nearest neighbor line
k-TNN	K-tunable nearest neighbor
LSTM	Long short-term memory
ML	Machine learning
NN	Nearest neighbor rule
PCA	Principal component analysis
RF	Random forest
RSL	Random substance learning
SC	Smart contract
SDN	Software-defined network
SLA	Service level agreements
SVR	Support vector regression
UAV	Unmanned aerial vehicles

Appendix A

Algorithm A1 High-Level K-Nearest Neighbors (HLKNN)

Inputs:

D_{Train}: the training set with n data points

D_{Test}: test set

k: the number of the nearest neighbors

Outputs:

P: predicted labels for the testing set

Begin
    foreach $x \in D_{Test}$ **do**
        $N_k(x) =$ kNearestNeighbors (D_{Train}, x, k) # k-nearest neighbors of x
        $C = \varnothing$
        foreach $s \in N_k(x)$ **do**
          $y = Class\ (s)$ # class label of the low-level neighborhood
          $C = C \cup y$
          $NN_k(s) =$ kNearestNeighbors (D_{Train}, s, k) # neighbors of the neighbors
          foreach $o \in NN_k(s)$
            $y = Class\ (o)$ # class label of the high-level neighborhood
            $C = C \cup y$
          end foreach
        end foreach
        $c = \underset{y \in Y}{argmax} \sum_{i:y=H_i(x)}^{t} 1$ # prediction after the majority voting
        $P = P \cup c$
    end foreach
End Algorithm

References

1. Jiang, Y.; Li, X.; Luo, H.; Yin, S.; Kaynak, O. Quo Vadis Artificial Intelligence? *Discov. Artif. Intell.* **2022**, *2*, 629869. [CrossRef]
2. Janiesch, C.; Zschech, P.; Heinrich, K. Machine Learning and Deep Learning. *Electron. Mark.* **2021**, *31*, 685–695. [CrossRef]
3. Sarker, I.H. Machine Learning: Algorithms, Real-World Applications and Research Directions. *SN Comput. Sci.* **2021**, *2*, 160. [CrossRef] [PubMed]
4. Han, J.; Pei, J.; Tong, H. *Data Mining: Concepts and Techniques*, 4th ed.; Morgan Kaufmann: Burlington, MA, USA, 2022.
5. Ahmad, S.R.; Bakar, A.A.; Yaakub, M.R.; Yusop, N.M.M. Statistical validation of ACO-KNN algorithm for sentiment analysis. *J. Telecommun. Electron. Comput. Eng.* **2017**, *9*, 165–170.
6. Kramer, O. *Dimensionality Reduction with Unsupervised Nearest Neighbors*; Springer: Berlin/Heidelberg, Germany, 2013; pp. 13–23.
7. Hasan, M.J.; Kim, J.; Kim, C.H.; Kim, J.-M. Health State Classification of a Spherical Tank Using a Hybrid Bag of Features and K Nearest Neighbor. *Appl. Sci.* **2020**, *10*, 2525. [CrossRef]
8. Beskopylny, A.N.; Stelmakh, S.A.; Shcherban, E.M.; Mailyan, L.R.; Meskhi, B.; Razveeva, I.; Chernilnik, A.; Beskopylny, N. Concrete Strength Prediction Using Machine Learning Methods CatBoost, k-Nearest Neighbors, Support Vector Regression. *Appl. Sci.* **2022**, *12*, 10864. [CrossRef]
9. Wang, J.; Zhou, Z.; Li, Z.; Du, S. A Novel Fault Detection Scheme Based on Mutual k-Nearest Neighbor Method: Application on the Industrial Processes with Outliers. *Processes* **2022**, *10*, 497. [CrossRef]
10. Lu, J.; Qian, W.; Li, S.; Cui, R. Enhanced K-Nearest Neighbor for Intelligent Fault Diagnosis of Rotating Machinery. *Appl. Sci.* **2021**, *11*, 919. [CrossRef]
11. Salem, H.; Shams, M.Y.; Elzeki, O.M.; Abd Elfattah, M.; Al-Amri, J.F.; Elnazer, S. Fine-Tuning Fuzzy KNN Classifier Based on Uncertainty Membership for the Medical Diagnosis of Diabetes. *Appl. Sci.* **2022**, *12*, 950. [CrossRef]
12. Miron, M.; Moldovanu, S.; Ștefănescu, B.I.; Culea, M.; Pavel, S.M.; Culea-Florescu, A.L. A New Approach in Detectability of Microcalcifications in the Placenta during Pregnancy Using Textural Features and K-Nearest Neighbors Algorithm. *J. Imaging* **2022**, *8*, 81. [CrossRef]
13. Rattanasak, A.; Uthansakul, P.; Uthansakul, M.; Jumphoo, T.; Phapatanaburi, K.; Sindhupakorn, B.; Rooppakhun, S. Real-Time Gait Phase Detection Using Wearable Sensors for Transtibial Prosthesis Based on a kNN Algorithm. *Sensors* **2022**, *22*, 4242. [CrossRef]
14. Nguyen, L.V.; Vo, Q.-T.; Nguyen, T.-H. Adaptive KNN-Based Extended Collaborative Filtering Recommendation Services. *Big Data Cogn. Comput.* **2023**, *7*, 106. [CrossRef]
15. Corso, M.P.; Perez, F.L.; Stefenon, S.F.; Yow, K.-C.; García Ovejero, R.; Leithardt, V.R.Q. Classification of Contaminated Insulators Using *k*-Nearest Neighbors Based on Computer Vision. *Computers* **2021**, *10*, 112. [CrossRef]
16. Syamsuddin, I.; Barukab, O.M. SUKRY: Suricata IDS with Enhanced kNN Algorithm on Raspberry Pi for Classifying IoT Botnet Attacks. *Electronics* **2022**, *11*, 737. [CrossRef]
17. Derhab, A.; Guerroumi, M.; Gumaei, A.; Maglaras, L.; Ferrag, M.A.; Mukherjee, M.; Khan, F.A. Blockchain and Random Subspace Learning-Based IDS for SDN-Enabled Industrial IoT Security. *Sensors* **2019**, *19*, 3119. [CrossRef]
18. Liu, G.; Zhao, H.; Fan, F.; Liu, G.; Xu, Q.; Nazir, S. An Enhanced Intrusion Detection Model Based on Improved kNN in WSNs. *Sensors* **2022**, *22*, 1407. [CrossRef] [PubMed]
19. Zheng, Q.; Wang, L.; He, J.; Li, T. KNN-Based Consensus Algorithm for Better Service Level Agreement in Blockchain as a Service (BaaS) Systems. *Electronics* **2023**, *12*, 1429. [CrossRef]
20. Fan, G.-F.; Guo, Y.-H.; Zheng, J.-M.; Hong, W.-C. Application of the Weighted K-Nearest Neighbor Algorithm for Short-Term Load Forecasting. *Energies* **2019**, *12*, 916. [CrossRef]

21. Lee, C.-Y.; Huang, K.-Y.; Shen, Y.-X.; Lee, Y.-C. Improved Weighted *k*-Nearest Neighbor Based on PSO for Wind Power System State Recognition. *Energies* **2020**, *13*, 5520. [CrossRef]

22. Gajan, S. Modeling of Seismic Energy Dissipation of Rocking Foundations Using Nonparametric Machine Learning Algorithms. *Geotechnics* **2021**, *1*, 534–557. [CrossRef]

23. Martínez-Clark, R.; Pliego-Jimenez, J.; Flores-Resendiz, J.F.; Avilés-Velázquez, D. Optimum k-Nearest Neighbors for Heading Synchronization on a Swarm of UAVs under a Time-Evolving Communication Network. *Entropy* **2023**, *25*, 853. [CrossRef] [PubMed]

24. Cha, G.-W.; Choi, S.-H.; Hong, W.-H.; Park, C.-W. Developing a Prediction Model of Demolition-Waste Generation-Rate via Principal Component Analysis. *Int. J. Environ. Res. Public Health* **2023**, *20*, 3159. [CrossRef]

25. Bullejos, M.; Cabezas, D.; Martín-Martín, M.; Alcalá, F.J. A K-Nearest Neighbors Algorithm in Python for Visualizing the 3D Stratigraphic Architecture of the Llobregat River Delta in NE Spain. *J. Mar. Sci. Eng.* **2022**, *10*, 986. [CrossRef]

26. Bullejos, M.; Cabezas, D.; Martín-Martín, M.; Alcalá, F.J. Confidence of a k-Nearest Neighbors Python Algorithm for the 3D Visualization of Sedimentary Porous Media. *J. Mar. Sci. Eng.* **2023**, *11*, 60. [CrossRef]

27. Zhang, L.; Zhu, Y.; Su, J.; Lu, W.; Li, J.; Yao, Y. A Hybrid Prediction Model Based on KNN-LSTM for Vessel Trajectory. *Mathematics* **2022**, *10*, 4493. [CrossRef]

28. Park, J.; Oh, J. Analysis of Collected Data and Establishment of an Abnormal Data Detection Algorithm Using Principal Component Analysis and K-Nearest Neighbors for Predictive Maintenance of Ship Propulsion Engine. *Processes* **2022**, *10*, 2392. [CrossRef]

29. Tamamadin, M.; Lee, C.; Kee, S.-H.; Yee, J.-J. Regional Typhoon Track Prediction Using Ensemble k-Nearest Neighbor Machine Learning in the GIS Environment. *Remote Sens.* **2022**, *14*, 5292. [CrossRef]

30. Mallek, A.; Klosa, D.; Büskens, C. Impact of Data Loss on Multi-Step Forecast of Traffic Flow in Urban Roads Using K-Nearest Neighbors. *Sustainability* **2022**, *14*, 11232. [CrossRef]

31. Kang, S. *k*-Nearest Neighbor Learning with Graph Neural Networks. *Mathematics* **2021**, *9*, 830. [CrossRef]

32. Mazón, J.N.; Micó, L.; Moreno-Seco, F. New Neighborhood Based Classification Rules for Metric Spaces and Their Use in Ensemble Classification. In Proceedings of the IbPRIA 2007 on Pattern Recognition and Image Analysis, Girona, Spain, 6–8 June 2007; pp. 354–361. [CrossRef]

33. Sánchez, J.S.; Pla, F.; Ferri, F.J. On the Use of Neighbourhood-Based Non-Parametric Classifiers. *Pattern Recognit. Lett.* **1997**, *18*, 1179–1186. [CrossRef]

34. Duda, R.O.; Hart, P.E.; Stork, D.G. *Pattern Classification*, 2nd ed.; John Wiley & Sons: New York, NY, USA, 2007; pp. 1–654. [CrossRef]

35. Orozco-Alzate, M.; Castellanos-Domínguez, C.G. Comparison of the Nearest Feature Classifiers for Face Recognition. *Mach. Vis. Appl.* **2006**, *17*, 279–285. [CrossRef]

36. Lou, Z.; Jin, Z. Novel Adaptive Nearest Neighbor Classifiers Based on Hit-Distance. In Proceedings of the 18th International Conference on Pattern Recognition (ICPR'06), Hong Kong, China, 20–24 August 2006; pp. 87–90. [CrossRef]

37. Altınçay, H. Improving the K-Nearest Neighbour Rule: Using Geometrical Neighbourhoods and Manifold-Based Metrics. *Expert Syst.* **2011**, *28*, 391–406. [CrossRef]

38. Witten, I.H.; Frank, E.; Hall, M.A. *Data Mining: Practical Machine Learning Tools and Techniques*, 3rd ed.; Morgan Kaufmann: Cambridge, MA, USA, 2016; pp. 1–664.

39. Aha, D.; Kibler, D. Instance-based learning algorithms. *Mach. Learn.* **1991**, *6*, 37–66. [CrossRef]

40. Nica, I.; Alexandru, D.B.; Crăciunescu, S.L.P.; Ionescu, Ș. Automated Valuation Modelling: Analysing Mortgage Behavioural Life Profile Models Using Machine Learning Techniques. *Sustainability* **2021**, *13*, 5162. [CrossRef]

41. Kelly, M.; Longjohn, R.; Nottingham, K. The UCI Machine Learning Repository. Available online: https://archive.ics.uci.edu (accessed on 28 July 2023).