Introduction to Machine Learning

Eric Medvet

16/3/2017

Outline

Teachers

- Eric Medvet
  - Dipartimento di Ingegneria e Architettura (DIA)
  - `http://medvet.inginf.units.it/`

Section 1

Machine Learning: what and why?

What is Machine Learning?

Definition

Machine Learning is the science of getting computer to learn without being explicitly programmed.

Definition

Data Mining is the science of discovering patterns in data.

In practice

A set of mathematical and statistical tools for:

- building a model which allows to predict an output, given an input (*supervised learning*)
- learn relationships and structures in data (*unsupervised learning*)

Machine Learning everyday

Example problem: spam

Discriminate between spam and non-spam emails.

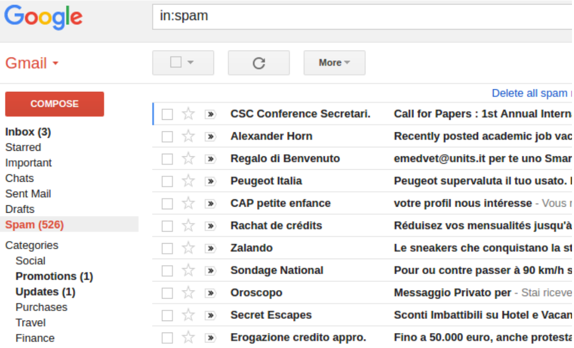


Figure: Spam filtering in Gmail.

Machine Learning everyday

Example problem: image understanding

Recognize objects in images.

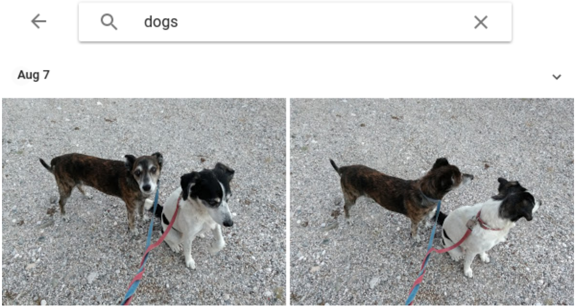


Figure: Object recognition in Google Photos.

Why ML/DM "today"?

- we collect more and more data (*big data*)
- we have more and more computational power

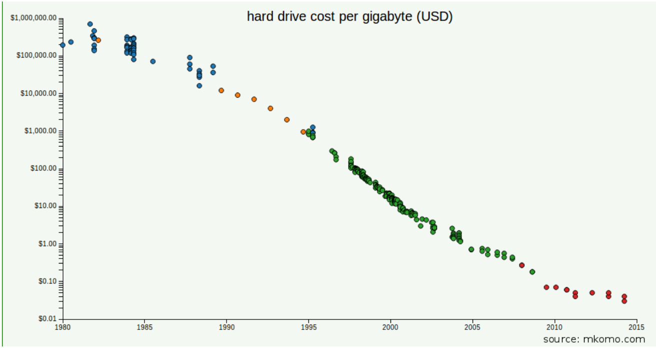


Figure: From `http://www.mkomo.com/cost-per-gigabyte-update`.

ML/DM is popular!

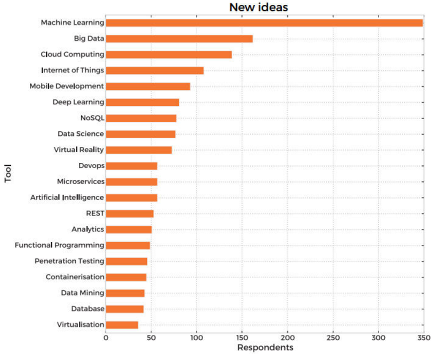


Figure: Popular areas of interest, from the Skill Up 2016: Developer Skills Report[2]

[1] https://techcus.com/p/r1zSmbXut/top-5-highest-paying-programming-languages-of-2016/.

[2] https://techcus.com/p/r1zSmbXut/top-5-highest-paying-programming-languages-of-2016/.

What does the Machine Learning practitioner?

Be able to:

1. design
2. implement
3. assess experimentally

an end-to-end Machine Learning or Data Mining system.

What does the Machine Learning practitioner?

Be able to:

1. design
2. implement
3. assess experimentally

an end-to-end Machine Learning or Data Mining system.

- Which is the problem to be solved? Which are the input and output? Which are the most suitable algorithms? How should data be prepared? Does computation time matter?

What does the Machine Learning practitioner?

Be able to:

1. design
2. implement
3. assess experimentally

an end-to-end Machine Learning or Data Mining system.

- Which is the problem to be solved? Which are the input and output? Which are the most suitable algorithms? How should data be prepared? Does computation time matter?
- Write some code!

What does the Machine Learning practitioner?

Be able to:

1. design
2. implement
3. assess experimentally

an end-to-end Machine Learning or Data Mining system.

- Which is the problem to be solved? Which are the input and output? Which are the most suitable algorithms? How should data be prepared? Does computation time matter?
- Write some code!
- How to measure solution quality? How to compare solutions? Is my solution general?

Subsection 1

Motivating example

The amateur botanist friend

He likes to collect Iris plants. He “realized” that there are 3 species, in particular, that he likes: *Iris setosa*, *Iris virginica*, and *Iris versicolor*. He'd like to have a tool to automatically *classify* collected samples in one of the 3 species.

Figure: Iris versicolor.

How to help him?

Let's help him

- Which is the problem to be solved?

Let's help him

- Which is the problem to be solved?
  - Assign exactly one specie to a sample.

Let's help him

- Which is the problem to be solved?
  - Assign exactly one specie to a sample.
- Which are the input and output?

Let's help him

- Which is the problem to be solved?
  - Assign exactly one specie to a sample.
- Which are the input and output?
  - Output: one species among I. setosa, I. virginica, I. versicolor.

Let's help him

- Which is the problem to be solved?
  - Assign exactly one specie to a sample.
- Which are the input and output?
  - Output: one species among I. setosa, I. virginica, I. versicolor.
  - Input: the plant sample...

Let's help him

- Which is the problem to be solved?
  - Assign exactly one specie to a sample.
- Which are the input and output?
  - Output: one species among I. setosa, I. virginica, I. versicolor.
  - Input: the plant sample...
    - a description in natural language?

Let's help him

- Which is the problem to be solved?
  - Assign exactly one specie to a sample.
- Which are the input and output?
  - Output: one species among I. setosa, I. virginica, I. versicolor.
  - Input: the plant sample...
    - a description in natural language?
    - a digital photo?

Let's help him

- Which is the problem to be solved?
    - Assign exactly one specie to a sample.
- Which are the input and output?
    - Output: one species among I. setosa, I. virginica, I. versicolor.
    - Input: the plant sample...
        - a description in natural language?
        - a digital photo?
        - DNA sequences?

Let's help him

- Which is the problem to be solved?
  - Assign exactly one specie to a sample.
- Which are the input and output?
  - Output: one species among I. setosa, I. virginica, I. versicolor.
  - Input: the plant sample...
    - a description in natural language?
    - a digital photo?
    - DNA sequences?
    - some measurements of the sample!

Iris: input and output

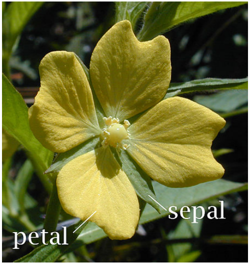


Figure: Sepal and petal.

Input: sepal length and width, petal length and width (in cm)
Output: the class
Example: $(5.1, 3.5, 1.4, 0.2) \rightarrow$ I. setosa

Other information

The botanist friend asked a senior botanist to inspect several samples and label them with the corresponding species.

Sepal length	Sepal width	Petal length	Petal width	Species
5.1	3.5	1.4	0.2	I. setosa
4.9	3.0	1.4	0.2	I. setosa
7.0	3.2	4.7	1.4	I. versicolor
6.0	2.2	5.0	1.5	I. virginica

Notation and terminology

- Sepal length, sepal width, petal length, and petal width are input variables (or independent variables, or features, or attributes).
- Species is the output variable (or dependent variable, or response).

Notation and terminology

$$\mathbf{X} = \begin{pmatrix} x_{1,1} & x_{1,2} & \cdots & x_{1,p} \\ x_{2,1} & x_{2,2} & \cdots & x_{2,p} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n,1} & x_{n,2} & \cdots & x_{n,p} \end{pmatrix} \quad \mathbf{y} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix}$$

- $x_1^T = (x_{1,1}, x_{1,2}, \ldots, x_{1,p})$ is an observation (or instance, or data point), composed of p variable values;

Notation and terminology

$$\mathbf{X} = \begin{pmatrix} x_{1,1} & x_{1,2} & \cdots & x_{1,p} \\ x_{2,1} & x_{2,2} & \cdots & x_{2,p} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n,1} & x_{n,2} & \cdots & x_{n,p} \end{pmatrix} \quad \mathbf{y} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix}$$

- $x_1^T = (x_{1,1}, x_{1,2}, \ldots, x_{1,p})$ is an observation (or instance, or data point), composed of p variable values; y_1 is the corresponding output variable value

Notation and terminology

$$\mathbf{X} = \begin{pmatrix} x_{1,1} & x_{1,2} & \cdots & x_{1,p} \\ x_{2,1} & x_{2,2} & \cdots & x_{2,p} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n,1} & x_{n,2} & \cdots & x_{n,p} \end{pmatrix} \quad \mathbf{y} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix}$$

- $x_1^T = (x_{1,1}, x_{1,2}, \ldots, x_{1,p})$ is an observation (or instance, or data point), composed of p variable values; y_1 is the corresponding output variable value
- $\mathbf{x}_2^T = (x_{1,2}, x_{2,2}, \ldots, x_{n,2})$ is the vector of all the n values for the 2nd variable (X_2).

Notation and terminology

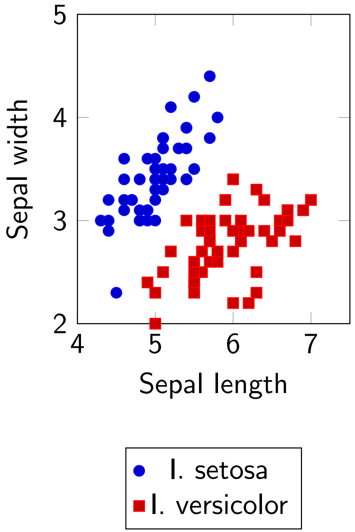
Different communities (e.g., statistical learning vs. machine learning vs. artificial intelligence) use different terms and notation:

- $x_j^{(i)}$ instead of $x_{i,j}$ (hence $x^{(i)}$ instead of x_i)
- m instead of n and n instead of p
- ...

Focus on the meaning!

Iris: visual interpretation

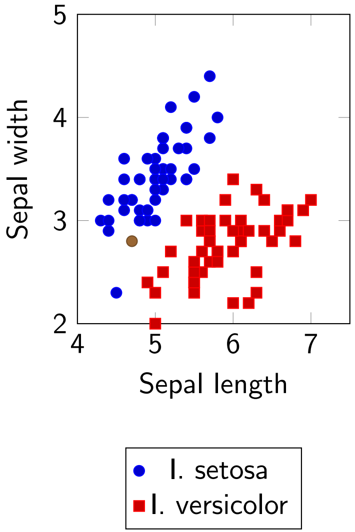
Simplification: forget petal and I. virginica $\rightarrow$ 2 variables, 2 species (binary classification problem).



Iris: visual interpretation

Simplification: forget petal and I. virginica $\rightarrow$ 2 variables, 2 species (binary classification problem).

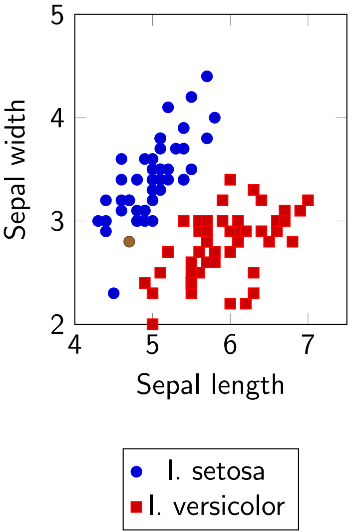
- *Problem*: given any new observation, we want to automatically assign the species.



Iris: visual interpretation

Simplification: forget petal and I. virginica → 2 variables, 2 species (binary classification problem).

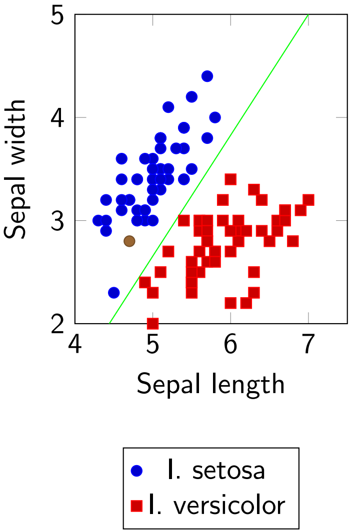
- *Problem*: given any new observation, we want to automatically assign the species.
- *Sketch of a possible solution*:



Iris: visual interpretation

Simplification: forget petal and I. virginica → 2 variables, 2 species (binary classification problem).

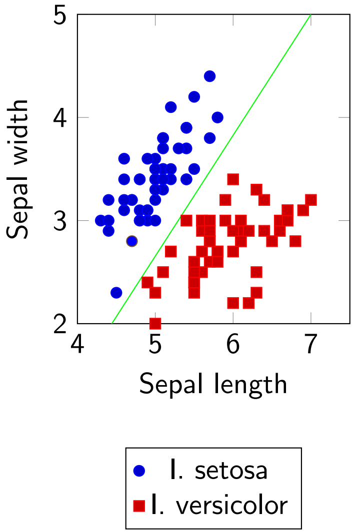
- *Problem*: given any new observation, we want to automatically assign the species.
- *Sketch of a possible solution*:
  1. learn a model (classifier)



Iris: visual interpretation

Simplification: forget petal and I. virginica $\rightarrow$ 2 variables, 2 species (binary classification problem).

- *Problem*: given any new observation, we want to automatically assign the species.
- *Sketch of a possible solution*:
  1. learn a model (classifier)
  2. "use" model on new observations



“A” model?

There could be many possible models:

- how to choose?
- how to compare?

Choosing the model

The choice of the model/tool/algorithm to be used is determined by many factors:

- Problem size (n and p)
- Availability of an output variable (y)
- Computational effort (when learning or "using")
- Explicability of the model
- ...

We will see many options.

Comparing many models

Experimentally: does the model work well on (new) data?

Comparing many models

Experimentally: does the model work well on (new) data?
Define “works well”:

- a single performance index?
- how to measure?
- repeatability/reproducibility. . .

We will see/discuss many options.

It does not work well...

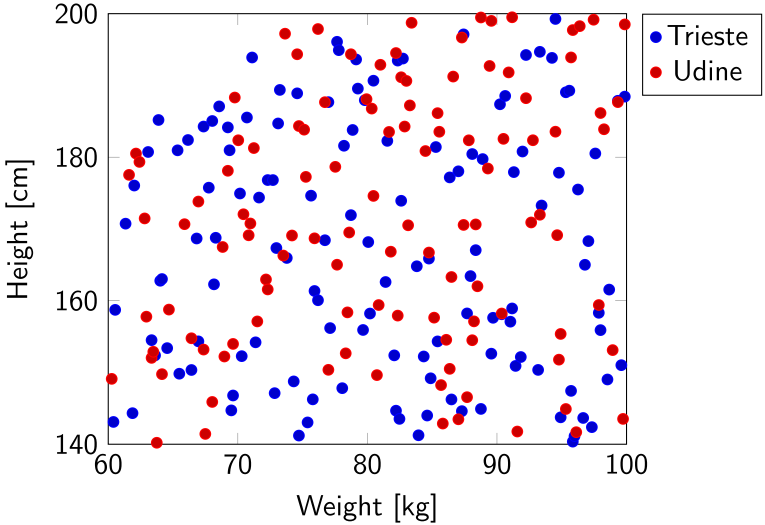
Why?

- the data is not informative
- the data is not representative
- the data has changed
- the data is too noisy

We will see/discuss these issues.

ML is not magic

Problem: find birth town from height/weight.



Q: which is the data issue here?

Implementation

When “solving” a problem, we usually need:

- explore/visualize data
- apply one or more learning algorithms
- assess learned models

“By hands?” No, with software!

ML/DM software

Many options:

- libraries for general purpose languages:
  - Java: e.g., `http://haifengl.github.io/smile/`
  - Python: e.g., `http://scikit-learn.org/stable/`
  - ...
- specialized sw environments:
  - Octave: `https://en.wikipedia.org/wiki/GNU_Octave`
  - R: `https://en.wikipedia.org/wiki/R_(programming_language)`
- from scratch

ML/DM software: which one?

- production/prototype
- platform constraints
- degree of (data) customization
- documentation availability/community size
- ...
- previous knowledge/skills

Section 2

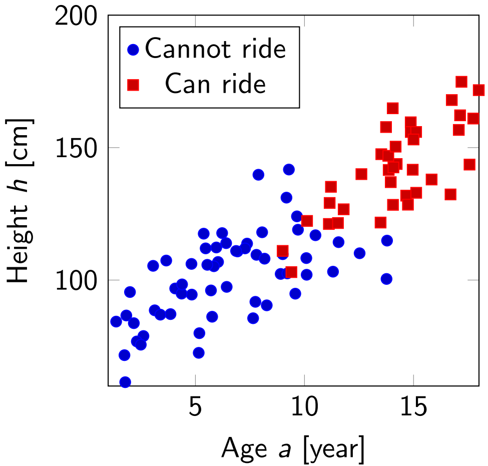
Tree-based methods

The carousel robot attendant

Problem: replace the carousel attendant with a robot which automatically decides who can ride the carousel.

Carousel: data

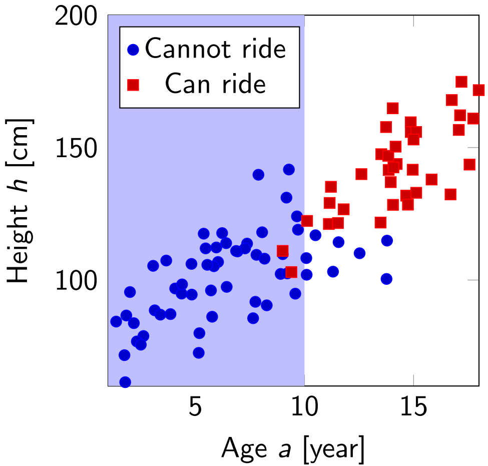
Observed human attendant's decisions.



How can the robot take the decision?

Carousel: data

Observed human attendant's decisions.

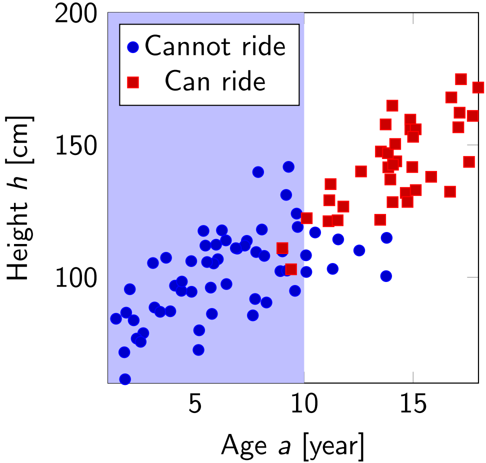


How can the robot take the decision?

- if younger than 10 $\rightarrow$ can't!

Carousel: data

Observed human attendant's decisions.

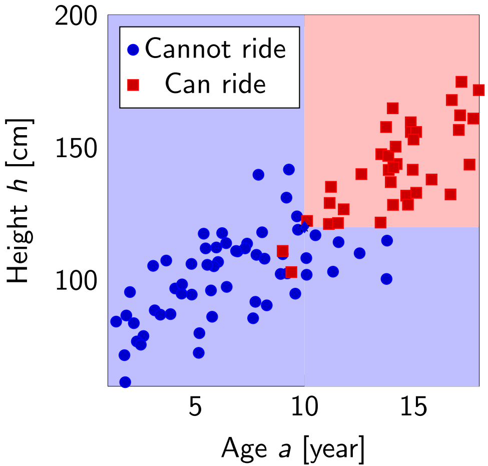


How can the robot take the decision?

- if younger than 10 $\rightarrow$ can't!
- otherwise:

Carousel: data

Observed human attendant's decisions.

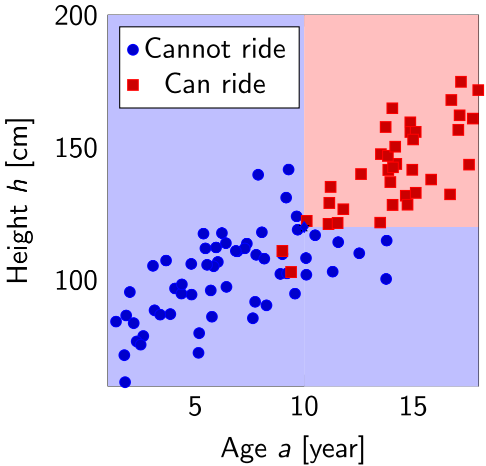


How can the robot take the decision?

- if younger than 10 $\rightarrow$ can't!
- otherwise:
  - if shorter than 120 $\rightarrow$ can't!
  - otherwise $\rightarrow$ can!

Carousel: data

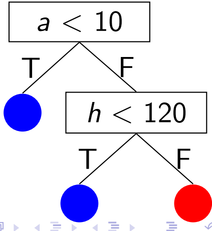
Observed human attendant's decisions.



How can the robot take the decision?

- if younger than 10 $\rightarrow$ can't!
- otherwise:
  - if shorter than 120 $\rightarrow$ can't!
  - otherwise $\rightarrow$ can!

Decision tree!



How to build a decision tree

Dividi-et-impera (recursively):

- find a cut variable and a cut value
- for left-branch, dividi-et-impera
- for right-branch, dividi-et-impera

How to build a decision tree: detail

Recursive binary splitting

```
function BuildDecisionTree(X, y)
    if ShouldStop(y) then
        ŷ ← most common class in y
        return new terminal node with ŷ
    else
        (i, t) ← BestBranch(X, y)
        n ← new branch node with (i, t)
        append child BuildDecisionTree(X|x_i<t, y|x_i<t) to n
        append child BuildDecisionTree(X|x_i≥t, y|x_i≥t) to n
        return n
    end if
end function
```

- Recursive binary splitting
- Top down (start from the "big" problem)

Best branch

function $\textsc{BestBranch}(\mathbf{X}, \mathbf{y})$

  $(i^\star, t^\star) \leftarrow \arg\min_{i,t} E(\mathbf{y}|_{\mathbf{x}_i \geq t}) + E(\mathbf{y}|_{\mathbf{x}_i < t})$

  return $(i^\star, t^\star)$

end function

Classification error on subset:

$$E(\mathbf{y}) = \frac{|\{y \in \mathbf{y} : y \neq \hat{y}\}|}{|\mathbf{y}|}$$

$$\hat{y} = \text{the most common class in } \mathbf{y}$$

- Greedy (choose split to minimize error now, not in later steps)

Best branch

$$(i^\star, t^\star) \leftarrow \arg\min_{i,t} E(\mathbf{y}|_{\mathbf{x}_i \geq t}) + E(\mathbf{y}|_{\mathbf{x}_i < t})$$

The formula say what is done, not how is done!

Q: different "how" can differ? how?

Stopping criterion

function SHOULDSTOP($\mathbf{y}$)
    if $\mathbf{y}$ contains only one class **then**
        return true
    else if $|\mathbf{y}| < k_{\mathsf{min}}$ **then**
        return true
    else
        return false
    end if
end function

Other possible criterion:

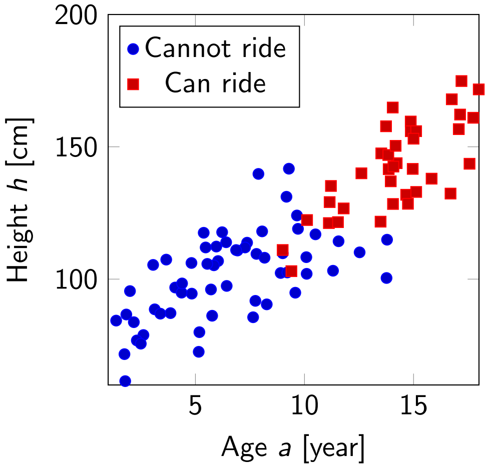
- tree depth larger than d_{max}

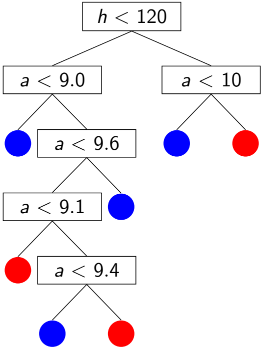
Categorical independent variables

- Trees can work with categorical variables
- Branch node is $x_i = c$ or $x_i \in C' \subset C$ (c is a class)
- Can mix categorical and numeric variables

Stopping criterion: role of $k_{\min}$

Suppose $k_{\min} = 1$ (never stop for **y** size)





Q: what's wrong?

Tree complexity

When the tree is "too complex"

- less readable/understandable/explicable
- maybe there was noise into the data

Q: what's noise in carousel data?

Tree complexity issue is not related (only) with $k_{\min}$

Tree complexity: other interpretation

- maybe there was noise into the data

The tree *fits* the learning data too much:

- it overfits (**overfitting**)
- does not generalize (high **variance**: model varies if learning data varies)

High variance

"model varies if learning data varies": what? why data varies?

- learning data is about the system/phenomenon/nature S
  - a collection of *observations* of S
  - a point of view on S

High variance

"model varies if learning data varies": what? why data varies?

- learning data is about the system/phenomenon/nature S
  - a collection of *observations* of S
  - a point of view on S
- learning is about understanding/knowing/explaining S

High variance

"model varies if learning data varies": what? why data varies?

- learning data is about the system/phenomenon/nature S
  - a collection of *observations* of S
  - a point of view on S
- learning is about understanding/knowing/explaining S
  - if I change the point of view on S, my knowledge about S should remain the same!

Fighting overfitting

- large $k_{\min}$ (large w.r.t what?)
- when building, limit depth
- when building, don't split if low overall impurity decrease
- after building, *prune*

(bias, variance will be detailed later)

Evaluation: k-fold cross-validation

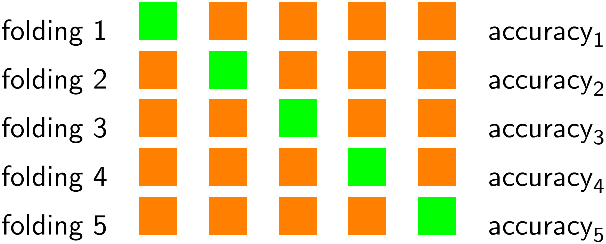
How to estimate the predictor performance on new (unavailable) data?

1. split learning data ($\mathbf{X}$ and $\mathbf{y}$) in k equal slices (each of $\frac{n}{k}$ observations/elements)
2. for each split (i.e., each $i \in \{1, \dots, k\}$)
   2.1 learn on all but k-th slice
   2.2 compute classification error on **unseen** k-th slice
3. average the k classification errors

In essence:

- can the learner generalize on available data?
- how the learned artifact will behave on unseen data?

Evaluation: *k*-fold cross-validation



$$\text{accuracy} = \frac{1}{k} \sum_{i=1}^{i=k} \text{accuracy}_i$$

Or with classification error rate or any other meaningful (effectiveness) measure

Q: how should data be split?

Subsection 1

Regression trees

Regression with trees

Trees can be used for regression, instead of classification.

decision tree vs. regression tree

Tree building: decision $\rightarrow$ regression

function $\textsc{BuildDecisionTree}(\mathbf{X}, \mathbf{y})$
    if $\textsc{ShouldStop}(\mathbf{y})$ **then**
        $\hat{y} \leftarrow$ most common class in $\mathbf{y}$
        return new terminal node with $\hat{y}$
    else
        $(i, t) \leftarrow \textsc{BestBranch}(\mathbf{X}, \mathbf{y})$
        $n \leftarrow$ new branch node with (i, t)
        append child $\textsc{BuildDecisionTree}(\mathbf{X}|_{\mathbf{x}_i < t}, \mathbf{y}|_{\mathbf{x}_i < t})$ to n
        append child $\textsc{BuildDecisionTree}(\mathbf{X}|_{\mathbf{x}_i \geq t}, \mathbf{y}|_{\mathbf{x}_i \geq t})$ to n
        return n
    end if
end function

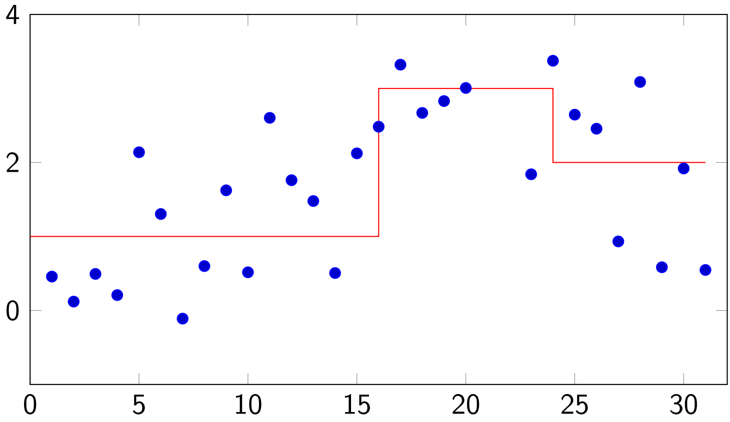
Q: what should we change?

Tree building: decision $\rightarrow$ regression

function $\textsc{BuildDecisionTree}(\mathbf{X}, \mathbf{y})$
  if $\textsc{ShouldStop}(\mathbf{y})$ **then**
    $\hat{y} \leftarrow \bar{y}$ $\triangleright$ mean $\mathbf{y}$
    return new terminal node with $\hat{y}$
  else
    $(i, t) \leftarrow \textsc{BestBranch}(\mathbf{X}, \mathbf{y})$
    $n \leftarrow$ new branch node with (i, t)
    append child $\textsc{BuildDecisionTree}(\mathbf{X}|_{\mathbf{x}_i < t}, \mathbf{y}|_{\mathbf{x}_i < t})$ to n
    append child $\textsc{BuildDecisionTree}(\mathbf{X}|_{\mathbf{x}_i \geq t}, \mathbf{y}|_{\mathbf{x}_i \geq t})$ to n
    return n
  end if
end function

Q: what should we change?

Interpretation



Regression and overfitting

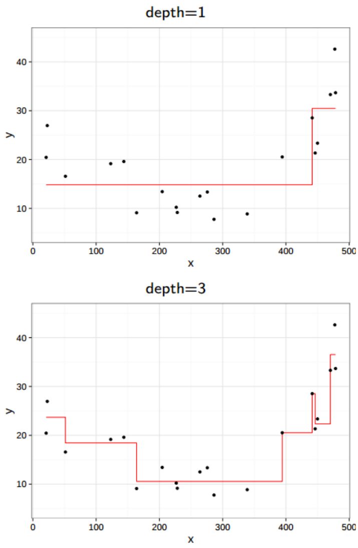


Image from F. Daolio

Trees in summary

Pros:

- ▲ easily interpretable/explicable
- ▲ learning and regression/classification easily understandable
- ▲ can handle both numeric and categorical values

Cons:

- ▼ not so accurate (**Q:** always?)

Tree accuracy?

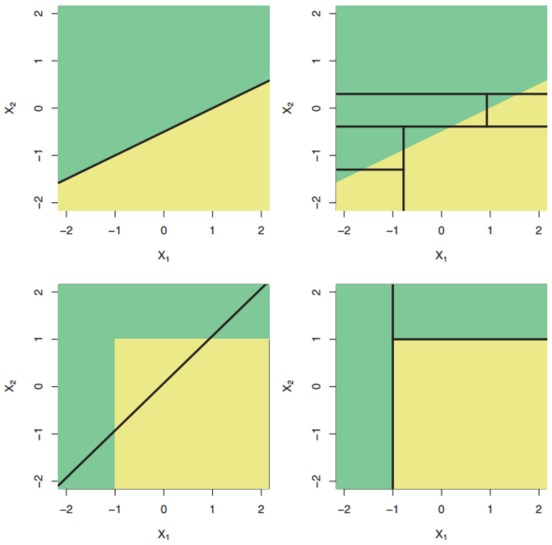
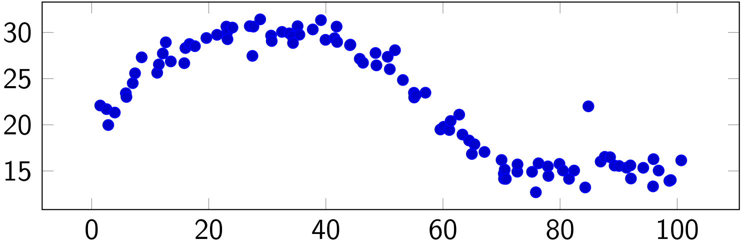


Image from An Introduction to Statistical Learning

Subsection 2

Trees aggregation

Weakness of the tree



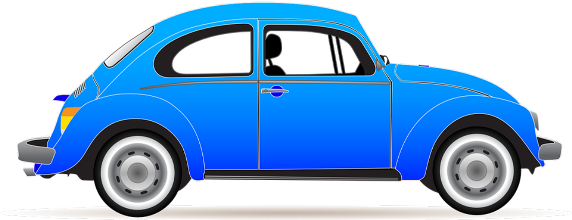
Small tree:

- low complexity
- will hardly fit the “curve” part
- *high bias, low variance*

Big tree:

- high complexity
- may overfit the noise on the right part
- *low bias, high variance*

The trees view



Small tree:

- “a car is something that moves”

Big tree:

- “a car is a made-in-Germany blue object with 4 wheels, 2 doors, chromed fenders, curved rear enclosing engine”

Big tree view

A big tree:

- has a detailed view of the learning data (high complexity)
- “trusts too much” the learning data (high variance)

What if we “combine” different big tree views and ignore details on which they disagree?

Wisdom of the crowds

What if we "combine" different big tree views and ignore details on which they disagree?

- many views
- independent views
- aggregation of views

$\approx$ *the wisdom of the crowds*: a collective opinion may be better than a single expert's opinion

Wisdom of the trees

- many views
- independent views
- aggregation of views

Wisdom of the trees

- many views
  - just use many trees
- independent views
- aggregation of views

Wisdom of the trees

- many views
  - just use many trees
- independent views

- aggregation of views
  - just average prediction (regression) or take most common prediction (classification)

Wisdom of the trees

- many views
  - just use many trees
- independent views
  - ??? learning is deterministic: same data $\Rightarrow$ same tree $\Rightarrow$ same view
- aggregation of views
  - just average prediction (regression) or take most common prediction (classification)

Independent views

Independent views $\equiv$ different points of view $\equiv$ *different* learning data

But we have only *one* learning data!

Independent views: idea!

Like in cross-fold, consider only a part of the data, but:

- instead of a subset
- a sample with repetitions

Independent views: idea!

Like in cross-fold, consider only a part of the data, but:

- instead of a subset
- a sample with repetitions

$$
\begin{aligned}
\mathbf{X} &= (x_1^T \, x_2^T \, x_3^T \, x_4^T \, x_5^T) && \text{original learning data} \\
\mathbf{X_1} &= (x_1^T \, x_5^T \, x_3^T \, x_2^T \, x_5^T) && \text{sample 1} \\
\mathbf{X_2} &= (x_4^T \, x_2^T \, x_3^T \, x_1^T \, x_1^T) && \text{sample 2} \\
\mathbf{X_i} &= \dots && \text{sample } i
\end{aligned}
$$

- (**y** omitted for brevity)
- learning data size is not a limitation (differently than with subset)

Independent views: idea!

Like in cross-fold, consider only a part of the data, but:

- instead of a subset
- a sample with repetitions

$$\begin{aligned}
\mathbf{X} &= (x_1^T x_2^T x_3^T x_4^T x_5^T) && \text{original learning data} \\
\mathbf{X_1} &= (x_1^T x_5^T x_3^T x_2^T x_5^T) && \text{sample 1} \\
\mathbf{X_2} &= (x_4^T x_2^T x_3^T x_1^T x_1^T) && \text{sample 2} \\
\mathbf{X_i} &= \dots && \text{sample } i
\end{aligned}$$

- ($\mathbf{y}$ omitted for brevity)
- learning data size is not a limitation (differently than with subset)

Bagging of trees (*bootstrap*, more in general)

Tree bagging

When learning:

1. Repeat B times
   1.1 take a sample of the learning data
   1.2 learn a tree (unpruned)

When predicting:

1. Repeat B times
   1.1 get a prediction from ith learned tree
2. predict the average (or most common) prediction

For classification, other aggregations can be done: majority voting (most common) is the simplest

How many trees?

B is a parameter:

- when there is a parameter, there is the problem of finding a good value
- remember k_{min}, depth (**Q:** impact on?)

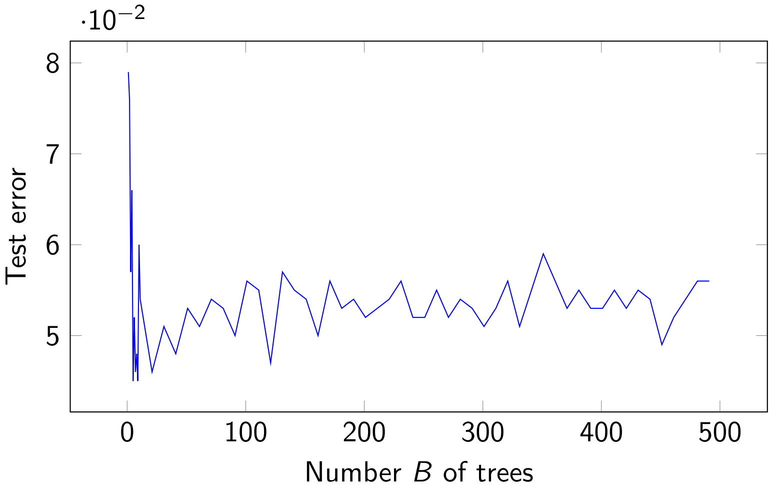
How many trees?

B is a parameter:

- when there is a parameter, there is the problem of finding a good value
- remember $k_{\min}$, depth (**Q:** impact on?)
- it has been shown (experimentally) that
  - for "large" B, bagging is better than single tree
  - increasing B does not cause overfitting
  - (for us: default B is ok! "large" $\approx$ hundreds)

Q: how better? at which cost?

Bagging



$\cdot 10^{-2}$
8
7
6
5
Test error
0
100
200
300
400
500
Number B of trees

Independent view: improvement

Despite being learned on different samples, bagging trees may be correlated, hence views are not very independent

- e.g., one variable is much more important than others for predicting (*strong predictor*)

Idea: force point of view differentiation by “hiding” variables

Random forest

When learning:

1. Repeat B times
   1.1 take a sample of the learning data
   1.2 consider only m on p independent variables
   1.3 learn a tree (unpruned)

When predicting:

1. Repeat B times
   1.1 get a prediction from ith learned tree
2. predict the average (or most common) prediction

- (observations and) variables are **random**ly chosen...
- ...to learn a **forest** of trees

Q: are missing variables a problem?

Random forest: parameter m

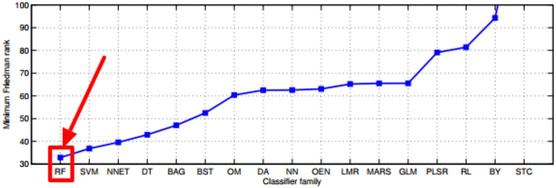
How to choose the value for m?

- $m = p \rightarrow$ bagging
- it has been shown (experimentally) that
  - m does not relate with overfitting
  - $m = \sqrt{p}$ is good for classification
  - $m = \frac{p}{3}$ is good for regression
  - (for us, default m is ok!)

Random forest

Experimentally shown: one of the "best" multi-purpose supervised classification methods

- Manuel Fernández-Delgado et al. "Do we need hundreds of classifiers to solve real world classification problems". In: *J. Mach. Learn. Res* 15.1 (2014), pp. 3133–3181



but...

No free lunch!

"Any two optimization algorithms are equivalent when their performance is averaged across all possible problems"

- David H Wolpert. "The lack of a priori distinctions between learning algorithms". In: *Neural computation* 8.7 (1996), pp. 1341–1390

Why free lunch?

- many restaurants, many items on menus, many possibly prices for each item: where to go to eat?
- no general answer
- but, if you are a vegan, or like pizza, then a best choice could exist

Q: problem? algorithm?

Nature of the prediction

Consider classification:

- tree $\rightarrow$ the class
- forest $\rightarrow$ the class, as resulting from a voting

Nature of the prediction

Consider classification:

- tree $\rightarrow$ the class
  - "virginica" is just "virginica"
- forest $\rightarrow$ the class, as resulting from a voting
  - "241 virginica, 170 versicolor, 89 setosa" is different than "478 virginica, 10 versicolor, 2 setosa"

Is this information useful/exploitable?

Confidence/tunability

Voting outcome:

- in classification, a measure of confidence of the decision
- in binary classification, voting threshold can be tuned to adjust bias towards one class (*sensitivity*)

Q: in regression?

Binary classification

Consider the problem of classifying a person ('s data) as suffering or not suffering from a disease X.

- **positive**: an observation of "suffering" class
- **negative**: an observation of "not suffering" class

In other problems, positive may mean a different thing: define it!

FPR, FNR

Given some labeled data and a classifier for the disease X problem, we can measure:

- the number of negative observations *wrongly* classified as positives: False Positives (**FP**)
- the number of positive observations *wrongly* classified as negatives: False Negatives (**FN**)

To decouple FP, FN from data size:

$$\mathrm{FPR} = \frac{\mathrm{FP}}{\mathrm{N}} = \frac{\mathrm{FP}}{\mathrm{FP} + \mathrm{TN}}$$

$$\mathrm{FNR} = \frac{\mathrm{FN}}{\mathrm{P}} = \frac{\mathrm{FN}}{\mathrm{FN} + \mathrm{TP}}$$

Accuracy and error rate

$$\text{Accuracy} = 1 - \text{Error Rate}$$

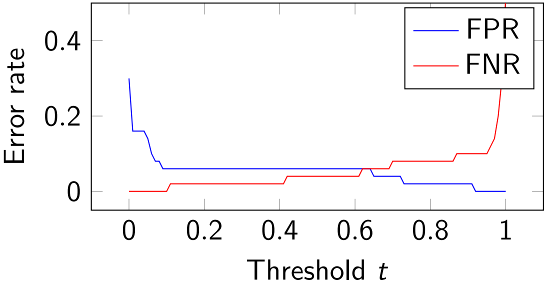
$$\text{Error Rate} = \frac{\text{FN} + \text{FP}}{\text{P} + \text{N}}$$

Q: Error Rate $\stackrel{?}{=} \frac{\text{FPR}+\text{FNR}}{2}$

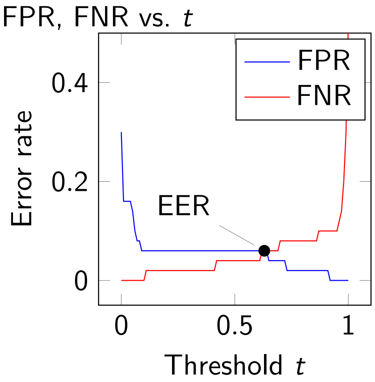
FPR, FNR and sensitivity

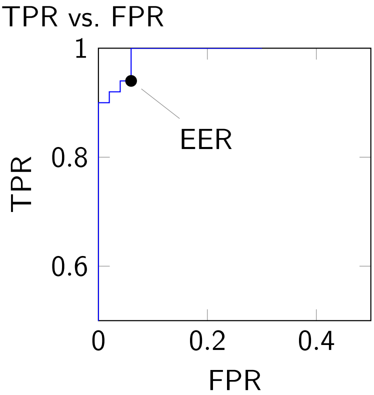
- Suppose FPR $= 0.06$, FNR $= 0.04$ with threshold set to 0.5 (default for RF)
- One could be interested in "limiting" the FNR...

Experimentally:



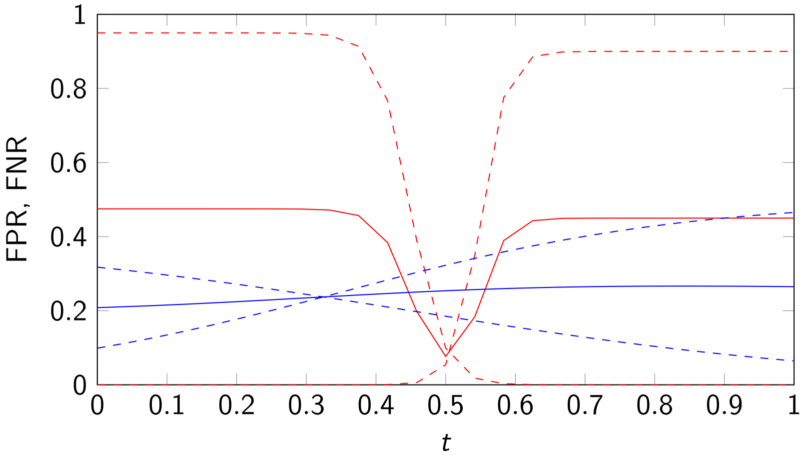
Receiver operating characteristic (ROC)





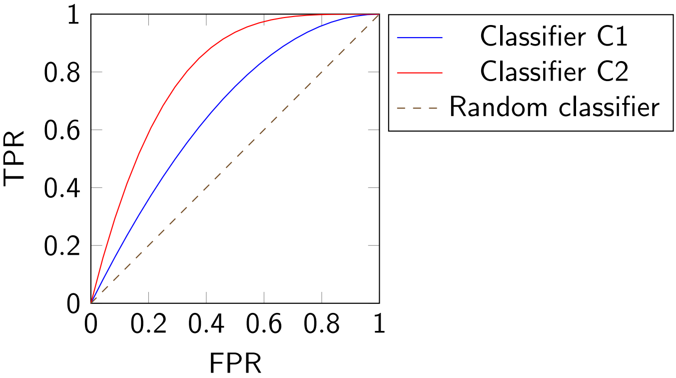
- Equal error rate (**EER**)

. . . is better than



- which is the best?
- robustness w.r.t. t?

ROC and comparison



C1 is better than C2: how much?

- EER
- Area under the curve (**AUC**)

Bagging/RF/boosting in summary

	Tree	Bagging	RF	Boosting
interpretability	▲			
numeric/categorical	▲	▲	▲	▲
accuracy	▼		▲	▲
test error estimate		▲	▲	
variable importance		▲	▲	▲
confidence/tunability		▲	▲	
fast to learn	▲*			▼
(almost) non-parametric		▲	▲	

*: **Q:** how faster? when? does it matter?