Third-year ISAE-SUPAERO engineering students
Research Area: Neuro & AI
December, 2020

Neuro & AI: Methods and Tools for Neuroergonomics

Introduction to Machine Learning

Nicolas Drougard[1]

[1]ISAE-SUPAERO DCAS, Toulouse, FRANCE

nicolas.drougard@isae-supaero.fr

Outline

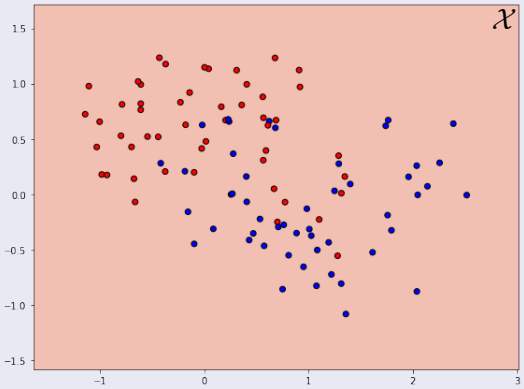
References

- [BBV04] Stephen Boyd, Stephen P Boyd, and Lieven Vandenberghe, *Convex optimization*, Cambridge university press, 2004, download link **here**.
- [HMS20] G. Haine, D. Matignon, and M. Salaün, *Mathématiques déterministes*, Tronc Commun Scientifique 1A, Formation Ingénieur ISAE-SUPAERO, 2020.
- [HTF09] Trevor Hastie, Robert Tibshirani, and Jerome Friedman, *The elements of statistical learning: data mining, inference, and prediction*, Springer Science & Business Media, 2009, download link **here**.

Outline

Decision Tree

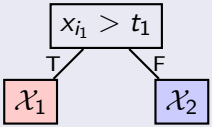
$\mathcal{X}$



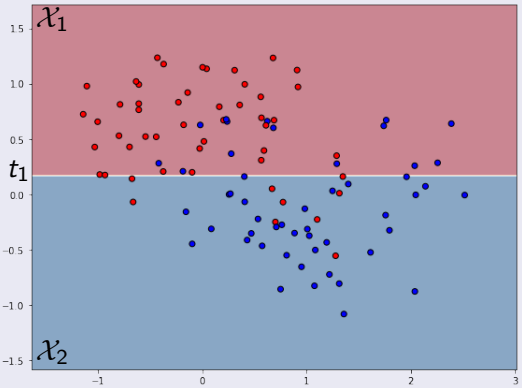
Splits using impurity criteria.

- sklearn: `tree.DecisionTreeClassifier`

Decision Tree



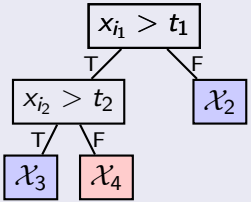
$i_1 = 2$



Splits using impurity criteria.

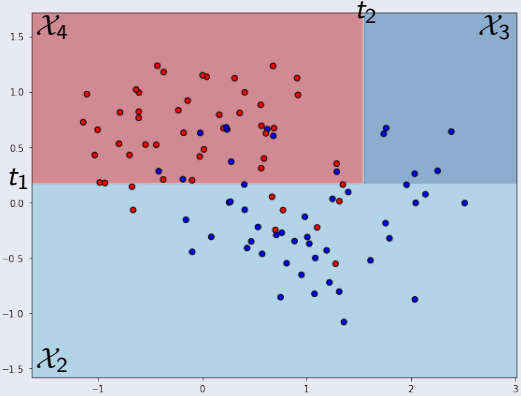
- sklearn: `tree.DecisionTreeClassifier`

Decision Tree



$i_1 = 2$

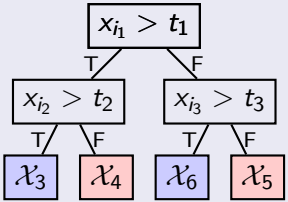
$i_2 = 1$



Splits using impurity criteria.

- sklearn: `tree.DecisionTreeClassifier`

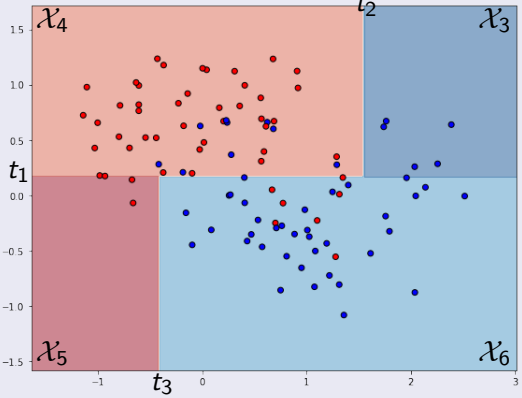
Decision Tree



$i_1 = 2$

$i_2 = 1$

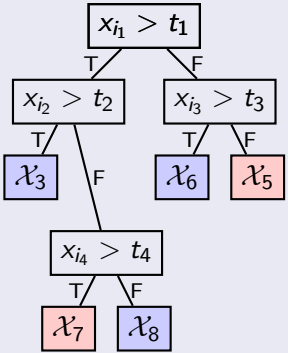
$i_3 = 1$



Splits using impurity criteria.

- sklearn: `tree.DecisionTreeClassifier`

Decision Tree

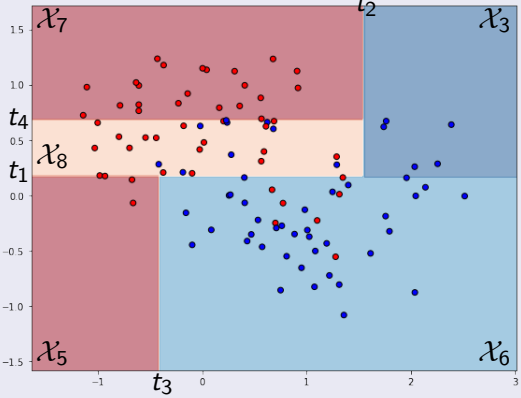


$i_1 = 2$

$i_2 = 1$

$i_3 = 1$

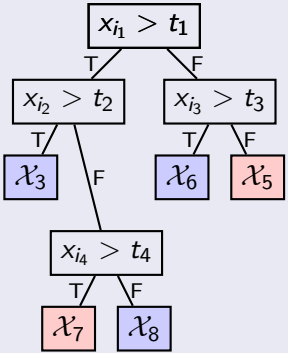
$i_4 = 2$



Splits using impurity criteria.

- sklearn: `tree.DecisionTreeClassifier`

Decision Tree



$i_1 = 2$

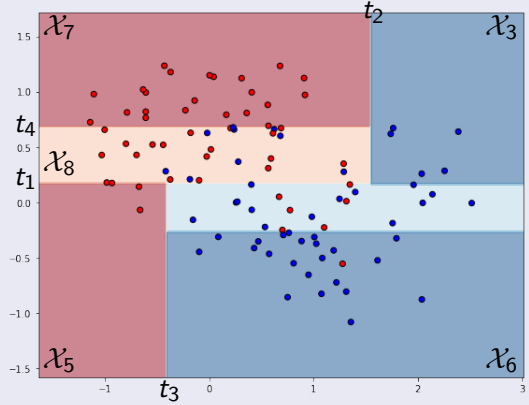
$i_2 = 1$

$i_3 = 1$

$i_4 = 2$

$i_5 = 2$

. . .



Splits using impurity criteria.

▶ sklearn: `tree.DecisionTreeClassifier`

Let's use the following notations:

- $Pos = \sum_{i=1}^{n} \mathbb{1}_{\{y_i = y_+\}}$
- $Neg = \sum_{i=1}^{n} \mathbb{1}_{\{y_i = y_-\}}$

Gini Impurity index

$$G = 2\left(\frac{Pos}{n}\right)\left(\frac{Neg}{n}\right)$$

Shannon Entropy

$$S = -\frac{Pos}{n}\ln\left(\frac{Pos}{n}\right) - \frac{Neg}{n}\ln\left(\frac{Neg}{n}\right)$$

Note that if $Pos = 0$ (or $Neg = 0$), $G = S = 0$.
These impurity criteria are maximal when $Pos = Neg$: $G = \frac{2}{4} = \frac{1}{2}$, and $S = -\ln(\frac{1}{2}) = \ln(2)$.

Random Forest

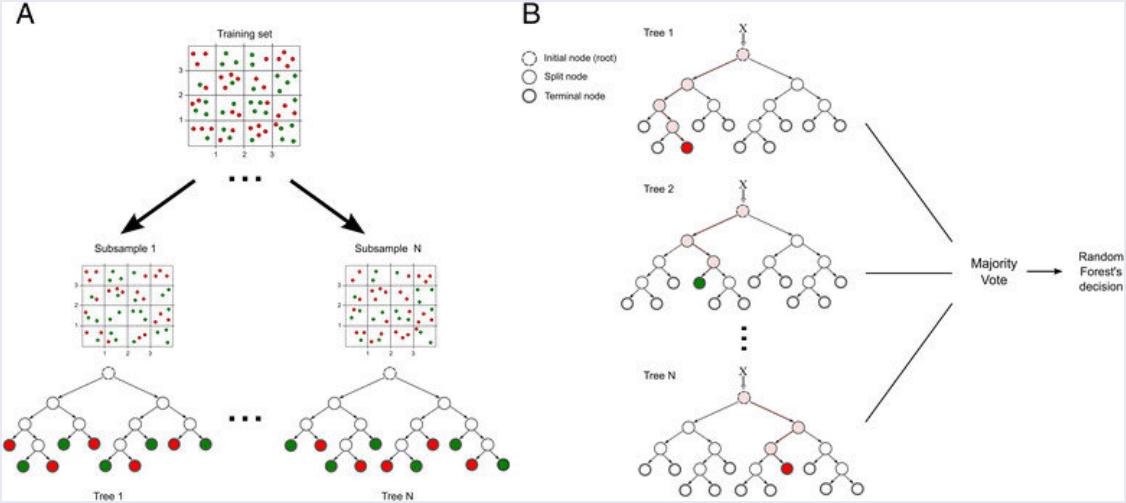


Figure from Machado et al. 2015,

- sklearn: `ensemble.RandomForestClassifier`

Linear Discriminant Analysis (LDA)

- Assumptions: $X \in \mathbb{R}^d$, $X \sim \mathcal{N}(\mu_y, \Sigma)$, $\forall y \in \mathcal{Y}$.
- Then, given $y \in \mathcal{Y}$, the density of X is:

$$f_y(x) = \frac{1}{(2\pi)^{\frac{d}{2}} \det(\Sigma)^{\frac{1}{2}}} \exp\left(-\frac{1}{2}(x-\mu_y)^T \Sigma^{-1}(x-\mu_y)\right).$$

- Using the Bayes rule, assuming a prior probability $\mathbb{P}(Y=y)$, the posterior probability is:

$$\mathbb{P}(Y=y \mid X=x) = \frac{f_y(x)\mathbb{P}(Y=y)}{\sum_{y\in\mathcal{Y}} f_y(x)\mathbb{P}(Y=y)}.$$

- Decision $y_+ \Leftrightarrow \frac{\mathbb{P}(Y=y_+ \mid X=x)}{\mathbb{P}(Y=y_- \mid X=x)} \geqslant 1$, decision $y_- \Leftrightarrow \frac{\mathbb{P}(Y=y_+ \mid X=x)}{\mathbb{P}(Y=y_- \mid X=x)} < 1$.
- Decision function $\nu(x) = \ln\left(\frac{\mathbb{P}(Y=y_+ \mid X=x)}{\mathbb{P}(Y=y_- \mid X=x)}\right)$ [Reminder: $c(x) = y_+ \Leftrightarrow \nu(x) \geqslant 0$].

Decision function of LDA

Reminder:

- densities: $f_y(x) = \frac{1}{(2\pi)^{\frac{d}{2}} \det(\Sigma)^{\frac{1}{2}}} \exp\left(-\frac{1}{2}(x-\mu_y)^T \Sigma^{-1}(x-\mu_y)\right)$,
- posterior probabilities: $\mathbb{P}(Y=y \mid X=x) = \frac{f_y(x)\mathbb{P}(Y=y)}{\sum_{y\in\mathcal{Y}} f_y(x)\mathbb{P}(Y=y)}$.

So, we can compute the decision function:

- $\frac{\mathbb{P}(Y=y_+ \mid X=x)}{\mathbb{P}(Y=y_- \mid X=x)} = \frac{f_{y_+}(x)\mathbb{P}(Y=y_+)}{f_{y_-}(x)\mathbb{P}(Y=y_-)}$.
- Decision function:

$$\begin{aligned}
\nu(x) &= \ln\left(\frac{\mathbb{P}(Y=y_+ \mid X=x)}{\mathbb{P}(Y=y_- \mid X=x)}\right) = \ln\left(\frac{f_{y_+}(x)}{f_{y_-}(x)}\right) + \ln\left(\frac{\mathbb{P}(Y=y_+)}{\mathbb{P}(Y=y_-)}\right) \\
&= \ln\left(\frac{\exp\left(-\frac{1}{2}(x-\mu_{y_+})^T\Sigma^{-1}(x-\mu_{y_+})\right)}{\exp\left(-\frac{1}{2}(x-\mu_{y_-})^T\Sigma^{-1}(x-\mu_{y_-})\right)}\right) + \ln\left(\frac{\mathbb{P}(Y=y_+)}{\mathbb{P}(Y=y_-)}\right)
\end{aligned}$$

Decision function of LDA

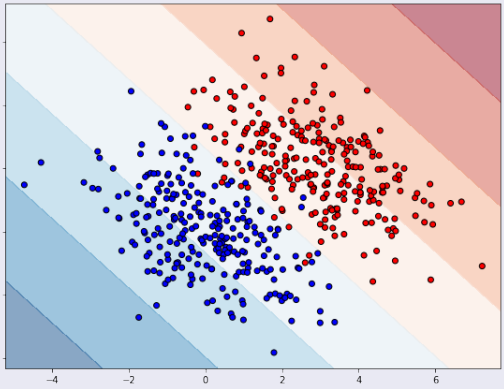
$$
\begin{aligned}
\nu(x) &= \ln\left(\frac{\exp\left(-\frac{1}{2}(x-\mu_{y_+})^T\Sigma^{-1}(x-\mu_{y_+})\right)}{\exp\left(-\frac{1}{2}(x-\mu_{y_-})^T\Sigma^{-1}(x-\mu_{y_-})\right)}\right) + \ln\left(\frac{\mathbb{P}(Y=y_+)}{\mathbb{P}(Y=y_-)}\right) \\
&= -\frac{1}{2}(x-\mu_{y_+})^T\Sigma^{-1}(x-\mu_{y_+}) + \frac{1}{2}(x-\mu_{y_-})^T\Sigma^{-1}(x-\mu_{y_-}) + \ln\left(\frac{\mathbb{P}(Y=y_+)}{\mathbb{P}(Y=y_-)}\right) \\
&= \frac{1}{2}\Big(-x^T\Sigma^{-1}x + \mu_{y_+}^T\Sigma^{-1}x + x^T\Sigma^{-1}\mu_{y_+} - \mu_{y_+}^T\Sigma^{-1}\mu_{y_+} \\
&\quad + x^T\Sigma^{-1}x - \mu_{y_-}^T\Sigma^{-1}x - x^T\Sigma^{-1}\mu_{y_-} + \mu_{y_-}^T\Sigma^{-1}\mu_{y_-}\Big) + \ln\left(\frac{\mathbb{P}(Y=y_+)}{\mathbb{P}(Y=y_-)}\right) \\
&= \frac{1}{2}\Big((\mu_{y_+}-\mu_{y_-})^T\Sigma^{-1}x + x^T\Sigma^{-1}(\mu_{y_+}-\mu_{y_-}) + (\mu_{y_-}-\mu_{y_+})^T\Sigma^{-1}(\mu_{y_-}+\mu_{y_+})\Big) + \ln(\ldots) \\
&= (\mu_{y_+}-\mu_{y_-})^T\Sigma^{-1}x + \frac{1}{2}(\mu_{y_-}-\mu_{y_+})^T\Sigma^{-1}(\mu_{y_-}+\mu_{y_+}) + \ln\left(\frac{\mathbb{P}(Y=y_+)}{\mathbb{P}(Y=y_-)}\right).
\end{aligned}
$$

Linear Discriminant Analysis (LDA)

The decision function is linear in x.

Prior & Gaussian parameter estimations:

- $\frac{\mathbb{P}(Y=y_+)}{\mathbb{P}(Y=y_-)} = \frac{Pos}{Neg}$.
- $\widehat{\mu_{y_+}} = \frac{1}{Pos}\sum_{i=1}^{n} \mathbb{1}_{\{y_i=y_+\}} x_i$,
  $\widehat{\mu_{y_-}} = \frac{1}{Neg}\sum_{i=1}^{n} \mathbb{1}_{\{y_i=y_-\}} x_i$.
- $\widehat{\Sigma} = \frac{1}{n-2}\left(\widehat{\Sigma_+} + \widehat{\Sigma_-}\right)$,
  $\widehat{\Sigma_+} = \sum_{i=1}^{n}(x_i-\mu_{y_+})^T(x_i-\mu_{y_+})\mathbb{1}_{\{y_i=y_+\}}$,
  $\widehat{\Sigma_-} = \sum_{i=1}^{n}(x_i-\mu_{y_-})^T(x_i-\mu_{y_-})\mathbb{1}_{\{y_i=y_-\}}$.

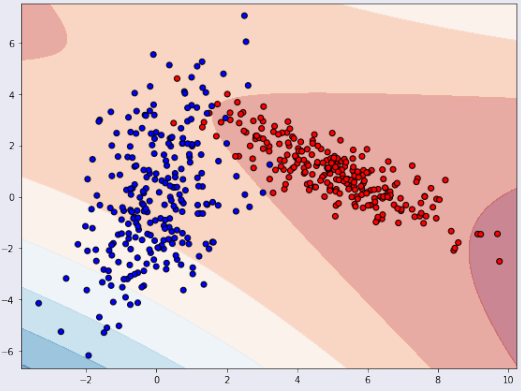


$$\Rightarrow \nu(x) = (\widehat{\mu_{y_+}} - \widehat{\mu_{y_-}})^T\Sigma^{-1}x + \frac{1}{2}(\widehat{\mu_{y_-}} - \widehat{\mu_{y_+}})^T\Sigma^{-1}(\widehat{\mu_{y_-}} + \widehat{\mu_{y_+}}) + \ln\left(\frac{Pos}{Neg}\right).$$

- sklearn: `discriminant_analysis.LinearDiscriminantAnalysis`

Quadratic Discriminant Analysis

- Now the covariance matrix depends on the class: Σ_y.
- Other assumptions hold.
- In this case, the decision function is **quadratic** in x.



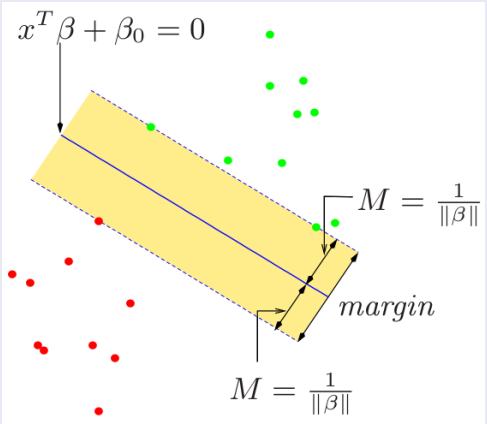
- sklearn: `discriminant_analysis.LinearDiscriminantAnalysis`

Support Vector Machine

- Decision function $\nu(x) = x^T\beta + \beta_0$, with $\beta \in \mathbb{R}^d$, $\|\beta\| = 1$ and $\beta_0 \in \mathbb{R}$.
- $x^T\beta + \beta_0 = 0 \Leftrightarrow$ hyperplane orthogonal to β.
- $|x^T\beta + \beta_0| =$ distance $x \leftrightarrow$ hyperplane
- Encoding $y_+ = 1$, $y_- = -1$.
- By choosing $\beta \in \mathbb{R}^d$, $\beta_0 \in \mathbb{R}$,
  maximize the margin M
  subject to $y_i(x_i^T \frac{\beta}{\|\beta\|} + \frac{\beta_0}{\|\beta\|}) \geqslant M$, $\forall 1 \leqslant i \leqslant n$,
  i.e. subject to $y_i(x_i^T\beta + \beta_0) \geqslant M\|\beta\|$, $\forall i$.

  $\Updownarrow$ with $M\|\beta\| = 1$

  minimize $\|\beta\|$
  subject to $y_i(x_i^T\beta + \beta_0) \geqslant 1$, $\forall 1 \leqslant i \leqslant n$.

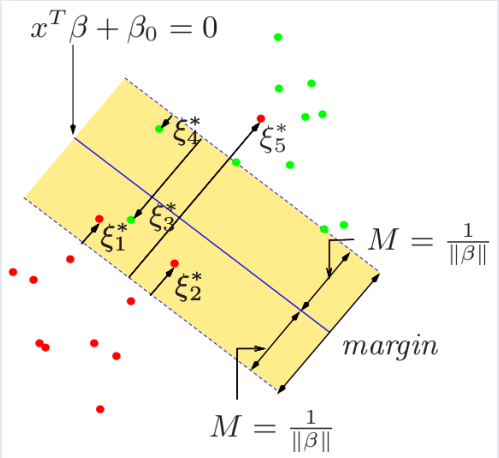


- sklearn: `svm.SVC`
- sklearn: `svm.SVR`

Support Vector Machine

If classes overlap, introduce ξ_i.

- Minimize $\|\beta\|$ subject to
$$\begin{cases} y_i(x_i{}^T\beta + \beta_0) \geqslant 1 - \xi_i, \ \forall 1 \leqslant i \leqslant n \\ \xi_i \geqslant 0, \text{ and } \sum_i \xi \leqslant \textit{constant} \end{cases}$$
$$\Updownarrow \text{ convex optimization [BBV04]}$$
minimize $\frac{1}{2}\|\beta\|^2 + C\sum_{i=1}^n \xi_i$
subject to $\xi \geqslant 0$, $y_i(x_i{}^T\beta + \beta_0) \geqslant 1$, $\forall i$.
- High (resp. low) $C > 0$ prioritizes a good classification (resp. a large margin).



▶ sklearn: `svm.SVC` ▶ sklearn: `svm.SVR`

More details in [HTF09].

Support Vector Machine and kernels

Using a kernel $K : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$, the resulting decision function has the non linear form

$$\nu(x) = \sum_{i=1}^{n} \alpha_i y_i K(x, xi) + \beta_0.$$

Some popular kernels

- polynomial: $K(x, x') = (1 + \langle x, x' \rangle)^d$,
- radial basis: $K(x, x') = \exp(-\gamma \|x - x'\|^2)$,
- sigmoid: $\frac{1}{1+e^{-\langle x, x' \rangle}}$.

▶ sklearn: `svm.SVC` ▶ sklearn: `svm.SVR`

More details in [HTF09].

Outline

Remarks and tools for BCIs

Brain Computer Interfaces

- Difficulties with physiological data (e.g. EEG):
  - signal-to-noise ratio very low,
  - few small datasets (time/money consuming experiments),
  - high dimensionality,
  - non-stationary,
  - variability over humans (participants),
  - variability over time (sessions),
  - variability over experiments (settings).
- Different problems, increasing difficulty in prediction:
  - within-recording-session prediction (intra-session),
  - across-session within-subject prediction (intra-subject),
  - across-subject prediction (inter-subject).

Remarks and tools for BCIs

EEG tools and BCI evaluation

- mne.tools
- moabb.neurotechx.com

Outline

Machine Learning

- Learning from data: dataset $d_n = \{x_i\}_{i=1}^n \in \mathcal{X}^n$, with $dim(\mathcal{X}) = d$ (usually $\mathcal{X} = \mathbb{R}^d$). An element of d_n, $x_i \in \mathcal{X}$, is called a **sample**.

  Matrix representation of the dataset: $d_n = \mathbb{X} = \begin{pmatrix} x_{1,1} & \dots & x_{1,d} \\ x_{2,1} & \dots & x_{2,d} \\ \vdots & & \vdots \\ x_{n,1} & \dots & x_{n,d} \end{pmatrix}$.

- **Supervised learning**
  - Settings: each sample x_i is associated with a target $y_i \in \mathcal{Y}$.
  - Goal: compute a function (predictor) $c : \mathcal{X} \to \mathcal{Y}$ that predicts the target given a sample. Sample values are called **features**, and $\mathcal{X}$ the feature space.
  - Two main problems:
    classification ($\mathcal{Y}$ is finite, y_i are called **labels**),
    and **regression** ($\mathcal{Y} = \mathbb{R}$).

Machine Learning

- **Unsupervised Learning**
  - Settings: samples are not associated with any target.
  - Goal: anomaly/novelty detection, clustering, discovering structures in the data d_n.
- **Transfert Learning**
  - Settings: two different domains, *i.e.* two different couples (feature space, target space): source domain $(\mathcal{X}_S, \mathcal{Y}_S)$ and target domain $(\mathcal{X}_T, \mathcal{Y}_T)$.
  - Goal: Use a dataset from source domain to help making predictions in the target domain.
- **Semi-supervised Learning**
  - Settings: the samples are not always associated with target.
  - Goal: predict the target given a sample.
- **Self-supervised Learning**
  - Settings: the samples are associated with targets predicted by another predictor.
  - Goal: predict the target given a sample.

Machine Learning

- **Reinforcement Learning**
  - Settings: environment simulator returning an observation $s_t \in \mathcal{S}$, and a reward $r_t \in \mathbb{R}$ for each action $a_t \in \mathcal{A}$ sent to it.
  - Goal: compute an optimal strategy $\pi : \mathcal{S} \rightarrow \mathcal{A}$ by interacting with the simulator, *i.e.* maximizing the expectation of the sum of future rewards, *cf.* class of Planning (C.Chanel) and Reinforcement Learning (G.Angelotti).
- **Generative Models**
  - Settings: samples are not associated with any target.
  - Goal: generation of samples that could be in the dataset d_n *i.e.* that look like samples in the dataset.

In any of these Machine Learning problems, the computations on data, leading to desired outputs, are called **training**.

Principal Component Analysis

Consider the matrix representation of the dataset:

$$d_n = \mathbb{X} = \begin{pmatrix} x_{1,1} & \dots & x_{1,d} \\ x_{2,1} & \dots & x_{2,d} \\ \vdots & & \vdots \\ x_{n,1} & \dots & x_{n,d} \end{pmatrix} = \begin{pmatrix} x^{(1)} & \dots & x^{(d)} \end{pmatrix} \in \mathcal{M}_{n,d}(\mathbb{R}).$$

As a symetric matrix, $\mathbb{X}^T\mathbb{X} \in \mathcal{M}_d(\mathbb{R})$ diagonalizes in real orthonormal basis (Theorems 2.11 or 4.4 in [HMS20]):

$$\mathbb{X}^T\mathbb{X} = PDP^{-1},$$

with $D = Diag(\lambda_1, \dots, \lambda_d)$, the eigenvalues $\lambda_1 > \dots > \lambda_d > 0$ the associated othonormal basis $\left\{ v^{(i)} \right\}_{i=1}^{d}$, and $P = \begin{pmatrix} v^{(1)} & \dots & v^{(d)} \end{pmatrix} \in \mathcal{M}_d(\mathbb{R})$ an orthogonal matrix.

Principal Component Analysis

Suppose that each feature is centered: $\forall i \in \{1, \ldots, d\}$,

$$\overline{x^{(i)}} = \frac{1}{n} \sum_{k=1}^{n} x_k^{(i)} = \frac{1}{n} \sum_{k=1}^{n} x_{k,i} = 0.$$

The sample covariance of features i and $j \in \{1, \ldots, d\}$ is

$$\widehat{Cov}\left(x^{(i)}, x^{(j)}\right) = \frac{1}{n} \sum_{k=1}^{n} x_{k,i} x_{k,j} - \left(\frac{1}{n} \sum_{k=1}^{n} x_{k,i}\right) \left(\frac{1}{n} \sum_{k=1}^{n} x_{k,j}\right) = \frac{1}{n} \sum_{k=1}^{n} x_{k,i} x_{k,j} = \frac{1}{n} \left(\mathbb{X}^T \mathbb{X}\right)_{i,j}.$$

Principal Component Analysis

Consider the projection of the samples on the k^{th} eigenvector: $\mathbb{X}v^{(k)} = \begin{pmatrix} \langle x_1, v^{(k)} \rangle \\ \vdots \\ \langle x_n, v^{(k)} \rangle \end{pmatrix}$.

$$\overline{\mathbb{X}v^{(k)}} = \frac{1}{n}\sum_{i=1}^{n}\left(\mathbb{X}v^{(k)}\right)_i = \frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{d} x_{i,j}v_j^{(k)} = \sum_{j=1}^{d} v_j^{(k)}\overline{x^{(j)}} = 0,$$

thus the vector $\mathbb{X}v^{(k)}$ is centered. The sample coviance of $\mathbb{X}v^{(i)}$ and $\mathbb{X}v^{(j)}$ for $i \neq j$ is then

$$\widehat{Cov}\left(\mathbb{X}v^{(i)}, \mathbb{X}v^{(j)}\right) = \frac{1}{n}\sum_{k=1}^{n}\left(\mathbb{X}v^{(i)}\right)_k\left(\mathbb{X}v^{(j)}\right)_k = \frac{1}{n}\left(v^{(i)}\right)^T\mathbb{X}^T\mathbb{X}v^{(j)} = \lambda_j\left(v^{(i)}\right)^T v^{(j)} = 0,$$

since the eigenvector basis is orthogonal.

Principal Component Analysis

The sample coviance of $\mathbb{X}v^{(i)}$ and $\mathbb{X}v^{(j)}$ for $i \neq j$ is then

$$\widehat{Cov}\left(\mathbb{X}v^{(i)}, \mathbb{X}v^{(j)}\right) = \frac{1}{n}\sum_{k=1}^{n}\left(\mathbb{X}v^{(i)}\right)_k\left(\mathbb{X}v^{(j)}\right)_k = \frac{1}{n}\left(v^{(i)}\right)^T\mathbb{X}^T\mathbb{X}v^{(j)} = \lambda_j\left(v^{(i)}\right)^T v^{(j)} = 0,$$

since the eigenvector basis is orthogonal. The sample variance of $\mathbb{X}v^{(i)}$ is

$$\widehat{Var}\left(\mathbb{X}v^{(i)}\right) = \frac{1}{n}\left(\mathbb{X}v^{(i)}\right)^T\mathbb{X}v^{(i)} = \frac{1}{n}\left(v^{(i)}\right)^T\mathbb{X}^T\mathbb{X}v^{(i)} = \frac{\lambda_i}{n}\left(v^{(i)}\right)^T v^{(i)} = \frac{\lambda_i^2}{n},$$

since $\left\|v^{(i)}\right\|^2 = 1$.

Principal Component Analysis

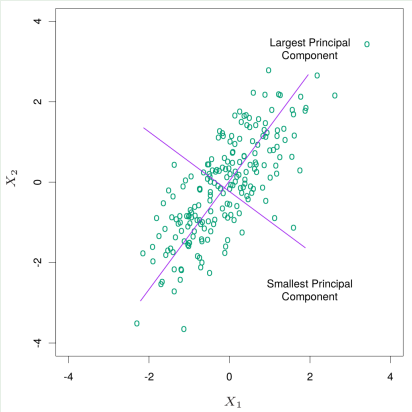
Thus, $\mathbb{X}v^{(1)}$ is the projection with the most variance.
Let $u = \sum_{i=1}^{d} u_i v^{(i)}$ another normalized vector. The sample variance of $\mathbb{X}u$ is

$$\begin{aligned}\widehat{Var}\left(\mathbb{X}u\right) &= \frac{1}{n}\left(\mathbb{X}u\right)^T \mathbb{X}u^{(i)} = \frac{1}{n}u^T\mathbb{X}^T\mathbb{X}u^{(i)} = \frac{1}{n}u^T\mathbb{X}^T\mathbb{X}\sum_{i=1}^{d} u_i v^{(i)} \\ &= \frac{1}{n}\sum_{i=1}^{d} u_i u^T\mathbb{X}^T\mathbb{X}v^{(i)} = \frac{1}{n}\sum_{i=1}^{d}\lambda_i u_i u^T v^{(i)} = \frac{1}{n}\sum_{i=1}^{d}\lambda_i u_i^2 \leqslant \frac{\lambda_1}{n},\end{aligned}$$

since $\|u\|^2 = 1$.

PCA: conclusion

- Projection on the first eigenvector $\mathbb{X}v^{(1)}$ is the **largest principal component** (direction with highest variance).
- Components $\mathbb{X}v^{(i)}$ have a null sample covariance.
- The sample variance of $\mathbb{X}v^{(i)}$ is $\frac{\lambda_i}{n}$.
- Directions $v^{(i)}$ are orthogonal.
- Useful for dimensionality reduction (take only the first principal components).



- sklearn: `decomposition.PCA`

Clustering

- Goal: partition the observations into groups ("clusters").
- Combinatorial algorithms
  - consider a number of clusters $k < n$.
  - look for a function $C : \{1, \ldots, n\} \to \{1, \ldots, k\}$.
  - where $C(i) \in \{1, \ldots, k\}$ is the cluster index of sample with index i.
  - minimize a loss based on a distance d between points:

$$L(C) = \frac{1}{2} \sum_{j=1}^{k} \sum_{C(i)=j} \sum_{C(i')=j} d(x_i, x_{i'})$$

Kmeans

- Given a cluster assignment C, compute the means of the vectors of each cluster j:

$$n_j = \sum_{i=1}^{n} \mathbb{1}_{\{C(i)=j\}},$$

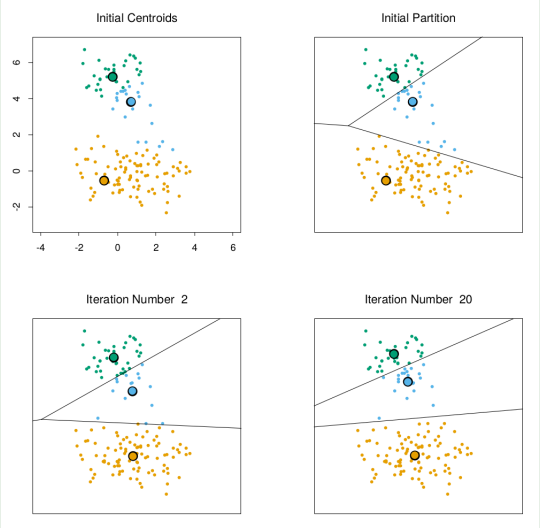
$$\mu_j = \frac{1}{n_j} \sum_{i=1}^{n} \mathbb{1}_{\{C(i)=j\}} x_i.$$

- Given a set of means $\{\mu_1, \ldots, \mu_k\}$, compute a cluster assignment C that minimizes

$$\sum_{j=1}^{k} n_j \sum_{C(i)=j} \|x_i - \mu_j\|^2.$$

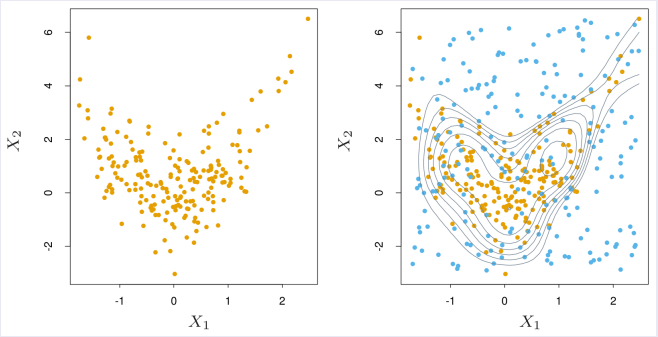
- Iteration until the assignments do not change.

Kmeans


Initial Centroids
Initial Partition
Iteration Number 2
Iteration Number 20
6
4
2
0
-2
-4
-2
0
2
4
6

Anomaly/Novelty detection

- 1-class classification
  (d_n is only "normal" data)
- Decision function
  $\sim$ estimation of the density
- Usual trick:
  - generate an artificial second class of"abnormal" data: sample uniformly data points over the rectangle containing the "normal" data.
  - use a classification algorithm on this binary classification problem.



Institut Supérieur de l'Aéronautique et de l'Espace

10 avenue Édouard Belin – BP 54032

31055 Toulouse Cedex 4 – France

Phone: +33 5 61 33 80 80

www.isae-supaero.fr

isae
Institut Supérieur de l'Aéronautique et de l'Espace
SUPAERO