Inferential statistics

Applied Data Analysis (ADA) - October 2025

Nomades Advanced Technologies

Gaspard Villa

❖ <u>Monday :</u> Metrics and training process
  ‣ Supervised & Unsupervised learning
  ‣ Evaluation metrics
  ‣ Training process

❖ <u>Tuesday :</u> Linear models

❖ <u>Wednesday :</u> Support Vector Machine models

❖ <u>Thursday :</u> Decision trees

❖ <u>Friday :</u> Time series analysis

Inferential statistics

Definition : Inferential statistics is the idea of <u>drawing conclusions</u> about a population base on <u>data from a sample</u>.
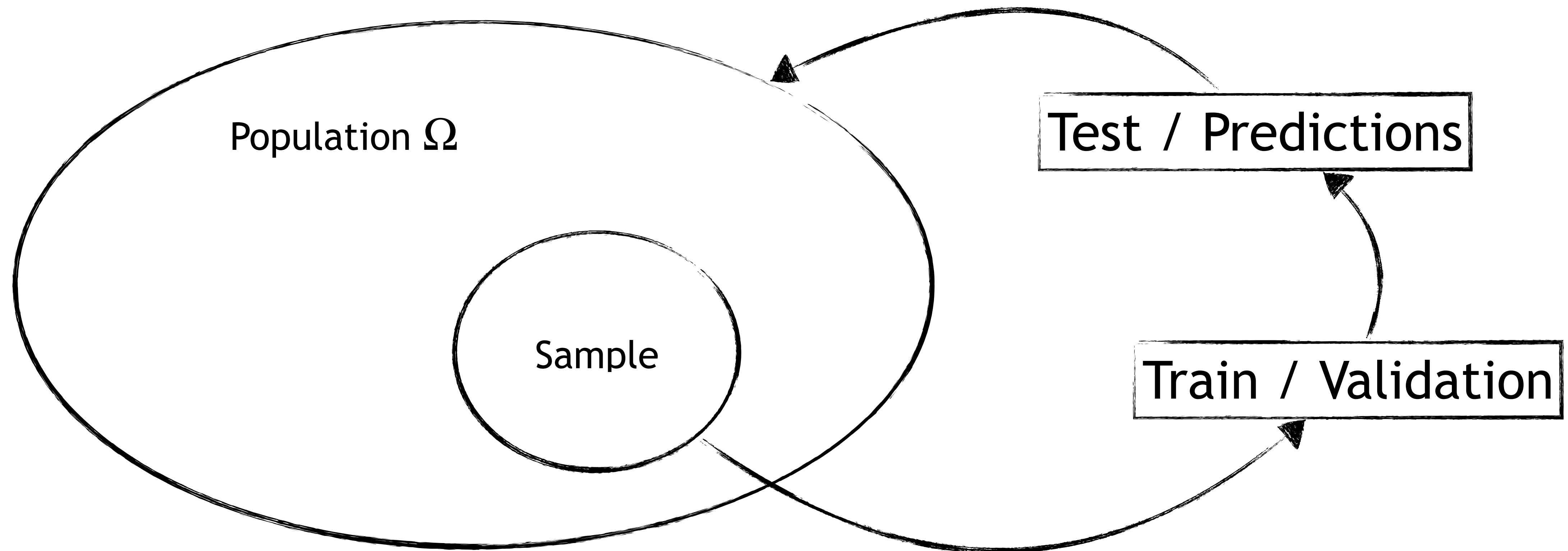
Underfitting

Definition : The model is too simple to capture underlying patterns. It can be visualised by a poor performance both on training and validation sets.

=> Make the model more complex by modifying its parameters or completely change the model.

Overfitting

Definition : The model learns noise and details from the training data. It can be visualised by a good performance on training set but poorly on validation set.

=> Simplify the model, regularisation or training data set not well defined regarding the validation set (split to redo).

Underfitting / Overfitting

__Practice :__ Use the data set overfitting_data.csv, train a Polynomial Regression model on it and plot the results. Do it for $n = 1, 2$ and 10.

Underfitting / Overfitting
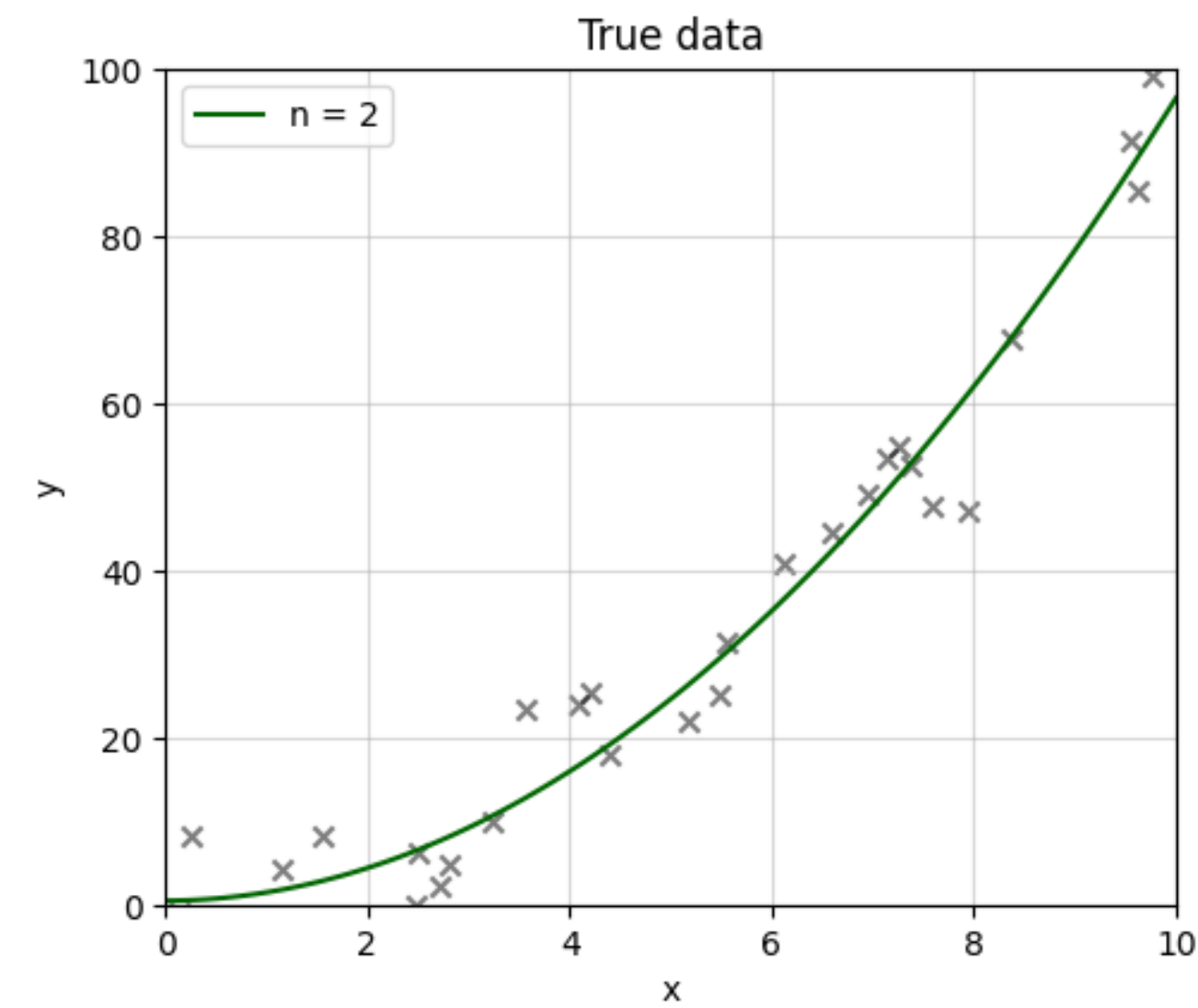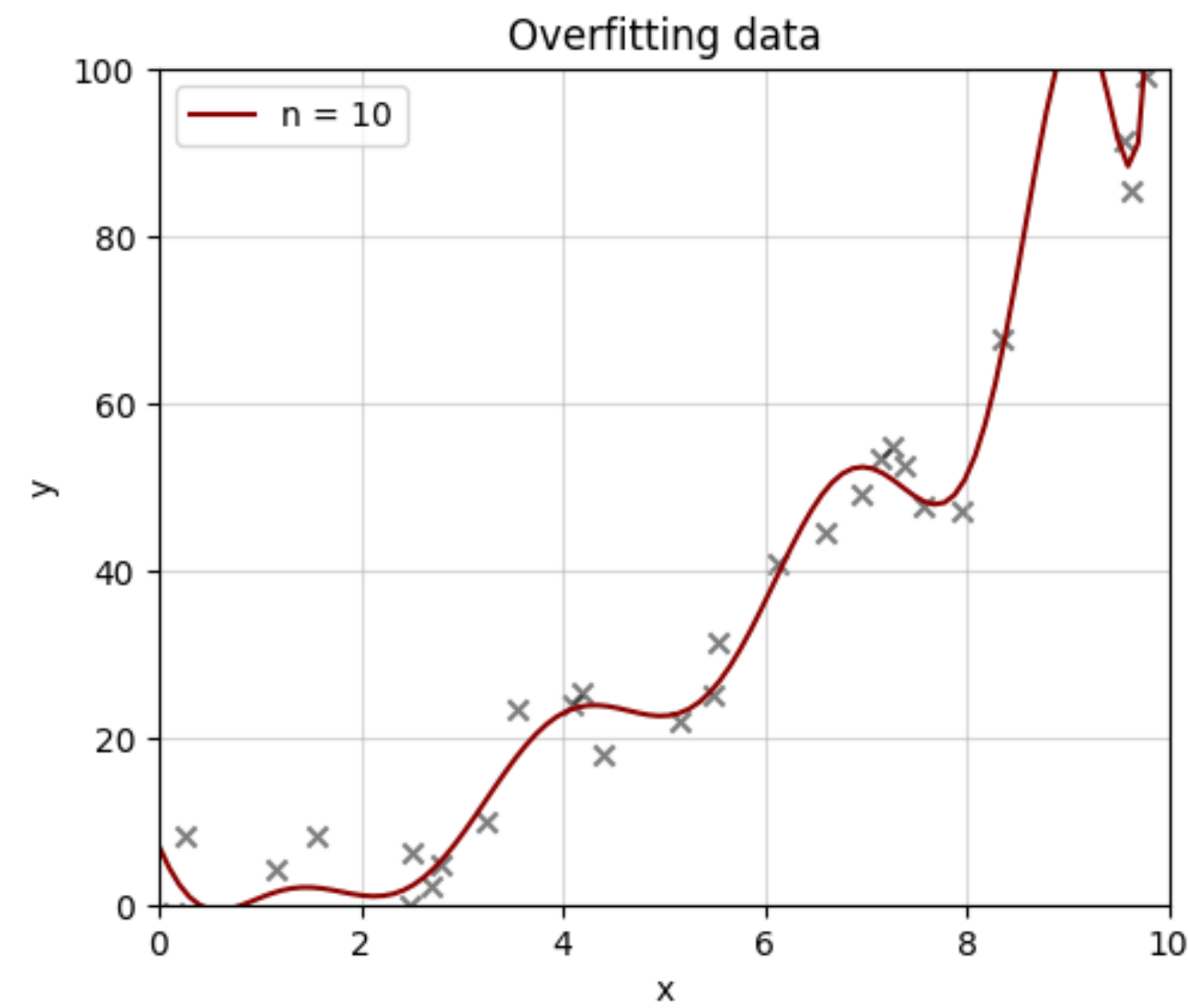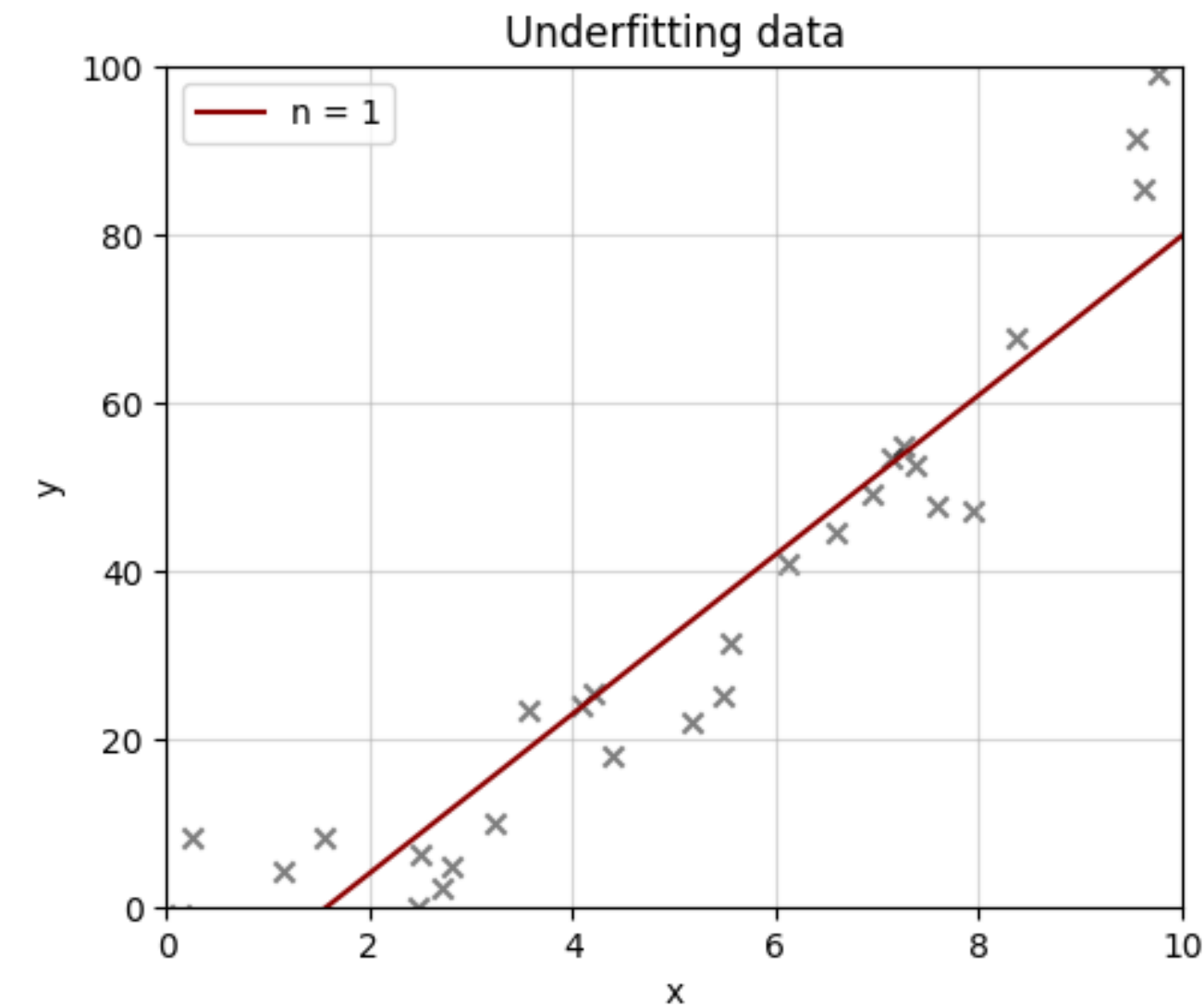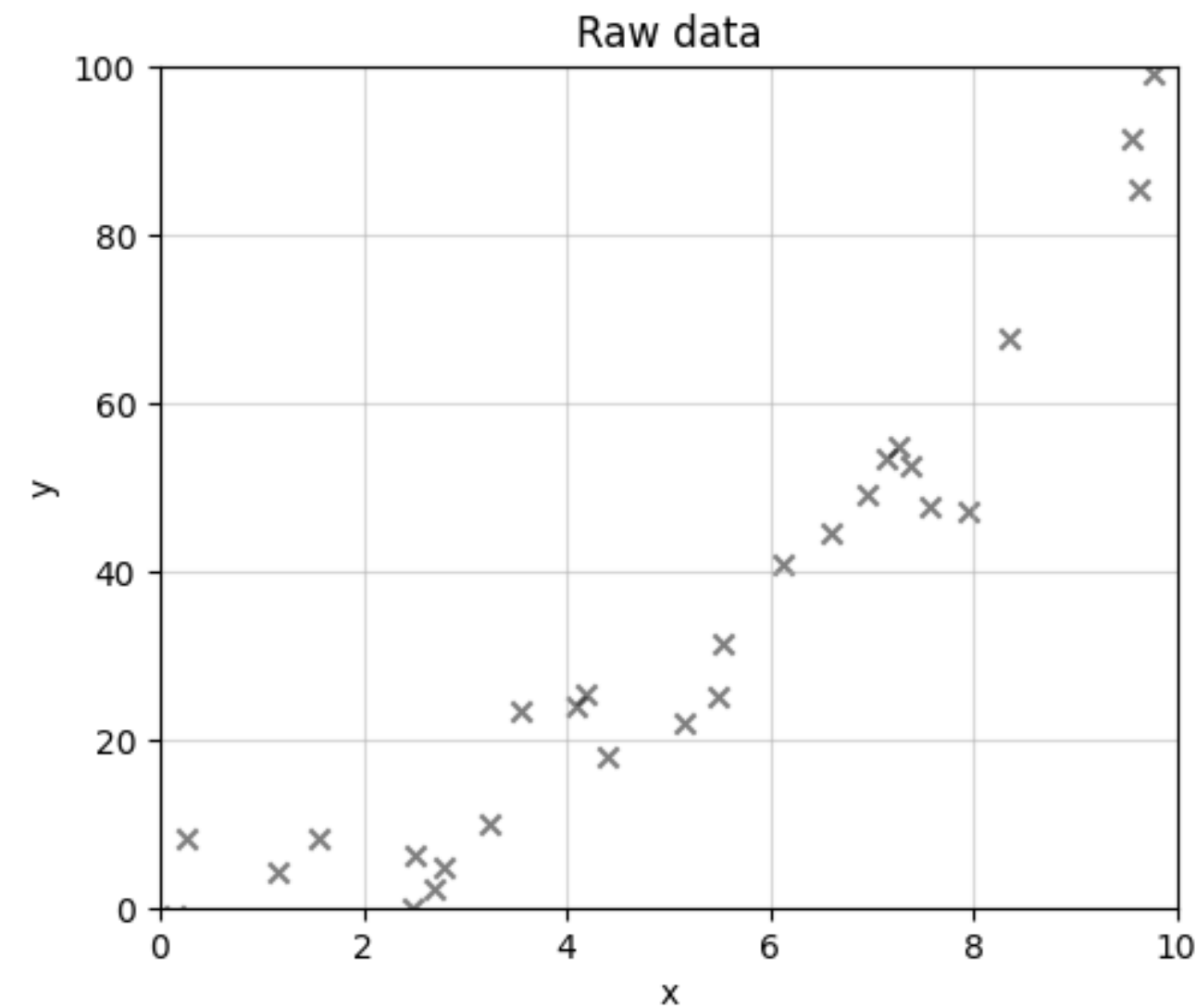
Supervised learning

Definition : Supervised learning is a type of machine learning where the model is trained on a <u>labeled dataset</u>, meaning each input data point is paired with the correct output (target).

=> The goal is for the model to learn the <u>mapping from inputs to outputs</u> and make predictions on new unseen data.

Supervised learning
- Example -

✣ Spam detection in mails => <u>Input :</u> email text / <u>Label :</u> spam or not spam

✣ Prediction of house prices => <u>Input :</u> number of rooms, area, location / <u>Label :</u> price.

✣ Medical diagnosis => <u>Input :</u> patient symptoms and test results / <u>Label :</u> disease present or not.

✣ Image classification => <u>Input :</u> image pixels / <u>Label :</u> "cat", "dog", etc...

Unsupervised learning

__Definition :__ Unsupervised learningn is a type of machine learning where the model is given <u>unlabeled data</u>.

=> The goal is to find patterns, groupings, or structures within the data without knowing the "correct" answers.

Unsupervised learning
- Example -

✤ Customer segmentation => <u>Task :</u> grouping customers by purchasing behaviour.

✤ Anomaly detection => <u>Task :</u> Finding fraudulent transactions in banking data.

✤ Topic modelling => <u>Task :</u> discovering themes in a collection of new articles.

✤ Dimensionality reduction => <u>Task :</u> visualizing high-dimensional data (e.g., genetics, images) using PCA.

Supervised learning
- Regression methods -

__Definition :__ It models the relationship between one or more independent variables (features) and a <u>continuous dependent variable</u> (target).

=> It's a supervised learning method.

Regression methods
- Illustration of some models -

‣ _Simple linear regression :_ single feature vs target.

‣ _Multiple linear regression :_ multiples features to predict the target.

‣ _Polynomial regression :_ Useful for modelling non-linear trends.

‣ _Regularized regression :_ Penalizes weights to prevent overfitting and manage multicollinearity => Improves generalisation.

Supervised learning
- Classification methods -

Definition : It models the relationship between one or more independent variables (features) and a <u>categorical dependent variable</u> (target).

=> It's also a supervised learning method.

Classification methods
- Example -

‣ *Logistic regression :* Probability model for binary tasks.

‣ *Decicision tree :* Tree based model, interpretable model.

‣ *Random forest :* multiple random trees.

‣ *Support Vector Machines (SVM) :* Find optimal separating hyperplane.

‣ *Neural Networks :* Flexible and powerful (especially for high-dimensional inputs).

Evaluation of models
- Metrics -

How do you know you model is working better than the others ?

➡ You define metrics to evaluate the model's performance.

Evaluation of models
- Metrics -

How do you know you model is working better than the others ?

➡ You define metrics to evaluate the model's performance.

➡ But can you use the same metrics for regression and classification tasks ?
➡ How do you chose your metric ?

Evaluation of models
- Metrics for regression -

Goal : We need to have a metric that evaluate how close predictions of continuous variables are to the actual target variables.

1 - Error terms : Let y_i be the true value and $\hat{y}_i$ the predicted value:

$$\epsilon_i = y_i - \hat{y}_i$$

$$\Rightarrow \ \text{Error} = \sum_{i=1}^{n} \epsilon_i$$

Evaluation of models
- Metrics for regression -

2 - Mean Absolute Error (MAE) : It measures average absolute difference.

$$\text{MAE} = \frac{1}{n} \sum_{i=1}^{n} |y_i - \hat{y}_i|$$

Remark : Easy to interpret but doesn't penalise large errors.

Evaluation of models
- Metrics for regression -

3 - Mean Squared Error (MSE) : It measures the square of errors.

$$\text{MSE} = \frac{1}{n} \sum_{i=1}^{n} \left(y_i - \hat{y}_i \right)^2$$

Remark : Commonly used in optimisation!

Evaluation of models
- Metrics for regression -

4 - *Root Mean Squared Error (RMSE)* : Its measure has the same units as the target variable.

$$\text{RMSE} = \sqrt{\text{MSE}}$$

Remark : More sensitive to outliers than MAE.

Evaluation of models
- Metrics for regression -

5 - R-Squared (R^2) : It measures a proportion of variance explained.

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}$$

Remark : 0 means that the prediction is not better than the mean as a naïve prediction.

Evaluation of models
- Metrics for regression -

6 - *Mean Average Percentage Error (MAPE)* : It measures a prediction accuracy of the models.

$$\text{MAPE} = \frac{1}{n} \sum_{i=1}^{n} \left| \frac{y_i - \hat{y}_i}{y_i} \right|$$

Remark : Very intuitive error interpretation in terms of relative error.

Evaluation of models
- Metrics for classification -

**Goal :** We need to have a metric that evaluate the quality of the prediction given by a specific model.

**0 - Basic concepts:**
- _True Postive (TP) :_ true value = 1, predicted value = 1
- _True Negative (TN) :_ true value = 0, predicted value = 0
- _False Postive (FP) :_ true value = 0, predicted value = 1
- _False Negative (FN) :_ true value = 1, predicted value = 0

Evaluation of models
- Metrics for classification -

1 - Accuracy: It measures the proportion of correct predictions.

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

Remark : Issues when facing imbalanced classes.

Evaluation of models
- Metrics for classification -

2 - _Precision_ : It measures how many predicted positives were correct.

$$\text{Precision} = \frac{???}{???}$$

Remark : Useful for imbalanced classes.

Evaluation of models
- Metrics for classification -

2 - *Precision :* It measures how many predicted positives were correct.

$$\text{Precision} = \frac{TP}{TP + FP}$$

Remark : Useful for imbalanced classes.

Evaluation of models
- Metrics for classification -

3 - Recall : It measures how many actual positives were captured.

$$\text{Recall} = \frac{???}{???}$$

Remark : Useful for imbalanced classes.

Evaluation of models
- Metrics for classification -

**3 - Recall :** It measures how many actual positives were captured.

$$\text{Recall} = \frac{TP}{TP + FN}$$

Remark : Useful for imbalanced classes.

Evaluation of models
- Metrics for classification -

4 - F1-score : It measures harmonic mean of precision and recall.

$$\text{F1-score} = 2\frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

Remark : Useful for imbalanced classes.

Evaluation of models
- Metrics for classification -

5 - _Confusion matrix :_ It shows all 4 outcomes (TP, TN, FP, FN) in a 2x2 table.

		Prediction	
		0	**1**
Truth	**0**	TN	FP
	1	FN	TP

Evaluation of models
- Metrics for classification -

6 - *Log-loss (cross-entropy loss)* : It measures probabilistic confidence in predictions.

$$\text{Log-loss} = -\frac{1}{n} \sum_{i=1}^{n} \Big[y_i \log(p_i) + (1 - y_i)\log(1 - p_i) \Big]$$

Remark : Commonly used in optimisation!

Practice
- Cross-validation using sklearn -

Training process
- General overview -

- ***<u>Data preprocessing :</u>*** Cleaning, normalisation, encoding, split, etc... (EDA)

- ***<u>Model selection :</u>*** Chose one or multiples model types.

- ***<u>Loss function definition :</u>*** Quantify how wrong the model is (not performance!)

- ***<u>Optimization :</u>*** Algorithms to minimize loss such as gradient descent.

- ***<u>Evaluation :</u>*** Measure performance on training and validation sets.

Practice
- KNN -

❖ Monday : Metrics and training process

❖ Tuesday : Linear models
  ‣ Linear regression
  ‣ Polynomial & Interaction terms regression
  ‣ Ridge & Lasso regression

❖ Wednesday : Support Vector Machine models

❖ Thursday : Decision trees

❖ Friday : Time series analysis

Correction of the exercises

Linear Models

Linear regression

Definition : It's a classic model for regression task, the idea is to predict a continuous target variable with a linear combination of the features (categorical or continuous).

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_n x_n + \epsilon$$

Assumptions : Linearity, independence, normality of errors.

Linear regression
- How it trained technically ? -

To compute the « right » parameters of the model, we are aiming to minimise a loss. This loss can be MSE like:

$$\text{MSE} = \frac{1}{n} \sum_{i=1}^{n} \left(y_i - \hat{y}_i \right)^2$$

How to minimise it ?

Linear regression
- Gradient descent -

We start from a random parameter β and we iterate from it until convergence of the method. More formally it gives a iteration n the following:

$$\beta_{n+1} = \beta_n - \alpha \nabla_\beta \mathsf{MSE}$$

$$= \beta_n - \alpha \frac{2}{n} X^T (X\beta - y)$$

Linear regression
- Practice -

Exercise : Load the data set `load_diabetes` from sklearn library, and train and test a linear regression model on this data set.

Polynomial and interaction terms regression

Definition : We stick to a linear regression architecture but adding new feature to our data set. We add features combinations (e.g. x_i^2 or $x_i x_j$) to our set of features and do the same as before.

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_1 x_2 + \beta_4 x_2^2 + \epsilon$$

Remark : The main work here is done in EDA to determine which interaction should be included in your set of features

Ridge regression

Definition : We face overfitting with linear regression and there is no inside parameters to play with to compensate this. Then, we use « regularizers ». Here we use L2-regularization, also called **Ridge regression :**

$$\text{loss} = \text{MSE} + \lambda \sum_i \beta_i^2$$

Specificities : Encourage small weights (but not zero) which helps keeping features when they are all important, keep the model smooth and reduce sensitivity to individual features.

Lasso regression

**Definition :** Same situation as before but instead we use L1-regularization:

$$\text{loss} = \text{MSE} + \lambda \sum_i |\beta_i|$$

Specificities : Encourage weights to zero (<u>sparsity</u>), useful for feature selection, model robust when only few features matter.

Ridge and Lasso regressions
- Practice -

Exercise : Still on the data set `load_diabetes`, train and test Ridge and Lasso regression models and cross-validate the best parameter to use for λ.