# Accelerated discovery of $M_6$@g-$N_4$ catalysts for $CO_2$ electroreduction via machine learning and DFT: Descriptor engineering and activity trend validation

**Ping Cheng[1], Hongyang Xu[1], Xiaoxiao Wang[1], Xiaoxiang Wang[2*], Tianci Wang[3], Chenyuan Yu[1], Wencong Sun[1], Yun Li[3], Weijia Huang[3], Chunguang Chen[1]**

1. School of Materials and Chemistry, University of Shanghai for Science and Technology, Shanghai, 200093, China
2. Institute for Carbon-Neutral Technology, Shenzhen Polytechnic University, Shenzhen, 518055, China
3. School of Energy and Power Engineering, University of Shanghai for Science and Technology, Shanghai, 200093, China

* Correspondence: E-Mail: wangxiaoxiang@szpu.edu.cn (X.W.)

**Principles of Machine Learning Algorithms**

(1) Ridge Regression

Ridge regression (RR) is a regularised linear method developed to handle predictor collinearity in statistics and machine learning. This method incorporates a quadratic regularization term into the ordinary least squares (OLS) framework to improve the ability to handle multiple linear regression problems. The principle of traditional OLS lies in estimating model parameters by minimizing the sum of squared errors between actual values and predicted values when dealing with general linear regression problems, thereby achieving fitting of the input data. However, when there is a strong correlation between input features, OLS may lead to model overfitting, which reduces the generalization ability. To alleviate the overfitting problem, a regularization term can

be added to the OLS framework to reduce model variance and enhance generalization ability. The objective function of Ridge Regression can be expressed as Equation (S1):

$$\min_{w}\left\{\sum_{i=1}^{n}\left(y_i-\sum_{j=1}^{p}x_{ij}w_j\right)^2+\alpha\sum_{j=1}^{p}\omega_j{}^2\right\} \tag{S1}$$

In the equation, "$n$" denotes the number of samples, "$p$" represents the number of features, "$y_i$" is the actual value of the $i$-th sample, "$x_{ij}$" stands for the $j$-th feature value of the $i$-th sample, "$\omega_i$" is the corresponding regression coefficient, and "$\alpha$" is the regularization term coefficient in Ridge Regression, which is used to adjust the strength of the regularization term.

(2) Decision Tree

A Decision Tree (DT) is a non-parametric supervised learning method that can summarize decision rules from datasets containing features and target values, and present these rules in a tree-like structure to solve classification or regression problems. Decision Tree Regression (DTR) constructs a model to predict the target value of new samples by learning the relationship between data features and target values. During the training process, the model starts from the root node and recursively splits the current dataset into two or more subsets. The criterion for selecting the optimal split point is to identify a specific feature or threshold such that the variance within the subsets is minimized after splitting based on this feature or threshold. The tree construction process continues until the stopping criteria are satisfied—for instance, when the tree reaches a preset maximum depth, the number of samples in a node falls below the minimum sample count required for splitting, or the variation in target values within the node becomes extremely small, rendering further splitting unable to yield significant improvements. In the prediction process, for a new sample, the model initiates from the root node and traverses down the tree according to the sample's feature values until it reaches a leaf node. The predicted value of the new sample is determined as the mean of the target values of all training samples contained in this leaf node.

DTR constructs the model by splitting the feature space, so it can handle non-linear relationships effectively. However, when the number of features is large, decision trees are highly prone to overfitting. Common improvement methods include adjusting the maximum depth and limiting the minimum number of samples required for node splitting. In addition, the Random Forest method based on ensemble learning can also improve the stability of decision trees and enhance the accuracy of predictions.

(3) Random Forest

Random Forest (RF) is an ensemble learning method based on the bagging strategy. By introducing random feature selection during the decision tree construction process, it effectively improves the prediction accuracy of the model. RF generates the final prediction result by constructing multiple decision trees and averaging their prediction results (for regression tasks) or conducting voting (for classification tasks). The prediction function of Random Forest can be expressed as Equation (S2):

$$H(x) = \frac{1}{T}\sum_{t=1}^{T} h_t(x) \tag{S2}$$

In the equation, $H(x)$ denotes the prediction result of the Random Forest for sample $x$; $T$ is the number of decision trees; and $h_t(x)$ represents the prediction result of the $t$-th decision tree for sample $x$. As shown in Figure S1, the Random Forest model consists of multiple decision trees, each trained on randomly sampled data and features from the original dataset. During the training process, each tree starts from the root node and splits based on the decision rules of internal nodes, ultimately outputting the prediction result at the leaf node. For classification tasks, each internal node of the decision tree represents a feature-based judgment, each branch corresponds to a different value of that feature, and the final leaf nodes correspond to different class labels.

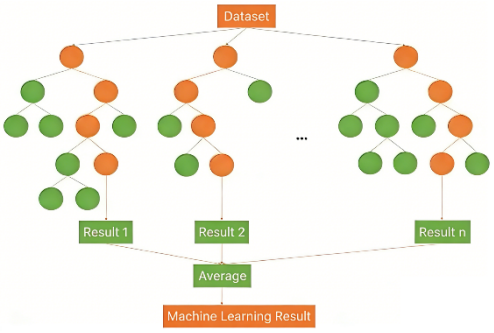


Figure S1. Schematic illustration of the random-forest structure.

(4) Gradient Boosting

Gradient Boosting (GB) constructs an accurate prediction model by iteratively integrating weak learners (e.g., decision trees). Gradient Boosting Machine (GBM) gradually reduces training errors through an iterative optimization strategy, and the specific form of its loss function can be selected according to the specific task (*e.g.*, regression or classification). The core iterative formula of GBM can be expressed as Equation (S3):

$$F_t(x) = F_{t-1}(x) + \eta \cdot h_t(x) \tag{S3}$$

In the formula, $F_t(x)$ denotes the prediction result of the model after the $t$-th iteration; $F_{t-1}(x)$ represents the prediction result after the $(t-1)$-th iteration; $\eta$ is the learning rate; and $h_t(x)$ stands for the weak learner (usually a decision tree) added in the $t$-th iteration. GBM reduces training errors through a stepwise optimization strategy, where each iteration fits the current prediction residual. It balances fitting ability and generalization ability by means of the learning rate $\eta$, thereby improving performance on unseen data, and has been widely applied in related fields.

(5) Support Vector Regression

Support Vector Regression (SVR) is a regression method developed based on the principle of structural risk minimization in statistical learning theory. As an extended application of Support Vector Machine (SVM) in regression problems, SVR constructs an $\varepsilon$-insensitive band to establish the optimal regression hyperplane, achieving optimal fitting of sample data while ensuring the generalization ability of the model. Its core

mechanism is as follows: An $\varepsilon$-insensitive loss function is introduced during the training process. No loss is calculated when the deviation between the predicted value and the true value does not exceed $\varepsilon$, and only linear penalties are imposed on samples exceeding this threshold. To achieve non-linear regression, SVR maps the original data to a high-dimensional feature space through a kernel function and solves a convex quadratic programming problem in this space. Its objective function includes both a regularization term (to control model complexity) and slack variables (to handle outlier samples), effectively preventing overfitting through structural risk minimization. After transforming the original optimization problem into a dual form via the Lagrange multiplier method, efficient algorithms such as Sequential Minimal Optimization (SMO) can be used for solving, ultimately obtaining a regression model determined by support vectors. This model can predict new samples using kernel tricks and optimal parameter sets (*C, ε,* kernel parameters), exhibiting excellent generalization performance while maintaining sparsity.

(6) Linear Regression

The basic assumption of the Linear Regression (LR) model is that there exist a linear relationship between independent variables and dependent variables. Specifically, the model attempts to fit data points using a straight line (extended to a hyperplane in high-dimensional space) to minimize prediction errors. Its mathematical expression is as Equation (S4):

$$Y = \beta_0 + \beta_1 X_1 + B_2 X_2 + \ldots + \beta_n X_n \quad \text{(S4)}$$

In the equation, $Y$ represents the target variable; $X_1$, $X_2$, $X_3$, ... $X_n$ denotes the independent variable features; $\beta_1$, $\beta_2$, $\beta_3$, ..., $\beta_n$ are the model parameters, where $\beta_0$ is the intercept. Equation (S4) reflects the basic principle of linear regression, *i.e.,* achieving the best fit to the data by optimizing parameters to minimize prediction errors. Parameter estimation can be solved by the Ordinary Least Squares (OLS) method, whose calculation formula is as Equation (S5):

$$\beta = \left(X'X\right)^{-1} X'Y \quad \text{(S5)}$$

In the formula, $\beta$ represents the parameter vector; $X$ is the design matrix, composed of

independent variable features; $Y$ is the observation vector of the target variable; and is $X'$ the transpose matrix of $X$. The OLS method estimates model parameters by minimizing the sum of squared residuals, where the residual represents the deviation between the observed value and the model's predicted value.

(7) K-Nearest Neighbors

K-Nearest Neighbors (KNN) is an instance-based, non-parametric learning method. Its basic idea is to predict the output of a new sample based on the classes or target values of its K nearest neighbor samples. In classification tasks, this method conducts voting on the class labels of the K nearest neighbor samples, and the final predicted class is the one with the most votes. In regression tasks, the predicted value is usually calculated as the average of the target values of the $K$ nearest neighbors, whose mathematical expression is as Equation (S6):

$$f(x)=\frac{1}{K}\sum_{i=1}^{K}y_i \tag{S6}$$

In the equation, $K$ represents the number of nearest neighbors; $y_i$ denotes the target values of the nearest neighbors. The selection of $K$ has a significant impact on the prediction performance of the KNN algorithm. If $K$ is too large, it may introduce neighbors with target values significantly different from the target, leading to large deviations in prediction results. Conversely, if $K$ is too small, the model's ability to characterize the overall data distribution may decline due to insufficient neighbor samples. Therefore, in practical applications, systematic strategies such as cross-validation are usually adopted to determine the optimal $K$. As a crucial part of hyperparameter tuning, this process can effectively balance the trade-off between the model's prediction accuracy and generalization ability.

(8) Multi-Layer Perceptron

The Multi-Layer Perceptron (MLP) is a typical feedforward artificial neural network model capable of mapping multiple input data to a single output. Its core characteristic lies in its hierarchical structure, which includes an input layer, one or more hidden layers, and an output layer. The input layer is responsible for receiving raw data, while the output layer generates the final prediction results. Hidden layers are

situated between the input and output layers, and both the number of these layers and the number of neurons in each can be optimized according to the specific requirements of the task. When constructing an MLP, there is no universal standard for the number of hidden layers; instead, an appropriate configuration can be selected based on factors such as dataset size, problem complexity, and available computational resources, in order to enhance the model's expressive capacity and generalization performance.

**Loss function selection**

In this study, all machine learning models were trained with the goal of minimizing the Mean Squared Error (*MSE*) to ensure consistency in the evaluation criteria for the regression task and to follow common practices in the field. For linear models and ensemble tree-based models, we utilized the default settings of their standard implementations. For the Multi-Layer Perceptron (MLP), we employed the PyTorch framework and selected the Adam optimizer to minimize the MSE loss. This choice is based on the widely recognized stability, adaptability, and efficient convergence of Adam in deep learning applications. All models were evaluated on the same training-validation-test dataset split to ensure a fair comparison.

**Methodology and Calculation Formulas**

In this study, we utilized the Ridge Regression (RR) model to obtain the weight coefficients for each feature descriptor. The top five ranked feature factors were selected, and their standardized data were incorporated into the following linear combination formula to calculate the intermediate variable Y:

$$Y = 0.361X_{Bader} - 0.166X_{HER} - 0.142X_{\angle CO2} + 0.107X_{EN} - 0.123X_{CE} \quad \text{(S7)}$$

Here, $X_{Bader}$, $X_{HER}$, $X_{\angle CO2}$, $X_{EN}$, and $X_{CE}$, represent the standardized values of the Bader charge, the hydrogen evolution reaction (HER) activity descriptor, the $CO_2$ bond angle, the electronegativity, and the cohesive energy, respectively. Subsequently, the predicted $CO_2$ adsorption free energy $\Delta G_{pred}$ for a catalyst is obtained through a denormalization transformation: where:

$$\Delta G_{pred} = Y \cdot \sigma_{target} + \mu_{target} \quad \text{(S8)}$$

$\sigma_{target}$ is the mean of the actual $CO_2$ adsorption free energy ($\Delta G$) for all catalyst samples in the training set. $\mu_{target}$ is the standard deviation of the actual $CO_2$ adsorption free energy ($\Delta G$) for all catalyst samples in the training set.

The interaction between a catalyst and the reaction intermediates must exhibit an optimal binding strength. It should be sufficiently strong to effectively activate the reactants, but not too strong to the intermediates poison the surface by becoming irreversibly bound, thereby inhibiting desorption and turnover. Through high-throughput calculations on 2,669 CO adsorption configurations, Wang[1] determined the optimal CO adsorption energy to be -0.67 eV, with a tolerance of ±0.1 eV. Moreover, for the design of N-doped graphene-supported metal catalyst, the optimal adsorption energy of CO was found to be -0.53 eV [2]. To quantitatively evaluate catalytic performance, we set $\Delta G_{opt}$=−0.5 eV as the ideal adsorption energy benchmark, based on the Sabatier principle and existing literature. The absolute deviation *"D"* between the predicted value and this benchmark is calculated as:

$$D = \left|\Delta G_{pred} - \Delta G_{opt}\right| = \left|\Delta G_{pred} + 0.5\right| \quad \text{(S9)}$$

## References


1. D. Wang, R. Cao, S. Hao, C. Liang, G. Chen, P. Chen, Y. Li and X. Zou, Green Energy & Environment, 2023, 8, 820-830.
2. H. Wang, R. Hu, R. Zhu, X. Yang, S. Yang, Y. Nie, J. Yu and X. Jiang, Applied Surface Science, 2025, 692.

Table S1. Partial statistical information of the dataset

| $M_6$@g-$N_4$ | ΔG-$CO_2$ | HER | Bader | ∠$CO_2$ | adv-O-C | C-M | AN | AR | VDM | EA | EN | IP | CE | I1 | WF | TC |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Tl | -0.552086 | -0.633502 | 0.035276 | 179.517 | 1.1775 | 0 | 81 | 196 | 196 | 30.88 | 1.62 | 6.1 | 1.19 | 6.1 | 4 | 46.5 |
| Au | -0.422274 | 0.547163 | -0.007527 | 179.131 | 1.1775 | 0 | 79 | 144 | 166 | 222.75 | 2.54 | 9.22 | 3.79 | 9.22 | 5.1 | 315 |
| Pt | 0.335013 | 0.418239 | 0.361501 | 144.271 | 1.2405 | 2.011 | 78 | 135 | 216 | 205.04 | 2.28 | 8.96 | 5.84 | 8.95 | 5.65 | 71.6 |
| Re | 0.953022 | 0.854238 | 0.841659 | 134.782 | 1.274 | 2.061 | 75 | 197 | 210 | 5.82 | 1.9 | 7.3 | 8.9 | 7.83 | 4.9 | 48 |
| W | -0.211305 | 0.122435 | 0.010523 | 179.366 | 1.1765 | 0 | 74 | 137 | 215 | 78.76 | 2.36 | 7.98 | 8.66 | 7.86 | 4.5 | 173 |
| Ta | 2.047283 | 0.647319 | 1.044296 | 133.958 | 1.2795 | 2.1765 | 73 | 140 | 222 | 31 | 1.5 | 7.6 | 8.06 | 7.54 | 4.6 | 57 |

| | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Ba | 1.487127 | 0.043174 | 1.443603 | 120.498 | 1.292 | 0 | 56 | 217 | 222 | 13.95 | 1.89 | 5.21 | 1.04 | 5.21 | 2.5 | 25 |
| Sb | -0.501916 | -0.12473 | 0.024656 | 179.713 | 1.1705 | 0 | 51 | 140 | 220 | 101.05 | 2.05 | 8.64 | 2.68 | 8.6 | 4.3 | 24 |
| Sn | -0.489491 | -0.911728 | 0.02386 | 179.741 | 1.1765 | 0 | 50 | 196 | 217 | 107.29 | 1.96 | 7.34 | 1.77 | 7.34 | 4.5 | 66 |
| In | -0.469066 | -0.175203 | 0.042614 | 179.681 | 1.177 | 0 | 49 | 144 | 207 | 37.04 | 1.78 | 5.79 | 0.74 | 5.78 | 4.1 | 81 |
| Cd | -0.462752 | -1.231544 | 0.051868 | 178.876 | 1.1775 | 0 | 48 | 150 | 207 | -68 | 1.69 | 6.28 | 1.27 | 8.99 | 4 | 96 |
| Ag | -0.296048 | -0.752384 | 0.016128 | 179.722 | 1.1765 | 0 | 47 | 126 | 172 | 125.86 | 1.93 | 7.57 | 2.92 | 7.57 | 4.26 | 429 |
| Pd | 0.118387 | 0.239396 | 0.320698 | 151.481 | 1.2155 | 2.092 | 46 | 137 | 163 | 54.24 | 2.2 | 8.34 | 8.82 | 8.33 | 5.1 | 72 |
| Rh | 0.134676 | 0.829033 | 0.431723 | 145.316 | 1.237 | 2.019 | 45 | 136 | 163 | 110.27 | 2.28 | 7.46 | 8.91 | 7.45 | 4.7 | 150 |
| Ru | 0.58336 | 0.281305 | 0.551203 | 141.690 | 1.2465 | 2.035 | 44 | 189 | 205 | 100.27 | 2.2 | 7.46 | 8.78 | 7.36 | 4.6 | 150 |

| | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Nb | 1.987528 | -0.562201 | 1.121636 | 132.391 | 1.2805 | 2.204 | 41 | 140 | 218 | 88.51 | 1.6 | 6.76 | 8.29 | 6.75 | 4.6 | 53 |
| Zr | 2.915666 | 1.399057 | 1.289604 | 131.593 | 1.288 | 2.2105 | 40 | 160 | 205 | 41.8 | 1.33 | 6.63 | 5.23 | 6.63 | 4.5 | 22 |
| Y | 0.606664 | 0.040986 | 1.187674 | 125.768 | 1.294 | 2.356 | 39 | 180 | 163 | 29.6 | 1.22 | 6.22 | 1.81 | 6.21 | 4.2 | 22 |
| Sr | 0.805159 | 0.046543 | 0.898281 | 1340752 | 1.2535 | 0 | 38 | 112 | 215 | 5.02 | 0.95 | 5.03 | 1.96 | 5.69 | 2.9 | 35 |
| Ge | -0.504618 | -0.655723 | 0.022897 | 179.469 | 1.1765 | 0 | 32 | 135 | 211 | 118.93 | 2.01 | 7.88 | 4.02 | 7.89 | 4.6 | 60 |
| Ga | -0.488112 | -0.294147 | 0.027506 | 179.631 | 1.177 | 0 | 31 | 156 | 187 | 29.06 | 1.81 | 5.99 | 1.15 | 5.99 | 4.3 | 40 |
| Zn | -0.487979 | -0.660544 | 0.029652 | 179.962 | 1.177 | 0 | 30 | 134 | 215 | -58 | 1.65 | 9.39 | 1.31 | 9.39 | 4.3 | 116 |
| Cu | -0.2941 | -0.171528 | 0.011312 | 179.706 | 1.1775 | 0 | 29 | 132 | 140 | 119.23 | 1.9 | 7.72 | 3.49 | 7.72 | 4.7 | 398 |
| Ni | -0.047948 | 0.306878 | 0.421596 | 148.649 | 1.228 | 1.912 | 28 | 162 | 163 | 111.65 | 1.91 | 7.63 | 5.15 | 7.63 | 5.01 | 90 |

| | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Mn | 0.155867 | 0.194603 | 0.7037761 | 136.633 | 1.2575 | 1.979 | 25 | 179 | 225 | -50 | 1.55 | 7.43 | 6.68 | 7.43 | 4.3 | 7.8 |
| Cr | 1.185195 | 0.408491 | 0.82707 | 140.625 | 1.263 | 2.013 | 24 | 185 | 180 | 65.21 | 1.66 | 6.77 | 6.6 | 6.76 | 4.5 | 93.9 |
| V | 2.034122 | 0.612191 | 1.022071 | 135.716 | 1.276 | 2.043 | 23 | 135 | 207 | 50.91 | 1.63 | 6.74 | 5.55 | 6.74 | 4.33 | 30.7 |
| Ti | -0.285838 | 0.92583 | 0.045692 | 179.016 | 1.177 | 2.247 | 22 | 147 | 140 | 7.28 | 1.54 | 6.82 | 4.47 | 6.82 | 4.33 | 21.9 |
| Sc | 0.624625 | -0.041309 | 1.006823 | 127.231 | 1.291 | 2.187 | 21 | 261 | 144 | 18 | 1.36 | 6.56 | 2.88 | 6.56 | 4.3 | 15 |
| Ca | 1.434614 | 0006315 | 1.057259 | 132.109 | 1.2785 | 2.449 | 20 | 197 | 231 | 2.37 | 1 | 6.11 | 1.58 | 6.11 | 2.9 | 156 |
| Al | -0.569745 | 0.601878 | 0.071635 | 179.573 | 1.177 | 0 | 13 | 182 | 205 | 41.76 | 1.61 | 5.99 | 3.39 | 5.98 | 4.28 | 237 |
| Mg | -0.433714 | 0.036369 | 0.078371 | 179.288 | 1.177 | 0 | 12 | 160 | 173 | -40 | 1.31 | 7.65 | 1.53 | 7.64 | 3.66 | 156 |

Table S2. Performance metrics of eight machine learning models on the training and test sets

| Model | $R^2$ (train/test) | RMSE(train/test) | MAE(train/test) |
|---|---|---|---|
| RR | 0.892/0.887 | 0.290/0.334 | 0.201/0.257 |
| DT | 0.908/0.855 | 0.290/0.334 | 0.225/0.278 |
| RF | 0.912/0.893 | 0.262/0.326 | 0.239/0.245 |
| GB | 0.964/0.837 | 0.167/0.382 | 0.152/0.294 |
| SVR | 0.907/0.685 | 0.269/0.551 | 0.221/0.397 |
| LR | 0.969/0.688 | 0.157/0.555 | 0.133/0.396 |
| KNN | 0.854/0.809 | 0.337/0.415 | 0.227/0.293 |
| MLP | 0.615/0.481 | 0.548/0.616 | 0.412/0.501 |

**Model Performance Analysis**

The key evaluation parameters ($R^2$, RMSE, and MAE) of all eight ML models on both the training and test datasets are explicitly provided in Table S2, addressing the requirement for a rigorous assessment of model performance. From Table S2, the Ridge Regression (**RR**) model is the most robust and well-generalized approach, achieving high and closely matched $R^2$ values of 0.892 (train) and 0.887 (test), with a negligible performance drop of only 0.005 between the two sets. Concurrently, its prediction errors is low and stable. The RMSE of RR increases marginally from 0.290 (train) to 0.334 (test), while MAE rises only slightly from 0.201 (train) to 0.257 (test). This near-parity in performance across training and test data directly indicates minimal overfitting and excellent generalization capability.

The Random Forest (**RF**) model also exhibits strong performance, with high $R^2$ values of 0.912 (train) and 0.893 (test) and only minor error increases, confirming its good generalization. However, its slightly larger performance gap (0.019) compared to RR (0.005), combined with its higher complexity, makes RR the preferred choice. The Gradient Boosting (**GB**) model shows clear overfitting, as its training $R^2$ of 0.964 drops sharply to 0.837 on the test set, while RMSE rises from 0.167 to 0.382. Similarly, the Linear Regression (**LR**) and Support Vector Regression (**SVR**) models display dramatic performance degradation, with test $R^2$ values plummeting to 0.688 and 0.685, respectively, alongside corresponding surges in prediction errors. These large

discrepancies between training and test performance confirm that these models overfit the training data. The Decision Tree (**DT**) and K-Nearest Neighbors (**KNN**) models show moderate performance but with noticeable generalization gaps. DT's training $R^2$ of 0.908 drops to 0.855 on the test set, while KNN's $R^2$ falls from 0.854 to 0.809, both indicating weaker generalization compared to RR. Finally, the Multi-Layer Perceptron (**MLP**) exhibits the poorest overall performance, with low $R^2$ values of 0.615 (train) and 0.481 (test) and consistently high errors across both datasets, which is indicative of significant underfitting and an inability to capture the underlying structure of the data.

Table S3. Model Evaluation Metrics

| Model | $R^2$ | MAE | MSE | RMSE |
|---|---|---|---|---|
| RR | 0.963 | 0.022 | 0.052 | 0.228 |
| DT | 0.928 | 0.142 | 0.102 | 0.319 |
| RF | 0.945 | 0.152 | 0.078 | 0.279 |
| GB | 0.954 | 0.469 | 0.065 | 0.255 |
| SVR | 0.914 | 0.210 | 0.121 | 0.348 |
| LR | 0.865 | 0.153 | 0.189 | 0.435 |
| KNN | 0.856 | 0.240 | 0.204 | 0.452 |
| MLP | 0.798 | 0.033 | 0.285 | 0.534 |

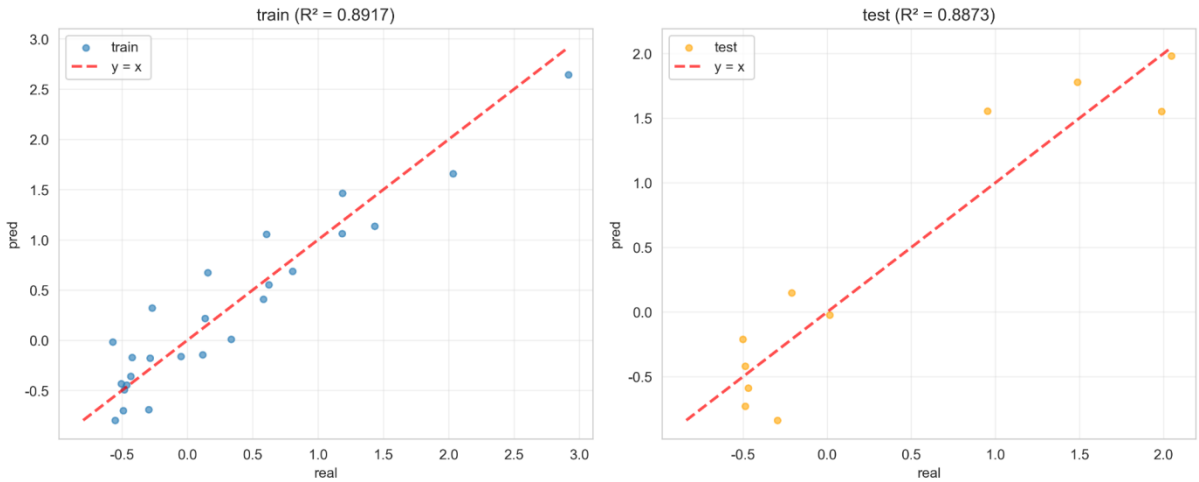


Figure S2. y=x scatter plots of predicted and actual values for the RR model in training and test sets.

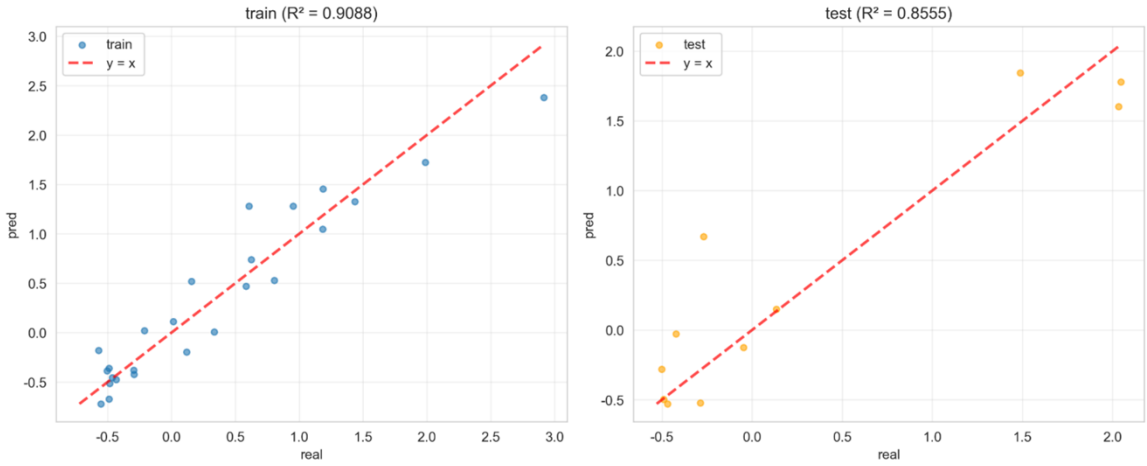


Figure S3. y=x scatter plots of predicted and actual values for the DT model in training and test sets.

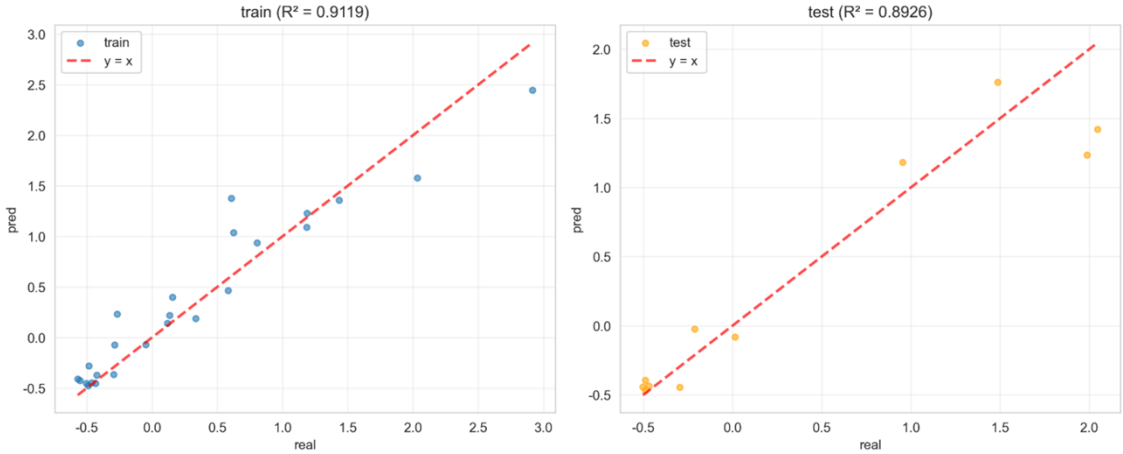


Figure S4. y=x scatter plots of predicted and actual values for the RF model in training and test sets.

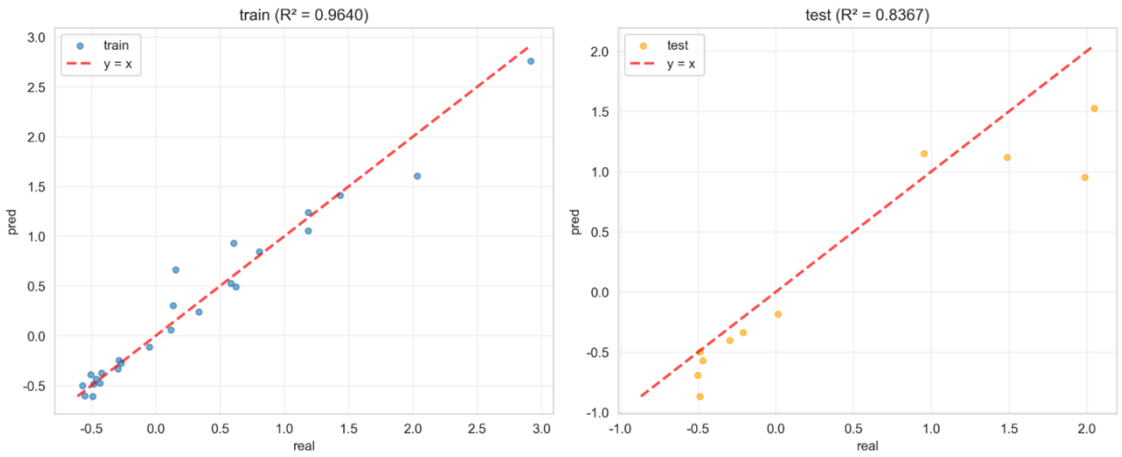


Figure S5. y=x scatter plots of predicted and actual values for the GB model in training and test sets.

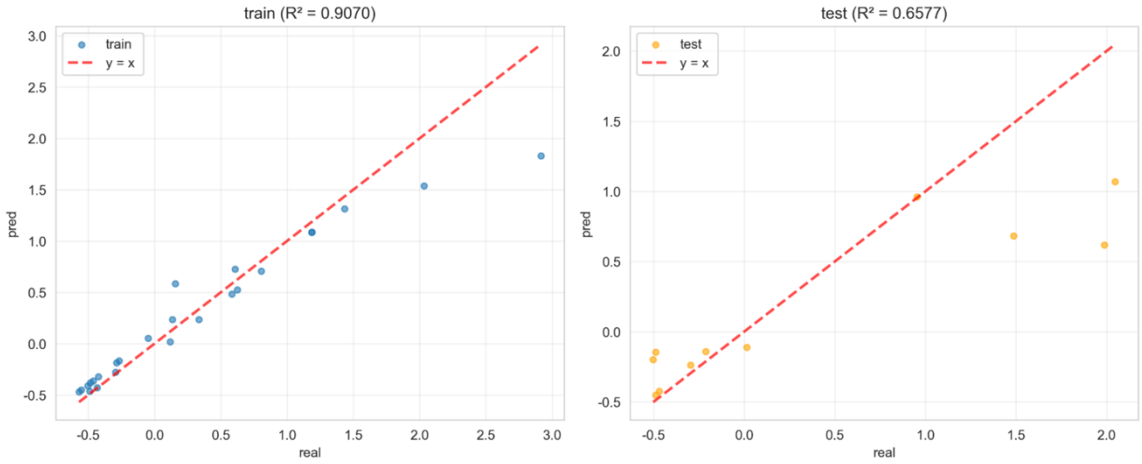


Figure S6. y=x scatter plots of predicted and actual values for the SVR model in training and test sets.

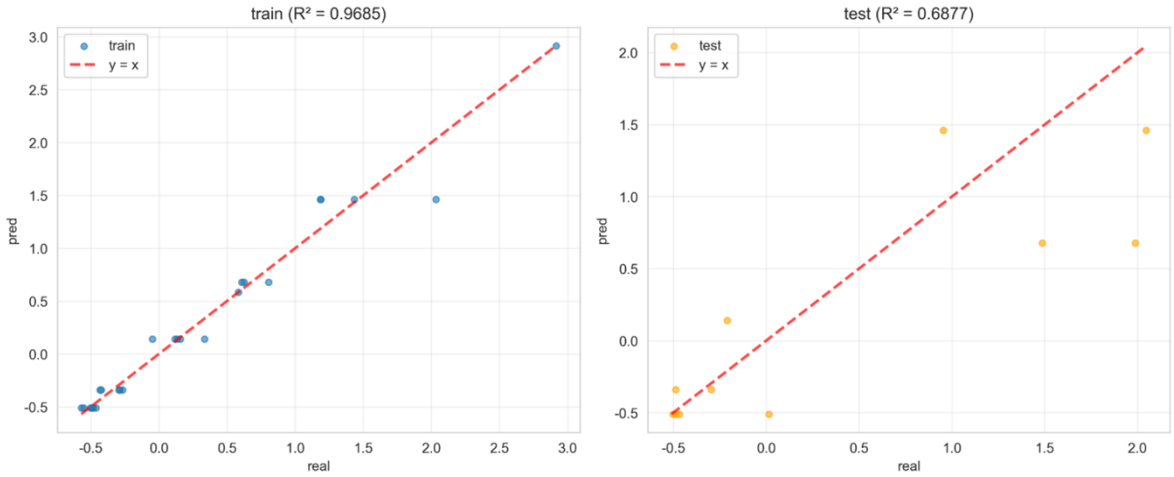


Figure S7. y=x scatter plots of predicted and actual values for the LR model in training and test sets.

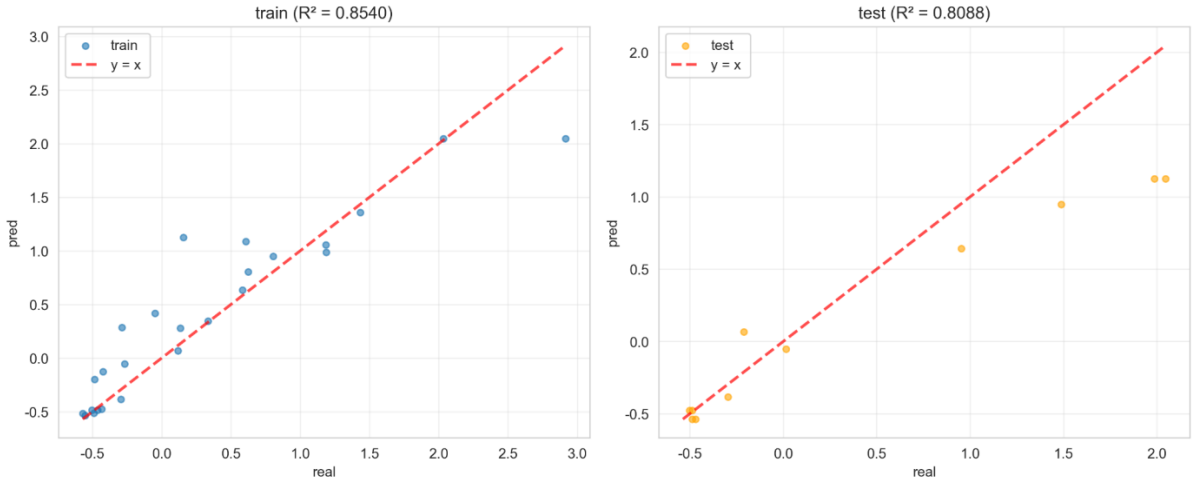


Figure S8. y=x scatter plots of predicted and actual values for the KNN model in training and test sets.

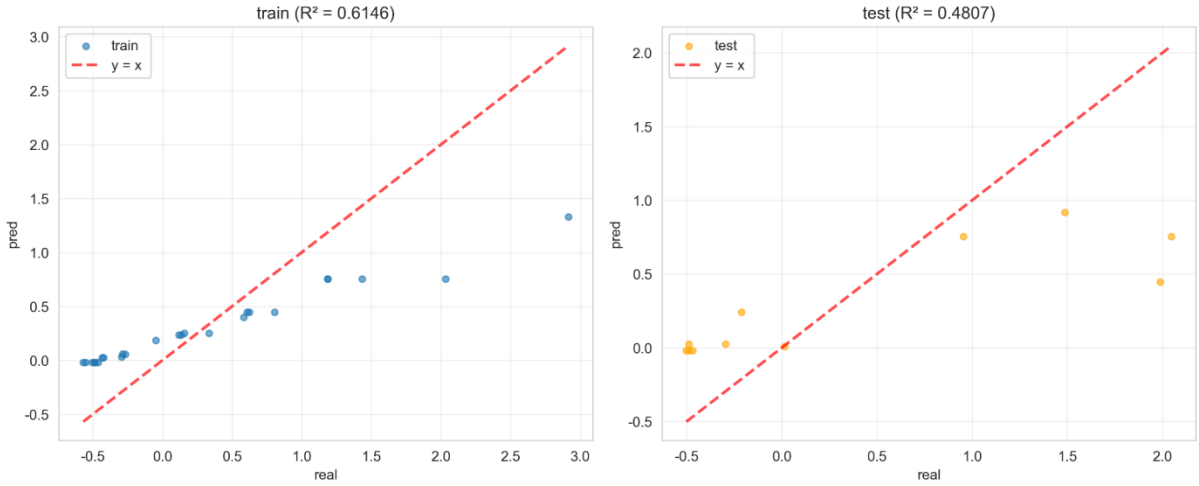


Figure S9. y=x scatter plots of predicted and actual values for the MLP model in training and test sets.