Advanced Data Analysis Project

Spectral Dataset

Week 3

LUT University

Alejandro Lopez Vargas
Sina Saeedi
Pau Ventura Rodríguez

Visit our GitHub to check the script.

Modelling Goal

The goal of the project is to compare the performance of various regression techniques, specifically Multiple Linear Regression (MLR), Principal Component Regression (PCR), Partial Least Squares (PLS), and k-PLS (Kernel Partial Least Squares) for predicting some plant traits using Spectral dataset that includes wavelength variables.

Let's discuss each of these techniques and modeling approaches:

1. Multiple Linear Regression (MLR): MLR is a traditional regression technique that aims to establish a linear relationship between the predictor variables (in this case, wavelength variables) and the target variable.
2. Principal Component Regression (PCR): PCR combines Principal Component Analysis (PCA) with MLR. First, we perform PCA on the wavelength variables to reduce their dimensionality while retaining most of the information. Then, apply MLR to the reduced set of Principal Components (PCs) instead of the original wavelengths. The number of PCs we decide to retain can impact the model's performance.
3. Partial Least Squares (PLS): PLS is a regression technique that aims to find latent variables (also known as factors) that explain the maximum variance in both the predictor and target variables. By selecting an appropriate number of latent variables, not only we build a predictive model but also achieve dimensionality reduction.
4. Kernel Partial Least Squares (k-PLS): k-PLS is an extension of PLS that employs kernel methods for non-linear regression. Similar to PLS, it aims to find latent variables that capture the relationships between predictor variables (wavelengths) and the target variable. It can be particularly useful when the relationship between variables is non-linear.

Model Calibration, Validation, and Testing Strategy

The strategy that we are going to use for this task, contains the following parts:

1- Data Preparation:

Data Cleaning: This involves handling missing values, outliers, and any data inconsistencies. For that purpose, we can use data-cleaning libraries in Python (e.g., Pandas)

Data Normalization: Ensure that wavelength variables are on a consistent scale. StandardScaler could be used in this part.

2- Data Splitting:

We will divide our dataset into three parts Calibration, Validation, and Test.

The calibration data is used for training and calibrating the models.

The validation data is reserved for assessing model performance during the calibration phase. We will set aside a separate test dataset, which is not used during model calibration or validation, for the final evaluation of model performance.

Usually, 70-80% of the data is used in the calibration phase, 10-15% in validation, and 10-15% of the data is reserved for testing. Since we created one dataset for each trait, we will have 20 train, validation, and test datasets in total.

3- Model Calibration:

For this part, we will train our 4 models (MLR, PCA, PLS, and K-PLS) based on the calibration data. We can use the Scikit-Learn library in Python for the training or just simply write the formulas and do the calculations for each part. Selecting the appropriate number of principal components and latent variables for PCA and PLS is also a part of model calibration. For further examination, we can recalibrate our models by using only the most important wavelength variables. We will then have to train each model 20 times, one for each trait. Note that some models such as PLS can predict multiple target variables, which would be our case but as we have that many null values on targets, we won't be using that feature of the model.

4- Model Validation:

Model validation is crucial to ensure the generalization of our models. We can perform k-fold cross-validation on the validation data to assess how well the models perform for the unseen data.

We will evaluate the performance of each model (MLR, PCR, PLS, k-PLS) on the validation dataset for all 20 traits by using appropriate regression evaluation metrics such as Mean Squared Error (MSE), Root Mean Squared Error (RMSE), R-squared (R^2), or others to compare model performance.

5- Model Testing:

For model testing, we will employ separate data that was not used in the calibration and validation phase. The test data serves as a real-world evaluation of our models. During the model testing phase, we will assess the performance of our calibrated models using various regression metrics such as:

- Mean Squared Error (MSE): measures the average squared difference between the predicted values and the actual values giving us the overall accuracy of the model.
- Root Mean Squared Error (RMSE): is easier to interpret since it is in the same units as the data.
- R-squared (R^2): represents the proportion of variance in the target variable that is explained by the model and provides an overall measure of model performance and fitness.
- Mean Absolute Error (MAE): the average absolute difference between predicted and actual values. It is less sensitive to outliers.

- Plotting residuals: this will help us check how our model predicts the target variables, and if it predicts better on a certain range of values.

6- Comparison and Interpretation:

Compare the results across the four modeling techniques for each of the 20 traits. Analyze which technique performs better in terms of prediction accuracy and robustness, using the previous metrics among others.
Also, we need to consider factors like model complexity, interpretability, and computational efficiency when interpreting the results.

The Mathematical Methods

1- Partial Least Squares (PLS):

PLS aims to maximize the covariance between the predictor variables (X) and the response variable (Y). For that purpose, we try to find the X score vector (T) and the Y score vector (U) after the decomposition and maximize the covariance between T and U. After the decomposition, we can calculate the predictor weight (W) and response weight (U) for calculating T and U.

$$X = TP^T + E \ , \ Y = UQ^T + F$$

$$T = X.W \ , \ U = Y.V$$

$$Maximizing\ Cov(T,\ U)$$

2- Principal Component Regression (PCR):

PCR combines Principal Component Analysis (PCA) with linear regression. First, PCA is applied to the predictor variables (X) to obtain principal components (PCs), and then linear regression is performed on these PCs.

$$Principal\ Components\ (PCs)\text{:}\ X' = XU,\ where\ U\ is\ the\ matrix\ of\ loadings.$$

$$Linear\ Regression\text{:}\ Y = b_0 + b_1PC_1 + b_2PC_2 + \ldots + b_n * PC_n$$

3- Kernel Partial Least Squares (K-PLS):

K-PLS extends PLS to handle nonlinear relationships by mapping the data into a higher-dimensional feature space using a kernel function (e.g., radial basis function kernel).

$$T = \Phi(X)W \quad , \quad U = YV$$

Φ(X) represents the kernel-transformed predictor matrix and we aim to maximize Cov(T, U) in the kernel feature space.

4- Multiple Linear Regression (MLR):

MLR models the linear relationship between the response variable (Y) and multiple predictor variables (X1, X2, ..., Xn).

$$Y = b_0 + b_1X_1 + b_2X_2 + \dots + b_n * X_n$$

Where Y is the response variable, X1, X2, ..., Xn are predictor variables, b0 is the intercept, and b1, b2, ..., bn are the coefficients.

These formulas provide a high-level understanding of the mathematical foundations of each modeling technique. In practice, the specific calculations may involve more steps and considerations, such as data preprocessing, regularization, and error minimization techniques.

Modeling Roles

For this project we will perform all steps (except modeling) together to make sure that all the concepts are learned by everyone. This might be a little bit more time consuming, but definitely worth it in order to get the learning outcomes. For the modeling part, Alejandro will build the MLR model, Sina will work on the PLS and Pau will work on the PCR. The K-PLS will be done by the three of us as it looks like the toughest model (or at least the one that we have worked less on).

Operations flowchart

The workflow can be found in a different document called “Ada_flowchart”.