Data Science
Topic: Data Mining

Elena Mocanu

University of Twente

November 16, 2022

Overview

Data Mining in the Data Science context

Data contains value and knowledge

Why Data Mining/Data Science?

- Multi-skilled AI
- mRNA vaccine
- TikTok recommendation alg.
- Lithium-metal batteries
- Hyper-accurate GPS
- GPT-3
- Digital contact tracing
- Green hydrogen
- Data trusts
- Remote everything

▸ Link

Data Mining in the Data Science context

PRESTATIEKENMERKEN

Klinische Gevoeligheid, Specificiteit en Nauwkeurigheid

De werking van de snelle SARS-CoV-2-antigeentest werd vastgesteld met neusuitstrijkjes die werden afgenomen bij symptomatische personen die vermoedelijk COVID-19 besmet waren. De resultaten tonen de volgende relatieve gevoelighei relatieve exactheid aan:

Klinisch resultaat voor snelle test op SARS-CoV-2-antigeen

Methode		RT-PCR		Totale Resultate
Snelle test op SARS-CoV-2-antigeen	**Resultaten**	Negatief	Positief	
	Negatief	433	5	438
	Positief	2	165	167
Totale Resultaten		435	170	605

Relatieve gevoeligheid: 97,1% (93,1%-98,9%)*
Relatieve Specificiteit: 99,5% (98,2%-99,9%)*
Nauwkeurigheid: 98,8% (97,6%-99,5%)*
***95% Betrouwbaarheidsintervallen**

Stratificatie van de positieve monsters na aanvang van de symptomen tussen 0-3 dagen heeft een positieve procent-overeenkomst (PPA) van 98,8% (n=81) en 4-7 dagen heeft een PPA van 96,8% (n=62).

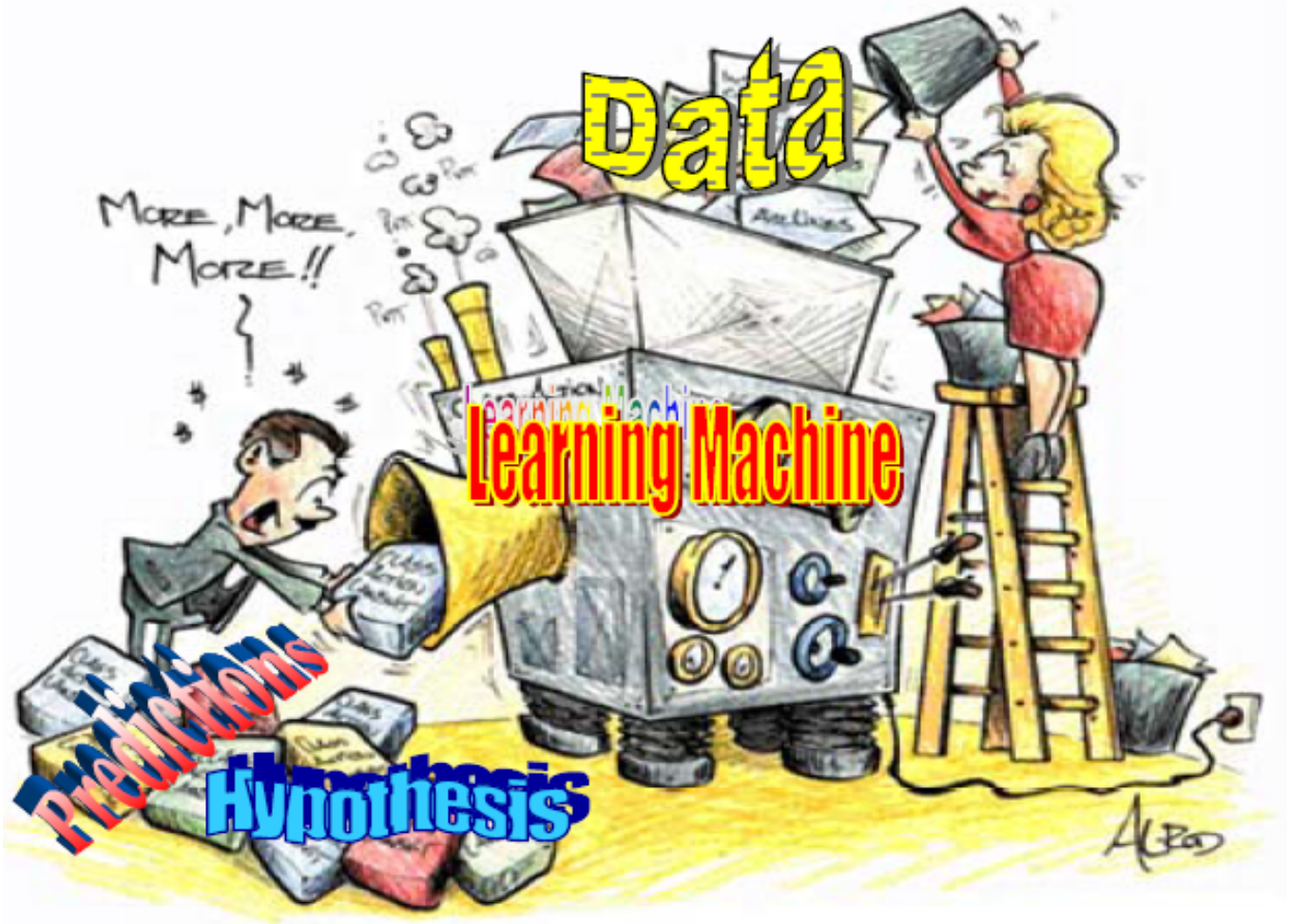
Positieve monsters met Ct-waarde ≤ 33 hebben een hogere positieve procent-overeenkomst (PPA) van 98,7% (n=153).

Detectiegrens (LOD)

De detectielimiet van de sneltest voor SARS-CoV-2-antigenenen is vastgesteld met behulp van beperkende verdunningen van een geïnactiveerd viraal monster. Het virale monster werd in een reeks concentraties vermengd met negatief menselijk neusmonster. Elk niveau is getest op 30 replicaten. Uit de resultaten blijkt dat de LOD $1{,}6*10^2$ $TCID_{50}$/mL is.

Gebruiksstudie

Aim of this topic & lecture



Machine Learning (ML) and Data

"Learning is any process by which a system improves performance from experience."

Herbert Simon

Machine Learning (ML) and Data

"Learning is any process by which a system improves performance from experience."

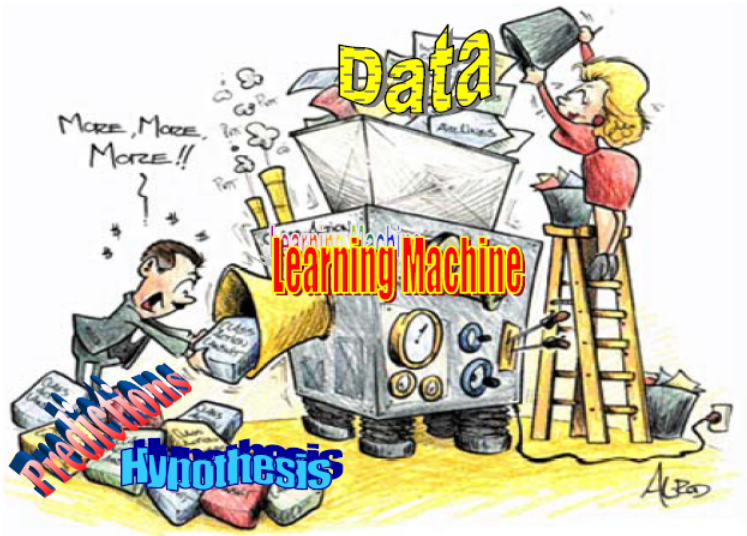
Herbert Simon

Machine Learning definition (Tom Mitchell, 1998)

- Machine Learning is the study of algorithms that:
  - improve their performance P
  - at some task T
  - with experience E.
- A well-defined learning task is given by $< P, T, E >$.

What is Data Mining?

- Given lots of data
- Discover patterns and models that are:
  - Valid: hold on new data with some certainty
  - Useful: should be possible to act on the item
  - Unexpected: non-obvious to the system
  - Understandable: humans should be able to interpret the pattern



ML in a Nutshell

- Tens of thousands of machine learning algorithms
- Hundreds new every year

Every machine learning algorithm has three components:

1. Representation - the space of allowed models (the *hypothesis space*)
2. Evaluation - how to judge (or prefer) one model vs. another (e.g. *utility function*, *loss function*, *scoring function*, *or fitness function*)
3. Optimization - a method to search among the models for the highest-scoring one

(1) Representation

- Linear Regression
- Decision trees
- Sets of rules / Logic programs
- Instances
- Graphical models (Bayes/Markov nets)
- Neural networks
- Support vector machines
- Ensemble models
- Etc.

(2) Evaluation

- Accuracy
- Precision and recall
- Mean squared error
- Likelihood
- Posterior probability
- Cost / Utility
- Margin
- Entropy
- K-L divergence
- Etc.

(3) Optimization

- Combinatorial optimization
  E.g.: Greedy search
- Convex optimization
  E.g.: Gradient descent
- Constrained optimization
  E.g.: Linear programming
- Nonconvex optimization
  E.g.: (Stochastic) Gradient Descent, Evolutionary Algorithms

Types of learning

Supervised learning

- correct output known for each training example
- learn to predict output when given an input vector
- the most used type
- wide-area of academic and industrial applications

Methods

- Naïve Bayes
- Support Vector Machine
- Artificial Neural Networks
- Decision Trees,...

Examples of specific tasks:

- Classification: discrete output
- Regression: real-valued output

Learns from data:

- **Training data include desired outputs.**

Unsupervised learning

- correct output is not known for the training examples
- create an internal representation of the input, capturing regularities/ structure in data
- the most promising type (most of the data is unlabeled)
- many applications

Methods

- k-mean clustering
- Restricted Boltzmann Machine (auto-encoders), ...

Examples of specific tasks:

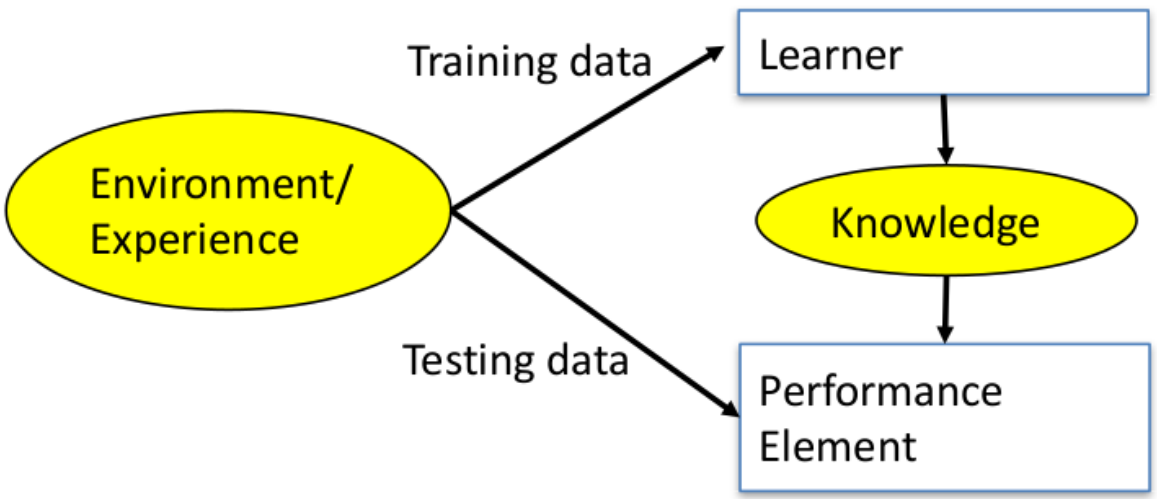
- Discover clusters
- Discover factors/structures

Learns from data:

- **Training data does not include desired outputs.**

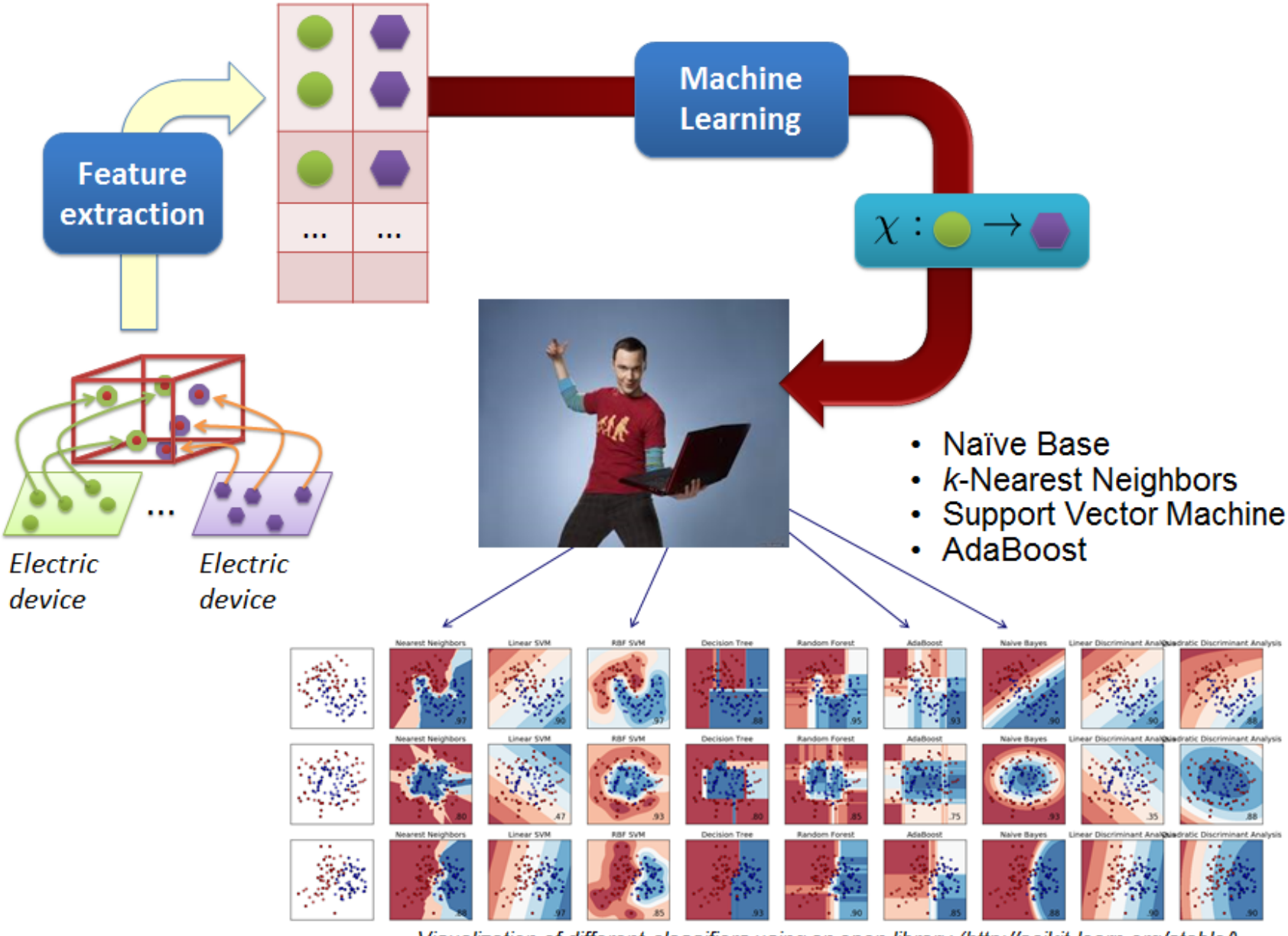
Designing a Learning System

- Choose the training experience
- Choose exactly what is to be learned (i.e. the target function)
- Choose how to represent the target function
- Choose a learning algorithm to infer the target function from the experience



Slide credit: Ray Mooney & Eric Eaton

Example of DM basic flow



Visualization of different classifiers using an open library (http://scikit-learn.org/stable/)

Example of DM basic flow

- ▷ Binary classification problem:
  - Is this email spam?
  - Is my house worth at least $200,000?
  - Is the banner/email clicked/opened by the user?
  - Is the current document about finance?
  - Is there a person in the image? Is it a man or a woman?
  - (!) Output variable is binary value (e.g. yes/no; true/false)
- ▷ Multi-class classification problems:
  - Which kind of flower is this?
  - What's the primary topic of this webpage?
  - Which digit/letter is drawn in the image?
  - (!) Output variable is a categorical value.
- ▷ Regression problems:
  - Predicting age of a person
  - Predicting the price of a house tomorrow
  - (!) Output variable is a real or continuous value.

Example of DM basic flow

Which of the following are classification problems? And which of them are regression problems?

1. Predicting the gender of a person by his/her handwriting style
2. Predicting house price based on area
3. Predicting nationality of a person
4. Predict the number of copies a music album will be sold next month
5. Predicting whether the stock price of a company will increase tomorrow

Example of DM basic flow

Which of the following are classification problems? And which of them are regression problems?

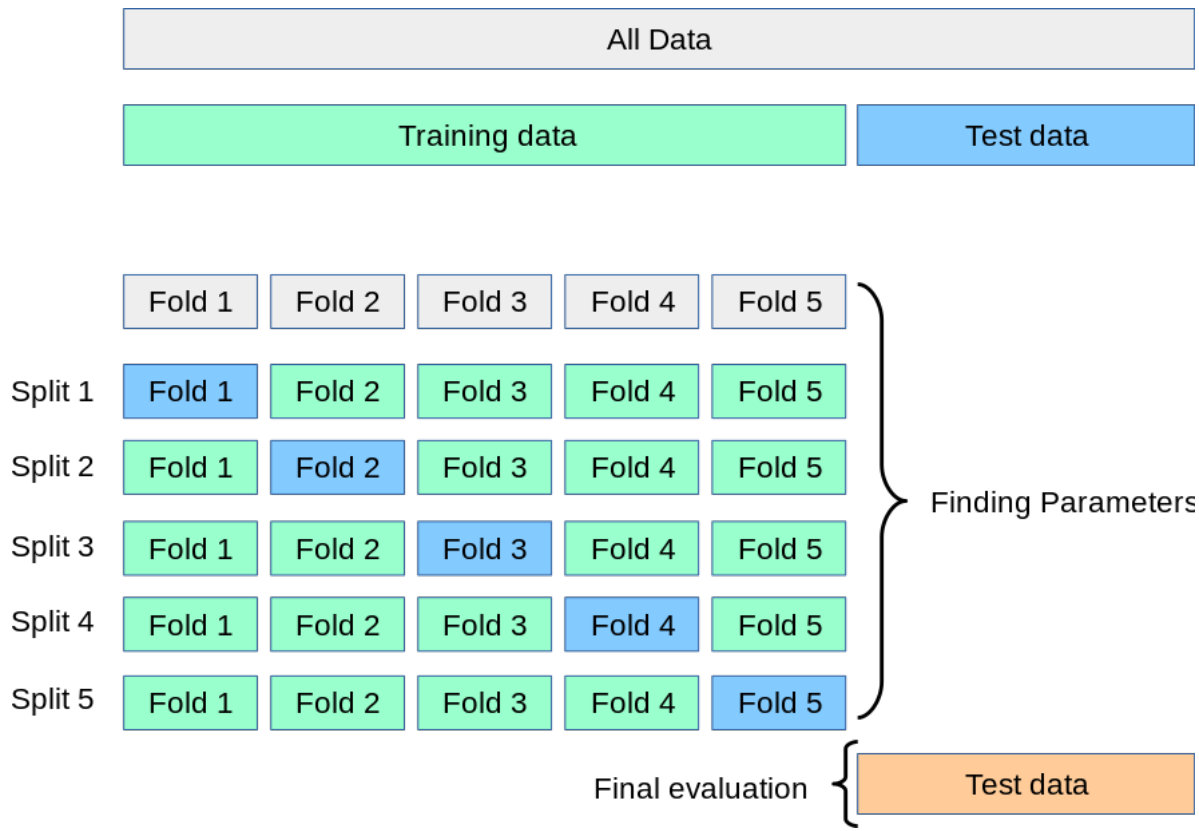
1. Predicting the gender of a person by his/her handwriting style
2. Predicting house price based on area
3. Predicting nationality of a person
4. Predict the number of copies a music album will be sold next month
5. Predicting whether the stock price of a company will increase tomorrow

DM basic flow

- Assuming a lot of data
- Fix the task (classification/regression/clustering)
- Data Visualization and Pre-processing (clean, feature selection/extraction, normalization)
- Input them into DM algorithm. (Garbage in! $\rightarrow$ Garbage out!)

Training/Testing split

- Split the data in training and testing
- Use K-fold Cross-Validation as part of the training
  - Split the data in k folds
  - Train on k-1 folds, and test on the remaining one.



Classification

1. Naïve Bayes
2. Decision Trees

Naïve Bayes

Illustration behind the Naive Bayes algorithm

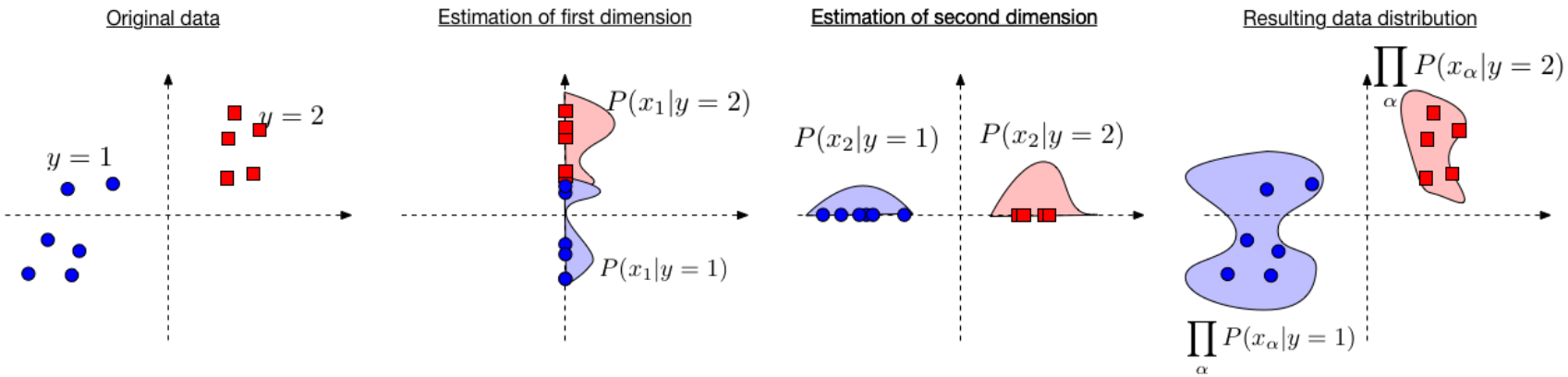
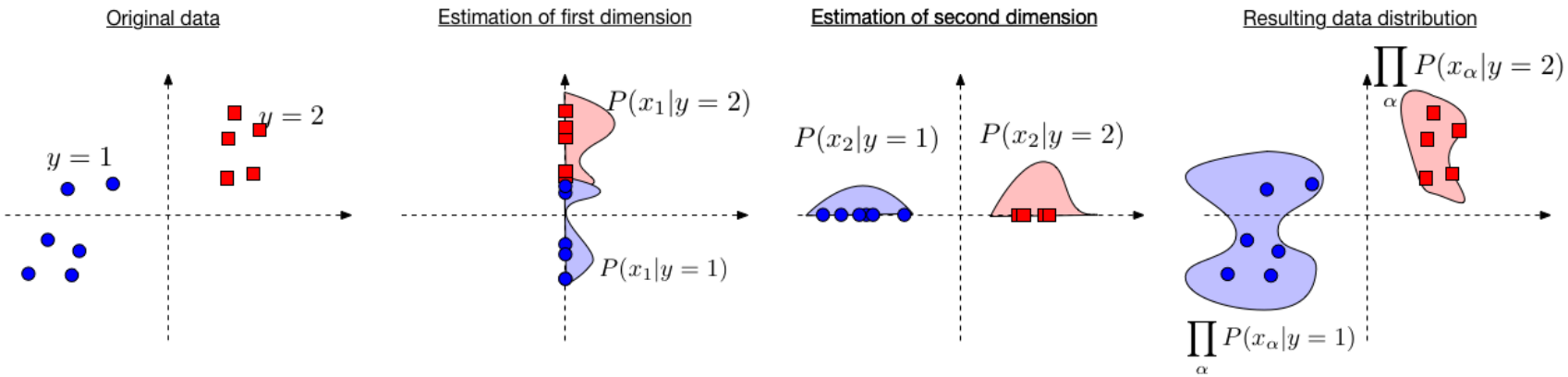


Illustration behind the Naive Bayes algorithm



- We estimate $P(x_\alpha|y)$ independently in each dimension (middle two images) and then obtain an estimate of the full data distribution by assuming conditional independence $P(x|y) = \Pi_\alpha P(x_\alpha|y)$ (very right image)

Bayes' Rule

- Recall Bayes' Rule:

$$P(hypothesis|evidence) = \frac{P(hyphotesis)P(evidence|hyphotesis)}{P(evidence)}$$

- Equivalently, we can write:

$$P(Y = y_k|X = x_i) = \frac{P(Y = y_k)P(X = x_i|Y = y_k)}{P(X = x_i)}$$

where X is a random variable representing the evidence and Y is a random variable for the label

- This is actually short for:

$$P(Y = y_k|X = x_i) = \frac{P(Y = y_k)P(X_1 = x_{i,1} \wedge ... \wedge X_d = x_{i,d}|Y = y_k)}{P(X_1 = x_{i,1} \wedge ... \wedge X_d = x_{i,d})}$$

where X_j denotes the random variable for the j^{th} feature

Naïve Bayes classifier

$$P(Y = y_k|X = x_i) = \frac{P(Y = y_k)P(X_1 = x_{i,1} \wedge ... \wedge X_d = x_{i,d}|Y = y_k)}{P(X_1 = x_{i,1} \wedge ... \wedge X_d = x_{i,d})}$$

Idea

1. Use the training data to estimate $P(Y)$ and $P(X|Y)$
2. Then, use Bayes rule to infer $P(Y|X_{new})$ for new data

- $P(Y = y_k)$ is easy to estimate from the data
- $P(Y = y_k)P(X_1 = x_{i,1} \wedge ... \wedge X_d = x_{i,d}|Y = y_k)$ is impractical to estimate, but necessary
- $P(X_1 = x_{i,1} \wedge ... \wedge X_d = x_{i,d})$ unnecessary to estimate

Naïve Bayes classifier

Problem: Estimating the joint probability distribution is not practical (severely overfits).

However, if we make the assumption that the attributes are independent given the class label, estimation is easy!

$$P(X_1, X_2, ..., X_d|Y) = \prod_{j=1}^{d} P(X_j|Y)$$

In other words, we assume all attributes are *conditionally independent* given Y.

Training Naïve Bayes - Example

Estimate $P(X_j|Y)$ and $P(Y)$ directly from the training data by counting!

Sky X_1	**Temp** X_2	**Humid** X_3	**Wind** X_4	**Water** X_5	**Forecast** X_6	**Play?** Y
sunny	warm	normal	strong	warm	same	yes
sunny	warm	high	strong	warm	same	yes
rainy	cold	high	strong	warm	change	no
sunny	warm	high	strong	cool	change	yes

$P(play) = ?$
$P(Sky = sunny|play) = ?$
$P(Humid = high|play) = ?$
...

$P(\neg play) = ?$
$P(Sky = sunny|\neg play) = ?$
$P(Humid = high|\neg play) = ?$
...

Training Naïve Bayes - Example

Estimate $P(X_j|Y)$ and $P(Y)$ directly from the training data by counting!

Sky	Temp	Humid	Wind	Water	Forecast	Play?
sunny	warm	normal	strong	warm	same	yes
sunny	warm	high	strong	warm	same	yes
rainy	cold	high	strong	warm	change	no
sunny	warm	high	strong	cool	change	yes

$P(play) = ?$
$P(Sky = sunny|play) = ?$
$P(Humid = high|play) = ?$
...

$P(\neg play) = ?$
$P(Sky = sunny|\neg play) = ?$
$P(Humid = high|\neg play) = ?$
...

Training Naïve Bayes - Example

Estimate $P(X_j|Y)$ and $P(Y)$ directly from the training data by counting!

Sky	Temp	Humid	Wind	Water	Forecast	Play?
sunny	warm	normal	strong	warm	same	yes
sunny	warm	high	strong	warm	same	yes
rainy	cold	high	strong	warm	change	no
sunny	warm	high	strong	cool	change	yes

$P(play) = 3/4$
$P(Sky = sunny|play) =?$
$P(Humid = high|play) =?$
...

$P(\neg play) = 1/4$
$P(Sky = sunny|\neg play) =?$
$P(Humid = high|\neg play) =?$
...

Training Naïve Bayes - Example

Estimate $P(X_j|Y)$ and $P(Y)$ directly from the training data by counting!

Sky	Temp	Humid	Wind	Water	Forecast	Play?
sunny	warm	normal	strong	warm	same	yes
sunny	warm	high	strong	warm	same	yes
rainy	cold	high	strong	warm	change	no
sunny	warm	high	strong	cool	change	yes

$P(play) = 3/4$
$P(Sky = sunny|play) =?$
$P(Humid = high|play) =?$
...

$P(\neg play) = 1/4$
$P(Sky = sunny|\neg play) =?$
$P(Humid = high|\neg play) =?$
...

Training Naïve Bayes - Example

Estimate $P(X_j|Y)$ and $P(Y)$ directly from the training data by counting!

Sky	Temp	Humid	Wind	Water	Forecast	Play?
sunny	warm	normal	strong	warm	same	yes
sunny	warm	high	strong	warm	same	yes
rainy	cold	high	strong	warm	change	no
sunny	warm	high	strong	cool	change	yes

$P(play) = 3/4$
$P(Sky = sunny|play) = 1$
$P(Humid = high|play) = ?$
...

$P(\neg play) = 1/4$
$P(Sky = sunny|\neg play) = ?$
$P(Humid = high|\neg play) = ?$
...

Training Naïve Bayes - Example

Estimate $P(X_j|Y)$ and $P(Y)$ directly from the training data by counting!

Sky	Temp	Humid	Wind	Water	Forecast	Play?
sunny	warm	normal	strong	warm	same	yes
sunny	warm	high	strong	warm	same	yes
rainy	cold	high	strong	warm	change	no
sunny	warm	high	strong	cool	change	yes

$P(play) = 3/4$
$P(Sky = sunny|play) = 1$
$P(Humid = high|play) = ?$
...

$P(\neg play) = 1/4$
$P(Sky = sunny|\neg play) = ?$
$P(Humid = high|\neg play) = ?$
...

Training Naïve Bayes - Example

Estimate $P(X_j|Y)$ and $P(Y)$ directly from the training data by counting!

Sky	Temp	Humid	Wind	Water	Forecast	Play?
sunny	warm	normal	strong	warm	same	yes
sunny	warm	high	strong	warm	same	yes
rainy	cold	high	strong	warm	change	no
sunny	warm	high	strong	cool	change	yes

$P(play) = 3/4$
$P(Sky = sunny|play) = 1$
$P(Humid = high|play) = ?$
...

$P(\neg play) = 1/4$
$P(Sky = sunny|\neg play) = 0$
$P(Humid = high|\neg play) = ?$
...

Training Naïve Bayes - Example

Estimate $P(X_j|Y)$ and $P(Y)$ directly from the training data by counting!

Sky	Temp	Humid	Wind	Water	Forecast	Play?
sunny	warm	normal	strong	warm	same	yes
sunny	warm	high	strong	warm	same	yes
rainy	cold	high	strong	warm	change	no
sunny	warm	high	strong	cool	change	yes

$P(play) = 3/4$
$P(Sky = sunny|play) = 1$
$P(Humid = high|play) = ?$
...

$P(\neg play) = 1/4$
$P(Sky = sunny|\neg play) = 0$
$P(Humid = high|\neg play) = ?$
...

Training Naïve Bayes - Example

Estimate $P(X_j|Y)$ and $P(Y)$ directly from the training data by counting!

Sky	Temp	Humid	Wind	Water	Forecast	Play?
sunny	warm	normal	strong	warm	same	yes
sunny	warm	high	strong	warm	same	yes
rainy	cold	high	strong	warm	change	no
sunny	warm	high	strong	cool	change	yes

$P(play) = 3/4$
$P(Sky = sunny|play) = 1$
$P(Humid = high|play) = 2/3$
...

$P(\neg play) = 1/4$
$P(Sky = sunny|\neg play) = 0$
$P(Humid = high|\neg play) =?$
...

Training Naïve Bayes - Example

Estimate $P(X_j|Y)$ and $P(Y)$ directly from the training data by counting!

Sky	Temp	Humid	Wind	Water	Forecast	Play?
sunny	warm	normal	strong	warm	same	yes
sunny	warm	high	strong	warm	same	yes
rainy	cold	high	strong	warm	change	no
sunny	warm	high	strong	cool	change	yes

$P(play) = 3/4$
$P(Sky = sunny|play) = 1$
$P(Humid = high|play) = 2/3$
...

$P(\neg play) = 1/4$
$P(Sky = sunny|\neg play) = 0$
$P(Humid = high|\neg play) =?$
...

Training Naïve Bayes - Example

Estimate $P(X_j|Y)$ and $P(Y)$ directly from the training data by counting!

Sky	Temp	Humid	Wind	Water	Forecast	Play?
sunny	warm	normal	strong	warm	same	yes
sunny	warm	high	strong	warm	same	yes
rainy	cold	high	strong	warm	change	no
sunny	warm	high	strong	cool	change	yes

$P(play) = 3/4$
$P(Sky = sunny|play) = 1$
$P(Humid = high|play) = 2/3$
...

$P(\neg play) = 1/4$
$P(Sky = sunny|\neg play) = 0$
$P(Humid = high|\neg play) = 1$
...

Laplace smoothing

- Notice that some probabilities estimated by counting might be zero (possible overfitting!)
- Fix by using Laplace smoothing
  - Adds 1 to each counting

$$P(X_j = v|Y = y_k) = \frac{c_v + 1}{\sum_{v' \in values(X_j)} c_{v'} + |values(X_j)|}$$

where

- c_v is the count of training instance with a value of v for attribute j and class label y_k
- $|values(X_j)|$ is the number of values X_j can take on

Using the Naïve Bayes classifier

- Now, we have

$$P(Y = y_k | X = x_i) = \frac{P(Y = y_k) \prod_{j=1}^{d} P(X_j = x_{i,j} | Y = y_k)}{P(X = x_i)}$$

  $P(X = x_i)$ is a constant for a given instance, and so irrelevant to our prediction

- To classify a new data point x,

$$h(\mathrm{x}) = \arg\max_{y_k} P(Y = y_k) \prod_{j=1}^{d} P(X_j = x_j | Y = y_k)$$

  where x_j is the j^{th} attribute value of x

- ▷ In practice (coding), we use log-probabilities to prevent underflow

$$h(\mathrm{x}) = \arg\max_{y_k} [\log(P(Y = y_k)) + \sum_{j=1}^{d} \log(P(X_j = x_j | Y = y_k))]$$

The Naïve Bayes classifier algorithm (summary)

- For each class label y_k
  - Estimate $P(Y = y_k)$ from the data
  - For each value $x_{i,j}$ of each attribute X_i
    Estimate $P(X_i = x_{i,j}|Y = y_k)$
- classify a new data point via:

$$h(\mathrm{x}) = \arg\max_{y_k} \left[\log(P(Y = y_k)) + \sum_{j=1}^{d} \log(P(X_j = x_j|Y = y_k)) \right] \quad (1)$$

▷ In practice, the independence assumption does not often hold true, but Naïve Bayes performs very well despite it.

Naive Bayes

Advantages

- Fast to train
- Fast to classify
- Not sensitive to irrelevant features

Disadvantages

- Assumes features independence

Examples of applications:

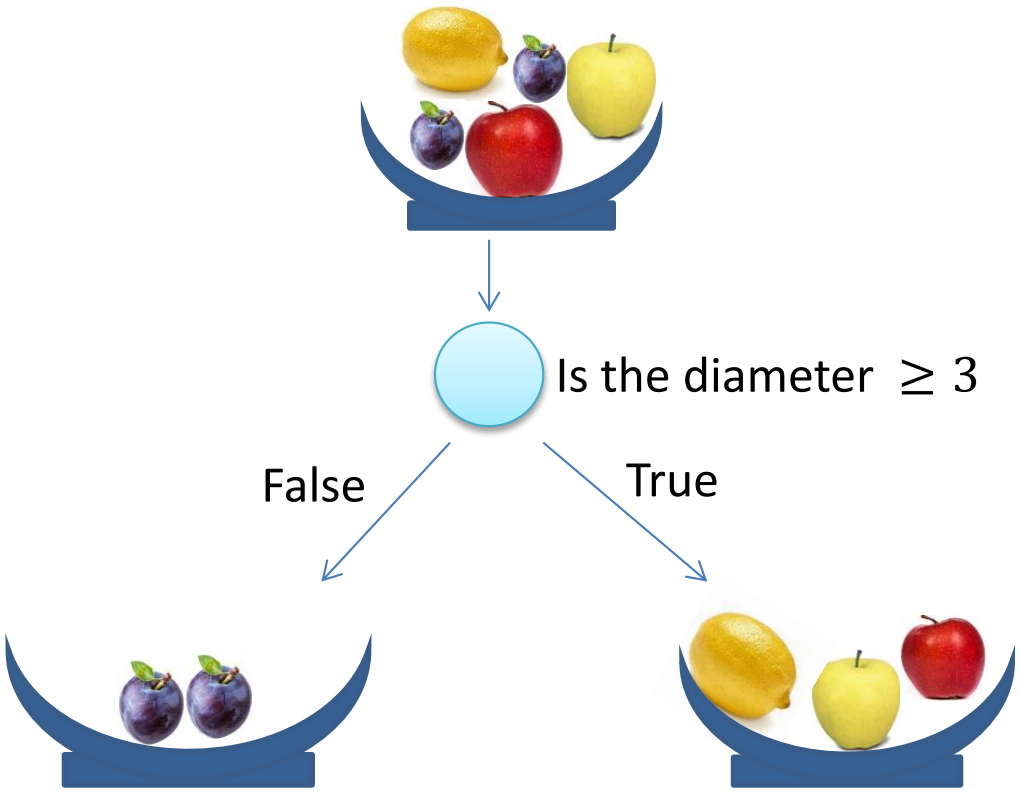
- Which e-mails are spam?
- Which e-mail are meeting notices?

Questions ?

Decision Trees

Decision Trees: what and why?

Decision Trees: what and why?



Decision Trees: what and why?

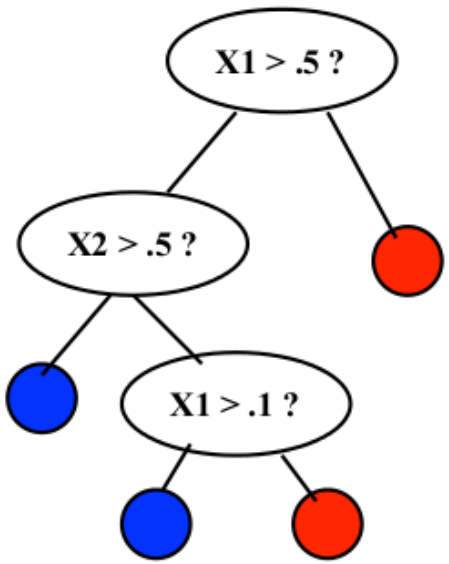
The problem: Given a set of training cases/objects and their attribute values, try to determine the target attribute value of new examples.

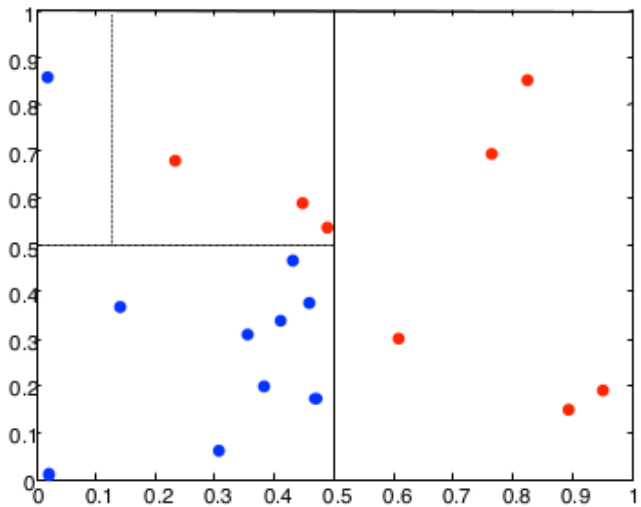
- Classification
- Prediction

Why decision tree?

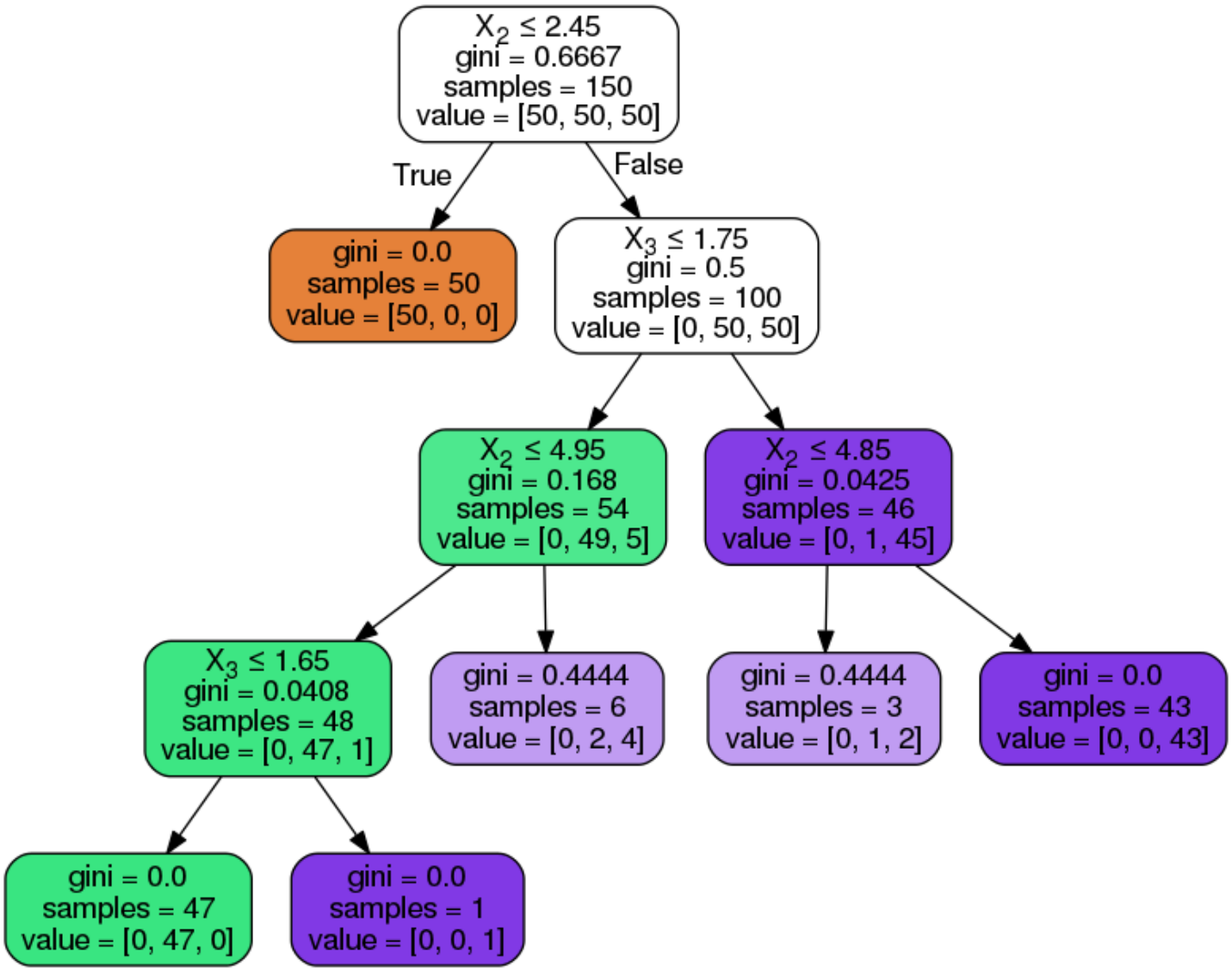
- Decision trees are powerful and popular tools for classification and prediction.
- Decision trees represent rules, which can be understood by humans and used in knowledge system such as database.

Decision Trees visualization





Decision Trees visualization



Definition

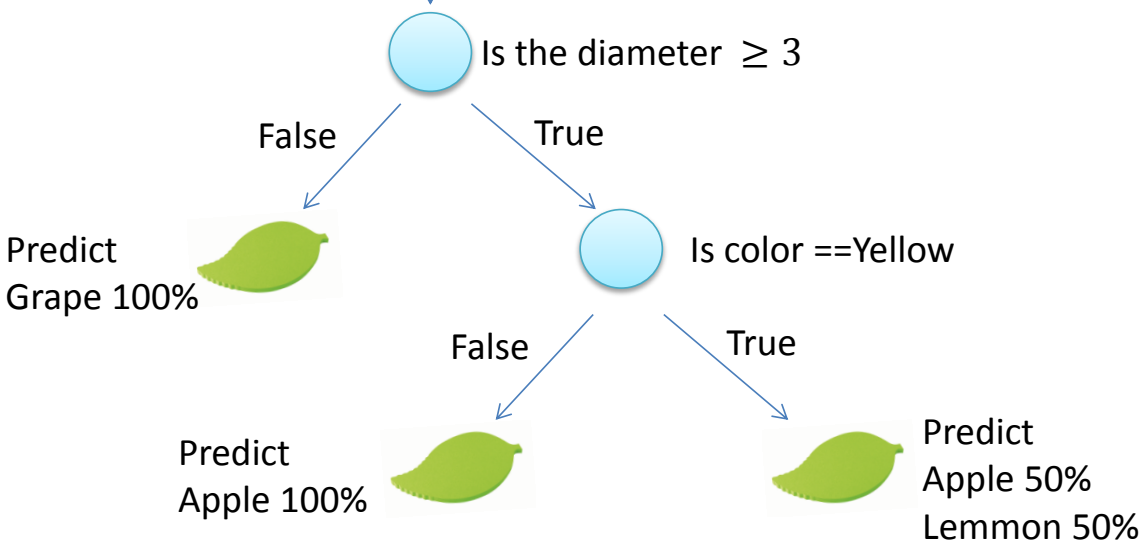
- Trees: key requirements
  - Attribute-value description: object or case must be expressible in terms of a fixed collection of properties or attributes (e.g., diameter, color, hot, mild, cold).
  - Predefined classes (target values): the target function has discrete output values (bollean or multiclass)
  - Sufficient data: enough training cases should be provided to learn the model.

- Decision tree is a classifier in the form of a tree structure
  - Decision node: specifies a test on a single attribute
  - Leaf node: indicates the value of the target attribute
  - Arc/edge: split of one attribute
  - Path: a disjunction of test to make the final decision

- Decision trees classify instances or examples by starting at the root of the tree and moving through it until a leaf node.

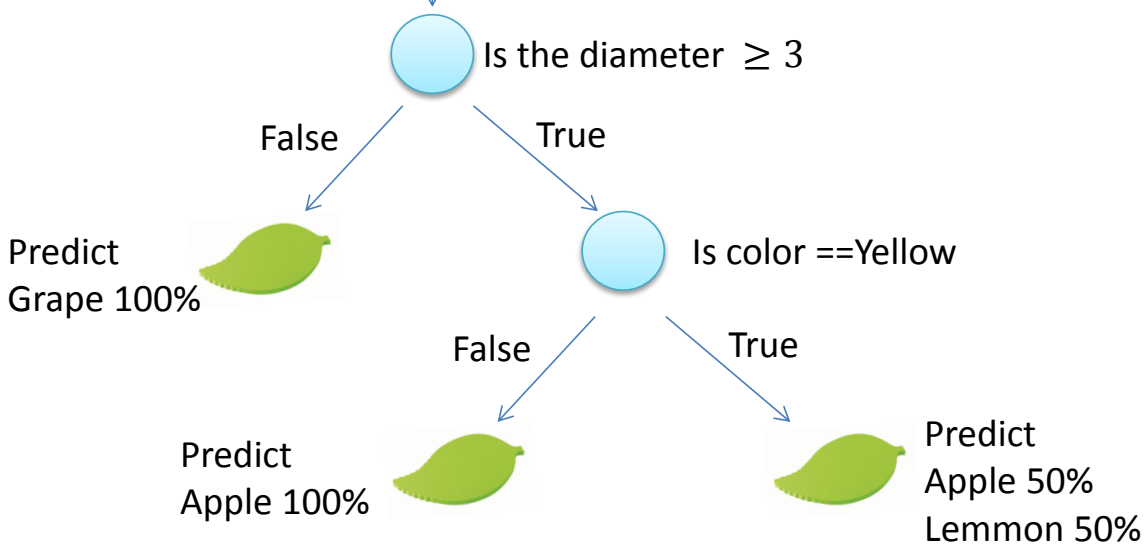
Identify: Decision node, Leaf node , Arc/edge, Path

Color	Diam	Label
Green	3	Apple
Yellow	3	Apple
Red	1	Grape
Red	1	Grape
Yellow	3	Lemon



Identify: Decision node, Leaf node , Arc/edge, Path

Color	Diam	Label
Green	3	Apple
Yellow	3	Apple
Red	1	Grape
Red	1	Grape
Yellow	3	Lemon



- Which questions to ask, and when?

CART

- Manny algorithms: ID3, C4.5, C5.0, CART, ...
- CART (Classification and Regression Trees) is introduced by Leo Breiman to refer to Decision Tree algorithms that can be used for classification or regression predictive modeling problems.
- Classically, this algorithm is referred to as "decision trees", but on some platforms like R they are referred to by the more modern term CART.
- The CART algorithm provides a foundation for important algorithms like bagged decision trees, random forest and boosted decision trees.

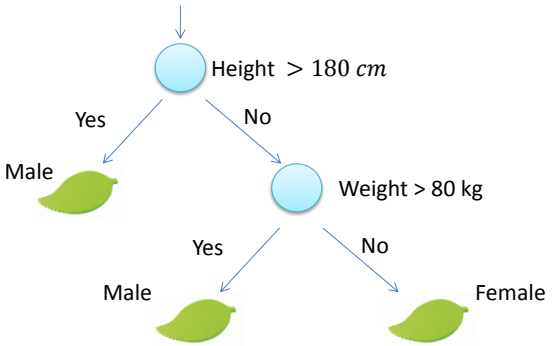
CART

- CART model representation
- Learn a CART Model From Data
- Greedy Splitting
  - CART for Regression
  - CART for Classification
- Stopping Criterion
- Pruning the Tree

CART model representation

- The representation for the CART model is a binary tree
- Each root node represents a single input variable (x) and a split point on that variable (assuming the variable is numeric).
- The leaf nodes of the tree contain an output variable (y) which is used to make a prediction.
- CART does not require any special data preparation
- The tree can be stored to file as a graph or a set of rules.

CART model representation -example

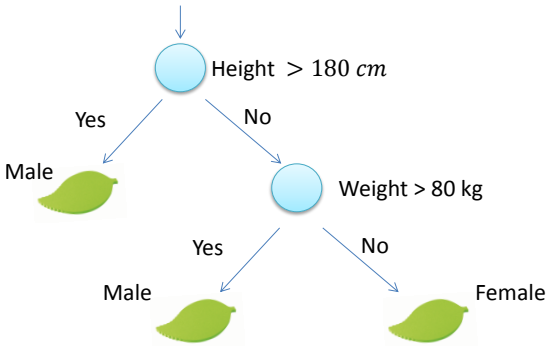


Given a dataset with two inputs (x) of height in centimeters and weight in kilograms predict the output of sex as male or female.

Storing the decision tree as a set of rules

1. If Height $>$ 180 cm Then Male
2. If Height $<=$ 180 cm AND Weight $>$ 80 kg Then Male
3. If Height $<=$ 180 cm AND Weight $<=$ 80 kg Then Female
4. Make Predictions With CART Models

CART model representation -example



Given a dataset with two inputs (x) of height in centimeters and weight in kilograms predict the output of sex as male or female.

Storing the decision tree as a set of rules

1. If Height $>$ 180 cm Then Male
2. If Height $<=$ 180 cm AND Weight $>$ 80 kg Then Male
3. If Height $<=$ 180 cm AND Weight $<=$ 80 kg Then Female
4. Make Predictions With CART Models

Ex: Given a new input [height = 160 cm, weight = 65 kg]

1. Height $>$ 180 cm: No
2. Weight $>$ 80 kg: No
3. Therefore: Female

Learn a CART Model From Data

- Creating a CART model involves selecting input variables and split points on those variables until a suitable tree is constructed.
- The selection of which input variable to use and the specific split or cut-point is chosen using a greedy algorithm to minimize a cost function.
- Greedy Splitting $\mapsto$ recursive binary splitting.
- Then the split with the best cost (lowest cost because we minimize cost) is selected.

Greedy Splitting

CART for Regression

For regression predictive modeling problems the cost function that is minimized to choose split points is the sum squared error across all training samples that fall within the class.

CART for Classification

For classification the Gini index [Corrado Gini, 1912] is used which provides an indication of how "pure" the leaf nodes are (how mixed the training data assigned to each node is).

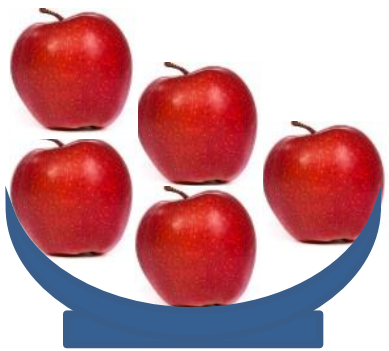
$$GINI(N) = 1 - \sum_{j} [p(j|t)]^2$$

where G is the Gini index over all classes, $p(j|N)$ is the relative frequency of class j at node N.

Quantifying uncertainty

- The best question is the one which reduce the uncertainty the most.
- Gini impurity quantifies how much uncertainty is in a node.
- Information gain quantifies how much that question is reducing the error.

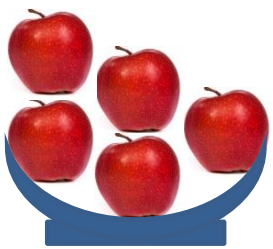
Low impurity

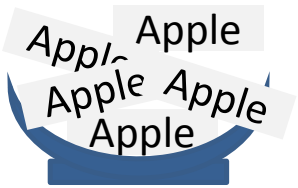


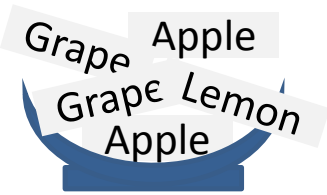
High impurity

Quantifying uncertainty

Impurity: Chance of being incorrect if you randomly assign a label to an example in the set





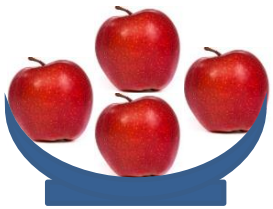


Quantifying uncertainty

Impurity: Chance of being incorrect if you randomly assign a label to an example in the set

Impurity = 0

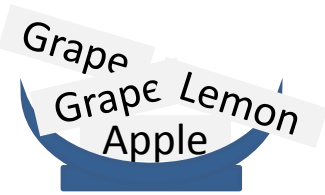
Apple



Impurity = 0.64

$1-(2/5)^2-(2/5)^2-(1/5)^2$

Apple

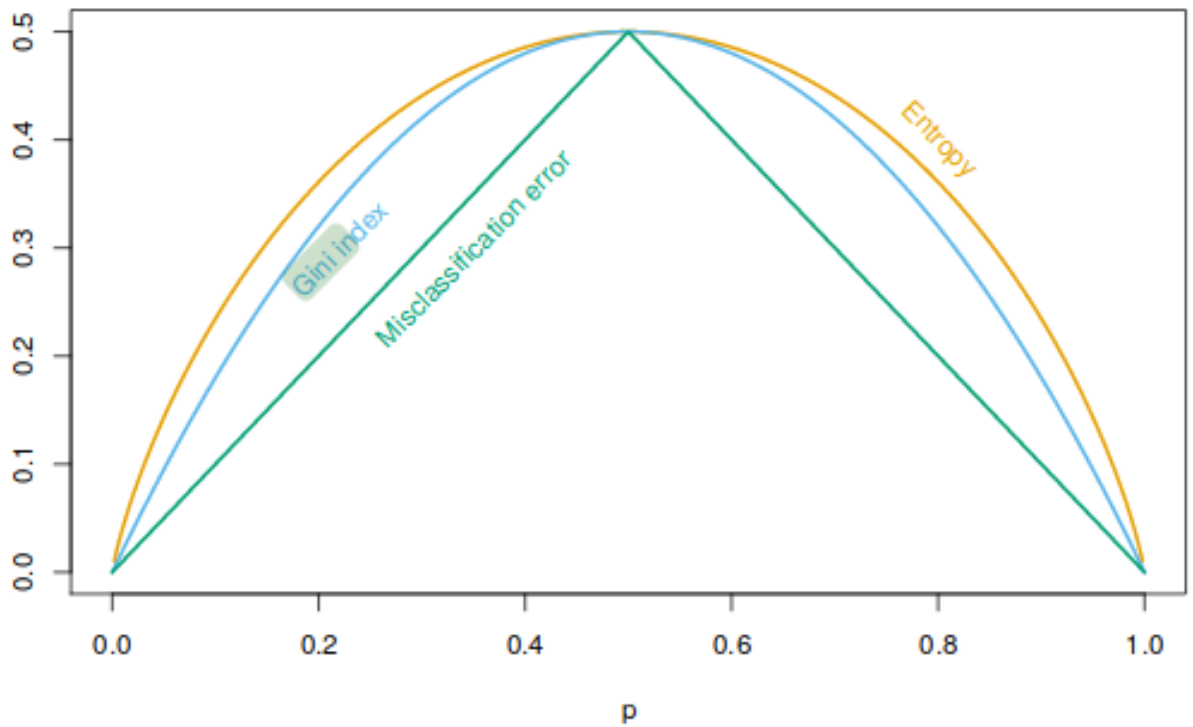


Query Selection and Node Impurity

- $p(j|N)$ proportion of class j samples from the total amount of samples at node N.
- Node impurity $i(N)$ is 0 when all patterns at the node are of the same class; Impurity becomes maximum when all the classes at node N are equally likely.
- Entropy impurity
$$i(N) = -\sum_j p(j|N)\log_2 p(j|N)$$
- Gini impurity Expected error rate at node N if the category label is selected randomly from the class distribution present at N.
$$i(N) = 1 - \sum_j [p(j|N)]^2$$
- Misclassification impurity Minimum probability that a training pattern will be misclassified at N.
$$i(N) = 1 - \max_j\ p(j|N)$$

Query Selection and Node Impurity

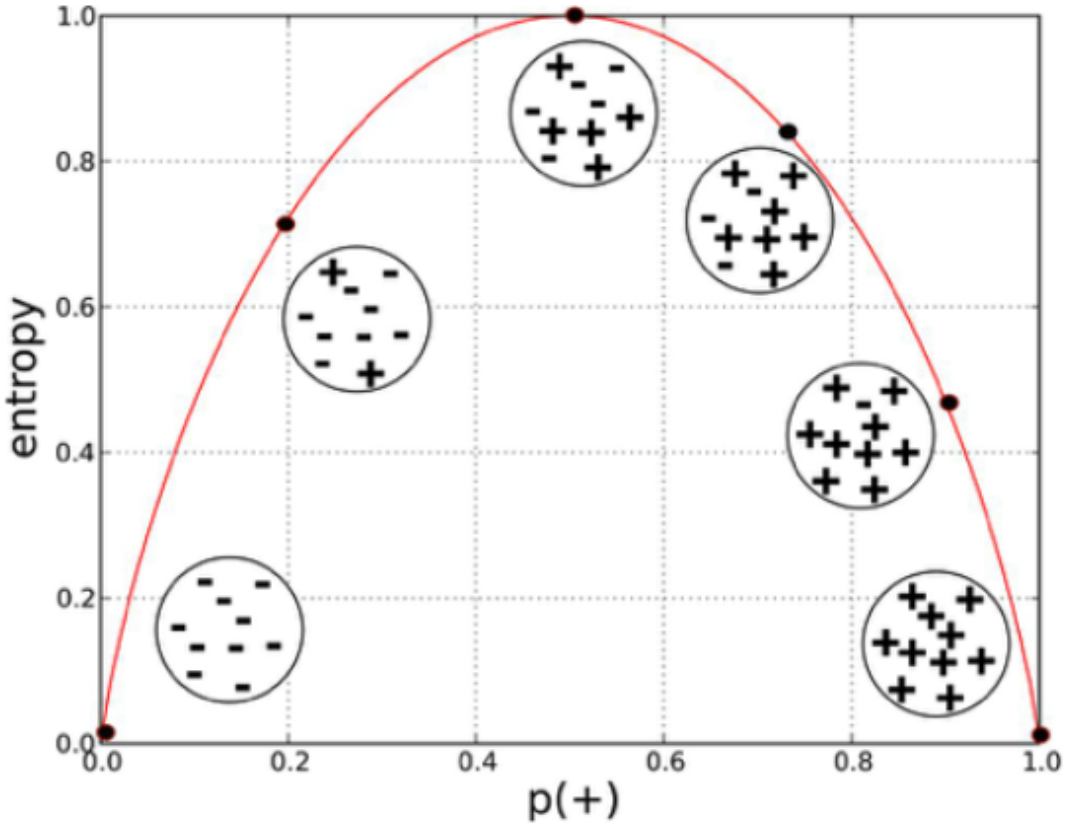
Node impurity measures for two-class classification, as a function of the proportion in class 2. Entropy has been scaled to pass through $(0.5, 0.5)$.



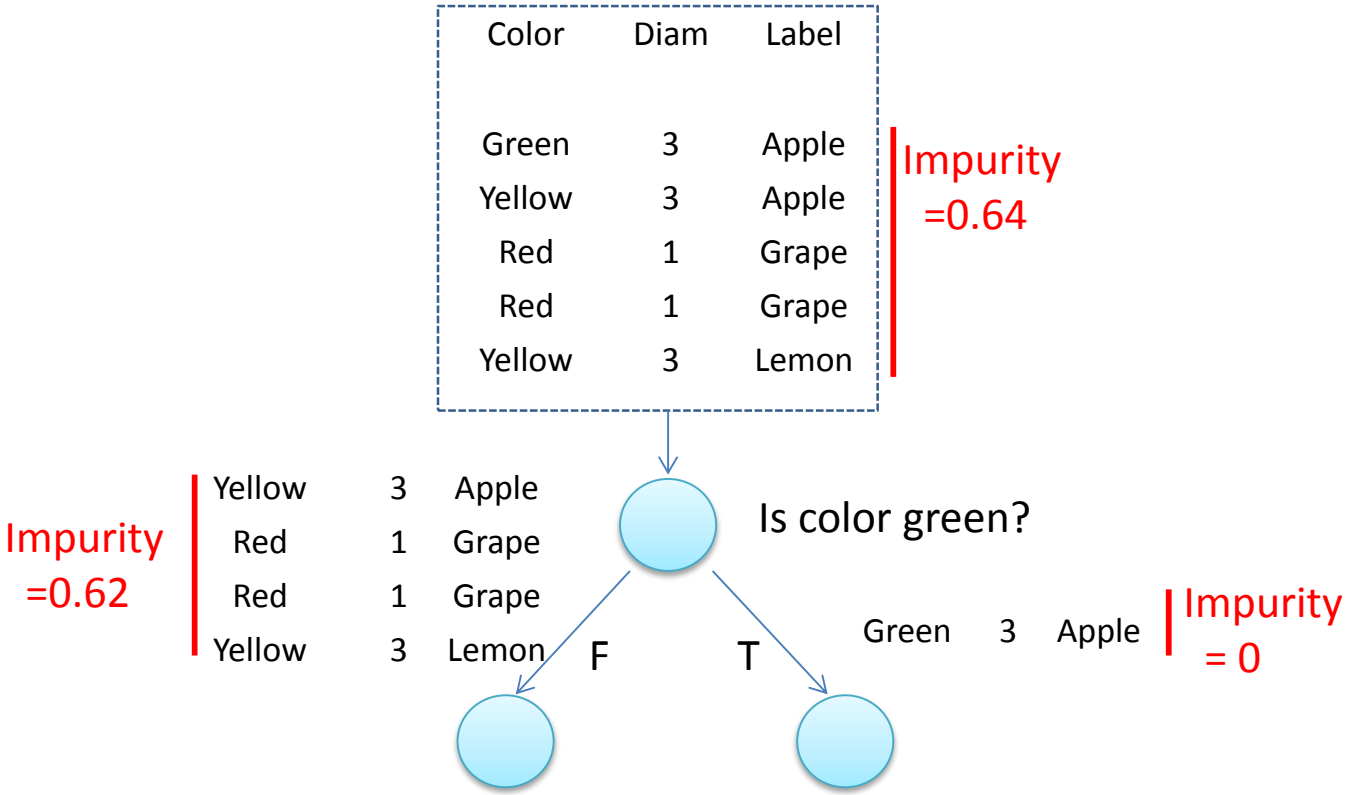
Source: Trevor Hastie et al., *Elements of Statistical Learning*, 2009.

Query Selection and Node Impurity

Entropy: Visual insights



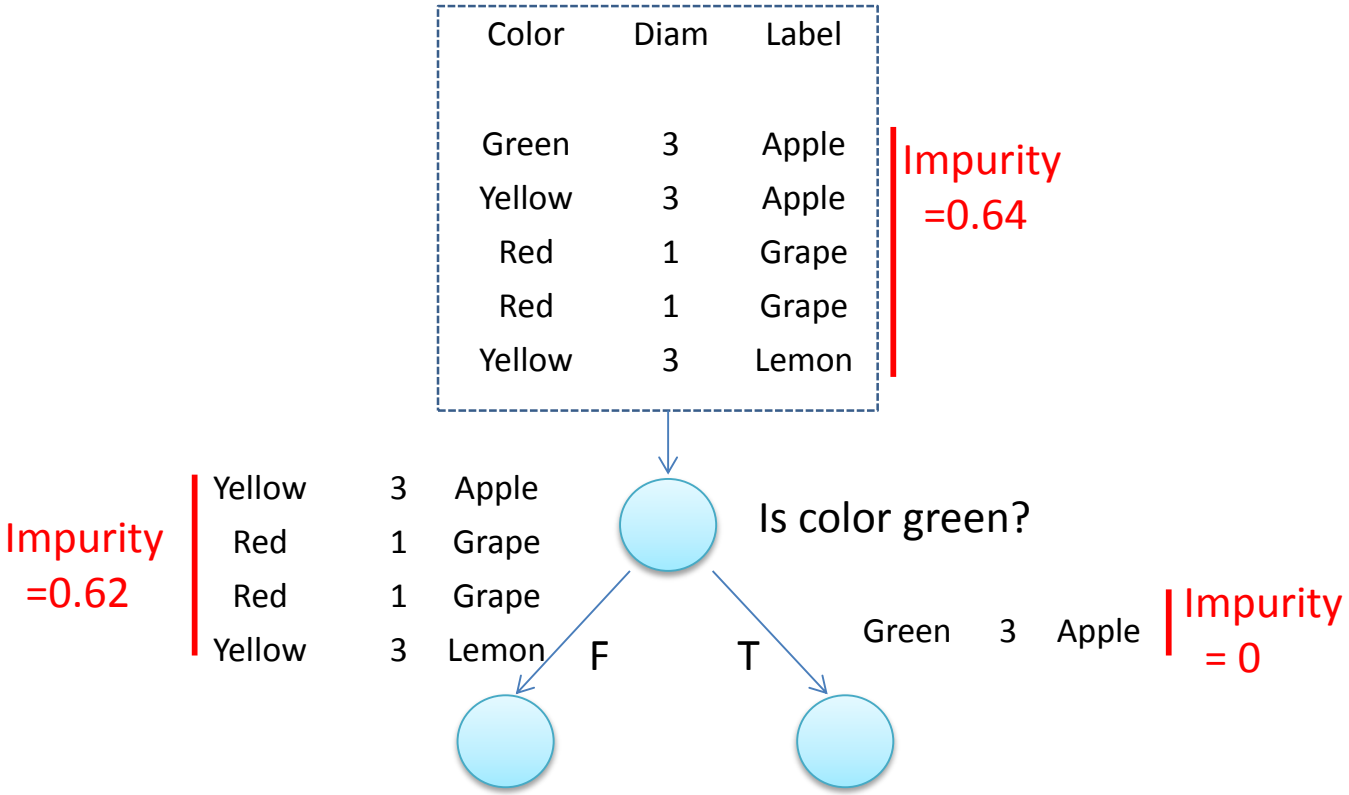
CART example (Gini impurity)



$$\text{Impurity}_{root} = 1 - (2/5)^2 - (2/5)^2 - (1/5)^2 = 0.64$$

$$\text{Average Impurity} = 4/5 \times 0.62 + 1/5 \times 0 \approx 0.5$$

CART example (Gini impurity)



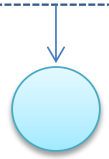
$$\text{Impurity}_{root} = 1 - (2/5)^2 - (2/5)^2 - (1/5)^2 = 0.64$$

$$\text{Average Impurity} = 4/5 \times 0.62 + 1/5 \times 0 \approx 0.5$$

$$\text{Information gain} = 0.64 - 0.5 = 0.14$$

CART example

Color	Diam	Label
Green	3	Apple
Yellow	3	Apple
Red	1	Grape
Red	1	Grape
Yellow	3	Lemon



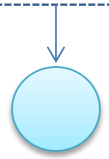
Information Gain

Question	Gain
Color==Green?	0.14
Diameter >=3	0.37
Color==Yellow	0.17
Color==Red	0.37
Diameter>=1	0

- At this specific node, repeat the calculation of the average impurity and information gain for all possible questions.

CART example

Color	Diam	Label
Green	3	Apple
Yellow	3	Apple
Red	1	Grape
Red	1	Grape
Yellow	3	Lemon

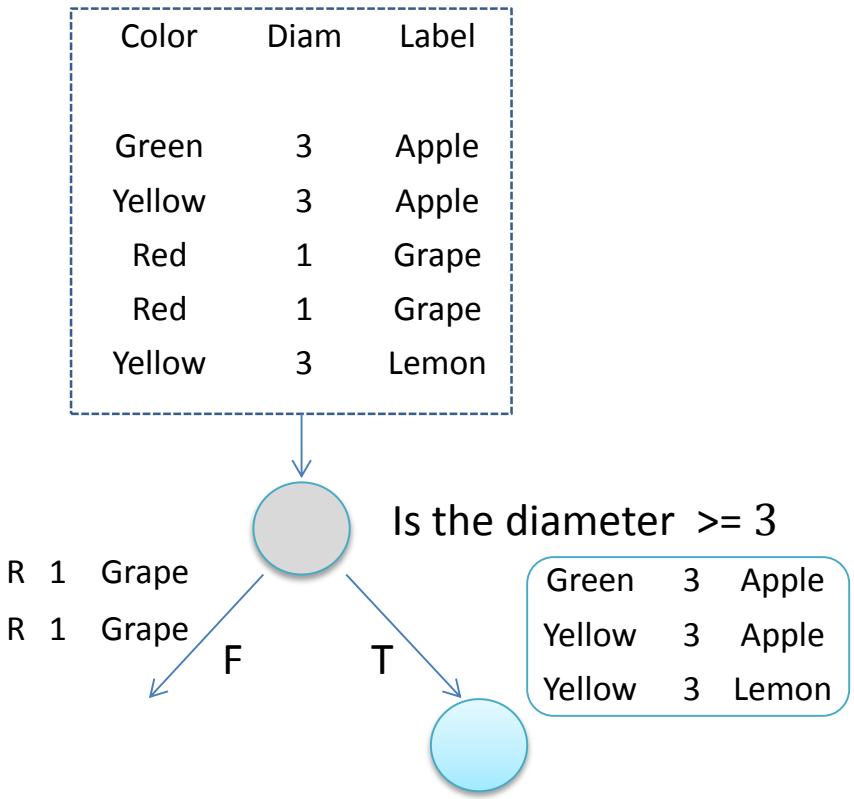


Information Gain

Question	Gain
Color==Green?	0.14
Diameter >=3	0.37
Color==Yellow	0.17
Color==Red	0.37
Diameter>=1	0

- At this specific node, repeat the calculation of the average impurity and information gain for all possible questions.
- Keep track of the question which produce the most gain
- Slit the data using that question

CART example



- Move to a new node
  - At this specific node, repeat the calculation of the average impurity and information gain for all possible questions.
  - Keep track of the question which produce the most gain
  - Slit the data using that question
- Repeat until a stopping criterion

Ending Tree Growth

- Models induce a tree by recursively selecting and subdividing attributes
  - random selection - noisy variables
  - inefficient production of inaccurate trees
- Efficient models
  - examine each variable
  - determine which will improve accuracy of entire tree
  - problem - this approach decides best split without considering subsequent splits
- Grow the tree until
  - additional splitting produces no significant information gain
  - statistical test - a χ^2 test
  - problem - trees that are too small
  - only compares one split with the next descending split

Discrete vs. Continuous Attributes

- Attribute Types
  - Boolean, Nominal, Ordinal, Integer, Continuous
- Continuous variables attributes - problems for decision trees
  - increase computational complexity of the task
  - promote prediction inaccuracy
  - lead to overfitting of data
- Convert continuous variables into discrete intervals
  - "greater than or equal to" and "less than"
  - optimal solution for conversion
  - difficult to determine discrete intervals ideal (e.g. size, number)

▷ Sort by value, then find best threshold for binary split

▷ Cluster into n intervals and do n-way split

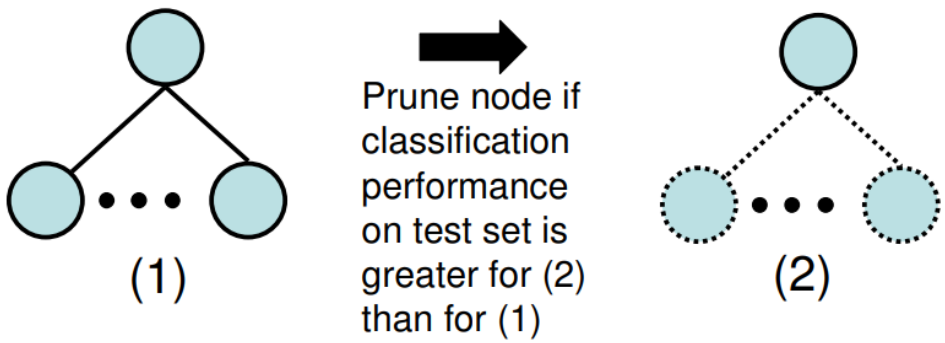
Pruning Trees

- There is another technique for reducing the number of attributes used in a tree $\rightarrow$ pruning
- Helps avoiding tree overfitting
- Two types of pruning:
  1. pre-pruning (forward pruning). We decide during the building process when to stop adding attributes (possibly based on their information gain). We can incorporate the preference of learning shorter trees within the tree growing process by imposing a limit on:
     - Maximum number of leaf nodes
     - Maximum depth of the tree
     - Minimum number of training instances at a leaf node

     (!) Sometimes attributes individually do not contribute much to a decision, but combined, they may have a significant impact
  2. post-pruning (backward pruning) waits until the full decision tree has built and then prunes the attributes. Two techniques:
     - Subtree Replacement
     - Subtree Raising

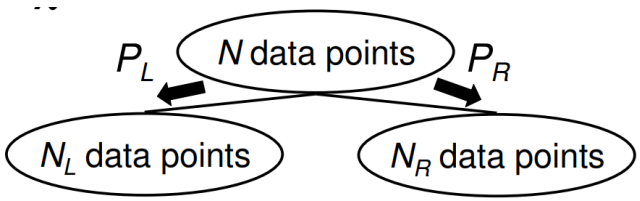
Decision Tree Pruning

- Construct the entire tree.
- Grow tree based on training data (unpruned tree)
- Start at the leaves, recursively eliminate splits
  1. Evaluate performance of the tree on test data (validation data)
  2. Prune the tree if the classification performance increase by removing the split



- Prune the tree by removing useless nodes based on:
  - Additional test data (not used for training)
  - Statistical significance tests (e.g. Chi-square criterion)

χ^2 Criterion: General case



- χ^2 Criterion: General case $\rightarrow$ K is a summation over all classes i, children j

$$K = \sum_{i,j} \frac{(N_{ij} - N'_{ij})^2}{N'_{ij}} \qquad (2)$$

Where

- N_{ij} = Number of points from class i in child j
- N'_{ij} = Number of points from class i in child j assuming a random selection
- $N'_{ij} = N_i \times P_j$

Small (Chi-square) values indicate low statistical significance $\rightarrow$ Remove the splits that are lower than a threshold $K < t$.

- Lower t $\rightarrow$ bigger trees (more overfitting).
- Larger t $\rightarrow$ smaller trees (less overfitting, but worse classification error).

Summary: Decision tree learning

- Inducing decision trees is one of the most widely used learning methods in practice
- Can out-perform human experts in many problems
- Strengths include
  - Fast
  - Simple to implement
  - Can convert results to a set of easily interpretable rules
  - Empirically valid in many commercial products
- Weaknesses include
  - Univariate splits using only one attribute at a time so limits types of trees
  - Large decisions trees may be hard to understand
  - Requires fixed length feature vectors
  - Non-incremental (i.e. batch method)

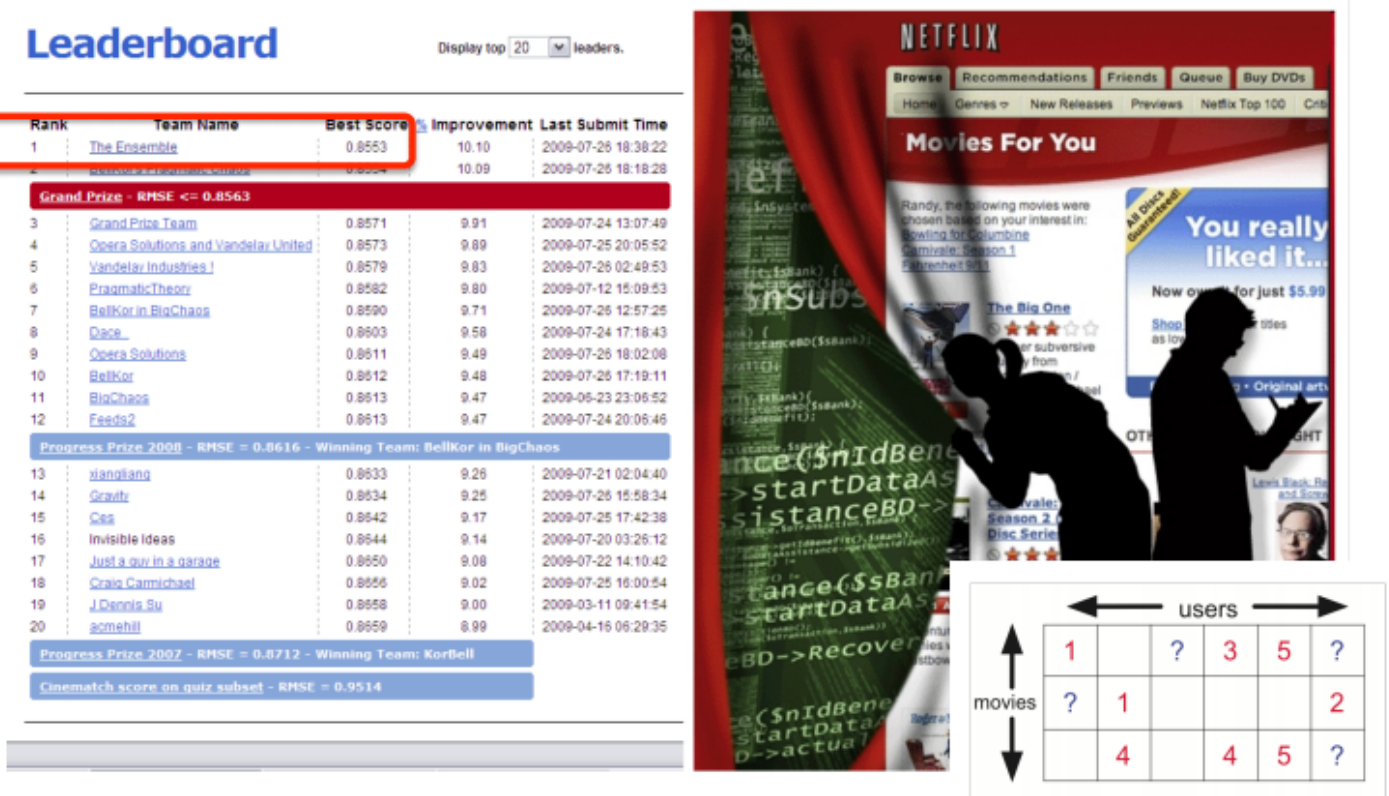
Summary: Decision tree learning

	Splitting criteria	Attribute type	Missing values	Pruning Strategy	Outlier Detection
ID3	Entropy Info. Gain	Categorical value	No	No pruning	Susceptible to outliers
CART	Gini Index	Categorical & numerical value	Yes	Cost-Complexity pruning	Can handle outliers
C4.5	Gain ratio Criteria	Categorical numerical value	Yes	Error Based pruning	Susceptible to outliers

From Decision Trees to Ensemble methods

- October 2006, Netflix: Make the company's recommendation engine 10% more accurate
- The data set: over 100 million ratings of 17,770 movies from 480,189 customers

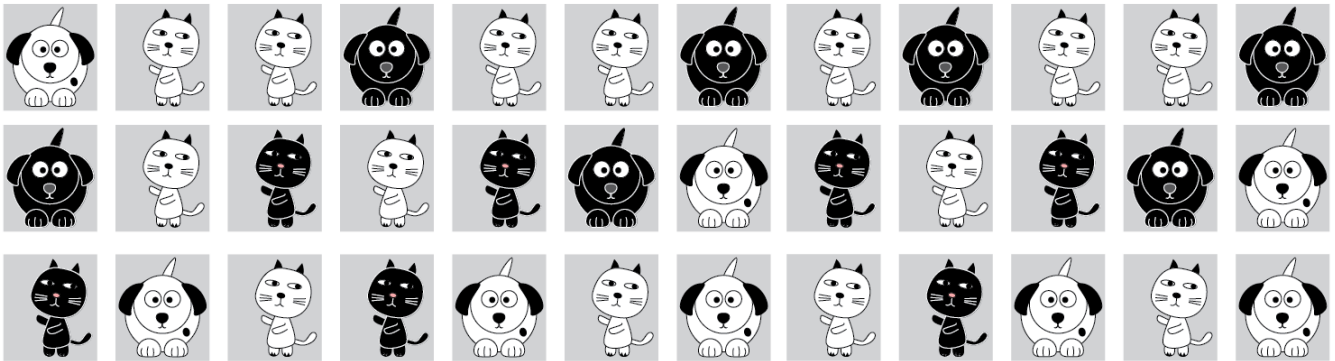
Machine learning competition with a $1 million prize



Classification Evaluation

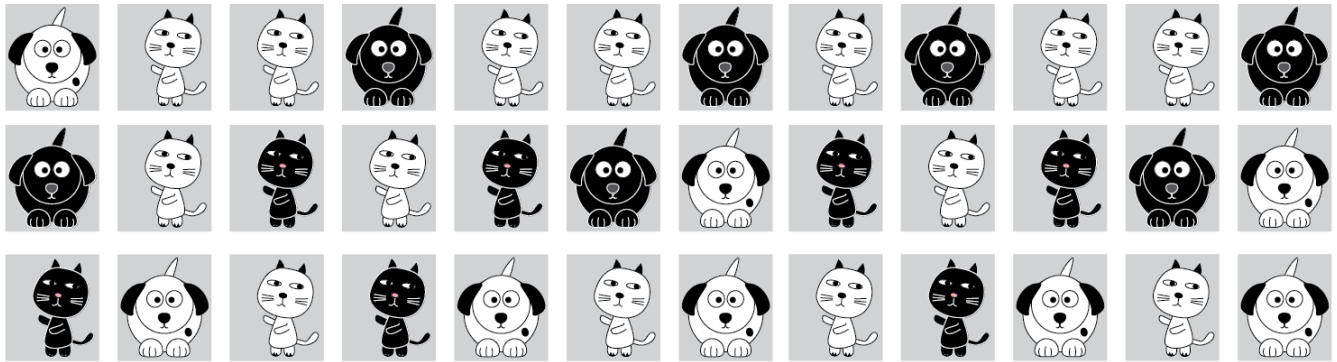
Confusion matrix

- Given a dataset

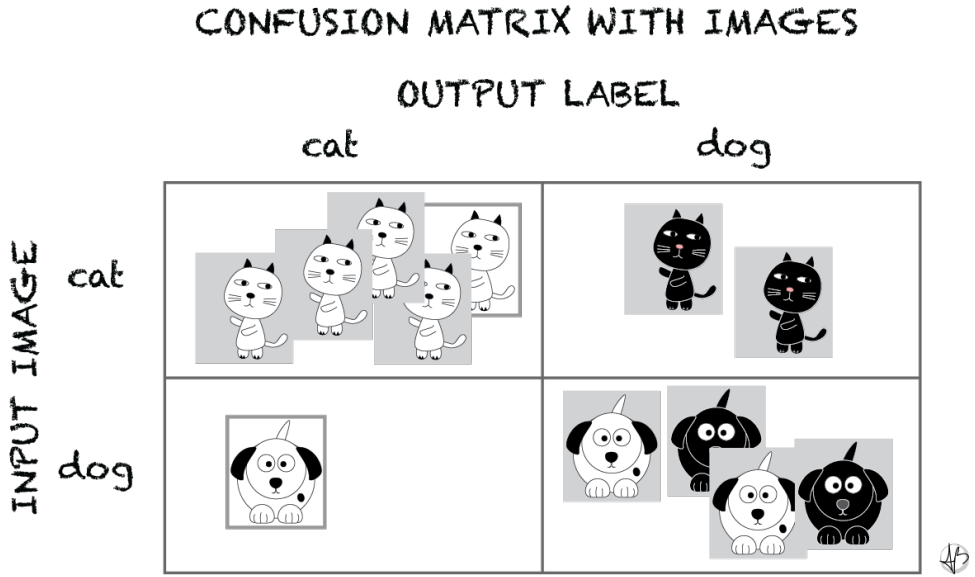


Confusion matrix

- Given a dataset



- Build a model able to classify cat versus dog.



Confusion matrix - intermezzo

False Positives?

False Positive

False Negative

Confusion matrix and various metrics

		Assigned class	
		Positive	*Negative*
Actual Class	*Positive*	TP	FN
	Negative	FP	TN

- Accuracy $= \frac{TP+TN}{TP+TN+FP+FN}$
- Sensitivity $=$True positive rate $=$ Recall $= \frac{TP}{TP+FN}$
- Specificity $=$True negative rate $= \frac{TN}{TN+FP}$
- Positive predictive value $=$ Precision $= \frac{TP}{TP+FP}$
- Negative predictive value $= \frac{TN}{TN+FN}$
- F1-score $= 2 \cdot \frac{Precision \cdot Recall}{Precision+Recall}$

- TP (True Positive)
- FP (False Positive)
- FN (False Negative)
- TN (True Negative)

Confusion matrix and various metrics

- Receiver Operating Characteristic (ROC) curve - plotting the true positive rate (TPR) against the false positive rate (FPR) at various threshold settings.
- Why we need so many evaluation metrics?
  - every data science applications area has his particularities
  - just one evaluation metrics is often not enough!

$$\begin{pmatrix} 10 & 0 \\ 0 & 1000 \end{pmatrix}$$

Accuracy=100%

$$\begin{pmatrix} 0 & 10 \\ 0 & 1000 \end{pmatrix}$$

Accuracy=1000/1010=99%

▷ Class Imbalance Problem !!!

Clustering

Clustering Algorithms

The problem of Clustering

- Given a set of points, with a notion of distance between points, group the points into some number of clusters, so that
  - Members of a cluster are close/similar to each other
  - Members of different clusters are dissimilar
- Usually:
  - Points are in a high-dimensional space
  - Similarity is defined using a distance measure (e.g. Euclidean, Cosine, Jaccard, edit distance, . . .)

K-means Algorithm: Objective

- The sum of squared errors (SSE) scoring function is defined as

$$SSE(C) = \sum_{i=1}^{k} \sum_{x_j \in C_i} ||x_j - \mu_i||^2$$

- The goal is to find the clustering that minimizes the SSE score:

$$C^* = \arg\min_{C} \{SSE(C)\}$$

- K-means employs a greedy iterative approach to find a clustering that minimizes the SSE objective. As such it can converge to a local optima instead of a globally optimal clustering.

K-means Algorithm: Objective

- K-means initializes the cluster means by randomly generating k points in the data space. Each iteration of K-means consists of two steps:
  1. cluster assignment
  2. centroid update

K-means Algorithm: Objective

- K-means initializes the cluster means by randomly generating k points in the data space. Each iteration of K-means consists of two steps:
  1. cluster assignment
  2. centroid update
- Given the k cluster means, in the cluster assignment step, each point $x_j \in D$ is assigned to the closest mean, which induces a clustering, with each cluster C_i comprising points that are closer to μ_i than any other cluster mean. That is, each point x_j is assigned to cluster C_{j^*}, where

$$C^* = \arg\min_{C}\{SSE(C)\}$$

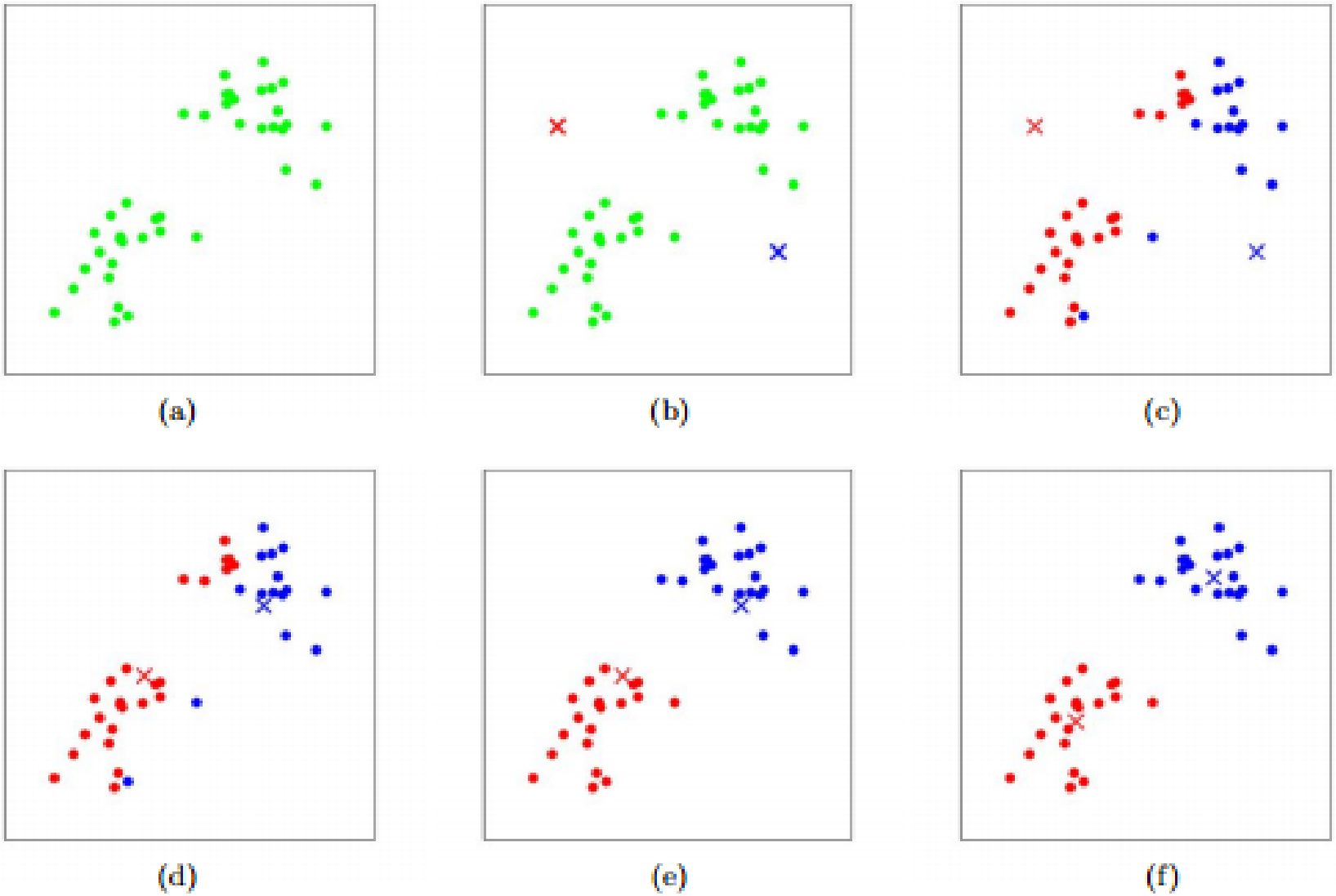
K-means Algorithm: Objective

- K-means initializes the cluster means by randomly generating k points in the data space. Each iteration of K-means consists of two steps:
  1. cluster assignment
  2. centroid update
- Given the k cluster means, in the cluster assignment step, each point $x_j \in D$ is assigned to the closest mean, which induces a clustering, with each cluster C_i comprising points that are closer to μ_i than any other cluster mean. That is, each point x_j is assigned to cluster C_{j^*}, where

$$C^* = \arg\min_{C}\{SSE(C)\}$$

- Given a set of clusters C_i, $i = 1, \ldots, k$, in the centroid update step, new mean values are computed for each cluster from the points in C_i.
- The cluster assignment and centroid update steps are carried out iteratively until we reach a fixed point or local minima.

K-means Algorithm



K-means Algorithm

1: K-means (D, k, ϵ)
2: $t = 0$
3: Randomly initialize k centroids: $\mu_1^t, \mu_2^t, ..., \mu_k^t \in R^d$
4: **repeat**
5: $t \leftarrow t + 1$
6: $C_j \leftarrow \emptyset$ for all $j = 1, ..., k$
7: **for** $x_j \in D$ **do**
8: $j \leftarrow arg\ \min_i ||x_j - \mu_i^t||^2$
9: $C_{j^*} \leftarrow C_{j^*} \cup \{x_i\}$
10: **end for**
11: **for** $i = 1$ to k **do**
12: $\mu \leftarrow \frac{1}{|C_i|} \sum_{x_j \in C_j} x_j$
13: **end for**
14: **until** $\sum_{i=1}^{k} ||\mu_i^t - \mu_i^{t+1}||^2 \leq \epsilon$

K-means Algorithm

1: K-means (D, k, ϵ)
2: $t = 0$
3: Randomly initialize k centroids: $\mu_1^t, \mu_2^t, ..., \mu_k^t \in R^d$
4: **repeat**
5: $t \leftarrow t + 1$
6: $C_j \leftarrow \emptyset$ for all $j = 1, ..., k$
7: **for** $x_j \in D$ **do**
8: $j \leftarrow arg\,\min_i ||x_j - \mu_i^t||^2$ ▷ step 1: cluster assignment
9: $C_{j^*} \leftarrow C_{j^*} \cup \{x_i\}$
10: **end for**
11: **for** $i = 1$ to k **do**
12: $\mu \leftarrow \frac{1}{|C_i|} \sum_{x_j \in C_j} x_j$ ▷ step 2: centroid update
13: **end for**
14: **until** $\sum_{i=1}^{k} ||\mu_i^t - \mu_i^{t+1}||^2 \leq \epsilon$

Clustering Algorithms

- Hierarchical Clustering
- Density-based Clustering
- Spectral Clustering
- Dynamic Time Warping
- Applications

Conclusions

Data Mining / Machine Learning in practice

- Repeat until ☺
  - Understand domain, prior knowledge, and goals
  - Data integration, selection, cleaning, pre-processing, etc.
  - Learn models
  - Interpret results
  - Consolidate and deploy discovered knowledge

What have we learned this lecture?

- Classification
  - Naive Bayes
  - Decision Trees
- Clustering
  - k-means

What have we learned this lecture?

- Classification
  - Naive Bayes
  - Decision Trees
- Clustering
  - k-means
- Supervised versus unsupervised learning
- What next..
  - Practical assignments
  - Project

Recommended videos on YouTube

- Bayes theorem, the geometry of changing beliefs
- Patrick Winston, MIT course, 11. Learning: Identification Trees, Disorder