LING 506 - TOPICS IN COMPUTATIONAL LINGUISTICS

Introductory Machine Learning

Yan Tang

Department of Linguistics, UIUC

Week 6

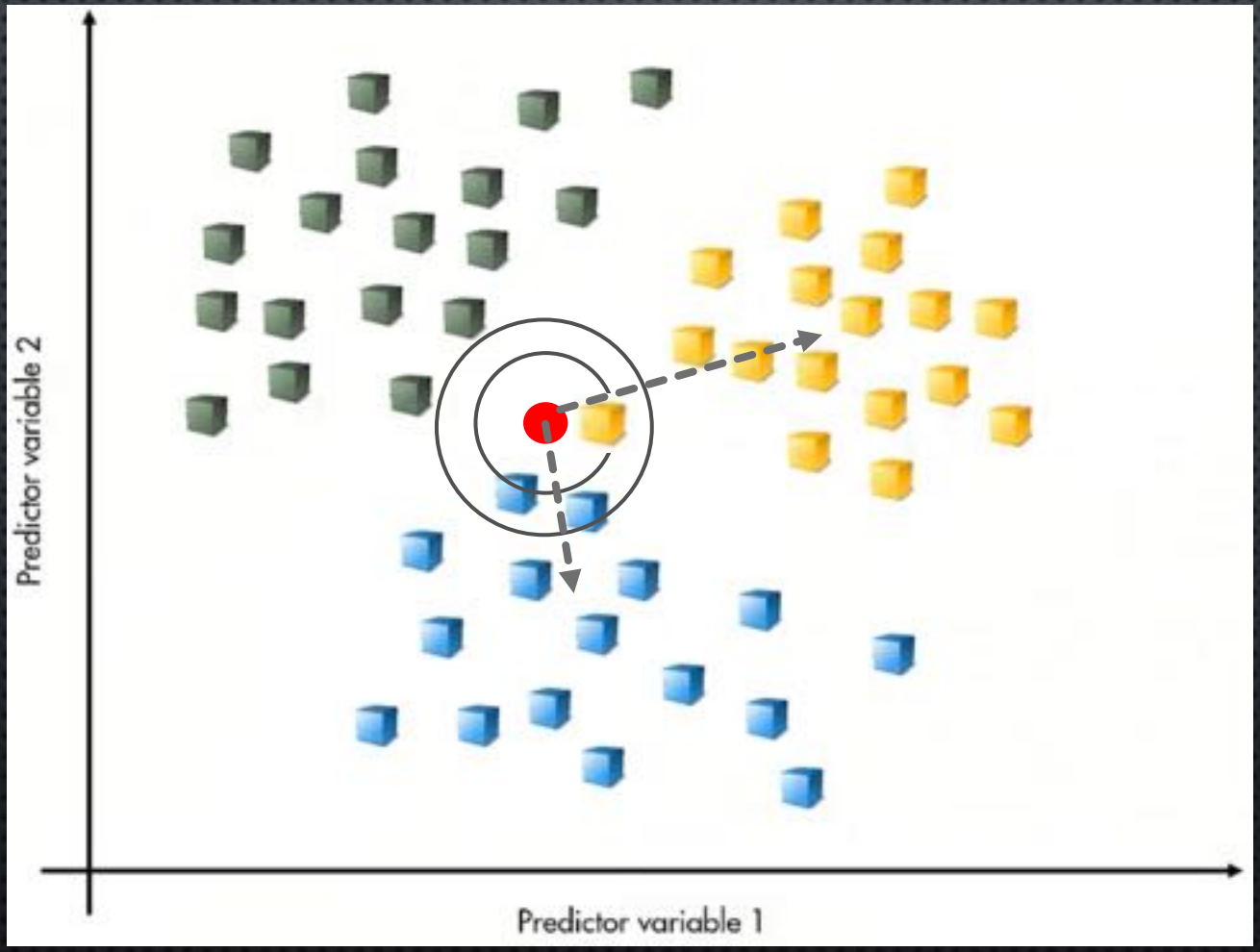
Last week…

- General steps to build a supervised ML system
  - Prepare data
    - Dealing with missing data
    - Dealing with text and categorial data
    - Data scaling and partition
  - Choose algorithm
    - Several aspects to consider, e.g. complexity, accuracy, scalability, etc.
  - Train model
    - Performance measure/cost function

Last week…

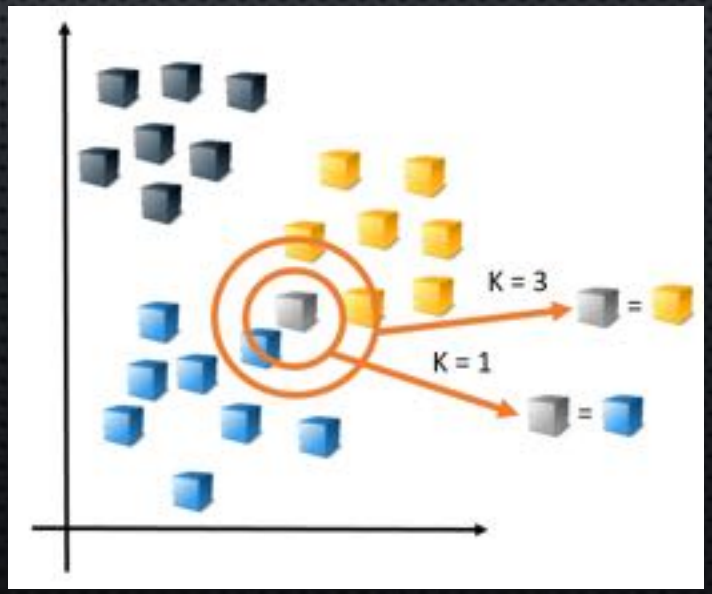
- Evaluate/validate and update model
  - Resubstituion
  - Hold out
  - Hyperparameter tuning
- Deploy model; make predictions

K-nearest neighbours



K-nearest neighbours (KNN)

- The simplest ML algorithm (arguably!)
- Only memorise training data instead of learning discriminative function
  - It doesn't "learn"…
  - Instance-based, lazy and non-parametric learning



K-nearest neighbours: principle

- The algorithm remembers the locations for all training data in the feature space
- Given a new observation and its location, it looks for the number of K nearest neighbouring data points according to their distances
  - Euclidean distance, Manhattan distance, etc

K-nearest neighbours: principle

- If K > 1, voting is used to assign the new observation to the class which the majority of the NN belong to

- Class probability, p_n :

$$p_n = \frac{C_n}{K}, \sum_{n=1}^{N} p_n = 1$$

C_n: number of samples in class n

K: number of nearest neighbours (neighbouring observations)

N: total number of classes

K-nearest neighbor: how many neighbours?

Considering the following situations:

Number of class: 3. [A, B, C]

When K = 1, $d_A = d_B$, $d_B = d_C$ or $d_A = d_c$

When K= 2, NNs = [A, B], [A, C] or [B, C]

When K = 3, NNs = [A, B, C]

When K = 4, NNs = [A, A, B, B], [B, B, C,C], etc

…

? How to break the tie?

K-nearest neighbours : how many neighbours?

If tie happens:

- Increase or decrease K
- Use different distance metric
- Add another feature
- Weight neighbours according to distance
- Assign a class stochastically
- Other empirical rules:
  - Even number of class ⇨ odd K
  - Odd number of class ⇨ even K

K-nearest neighbours : data requirement

Good practice for data preparation:

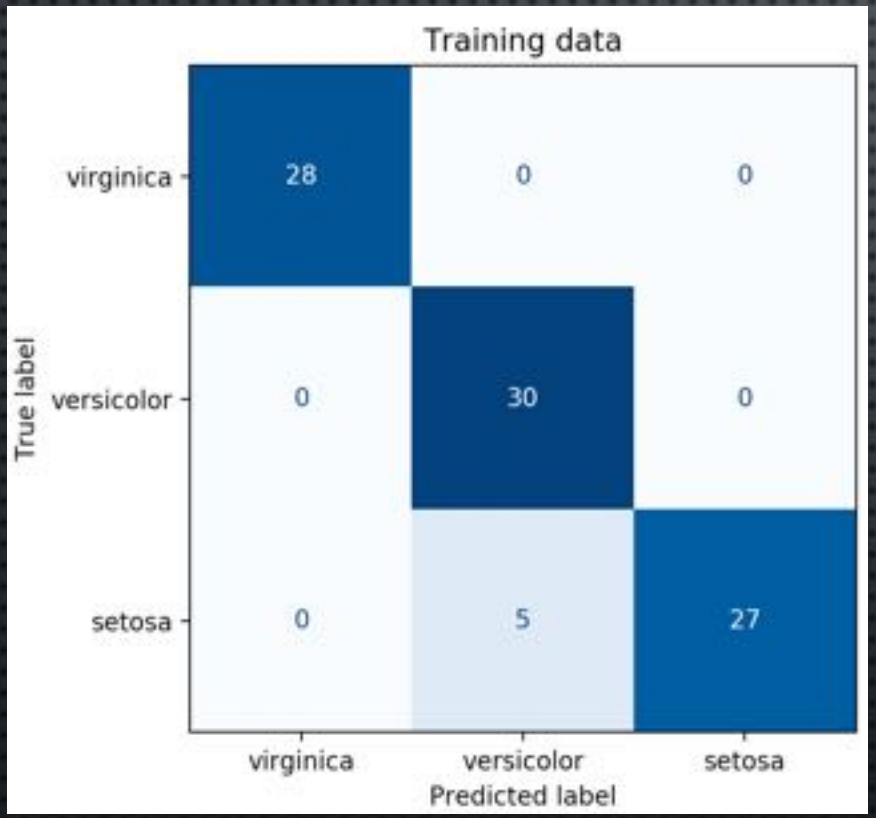
- Rescale data
  - Standardisation
  - Normalisation: [0, 1]
- Taking care of missing data
- Dimensionality reduction

K-nearest neighbours: Iris data

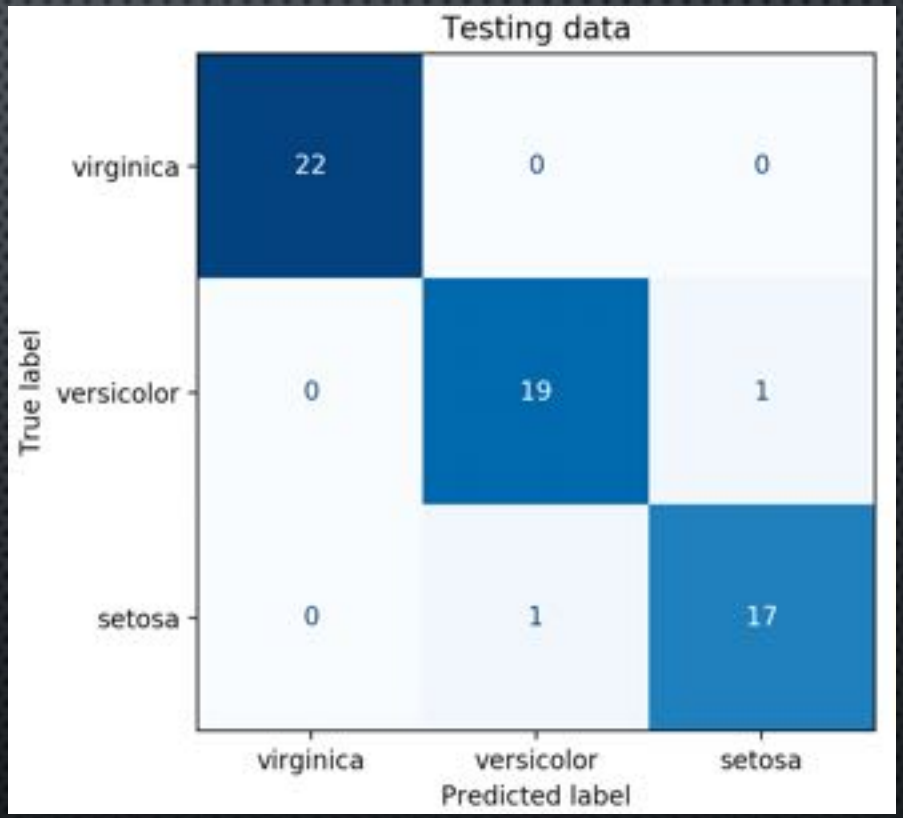
- No. of samples = 150
- Classes = {"*setosa*", "*versicolor*", "*virginica*"}
- Training : testing = 0.6 : 0.4
- K = 5
- Weights = ”uniform”
  - Optional: “distance”
- Metric = “Minkowski”
  - p=2, i.e. Euclidean distance

K-nearest neighbours: Iris data

Error rate: 5.6%

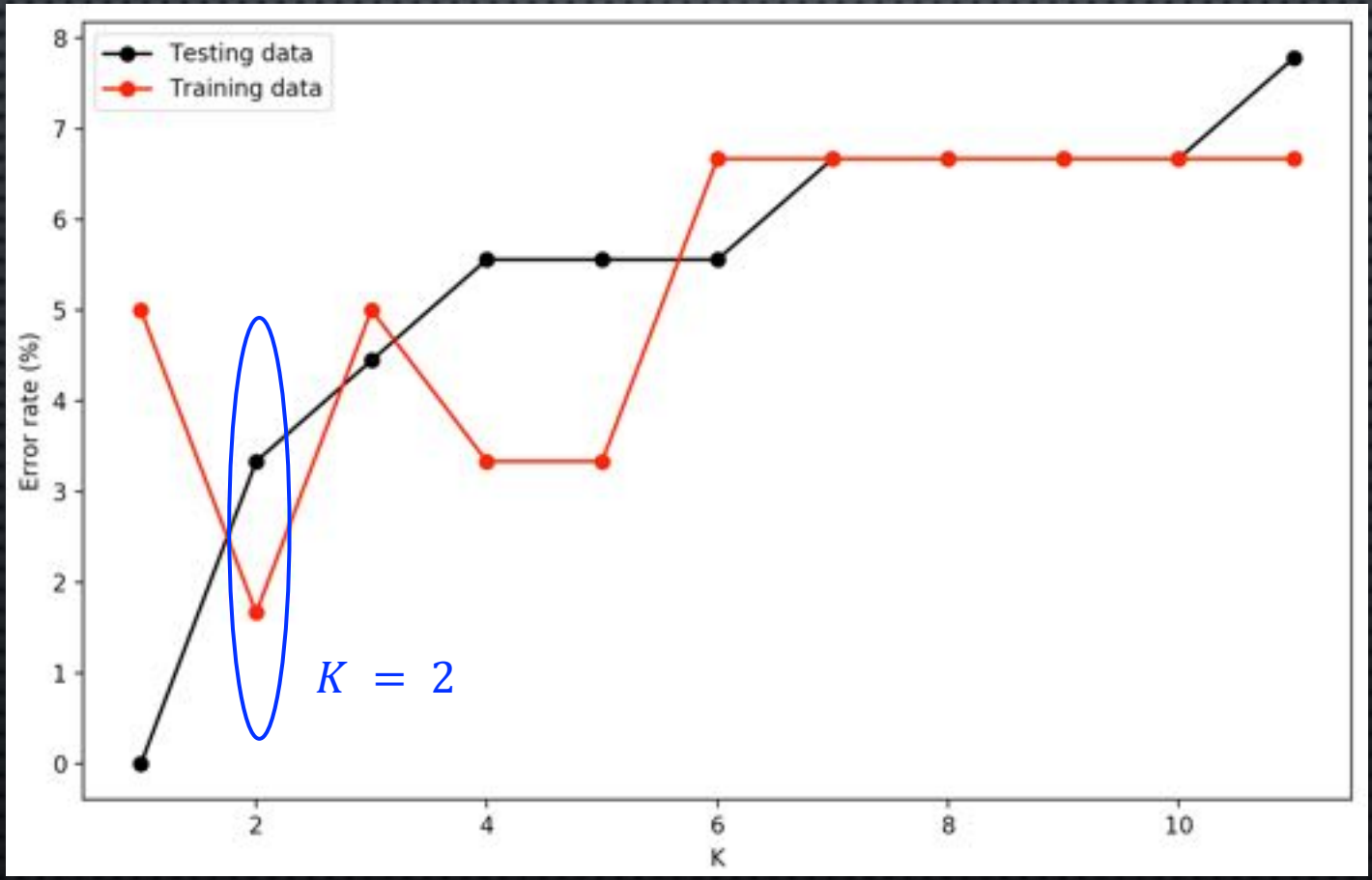


Error rate: 3.3%



K-nearest neighbours: Iris data

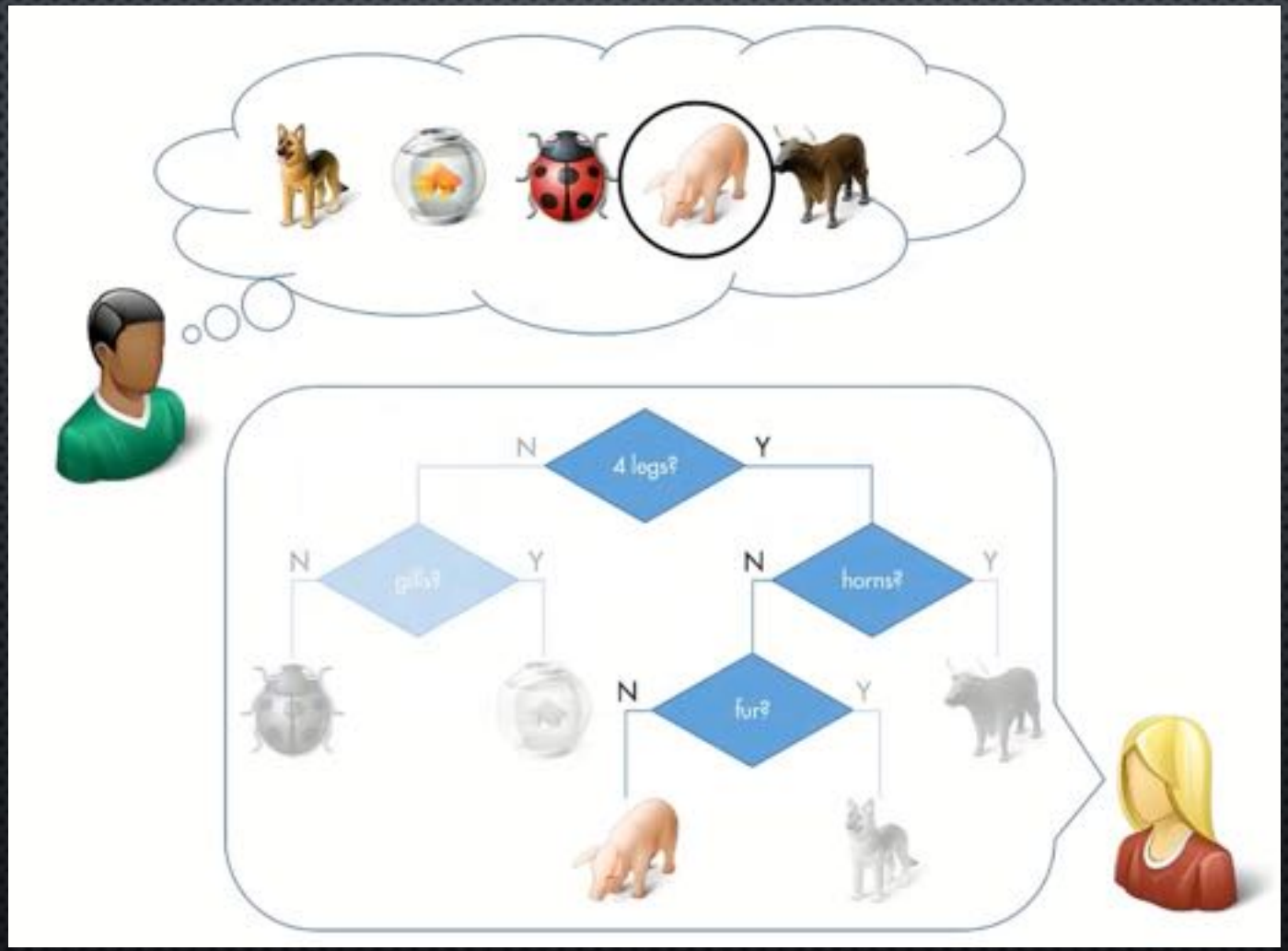
Error rate as a function of $K \in [2, 12]$



K-nearest neighbours : limitation

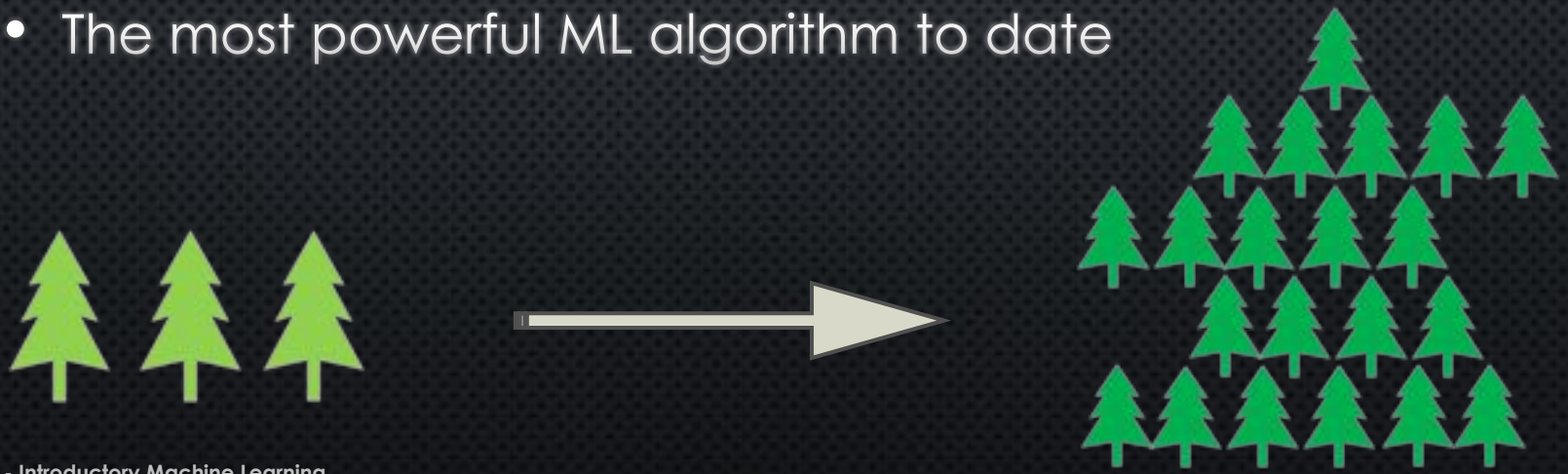
- Works well with small number of variables/features
- Performance deteriorates with increase of variables
  - Why? All is about use of "distance" or "similarity" between observations
  - "Curse of dimensionality"
- Require features to be all numeric or categorial

Decision trees (TDs)

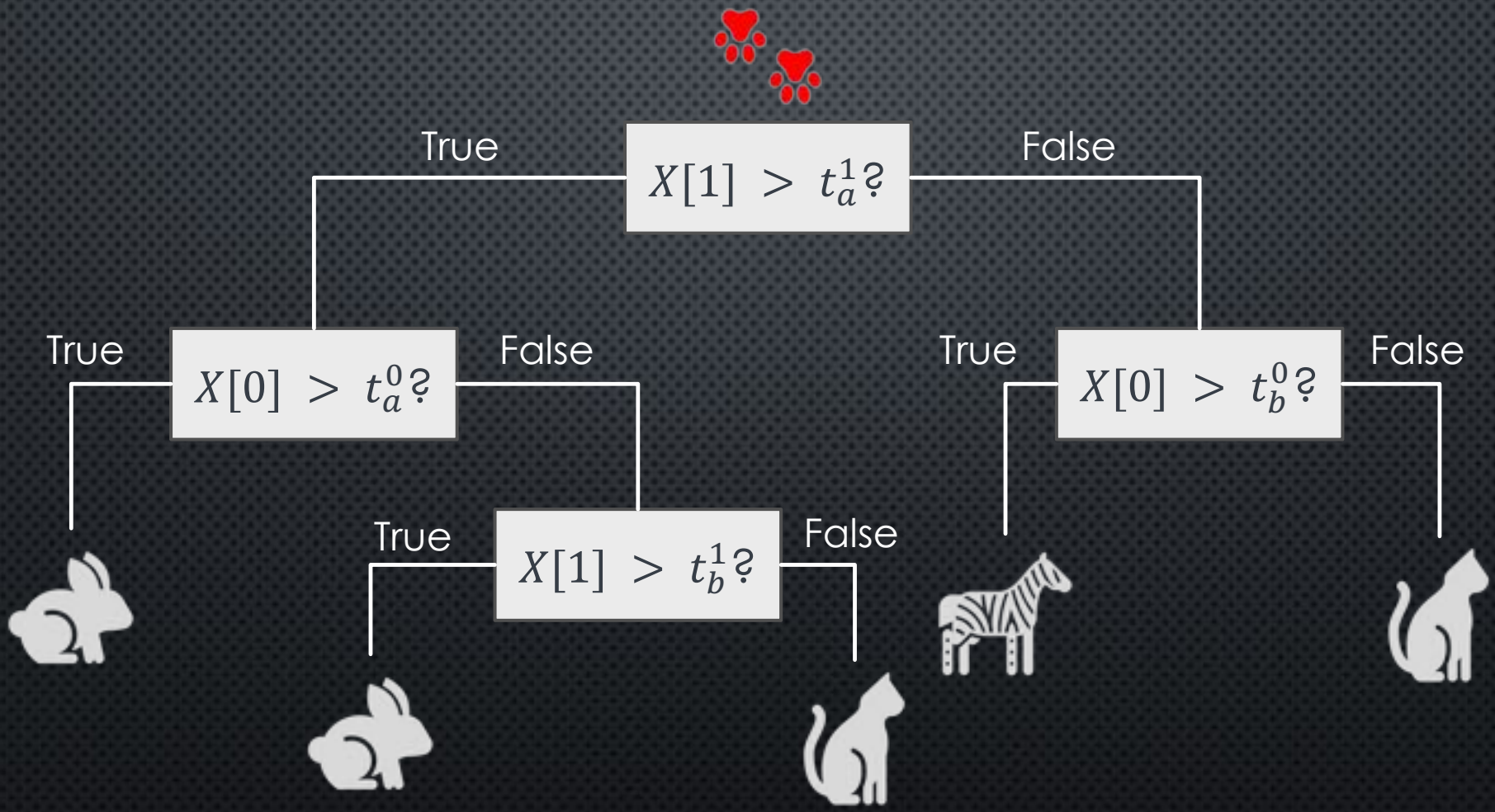


Decision trees

- Builds a hierarchy by "asking" "true or false" questions ⇨ decisions!
- Also nonparametric model
- Widely used for classification and regression
- Elementary units of random forests
  - The most powerful ML algorithm to date



Decision trees



Decision trees: principle

Building the tree, starting from the root node (i.e. the entire data set)

- Determine the most informative test for a target variable
  - i.e. decide on variable $X[i]$ and criterion t_x^i
- Split into two branches (for binaural trees)
- Test the quality of splitting for each branch using Gini impurity, G
  - If $G = 0$, no heterogenous element
  - Otherwise, repeat the entire process until $G = 0$ or meeting other criteria

Decision trees: CART algorithm

Gini impurity: the impurity of each so-determined class G_j at node j:

$$G_j = 1 - \sum_{n=1}^{N} p_{j,n}^2$$

$$p_{j,n} = \frac{m_{j,n}}{M_j} \qquad \sum_{n=1}^{N} p_{j,n} = 1$$

$p_{j,n}$: Ratio of instances in class n among the total observations in j

N: number of classes

$m_{j,n}$: number of observations in class n at j

M_j : total number of observations at j

Decision trees: CART algorithm

Determination of variable $X[i]$ and criterion t_x^i

CART cost function to minimise:

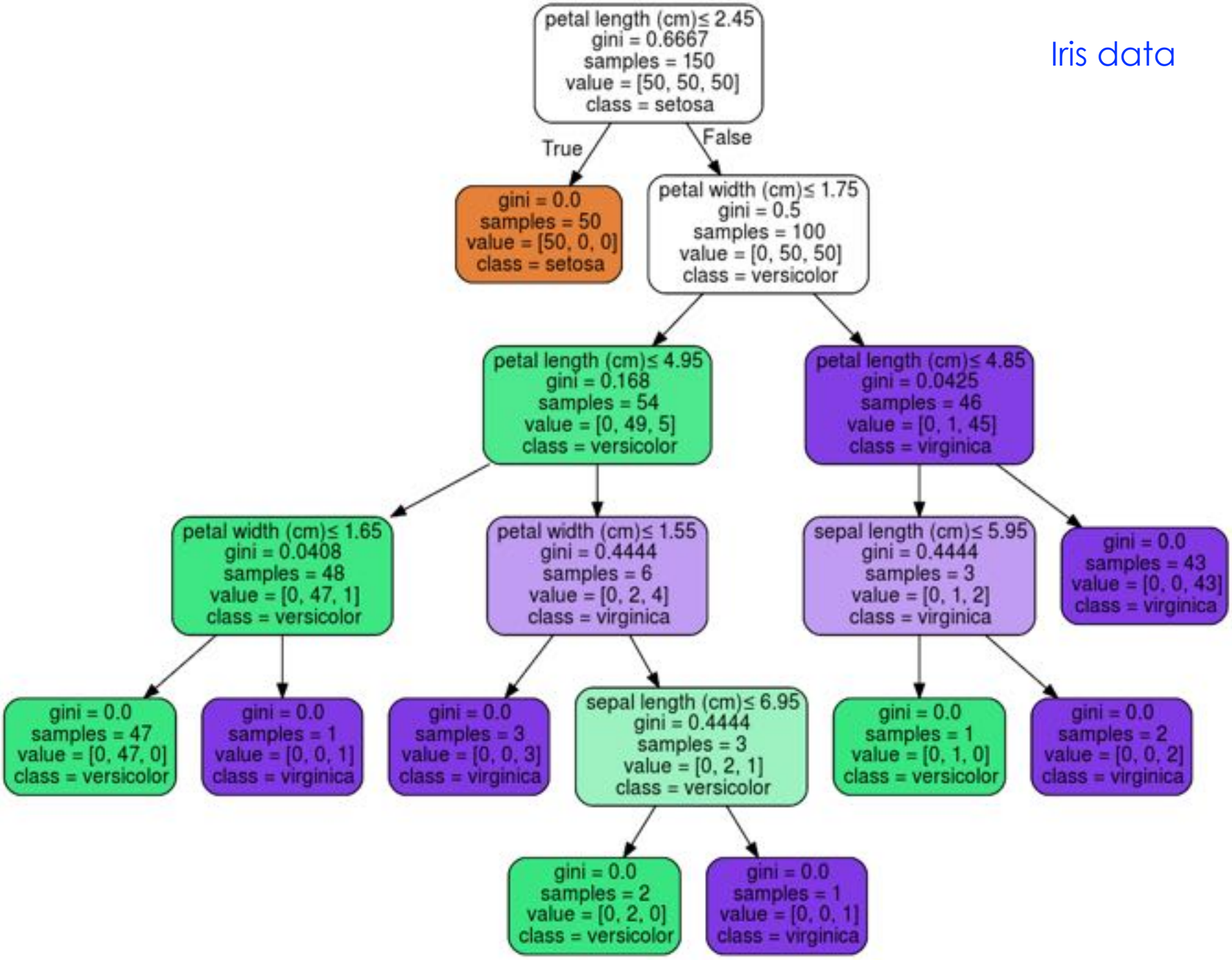
$$C\left(X[i], t_x^i\right) = \frac{m_{left}}{M} G_{left} + \frac{m_{right}}{M} G_{right}$$

m_{left}, m_{right}: number of observations in the left/right branch

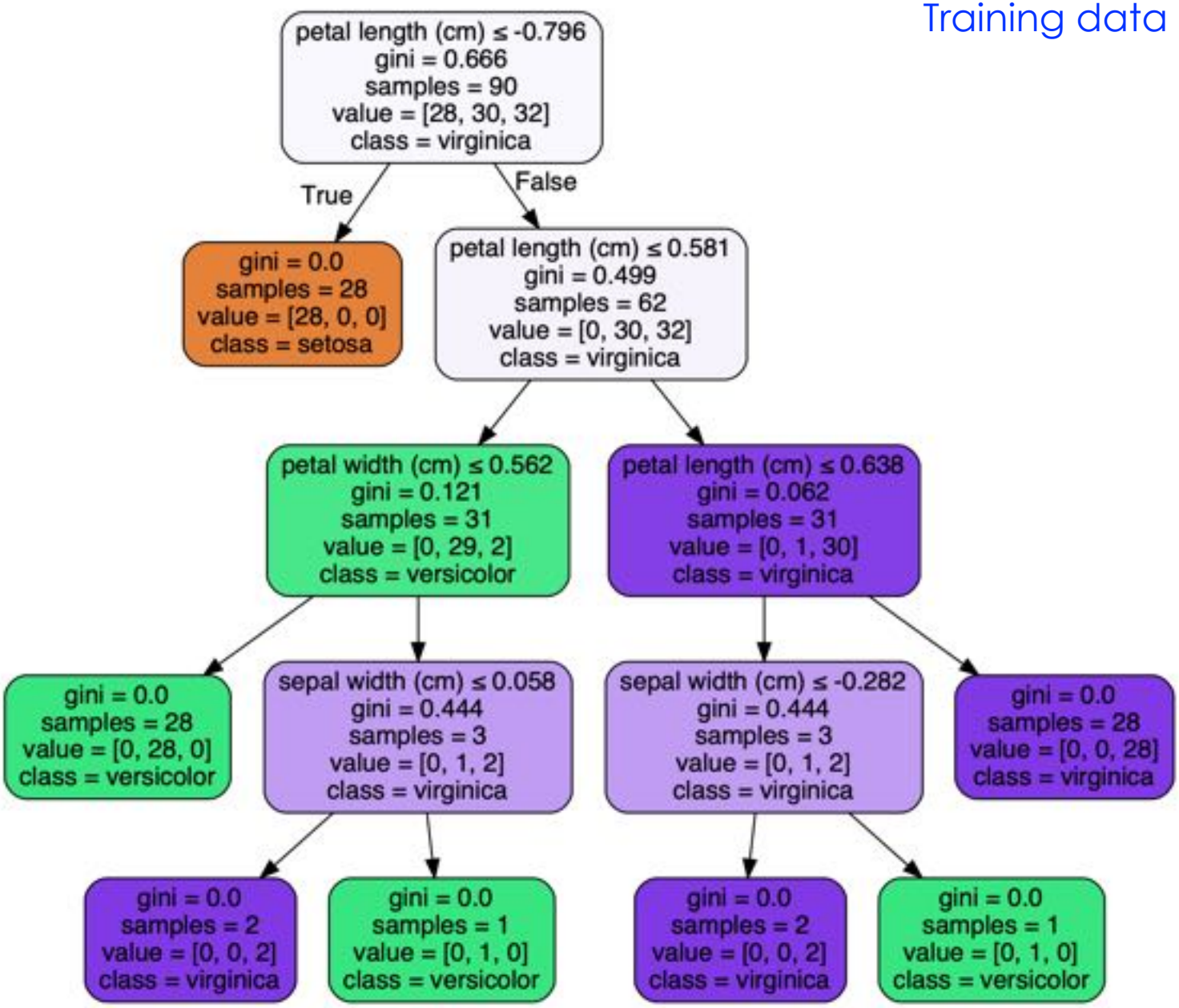
G_{left}, G_{right}: impurity of the left/right branch

M: total number of observations in that node

Iris data

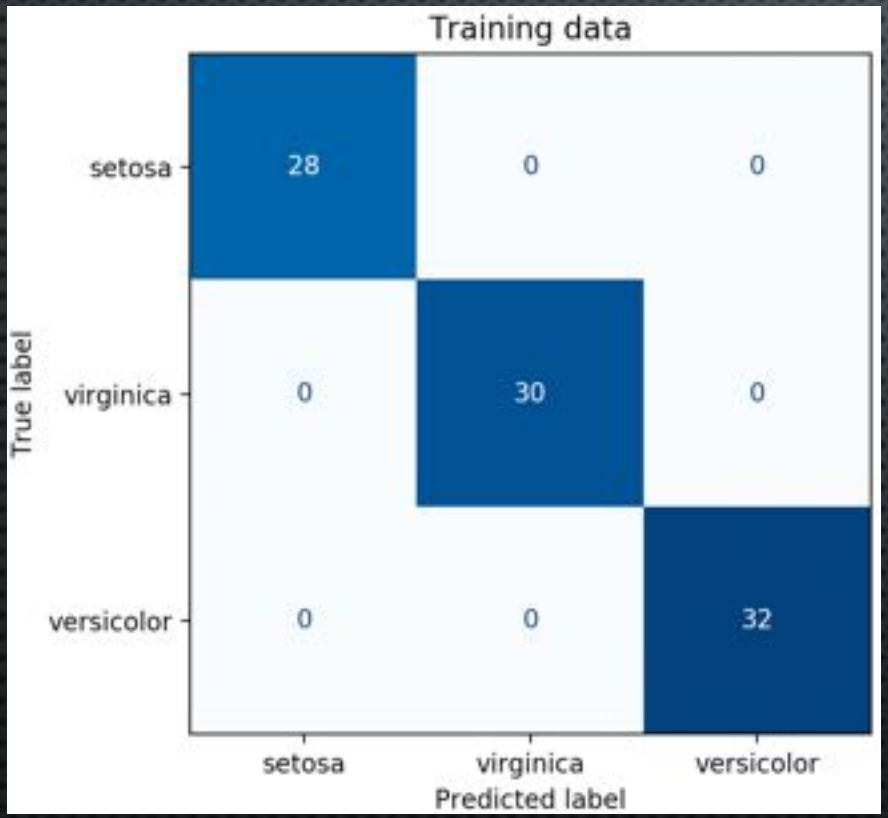

petal length (cm)≤ 2.45
gini = 0.6667
samples = 150
value = [50, 50, 50]
class = setosa
True
False
gini = 0.0
samples = 50
value = [50, 0, 0]
class = setosa
petal width (cm)≤ 1.75
gini = 0.5
samples = 100
value = [0, 50, 50]
class = versicolor
petal length (cm)≤ 4.95
gini = 0.168
samples = 54
value = [0, 49, 5]
class = versicolor
petal length (cm)≤ 4.85
gini = 0.0425
samples = 46
value = [0, 1, 45]
class = virginica
petal width (cm)≤ 1.65
gini = 0.0408
samples = 48
value = [0, 47, 1]
class = versicolor
petal width (cm)≤ 1.55
gini = 0.4444
samples = 6
value = [0, 2, 4]
class = virginica
sepal length (cm)≤ 5.95
gini = 0.4444
samples = 3
value = [0, 1, 2]
class = virginica
gini = 0.0
samples = 43
value = [0, 0, 43]
class = virginica
gini = 0.0
samples = 47
value = [0, 47, 0]
class = versicolor
gini = 0.0
samples = 1
value = [0, 0, 1]
class = virginica
gini = 0.0
samples = 3
value = [0, 0, 3]
class = virginica
sepal length (cm)≤ 6.95
gini = 0.4444
samples = 3
value = [0, 2, 1]
class = versicolor
gini = 0.0
samples = 1
value = [0, 1, 0]
class = versicolor
gini = 0.0
samples = 2
value = [0, 0, 2]
class = virginica
gini = 0.0
samples = 2
value = [0, 2, 0]
class = versicolor
gini = 0.0
samples = 1
value = [0, 0, 1]
class = virginica

Training data

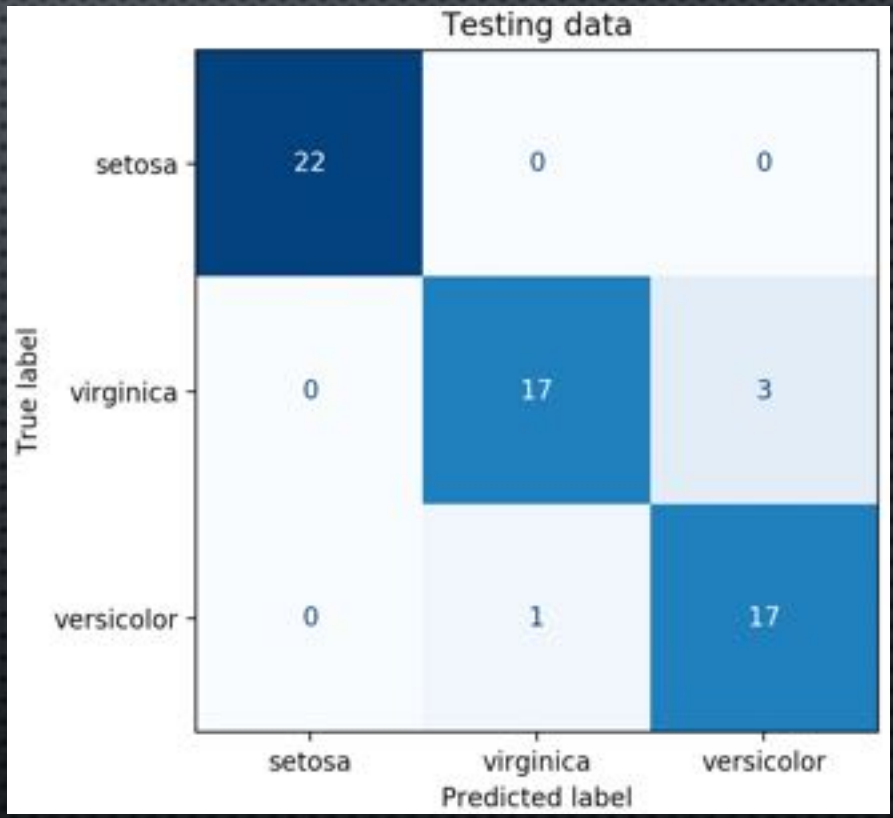


Decision trees: Iris data

Error rate: 0.0%



Error rate: 6.7%



Decision trees: impurity or entropy

Entropy E_j at node j:

$$E_j = -\sum_{n=1}^{N} p_{j,n} \cdot \log_2 (p_{j,n})$$

$p_{j,n}$: Ratio of instances in class n among the total observations in j

N: number of classes

- Measures the average information content of a message
- When 0, all the observations are homogeneous
- Impurity tends to isolate most frequent class in its own branch; entropy may produce somewhat more balance trees

Decision trees: summary

- Advantages:
  - Do not require data scaling; binary decisions regarding to a specific variable. No impact to other variables
  - Allows a mixture of numeric and categorical predictors
  - White box model; easy to interpret
- Disadvantages:
  - Prone to overfitting : tree pruning
  - Can be unstable: use it within an ensemble
- Note that scikit-learn implementation does not support categorical variables for now

https://scikit-learn.org/stable/modules/tree.html#tree