applied sciences

MDPI

Article

Integrating Interpolation and Extrapolation: A Hybrid Predictive Framework for Supervised Learning

Bo Jiang [1,†], Xinyi Zhu [2,*,†], Xuecheng Tian [3,†], Wen Yi [4] and Shuaian Wang [3]

[1] Institute of Data and Information, Shenzhen International Graduate School, Tsinghua University, Shenzhen 518055, China; jb22@mails.tsinghua.edu.cn
[2] Department of Aeronautical and Aviation Engineering, The Hong Kong Polytechnic University, Hung Hom, Hong Kong
[3] Faculty of Business, The Hong Kong Polytechnic University, Hung Hom, Hong Kong; xuecheng-simon.tian@connect.polyu.hk (X.T.); hans.wang@polyu.edu.hk (S.W.)
[4] Faculty of Construction and Environment, The Hong Kong Polytechnic University, Hung Hom, Hong Kong; wen.yi@polyu.edu.hk
* Correspondence: xinyi1998.zhu@connect.polyu.hk
† These authors contributed equally to this work.

Abstract: In the domain of supervised learning, interpolation and extrapolation serve as crucial methodologies for predicting data points within and beyond the confines of a given dataset, respectively. The efficacy of these methods is closely linked to the nature of the dataset, with increased challenges when multivariate feature vectors are handled. This paper introduces a novel prediction framework that integrates interpolation and extrapolation techniques. Central to this method are two main innovations: an optimization model that effectively classifies new multivariate data points as either interior or exterior to the known dataset, and a hybrid prediction system that combines *k*-nearest neighbor (*k*NN) and linear regression. Tested on the port state control (PSC) inspection dataset at the port of Hong Kong, our framework generally demonstrates superior precision in predictive outcomes than traditional *k*NN and linear regression models. This research enriches the literature by illustrating the enhanced capability of combining interpolation and extrapolation techniques in supervised learning.

Keywords: interpolation; extrapolation; *k*-nearest neighbor (*k*NN); linear regression; ship deficiency prediction

Citation: Jiang, B.; Zhu, X.; Tian, X.; Yi, W.; Wang, S. Integrating Interpolation and Extrapolation: A Hybrid Predictive Framework for Supervised Learning. *Appl. Sci.* **2024**, *14*, 6414. https://doi.org/10.3390/app14156414

Academic Editor: Douglas O'Shaughnessy

Received: 21 June 2024
Revised: 19 July 2024
Accepted: 20 July 2024
Published: 23 July 2024

1. Introduction

Consider a dataset $S_N = \{(x_i, y_i) : i = 1, \ldots, N\}$, where $x_i \in X \subseteq \mathbb{R}^p$ is an input feature vector, p is the dimension of x_i, and $y_i \in Y \subseteq \mathbb{R}$ is a corresponding univariate response variable. Assuming that the data points (x_i, y_i) are independently drawn and identically distributed from a probability space $X \times Y$, the goal of supervised learning is to learn a function f that bridges this finite dataset S_N to the encompassing space $X \times Y$. Once f is well estimated, it can be used to predict the conditional mean of the response variable for a new feature vector x_0 through $f(x_0)$.

1.1. Interpolation and Extrapolation

Interpolation and extrapolation serve as two primary frameworks in supervised learning algorithms ranging from function approximation to deep learning [1], with applications spanning engineering [2], science [3], economics [4], and statistics [5]. Interpolation predicts a new sample's target value based on known data points within a specified range [6]. Minda et al. [7] compared the most common interpolation methods. It is vital to note that to make interpolation applicable, the new observation must lie within the known sample space. For instance, *k*-nearest neighbor (*k*NN) is a good example method of interpolation.

It finds a sample's nearest neighbors in a local subspace that centers around the sample under the defined distance metric (e.g., Euclidean distance). As shown in Figure 1, the target value of a new observation x_0, which falls within the known data range, can be approximated as $f^{\text{in}}(x_0) = [f(x_1) + f(x_2) + f(x_3)]/3$, where x_1, x_2, and x_3 are the three nearest training points to x_0. For interpolation, a critical consideration is selecting the appropriate interpolation function. Over the years, various functions have emerged, including linear interpolation, polynomial interpolation, and spline interpolation [8], which have varying properties including accuracy, computational cost, data point requirements, and functional smoothness. Linear interpolation, while simple and fast, may not capture complex relationships between features and targets [9]. Polynomial interpolation, utilizing the lowest-degree polynomial to fit all data points, includes methods like Newton and Hermit interpolations [10]. Challu et al. [11] proposed neural hierarchical interpolation for time series (NHITS), a model integrating hierarchical interpolation and multi-rate data sampling methods. Sekulić et al. [12] investigated the significance of incorporating observations from the nearest locations along with their distances from the target prediction site through the implementation of random forest spatial interpolation (RFSI). Although polynomial interpolation can offer higher precision than linear interpolation, it is computationally intensive and may exhibit oscillations. Nearest-neighbor interpolation, a zero-order polynomial interpolation, assigns the value of an interpolated point based on its nearest existing data point(s).

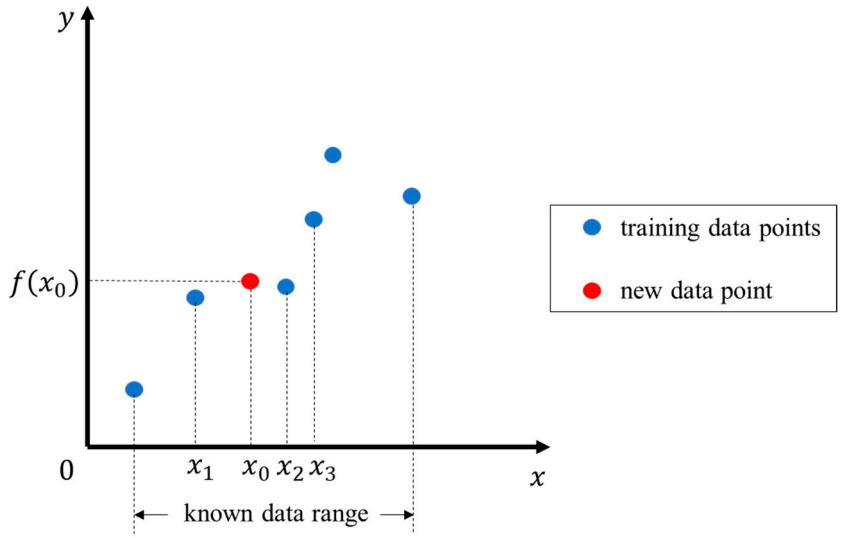

Figure 1. An illustrative example of interpolation.

Extrapolation is inherently more challenging than interpolation, as it predicts outside the known data space. Linear extrapolation posits a linear relationship between features and targets, offering simplicity but sometimes missing underlying distribution complexities. As shown in Figure 2a, using a function f^{ex} fitted for known points via linear extrapolation techniques, the target value of the new observation x_0, which falls outside the known data space, can be approximated as $f^{\text{ex}}(x_0)$. Polynomial extrapolation can fit non-linear data effectively, as shown in Figure 2b. Selecting the appropriate extrapolation method requires understanding the data's inherent characteristics, e.g., whether they are continuous, smooth, or periodic. Incorporating domain knowledge often proves valuable for extrapolations [13]. Webb et al. [14] addressed the challenge of learning representations that facilitate extrapolation and proposed a novel visual analogy benchmark that enables a graded assessment of extrapolation based on the distance from the convex domain defined by the training dataset. Zhu et al. [15] systematically explored the extrapolation behavior of deep operator networks through rigorous quantification of extrapolation complexity and proposed a novel strategy involving bias–variance trade-off for extrapolation.

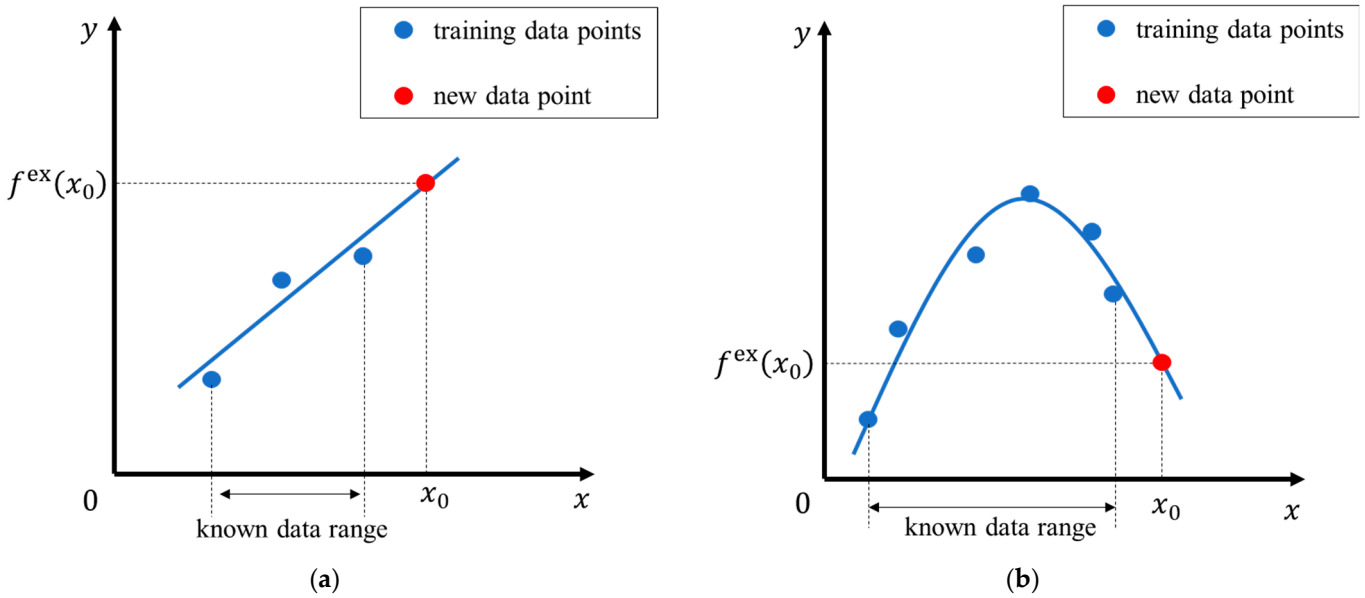

Figure 2. Illustrative examples of extrapolation: (**a**) linear extrapolation; (**b**) polynomial extrapolation.

The effectiveness of extrapolation relies on the assumption about the functional form [16]. In Figure 3, which illustrates three known data points (from x_1 to x_3), the true curve is a third-order polynomial (solid black line), but the polynomial extrapolation wrongly assumes a quadratic polynomial curve (black dashed line). This underscores that extrapolation is inherently uncertain, with a heightened risk of yielding misleading results. Such issues are optimally mitigated when the functional forms assumed by the extrapolation technique closely mirror the underlying nature of the data.

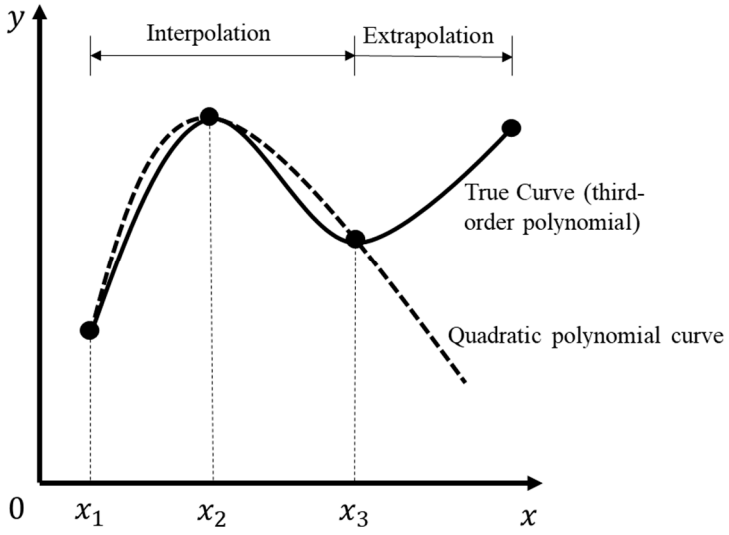

Figure 3. The effectiveness of extrapolation with different functional forms.

Interpolation and extrapolation can be viewed as linear approximation methods within the unit disk of the complex plane [17]. The most effective methods identified for interpolation and extrapolation include widely adopted techniques such as cubic spline interpolation and Gaussian processes regression [18]. Rosenfeld et al. [19] provided a rigorous demonstration that extrapolation poses significantly greater computational challenges than interpolation based on reweighting of sub-group likelihoods, while the statistical complexity remains relatively unchanged.

Interpolation and extrapolation, while serving distinct roles, are both crucial for making predictions from data. Interpolation is primarily employed to fill gaps in existing records, acting as a bridge to seamlessly integrate missing data within known boundaries, and *k*NN serves as a predictive model with good interpolation abilities. On the other hand, extrapolation goes beyond these bounds, making predictions for entirely new observations based on the trends and patterns identified in the existing dataset, and linear regression serves as a predictive model with good extrapolation abilities. The accuracy and efficacy of these methods, however, are heavily influenced by the context in which they are used.

When working with a univariate feature variable, classifying a new data point as either interior or exterior to the known dataset is relatively straightforward. If the point falls within the dataset's range, interpolation is the method of choice. Conversely, if it lies outside this range, extrapolation should be employed. However, the task becomes more difficult with a multivariate feature vector. In such cases, the task of determining whether a new data point is interior or exterior to the existing feature space grows complex. The presence of multiple dimensions can lead to scenarios where a point might be deemed interior in one feature dimension but exterior in another. Consequently, this complexity gives rise to a pressing research question: *How can the intricacies of multivariate data be effectively dealt with by leveraging the strengths of both interpolation and extrapolation while mitigating their limitations?*

1.2. Contributions and Organization

To address the above research question, we establish a mathematical programming model to classify whether a new multivariate data point is interior or exterior to the known dataset. By solving the established optimization model, we obtain the defined centrality coefficient of the new data point. Accordingly, we propose a novel hybrid prediction framework that integrates both interpolation and extrapolation methods by taking advantage of the centrality coefficient. If the new observation is an interior point to the known dataset, we can use prediction methods with good interpolation abilities, such as *k*NN. Otherwise, we can use prediction methods with good extrapolation abilities, such as linear regression. Consequently, our hybrid prediction framework takes advantage of both interpolation and extrapolation abilities.

Our framework distinguishes itself from the existing interpolation and extrapolation methods in several ways:

1. It can handle both interior and exterior data points without prior knowledge or assumptions;
2. It flexibly selects the optimal prediction strategy by considering the centrality coefficient obtained from the optimization model;
3. It enhances the precision of predictions by harnessing the collective power of both interpolation and extrapolation abilities.

As a practical application, we harness our framework to address the ship deficiency prediction problem using the port state control (PSC) inspection dataset for the port of Hong Kong. A comparative analysis against the simple uses of *k*NN and linear regression reveals that our model excels in specific scenarios. This paper, therefore, stands as a valuable addition to the literature, offering a refreshed and effective method that melds the advantages of both interpolation and extrapolation.

The remainder of this paper is organized as follows. Section 2 presents our optimization model for classifying exterior or interior data points and describes our hybrid framework combining *k*NN and linear regression. Section 3 describes the numerical experiments within the considered case study, focusing on ship deficiency prediction. Section 4 concludes our paper and suggests future research directions.

2. Problem Statement and Model Setup

In this section, we introduce a prediction framework that integrates interpolation and extrapolation. In Section 2.1, we develop a mathematical optimization model to determine

the centrality coefficient of new data points. Following that, Section 2.2 outlines our hybrid approach, which merges kNN and linear regression.

2.1. Optimization Model M0

In our predictive framework, a pivotal decision revolves around classifying a new observation as either an interior or an exterior point relative to the known dataset. This distinction is important for choosing which method to use for prediction. In this paper, the kNN model is used for predicting interior points by interpolation, and linear regression is used for predicting exterior points by extrapolation. Consequently, pinpointing this classification is of paramount importance.

To this end, we introduce the optimization model M0. Model M0 harnesses a linear programming model to decisively categorize the new observation. Its core principle is to compute the shortest distance between the new observation and the known dataset's convex hull, which represents the smallest convex set encompassing all dataset points. A zero minimum distance suggests that the observation resides within the convex hull, designating it as an interior point. Conversely, a positive minimum distance indicates the observation's position outside the convex hull, categorizing it as an exterior point. Table 1 shows all of the notations needed in the optimization model.

Table 1. Notations in the optimization model.

Parameters	
$S_N = \{(x_i, y_i) : i = 1, \ldots, N\}$	A dataset with N known samples, $x_i \in X \subseteq \mathbb{R}^p$, $y_i \in Y \subseteq \mathbb{R}$, where p is the dimension of input feature vector x_i.
$x_0 = \left(x_0^1, \ldots, x_0^p\right)$	A new data point.
Decision variables	
$\lambda(x_0) = [\lambda_1(x_0), \ldots, \lambda_N(x_0)]$	The weighted vector for the new data point x_0, where $\lambda_i(x_0)$ represents the weight of data point $(x_i, y_i)(i \in \{1, \ldots, N\})$ in relation to the new data point.

To ascertain the distance between a new observation x_0 and the known dataset's convex hull, we formulate the optimization model, termed M0, as follows:

$$\min_{\lambda(x_0)} \left| \sum_{i=1}^{N} \lambda_i(x_0) x_i - x_0 \right| \tag{1}$$

subject to

$$\sum_{i=1}^{N} \lambda_i(x_0) = 1 \tag{2}$$

$$\lambda_i(x_0) \geq 0, \forall i \in \{1, \ldots, N\}. \tag{3}$$

Objective Function (1) aims to minimize the Manhattan distance between x_0 and the convex hull defined by the known dataset. The optimal value of this objective function is termed the centrality coefficient, gauging the proximity of the new data point relative to the dataset. Constraint (2) ensures the sum of weights for all of the known data points equals one. Constraints (3) define the domains of the decision variables.

Defining $\lambda^*(x_0) = \left[\lambda_1^*(x_0), \ldots, \lambda_N^*(x_0)\right]$ as the optimal solution obtained by solving the optimization model, we have

$$d^*(x_0) = \left| \sum_{i=1}^{N} \lambda_i^*(x_0) x_i - x_0 \right|,$$

which denotes the centrality coefficient of x_0 and serves as the deciding factor for choosing between interpolation and extrapolation methods for x_0. If $d^*(x_0) = 0$, this represents that the new data point is an interior point with respect to the known dataset. If $d^*(x_0) > 0$, this represents that the new data point is not an interior point to the given dataset; that is, the new data point has an exterior nature relative to the given dataset.

Before applying our optimization model to classify the new observation as exterior or interior, we need to tackle two challenges. First, numerical features are naturally integrated into our optimization model without the need for additional processing. However, categorical features pose a unique challenge. These categorical data points do not have a clear numerical relationship, making it challenging to directly measure the categorical distance in our model. Second, the current form of our objective function is non-linear due to the absolute term, which can be challenging to handle for many commercial solvers. To resolve this, we need to convert it into an equivalent linear form.

2.1.1. Pre-Processing Procedures for Categorical Features

Given the feature sets, we assume that the index set of categorical features is denoted by C, and the index set of numerical features is denoted by U; thus, we have $|C| + |U| = p$. Our challenge revolves around appropriately integrating categorical features c $(c \in C)$ into the optimization model, while numerical features u $(u \in U)$ do not pose such a concern.

To accommodate the categorical features, one-hot encoding is employed. Essentially, this process takes a categorical feature and produces binary columns for each category of the feature. Assume that feature c $(c \in C)$ is a categorical feature with l_c categories; for x_i $(i \in \{0, 1, \ldots, N\})$ (note that we simultaneously consider known data samples and the new observation), we can obtain an l_c-dimensional column vector $x_i^c = \left(x_i^{c,1}, \ldots, x_i^{c,l_c}\right)$ with binary components $x_i^{c,k}$, $i \in \{0, 1, \ldots, N\}$, $k \in \{1, \ldots, l_c\}$. Specifically, we set the values of the binary components $x_i^{c,k}$, $i \in \{0, 1, \ldots, N\}$, $k \in \{1, \ldots, l_c\}$ to 0.5 or 0. This indicates that if x_i $(i \in \{1, \ldots, N\})$ and x_0 differ in the categorical feature c $(c \in C)$, the term $\sum_{k=1}^{l_c} \left| x_i^{c,k} - x_0^{c,k} \right|$ equals one; otherwise, it equals 0. The reason for choosing 0.5 as the binary value is further explained in Example 1. Therefore, for feature c $(c \in C)$, we modify the counterpart in the objective function as $\sum_{i=1}^{N} \lambda_i(x_0) \sum_{k=1}^{l_c} \left| x_i^{c,k} - x_0^{c,k} \right|$.

Example 1. *Suppose that the data points only have one categorical feature and we have adopted the binary processing procedure for all data points. There is a new data point x_0 whose category is different from the categories of all of the known data points. Therefore, we obtain $d^*(x_0) = \sum_{i=1}^{N} \lambda_i(x_0) \sum_{k=1}^{l_c} \left| x_i^{c,k} - x_0^{c,k} \right| = \sum_{i=1}^{N} \lambda_i(x_0) = 1$ because $\sum_{k=1}^{l_c} \left| x_i^{c,k} - x_0^{c,k} \right| = 1$, $\forall i \in \{1, \ldots, N\}$, which means that the new data point is an exterior point of the known data points. However, suppose that the categories of all of the known data points are identical and the new data point has the same category. Then, we obtain $d^*(x_0) = \sum_{i=1}^{N} \lambda_i(x_0) \sum_{k=1}^{l_c} \left| x_i^{c,k} - x_0^{c,k} \right| = 0$ because $\sum_{k=1}^{l_c} \left| x_i^{c,k} - x_0^{c,k} \right| = 0, \forall i \in \{1, \ldots, N\}$, which means that this new observation is a point interior to the original dataset. From this example, it is clear that coding the categorical feature by 0.5 and 0 ensures that the $\sum_{k=1}^{l_c} \left| x_i^{c,k} - x_0^{c,k} \right|$ part is 1 or 0, effectively measuring the categorical differences.*

Therefore, Objective Function (1) can be transformed into

$$\min_{\lambda(x_0)} \sum_{i=1}^{N} \lambda_i(x_0) \sum_{c \in C} \sum_{k=1}^{l_c} \left| x_i^{c,k} - x_0^{c,k} \right| + \sum_{u \in U} \left| \sum_{i=1}^{N} \lambda_i(x_0) x_i^u - x_0^u \right|, \tag{4}$$

where the features are divided into categorical features c $(c \in C)$ and numerical features u $(u \in U)$.

2.1.2. The Linearization of Absolute Terms

Objective Function (4) is non-linear due to the absolute terms. To linearize them, we introduce auxiliary non-negative decision variables for categorical and numerical features, respectively. For categorical features c $(c \in C)$, we define

$$z_i^{c,k} = \left| x_i^{c,k} - x_0^{c,k} \right|, \quad c \in C,\ i \in \{1,\ldots,N\}, k \in \{1,\ldots,l_c\},$$

and add the following constraints:

$$x_i^{c,k} - x_0^{c,k} \le z_i^{c,k}, \quad c \in C,\ i \in \{1,\ldots,N\}, k \in \{1,\ldots,l_c\}, \tag{5}$$

$$-\left(x_i^{c,k} - x_0^{c,k}\right) \le z_i^{c,k}, \quad c \in C,\ i \in \{1,\ldots,N\}, k \in \{1,\ldots,l_c\}. \tag{6}$$

For numerical features u $(u \in U)$, we define

$$z_u = \left| \sum_{i=1}^{N} \lambda_i(x_0) x_i^u - x_0^u \right|, \quad u \in U,$$

and add the following constraints:

$$\sum_{i=1}^{N} \lambda_i(x_0) x_i^u - x_0^u \le z_u, \quad u \in U, \tag{7}$$

$$-\left(\sum_{i=1}^{N} \lambda_i(x_0) x_i^u - x_0^u \right) \le z_u, \quad u \in U. \tag{8}$$

After linearization, we obtain the following linear optimization model:

Model M0:

$$\min_{\lambda(x_0)} \sum_{i=1}^{N} \lambda_i(x_0) \sum_{c \in C} \sum_{k=1}^{l_c} z_i^{c,k} + \sum_{u \in U} z_u \tag{9}$$

subject to Constraints (2)–(3) and (5)–(8).

This model can be directly solved by commercial solvers (e.g., Gurobi and Cplex) to obtain $d^*(x_0)$ for each new data point. Next, we discuss how to choose the best prediction method for new observations based on their centralities to the original dataset.

2.2. Predictive Frameworks

In this section, we first introduce two traditional prediction models, *k*NN and linear regression, which are then integrated into our proposed hybrid prediction framework.

2.2.1. Model M1: *k*NN

Consider a training dataset S_N and a new observation with feature vector x_0. We predict the output value of the new observation by calculating the average value of the k-nearest training points to x_0 under the defined distance metric (e.g., Euclidean distance). Mathematically, the output value of the new observation using *k*NN could be defined as

Model M1:

$$f^{k\mathrm{NN}}(x_0) = \frac{1}{k} \sum_{i \in \mathcal{N}_k(x_0)} y_i, \tag{10}$$

where $\mathcal{N}_k(x_0) = \left\{ i = 1,\ldots,N : \sum_{j=1}^{N} \mathbb{I}\left[\|x_0 - x_i\| \ge \|x_0 - x_j\| \right] \le k \right\}$ is the neighborhood set of the k-nearest data points to x_0. Here, $\|\cdot\|$ denotes the Euclidean norm. k is a hyperparameter, and the optimal value k^* can be determined by validation.

2.2.2. Model M2: Linear Regression

We assume that the target value of x_i ($i \in \{1, \ldots, N\}$) can be predicted using a linear regression model given by

$$\hat{y}_i = \omega^{\mathrm{T}} x_i + b,$$

where ω is the weighted vector and b is a bias. The parameters ω and b can be obtained by minimizing the sum of squared errors between the predicted values and actual values of known data points, shown as follows:

$$(\omega^*, b^*) \in \underset{(\omega,b)}{\operatorname{argmin}} \sum_{i=1}^{N} (\hat{y}_i - y_i)^2.$$

Hence, for a new data point x_0, the predicted value can be obtained by

Model M2:

$$f^{\mathrm{LR}}(x_0) = \hat{y}_0 = (\omega^*)^{\mathrm{T}} x_0 + b^*. \tag{11}$$

2.2.3. Model M3: The Hybrid Prediction Model

As mentioned above, *k*NN serves as a predictive model with good interpolation abilities, while linear regression serves as a predictive model with good extrapolation abilities. However, the accuracy and efficacy of these two methods are heavily influenced by the underlying data structure. Therefore, we propose a hybrid prediction framework leveraging the strengths of both interpolation and extrapolation while mitigating their limitations.

The centrality of a new observation to the original dataset is paramount in selecting the most appropriate predictive approach for it. This centrality, determined by our previous optimization model M0, classifies observations as either interior or exterior to the known dataset. If a new observation is a point interior to the known dataset, we can use prediction methods with good interpolation abilities, such as *k*NN. If a new observation is an exterior point but is not too far away from the known dataset, we may combine different prediction methods by leveraging both interpolation and extrapolation abilities. If a new observation is an exterior point and is significantly far away from the known dataset, we can use prediction methods with good extrapolation abilities, such as linear regression.

By solving model M0, we know whether the new observation x_0 is exterior or interior to the given dataset. Taking the advantages of *k*NN and linear regression, we propose a hybrid prediction framework to improve prediction quality. Let us consider the prediction outcomes from the two predictive models as $f^{k\mathrm{NN}}(x_0)$ (from *k*NN) and $f^{\mathrm{LR}}(x_0)$ (from linear regression). The hybrid framework computes a weighted average of these two outcomes as follows:

Model M3:

$$f^{\mathrm{hybrid}}(x_0) = (1 - \theta[d^*(x_0)]) f^{k\mathrm{NN}}(x_0) + \theta[d^*(x_0)] f^{\mathrm{LR}}(x_0), \tag{12}$$

where $\theta[d^*(x_0)]$ is a weighting parameter driven by the centrality coefficient $d^*(x_0)$ of x_0. The choice of $\theta[d^*(x_0)]$ reflects our confidence in either of the predictive models:

$$\theta[d^*(x_0)] = \begin{cases} 0, & \text{if } d^*(x_0) = 0 \\ \frac{d^*(x_0)}{h}, & \text{if } d^*(x_0) \in (0, h) \\ 1, & \text{if } d^*(x_0) \geq h. \end{cases} \tag{13}$$

Here, h is a hyperparameter determining the boundary across which we transition from *k*NN to linear regression, and the optimal parameter h^* can be determined by validation (to be introduced in Section 3.3). Theoretically, a bigger h means that M1 (i.e., *k*NN) plays a more important role in determining the final prediction outcome. Conversely, a smaller h

means that M2 (i.e., linear regression) plays a greater role in determining the final prediction outcome.

2.2.4. The Evaluation Metric

Suppose that the test dataset is defined as $T_M = \{(x_j, y_j) : j = N+1, \ldots, N+M\}$, where $x_j \in X \subseteq \mathbb{R}^p$, $y_j \in Y \subset \mathbb{R}$, and p is the dimension of input feature vector x_j. In order to evaluate the accuracy and efficacy of the different predictive models, we define the mean squared error (MSE) between the predictive values and the actual values as follows:

$$\text{MSE} = \frac{1}{M} \sum_{j=N+1}^{N+M} [f(x_j) - y_j]^2,$$

where $f(x_j)$ is the predictive value for a test data point x_j ($j \in \{N+1, \ldots, N+M\}$) through a specific predictive method, and y_j is the real target value for x_j ($j \in \{N+1, \ldots, N+M\}$). The smaller the MSE, the better the predictive performance of the model.

3. Numerical Experiments

We apply our framework to the ship deficiency prediction problem, using the PSC inspection dataset for the port of Hong Kong as a case study. We test the performance of our framework by comparing the prediction results with the *k*NN model and the linear regression model.

3.1. Dataset Description

To evaluate the effectiveness of the hybrid prediction framework, we use the ship deficiency prediction problem and the PSC inspection dataset as a case study, representing an essential issue for maritime transportation.

To identify ships that are potentially deficient or pose a higher detention risk for port authorities, the Tokyo Memorandum of Understanding (MoU) introduced a ship selection scheme in 2014, namely, the new inspection regime (NIR) [20], to evaluate the risk level of ships in Asia–Pacific regions. The NIR considers seven features related to the characteristics and historical inspection records of a ship, including ship type, ship age, ship flag performance, ship recognized organization (RO) performance, ship company performance, the number of deficiencies within the previous 36 months, and the number of detentions within the previous 36 months. Each candidate value of a certain feature is assigned a fixed weighting point, and a ship's risk level is determined by the sum of the seven features' weighting points.

The weighted-sum method introduced by the NIR assumes a straightforward additive relationship between the weighting points and ship deficiencies. However, as highlighted by Tian and Zhu [21], the reality is not that simple. The inherent interdependence between features, or the so-called coupling effect, makes a linear additive model like NIR potentially ineffective. This is where nonlinear models, especially machine learning models, come to the fore. As validated by Wang et al. [22], Bayesian network (BN) models have showcased superior performance in predicting ship deficiencies compared with the NIR's linear model. This, coupled with other noteworthy contributions by Dinis et al. [23], Yan and Wang [24], and Rao et al. [25], further solidifies the argument for a departure from simple linear models to more intricate, nonlinear models.

For our study, the goal is to predict the number of deficiencies for a ship. We rely on the seven features, as highlighted by the NIR, that are suspected to be in direct correlation with a ship's deficiency count. The dataset consists of PSC inspection records from January 2015 to December 2019, specifically from the port of Hong Kong. These records were sourced from the Tokyo MoU database. For the sake of data quality, incomplete records were excluded, resulting in a final dataset of 3026 inspection records.

3.2. Data Pre-Processing

The dataset contains a mix of categorical and numerical features. In this section, we encode the data of categorical features according to the method introduced in Section 2.1.1 and normalize the data of numerical features.

3.2.1. Encoding Method for Categorical Features

Categorical features in our dataset need special attention. For example, ship type includes six distinct categories. To numerically represent these categories, we adopt a one-hot encoding approach, encoding each category as a distinct binary vector. We encode each value as a one-hot vector, such as (0.5,0,0,0,0,0) for the first type, (0,0.5,0,0,0,0) for the second type, and so on. Table 2 provides a comprehensive overview of the encoding methodology for each categorical feature.

Table 2. Encoding methodology for categorical features.

Num	Categorical Features	Distinct Categories	Encoding Method
1	Ship type	Bulk carrier	(0.5,0,0,0,0,0)
		Container ship	(0,0.5,0,0,0,0)
		General cargo	(0,0,0.5,0,0,0)
		Passenger ship	(0,0,0,0.5,0,0)
		Chemical/oil tanker	(0,0,0,0,0.5,0)
		Other types	(0,0,0,0,0,0.5)
2	Ship flag performance	White	(0.5,0,0,0)
		Grey	(0,0.5,0,0)
		Black	(0,0,0.5,0)
		Other types	(0,0,0,0.5)
3	Ship RO performance	High	(0.5,0,0)
		Medium	(0,0.5,0)
		Low	(0,0,0.5)
4	Ship company performance	High	(0.5,0,0,0)
		Medium	(0,0.5,0,0)
		Low	(0,0,0.5,0)
		Other types	(0,0,0,0.5)

3.2.2. Normalization for Numerical Features

The value ranges of three numerical features in our dataset are shown in Table 3. The values of all numerical features are integers.

Table 3. The value ranges of numerical features.

Num	Numerical Features	Value Range
1	Ship age	{0, 1, . . ., 48}
2	Deficiencies within the previous 36 months	{0, 1, . . ., 55}
3	Detentions within the previous 36 months	{0, 1, . . ., 18}

In our study, the min–max normalization method is employed to normalize the data of numerical features in our dataset. In the experiments, the training dataset $S_N = \{(x_i, y_i) : i = 1, \ldots, N\}$ is first linearly mapped to the [0, 1] interval. Let $x_{\max} = \max_{i \in \{1,\ldots,N\}} x_i$ and $x_{\min} = \min_{i \in \{1,\ldots,N\}} x_i$; we obtain the normalized training data as follows:

$$x_i' = \frac{x_i - x_{\min}}{x_{\max} - x_{\min}}, \; i \in \{1, \ldots, N\}.$$

Then, we normalize the test dataset $T_M = \{(x_j, y_j) : j = N+1, \ldots, N+M\}$ with $x_{\max}$ and $x_{\min}$; that is:

$$x'_j = \frac{x_j - x_{\min}}{x_{\max} - x_{\min}},\ j \in \{N+1, \ldots, N+M\}.$$

3.3. Computational Procedures

The performance of our hybrid framework is assessed using a dataset comprised of 3026 PSC inspection records. Rather than splitting the dataset into large and uneven subsets where the representation of exterior data points might be skewed, we opt for smaller and more manageable batches. This not only provides a more balanced proportion of interior and exterior points but also facilitates multiple comparative validations to confirm the effectiveness of our hybrid model M3.

To achieve this, the dataset is randomly divided into 30 batches, each containing 100 records. Each batch is then randomized and further divided into three subsets: a training set D_{train}, a validation set D_{valid}, and a test set D_{test}, encompassing 60%, 20%, and 20% of the records, respectively. This translates to 60, 20, and 20 records per subset.

For each data batch, firstly, D_{train} is used to train two models: the kNN model $f_1^{k\text{NN}}$ and the linear regression model f_1^{LR}. The optimal parameter k^* for the kNN model is determined with the corresponding minimum MSE on D_{valid} from $\mathcal{K} = \{1, 2, \ldots, 10\}$. Then, we use D_{valid} to determine the optimal parameter h^* for the hybrid prediction model f_1^{hybrid}. Specifically, for each data point $\left(x_j^{\text{v}}, y_j^{\text{v}}\right)$ $(j \in \{1, \ldots, |D_{\text{valid}}|\})$ in D_{valid}, we obtain $f_1^{k\text{NN}}\left(x_j^{\text{v}}\right)$ and $f_1^{\text{LR}}\left(x_j^{\text{v}}\right)$ by using kNN and linear regression models. Then, we solve the Optimization Model M0 using Gurobi to obtain $d^*\left(x_j^{\text{v}}\right)$ and calculate

$$d_{\max} = \max_{x_j^{\text{v}} \in D_{\text{valid}}} \left\{ d^*\left(x_j^{\text{v}}\right) \right\}.$$

We denote by H the result of retaining $d_{\max}$ to one decimal place, and $\mathcal{H} = \{0.1, 0.2, \ldots, H\}$ is the tuning range for the hyperparameter h. Then, we are able to calculate $f_1^{\text{hybrid}}\left(x_j^{\text{v}}\right)$ by choosing a candidate h from $\mathcal{H}$, and select h^* that minimizes the MSE of f_1^{hybrid} on D_{valid} as the optimal hyperparameter of the hybrid framework. Finally, we concatenate D_{train} and D_{valid} to retrain the kNN model $f_2^{k\text{NN}}$, the linear regression model f_2^{LR}, and our hybrid prediction model f_2^{hybrid}. For each data point in D_{test}, we calculate the MSEs of $f_2^{k\text{NN}}$, f_2^{LR}, and f_2^{hybrid}. Algorithm 1 describes the detailed computational procedures of the experiment.

3.4. Results and Discussion

The experiments are conducted on a computer with an AMD Ryzen 5 4600U and 16 GB (3200 MHz) RAM under the Windows 10 operating system. The models are implemented in Python programming language using Gurobi 9.5.2 as the solver. The optimal hyperparameter k^* for the kNN model and h^* for our hybrid prediction framework are determined for each data batch, according to the procedures described in Section 3.3. The k^* and h^* for each batch are shown in Tables 4 and 5, respectively.

Algorithm 1: Computational procedures of the experiment

Input: The whole dataset (3026 PSC inspection records).
Output: MSEs of three prediction models M1, M2, and M3.

Step 1: Divide the whole dataset into 30 batches, each containing 100 records. Randomly divide each data batch into a training dataset D_{train}, a validation dataset D_{valid}, and a test dataset D_{test}.

Step 2: For each data batch, use D_{train} to train the kNN model $f_1^{k\text{NN}}$ and the linear regression model f_1^{LR}. The optimal parameter k^* is determined by D_{valid} from $\mathcal{K} = \{1, \ldots, 10\}$.

Step 3: Use D_{valid} to find the optimal h^* for the hybrid prediction model f_1^{hybrid}.

For $\left(x_j^{\text{v}}, y_j^{\text{v}}\right) \in D_{\text{valid}}$:

Obtain $f_1^{k\text{NN}}\left(x_j^{\text{v}}\right)$ and $f_1^{\text{LR}}\left(x_j^{\text{v}}\right)$;

Obtain $d^*\left(x_j^{\text{v}}\right)$ from Optimization Model M0;

Calculate $d_{\max} = \max\limits_{x_j^{\text{v}} \in D_{\text{valid}}} \left\{d^*\left(x_j^{\text{v}}\right)\right\}$ and obtain $\mathcal{H} = \{0.1,\ 0.2, \ldots, H\}$, where H represents the result of retaining $d_{\max}$ to one decimal place.

For $h \in \mathcal{H}$:

For $\left(x_j^{\text{v}}, y_j^{\text{v}}\right) \in D_{\text{valid}}$:

Calculate $\theta\left[d^*\left(x_j^{\text{v}}\right)\right]$ by using piecewise function (13) and obtain $f_1^{\text{hybrid}}\left(x_j^{\text{v}}\right) = \left(1 - \theta\left[d^*\left(x_j^{\text{v}}\right)\right]\right) f_1^{k\text{NN}}\left(x_j^{\text{v}}\right) + \theta\left[d^*\left(x_j^{\text{v}}\right)\right] f_1^{\text{LR}}\left(x_j^{\text{v}}\right)$.

Calculate the MSE of f_1^{hybrid} on D_{valid}.

Set h^* with the minimum MSE on D_{valid}.

Step 4: Concatenate D_{train} and D_{valid} to retrain the kNN model $f_2^{k\text{NN}}$, the linear regression model f_2^{LR}, and our hybrid prediction model f_2^{hybrid}.

Step 5: Calculate the MSEs of $f_2^{k\text{NN}}$, f_2^{LR}, and f_2^{hybrid} on D_{test}.

Table 4. Optimal hyperparameter k^* for 30 data batches.

Batch	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
k^*	2	1	6	1	2	5	6	8	10	5
Batch	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
k^*	3	1	10	8	5	9	10	8	9	5
Batch	**21**	**22**	**23**	**24**	**25**	**26**	**27**	**28**	**29**	**30**
k^*	5	5	6	7	7	7	7	3	4	5

Table 5. Optimal hyperparameter h^* for 30 data batches.

Batch	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
h^*	0.4	0.1	1	2.5	3.3	2	0.1	1.5	0.8	0.1
Batch	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
h^*	4.7	5.7	2.7	2	3	1.2	3.3	0.5	1.3	1.7
Batch	**21**	**22**	**23**	**24**	**25**	**26**	**27**	**28**	**29**	**30**
h^*	2.1	1.9	2.1	3.2	2.7	1	0.3	0.1	0.8	0.3

For an insightful comparative analysis, we examine the MSEs of three models M1 (kNN), M2 (linear regression), and M3 (hybrid prediction model) on the test dataset of every data batch, focusing on the predicted ship deficiency numbers, and the results are shown in Table 6 and Figure 4.

Table 6. MSEs of three models M1, M2, and M3 for 30 data batches.

Batch	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
M1 (*k*NN)	4.75	3.60	2.13	19.10	18.71	15.08	3.12	42.87	20.31	8.16
M2 (LR)	4.69	12.00	30.97	16.91	19.49	9.29	3.62	38.89	34.20	50.52
M3 (hybrid)	4.42	11.89	29.77	19.92	18.50	10.44	3.82	40.88	22.52	49.73
Batch	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
M1 (*k*NN)	17.38	20.90	11.34	8.53	12.84	8.20	10.61	13.17	9.84	5.95
M2 (LR)	30.68	28.03	10.71	32.78	11.65	8.12	11.84	19.58	20.88	9.29
M3 (hybrid)	17.14	22.81	10.36	14.61	12.55	7.23	10.19	19.40	20.63	5.63
Batch	**21**	**22**	**23**	**24**	**25**	**26**	**27**	**28**	**29**	**30**
M1 (*k*NN)	10.37	31.92	7.83	5.23	19.05	13.40	6.47	14.82	14.73	21.60
M2 (LR)	22.30	27.94	5.98	89.45	21.63	15.81	25.48	10.55	74.54	20.06
M3 (hybrid)	19.65	28.94	7.71	60.83	18.64	18.98	24.89	11.66	74.59	20.36

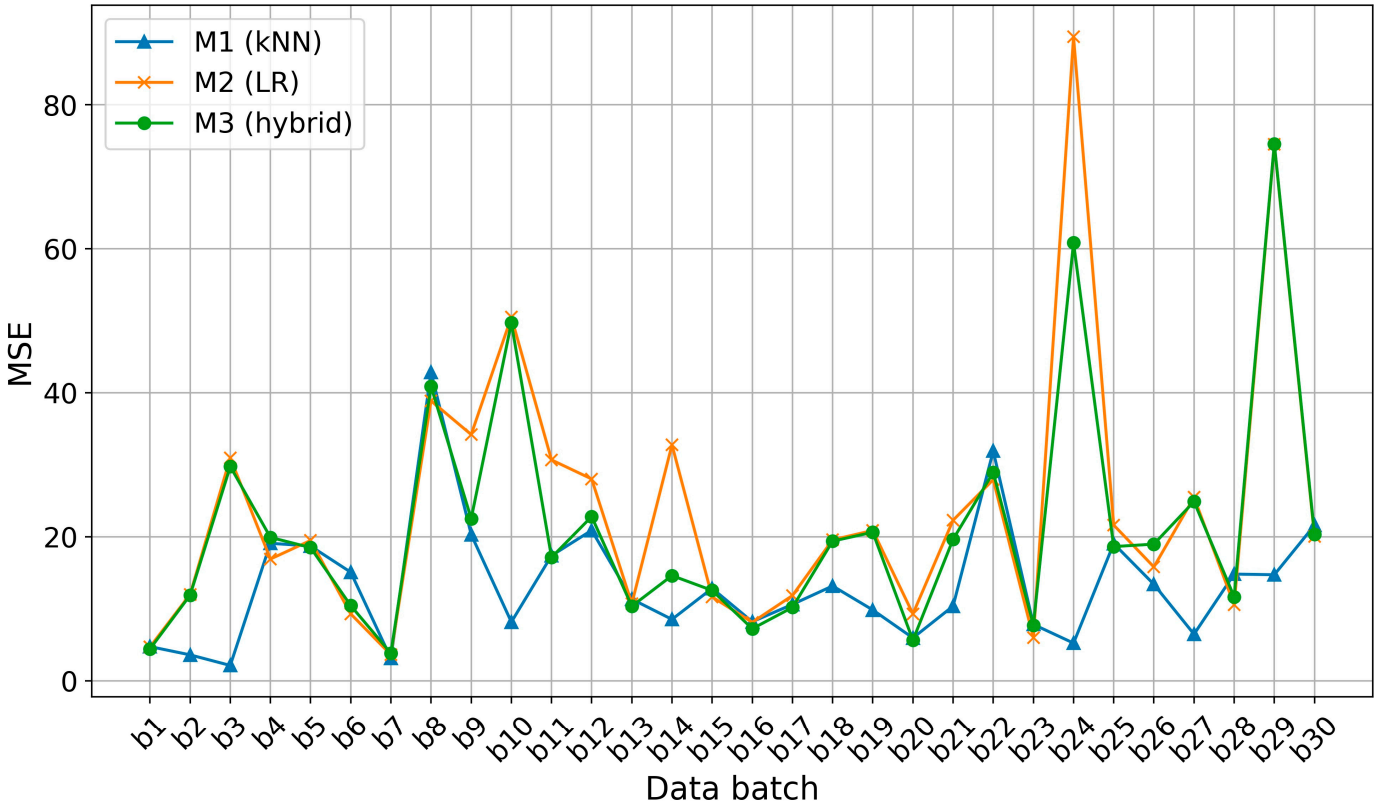

Figure 4. MSEs of three models M1, M2, and M3 for 30 data batches.

These results are then categorized into three distinct groups: data batches with the minimum, intermediate, or maximum MSEs in model M3 (hybrid prediction model), which are illustrated in Figures 5–7, respectively.

The results of computational experiments demonstrate that M3 (the hybrid prediction model) consistently surpasses M1 (*k*NN) or M2 (linear regression) in a majority of the data batches, yielding lower MSEs, which validates the effectiveness of our proposed hybrid predictive framework. Specifically, M3 excels over M1 or M2 in 86.7% of the 30 data batches.

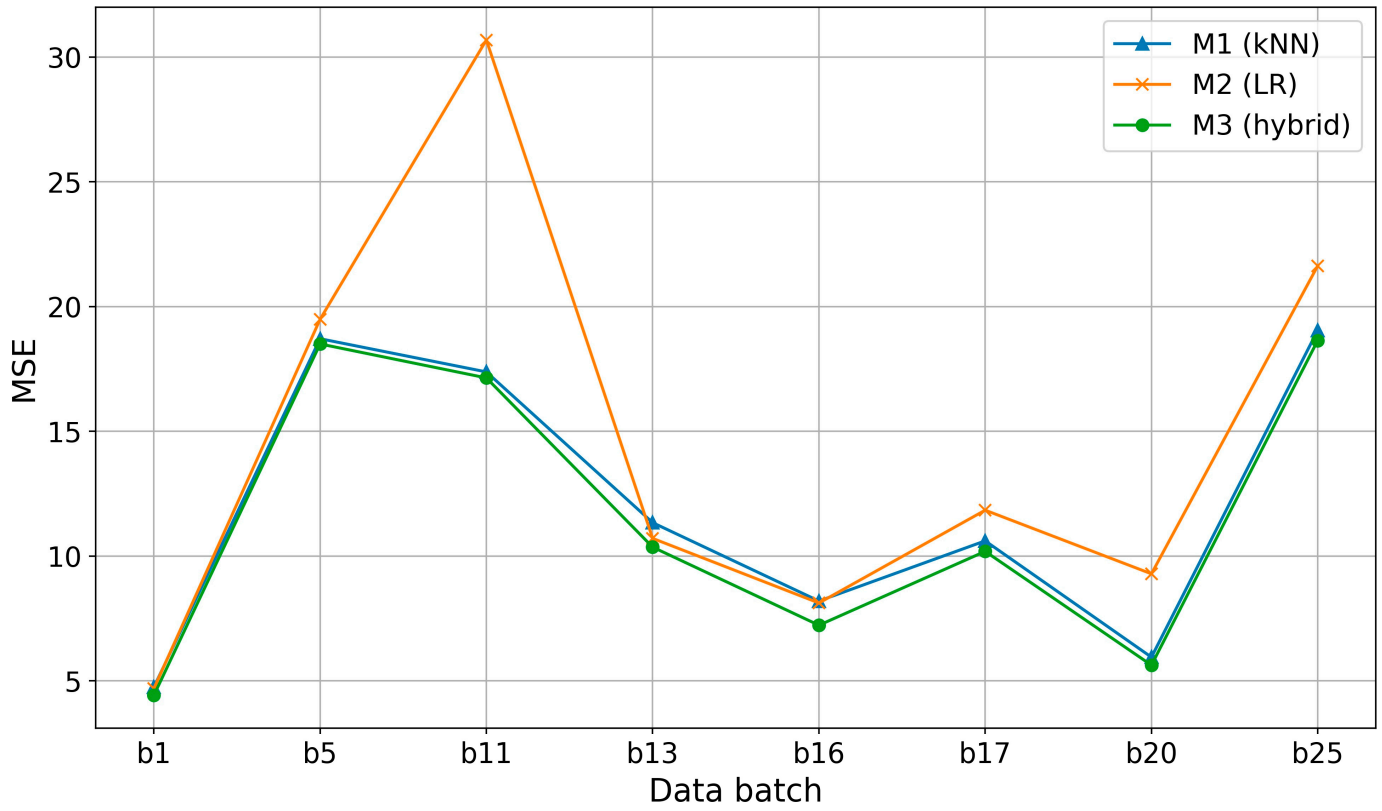

Figure 5. Data batches with the minimum MSE in M3.

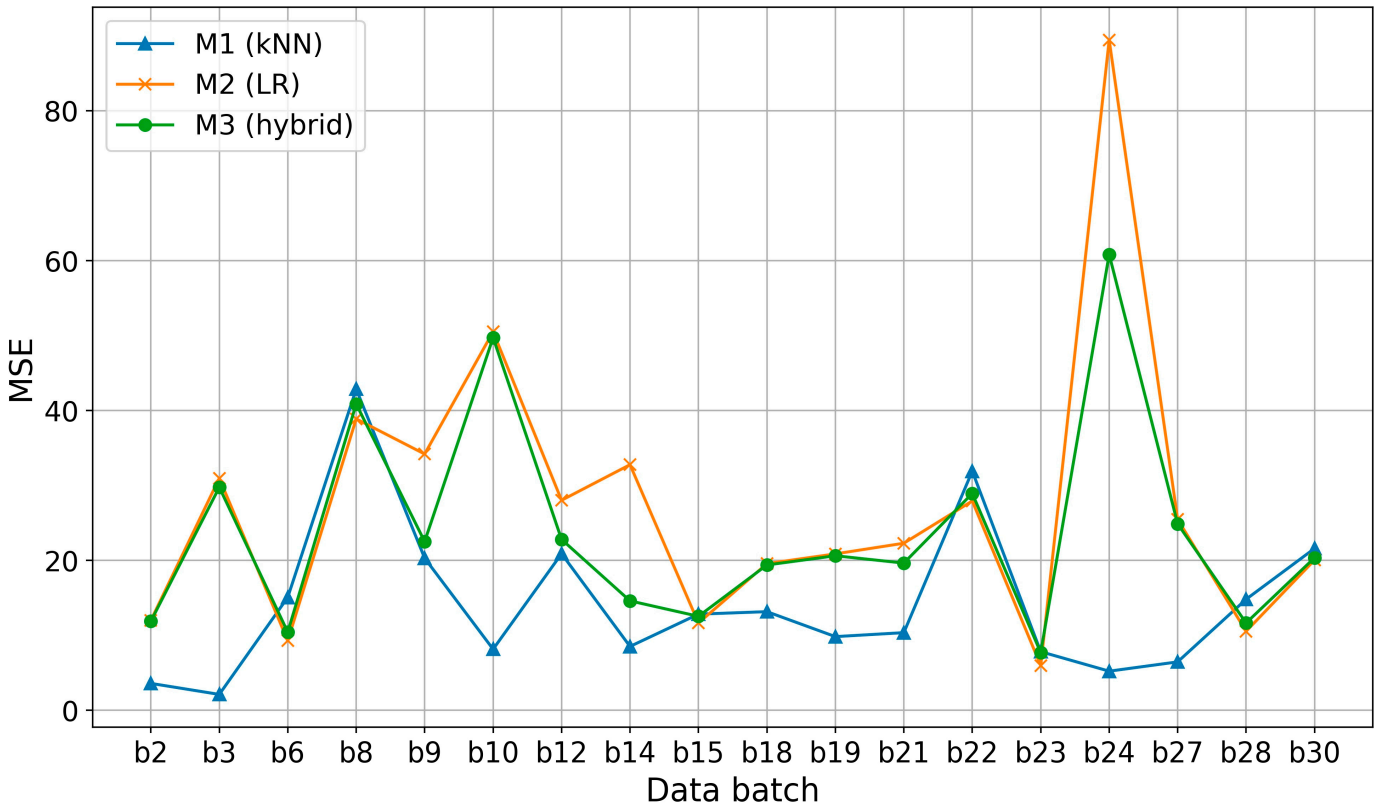

Figure 6. Data batches with the intermediate MSE in M3.

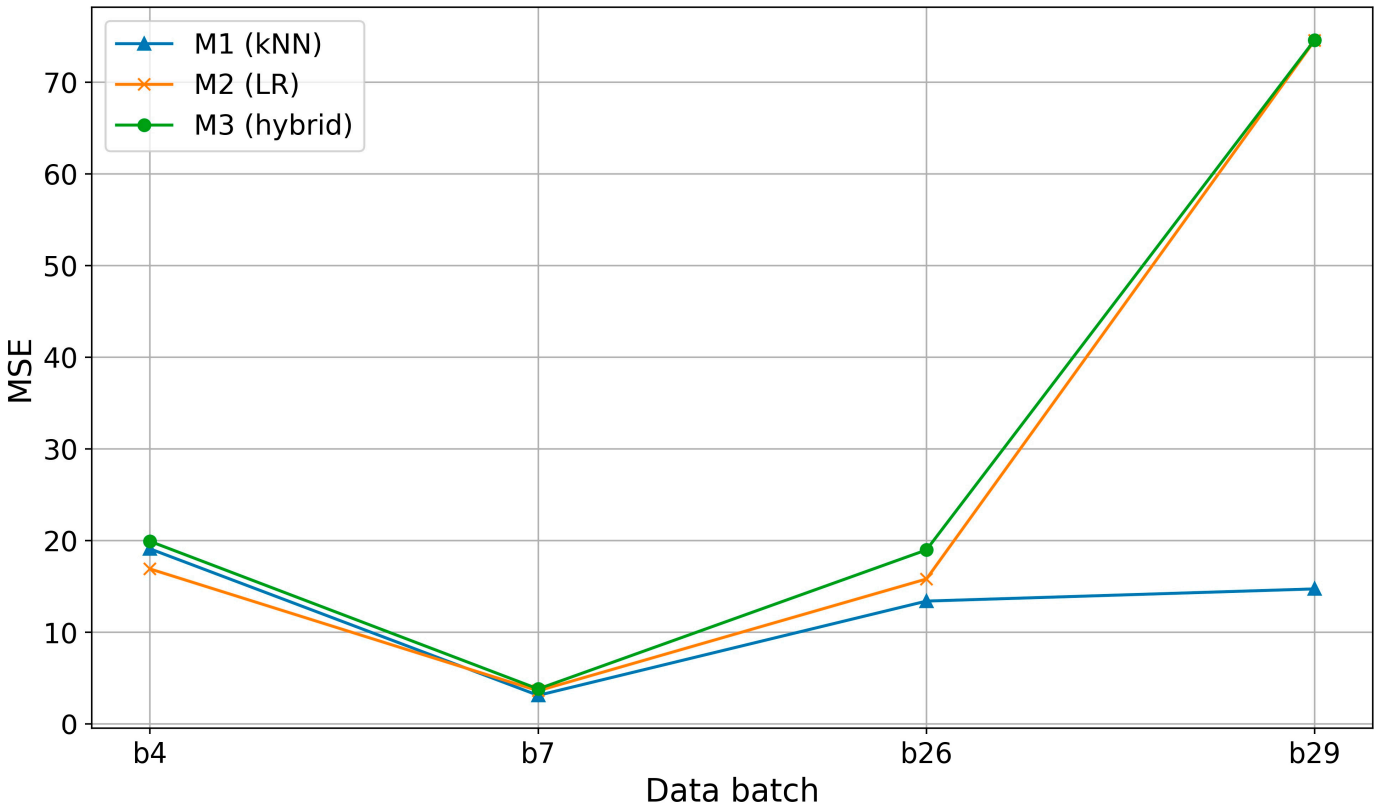

Figure 7. Data batches with the maximum MSE in M3.

As illustrated in Figure 5, M3 showcases superior accuracy in eight data batches, as indicated by its lowest MSEs. Within these batches, M1 achieves the highest MSEs in three batches (1, 13, 16), while M2 achieves the highest MSEs in the other five batches (5, 11, 17, 20, 25). However, a different scenario emerges in Figure 6, where M3 achieves the intermediate MSEs among three methods in 18 batches. Here, M3 exceeds one of the other two models but does not perform best in terms of predictive accuracy. In these 18 batches, M1 achieves the lowest MSEs in 11 batches (2, 3, 9, 10, 12, 14, 18, 19, 21, 24, 27), while M2 achieves the lowest MSEs in 7 batches (6, 8, 15, 22, 23, 28, 30). To add another layer of complexity, Figure 7 shows scenarios where M3's performance is the least impressive. Specifically, in four batches, M3 records the highest MSEs, with M1 demonstrating the lowest MSEs in three batches (7, 26, 29) and M2 realizing the lowest MSEs in data batch 4 only.

Diving deeper into these outcomes, the performance variability of M3 can be traced back to the spatial distribution of data points within the feature space. M3 excels in scenarios where data points are spread out, effectively harnessing the strengths of both M1 and M2. Examples of this phenomenon are data batches 11, 17, 20, and 25, where M3 substantially outperforms its counterparts, as shown in Figure 5. In contrast, when data points congregate or cluster closely in the feature space, M3 does not outperform *k*NN. Data batch 26 serves as an example, where M3's MSE is the highest among the three models.

4. Conclusions

Interpolation and extrapolation serve as pivotal techniques to predict the target values of new feature observations based on a known dataset. The precision of these predictive methods, however, is heavily influenced by the context in which they are used. Addressing these issues, we introduce a hybrid prediction framework, combining the virtues of both interpolation and extrapolation.

Central to our framework are two components: a sophisticated optimization model and a novel hybrid prediction approach. The optimization model employs a convex hull

to ascertain the spatial positioning (i.e., the centrality) of a new data point in relation to a known dataset. Using the centrality coefficient, the hybrid prediction method merges *k*NN and linear regression. This dual-action technique facilitates predictions grounded on a data point's centrality to a given dataset. In essence, our framework's versatility lies in its ability to adeptly handle both interior and exterior data points, ensuring the optimal prediction methodology is utilized.

In our computational experiments, we use a PSC inspection dataset in the ship deficiency prediction domain. Benchmarked against *k*NN and linear regression methods, our framework showcases specific advantages using the metric of MSE. Significantly, it displays superior accuracy in forecasting deficiencies across 86.7% of data batches, elevating the prediction's accuracy.

In future research, this framework can be applied in resolving other relevant problems. Moreover, assimilating an array of interpolation and extrapolation techniques or more sophisticated methods like random forest, XGBoost, neural network, and deep generative models could further refine and enhance our framework's capabilities.

Author Contributions: Conceptualization, X.T. and S.W.; methodology, B.J., X.Z., X.T., W.Y. and S.W.; software, B.J. and X.Z.; validation, B.J.; formal analysis, B.J. and X.Z.; investigation, B.J. and X.Z.; resources, S.W.; data curation, B.J. and X.Z.; writing—original draft preparation, B.J. and X.Z.; writing—review and editing, X.T., W.Y. and S.W.; visualization, B.J.; supervision, S.W.; project administration, W.Y.; funding acquisition, W.Y. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Dataset available on request from the authors.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Balestriero, R.; Pesenti, J.; LeCun, Y. Learning in high dimension always amounts to extrapolation. *arXiv* **2021**, arXiv:2110.09485.
2. Sarfraz, M.; Ishaq, M.; Hussain, M.Z. Shape designing of engineering images using rational spline interpolation. *Adv. Mater. Sci. Eng.* **2015**, *2015*, 260587. [CrossRef]
3. Colwell, R.K.; Chao, A.; Gotelli, N.J.; Lin, S.Y.; Mao, C.X.; Chazdon, R.L.; Longino, J.T. Models and estimators linking individual-based and sample-based rarefaction, extrapolation and comparison of assemblages. *J. Plant Ecol.* **2012**, *5*, 3–21. [CrossRef]
4. Hennessy, C.A.; Strebulaev, I.A. Beyond random assignment: Credible inference and extrapolation in dynamic economies. *J. Financ.* **2020**, *75*, 825–866. [CrossRef]
5. Talvitie, J.; Renfors, M.; Lohan, E.S. Distance-based interpolation and extrapolation methods for RSS-based localization with indoor wireless signals. *IEEE Trans. Veh. Technol.* **2015**, *64*, 1340–1353. [CrossRef]
6. Steffensen, J.F. *Interpolation*, 2nd ed.; Chelsea Publishing Company: New York, NY, USA, 1951.
7. Minda, A.A.; Barbinita, C.I.; Gillich, G.R. A review of interpolation methods used for frequency estimation. *Rom. J. Acoust. Vib.* **2020**, *17*, 21–26.
8. Press, W.H. *Numerical Recipes: The Art of Scientific Computing*, 3rd ed.; Cambridge University Press: Cambridge, UK, 2007.
9. Blu, T.; Thévenaz, P.; Unser, M. Linear interpolation revitalized. *IEEE Trans. Image Process.* **2004**, *13*, 710–719. [CrossRef] [PubMed]
10. Tal-Ezer, H. High degree polynomial interpolation in Newton form. *SIAM J. Sci. Stat. Comput.* **1991**, *12*, 648–667. [CrossRef]
11. Challu, C.; Olivares, K.G.; Oreshkin, B.N.; Garza Ramirez, F.; Mergenthaler Canseco, M.; Dubrawski, A. Nhits: Neural hierarchical interpolation for time series forecasting. In Proceedings of the 37th AAAI Conference on Artificial Intelligence, Washington, DC, USA, 7–14 February 2023.
12. Sekulić, A.; Kilibarda, M.; Heuvelink, G.B.M.; Nikolić, M.; Bajat, B. Random forest spatial interpolation. *Remote Sens.* **2020**, *12*, 1687. [CrossRef]
13. Scott Armstrong, J.; Collopy, F. Causal forces: Structuring knowledge for time--series extrapolation. *J. Forecast.* **1993**, *12*, 103–115. [CrossRef]
14. Webb, T.; Dulberg, Z.; Frankland, S.; Petrov, A.; O'Reilly, R.; Cohen, J. Learning representations that support extrapolation. In Proceedings of the 37th International Conference on Machine Learning, PMLR, Vienna, Austria, 13–18 July 2020.
15. Zhu, M.; Zhang, H.; Jiao, A.; Karniadakis, G.E.; Lu, L. Reliable extrapolation of deep neural operators informed by physics or sparse observations. *Comput. Methods Appl. Mech. Eng.* **2023**, *412*, 116064. [CrossRef]

16. Brezinski, C.; Zaglia, M.R. *Extrapolation Methods: Theory and Practice*; Elsevier: Amsterdam, The Netherlands, 2013.
17. Liu, Y.; Xue, Y.; Taniguchi, M. Robust linear interpolation and extrapolation of stationary time series in Lp. *J. Time Ser. Anal.* **2020**, *41*, 229–248. [CrossRef]
18. McCartney, M.; Haeringer, M.; Polifke, W. Comparison of machine learning algorithms in the interpolation and extrapolation of flame describing functions. *J. Eng. Gas Turbines Power* **2020**, *142*, 061009. [CrossRef]
19. Rosenfeld, E.; Ravikumar, P.; Risteski, A. An online learning approach to interpolation and extrapolation in domain generalization. In Proceedings of the 25th International Conference on Artificial Intelligence and Statistics, PMLR, Valencia, Spain, 28–30 March 2022.
20. Tokyo MoU. Information Sheet of the New Inspection Regime (NIR). Available online: http://www.tokyo-mou.org/doc/NIR-information%20sheet-r.pdf (accessed on 31 January 2024).
21. Tian, S.; Zhu, X. Data analytics in transport: Does Simpson's paradox exist in rule of ship selection for port state control. *Electron. Res. Arch.* **2023**, *31*, 251–272. [CrossRef]
22. Wang, S.; Yan, R.; Qu, X. Development of a non-parametric classifier: Effective identification, algorithm, and applications in port state control for maritime transportation. *Transp. Res. Part B Methodol.* **2019**, *128*, 129–157. [CrossRef]
23. Dinis, D.; Teixeira, A.P.; Soares, C.G. Probabilistic approach for characterising the static risk of ships using Bayesian networks. *Reliab. Eng. Syst. Saf.* **2020**, *203*, 107073. [CrossRef]
24. Yan, R.; Wang, S. Ship detention prediction using anomaly detection in port state control: Model and explanation. *Electron. Res. Arch.* **2022**, *30*, 3679–3691. [CrossRef]
25. Rao, A.R.; Wang, H.; Gupta, C. Predictive analysis for optimizing port operations. *arXiv* **2024**, arXiv:2401.14498.