Machine Learning-Driven Analysis of Activation Energy for Metal Halide Perovskites

Vimi Patel[(0009-0005-5364-5683)a], *Kunjrani Sorathia*[(0009-0007-6986-4519)a], *Kushal Unjiya*[(0009-0001-3521-1858)a], *Raj Dashrath Patel*[(0009-0000-4983-6915)b], *Siddhi Vinayak Pandey*[(0000-0002-1574-2846)*b], *Abul Kalam*[(0000-0002-0930-3791)c], *Daniel Prochowicz*[(0000-0002-5003-5637)d], *Seckin Akin*[(0000-0001-9852-7246)e], *Pankaj Yadav*[(0000-0002-1858-8397) * b,f]

[a] *Department of Information and Communication Technology, Adani University, Ahmedabad-382421, Gujarat, India*

[b] *Department of Solar Energy, School of Energy Technology, Pandit Deendayal Energy University, Gandhinagar-382007, Gujarat, India.*

[c] *Department of Chemistry, Faculty of Science, King Khalid University, Abha 61413, P.O. Box 9004, Saudi Arabia*

[d] *Institute of Physical Chemistry, Polish Academy of Sciences, Kasprzaka 44/52, 01-224 Warsaw, Poland.*

[e] *Department of Metallurgical and Materials Engineering, Karamanoglu Mehmetbey University, 70200, Karaman, Turkey.*

[f] *Department of Physics, School of Energy Technology, Pandit Deendayal Energy University, Gandhinagar-382 007, Gujarat, India.*

Corresponding email: siddhivinayak.ele17@gmail.com, pankajphd11@gmail.com, pankaj.Yadav@sse.pdpu.ac.in

1. Materials and Methods

1.1. Crystal synthesis and Characterization

High-quality MAPbI3 and MAPbBr3 crystals were synthesized in this study following the method reported by Makhsud et al.[1] Electrochemical impedance spectroscopy (EIS) measurements were subsequently performed using a Bio-Logic SP-300 potentiostat equipped with a frequency response analyzer and the Semiconductor Analysis and Testing Solutions (SATs) probe station. For the measurements, a 100 nm-thick conductive carbon paste was applied to the MHP single crystals (SCs) with an electrode spacing of 1 mm. The coated crystals were then dried in an oven at 80 °C for 10 minutes to ensure proper adhesion and conductivity.

1.2. Computational Analysis

The machine learning implementation in this study was conducted primarily on Google Colab, accessed via a local system equipped with an AMD Ryzen 3 5300U processor, Radeon Graphics, 8.00 GB RAM, and a 64-bit Windows 11 operating system. Google Colab provided a robust computational environment for feature engineering, model training, and data visualization, facilitating efficient execution and analysis.

2. Temperature dependent EIS:

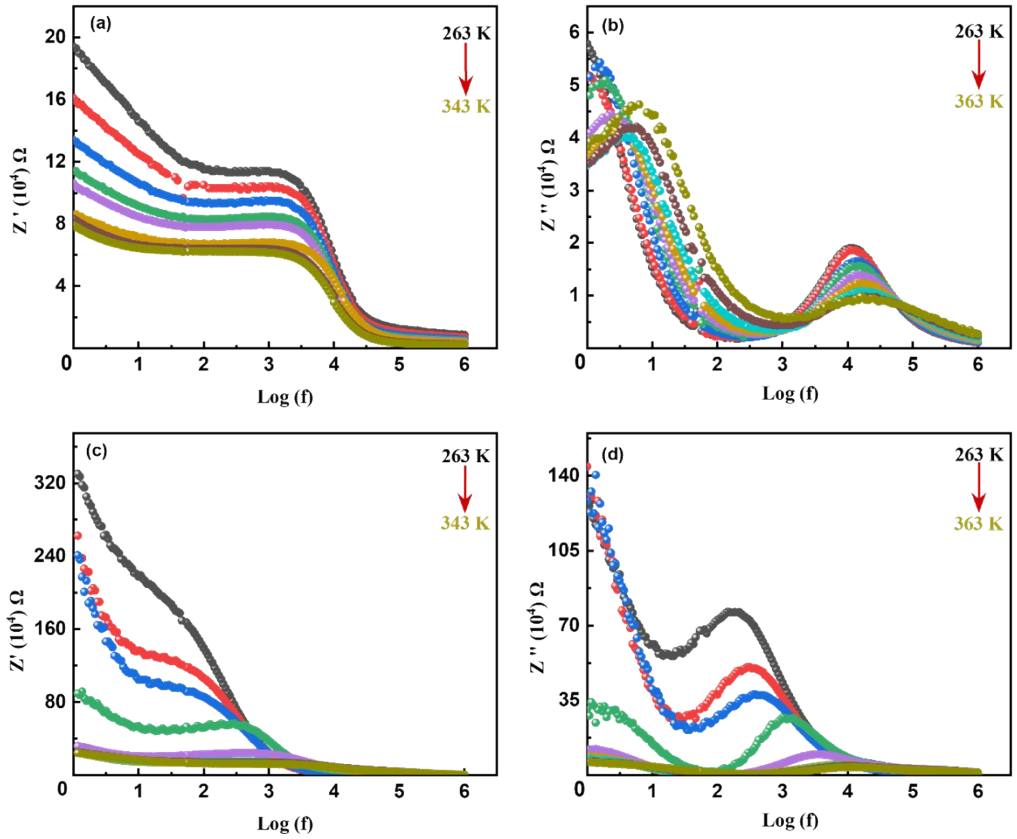

Figure S1. Figure S1 presents the temperature dependence of real and imaginary part of EIS as a function of Frequency. Figure S1 (a) and (c) depicts real impedance(Z') of both the SCs with a successive change in slope around at mid-frequency range (KHz). Figure S1 (b) and (d) illustrates the imaginary impedance (Z'') of both the SCs as a function of frequency at different temperatures.

3. Outlier Detection and Handling:

3.1. Outlier Detection using Box plots

Figure S2 illustrates the key statistical measures for two input features (Temperature and Frequency), with a primary aim on identifying outliers in this study. Given the limited number of datapoints in the Ionic Radius feature of the dataset, a box plot for the same would not provide any useful insight and hence is not included here. As is evident from the figure, the Frequency feature of the dataset has a considerable number of outliers that could adversely affect prediction performance if not addressed. Additionally, the figure also provides some other valuable information, such as the range, median, and mean values of the features among others, which proved beneficial for further analysis in this study.

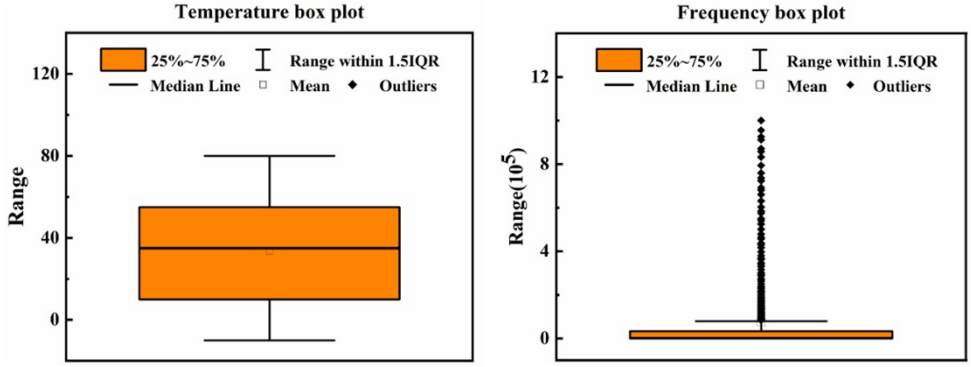

Figure S2. Box plots illustrating the distributions of two input features: temperature and frequency

3.2. Outlier Handling

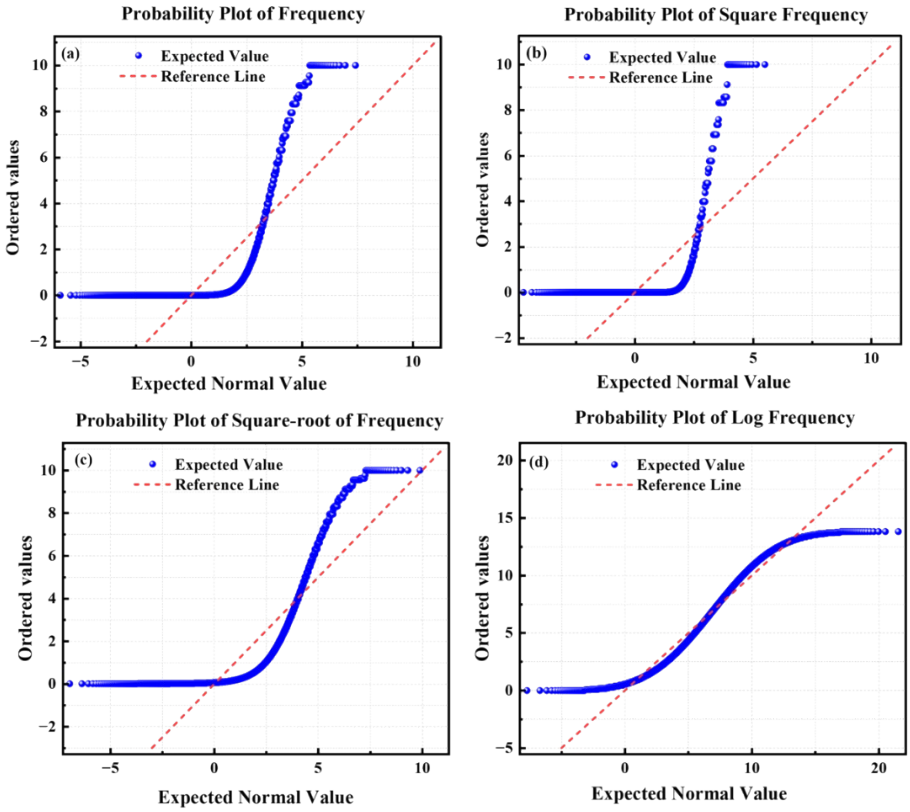

Figure S3. Probability plot of: (a) Original Frequency, (b) Square transformed Frequency, (c) Square Root-transformed Frequency, (d) Log transformed Frequency.

Figure S3 illustrates the transformation of the frequency feature's distribution. The frequency column in our dataset measured from 1 MHz to 1 Hz, created challenges due to the significant presence of outliers in observed real and imaginary values, leading to skewed distributions. To handle outliers and mitigate their impact on predictions, we used various mathematical transformations (i.e. log, square and square root), which offer an effective method for addressing outliers by compressing the range of values and reducing their influence whose further information is provided through the figure S3. While square and square root transformations reduced the number of outliers, the log transformation proved to be the most effective. Mathematically, the log transformation stabilizes variance by reducing the magnitude of larger values more aggressively than smaller ones.[2] This normalization brings the data closer to a Gaussian distribution and enhances the overall performance of ML model.[3] As shown in figure S3, the transformed frequency data align closely with a red line indicating reduced range and fewer outlier after applying the log transformation. There are two ways this can impact the model performance. First, the reduced influence of outliers allows the model to focus on the underlying patterns in the data, rather than being skewed by extreme values. Second, the normalized scale ensures better convergence during model training, as features with more uniform distributions contribute equally to the learning process.[3] After transforming the data, it was crucial to ensure that each feature contributed equally during the training and testing stages. Since the features initially had varying ranges, this required scaling the dataset to a consistent range to prevent any single feature from disproportionately affecting the model's performance. We chose the Min-Max scaling technique to scale the transformed data to the default range of [0,1], further improving the training process. Ideally, the feature scaling, or normalization, is an essential step in data preprocessing, particularly when features like the frequency in our dataset have a much larger scale compared to others. By using Python's sklearn library and applying the MinMax Scaler, we effectively rescaled the features to a consistent range, enhancing the model's ability to learn

from the data. While other scaling techniques are available, we opted for Min-Max scaling due to its simplicity and interpretability.

4. Performance Comparison and Model Optimization:

4.1. Performance Comparison

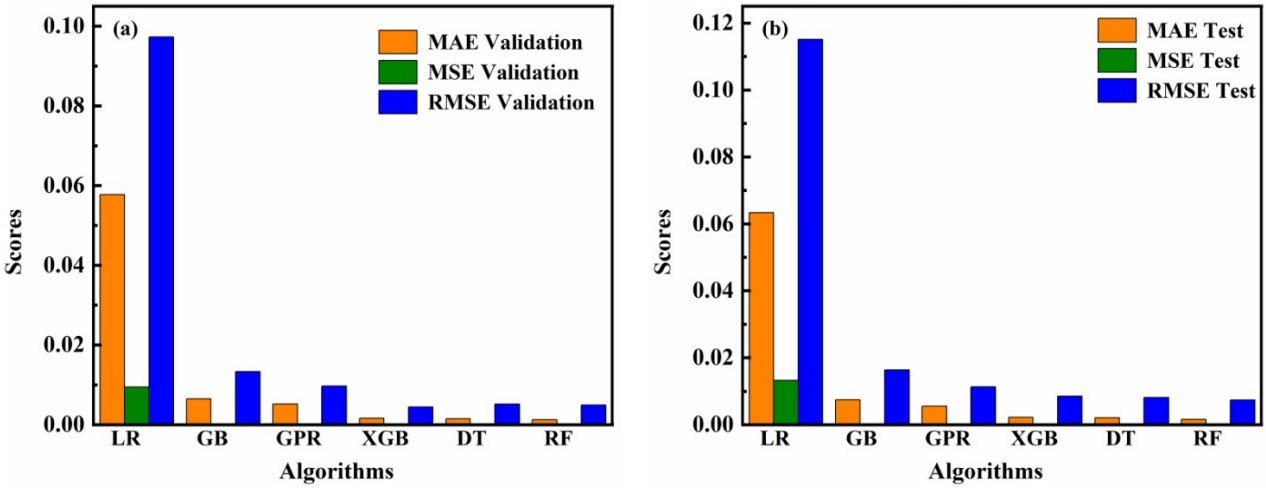

Figure S4. Visualization of error metrics (MAE, MSE, RMSE) of different regression algorithms based on (a) Validation Data, (b) Test Data of MHPSCs, $MAPbBr_3$ and $MAPbI_3$ respectively.

In our analysis of six algorithms—Linear Regression, Gradient Boosting, Gaussian Process Regressor, Random Forest, Decision Tree, and XGBoost, **Figure S4** presents performance comparison based on three performance metrics (MAE, MSE, RMSE) on the validation dataset in S4(a), and on the test dataset in S4(b). The Decision Tree (DT) model was excluded due to overfitting, as its zero error on the training dataset and significantly higher error on the validation dataset (Figure. S4). This difference in error highlights that the model was overly focused on capturing specific details and noise in the training data, leading to a lack of generalization. Instead of learning broader patterns, the DT model memorized specific pattern to the training set, which resulted in poor performance when applied to unseen data.[4] While hyperparameter tuning partially mitigated this issue, with a reduction in max_depth compared to the base model suggesting early stopping, these adjustments were insufficient. The DT model's performance on the validation and test datasets still lagged behind that of Gradient Boosting (GB), Random Forest (RF), and XGBoost (XGB). Therefore, the DT model was discarded from further testing on unseen data.

4.2. Error metrics

Mean Squared Error (MSE): MSE is a cost function that calculates the average of the squares of the errors .i.e., the average squared difference between the estimated values and the actual value.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \bar{y}_i)^2 \quad (1)$$

where y_i is the actual value, $\bar{y}_i$ is the predicted value, and n is the number of observations.

Mean Absolute Error (MAE): MAE is the average absolute errors in a set of predictions, ignoring the direction of the errors. It's computed as:

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \bar{y}_i| \quad (2)$$

where y_i is the actual value, $\bar{y}_i$ is the predicted value, and n is the number of observations.

Root Mean Squared Error (RMSE): RMSE is a cost function that can be represented as the square root of the mean square error, bringing the scale of the errors to be the same as the scale of targets.

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \bar{y}_i)^2} \quad (3)$$

where y_i is the actual value, $\bar{y}_i$ is the predicted value, and n is the number of observations.

R-Squared (R^2): R-Squared indicates the proportion of the variance in the dependent variable that is predictable from the independent variables.

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \bar{y}_i)^2}{\sum_{i=1}^{n}(y_i - y'_i)^2} \quad (4)$$

where y_i is the actual value, $\bar{y}_i$ is the predicted value, and y'_i is the mean of the actual values, and n is the number of observations.

4.3. Model Optimization

	Random Forest Regression
Hyperparameters	max_depth: 30 n_estimators: 500 random_state: 42

Table S1. Tuned parameter of Chosen model Random Forest Regression

The top-performing models from the performance comparison were further optimized using Grid Search CV, with Random Forest emerging as the preferred regression method due to its lowest error rates. After hyperparameter tuning with GridSearchCV, the details for Random Forest are provided in Table S1.

4.4. Comparison of Actual and Predicted data

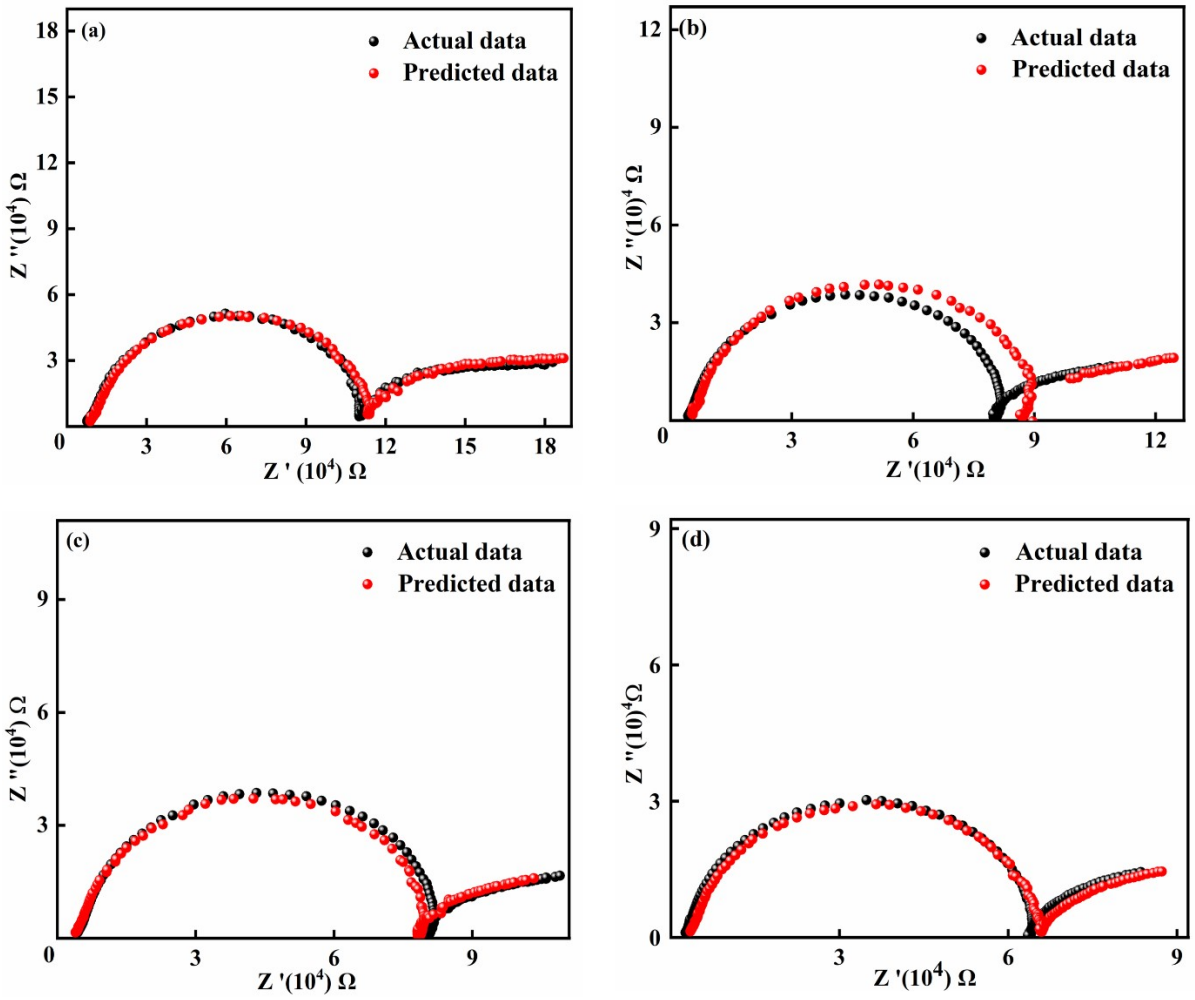

Figure S5. Comparison of actual and predicted Nyquist plots using RF regressor for $MAPbI_3$ at (a) $268\ K$, (b) $293\ K$, (c) $298\ K$, (d) $328\ K$

Figure S5 illustrates the comparison between actual and predicted Nyquist plots at four distinct temperatures, none of which were included in the training, validation or test data for the machine learning model. The data at these temperatures were entirely unseen by the Random Forest Regressor during model development. The plots offer a clear visual representation of the model's predictive accuracy and performance across different conditions, effectively demonstrating the strong capability of the Random Forest Regressor to generalize and make accurate predictions even for data outside the range used for training.

5. Fitted Nyquist plots and Parameter Extraction at unseen temperatures:

5.1. Fitting Nyquist plots at unseen temperatures on the predicted data

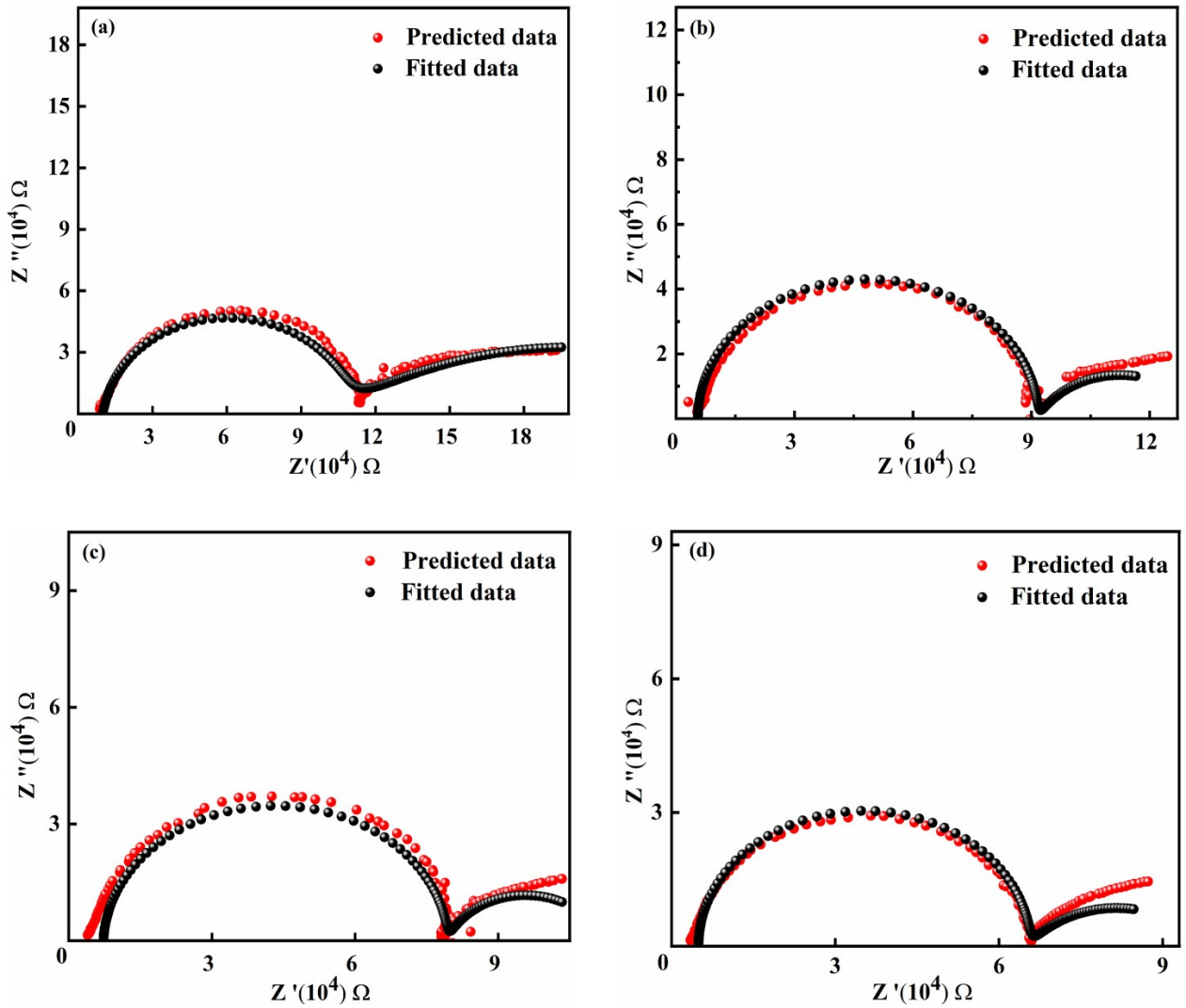

Figure S6. Fitted Nyquist plots for $MAPbI_3$ based on predicted data at (a) $268\,K$, (b) $293\,K$, (c) $298\,K$, (d) $328\,K$

Figure S6 illustrates the fitted Nyquist plots for $MAPbI_3$ based on the data predicted at the unseen temperatures using the Random Forest Regressor. Fitting was carried out using the "custom circuit" function from the impedance library to match the expected circuit shape for which the (R- (R, CPE) - (R, CPE)) Circuit was used. Initial estimates were provided for the fitting parameters to initiate the curve-fitting effectively and subsequent optimized fitting parameters were extracted from the same.

5.2. Parameter extraction of circuit fitting at unseen temperatures

Parameter	Parameter Value	Standard Deviation (±)
R_0	3.08e+03	1.17e+02 ohm
CPE_1_0	2.87e-10	4.91e-11 $F/s^{(a-1)}$
a_0	0.993	
R_1	6.12e+04	1.98e+02 ohm
R_2	3.85e+04	1.23e+01 ohm
CPE_2_0	5.82e-06	1.37e-07 $F/s^{(a1)}$
a_1	0.813	

Table S3. Extracted Fitting Parameters for the (R- (R, CPE) - (R, CPE)) Circuit at $333\,K$ of $MAPbBr_3$

Parameter	Parameter Value	Standard Deviation (±)
R_0	1.07e+04	4.63e+02 ohm
CPE_1_0	1.12e-09	1.69e-10 $F/s^{(a-1)}$
a_0	0.770	
R_1	1.32e+05	2.86e+02 ohm
R_2	1.91e+05	2.07e+01 ohm
CPE_2_0	8.68e-07	2.32e-08 $F/s^{(a1)}$
a_1	0.832	

Table S4. Extracted Fitting Parameters for the (R- (R, CPE) - (R, CPE)) Circuit at $333\,K$ of $MAPbI_3$

Parameter	Parameter Value	Standard Deviation (±)
R_0	5.53e+03	9.19e+02 ohm
R_1	8.60e+04	1.96e+03 ohm
CPE_1_0	2.10e-10	3.32e-10 $F/s^{(a-1)}$
a_0	0.999	
R_2	4.20e+04	2.79e+02 ohm
CPE_2_0	4.84e-06	8.17e-07 $F/s^{(a1)}$
a_1	0.724	

Table S5. Extracted Fitting Parameters for the (R- (R, CPE) - (R, CPE)) Circuit at $293\,K$ of $MAPbI_3$

Parameter	Parameter Value	Standard Deviation (±)
R_0	9.51e+03	6.45e+02 ohm
R_1	9.35e+04	1.15e+01 ohm

CPE_1_0	2.75e-10	1.30e-10 $F/s^{(a□0)}$
a_0	0.967	
R_2	1.92e+05	9.24e+00 ohm
CPE_2_0	2.62e-06	7.33e-08 $F/s^{(a1)}$
a_1	0.415	

Table S6. Extracted Fitting Parameters for the (R- (R, CPE) - (R, CPE)) Circuit at $268\ K$ of $MAPbI_3$

Parameter	Parameter Value	Standard Deviation (±)
R_0	7.24e+03	3.52e+02 ohm
R_1	7.21e+04	6.33e+02 ohm
CPE_1_0	3.29e-10	1.37e-10 $F/s^{(a□0)}$
a_0	0.974	
R_2	3.29e+04	1.24e+02 ohm
CPE_2_0	3.68e-06	2.81e-07 $F/s^{(a1)}$
a_1	0.786	

Table S7. Extracted Fitting Parameters for the (R- (R, CPE) - (R, CPE)) Circuit at $298\ K$ of $MAPbI_3$

Parameter	Parameter Value	Standard Deviation (±)
R_0	4.92e+03	2.89e+02 ohm
R_1	6.05e+04	6.97e+02 ohm
CPE_1_0	2.24e-10	1.60e-10 $F/s^{(a□0)}$
a_0	1.00	
R_2	3.23e+04	1.23e+02 ohm
CPE_2_0	7.32e-06	5.23e-07 $F/s^{(a1)}$
a_1	0.619	

Table S8. Extracted Fitting Parameters for the (R- (R, CPE) - (R, CPE)) Circuit at $328\ K$ of $MAPbI_3$

References:

1 Saidaminov, M. I.; Abdelhady, A. L.; Murali, B.; Alarousu, E.; et al., High-quality bulk hybrid perovskite single crystals within minutes by inverse temperature crystallization, *Nat. Commun.*, 2015, 6, 7586.

2 Atkinson, A. C., Non-constant Variance and the Design of Experiments for Chemical Kinetic Models, in *Computer Aided Chemical Engineering*, Edited by S. P. Asprey and S. Macchietto, Elsevier, 2003, 16, 141–158.

3 Feng, C.; Wang, H.; Lu, N.; Chen, T.; He, H.; Lu, Y.; Tu, X. M., Log-transformation and its implications for data analysis, *Shanghai Arch. Psychiatry*, 2014, 26(2), 105–109.

4 Halabaku, E.; Bytyçi, E., Overfitting in machine learning: A comparative analysis of decision trees and random forests, *Intell. Autom. Soft Comput.*, 2024, 39(6), 987–1006.