The Introduction of Machine Learning and Some Methods of Supervised Classification

Sothea HAS
Sorbonne Université

LPSM, Université Paris-Diderot
sothea.has@lpsm.paris

May 20, 2019

Overview

Introduction

What's Machine Learning (ML)?

What's Machine Learning (ML)?

An informal definition by **Arthur Samuel** (1959):

"the field of study that gives computers the ability to learn without being explicitly programmed".

What is Machine Learning (ML)?

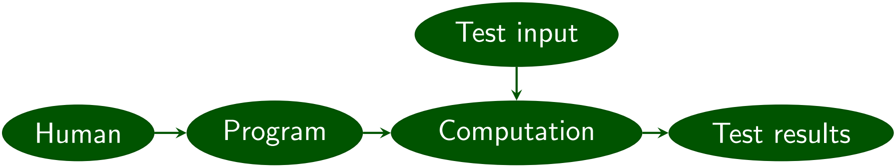
A more formal definition by **Tom M. Mitchell** (1997):

"A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E".

Traditional Programming vs Machine Learning

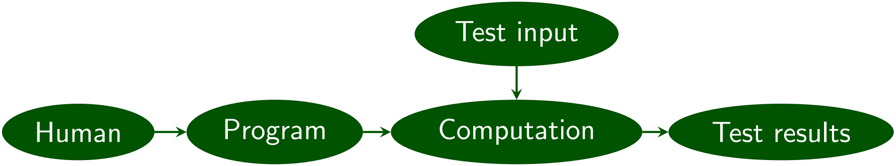
Traditional Programming vs Machine Learning

- Traditional programming:

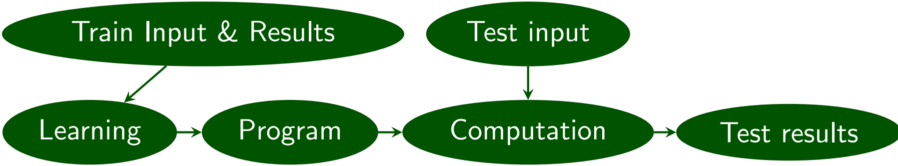


Traditional Programming vs Machine Learning

- Traditional programming:

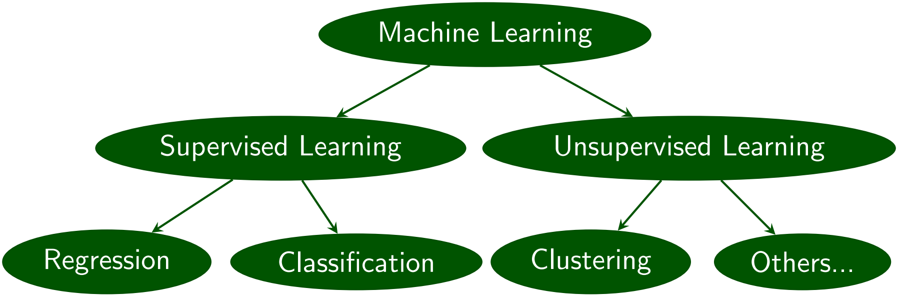


- Machine learning:



Branches of Machine Learning

Branches of Machine Learning



Predicting or explaining a variable using another group of variables.

Understanding the structure of a group of variables.

General Setting of Machine Learning

- Example:
  - $X = (Age, Weight, Height, Salary) \in \mathbb{R}^4$.
  - $Y =$ Size of the ring finger or a favorite sport among $\{$soccer, volleyball, basketball, boxing$\}$.
  - Classifying people according to X.

General Setting of Machine Learning

- Example:
  - $X = (Age, Weight, Height, Salary) \in \mathbb{R}^4$.
  - Y = Size of the ring finger or a favorite sport among {soccer, volleyball, basketball, boxing}.
  - Classifying people according to X.
- Supervised learning: learning a function $f : \mathcal{X} \to \mathcal{Y}$ such that

$$Y \approx f(X)$$

  where
  - Input or predictor: $X \in \mathcal{X} = \mathbb{R}^d$
  - Output or response variable: $Y \in \mathcal{Y} = \begin{cases} \mathbb{R} & : \text{Regression.} \\ \{1, 2, ..., K\} & : \text{Classification.} \end{cases}$

General Setting of Machine Learning

- Example:
  - $X = (Age, Weight, Height, Salary) \in \mathbb{R}^4$.
  - $Y =$ Size of the ring finger or a favorite sport among {soccer, volleyball, basketball, boxing}.
  - Classifying people according to X.
- Supervised learning: learning a function $f : \mathcal{X} \to \mathcal{Y}$ such that

$$Y \approx f(X)$$

  where
  - Input or predictor: $X \in \mathcal{X} = \mathbb{R}^d$
  - Output or response variable: $Y \in \mathcal{Y} = \begin{cases} \mathbb{R} & : \text{Regression.} \\ \{1, 2, ..., K\} & : \text{Classification.} \end{cases}$
- Unsupervised learning: **forget** Y and **focus only** on X.
  - Dimensional reduction.
  - Grouping or clustering structure...

Supervised Classification

Motivation Example: Spam Dataset

- Size: 4601×58.
  - Column 1st-48th: % of the corresponding words ($[0, 100]$).
  - Column 49st-54th: % of the corresponding characters ($[0, 100]$).
  - Column 55th: average length of uninterrupted sequences of capital letters (≥ 1).
  - Column 56th: length of longest uninterrupted sequence of capital letters ($\mathbb{N}$).
  - Column 57th: total number of capital letters in the e-mail ($\mathbb{N}$).
  - Column 58th: spam or non-spam.
- Available at: http://archive.ics.uci.edu/ml/machine-learning-databases/spambase/
- Objective: construct email spam filters based on this dataset.

$D_n =$

ID	**make**	...	**charSemicolon**	**...**	**capitalTotal**	**type**
1	0	...	0	...	278	spam
2	0.21	...	0	...	1028	spam
...	...	...	...	...	...	...
4601	0	...	0	...	40	non-spam

Problem Formulation

Problem Formulation

- Input-output: $(X, Y) \in \mathcal{X} \times \mathcal{Y} = \mathbb{R}^{57} \times \{0, 1\}$

Problem Formulation

- Input-output: $(X, Y) \in \mathcal{X} \times \mathcal{Y} = \mathbb{R}^{57} \times \{0, 1\}$ where output $Y = \begin{cases} 1, & \text{spam} \\ 0, & \text{non-spam} \end{cases}$
- Find a classifier $f : \mathbb{R}^{57} \to \{0, 1\}$ minimizing the following misclassification error:

$$\mathcal{R}(f) = \mathbb{E}[\mathbb{1}_{\{f(X) \neq Y\}}] = \mathbb{P}[f(X) \neq Y]$$

Some Well-known Results on Supervised Classification

Some Well-known Results on Supervised Classification

Some Well-known Results on Supervised Classification

- $\eta(x) = \mathbb{E}[Y|X = x] = \mathbb{P}(Y = 1|X = x)$.

Some Well-known Results on Supervised Classification

- $\eta(x) = \mathbb{E}[Y|X = x] = \mathbb{P}(Y = 1|X = x)$.
- A classifier g is optimal or known as Bayes classifier if

$$\mathcal{R}(g) = \inf_{f:\mathcal{X}\to\{0,1\}} \mathcal{R}(f) = \inf_{f:\mathcal{X}\to\{0,1\}} \mathbb{P}[f(X) \neq Y] = \mathcal{R}^*$$

Some Well-known Results on Supervised Classification

- $\eta(x) = \mathbb{E}[Y|X = x] = \mathbb{P}(Y = 1|X = x)$.
- A classifier g is optimal or known as Bayes classifier if

$$\mathcal{R}(g) = \inf_{f:\mathcal{X}\to\{0,1\}} \mathcal{R}(f) = \inf_{f:\mathcal{X}\to\{0,1\}} \mathbb{P}[f(X) \neq Y] = \mathcal{R}^*$$

- Bayes classifier [Devroye et al., 1997] is defined by,

$$g(x) = \begin{cases} 1, & \text{if } \eta(x) \geq 0.5 \\ 0, & \text{Otherwise} \end{cases}$$

Some Well-known Results on Supervised Classification

- $\eta(x) = \mathbb{E}[Y|X = x] = \mathbb{P}(Y = 1|X = x)$.
- A classifier g is optimal or known as Bayes classifier if

$$\mathcal{R}(g) = \inf_{f:\mathcal{X}\to\{0,1\}} \mathcal{R}(f) = \inf_{f:\mathcal{X}\to\{0,1\}} \mathbb{P}[f(X) \neq Y] = \mathcal{R}^*$$

- Bayes classifier [Devroye et al., 1997] is defined by,

$$g(x) = \begin{cases} 1, & \text{if } \eta(x) \geq 0.5 \\ 0, & \text{Otherwise} \end{cases}$$

- Looks easy right?

Some Well-known Results on Supervised Classification

- $\eta(x) = \mathbb{E}[Y|X = x] = \mathbb{P}(Y = 1|X = x)$.
- A classifier g is optimal or known as Bayes classifier if

$$\mathcal{R}(g) = \inf_{f:\mathcal{X}\to\{0,1\}} \mathcal{R}(f) = \inf_{f:\mathcal{X}\to\{0,1\}} \mathbb{P}[f(X) \neq Y] = \mathcal{R}^*$$

- Bayes classifier [Devroye et al., 1997] is defined by,

$$g(x) = \begin{cases} 1, & \text{if } \eta(x) \geq 0.5 \\ 0, & \text{Otherwise} \end{cases}$$

- Looks easy right? But wait!

Some Well-known Results on Supervised Classification

- $\eta(x) = \mathbb{E}[Y|X = x] = \mathbb{P}(Y = 1|X = x)$.
- A classifier g is optimal or known as Bayes classifier if

$$\mathcal{R}(g) = \inf_{f:\mathcal{X}\to\{0,1\}} \mathcal{R}(f) = \inf_{f:\mathcal{X}\to\{0,1\}} \mathbb{P}[f(X) \neq Y] = \mathcal{R}^*$$

- Bayes classifier [Devroye et al., 1997] is defined by,

$$g(x) = \begin{cases} 1, & \text{if } \eta(x) \geq 0.5 \\ 0, & \text{Otherwise} \end{cases}$$

- Looks easy right? But wait!
- Look at this guy: $\eta(x) = \mathbb{P}(Y = 1|X = x)$.

Some Well-known Results on Supervised Classification

- $\eta(x) = \mathbb{E}[Y|X = x] = \mathbb{P}(Y = 1|X = x)$.
- A classifier g is optimal or known as Bayes classifier if

$$\mathcal{R}(g) = \inf_{f:\mathcal{X}\to\{0,1\}} \mathcal{R}(f) = \inf_{f:\mathcal{X}\to\{0,1\}} \mathbb{P}[f(X) \neq Y] = \mathcal{R}^*$$

- Bayes classifier [Devroye et al., 1997] is defined by,

$$g(x) = \begin{cases} 1, & \text{if } \eta(x) \geq 0.5 \\ 0, & \text{Otherwise} \end{cases}$$

- Looks easy right? But wait!
- Look at this guy: $\eta(x) = \mathbb{P}(Y = 1|X = x)$.
- In practice, η is intractable!

Empirical Setting

Empirical Setting

- Suppose all the observations (emails) $D_n = \{(x_i, y_i)_{i=1}^n\}$ $(n = 4601, d = 57)$ are *iid* copies of (X, Y).

$D_n =$

ID	$x^{(1)}$	$x^{(2)}$	...	$x^{(d)}$	y
1	x_1^1	x_1^2	...	x_1^d	y_1
2	x_2^1	x_2^2	...	x_2^d	y_2
...	...	...	...	...	...
n	x_n^1	x_n^2	...	x_n^d	y_n

Empirical Setting

- Suppose all the observations (emails) $D_n = \{(x_i, y_i)_{i=1}^n\}$ $(n = 4601, d = 57)$ are *iid* copies of (X, Y).

$$D_n = \begin{array}{|c|cccc|c|}
\hline
ID & x^{(1)} & x^{(2)} & \dots & x^{(d)} & y \\
\hline
1 & x_1^1 & x_1^2 & \dots & x_1^d & y_1 \\
\hline
2 & x_2^1 & x_2^2 & \dots & x_2^d & y_2 \\
\hline
\dots & \dots & \dots & \dots & \dots & \dots \\
\hline
n & x_n^1 & x_n^2 & \dots & x_n^d & y_n \\
\hline
\end{array}$$

- Empirical misclassification error:

$$\mathcal{R}_n(f) = \frac{1}{n}\sum_{i=1}^{n} \mathbb{1}_{\{f(x_i)\neq y_i\}}$$

Empirical Setting

- Suppose all the observations (emails) $D_n = \{(x_i, y_i)_{i=1}^n\}$ $(n = 4601, d = 57)$ are *iid* copies of (X, Y).

$D_n =$

ID	$x^{(1)}$	$x^{(2)}$	...	$x^{(d)}$	y
1	x_1^1	x_1^2	...	x_1^d	y_1
2	x_2^1	x_2^2	...	x_2^d	y_2
...	...	...	...	...	...
n	x_n^1	x_n^2	...	x_n^d	y_n

- Empirical misclassification error:

$$\mathcal{R}_n(f) = \frac{1}{n}\sum_{i=1}^{n} \mathbb{1}_{\{f(x_i)\neq y_i\}}$$

- Objective: finding a data-based classifier f_n minimizing the empirical error of the **testing set** (new unseen observation).

k-Nearest Neighbors Classifier

k-Nearest Neighbors Classifier (k-NN)

- $d(x, y) = \|x - y\|_2^2$ defined on $\mathcal{X}(= \mathbb{R}^{57})$.

k-Nearest Neighbors Classifier (k-NN)

- $d(x, y) = \|x - y\|_2^2$ defined on $\mathcal{X}(= \mathbb{R}^{57})$.
- A new observation (email) x, define $N_k(x) = \{x_{(i)}\}_{i=1}^{k}$ i.e.,

$$d(x, x_{(1)}) \leq d(x, x_{(2)}) \leq ... \leq d(x, x_{(k)})$$

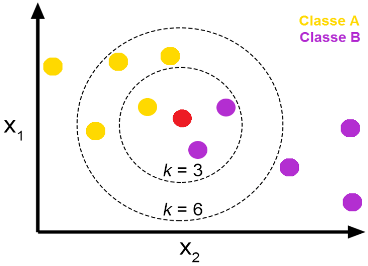
k-Nearest Neighbors Classifier (k-NN)

- $d(x, y) = \|x - y\|_2^2$ defined on $\mathcal{X}(= \mathbb{R}^{57})$.
- A new observation (email) x, define $N_k(x) = \{x_{(i)}\}_{i=1}^{k}$ i.e.,

$$d(x, x_{(1)}) \leq d(x, x_{(2)}) \leq \ldots \leq d(x, x_{(k)})$$

- The prediction of x by k-NN classifier = **majority class** among $\{y_{(i)}\}_{i=1}^{k}$.

$$k\text{-NN}(x) = \begin{cases} 1, & \hat{\eta}(x) \geq 0.5 \\ 0, & \text{Otherwise} \end{cases} \text{ where } \hat{\eta}(x) = \frac{1}{k}\sum_{i=1}^{k} y_{(i)}$$



1-NN

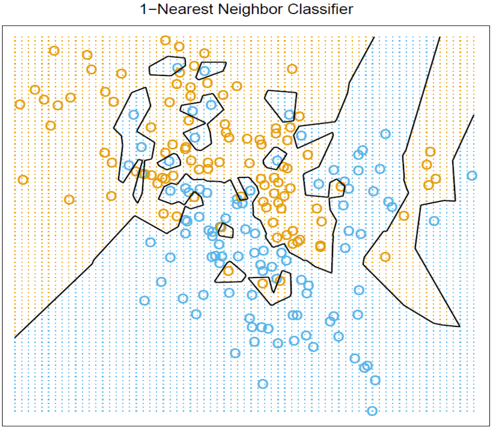


Figure: An example taken from [Hastie et al., 2009]

15-NN

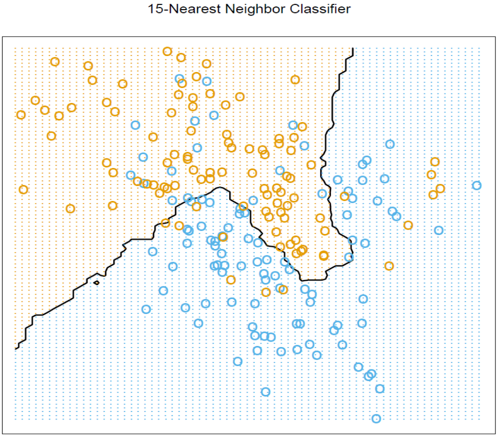


Figure: An example taken from [Hastie et al., 2009]

Cross-Validation Technique

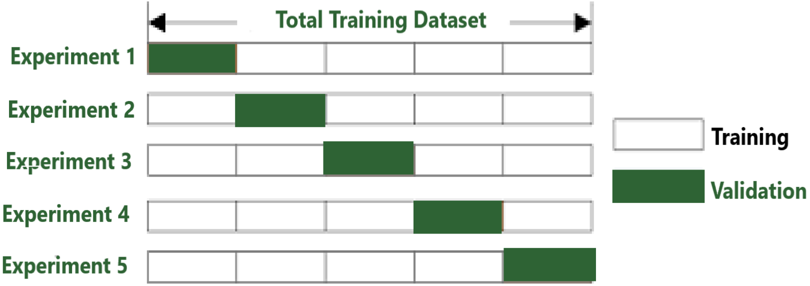


Figure: Representation of 5-folds cross-validation.

Choosing k using Cross-Validation Technique

K-folds cross-validation:

1. Randomly split D_n into K folds.
2. for $j = 1, 2, ..., k_{\text{max}}$:

   for $i = 1, 2, ..., K$:

   - Validation (testing) set: D_{n_i}.
   - Training set: the remaining ones $(D_n - D_{n_i})$.
   - Compute the **error**:

   $$\mathcal{R}_j^i = \frac{1}{|D_{n_i}|} \sum_{x_\ell \in D_{n_i}} \mathbb{1}_{\{j\text{-NN}(x_\ell) \neq y_\ell\}}$$

   Compute: $\mathcal{R}_j = \frac{1}{K} \sum_{i=1}^{K} \mathcal{R}_j^i$
3. $k = \text{argmin}_{1 \leq j \leq k_{\text{max}}} \mathcal{R}_j$

Choosing k using Cross-Validation Technique

K-folds cross-validation:

1. Randomly split D_n into K folds.
2. **for** $j = 1, 2, ..., k_{\text{max}}$:
   for $i = 1, 2, ..., K$:
   - Validation (testing) set: D_{n_i}.
   - Training set: the remaining ones $(D_n - D_{n_i})$.
   - Compute the **error**:

$$\mathcal{R}_j^i = \frac{1}{|D_{n_i}|} \sum_{x_\ell \in D_{n_i}} \mathbb{1}_{\{j\text{-NN}(x_\ell) \neq y_\ell\}}$$

   Compute: $\mathcal{R}_j = \frac{1}{K} \sum_{i=1}^{K} \mathcal{R}_j^i$
3. $k = \text{argmin}_{1 \leq j \leq k_{\text{max}}} \mathcal{R}_j$

Remark:

- X should be renormalized to erase the influence of the units.

Choosing k using Cross-Validation Technique

K-folds cross-validation:

1. Randomly split D_n into K folds.
2. for $j = 1, 2, ..., k_{\max}$:
   for $i = 1, 2, ..., K$:
   - Validation (testing) set: D_{n_i}.
   - Training set: the remaining ones $(D_n - D_{n_i})$.
   - Compute the **error**:

$$\mathcal{R}_j^i = \frac{1}{|D_{n_i}|} \sum_{x_\ell \in D_{n_i}} \mathbb{1}_{\{j\text{-NN}(x_\ell) \neq y_\ell\}}$$

   Compute: $\mathcal{R}_j = \frac{1}{K} \sum_{i=1}^{K} \mathcal{R}_j^i$
3. $k = \operatorname{argmin}_{1 \leq j \leq k_{\max}} \mathcal{R}_j$

Remark:

- X should be renormalized to erase the influence of the units.
- Other options of d.

Linear & Quadratic Discriminant Analysis

Review about Gaussian Distribution

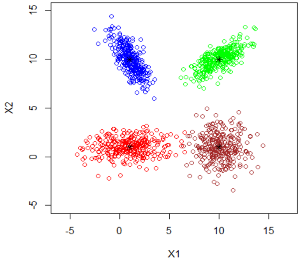
$X \sim \mathcal{N}_d(\mu, \Sigma)$ if it has the following density:

$$\phi(x; \mu, \Sigma) = \frac{1}{(2\pi)^{d/2}|\Sigma|^{1/2}} \exp\left[-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\right]$$

Example:

$$\mu_1 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \Sigma_1 = \begin{bmatrix} 4 & 0.1 \\ 0.1 & 1 \end{bmatrix} \qquad \mu_2 = \begin{bmatrix} 1 \\ 10 \end{bmatrix}, \Sigma_2 = \begin{bmatrix} 1 & -1 \\ -1 & 2 \end{bmatrix}$$

$$\mu_3 = \begin{bmatrix} 10 \\ 10 \end{bmatrix}, \Sigma_3 = \begin{bmatrix} 2 & 1 \\ 1 & 1 \end{bmatrix} \qquad \mu_4 = \begin{bmatrix} 10 \\ 1 \end{bmatrix}, \Sigma_4 = \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix}$$



Linear & Quadratic Discriminant Analysis

Recall that $\eta(x) = \mathbb{P}(Y = 1|X = x)$.

Linear & Quadratic Discriminant Analysis

Recall that $\eta(x) = \mathbb{P}(Y = 1|X = x)$.

- $\mathbb{P}(Y = k|X = x)$: Posterior probability ($k \in \{0, 1\}$).
- $p_k = \mathbb{P}(Y = k)$: Prior probability.
- $f_k(x) = \mathbb{P}(X = x|Y = k)$: class-conditional probability.

Linear & Quadratic Discriminant Analysis

Recall that $\eta(x) = \mathbb{P}(Y = 1|X = x)$.

- $\mathbb{P}(Y = k|X = x)$: Posterior probability ($k \in \{0, 1\}$).
- $p_k = \mathbb{P}(Y = k)$: Prior probability.
- $f_k(x) = \mathbb{P}(X = x|Y = k)$: class-conditional probability.

Bayes's formula:

$$\mathbb{P}(Y = k|X = x) = \frac{p_k f_k(x)}{p_0 f_0(x) + p_1 f_1(x)}, k \in \{0, 1\}$$

Linear & Quadratic Discriminant Analysis

Recall that $\eta(x) = \mathbb{P}(Y = 1|X = x)$.

- $\mathbb{P}(Y = k|X = x)$: Posterior probability ($k \in \{0, 1\}$).
- $p_k = \mathbb{P}(Y = k)$: Prior probability.
- $f_k(x) = \mathbb{P}(X = x|Y = k)$: class-conditional probability.

Bayes's formula:

$$\mathbb{P}(Y = k|X = x) = \frac{p_k f_k(x)}{p_0 f_0(x) + p_1 f_1(x)}, k \in \{0, 1\}$$

(H_1) Gaussian hypothesis:

$$f_k(x) = \phi(x; \mu_k, \Sigma_k) = \frac{1}{(2\pi)^{d/2}|\Sigma_k|^{1/2}} \exp\left[-\frac{1}{2}(x - \mu_k)^T \Sigma_k^{-1}(x - \mu_k)\right]$$

LDA & QDA

In general (K classes):

LDA & QDA

In general (K classes):

Maximizing $\mathbb{P}(Y = k|X = x)$

LDA & QDA

In general (K classes):

Maximizing $\mathbb{P}(Y = k|X = x) \Leftrightarrow$ maximizing $p_k f_k(x)$

LDA & QDA

In general (K classes):

$$\begin{aligned}\text{Maximizing } \mathbb{P}(Y=k|X=x) &\Leftrightarrow \text{maximizing } p_k f_k(x)\\ &\Leftrightarrow \text{maximizing } \log(p_k f_k(x))\end{aligned}$$

LDA & QDA

In general (K **classes)**:

$$\text{Maximizing } \mathbb{P}(Y = k|X = x) \Leftrightarrow \text{maximizing } p_k f_k(x)$$
$$\Leftrightarrow \text{maximizing } \log(p_k f_k(x))$$
$$\Leftrightarrow \text{maximizing:}$$

$$\delta_k^{(q)}(x) = -\frac{1}{2}\log(|\Sigma_k|) - \frac{1}{2}(x - \mu_k)^T \Sigma_k^{-1}(x - \mu_k) + \log(p_k) \quad (QDA)$$

LDA & QDA

In general (K classes):

Maximizing $\mathbb{P}(Y = k | X = x) \Leftrightarrow$ maximizing $p_k f_k(x)$

$\Leftrightarrow$ maximizing $\log(p_k f_k(x))$

$\Leftrightarrow$ maximizing:

$$\delta_k^{(q)}(x) = -\frac{1}{2}\log(|\Sigma_k|) - \frac{1}{2}(x - \mu_k)^T \Sigma_k^{-1}(x - \mu_k) + \log(p_k) \quad (QDA)$$

(H_2) Homoscedasticity hypothesis:

$$\Sigma_k = \Sigma, \forall k$$

LDA & QDA

In general (K **classes)**:

Maximizing $\mathbb{P}(Y=k|X=x) \Leftrightarrow$ maximizing $p_k f_k(x)$

$\Leftrightarrow$ maximizing $\log(p_k f_k(x))$

$\Leftrightarrow$ maximizing:

$$\delta_k^{(q)}(x) = -\frac{1}{2}\log(|\Sigma_k|) - \frac{1}{2}(x-\mu_k)^T\Sigma_k^{-1}(x-\mu_k) + \log(p_k) \quad (QDA)$$

(H_2 **) Homoscedasticity hypothesis**:

$$\Sigma_k = \Sigma, \forall k$$

Thus, we maximize:

$$\delta_k^{(\ell)}(x) = x^T\Sigma^{-1}\mu_k - \frac{1}{2}\mu_k^T\Sigma^{-1}\mu_k + \log(p_k) \quad (LDA)$$

LDA & QDA

In general (K classes):

Maximizing $\mathbb{P}(Y = k|X = x) \Leftrightarrow$ maximizing $p_k f_k(x)$

$\Leftrightarrow$ maximizing $\log(p_k f_k(x))$

$\Leftrightarrow$ maximizing:

$$\delta_k^{(q)}(x) = -\frac{1}{2}\log(|\Sigma_k|) - \frac{1}{2}(x-\mu_k)^T\Sigma_k^{-1}(x-\mu_k) + \log(p_k) \quad (QDA)$$

(H_2) Homoscedasticity hypothesis:

$$\Sigma_k = \Sigma, \forall k$$

Thus, we maximize:

$$\delta_k^{(\ell)}(x) = x^T\Sigma^{-1}\mu_k - \frac{1}{2}\mu_k^T\Sigma^{-1}\mu_k + \log(p_k) \quad (LDA)$$

Particularly,

$$x \text{ is a spam iff } \delta_1^{(\ell)}(x) > \delta_0^{(\ell)}(x) \qquad (LDA)$$

$$x \text{ is a spam iff } \delta_1^{(q)}(x) > \delta_0^{(q)}(x) \qquad (QDA)$$

LDA & QDA

In practice (K classes):

$$\delta_k^{(\ell)}(x) = x^T\Sigma^{-1}\mu_k - \frac{1}{2}\mu_k^T\Sigma^{-1}\mu_k + \log(p_k) \qquad (LDA)$$

$$\delta_k^{(q)}(x) = -\frac{1}{2}\log(|\Sigma_k|) - \frac{1}{2}(x-\mu_k)^T\Sigma_k^{-1}(x-\mu_k) + \log(p_k) \qquad (QDA)$$

LDA & QDA

In practice (K classes):

$$\delta_k^{(\ell)}(x) = x^T\Sigma^{-1}\mu_k - \frac{1}{2}\mu_k^T\Sigma^{-1}\mu_k + \log(p_k) \qquad (LDA)$$

$$\delta_k^{(q)}(x) = -\frac{1}{2}\log(|\Sigma_k|) - \frac{1}{2}(x-\mu_k)^T\Sigma_k^{-1}(x-\mu_k) + \log(p_k) \qquad (QDA)$$

We estimate:

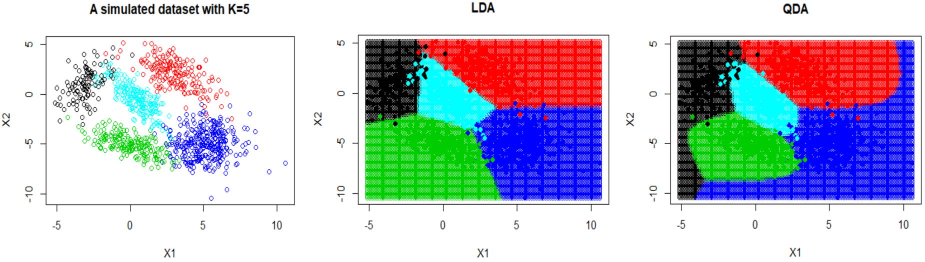
$\hat{p}_k = n_k/n$ where n_k is the number of points in class k

$$\hat{\mu}_k = \frac{1}{n_k}\sum_{y_i=k} x_i$$

$$\hat{\Sigma} = \frac{1}{n-K}\sum_{k=1}^{K}\sum_{y_i=k}(x_i-\mu_k)(x_i-\mu_k)^T \qquad (LDA)$$

$$\hat{\Sigma}_k = \frac{1}{n-1}\sum_{y_i=k}(x_i-\mu_k)(x_i-\mu_k)^T \qquad (QDA)$$

LDA & QDA



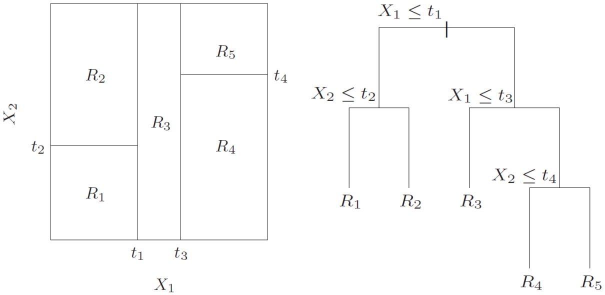
Misclassification error: 0.045 0.04

Classification Trees

Classification Trees

Tree Principle:

- Construct a recursive partition of D_n by splitting at each time a certain **variable** at a certain **value**.
- Prediction $=$ majority vote.



Classification Trees

Remarks:

- Easy to interpret.
- Quality of prediction depends on the structure of the tree.
- Issue: finding an optimal tree is hard!

Classification Trees

Remarks:

- Easy to interpret.
- Quality of prediction depends on the structure of the tree.
- Issue: finding an optimal tree is hard!

Construction:

- Branching or growing (greedy top-down approach).
- Pruning (bottom-up approach).

Classification Trees

Remarks:

- Easy to interpret.
- Quality of prediction depends on the structure of the tree.
- Issue: finding an optimal tree is hard!

Construction:

- Branching or growing (greedy top-down approach).
- Pruning (bottom-up approach).

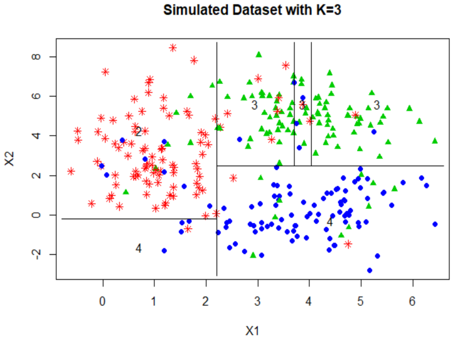
Available algorithms:

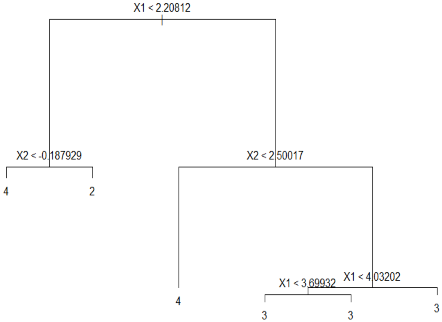
- ID3 (Iterative Dichotomiser 3).
- C4.5 (successor of ID3).
- CART (Classification And Regression Tree).
- Others...

Branching or Growing a Tree

Branching (top-down):

- Start from the root: D_n.
- Recursively split those regions along a certain **variable** at a certain **value**.
- Split so that the two parts are as **homogeneous** or **pure** as possible.





Branching or Growing a Tree

At the jth split:

- R_j: the region to be split.
- $\hat{p}_j^k = \frac{1}{|R_j|}\sum_{x_i \in R_j} \mathbb{1}_{\{y_i = k\}}$: probability of class k in R_j.
- $k^* = k_j^* = \text{argmax}_{1 \leq k \leq K}\, \hat{p}_j^k$: majority class of region R_j.

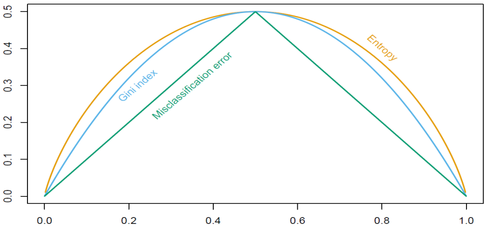
Branching or Growing a Tree

At the jth split:

- R_j: the region to be split.
- $\hat{p}_j^k = \frac{1}{|R_j|}\sum_{x_i \in R_j} \mathbb{1}_{\{y_i = k\}}$: probability of class k in R_j.
- $k^* = k_j^* = \text{argmax}_{1 \leq k \leq K}\, \hat{p}_j^k$: majority class of region R_j.

Splitting criteria (impurity measures):

- Misclassification error: $\frac{1}{|R_j|}\sum_{x_i \in R_j} \mathbb{1}_{\{y_i \neq k_j^*\}} = 1 - \hat{p}_j^{k^*}$.
- Gini index: $\sum_{k \neq k'} \hat{p}_j^k \hat{p}_j^{k'} = \sum_{k=1}^K \hat{p}_j^k (1 - \hat{p}_j^k)$.
- Cross-entropy or deviance: $-\sum_{k=1}^K \hat{p}_j^k \log(\hat{p}_j^k)$.



Branching or Growing a Tree

Splitting procedure:

- Search for $x^{(i)}$ and t_j minimizing one of these criteria.
- Split R_j into two parts: $R_j^{(1)} = \{x : x^{(i)} \leq t_j\}$ and $R_j^{(2)} = \{x : x^{(i)} > t_j\}$.
- Continue until a stopping criterion is met.

Branching or Growing a Tree

Splitting procedure:

- Search for $x^{(i)}$ and t_j minimizing one of these criteria.
- Split R_j into two parts: $R_j^{(1)} = \{x : x^{(i)} \leq t_j\}$ and $R_j^{(2)} = \{x : x^{(i)} > t_j\}$.
- Continue until a stopping criterion is met.

Stopping criteria:

- Depth of the tree.
- Minimum number of points in a region.

Branching or Growing a Tree

Splitting procedure:

- Search for $x^{(i)}$ and t_j minimizing one of these criteria.
- Split R_j into two parts: $R_j^{(1)} = \{x : x^{(i)} \leq t_j\}$ and $R_j^{(2)} = \{x : x^{(i)} > t_j\}$.
- Continue until a stopping criterion is met.

Stopping criteria:

- Depth of the tree.
- Minimum number of points in a region.

Construction aspect:

- Too complex tree may lead to **over-fitting**.
- Too simple tree might be too **weak** for prediction.

Pruning a Tree

Pruning (bottom-up):

- Aim to find a more effectively simple subtree from a given complex tree.
- Merging or collapsing nodes to shorten the tree.
- Based on **dynamic programming** principle.

Pruning aTtree

Cost complexity for a tree T:

$$C_\lambda(T) = \sum_{j=1}^{|T|} n_j Q_j(T) + \lambda |T|$$

Pruning aTtree

Cost complexity for a tree T:

$$C_\lambda(T) = \sum_{j=1}^{|T|} n_j Q_j(T) + \lambda|T|$$

where

- n_j: number of points in region R_j.

Pruning aTtree

Cost complexity for a tree T:

$$C_\lambda(T) = \sum_{j=1}^{|T|} n_j Q_j(T) + \lambda|T|$$

where

- n_j: number of points in region R_j.
- $Q_j(T)$: either of the splitting criteria.

Pruning aTtree

Cost complexity for a tree T:

$$C_\lambda(T) = \sum_{j=1}^{|T|} n_j Q_j(T) + \lambda |T|$$

where

- n_j: number of points in region R_j.
- $Q_j(T)$: either of the splitting criteria.
- $|T|$: number of nodes or leaves of T.

Pruning aTtree

Cost complexity for a tree T:

$$C_\lambda(T) = \sum_{j=1}^{|T|} n_j Q_j(T) + \lambda|T|$$

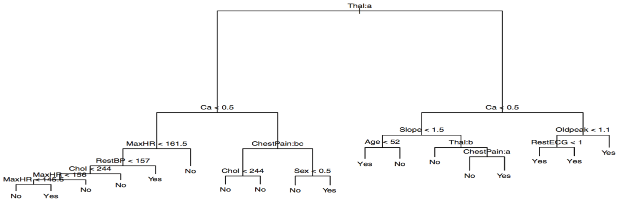
where

- n_j: number of points in region R_j.
- $Q_j(T)$: either of the splitting criteria.
- $|T|$: number of nodes or leaves of T.
- λ: tuning parameter (trad-off between the size of the tree and goodness of fit to the data).

Pruning a Tree

Pruning procedure for a given λ:

1. Start with a complex tree T_0.
   for $j = 1, 2, \dots, |T_0| - 1$:
   - Weakest link pruning: subtree of size $|T_0| - j$.
   - Choose T_j with the **smallest per-node increase** in $\sum_{i=1}^{|T_0|-j} n_i Q_i(T)$.
2. We produce a finite sequence of trees: $T_0 \supset T_1 \supset \dots \supset T_{|T_0|-1}$.
3. Pick the one minimizes $C_\lambda(T)$.

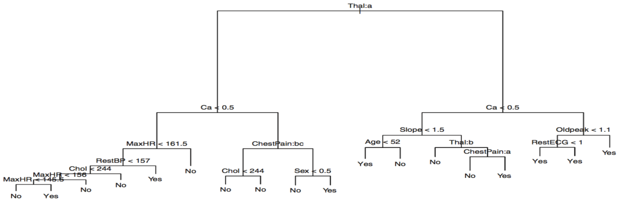


(See, for example, [Breiman et al., 1984] and [Hastie et al., 2009])

Pruning a Tree

Pruning procedure for a given λ:

1. Start with a complex tree T_0.
   for $j = 1, 2, ..., |T_0| - 1$:
   - Weakest link pruning: subtree of size $|T_0| - j$.
   - Choose T_j with the **smallest per-node increase** in $\sum_{i=1}^{|T_0|-j} n_i Q_i(T)$.
2. We produce a finite sequence of trees: $T_0 \supset T_1 \supset ... \supset T_{|T_0|-1}$.
3. Pick the one minimizes $C_\lambda(T)$.

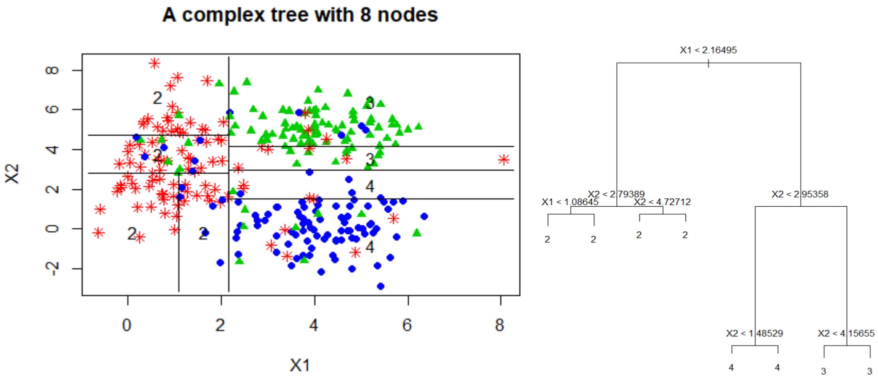


(See, for example, [Breiman et al., 1984] and [Hastie et al., 2009])

Remark: λ is chosen using cross-validation technique.

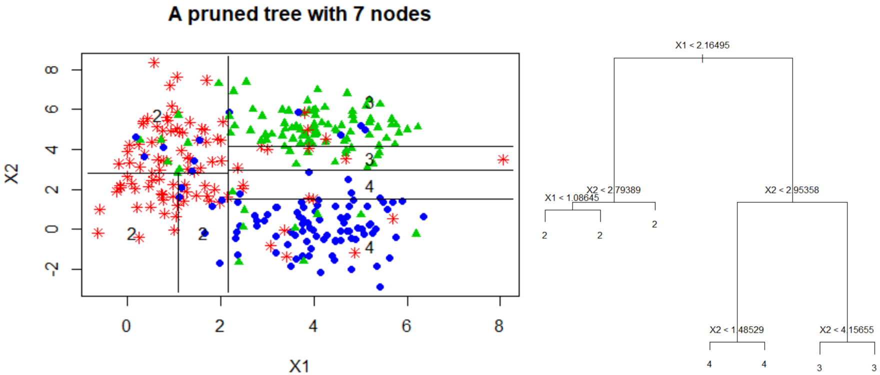
An Example of the Procedure

A complex tree T_0



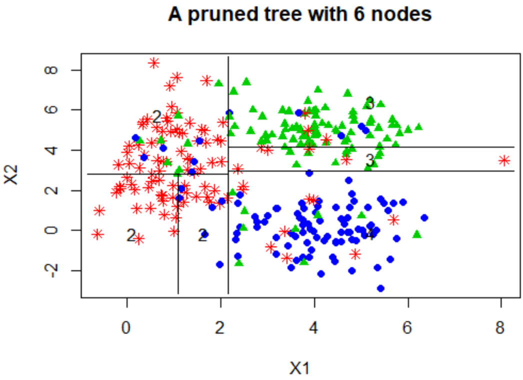
An Example of the Procedure

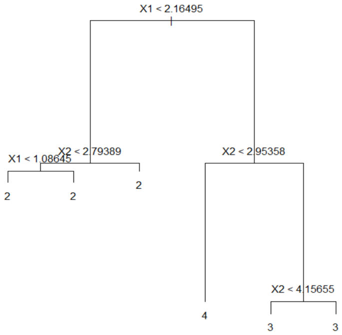
A subtree $T_1 \subset T_0$



An Example of the Procedure

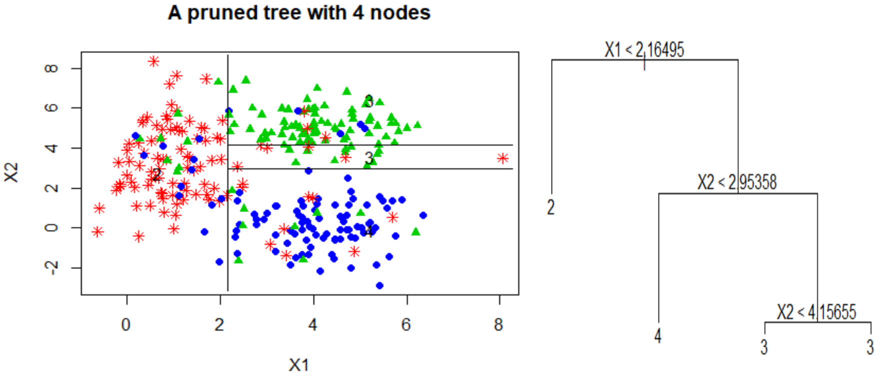
A subtree $T_2 \subset T_1 \subset T_0$





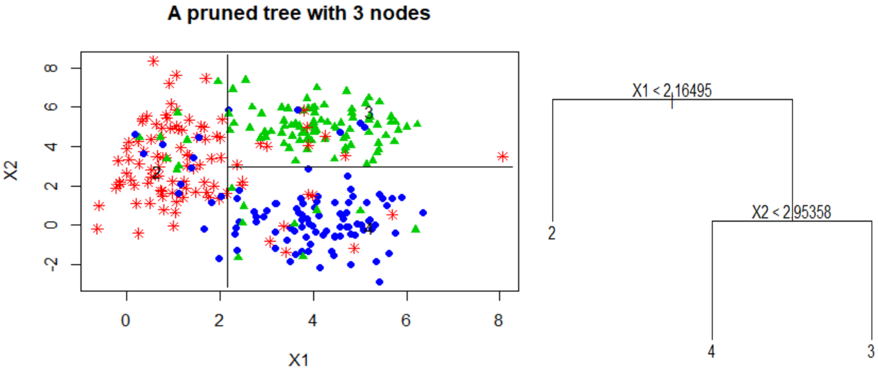
An Example of the Procedure

A subtree $T_4 \subset T_3 \subset T_2 \subset T_1 \subset T_0$



An Example of the Procedure

A subtree $T_5 \subset T_4 \subset T_3 \subset T_2 \subset T_1 \subset T_0$



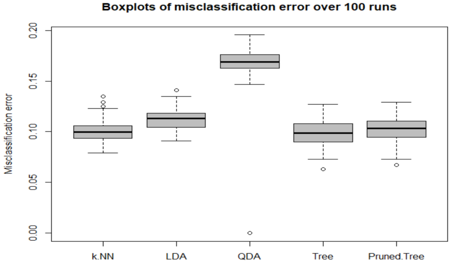
Application

Numerical Results: `Spam` Dataset

Spam dataset

- $D_n \in \mathbb{R}^{4601\times 58}$ with $(x_i, y_i) \in \mathbb{R}^{57} \times \{1, 0\}$ (spam or not).
- All methods are performed using `R` program via `R-studio` available at: https://www.rstudio.com/.
- Dataset available at: http://archive.ics.uci.edu/ml/machine-learning-databases/spambase/.
- Details in practical session (tomorrow afternoon)!

Numerical Results: Spam Dataset



Average	k-NN	LDA	QDA	Tree	Pruned Tree
Error	0.10103	0.11212	0.16785	**0.09953**	0.10257
SD	0.01074	0.00993	0.01919	0.01197	0.01250

Table: Average miscalssification errors and standard errors over 100 runs.

Summary and Further Methods

Conclusion and Further Methods

Summary:

Conclusion and Further Methods

Summary:

- What is Machine Learning.

Conclusion and Further Methods

Summary:

- What is Machine Learning.
- Definition, general setting and branches of ML.

Conclusion and Further Methods

Summary:

- What is Machine Learning.
- Definition, general setting and branches of ML.
- Three methods of supervised classification.

Conclusion and Further Methods

Summary:

- What is Machine Learning.
- Definition, general setting and branches of ML.
- Three methods of supervised classification.
- Classification tree and k-NN seem to be the most preferable ones on the `Spam` dataset.

Conclusion and Further Methods

Summary:

- What is Machine Learning.
- Definition, general setting and branches of ML.
- Three methods of supervised classification.
- Classification tree and k-NN seem to be the most preferable ones on the `Spam` dataset.

Further (ensemble learning) methods:

Conclusion and Further Methods

Summary:

- What is Machine Learning.
- Definition, general setting and branches of ML.
- Three methods of supervised classification.
- Classification tree and k-NN seem to be the most preferable ones on the `Spam` dataset.

Further (ensemble learning) methods:

- Bagging and Boosting.

Conclusion and Further Methods

Summary:

- What is Machine Learning.
- Definition, general setting and branches of ML.
- Three methods of supervised classification.
- Classification tree and k-NN seem to be the most preferable ones on the `Spam` dataset.

Further (ensemble learning) methods:

- Bagging and Boosting.
- Random forest.

Conclusion and Further Methods

Summary:

- What is Machine Learning.
- Definition, general setting and branches of ML.
- Three methods of supervised classification.
- Classification tree and k-NN seem to be the most preferable ones on the `Spam` dataset.

Further (ensemble learning) methods:

- Bagging and Boosting.
- Random forest.
- Neural networks...

References

Breiman, L., Friedman, J., Stone, C. J., and Olshen, R. (1984).
Classification and Regression Trees.
Wadsworth.

Devroye, L., Györfi, L., and Lugosi, G. (1997).
A Probabilistic Theory of Pattern Recognition.
Springer.

Györfi, L., Kohler, M., Krzyżak, A., and Walk, H. (2002).
A Distribution-Free Theory of Nonparametric Regression.
Springer.

Hastie, T., Robert Tibshirani, and Jerome Friedman (2009).
The Elements of Statistical Learning.
Springer.

Thank you

Question?