Recursive Feature Elimination

Week 5 – Advanced Topic 1

What is Recursive Feature Elimination (RFE)?

- RFE is a technique used for selecting the most important features in a dataset.
- RFE selects features by progressively narrowing down the feature set. It is using an external estimator that assigns weights to the features (such as the coefficients in a linear model) and it recursively removes the least important features of the dataset until a desired number of features is reached or an optimal performance is achieved[1].

Key points

- RFE is a **wrapper** method, i.e., it is a **model-based** feature selection method.
- The features are eliminated **recursively**, ensuring that the feature subset is refined over multiple iterations. In this manner, not too many important features are removed quickly from the model.

Use cases of RFE

Gene Selection for Cancer Classification using Support Vector Machines[2]

- The authors wanted to address overfitting which in their case arises from the large number *n* of features (thousands of genes) while the number *l* of training patterns is small (a few dozen patients).
- They built a classifier using Support Vector Machines (SVMs) based on RFE.

Use cases of RFE

Algorithm SVM-RFE:

Inputs:

Training examples

$$\mathbf{X}_0 = [\mathbf{x}_1, \mathbf{x}_2, \ldots \mathbf{x}_k, \ldots \mathbf{x}_\ell]^T$$

Class labels

$$\mathbf{y} = [y_1, y_2, \ldots y_k, \ldots y_\ell]^T$$

Initialize:

Subset of surviving features

$$\mathbf{s} = [1, 2, \ldots n]$$

Feature ranked list

$$r = [\,]$$

Repeat until $\mathbf{s} = [\,]$

Restrict training examples to good feature indices

$$X = X_0(:, \mathbf{s})$$

Train the classifier

$$\alpha = \textit{SVM-train}(X, \mathbf{y})$$

Compute the weight vector of dimension length($\mathbf{s}$)

$$\mathbf{w} = \sum_k \alpha_k \, y_k \, \mathbf{x}_k$$

Compute the ranking criteria

$$c_i = (w_i)^2, \quad \text{for all i}$$

Find the feature with smallest ranking criterion

$$f = \textit{argmin}(\mathbf{c})$$

Update feature ranked list

$$\mathbf{r} = [\mathbf{s}(f), \mathbf{r}]$$

Eliminate the feature with smallest ranking criterion

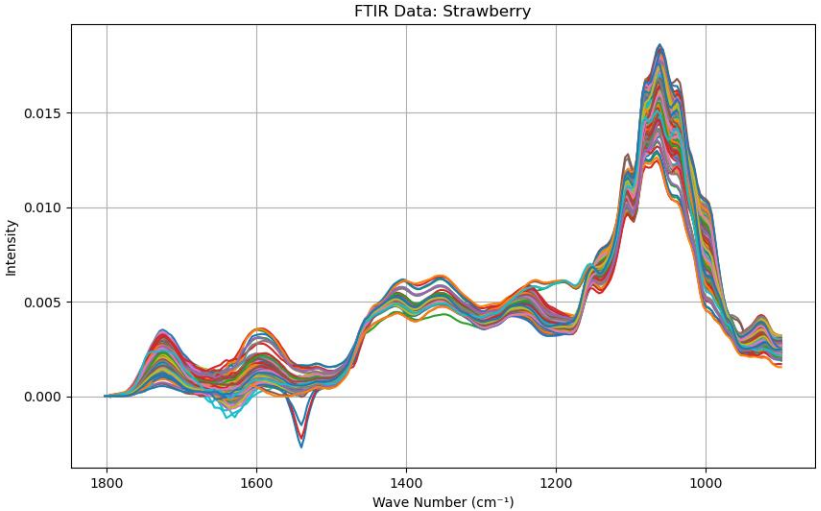
$$\mathbf{s} = \mathbf{s}(1{:}f-1, f+1{:}\text{length}(\mathbf{s}))$$

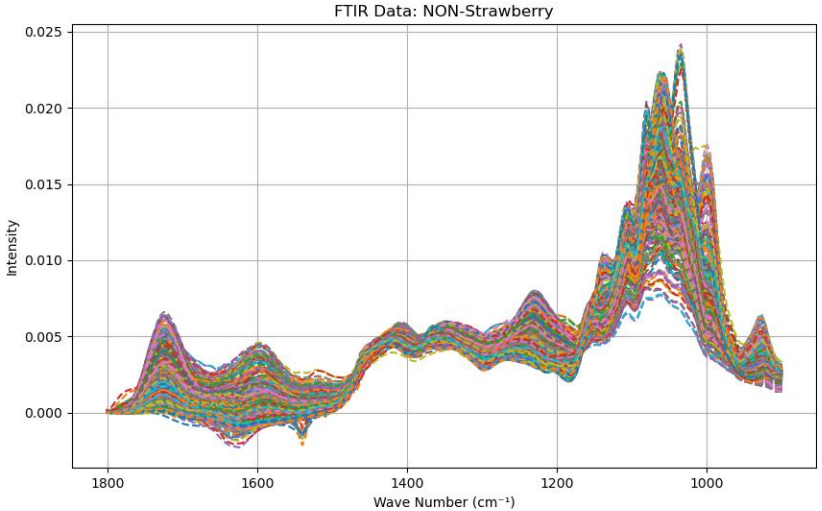
Output:

Feature ranked list $\mathbf{r}$.

Use cases of RFE

Example with FTIR data of fruit pulp

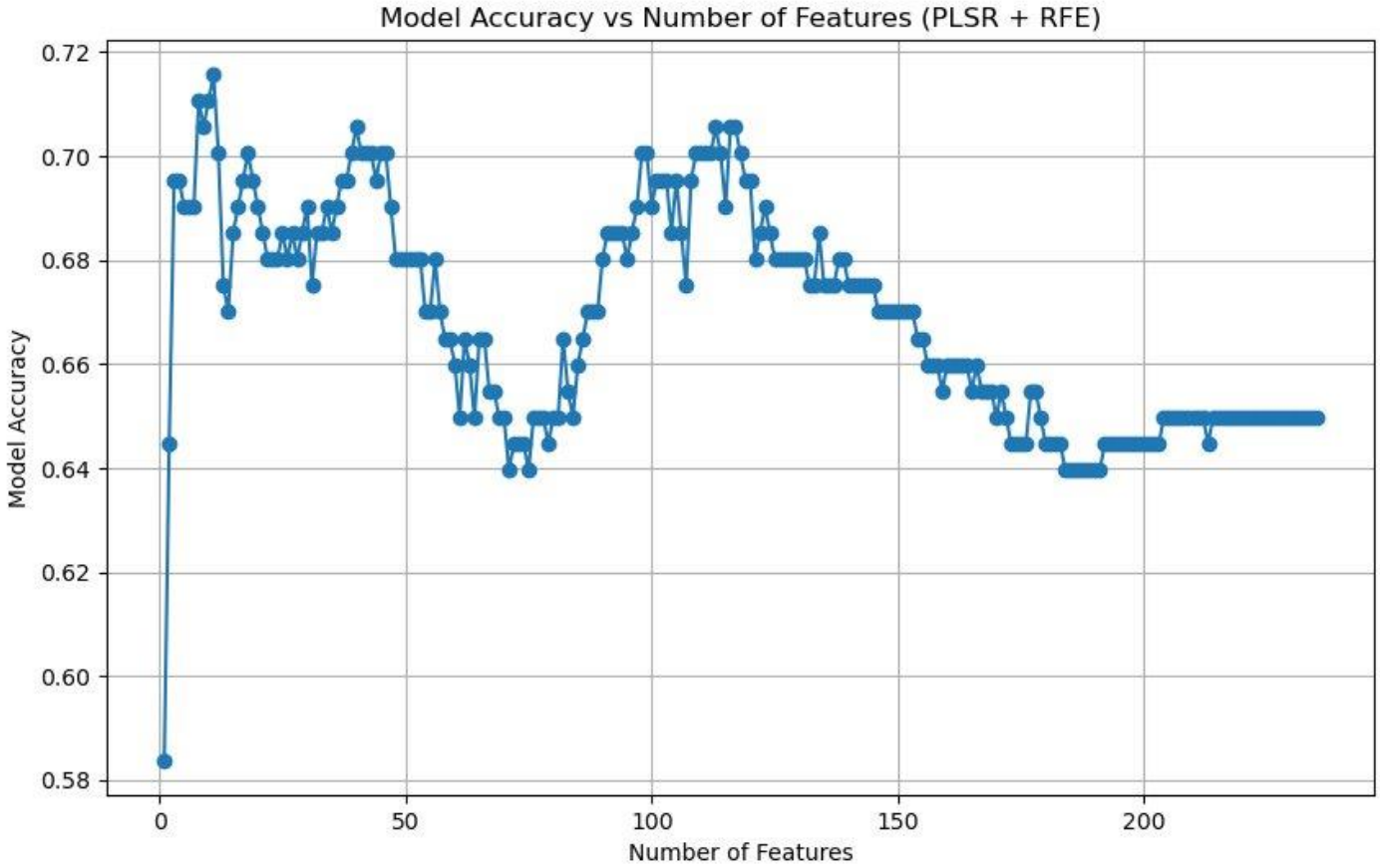




Use cases of RFE

Example with FTIR data of fruit pulp

```
37
38  # Step 3: Recursive Feature Elimination (RFE)
39
40  # Initialize a PLSRegression model
41  model = PLSRegression(n_components=1)  # We can adjust the number of components based on data
42
43  # Perform RFE to recursively eliminate features
44  rfe = RFE(model, n_features_to_select=10, step=1)  # Step-wise elimination
45  rfe.fit(X_train, y_train)
46
47  # Get the ranking of the features
48  ranking = rfe.ranking_
49
```


Model Accuracy vs Number of Features (PLSR + RFE)
Model Accuracy
Number of Features
0.72
0.70
0.68
0.66
0.64
0.62
0.60
0.58
0
50
100
150
200

NTNU

Pros

- efficient for **small** datasets
- handles the features that are highly **correlated** by **re-adjustments** in importance after a feature(s) is removed
- **speeding** up classification algorithms and **enhancing** model **comprehensibility**
- can be adapted to various types of data and classifiers, making it versatile for **different applications**

Cons

- **high computational** requirements due to excessive features in high-dimensional datasets
- multiple **iterations** can lead to **longer** processing **times** compared to simpler methods
- the performance is highly **dependent** on the choice of the underlying **classifier** which can complicate the interpretation of feature importance
- RFE does **not** always pick up on **multicollinearity** within the variables.

Summary

A powerful tool to increases model **performance**

- Reduces **dimensionality** by iteratively removing the least important features, based on an underlying **model's ability** to rank feature importance
- Identify most/least important features (**optimal**)
- **Expensive** for complex datasets

References

[1] Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, Blondel M, Prettenhofer P, Weiss R, Dubourg V, Vanderplas J. Scikit-learn: Machine learning in Python. the Journal of machine Learning research. 2011 Nov 1;12:2825-30.

[2] Guyon I, Weston J, Barnhill S, Vapnik V. Gene selection for cancer classification using support vector machines. Machine learning. 2002 Jan;46:389-422.