## 3.1. Machine learning methods

### 3.1.0. Introduction

Machine learning models have been increasingly used in recent years to model physical processes. These models can be used to predict the behavior of complex systems, such as mechanical systems and heat processing systems, with high accuracy. Groensfelder et al [1] investigated the prediction accuracy of multi-parametric models derived from numerical data for mechanical systems. They compared polynomial matrix equations to regression models and neural network models provided by machine learning toolboxes. Kumar [2] provided a structured overview of integrating physics-based modeling with state-of-the-art machine learning techniques. In the field of heat processing, ML-based approaches have been shown to effectively model the nonlinear complex heat and mass transfer processes during drying and are able to predict drying kinetics without much computational effort, such as traditional modeling techniques [3].

Machine learning models can be used to model physical processes by learning from data and identifying patterns in the data. These models can then be used to make predictions about the behavior of the system. The use of machine learning models can help to reduce the computational cost of modeling complex systems and can provide more accurate predictions than traditional modeling techniques.

Machine learning models can also be used to simulate costly experiments. Neural networks can be trained to predict the results of expensive experimental studies or time-consuming simulations reliably and thus replace them [1]. This can save time and money and accelerate technical processes. Herty et al [1] introduced a new neural network that is based on many layers and can be classified into the category of deep learning. Furthermore, it is shown how a network already trained for a data set can be efficiently and time-savingly adapted to a new data set with changed input and/or changed target variables by means of a filter method. The filter method replaces the often very time-consuming training of a neural network by forward and backpropagation. The applicability and efficiency of the presented methods are demonstrated by means of two technical applications, and the performance is analyzed regarding the prediction quality and the computing time [1].

*1: Groensfelder, T., Giebeler, F., Geupel, M. et al. Application of machine learning procedures for mechanical system modelling: capabilities and caveats to prediction-accuracy. Adv. Model. and Simul. in Eng. Sci. 7, 26 (2020).*

*2: Kumar, V. Integrating Physics-Based Modeling* *With Machine Learning: A Survey. IEEE Trans. Neural Netw. Learn. Syst. 31, 1–16 (2020).*

*3: Li, J., Zhang, Y., Li, Y. et al. Fundamental Understanding of Heat and Mass Transfer Processes for Drying of Porous Materials: A Review. Energies 15, 9347 (2022).*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | **Mean** | |  |
|  |  | **Microfin tube h** | **Plain tube h** | **Deviation** |
| **Input Raw** | Mass flux | 282.678974 | 287.516213 | 0.016824 |
| Saturation pressure | 546880 | 533260.976 | 0.025539 |
| Heat flux | 12442.84554 | 12606.37154 | 0.012972 |
| Quality | 0.505296 | 0.494793 | 0.021227 |
| Pressure drop | 11966.44706 | 7230.353147 | 0.655029 |
| **Input Calculated** | Reynolds number | 5715.992018 | 5786.624969 | 0.012206 |
| Two-phase multiplier | 0.22905 | 0.233213 | 0.017847 |
| Froude number | 0.66961 | 0.687362 | 0.025827 |
| Weber number | 67.137622 | 68.23943 | 0.016146 |
| Bond number | 0.165302 | 0.167358 | 0.012288 |
| **Output** | Heat transfer coefficient | 5855.256499 | 4170.965336 | 0.403813 |
|  |  |  |  |  |

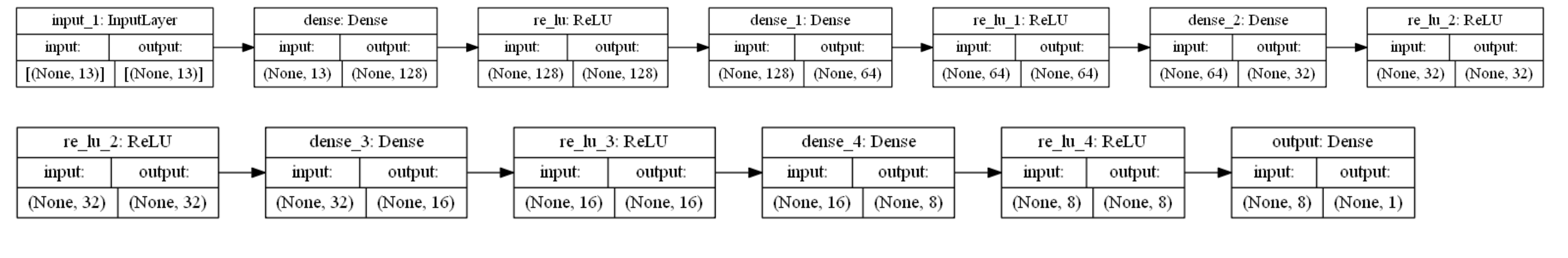
Table 1- Input and output statistics by tube type

### 3.1.1 Artificial Neural Network (ANN)

Artificial Neural Networks (ANN) are a class of machine learning algorithms that are loosely inspired by the brain. They are typically organized in layers, with the first layer being the input layer, the last layer being the output layer, and any layers in between being hidden layers. Each layer is made up of one or more neurons. Each neuron takes a weighted sum of its inputs, applies an activation function, and passes the result to the next layer. The weights and biases of each neuron are adjusted during training. The network is trained by passing training data through the network, calculating the error at the output layer, and then propagating the error back through the network to adjust the weights and biases. This process is repeated until the error reaches a minimum. Once trained, the network can be used to make predictions on new data.

ANNs are useful for modeling complex, non-linear relationships between inputs and outputs. They are also useful for modeling relationships where the inputs are not fully understood or cannot be easily modeled. ANNs can be used for regression or classification problems. They are often used for image recognition, speech recognition, and natural language processing. They are also used for time-series forecasting and financial modeling.

The basic structure of an ANN is shown below for 3 input variables, one hidden layer of 4 cells and 2 output variables (4-2):



A diagram of a network

Description automatically generated

The input layer, hidden layers and the output layer are shown in red, blue and green, respectively. The connections between the neurons are shown in black. The weights and biases are shown in gray.

The output of a neuron is given by:

where is the ’th input, is the ’th weight, is the bias, and is the activation function. The activation function is typically a non-linear function such as the sigmoid function or the hyperbolic tangent function. The weights and biases are optimized during training to minimize the error at the output layer.

In this study, we used a feed-forward neural network with 5 hidden layers. The number of neurons in each layer was 128, 64, 32, 16, and 8 (128-64-32-16-8-1). The activation function was the rectified linear unit (ReLU). The ReLU function is given by:

Network architecture and activation function was selected using a 3-fold cross validation scheme; the hyperparameter set yielding the highest score is chosen to retrain the model on the whole training dataset and evaluation is performed on a separate holdout set of data. The details of sampling for training and test sets are given in ***section 3.2****.*

Limited-memory Broyden-Fletcher-Goldfarb-Shanno (L-BFGS) algorithm is used for network weight optimization. L-BFGS is a quasi-Newton method that approximates the Hessian matrix using the previous gradient vectors, which is a fast and suitable solver that works well for relatively small datasets where batch processing is not necessary. By this means, we aim for avoiding optimization errors due to stochastic optimization approaches.

### 3.1.2 Locally Weighted Linear Regression (LWR)

LWR is a non-parametric method that fits a linear model to a subset of data points that are close to a given query point. The idea is to give more weight to the data points that are near the query point, and less weight to the data points that are far away. This way, the model can capture the local structure of the data and adapt to non-linear patterns.

In this study, we restrict locality to the nearest k data points of query sample and define error function as follows:

where is the query sample, is the true function, is the approximation function, is the distance between the query sample and the data point and is the kernel function. The kernel function is used to give more weight to the data points that are close to the query point. Although several choices are available for the kernel function with some esoteric options, we use the Gaussian kernel function for this study. The Gaussian kernel function is given by:

where is a hyperparameter that controls the width of the kernel. The hyperparameter is tuned using cross-validation.

Distance between two points is calculated using the Euclidean distance:

where is the component of the query sample, and is the component of the data point.

We choose a linear expression for approximation function:

where is the weight vector. Since the problem is constrained to a neighborhood of query sample, we use normal equation to optimize approximation parameter :

where is the design matrix for neighborhood of query sample , is the diagonal matrix of weights, and is the output vector for respective design matrix.

### 3.1.3 Other Model Considerations

We did not consider tree-based regression models in this work, as we were looking for a model that could extrapolate and extract analytical relationships between input and output variables.

Tree based models, such as decision trees, random forests, and gradient boosting, are known to struggle with extrapolation, as they tend to predict the average value of the leaf nodes, which are bounded by the range of the training data. Furthermore, tree-based models are not easily interpretable, as they do not provide explicit formulas or coefficients that relate the input and output variables, thus causing additional correlation studies to be difficult. Therefore, tree-based models are not suitable for the purpose of this work, which aims to find a model that can generalize well beyond the observed data and reveal the underlying physical mechanisms.

### 3.2 Validation Setting

Due to scarcity of the data, we adopted following approach for sampling train, validation and test sets in a balanced manner: (1) Principal component analysis (PCA) is applied to reduce the dimensionality of experiment data and visualize the potential clusters, (2) a Gaussian mixture model (GMM) is used to estimate the number and parameters of the clusters, (3) cluster labels are used to perform a 5-fold group cross validation, where we split the data into five groups while preserving the cluster proportions in each group. (4) We trained and assessed our model on each group and reported the average performance.

A screen shot of a graph

Description automatically generated

Figure 1- Experimentation samples in principal component space

Within each of 5 folds, data is partitioned into 3 sub-folds to perform grid search for model hyperparameters, therefore total training process is based on 5-3-fold cross validation. The same stratification technique is applied when forming sub-folds for hyperparameter optimization.

The number of components in GMM application is taken as 3, by visually examining the distribution of the experiment data in principal component space. Each distribution is assumed to have a full covariance matrix to respect input correlations.

Details of PCA and additional observations are provided in ***Section 4.1***.

### 3.3 Evaluation metrics

Different metrics are used to evaluate models’ predictive performances to capture different abilities of models. Used evaluation metrics are score, mean absolute error (MAE), root mean squared error (RMSE), mean absolute percentage error (MAPE), and weighted absolute percentage error (WAPE). Calculation methods of each performance metric are given below. In all metric calculations, is the actual heat transfer coefficient (or pressure drop) for the test sample, is the predicted heat transfer coefficient (or pressure drop) for the test sample and is the mean heat transfer coefficient (or pressure drop) of all samples in the test set.

is a statistical measure of how close the data are to the fitted regression line. It is also known as the coefficient of determination, or the coefficient of multiple determination for multiple regression. In this study, we use the following definition of :

Mean absolute error (MAE) is a measure of difference between two continuous variables. For two vectors and , MAE is defined as:

Root mean squared error (RMSE) is a quadratic scoring rule that also measures the average magnitude of the error. It is defined as:

can be driven up by outliers and does not indicate under- or over-estimation. To address this issue, we use the mean absolute percentage error (MAPE) and weighted absolute percentage error (WAPE) scores whose details are given below that penalize errors relative to the true value.

Mean absolute percentage error (MAPE) is a measure of prediction accuracy of a forecasting method in statistics. MAPE is defined as:

Weighted absolute percentage error (WAPE) is a measure of prediction accuracy of a forecasting method in statistics. WAPE is defined as:

Weighted percentage bias is a measure of systematic error in regression models. Weighted percentage bias is defined as:

### 3.4 Sampling for extrapolation performance

In this section, we demonstrate the selected models' strength for making accurate estimations outside observed ranges during the experiment.

Extrapolation performance analysis is conducted by following approach: (1) clusters are identified in principal component space through fitting a Gaussian Mixture Model (GMM), (2) and a Ledoit-Wolf (LW) covariance estimator is fitted separately to each identified cluster again in principal component space. (3) Finally, Mahalanobis distances are calculated based on LW estimates and samples with highest distance from distribution centers are marked as extrapolation samples.

The LW estimator is a shrinkage estimator that shrinks the sample covariance matrix towards a structured estimator. The structured estimator is a matrix with a constant diagonal and constant off-diagonal elements. The LW estimator is given by:

where is the LW estimator, is the sample covariance matrix, is the structured estimator, and is the shrinkage parameter. The shrinkage parameter is given by:

where is the diagonal element of the sample covariance matrix, is the diagonal element of the structured estimator, and is the average of the diagonal elements of the sample covariance matrix.

Mahalanobis distance is a measure of the distance between a point and a distribution. It is a multi-dimensional generalization of the one-dimensional Euclidean distance. It is defined as the square root of the sum of the squared differences between the point and the mean of the distribution, divided by the covariance matrix of the distribution. It is given by:

where is the point, is the mean of the distribution, and is the covariance matrix of the distribution.

Samples to be used for extrapolation are then determined by Mahalanobis distance estimations coming from LW estimators fitted on each cluster. 10 samples with the highest Mahalanobis distance from each cluster are then held out for test and the rest of the samples are used for training the models.A graph of red and blue dots

Description automatically generated

Figure 2 - Training and extrapolation samples in principal component space

## 4. Results and Discussion

A collage of blue dots

Description automatically generated

Figure 3- Input and output interaction plots of the system

### 4.1. Validation

A screen shot of a graph

Description automatically generated

Figure 4- Finned vs plain tube samples in principal component space

|  |  |  |  |
| --- | --- | --- | --- |
|  | **PC1** | **PC2** | **PC3** |
| **Mass flux** | 0.3564 | 0.3828 | -0.1576 |
| **Saturation pressure** | 0.0366 | 0.0390 | 0.8466 |
| **Heat flux** | -0.0334 | 0.0465 | 0.0043 |
| **Quality** | -0.3549 | 0.3716 | -0.0112 |
| **Pressure** **drop** | 0.0005 | 0.3995 | 0.4582 |
| **Reynolds number** | 0.4600 | -0.0307 | -0.0279 |
| **Two-phase multiplier** | 0.3643 | -0.3642 | 0.1199 |
| **Froude number** | 0.3647 | 0.3765 | -0.1162 |
| **Weber number** | 0.3658 | 0.3783 | -0.0437 |
| **Bond number** | 0.3653 | -0.3656 | 0.1333 |

Table 2- Projection axes for principal components across input variables

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Mass flux** | **Saturation pressure** | **Heat flux** | **Quality** | **Pressure** **drop** | **Reynolds number** | **Two-phase multiplier** | **Froude number** | **Weber number** | **Bond number** | **Heat transfer coefficient** |
| **Category** |  |  |  |  |  |  |  |  |  |  |  |
| **0** | 190.394 | 553191.000 | 12336.142 | 0.482 | 5589.500 | 3998.390 | 0.253 | 0.284 | 28.588 | 0.181 | 5487.154 |
| **1** | 380.788 | 539760.556 | 12318.045 | 0.425 | 16670.167 | 8900.799 | 0.310 | 1.129 | 112.751 | 0.216 | 6434.455 |
| **2** | 286.271 | 539451.905 | 12924.940 | 0.518 | 12422.381 | 5496.968 | 0.208 | 0.638 | 63.710 | 0.152 | 5679.046 |
| **3** | 191.584 | 540143.645 | 12098.984 | 0.499 | 3007.376 | 3821.949 | 0.227 | 0.286 | 28.543 | 0.164 | 3717.482 |
| **4** | 380.550 | 536835.695 | 12763.618 | 0.441 | 10515.500 | 8496.691 | 0.283 | 1.127 | 112.227 | 0.200 | 4578.828 |
| **5** | 289.702 | 532606.721 | 12729.166 | 0.521 | 7815.455 | 5295.849 | 0.201 | 0.653 | 64.865 | 0.146 | 4238.314 |

Table 3- Input variable statistics by clusters identified in principal component space

### 4.2. Heat transfer coefficient

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Selected ANN config** | **R2** | **RMSE** | **MAE** | **MAPE** | **Bias%** | **WAPE%** | **Pearson-R** |
| **Fold ID** |  |  |  |  |  |  |  |  |
| **0** | ((128, 64, 32, 16, 8), 'relu') | 72.6% | -842.61 | -443.99 | -7.4% | 2.1% | -8.0% | 80.5% |
| **1** | ((128, 64, 32, 16, 8), 'relu') | 56.9% | -759.76 | -430.31 | -8.2% | -2.7% | -9.8% | 75.9% |
| **2** | ((128, 64, 32, 16, 8), 'relu') | 44.7% | -700.55 | -368.63 | -7.1% | -1.1% | -8.9% | 75.3% |
| **3** | ((128, 64, 32, 16, 8), 'relu') | 56.8% | -793.18 | -508.60 | -8.8% | 1.5% | -9.4% | 62.0% |
| **4** | ((128, 64, 32, 16, 8), 'relu') | 55.4% | -725.95 | -505.13 | -9.0% | 4.5% | -9.4% | 59.8% |

Table 4- Cross validation results for ANN configurations -

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Score** | **Average** | **Std** | **Min** | **25%** | **50%** | **75%** | **Max** | **Test Score** |
| **Test R2** | 57.3% | 9.9% | 44.7% | 55.4% | 56.8% | 56.9% | 72.6% | **76.3%** |
| **Test RMSE** | -764.41 | 55.95 | -842.61 | -793.18 | -759.76 | -725.95 | -700.55 | **571.15** |
| **Test MAE** | -451.33 | 58.11 | -508.60 | -505.13 | -443.99 | -430.31 | -368.63 | **385.29** |
| **Test MAPE** | -8.1% | 0.8% | -9.0% | -8.8% | -8.2% | -7.4% | -7.1% | **7.3%** |
| **Test Bias%** | 0.8% | 2.8% | -2.7% | -1.1% | 1.5% | 2.1% | 4.5% | **0.6%** |
| **Test WAPE%** | -9.1% | 0.7% | -9.8% | -9.4% | -9.4% | -8.9% | -8.0% | **7.7%** |
| **Test Pearson-R** | 70.7% | 9.2% | 59.8% | 62.0% | 75.3% | 75.9% | 80.5% | **76.4%** |

Table 5- Cross validation and test score statistics -

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Fold ID** | **Selected LWR config** | **Test R2** | **Test RMSE** | **Test MAE** | **Test MAPE** | **Test Bias%** | **Test WAPE%** | **Test Pearson-R** |
| **0** | (1, 20) | 73.5% | -828.87 | -453.28 | -7.8% | 3.2% | -8.2% | 83.8% |
| **1** | (1, 20) | 68.6% | -675.94 | -464.50 | -9.4% | 0.5% | -8.8% | 73.1% |
| **2** | (0.7, 20) | 56.3% | -765.69 | -476.13 | -9.6% | -5.1% | -10.8% | 70.5% |
| **3** | (0.7, 20) | 34.9% | -760.23 | -406.56 | -8.5% | -2.7% | -10.0% | 77.6% |
| **4** | (0.7, 20) | 23.4% | -951.70 | -583.23 | -10.3% | 3.1% | -11.3% | 38.5% |
| **Test Set** | **(0.7, 20)** | **64.5%** | **698.13** | **461.61** | **8.9%** | **4.6%** | **10.0%** | **76.4%** |

Table 6- Cross validation and test scores for LWR configurations -

A graph with red and blue dots

Description automatically generated

Figure 5 - Actual vs predicted heat transfer coefficients for plain tube

A graph with a line and dots

Description automatically generated with medium confidence

Figure 6- Actual vs predicted heat transfer coefficients for micro finned tube

### 4.3. Pressure drop

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Fold ID** | **Model type** | **Configuration** | **Test R2** | **Test RMSE** | **Test MAE** | **Test MAPE** | **Test Bias%** | **Test WAPE%** | **Test Pearson-R** |
| **0** | ANN | ((128, 64, 32, 16, 8), 'relu') | 94.6% | -1208.29 | -873.77 | -9.4% | 7.8% | -9.3% | 97.3% |
| **1** | ANN | ((128, 64, 32, 16, 8), 'relu') | 66.8% | -3196.07 | -1723.78 | -14.9% | -14.7% | -21.5% | 91.3% |
| **2** | ANN | ((128, 64, 32, 16, 8), 'relu') | 86.1% | -2639.49 | -1348.12 | -12.5% | 6.8% | -12.1% | 89.2% |
| **3** | ANN | ((128, 64, 32, 16, 8), 'relu') | 88.6% | -1864.51 | -1250.50 | -13.1% | -0.3% | -13.1% | 88.6% |
| **4** | ANN | ((128, 64, 32, 16, 8), 'relu') | 87.1% | -1834.56 | -1361.69 | -21.9% | -3.7% | -18.5% | 87.5% |
| **0** | LWR | (2, 20) | 92.9% | -1381.97 | -1012.09 | -12.9% | 0.5% | -11.5% | 92.9% |
| **1** | LWR | (2, 20) | 91.1% | -1521.85 | -1184.95 | -29.6% | -3.7% | -21.3% | 91.8% |
| **2** | LWR | (0.7, 20) | 88.8% | -1847.06 | -1197.47 | -12.3% | 3.0% | -11.8% | 89.5% |
| **3** | LWR | (1, 20) | 85.0% | -2740.48 | -1313.47 | -13.2% | 6.1% | -12.4% | 87.1% |
| **4** | LWR | (0.7, 20) | 63.8% | -3336.76 | -1937.61 | -17.8% | -5.5% | -21.2% | 72.7% |

Table 7 - Cross validation results for pressure drop models

A graph of a graph with a number of points

Description automatically generated with medium confidence

Figure 7 - Actuals vs predictions for pressure drop in plain tube

A graph of a graph with blue and red dots

Description automatically generated

Figure 8 - Actuals vs predictions for pressure drop in micro finned tube

### 4.4. Extrapolation performance

A screen shot of a graph

Description automatically generated

Figure 9 - Actual vs prediction plots for extrapolations on plain tube heat transfer coefficients

A graph with red dots

Description automatically generated

Figure 10 - Actual vs prediction plots for extrapolations on micro finned tube heat transfer coefficients

|  |  |
| --- | --- |
| **Score** | **Value** |
| R2 | 61.65% |
| MAPE | 9.06% |
| RMSE | 915.491 |

Table 8- Performance metrics for extrapolation of on Plain tube

## 5. N/a

In the existing work, detailed comparative research has been undertaken to evaluate the predictive capabilities of four prominent ML techniques. These techniques include linear regression, which establishes a linear relationship between variables for prediction; SVM, a powerful classifier that can regress or maximize the margin between classes or fitting points; DTR, which employs recursive splitting to model non-linear relationships; and ensemble-based RF, which associations multiple decision trees to enhance forecast accurateness and mitigate overfitting. The forthcoming sections delineate the distinctive mechanisms and performance traits of each method, offering a comprehensive understanding of their applicability and effectiveness across various scenarios.

LR is a statistical method employed for modeling the relation between a dependent variable and one or more independent ones by fitting a linear equation to the spotted data points. The formula for simple linear regression can be represented by Eq. 6.

(6)

where is the dependent variable, is the independent one, is the y-intercept, is the slope coefficient, and represents the error term accounting for the variability not explained by the model. The objective of LR is to estimate the values of and that minimize the sum of squared differences between the real and guessed values of the dependent variable, thus establishing a linear relationship that allows for prediction and inference based on the independent variables [23].

SVM is a ML method employed for classification and regression tasks. In classification, SVM aims to obtain a hyperplane that best separates different classes by maximizing the margin between them. The equation for a linear SVM can be written as:

(7)

Here, represents the predicted output, w denotes the weight vector, x is the input feature vector, and is the bias term. The objective is to learn the optimal values of and that minimize the regression error while allowing for a specified tolerance margin. The regression SVM formula is:

(8)

In both cases, SVM can be extended to handle non-linear relationships employing kernel functions, which implicitly map the input data into a higher-dimensional space. This capability enables SVM to effectively capture complex patterns and make accurate predictions for various types of data. [24].

DTR is a non-linear regression algorithm used in ML. It predicts a continuous target variable by recursively splitting the feature space into subsets based on the input variables' values. Each split represents a decision node in the tree, leading to terminal nodes where predictions are made. The formula for DTR involves creating a tree structure that predicts the target variable by averaging the target values of the training samples within each terminal node. Mathematically, the DTR model can be represented as:

(9)

Here, represents the predicted target variable, is the target value of the training sample within the terminal node, and is the number of samples in the terminal node. DTR can capture complex relationships in data and handle non-linear patterns effectively. However, it can also be prone to overfitting if not appropriately controlled through hyperparameters or ensemble techniques like Random Forest or Gradient Boosting [25].

RF is a powerful ensemble ML algorithm that relates multiple decision trees to enhance prediction accuracy and decrease overfitting. It can be used for both classification and regression tasks. The formula for RF involves creating a gathering of decision trees, where each tree is trained on a subset of the data and potentially with different subsets of features. The predictions of individual trees are then combined through averaging (for regression) or voting (for classification) to obtain the final prediction. In the context of regression tasks, the formulation for aggregating predictions within a RF can be represented as:

(10)

where represents the number of individual trees within the Random Forest ensemble [19].

A group of blue dots

Description automatically generated

A group of blue dots

Description automatically generated