Battery Capacity Prediction Using Electrochemical Impedance Spectroscopy Data

Amir Babamahmoudi
Computer Science MSc Grad
University of Alberta
amirbabamahmoudi@gmail.com

1 Introduction

This report presents a comprehensive machine learning approach for predicting battery capacity using electrochemical impedance spectroscopy (EIS) measurements. The analysis encompasses data preprocessing, systematic model selection through cross-validation, hyper-parameter optimization via grid search, and rigorous evaluation using both regression and classification metrics. The implementation demonstrates best practices in machine learning workflow including proper train-test splitting, feature standardization, and comprehensive evaluation metrics.

The dataset contains impedance measurements for 467 battery samples, resulting in 138 features per battery. The target variable is the measured capacity in mAh, ranging from approximately 6,522 to 10,027 mAh.

2 Data Analysis and Preprocessing

2.1 Dataset Overview

The dataset comprises:

- **Sample size:** 467 battery samples
- **Features:** 138 impedance measurements
- **Target variable:** Battery capacity (mAh)

Table 1: Dataset Statistics Summary

Statistic	**Min**	**Max**	**Mean**	**Std**
Capacity (mAh)	6522.40	10027.01	8982.47	775.91

2.2 Data Splitting Strategy

A critical aspect of the methodology was maintaining strict separation between training and testing data to prevent data leakage. The dataset was split using:

- **Training set:** 373 samples (80%)
- **Test set:** 94 samples (20%)
- **Random state:** 42 (for reproducibility)

2.3 Feature Preprocessing

The preprocessing pipeline implemented several key steps to ensure robust model performance:

1. **Missing Value Handling:** Any missing values were imputed using median values from the training set only. Although this dataset did not have any missing values, this function was implemented for the future if needed.

2. **Feature Standardization:** StandardScaler was applied to normalize features to zero mean and unit variance.

3. **Fit-Transform Protocol:** The scaler was fitted exclusively on training data and then applied to both training and test sets to prevent data leakage.

The standardization process is crucial for impedance data as measurements can vary significantly across different frequency ranges. The mathematical transformation applied was:

$$z = \frac{x - \mu_{train}}{\sigma_{train}} \tag{1}$$

where μ_{train} and σ_{train} are the mean and standard deviation computed from the training set only.

3 Evaluation Metrics and Rationale

The model evaluation employed both regression and classification metrics to provide comprehensive performance assessment:

3.1 Regression Metrics

Root Mean Square Error (RMSE): Provides interpretable error measurement in the same units as capacity (mAh). RMSE penalizes larger prediction errors more heavily, making it suitable for applications where large capacity prediction errors are particularly problematic.

Mean Absolute Error (MAE): Offers robust error measurement less sensitive to outliers than RMSE. This metric provides intuitive understanding of average prediction accuracy.

R^2 Score: Indicates the proportion of variance in capacity explained by the model. Values closer to 1.0 indicate better model performance, with R^2 $>$ 0.8 generally considered excellent for practical applications.

Mean Squared Error (MSE): Fundamental metric for optimization and mathematical analysis, though less interpretable than RMSE.

3.2 Classification Metrics

Battery capacity was binned into five categories for classification analysis:

Table 2: Battery capacity bins and sample counts

Bin	Capacity Range (mAh)	Number of Samples
1	$\leq$ 7,000	2
2	7,000 – 7,400	17
3	7,400 – 8,000	40
4	8,000 – 8,500	59
5	$>$ 8,500	349

F1 Score: Harmonic mean of precision and recall, providing balanced assessment of classification performance. F1 scores are particularly valuable when class distributions are imbalanced, as they account for both false positives and false negatives.

Cohen's Kappa Score: Measures agreement between predicted and actual classifications while accounting for chance agreement. Kappa scores > 0.7 indicate substantial agreement, making this metric valuable for assessing practical classification reliability.

Accuracy: Overall classification correctness, providing intuitive performance understanding.

4 Model Selection

4.1 Candidate Models

Five different regression algorithms were evaluated to identify the most suitable approach for this impedance-based capacity prediction task:

1. **Linear Regression:** Baseline linear model assuming linear relationships between impedance features and capacity
2. **Ridge Regression:** Linear model with L2 regularization to handle potential multicollinearity in impedance measurements
3. **Lasso Regression:** Linear model with L1 regularization for automatic feature selection
4. **Random Forest:** Ensemble method capable of capturing non-linear relationships and feature interactions
5. **Support Vector Regression (SVR):** Non-parametric method with linear kernel for robust prediction

4.2 Model Characteristics and Rationale

Linear Models: Given that impedance measurements may exhibit linear relationships with capacity degradation, linear models serve as important baselines. Ridge regression addresses the potential multicollinearity among frequency-related features, while Lasso can identify the most relevant frequency components.

Random Forest: This ensemble of decision trees is particularly suitable for impedance data because:

- Handles non-linear relationships between impedance and capacity
- Naturally manages feature interactions across different frequencies
- Provides inherent feature importance rankings
- Robust to outliers in impedance measurements

Support Vector Regression: Offers robust prediction capabilities and can handle the high-dimensional nature of the impedance feature space effectively.

5 Experimental Methodology

5.1 Cross-Validation Strategy

Model selection was performed using 4-fold cross-validation on the training set. This approach was chosen for several reasons:

- **Robust evaluation:** Multiple validation folds provide more reliable performance estimates than single holdout validation
- **Computational efficiency:** 4-fold CV balances statistical reliability with computational cost
- **Data utilization:** Maximizes use of limited training data (373 samples) while maintaining validation rigor

The cross-validation process strictly maintained the train-test separation, with the test set remaining completely unseen during model selection.

5.2 Model Selection Results

Table 3: Cross-Validation Results for Model Selection

Model	**CV Score (R^2)**
Random Forest	0.8387
Lasso Regression	0.7236
Ridge Regression	0.7213
Support Vector Regression	0.4590
Linear Regression	0.3751

Random Forest demonstrated superior performance with a cross-validation R^2 score of 0.8387, significantly outperforming linear models and SVR.

5.3 Hyperparameter Optimization

Following model selection, grid search optimization was performed on the Random Forest model to fine-tune its hyperparameters:

Table 4: Grid Search Parameter Space

Parameter	**Search Space**
n_estimators	[25, 50, 100]
max_depth	[3, 5, 7]
min_samples_split	[2, 4, 6]
min_samples_leaf	[1, 4]

The grid search evaluated 54 parameter combinations using 4-fold cross-validation, resulting in optimal parameters:

Table 5: Optimized Hyperparameters

Parameter	**Optimal Value**
n_estimators	50
max_depth	7
min_samples_split	4
min_samples_leaf	1

5.4 Final Model Training

After hyperparameter optimization, the final model was trained on the complete training set (373 samples) using the optimized parameters. This approach maximizes the use of available training data while maintaining proper validation methodology.

6 Results

6.1 Regression Performance

Table 6: Model Performance on Training and Test Sets

Metric	Training Set	Test Set
RMSE (mAh)	225.99	252.99
MAE (mAh)	156.27	172.92
R^2 Score	0.9173	0.8798
MSE (mAh^2)	51,070.91	64,004.72

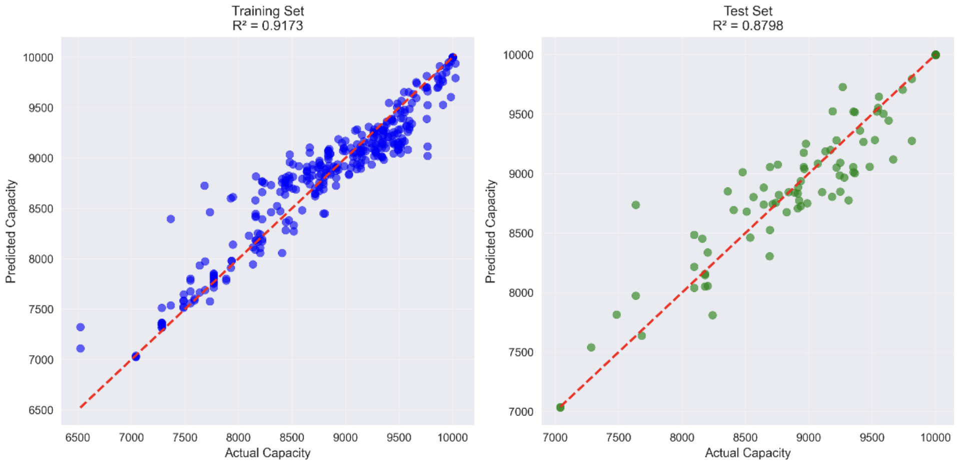

Figure 1: R^2 score of the model on the test and the training set.

The model demonstrates excellent performance, achieving an R^2 score of 0.8798 on the test set, meaning that approximately 88% of the variance in battery capacity is explained by the impedance features. The test RMSE of 252.99 mAh corresponds to about 7.2% of the observed capacity range, reflecting strong practical accuracy for many engineering applications. Moreover, the training set RMSE is 225 mAh (6.42% of the range), which is very close to the test error, indicating minimal overfitting and strong generalization capability.

6.2 Classification Performance

Table 7: Classification Metrics Summary

Metric	Training Set	Test Set
Accuracy	0.8954	0.9149
F1 Score (Macro)	0.6450	0.7942
F1 Score (Weighted)	0.8847	0.9134
Cohen's Kappa	0.7352	0.7529

The classification results demonstrate strong performance with 91.49% accuracy on the test set and a Cohen's Kappa score of 0.7529, indicating substantial agreement beyond chance.

6.3 Per-Bin Classification Analysis

Table 8: Per-Bin Classification Performance (Test Set)

Capacity Bin	Accuracy	F1 Score
Bin 2 (7000-7400)	0.6667	0.8000
Bin 3 (7400-8000)	0.7500	0.6667
Bin 4 (8000-8500)	0.6923	0.7500
Bin 5 (>8500)	0.9730	0.9600

The variation in per-bin performance can be attributed to the uneven distribution of samples across bins. As shown in the dataset statistics, most samples fall into higher-capacity bins, particularly Bin 5, which contains 349 instances and therefore achieves higher accuracy and F1 scores. In contrast, Bin 1 contains only two samples in total, none of which were present in the test set; consequently, no performance metrics are reported for this bin. This absence of performance in certain bins, especially those with very few samples, contributes to the lower macro F1 score observed. Increasing the number of samples in underrepresented bins, particularly Bin 1, would likely improve the macro F1 score and result in a more balanced classification across all categories.

6.4 Model Generalization Assessment

The comparison between training and test performance indicates good generalization:

- R^2 difference: 0.0375 (training: 0.9173, test: 0.8798)
- RMSE difference: 27.00 mAh (training: 225.99, test: 252.99)
- Classification accuracy difference: -0.0195 (test actually outperformed training)

These modest differences suggest the model generalizes well without significant overfitting.

7 Discussion and Conclusion

This study successfully developed a robust machine learning pipeline for battery capacity prediction using electrochemical impedance spectroscopy data, with the Random Forest model achieving excellent performance. The methodology incorporated strict separation of training and test sets with an appropriate preprocessing workflow, preventing data leakage and ensuring reliable performance estimates. Feature standardization ensured that all impedance frequency components contributed appropriately, while cross-validation for model selection followed by grid search optimization aligned with best practices in machine learning workflows.

Comprehensive evaluation using both regression and classification metrics provided multiple perspectives on model performance, making the approach suitable for different practical applications. The Random Forest model excelled due to its ability to capture non-linear relationships between impedance and capacity, its natural handling of feature interactions across different frequencies, and its robustness to potential outliers. Ensemble averaging further reduced prediction variance, improving model stability and generalization.

The results demonstrate the value of impedance spectroscopy data for battery state assessment and provide a solid foundation for practical capacity prediction applications. Key contributions of this work include:

- Systematic evaluation of multiple regression algorithms
- Implementation of proper data splitting, preprocessing, and scaling protocols
- Comprehensive evaluation using both regression and classification metrics
- Demonstration of good model generalization without overfitting

Technical Implementation

The complete analysis was implemented in Python using scikit-learn, pandas, and numpy. The modular code structure includes separate classes for data loading, preprocessing, model training, and evaluation, promoting reproducibility and maintainability. All models and results are saved for future reference and deployment.