Introduction to the basics of AI

Sessions 2-3

Z. TAIA-ALAOUI

Outline

- Statistical Tools
- Principal Component Analysis
- Usual Classification/Clustering Models
- Implementation

Statistical Tools - Dataset

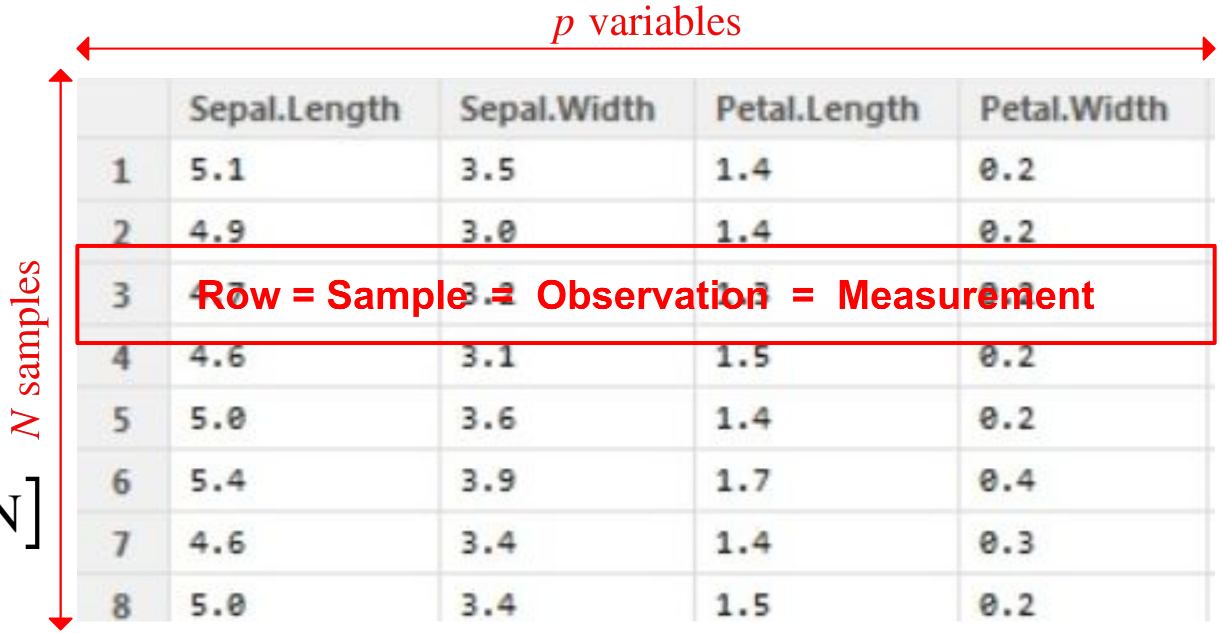
- **Sample**

$$X_{k\in[1,N]} \in R^p$$

- **Set of samples**

$$\mathbf{X} = \{X_k \in \mathrm{R}^p\}_{k\in[1,\mathrm{N}]}$$

IRIS DATASET


p variables
N samples
Row = Sample = Observation = Measurement

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width
1	5.1	3.5	1.4	0.2
2	4.9	3.0	1.4	0.2
3	4.7	3.2	1.3	0.2
4	4.6	3.1	1.5	0.2
5	5.0	3.6	1.4	0.2
6	5.4	3.9	1.7	0.4
7	4.6	3.4	1.4	0.3
8	5.0	3.4	1.5	0.2

Statistical Tools - Dataset

- **Set of N samples expressed in a space of p Variables**

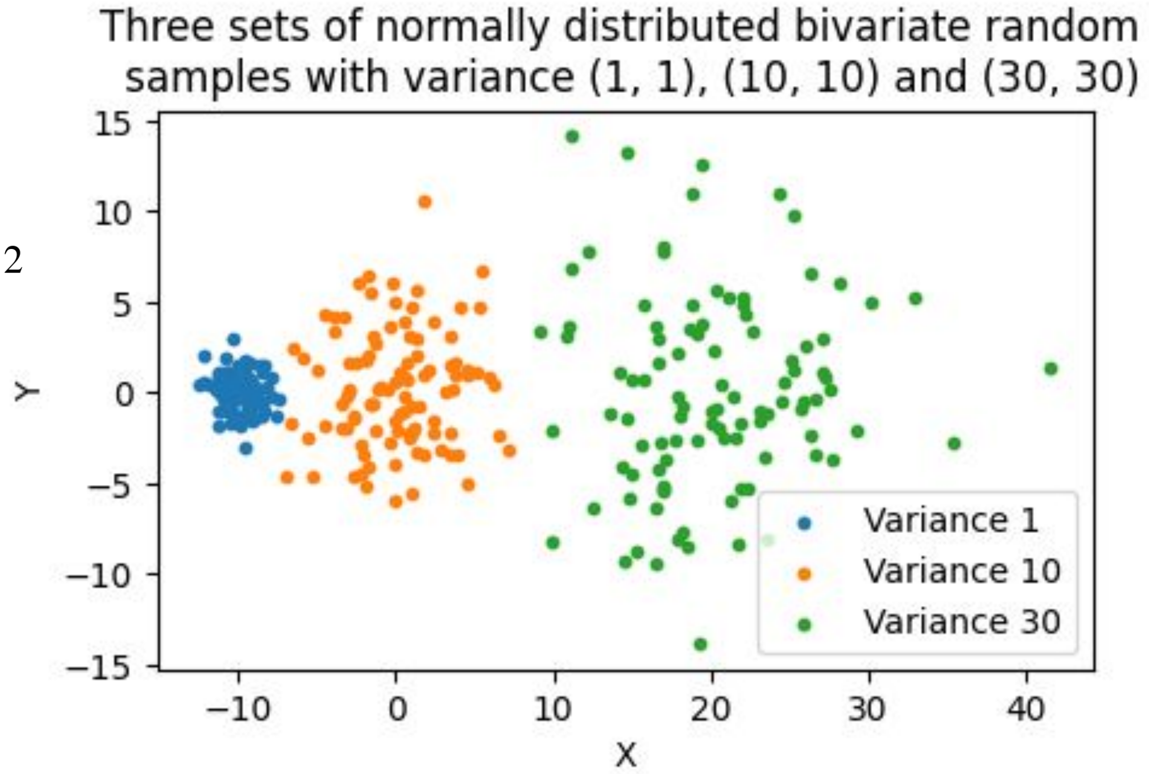
$$\mathbf{X} = \begin{array}{c} \\ \\ \\ \\ X_i \\ \\ \\ \end{array} \begin{array}{c} \begin{array}{ccccc} & & & V_j & \end{array} \\ \begin{pmatrix} x_{11} & x_{12} & x_{13} & \cdots & x_{1p} \\ x_{21} & x_{22} & . & . & . \\ . & . & . & . & . \\ . & . & . & . & . \\ . & . & . & . & . \\ x_{N1} & . & . & . & x_{Np} \end{pmatrix} \end{array} = [V_1, V_2, \ldots, V_p] = \begin{bmatrix} X_1^{\mathrm{T}} \\ X_2^{\mathrm{T}} \\ . \\ . \\ . \\ X_N^{\mathrm{T}} \end{bmatrix}$$

$$X_i = \begin{bmatrix} x_{i1} \\ x_{i2} \\ . \\ . \\ . \\ x_{ip} \end{bmatrix} \qquad V_j = \begin{bmatrix} x_{1j} \\ x_{2j} \\ . \\ . \\ . \\ x_{Nj} \end{bmatrix}$$

Statistical Tools - Variance

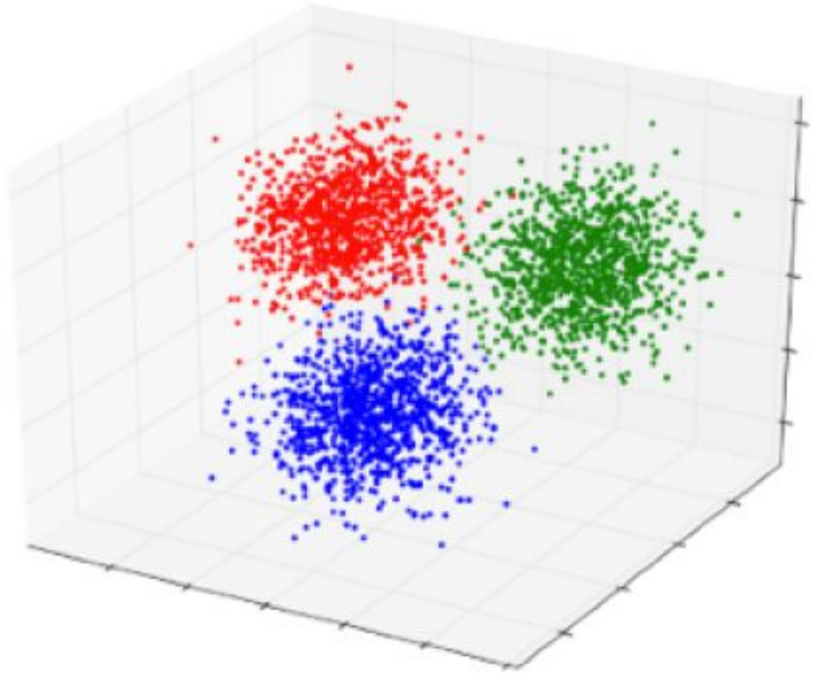
$$Var(V_j) = \frac{1}{N-1}\sum_{i=1}^{N}\left(x_{ij} - \overline{V_j}\right)^2$$

$$\overline{\mathrm{V}}_{\mathrm{j}} = \frac{1}{N}\sum_{i=1}^{N} x_{ij}$$



Statistical Tools - Co-Variance

$$Cov\left(\boldsymbol{V}_i, \boldsymbol{V}_j \right) = \frac{1}{N-1} \sum_{k=1}^{N} \left(x_{ki} - \overline{V}_i \right) \left(x_{kj} - \overline{V}_j \right)$$



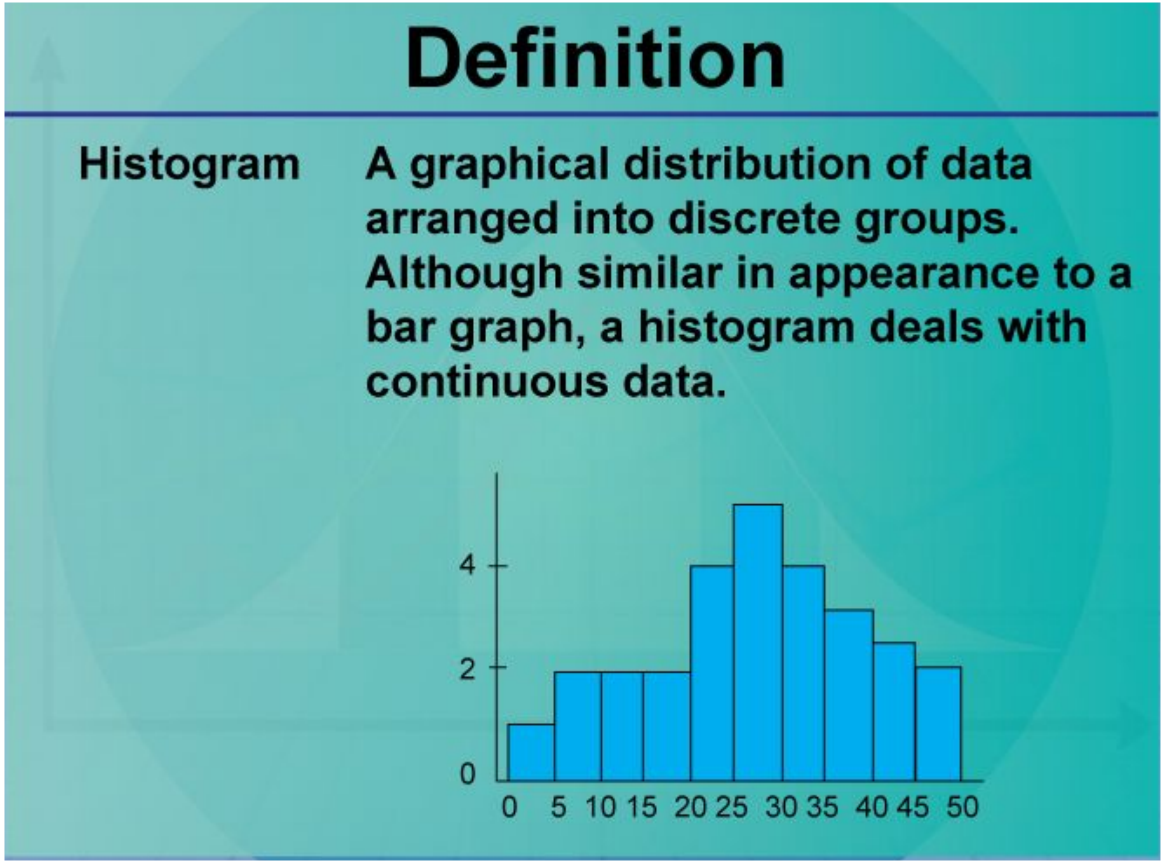
Statistical Tools - Continuous Variables

- **Covariance of matrix = A**

$$\begin{bmatrix} Var(\boldsymbol{X}_1) & Cov(\boldsymbol{X}_1, \boldsymbol{X}_2) & . & . \; . & Cov(\boldsymbol{X}_1, \boldsymbol{X}_N) \\ Cov(\boldsymbol{X}_2, \boldsymbol{X}_1) & Var(\boldsymbol{X}_2) & . & . \; . & Cov(\boldsymbol{X}_2, \boldsymbol{X}_N) \\ . & . & Var(\boldsymbol{X}_3) & . \; . & . \\ . & . & . & . \; . & . \\ . & . & . & . \; . & . \\ Cov(\boldsymbol{X}_N, \boldsymbol{X}_1) & . & . & . \; . & Var(\boldsymbol{X}_N) \end{bmatrix} = \frac{1}{N-1} X^T X$$

For standardized data

Statistical Tools - Visualization



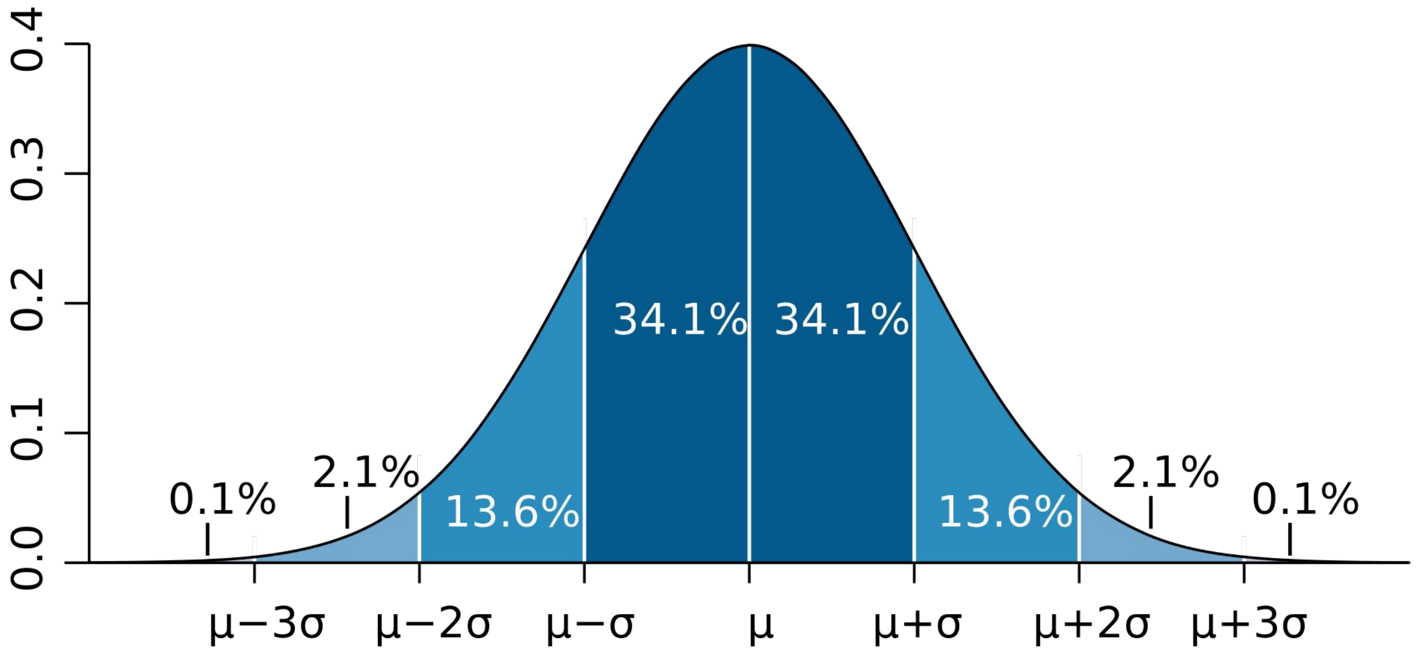
Definition

Histogram — A graphical distribution of data arranged into discrete groups. Although similar in appearance to a bar graph, a histogram deals with continuous data.

0
2
4
0
5
10
15
20
25
30
35
40
45
50

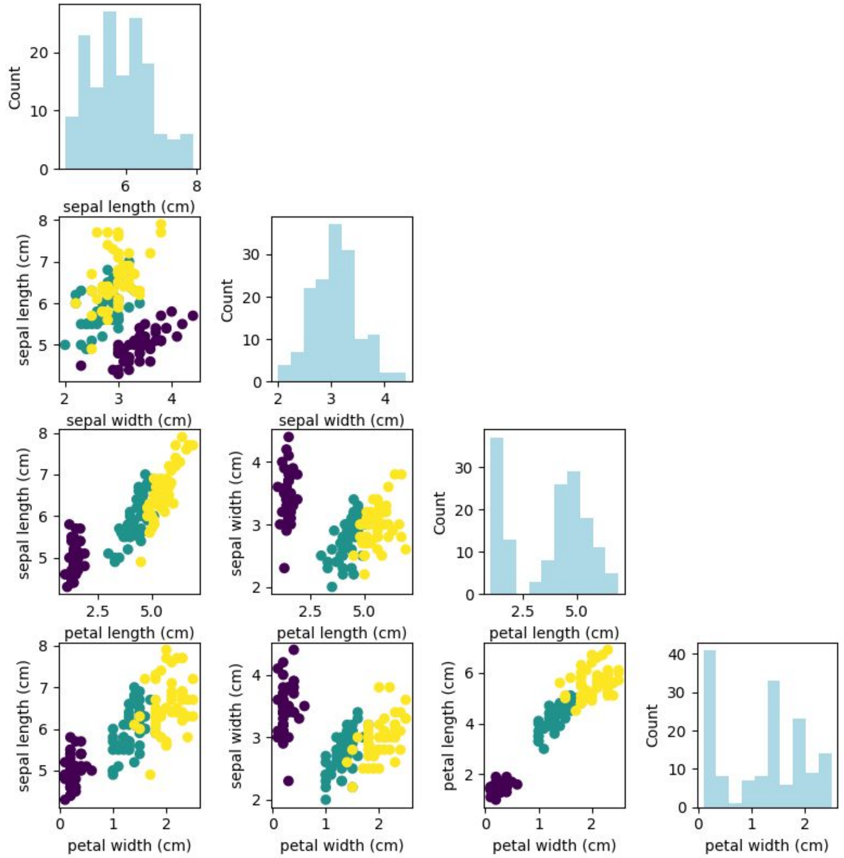
Statistical Tools - Visualization

Statistical Distribution (Example of Normal of distribution)

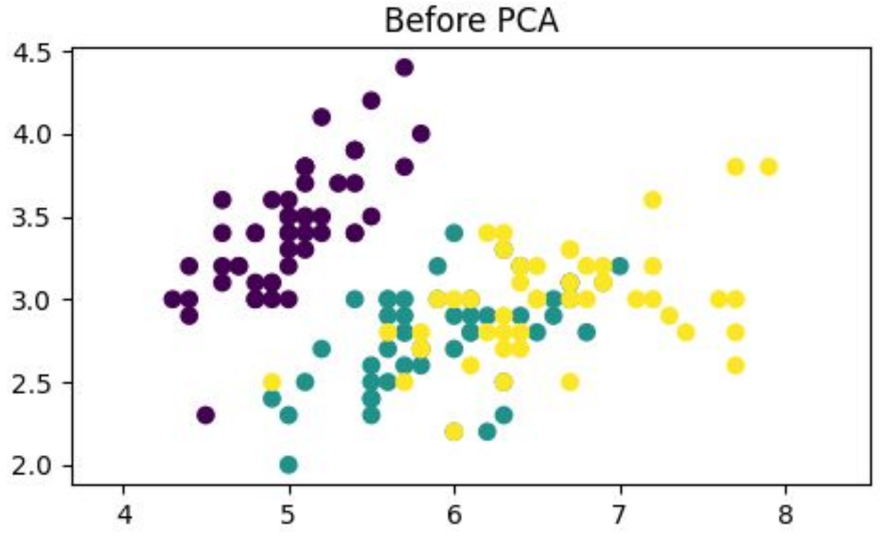


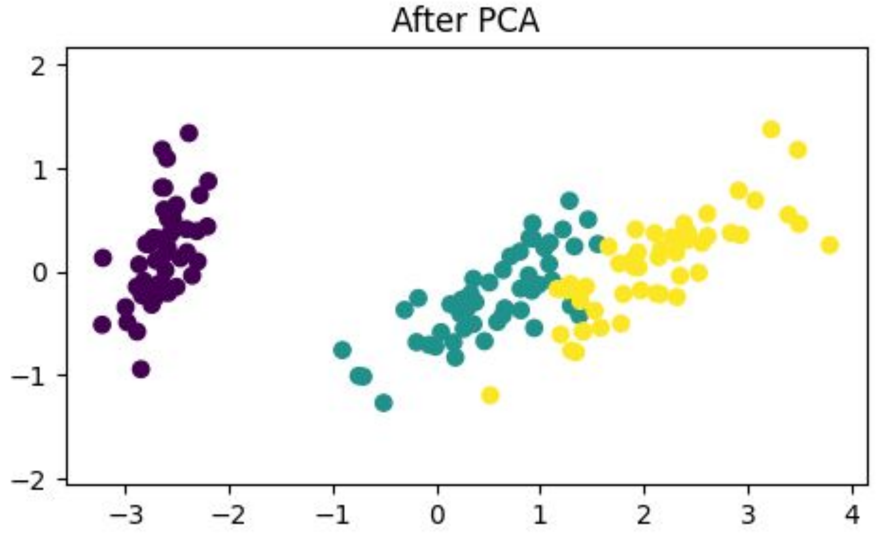
Statistical Tools - Visualization

Scatter Plots

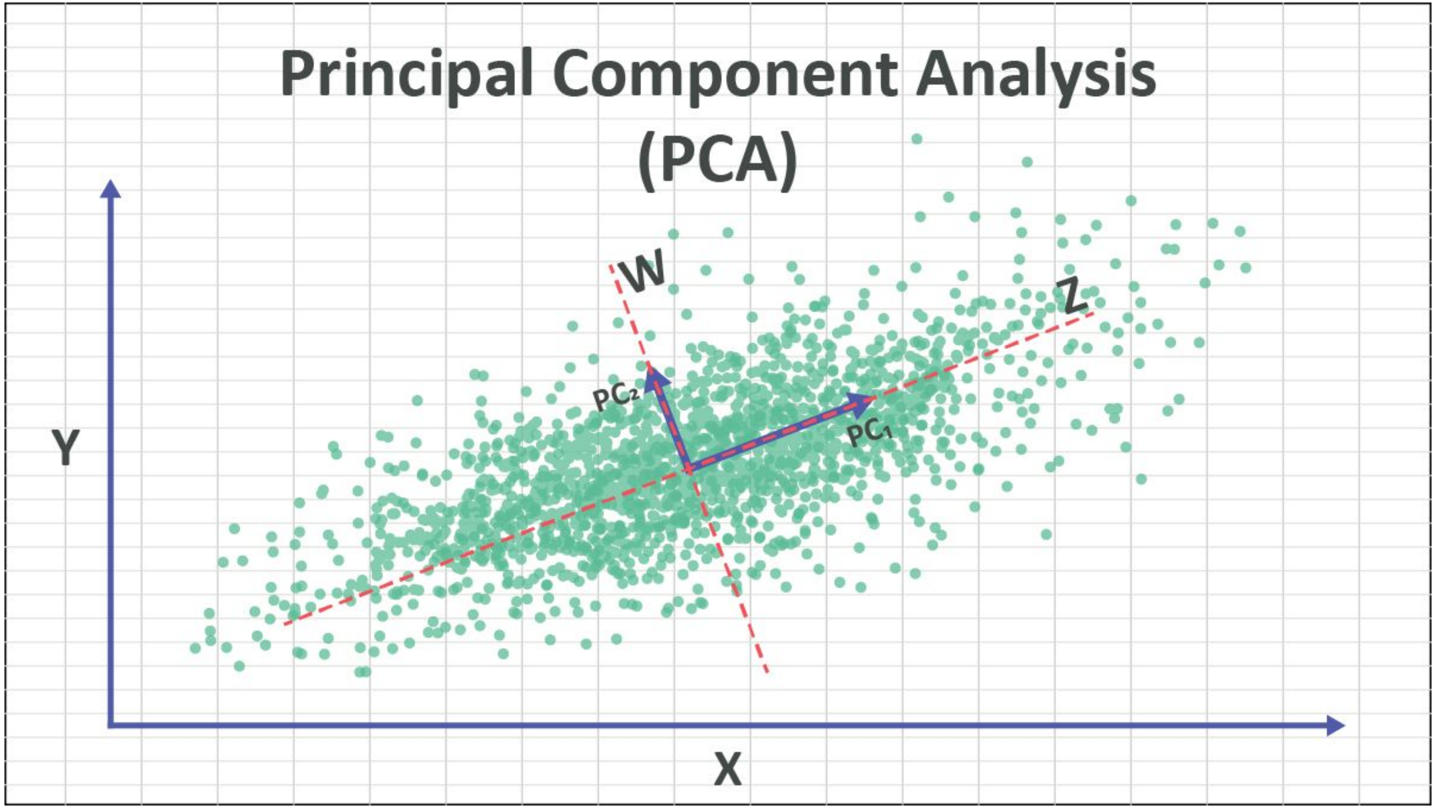


Principal Component Analysis





Principal Component Analysis



Principal Component Analysis

- PCA computes directions according to which the data **variance is maximized**
- The directions are given by a mathematical frame of **orthogonal independent vectors**
- PCA does **not lead to dimension loss**
- PCA can be used to **reduce dimension**
- PCA can help for **explainability**
- PCA can be used to **select important features**

PCA Computation

- Each square matrix A (such as the covariance of X!) can be decomposed as:

$$A = PDP^{-1}$$

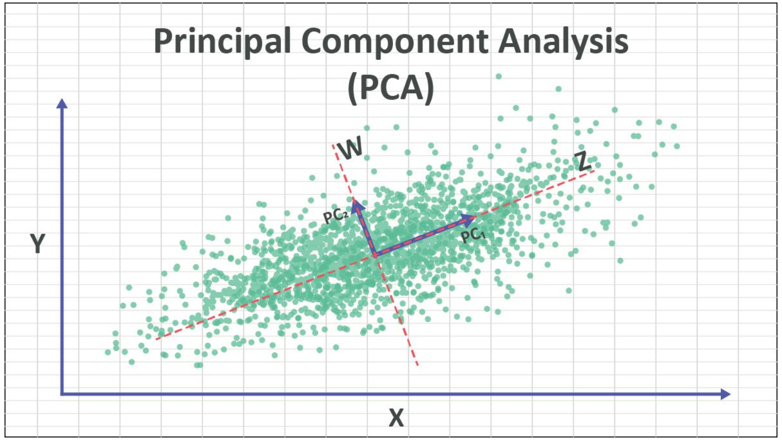
$$D = P^{-1}AP = \begin{pmatrix} \lambda_1 & 0 & 0 & \ldots 0 & 0 \\ 0 & \lambda_2 & 0 & \ldots & 0 \\ 0 & 0 & . & \ldots & . \\ . & . & . & \ldots & . \\ . & . & . & \ldots & 0 \\ 0 & 0 & 0 & \ldots & \lambda_p \end{pmatrix}, \lambda \text{ being the eigenvalues of A}$$

https://en.wikipedia.org/wiki/Diagonalizable_matrix

PCA Computation

- Eigenvectors

$$AV = \lambda V$$



- **Eigenvectors give the PCA components**
- **Eigenvalues give the explained variances of the components**

PCA Computation

- New coordinates

$$X' = X.V$$

- **The new coordinates hopefully improve data separability in the new space defined by vectors V**
- **If so, training can be performed using the new coordinates (new feature space)**

MCQ

- Is PCA a supervised or unsupervised method ?

- Can you cite 2 different uses of PCA ?

MCQ

- Is PCA a supervised or unsupervised method ?
  - Unsupervised

- Can you cite 2 different uses of PCA ?

MCQ

- Is PCA a supervised or unsupervised method ?
    - Unsupervised

- Can you cite 2 different uses of PCA ?
    - Dimensionality Reduction

MCQ

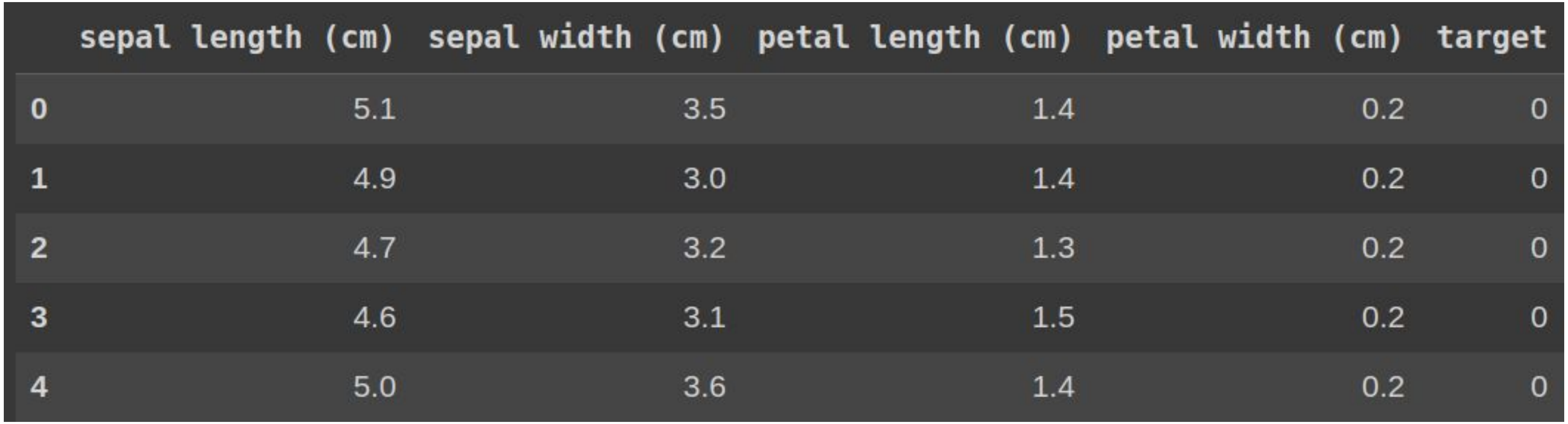
- Is PCA a supervised or unsupervised method ?
    - Unsupervised

- Can you cite 2 different uses of PCA ?
    - Dimensionality Reduction
    - Feature Selection

Training a classification model - Supervised

- Training supposes an **input and an output** (X, y)
- Supervised learning supposes also the **existence of truth** y_true
- Training involves the computation and an **error that quantifies** the **gap** between **prediction** y_predict and **truth** y_true

Training a classification model - Supervised



	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)	target
0	5.1	3.5	1.4	0.2	0
1	4.9	3.0	1.4	0.2	0
2	4.7	3.2	1.3	0.2	0
3	4.6	3.1	1.5	0.2	0
4	5.0	3.6	1.4	0.2	0

Training a classification model - Supervised

- In experimental research, the collected data must be **labeled**.
- Supervised training feeds the model with **measurements *and* labels**
- The model can be validated only through splitting the data into at least two groups: **Train VS Test (70%, 30%)**, or three groups: **Train (60%) VS Validate (10%) VS Test (30%)**

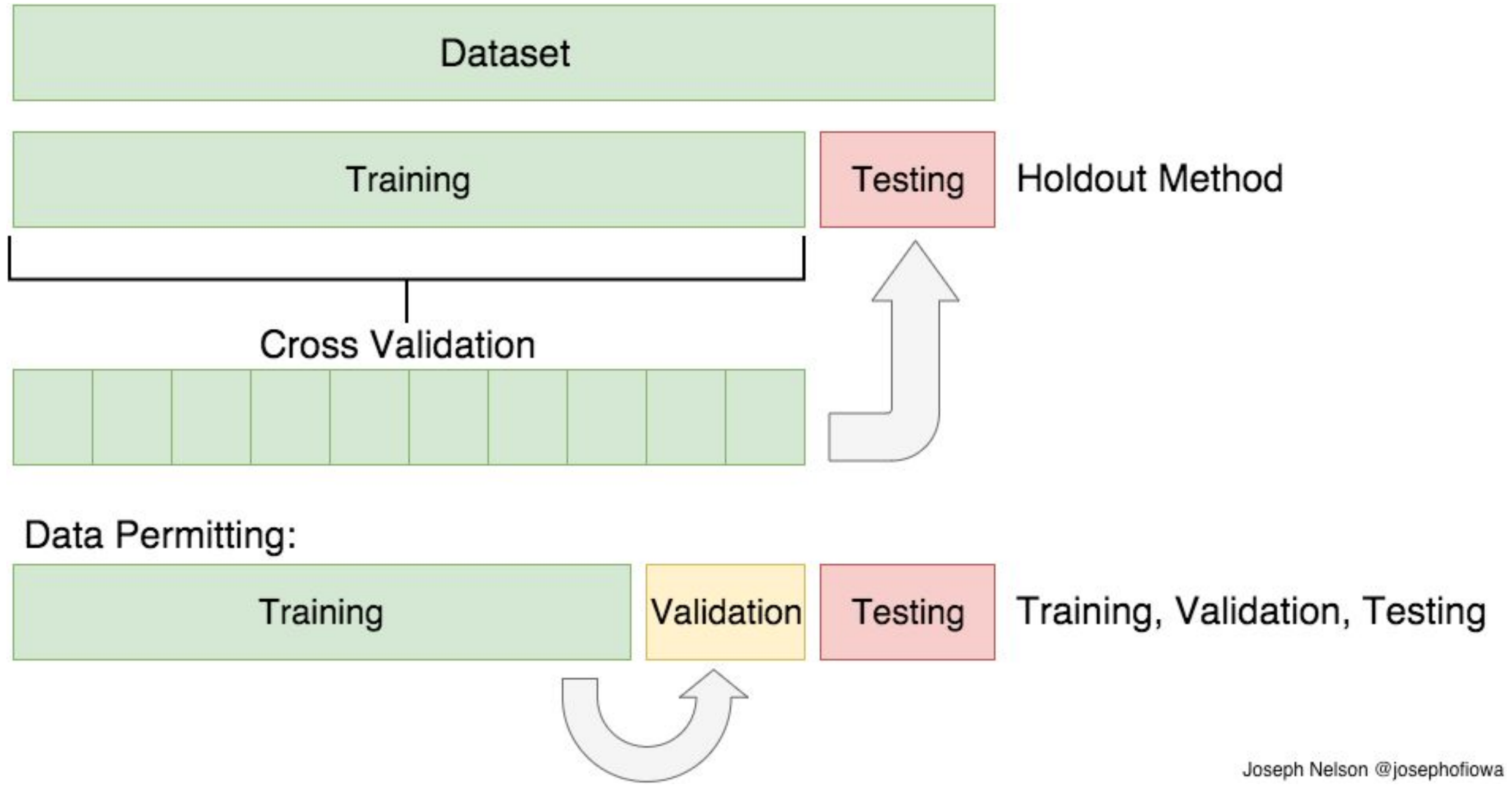
Training a classification model - Loss Function

- A **loss function** is a measure that is computed during training to fit the model parameters

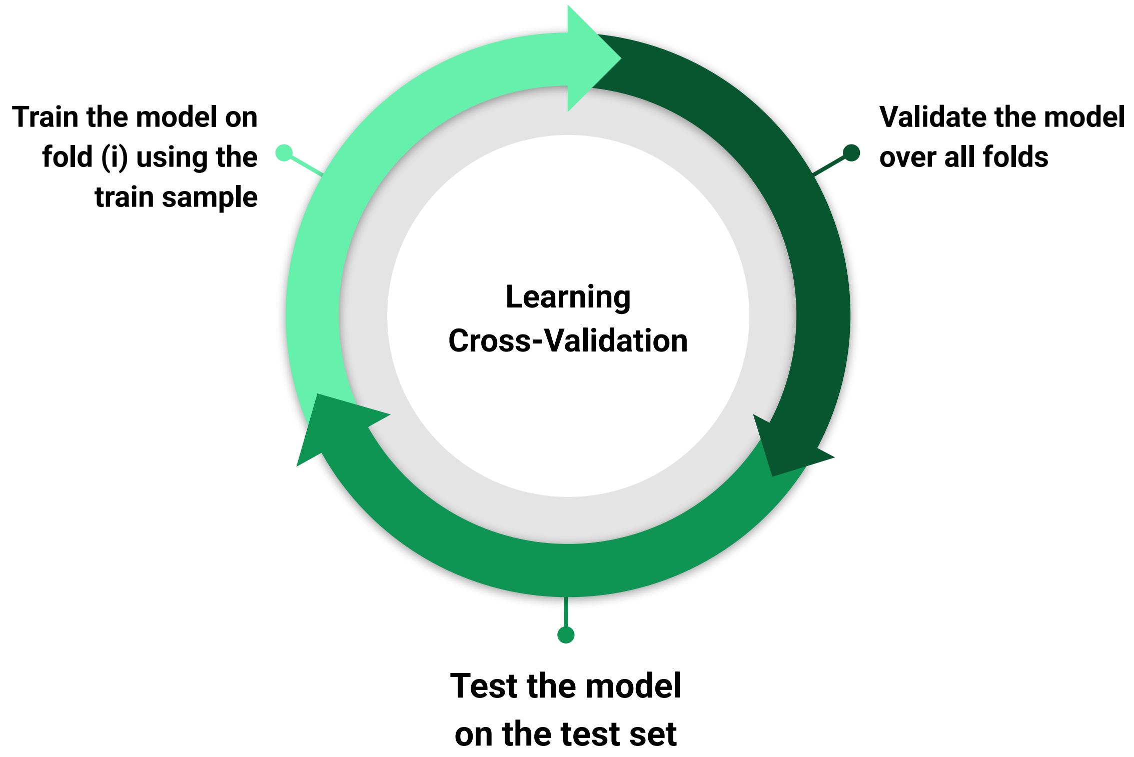
- **Some models do not have a loss function** such as KNN

- Some examples of loss functions for **classification** models are: *Softmax Cross **Entropy**, Sparse Cross **Entropy**, and Kullback-Leibler **Divergence***

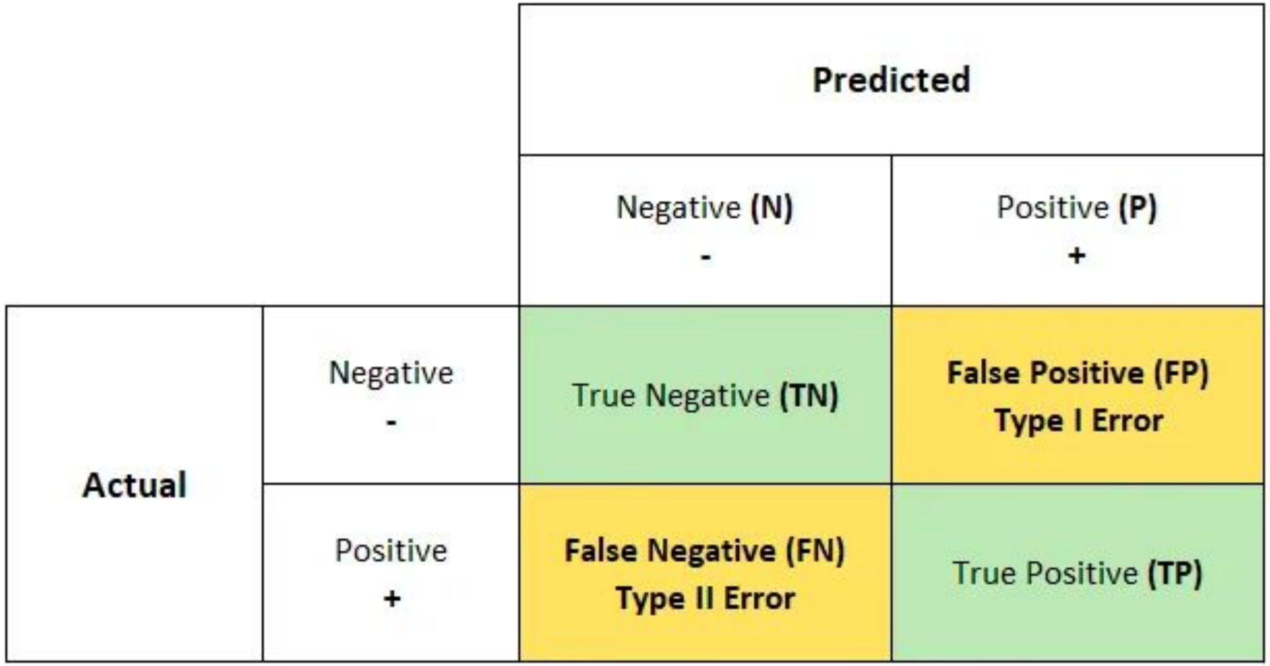
Training a classification model - Cross Validation



Training a classification model - Validation and Test

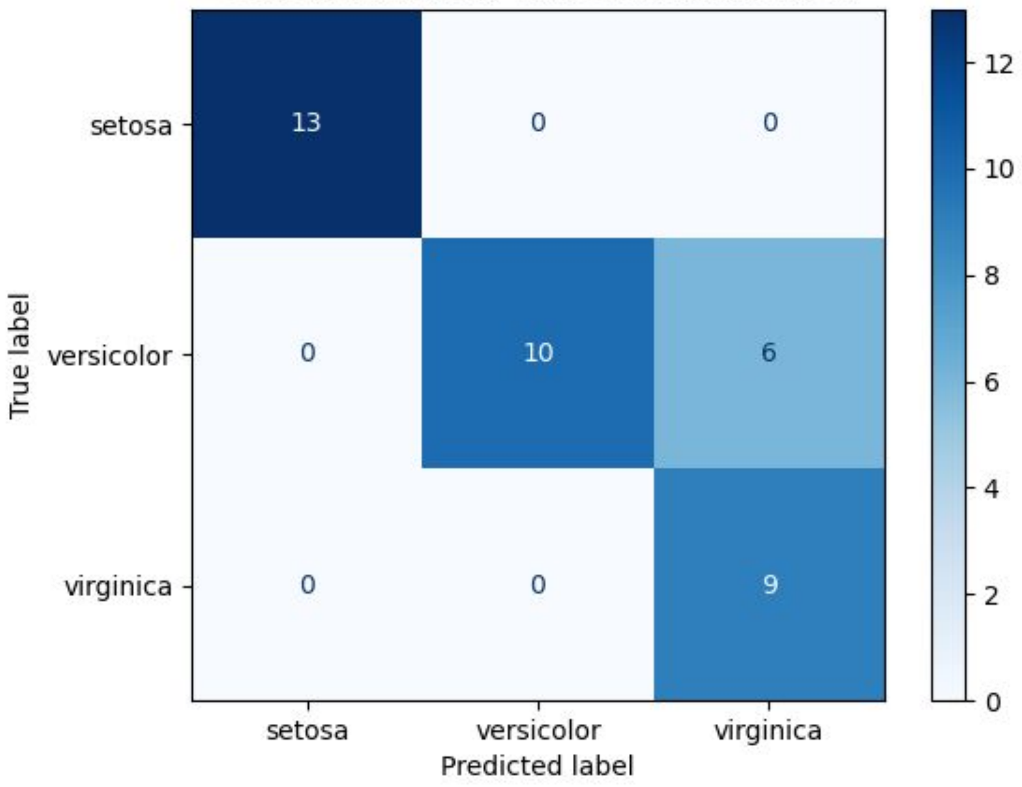

Train the model on fold (i) using the train sample
Validate the model over all folds
Learning Cross-Validation
Test the model on the test set

Training a classification model - Classification Errors

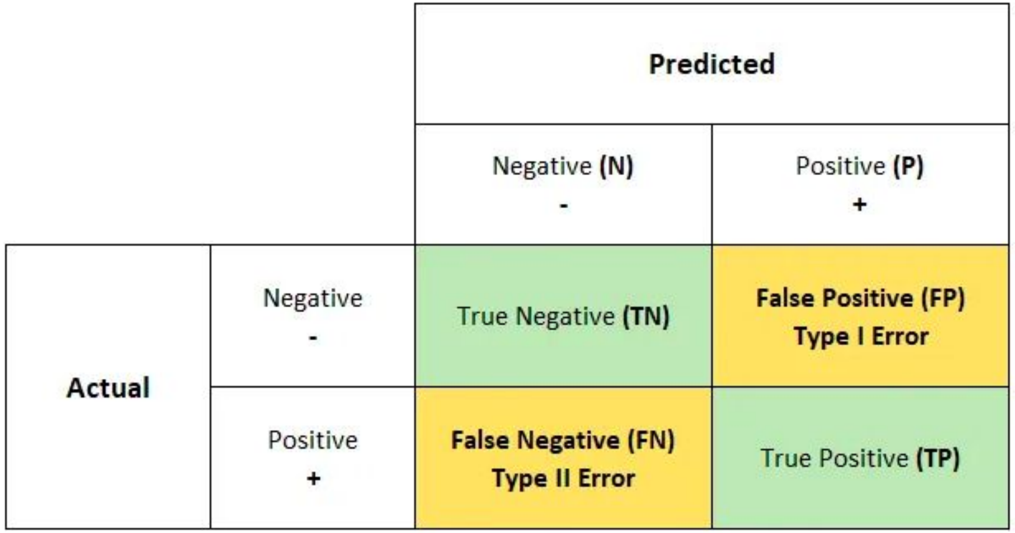


		Predicted	
		Negative **(N)** -	Positive **(P)** +
Actual	Negative -	True Negative **(TN)**	**False Positive (FP) Type I Error**
	Positive +	**False Negative (FN) Type II Error**	True Positive **(TP)**

Training a classification model - Classification Errors



Training a classification model - Accuracy Measures



		Predicted	
		Negative **(N)** -	Positive **(P)** +
Actual	Negative -	True Negative **(TN)**	**False Positive (FP) Type I Error**
	Positive +	**False Negative (FN) Type II Error**	True Positive **(TP)**

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

$$Recall = \frac{TruePositive}{TruePositive + FalseNegative}$$

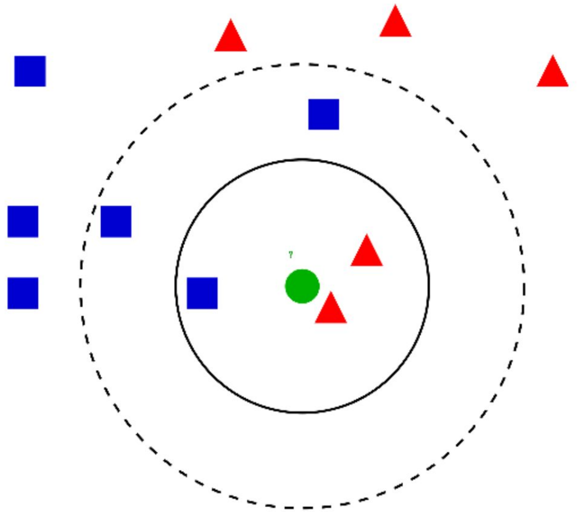
$$Precision = \frac{TruePositive}{TruePositive + FalsePositive}$$

$$F1 = 2.\frac{Precision \times Recall}{Precision + Recall}$$

K-NN

- K-NN is typically used as a classification algorithm, working off the assumption that similar points can be found near one another.
- For classification problems, a class label is assigned on the basis of a majority vote—i.e. the label that is most frequently represented around a given data point is used.

$$d(x,y) = \sqrt{\sum_{i=1}^{n} (y_i - x_i)^2}$$



K-NN

- **Advantages**:
  - Easy to implement: Given the algorithm’s simplicity and accuracy, it is one of the first classifiers that a new data scientist will learn.
  - Adapts easily: As new training samples are added, the algorithm adjusts to account for any new data since all training data is stored into memory.
  - Few hyperparameters: KNN only requires a k value and a distance metric, which is low when compared to other machine learning algorithms.

k-NN

- **Disadvantages**
  - Does not scale well: Since KNN is a lazy algorithm, it takes up more memory and data storage compared to other classifiers. This can be costly from both a time and infrastructure perspective.

k-NN

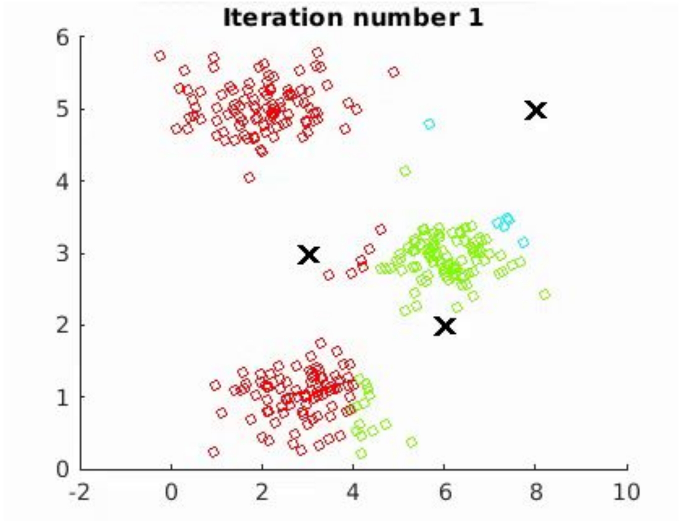
- **Disadvantages**
    - Curse of dimensionality: The KNN algorithm tends to fall victim to the curse of dimensionality, which means that it doesn’t perform well with high-dimensional data inputs.

k-NN

- **Disadvantages**
  - Prone to overfitting: Due to the “curse of dimensionality”, KNN is also more prone to overfitting. While feature selection and dimensionality reduction techniques are leveraged to prevent this from occurring, the value of k can also impact the model’s behavior.
  - Lower values of k can overfit the data, whereas higher values of k tend to “smooth out” the prediction values since it is averaging the values over a greater area, or neighborhood.

K-Means clustering

- K-means algorithm is an iterative algorithm that tries to partition the dataset into Kpre-defined distinct non-overlapping subgroups (clusters) where each data point belongs to only one group



K-Means clustering

- K-Means is an unsupervised/exploratory algorithm.

- It assigns data points to a cluster such that the sum of the squared distance between the data points and the cluster's centroid (arithmetic mean of all the data points that belong to that cluster) is at the minimum.

- The less variation we have within clusters, the more homogeneous (similar) the data points are within the same cluster.

K-Means clustering

- K-means in a few steps (1)
  - Specify number of clusters K.
  - Initialize centroids by first shuffling the dataset and then randomly selecting K data points for the centroids without replacement.
  - Keep iterating until there is no change to the centroids. i.e assignment of data points to clusters isn’t changing.

K-Means clustering

- **Advantages**

  Relatively simple to implement.

  Scales to large data sets.

  Guarantees convergence.

  Easily adapts to new examples.

  Generalizes to clusters of different shapes and sizes, such as elliptical clusters.

K-Means clustering

- **Limitations**

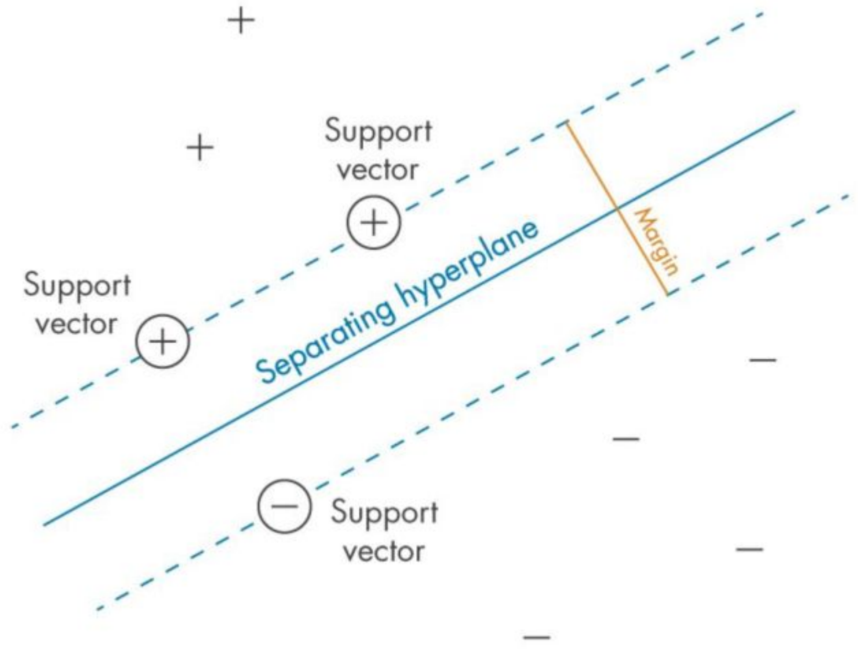
  Manual tuning of K

  Depending on initial Values

  Clustering data of varying sizes and density

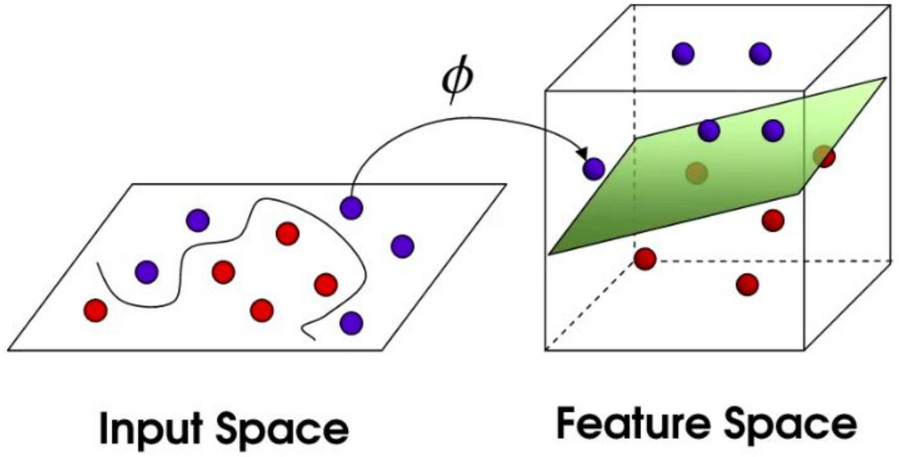
Support Vector Machine

Linear SVM: Optimal hyperplane for linearly separable patterns



Support Vector Machine

Non-Linear SVM: Extend to patterns that are not linearly separable by transformations of original data to map into new space – the Kernel function



Sources

https://towardsdatascience.com/train-test-split-and-cross-validation-in-python-80b61beca4b6

https://medium.com/analytics-vidhya/what-is-a-confusion-matrix-d1c0f8feda5

https://github.com/AmmirMahdi/Support-Vector-Machine-With-Python/tree/master

https://vitalflux.com/classification-model-svm-classifier-python-example/

https://towardsdatascience.com/the-kernel-trick-c98cdbcaeb3f