Regression & Optimization Documentation and report

Oscar Boix Ripolles
R00257119@mymtu.ie

Content

Oscar Boix Ripolles

Overview

The provided code is a Python script for machine learning

Libraries Used

- “numpy”: Used for array operations.
- “pandas”: Used for data manipulation.
- “matplotlib.pyplot”: Provides graphical plotting tools.
- “sklearn”
    - “model_selection”: Provides tools for model selection.

Functions

input_data()

- Loads data from "energy_performance.csv" file and splits it into features and targets.

- Outputs feature data and target values.

parameter_vector_size(degree)

- Calculates the size of the polynomial parameter vector based on the given degree.

- Returns the number of parameters in the polynomial.

calculate_model_function(degree, data, param_vector)

- Calculates the model function using the polynomial and given parameter vector.

- Returns the calculated model values.

linearize(degree, data, p0)

- Linearizes the model function by estimating partial derivatives.

- Returns the initial model values and the Jacobian matrix.

calculate_update(target, f0, jac)

- Calculates the parameter vector update using the normal equation.

- Returns the parameter vector update.

regression(degree, data, target)

- Performs regression to determine the optimal parameter vector.

- Returns the optimized parameter vector.

k_fold_cross_validation(data, target, degree)

- Performs k-fold cross-validation to assess model accuracy.

- Returns the average accuracy score.

plot(target, predicted_target)

- Plots the predicted targets against the true targets.

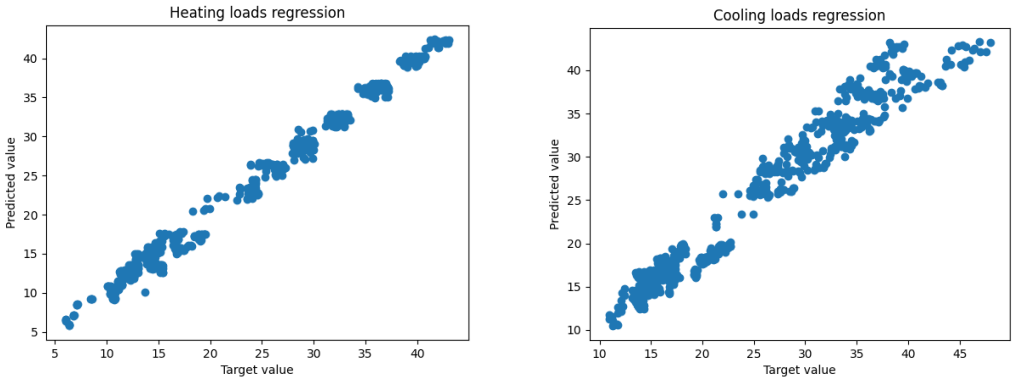
main()

The main function of the script.

Usage

- Execute the main() function to run the entire workflow.
- The program will run the cross validation and find the best polynomial degree for the model.
- The program will run with the selected degree and all data and will plot the results.
- Review the plots generated to visualize the model performance.

Results



After the cross validation, the automatically selected degree is 2 for both heating loads and cooling loads due to the less difference between the predictions calculated and the target along all cross-validation.

Finally, the program uses all data to calculate the parameters of a 2-degree polynomial, first for heating loads and then for cooling loads, and then plots predicted values vs target values because the data is 8-dimensional and we cannot plot that data in a visible and comprehensive way.

In these graphs above, the closer to the diagonal the best the parameters has predicted the real values. As we can see in this case, the heating loads have been predicted better than the cooling loads as data is more grouped along the diagonal. The cooling loads prediction is also good as more or less the data is found in the diagonal but both polynomials could improve with higher degrees.