A2: Regression Model Selection in Predicting Wear Rate of Mechanical Component

Part 1: Polynomial Models

Data Preprocessing

The first part of this homework involves building polynomial regression models of different orders (ranging from 1 to 6) to predict the wear rate of a mechanical component based on three features: RPM (Revolutions per minute), Load (Applied load in Newtons), and Hardness (Hardness value of the component in HV).

Before diving into the modeling process, I imported the necessary libraries, loaded the training dataset from a CSV file, and specified the features (X) and the target variable (y).

Polynomial Regression

Feature Engineering

For each polynomial order within the specified range, I created polynomial features based on RPM, Load, and Hardness using the **PolynomialFeatures** transformer from scikit-learn. These polynomial features allow us to capture potential nonlinear relationships between the predictors and the target variable.

Cross-Validation

I performed 5-fold cross-validation for each polynomial order. Cross-validation is a technique used to assess the predictive performance of a model. In this context, it involves splitting the dataset into five subsets (folds), training the model on four of these subsets and testing it on the fifth fold. This process is repeated five times, each time with a different fold serving as the test set.

RMSE Calculation

In each fold of the cross-validation, I calculated the Root Mean Squared Error (RMSE) as the evaluation metric. RMSE measures the average magnitude of the errors (differences between predicted and actual values) and is calculated as the square root of the mean of the squared errors. Specifically, for each fold, I calculated the negative mean squared error (neg_mean_squared_error) using scikit-learn's **cross_val_score** function and then took the square root of its absolute value to obtain the RMSE.

Model Selection

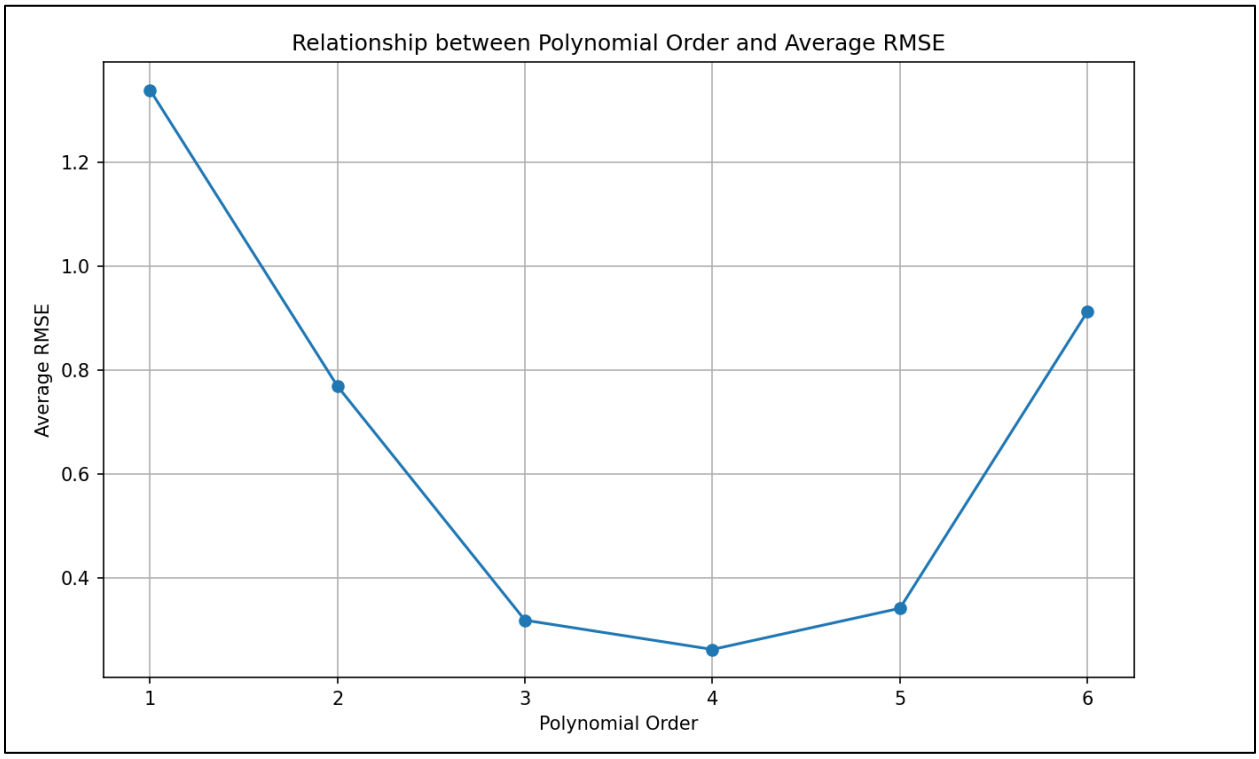
I recorded the average RMSE for each polynomial order across the five folds. The polynomial order that resulted in the lowest average RMSE was chosen as the best model order. In our case, the best polynomial order was determined to be 4.

Model Retraining and Saving

To ensure the best model's performance, I retrained the model using the entire training dataset with the selected polynomial order. The retrained model was saved using Python's joblib library.

Graphical Representation

I visualized the relationship between polynomial order and average RMSE by creating a plot. The x-axis represents the polynomial order, and the y-axis represents the average RMSE. This plot helps us visualize how the RMSE changes with the complexity (polynomial order) of the model.



Cross-Validation Principle

5-fold cross-validation is a widely used technique in machine learning for model evaluation and selection. Its principle can be summarized as follows:

1. The dataset is randomly divided into five subsets or "folds."
2. The model is trained on four of these folds and evaluated on the remaining fold. This process is repeated five times, with each fold serving as the test set exactly once.
3. At the end of the process, five evaluation scores are obtained (in this case, RMSE values).
4. The average of these scores is calculated to assess the model's performance.

Cross-validation helps to improve the generalizability of the models by providing a more robust estimate of their performance. It ensures that the model is evaluated on different subsets of the data, which helps identify any overfitting issues. By averaging the evaluation scores, it provides a more stable and reliable estimate of the model's predictive performance on unseen data.

Part 2: Ridge-regularized Polynomial Models

Ridge Regularization

In the second part of the homework, I extended the polynomial regression models from Part 1 by applying Ridge regularization. Ridge regularization is a technique used to prevent overfitting in regression models by adding a penalty term to the loss function. This penalty term is controlled by a hyperparameter called "alpha."

Model Selection with Ridge Regularization

I followed a similar process as in Part 1 for generating polynomial features based on RPM, Load, and Hardness.

Standardization

Before applying Ridge regularization, I standardized the polynomial features using the **StandardScaler** from scikit-learn. Standardization ensures that all features have a mean of 0 and a standard deviation of 1, which is a common practice when using Ridge regularization.

Cross-Validation with Ridge Regression

For each combination of polynomial order (ranging from 1 to 6) and penalty coefficient (alpha), I performed 5-fold cross-validation using Ridge regression. In this case, RMSE was again used as the evaluation metric. I aimed to find the combination of polynomial order and alpha that resulted in the lowest average RMSE.

Model Retraining and Saving

Similar to Part 1, after determining the best combination of polynomial order and penalty coefficient, I retrained the Ridge-regularized model using the entire training dataset. The retrained model was saved using joblib.

Comparison with Part 1

Interestingly, in Part 2, the combination of polynomial order and penalty coefficient that resulted in the lowest average RMSE was found to be the same as in Part 1, which was a polynomial order of 4. This suggests that the Ridge regularization did not significantly alter the model's complexity compared to the non-regularized polynomial model.

Conclusion

In this homework, I built and evaluated polynomial regression models to predict the wear rate of a mechanical component based on RPM, Load, and Hardness features. I also extended these models by applying Ridge regularization.

The principle of 5-fold cross-validation was discussed, highlighting its importance in assessing model performance and improving generalizability.

Ultimately, the selected best-performing model was a polynomial regression model of order 4, which achieved the lowest average RMSE. This model was retrained and saved for further evaluation.

The Ridge-regularized model with the same polynomial order did not result in a significantly different model, suggesting that the non-regularized model was already adequately fitting the data. Ridge

regularization is often more beneficial when dealing with highly complex models that are prone to overfitting.

The saved models are ready for further testing on the test dataset, which is held by the instructor to evaluate the models' generalizability and predictive accuracy.