Chapter 6 Supervised Learning

**6. 1 Introduction**

Supervised learning is a subdomain of machine learning widely used in predictive modeling tasks where labeled data is available. This method includes training algorithms on a dataset with known outputs to predict outcomes on new, unseen data. In the context of materials science, particularly battery research, supervised learning can play a pivotal role in predicting key material properties such as energy density based on a compound's chemical and physical attributes.

In this study, supervised learning techniques are applied to predict the **energy density (in Wh/kg)** of battery materials using a dataset comprising multiple chemical and structural descriptors. The primary goal is to identify the most effective model for accurate prediction among several supervised learning algorithms, including Linear Regression, Random Forest Regressor, Support Vector Regressor (SVR), and Extreme Gradient Boosting (XGBoost). In this chapter, we discuss the process of how the data was used for training the models and evaluating the results. The code snippets are added as demonstration to the reader.

**6.2 Data Preprocessing**

**6.2.1 Encoding and Scaling**

A black background with white text

AI-generated content may be incorrect.The categorical column "Type" was label-encoded to transform it into a binary numerical format (0 for Anode and 1 for Cathode). Additionally, since Support Vector Machines and other kernel-based methods are sensitive to feature scales, standard scaling was applied to ensure normalized input features during modeling.

**6.2.2 Outlier Detection and Removal**

A computer screen with text

AI-generated content may be incorrect.To enhance model performance and reduce the influence of noise, outliers in the target variable were removed using the interquartile range (IQR) method. Observations falling outside 1.5 times the IQR from the first and third quartiles were excluded.

**6.2.3 Train-Test Split and Shaping**

A computer screen shot of a program code

AI-generated content may be incorrect.The data was split such that 80% was used to train the models and the remaining 20% was used to test the various regression models. This yielded 626 instances for training the models and 157 instances for testing the models. Each algorithm would therefore be trained with 626 instance features to predict 626 labels. The model would then be used on the remaining 1836 instance’s test features to make label (disposition) predictions. These label predictions would be compared to the given test labels to determine how well the model performed.

**6.3. Modeling Approaches**

**6.3.1 Linear Regression**

A screenshot of a computer program

AI-generated content may be incorrect.As an example of how the models were trained, here are some code snippets to understand the process better. It is interpretable and computationally efficient, making it an ideal baseline model. The model was trained using the least squares approach. The evaluation metrics for all the models will be discussed in detail in the model evaluation section.

**6.3.2 Random Forest Regressor**

A computer screen shot of a program

AI-generated content may be incorrect.Random Forest is an ensemble learning method that constructs multiple decision trees and outputs the mean prediction. It is robust against overfitting and effective in capturing nonlinear relationships.

The Random Forest model significantly outperformed Linear Regression, indicating the presence of complex nonlinear dependencies among the features.

**6.3.3 Support Vector Regressor (SVR)**

A screen shot of a computer program

AI-generated content may be incorrect.Support Vector Machines aim to find a function that deviates from the target values by a value less than epsilon. With kernel tricks, SVR can handle high-dimensional, nonlinear problems.

The SVR model performed poorly, it was less effective than Linear Regression as well as Random Forest, possibly due to its sensitivity to parameter settings and data scaling.

**6.3.4 Tuned Support Vector Regressor**

A computer screen with text and images

AI-generated content may be incorrect.A GridSearchCV approach was adopted to optimize the SVR model across multiple hyperparameters (C, epsilon, and kernel). After tuning, performance improved slightly.

Hyperparameter tuning yielded modest improvements, reinforcing the idea that SVR requires meticulous configuration. Using the scaled dataset produced more favorable results, outperforming the Linear Regression model.

**6.3.5 XGBoost Regressor**

A screen shot of a computer code

AI-generated content may be incorrect.XGBoost outperformed both Linear Regression model and SVR model, demonstrating its capacity to capture complex interactions and minimize error. However, the XGBoost underperforms compared to the Random Forest model and the Tuned Support Vector Regressor.

**6.3.6 Tuned XGBoost Regressor**

A screen shot of a computer program

AI-generated content may be incorrect.Further tuning of XGBoost was performed using GridSearchCV over n\_estimators, max\_depth, and learning\_rate. This yielded better results; the results were comparable to Random Forest and the Tuned SVR.

**6.4. Model Evaluation**

**6.4.1 Performance Metrics**

All models were evaluated using common regression metrics: Mean Absolute Error (MAE), Root Mean Square Error (RMSE), and the Coefficient of Determination (R^2).

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Linear Regression | Random Forest Regressor | Support Vector Regressor | Tuned Support Vector Regressor | XGBoost Regressor | Tuned XGBoost Regressor |
| MAE | 349.01 | 133.25 | 493.66 | 171.77 | 155.86 | 209.71 |
| RMSE | 503.83 | 355.38 | 688.79 | 368.69 | 416.32 | 381.51 |
| R² | 0.63 | 0.82 | 0.31 | 0.80 | 0.75 | 0.79 |

Among the regression models evaluated, the Random Forest Regressor demonstrated superior predictive performance across all key metrics. It achieved the lowest Mean Absolute Error (MAE) of 133.25 and Root Mean Squared Error (RMSE) of 355.38, along with the highest coefficient of determination (R²) value of 0.82, indicating strong accuracy and generalizability. In contrast, the Linear Regression model, used as a baseline, yielded a higher MAE of 349.01, RMSE of 503.83, and a lower R² score of 0.63, reflecting a weaker fit and reduced predictive capability. The Support Vector Regressor (SVR) initially exhibited the poorest performance, with the highest MAE (493.66) and RMSE (688.79), and the lowest R² value (0.31). However, after hyperparameter tuning, the performance of SVR improved substantially, achieving an MAE of 171.77, RMSE of 368.69, and an R² of 0.80, thus approaching the performance level of the Random Forest model.

The XGBoost Regressor also showed competitive results, with an MAE of 155.86, RMSE of 416.32, and R² of 0.75. Following hyperparameter tuning, XGBoost demonstrated a modest improvement in RMSE (381.51) and R² (0.79), although the MAE increased to 209.71, suggesting that the tuning process had mixed effects on its overall error distribution. These findings underscore the effectiveness of ensemble tree-based methods, particularly Random Forest, in modeling the target variable with high accuracy. Moreover, the significant improvement observed in the tuned SVR highlights the critical role of model optimization. Overall, Random Forest and the tuned SVR emerged as the most robust models for this regression task, outperforming both linear and other non-linear alternatives.

**6.4.2 Predicted Vs Actual plots**

|  |  |
| --- | --- |
| **Model** |  |
| Linear Regression |  |
| Random Forest Regressor |  |
| Support Vector Regressor |  |
| Tuned Support Vector Regressor |  |
| XGBoost Regressor |  |
| Tuned XGBoost Regressor |  |

The table for Predicted vs Actual plots visually represents how well the model has actually fit the data. It is very clear that the Random Forest model outperforms the other models, in terms minimizing the error between the Predicted and the Actual values of the test set.

**6.5. Feature Importance**

A graph with blue bars

AI-generated content may be incorrect.The trained Machine Learning algorithms were able to measure the relative importance of each feature in the dataset. Due to its performance metric being the best overall, going forward the Random Forest model is considered the best model. Hence, the feature importance resulting from the Random Forest model is further discussed aided by the following figure:

This figure highlighs how much each input variable contributes to the model’s predictive performance. Among the features evaluated, Capacity\_per\_gram\_in\_mAh stands out with a dominant importance score of approximately 0.74, indicating it is the most critical variable in the model. This suggests that the model relies heavily on this feature to make accurate predictions. The next most influential feature is Voltage\_in\_V, which has a significantly lower score of around 0.21, but still contributes meaningfully to the prediction outcome. The remaining features — Efficiency\_in\_percent, Molecular\_weight, and Type — show very low importance scores, each contributing minimally to the model's performance. This implies that while these features are present in the dataset, they offer little additional predictive value compared to Capacity and Voltage. In practical terms, the Random Forest model could perform well even if these low-importance features were removed or deprioritized.

**6.6 Conclusion**

This chapter explored the application of supervised learning techniques to predict the energy density of battery materials. Among the six models tested, the **Random Forest Regressor** (with an R^2 value of 0.82) and **Tuned XGBoost Regressor** (with an R^2 value of 0.82)delivered the most accurate and robust predictions. The success of ensemble methods like Random Forst and XGBoost highlights the presence of complex, nonlinear relationships within the dataset.

The findings of this study reinforce the utility of machine learning in materials science and the predictive modeling of material properties. Future work could involve the integration of domain-specific feature engineering, model interpretability tools such as SHAP values, and expanding the dataset to include experimental uncertainties and real-world deployment constraints.