2nd RESTART Tech Camp on 5G and Open RAN

AI in O-RAN: Use cases, good practices and hands-on

Salvatore D’Oro
Institute for the Wireless Internet of Things
Northeastern University
s.doro@northeastern.edu

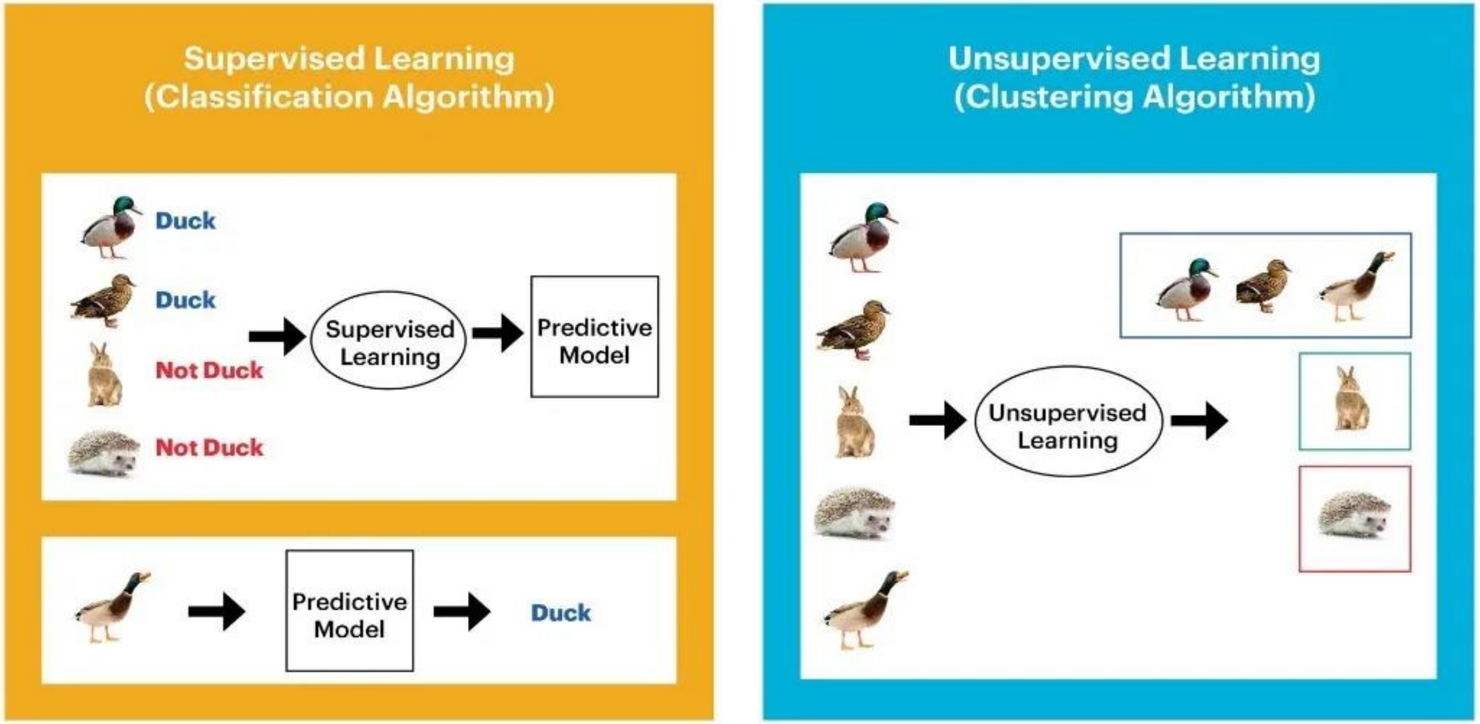
Andrea Pimpinella
University of Bergamo
andrea.pimpinella@unibg.it

Supervised vs Un-Supervised

Learning Paradigms

$$\mathcal{F} : \mathbf{x} \in \mathcal{X} \subseteq \mathbb{R}^n \rightarrow \mathbf{y} \in \mathcal{Y} \subseteq \mathbb{R}^m$$

- **Supervised**: data labels are input of the learning process
- **Unsupervised**: data labels are not known in advance



[1]

Supervised Learning

In a Nutshell

Key: data labels (i.e., ground truth) are input of learning process

- Input: $\boldsymbol{X}$ = [**f**^{1}, **f**^{2}, …**f**^{N}], where **f**^{x} is a feature (column) vector of M entries ($\boldsymbol{X}$ is a NxM matrix)
- The i-th row of $\boldsymbol{X}$ (**x**$_i$) is a vector of N entries: it is called **sample** or **observation** **(in our case, it represents the i-th user)**
- Output: **y** = [y_1, y_2, …, y_M] is a vector of M entries, where y_i is the **label** of the i-th observation
- Goal: Given a **new** observation **x**$_k$, predict the corresponding label $\hat{y}_k$

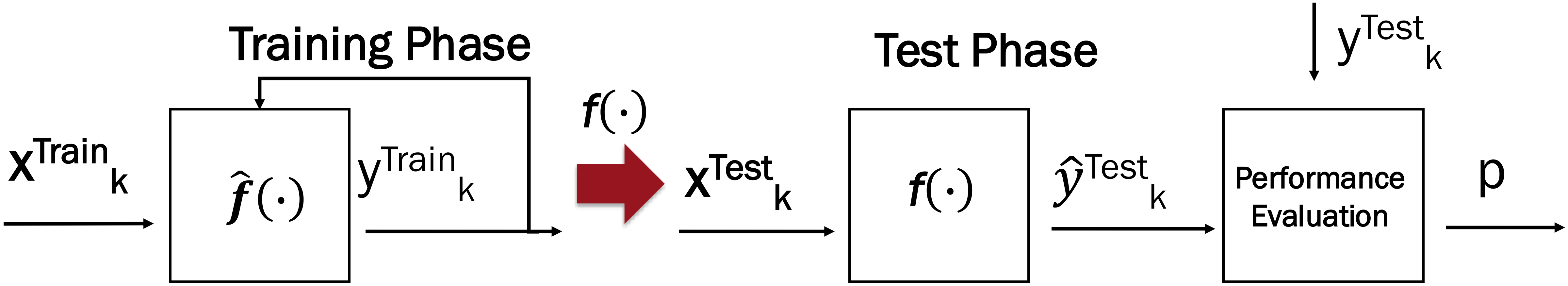
$$\xrightarrow{\mathbf{x}_k} \quad \text{ML Algorithm} \quad \xrightarrow{\hat{y}_k}$$

Supervised Learning (cd.)

In a Nutshell

Key: data labels (i.e., ground truth) are input of learning process

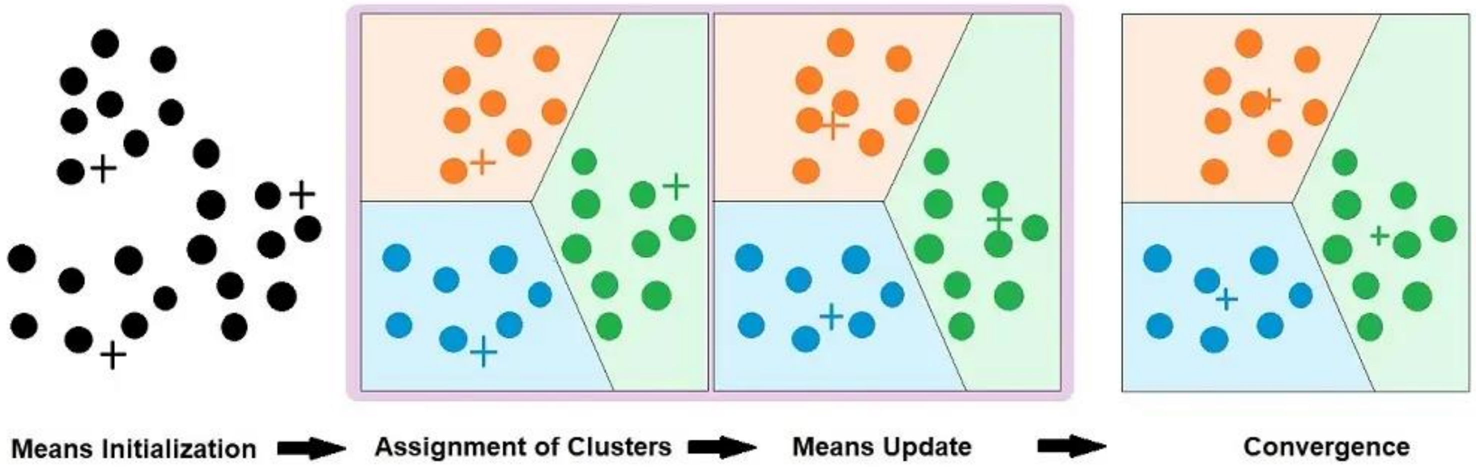
- To do so, the ML Algorithm has first to be **Trained**, i.e., it needs to learn the relationship between the Input and the Output
- Therefore, $\boldsymbol{X}$ and **y** are split in: ($\boldsymbol{X}^{Train}$, **y**$^{\text{Train}}$) and ($\boldsymbol{X}^{Test}$, **y**$^{\text{Test}}$)
- ($\boldsymbol{X}^{Train}$, **y**$^{\text{Train}}$) is used to train the Classifier (tune hyper-parameters, choose features "wheight" in the model, etc.)
- ($\boldsymbol{X}^{Test}$, **y**$^{\text{Test}}$) is used to evaluate the performance **p** (**never** used during Training)



Un-Supervised Learning

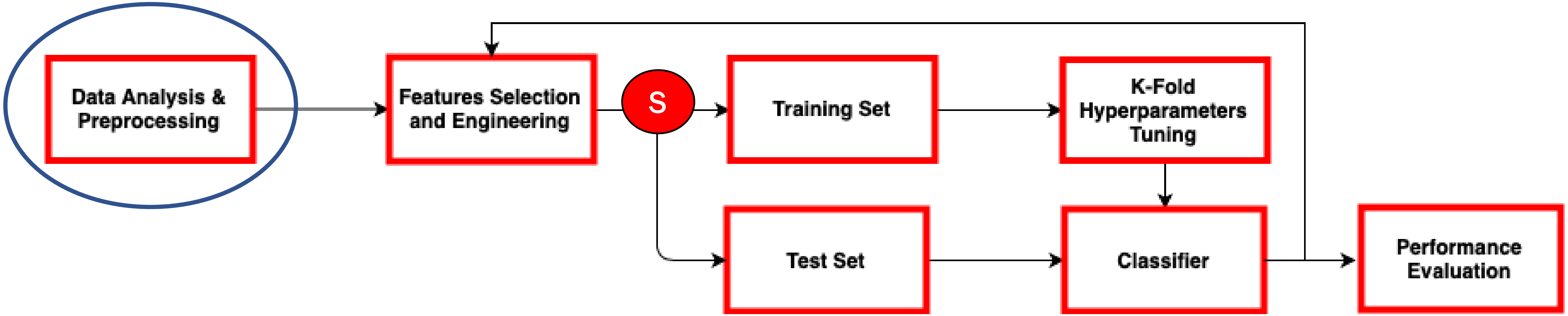
K-Means in a Nutshell

Key: data labels (i.e., ground truth) are not known in advance



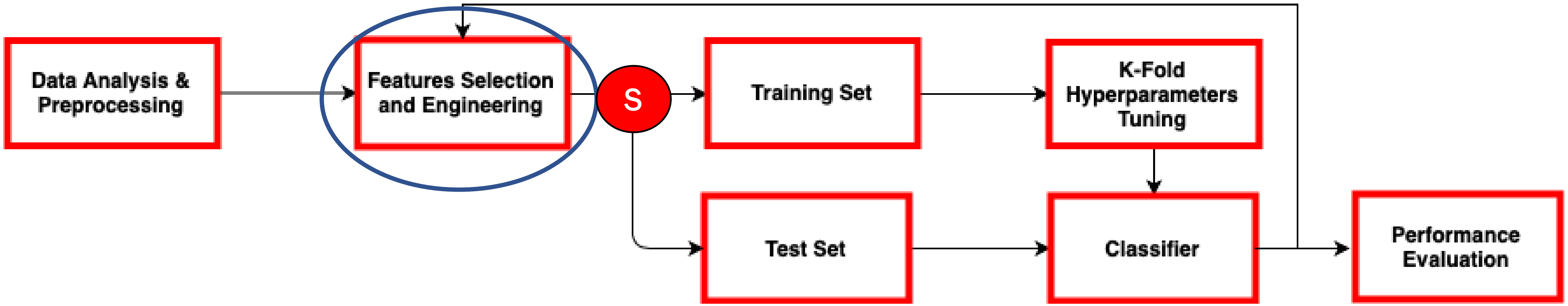
- **Goal of clustering**: group data points that are similar to each others according to input features
- K-means algorithm works **iteratively** to reach convergence:
  1. **Random initialize** k data points as clusters' **centroids**
  2. **Group** each other data point with its **closest** centoid
  3. **Update** each centroid positions as the average of all the data points in the same cluster
  4. Go to 2 and **repeat** until centroids' positions do not change *significantly*

A Traditional Learning Pipeline



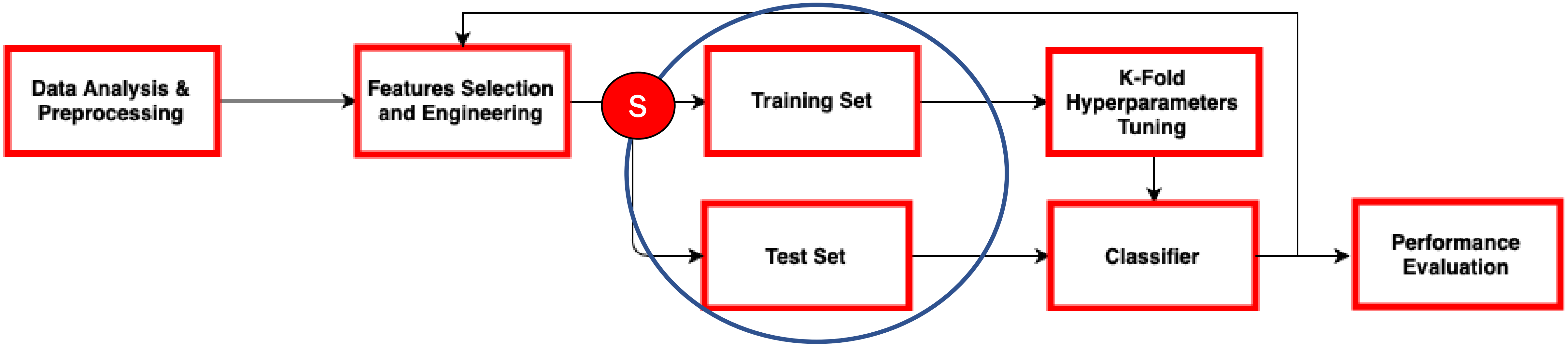
- **KPI Analysis:**
  - Data Visualization: How are fetures distributed?
  - Are features correlated with Slice Ids (i.e, ground truth) ?
  - Does Slice ID *condition* on data distributions?

A Traditional Learning Pipeline



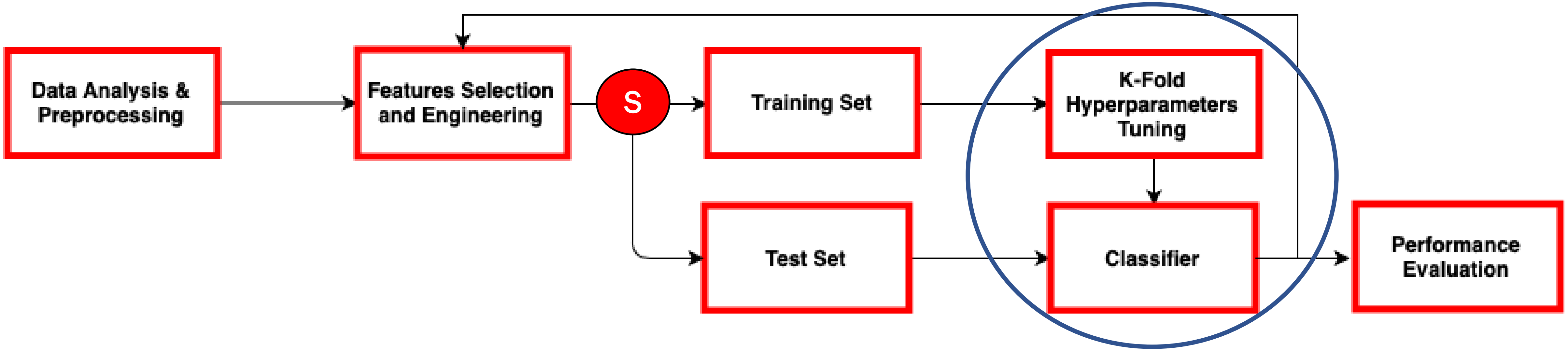
- **Features Engineering**
  - Are all the features needed for an accurate model? → *Domain Experts Knowledge* is required here!
  - Usually, ML classifiers prefer in input Gaussian distributed features → would any data transformation benefit performance?

A Traditional Learning Pipeline



- Once features are ready, you can proceed with the training phase. The data can be **split** as it follows:
  - **80%** of the dataset is retained for **Training** (and also to perform the analysis of previous steps)
  - **20%** of the dataset is used **only** for **Testing** (i.e., performance evaluation)

A Traditional Learning Pipeline

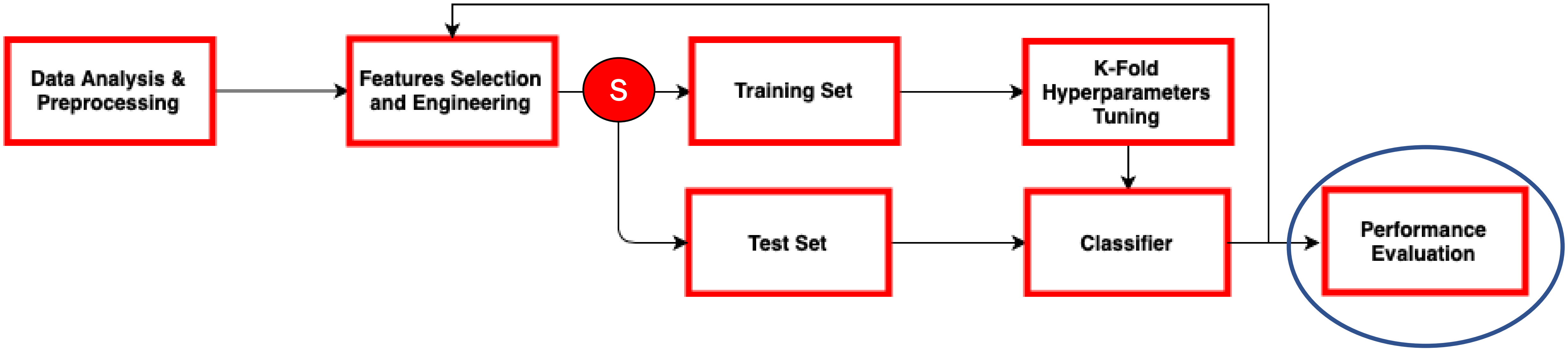


- **Supervised Learning**: Regularised Logistic Regression, Decision Trees, Random Forest, Neural Networks, etc.
- **Unsupervised Learning**: K-means, DB-SCAN, Neural Networks, etc.

K-Fold Cross Validation

1. Assume you choose **Classifier A**, which requires the tuning of Hyper-Parameter $\boldsymbol{\alpha}$
2. Assume also that $\boldsymbol{\alpha}$ can take only positive integer values
3. Select a set of candidate values for $\boldsymbol{\alpha}$, e.g., [1,5, 10, 20, 50]
4. Split your **Training Dataset** in two parts, namely, **SubTraining Set** and **Validation Set**
5. **For each candidate value of $\boldsymbol{\alpha}$**, train your classifier and evaluate prediction performance on the Validation Set
6. Select the value $\boldsymbol{\alpha}_{\text{best}}$ which **maximises** the prediction performance on the Validation Set
7. Retrain the classifier on the whole Training Set with $\boldsymbol{\alpha} = \boldsymbol{\alpha}_{\text{best}}$ and evaluate final prediction performance on the **Test Set**

A Traditional Learning Pipeline

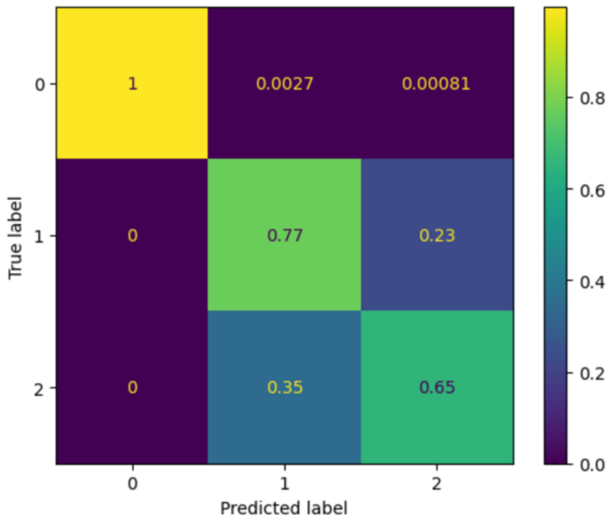


- There are several ways to express prediction performance, depending on the task (supervised vs unsupervised learning) and on the problem at hand (e.g., balancedness of data labels in supervised problems)

Performance Evaluation

Supervised vs Unsupervised Learning

Supervised



- ❖ **Confusion Matrix**
- ❖ **F1-Score[1]**
- ❖ **Accuracy**

Unsupervised

- ❖ **Silhouette Score**
- ❖ **Completeness[2]:** A clustering result satisfies completeness if all the data points that are members of a given class are elements of the same cluster.
- ❖ **Homogeneity[3]:** A clustering result satisfies homogeneity if all of its clusters contain only data points which are members of a single class.

[1] https://scikit-learn.org/stable/modules/generated/sklearn.metrics.f1_score.html

[2] https://scikit-learn.org/stable/modules/generated/sklearn.metrics.completeness_score.html

[3] https://scikit-learn.org/stable/modules/generated/sklearn.metrics.homogeneity_score.html#sklearn.metrics.homogeneity_score

And now... hands-on!

Objectives

- Implement the Learning Pipeline to:
    i. Provide a qualitative/descriptive analysis of the data
    ii. Optimise as much as you can the selected algortithm(s) according K-Fold Cross Validation
    iii. Compare performance of Supervised vs Unsupervised approaches

➢ **Task**:

1. use ***Train_Predict*** code to perform data preprocessing, analysis, training and prediction. Select the algorithms setup that maximize your performance for Supervised and Unsupervised approaches
2. use ***Test_Performance_Fake*** code to test the performance of your final, best algorithms (supervised/unsupervised) on *dataset_restart_testing_fake.pkl* toy dataset

And now... hands-on!

Project Tools

python™

+

scikit learn & matplotlib

Google Colab: https://colab.research.google.com/

Pycharm: https://www.jetbrains.com/pycharm/

Intro to Google CoLab and Jupyter Notebooks: https://www.youtube.com/watch?v=inN8seMm7UI

ScikitLearn Library: https://scikit-learn.org/stable/

And now... hands-on!

Project Tools

Data Repository:

https://github.com/wineslab/restart_assignment_repo

HAVE FUN!