Machine Learning

Rudolf Mayer

(mayer@ifs.tuwien.ac.at)

October 30th, 2019

Outline

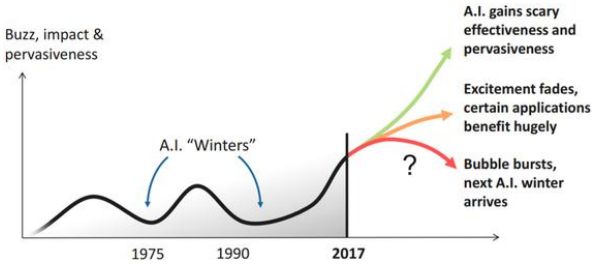
- Short recap
- K-nn (continued)
- Data preparation
- Evaluation (intro)
- Decision Trees
- Evaluation (continued)

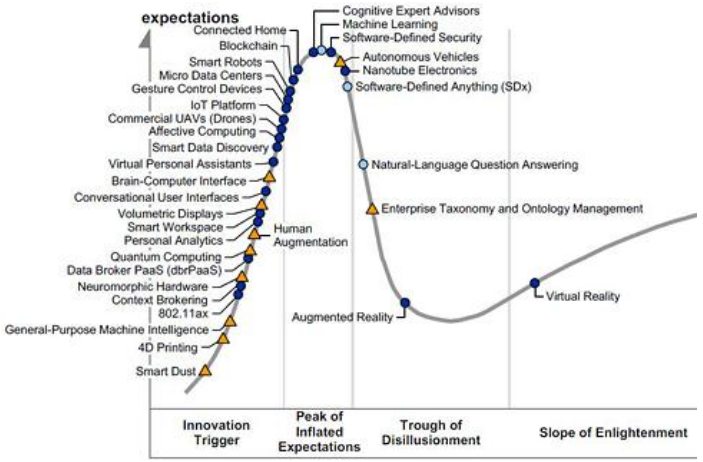
Outline

- Short recap
- K-nn (continued)
- Data preparation
- Evaluation (intro)
- Decision Trees
- Evaluation (continued)

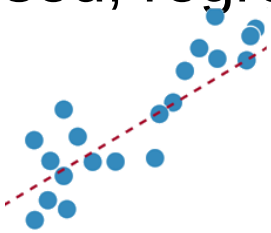
Recap – Lecture October 9th

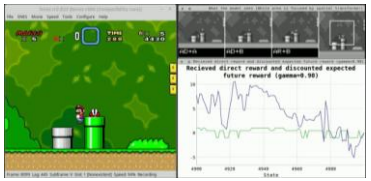
- ML Definitions & setting





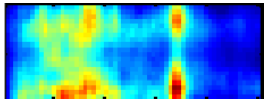
  - Supervised, unsupervised, regression, classification, reinforcement, ..

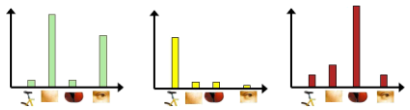




  - Data Science/Mining process

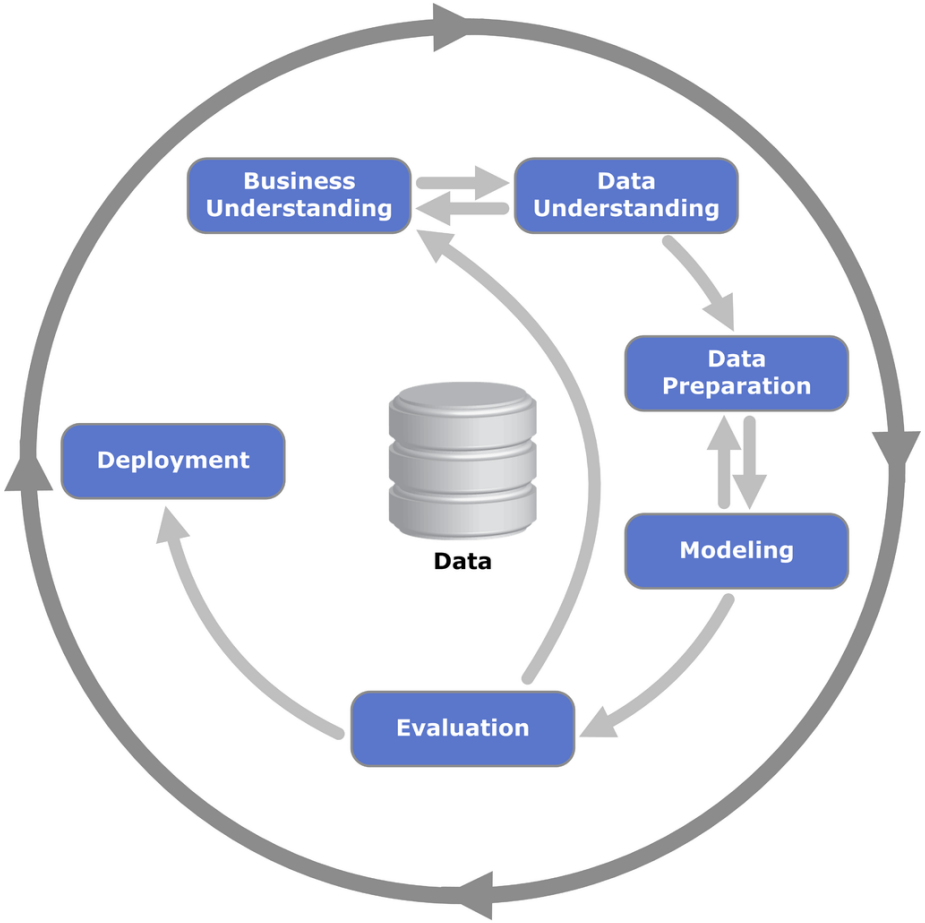
  - Feature extraction



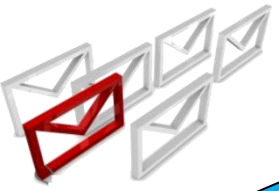


Recap – Lecture October 9th

- Data Mining Process



Recap – Lecture October 9th



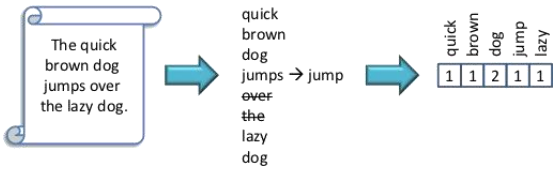
label

Ground Truth

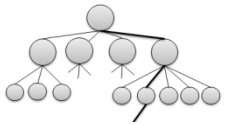
Data

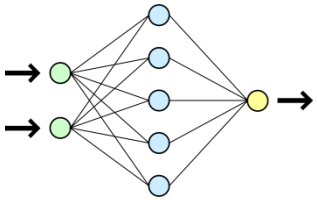
extract/ transform

Features



train

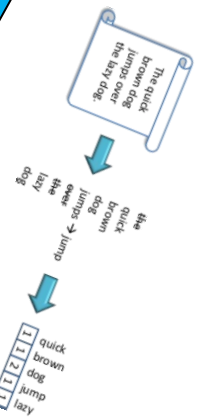




ML Model

Features

extract/ transform



apply

Unseen Data

Prediction

Continuous (regression)

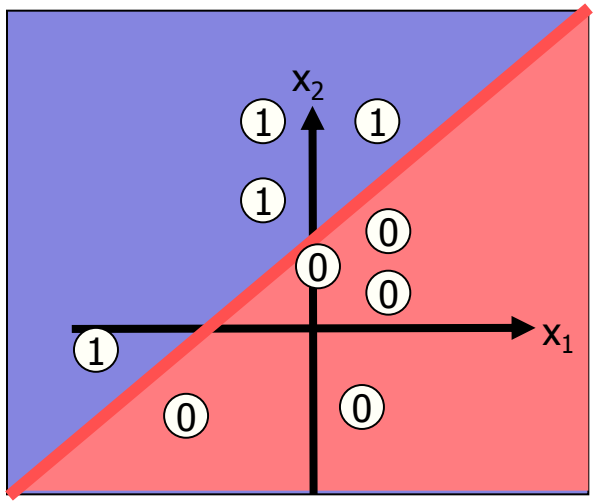
Binary (one out of two classes)

Multi-class (one out of n classes)

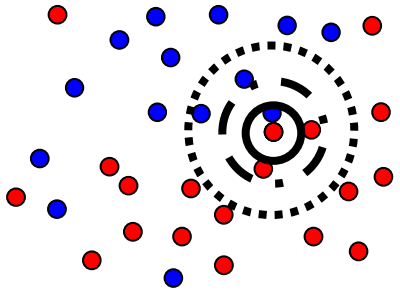
Multi-label (one or more out of n classes)

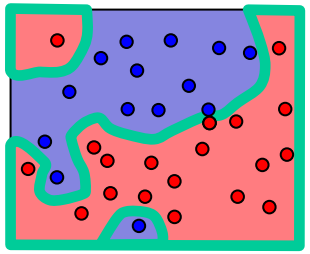
Recap

- Perceptron (1950s)
  - ***Linear seperation***
    - Linear combination of inputs
    - Activation function
  - Learning weights & bias
    - Stopping criterion



- k-NN Classification
  - Searching for k-closest neighbours
  - Classification follows majority
  - ***Lazy learner***
  - Weighted majority voting
  - Optimisations for finding neighbours





Recap

- Classifiers discussed so far based on different principles:
  - Linear vs. more complex decision boundaries
  - Online / iterative learning vs. analytic solution
  - Probabilities, distances (instance based), decision rules, hyperplanes, ...
  - Lazy learning vs. model building

Recap

- Intro to evaluation
  - Accuracy, precision, recall
  - Type I / Type II errors
  - Training/test set, ...

	true	false
true	True positive (TP)	False positive (FP, Type I error)
false	False negative (FN, Type II error)	True negative (TN)

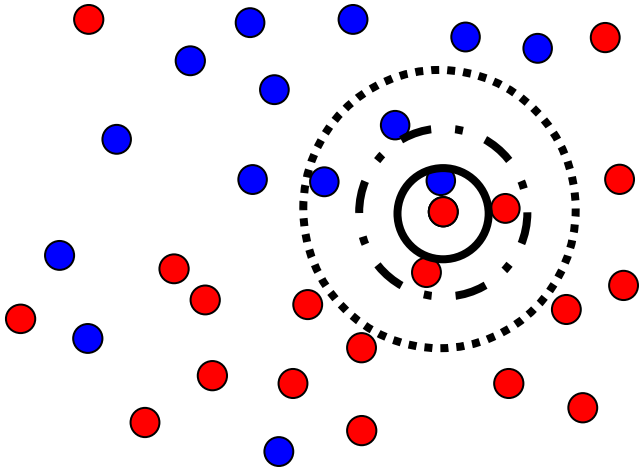
- Data types & preparation
  - Categorical (e.g. size: tiny/small/large) vs. numerical (size in cm)
  - Scaling/Normalisation, 1-n coding, ...,
  - missing values/imputation, ..

Outline

- Short recap
- K-nn (continued)
- Data preparation
- Evaluation (intro)
- Decision Trees
- Evaluation (continued)

k-NN: majority voting

- Simple case: all k neighbours are equally important
  - Majority decides on the class
  - *Is that an issue?*



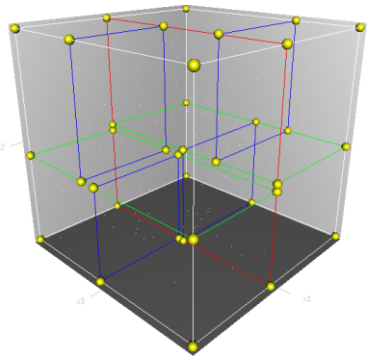
k-NN: majority voting

- Weighted approaches: more influence to closer neighbours
  - Rank weighted: weight determined by rank
    - E.g. each neighbour has a vote 1/r, where r is the rank (1, 2, .., k)
  - Distance weighted: weight determined by distance
    - E.g. each neighbour has a vote 1/d, where d is the (Euclidean) distance from the sample to the neighbour
  - *Differences in these two approaches?*

- *What happens if you chose k=n (number of elements?)*

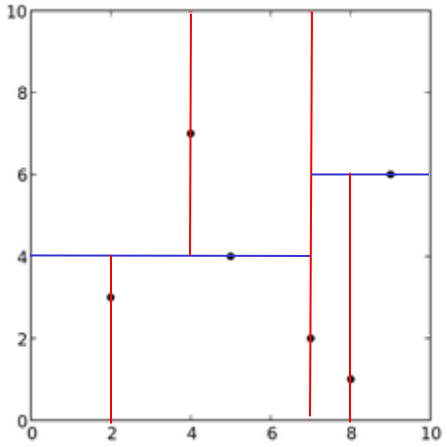
k-NN: other search strategies

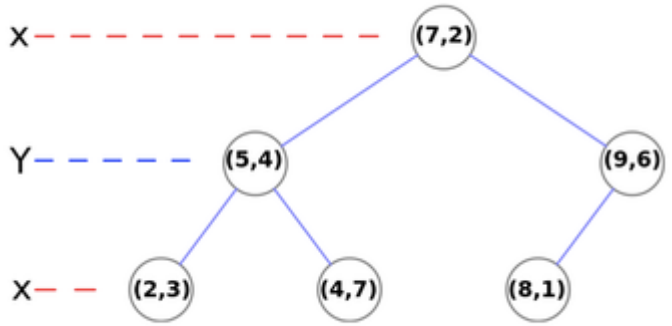
- Becomes computationally expensive with many items to classify
  - Linear (brute-force) search: O (Nd) *N = # samples, d = dimension*
- Search space-partitioning
  - E.g. K-d-Tree
    - Proposed 1975 by Jon Louis Bentley
    - Binary Search Tree (BST)
    - Every node is k-dimensional point
    - Non-leaf node: splitting hyperplane
      - Divides space into two parts



k-NN Search: K-d-Tree

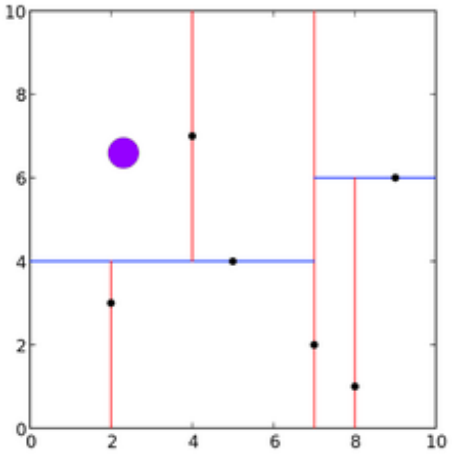
- Recursively split space along points
  - Alternating on the dimensions (axis)

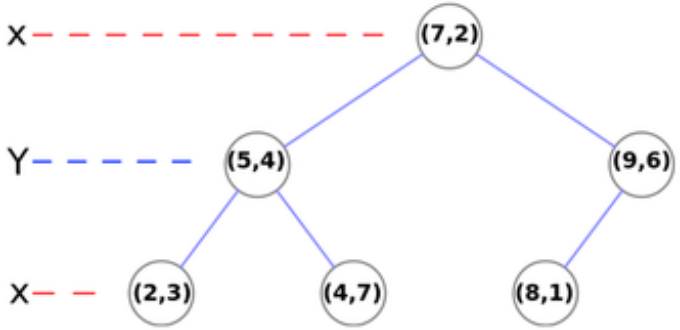




k-NN Search: K-d-Tree

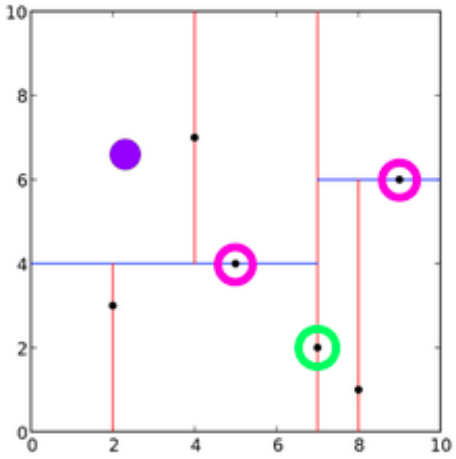
- How to find the neighbours of  ?

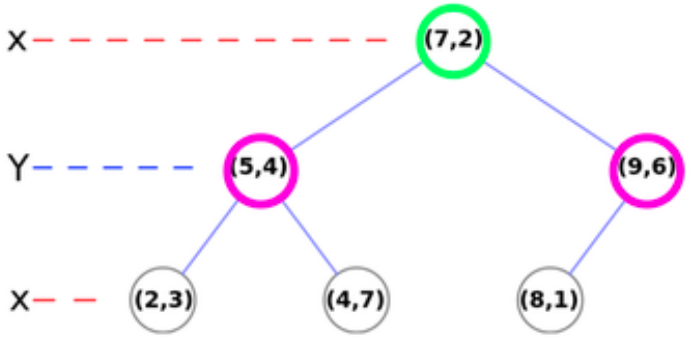




k-NN Search: K-d-Tree

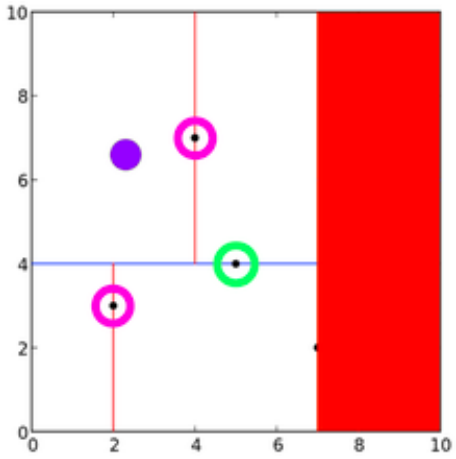
- How to find the neighbours of ● ?
  - Find closest point in first level: (5,4) or (9,6)

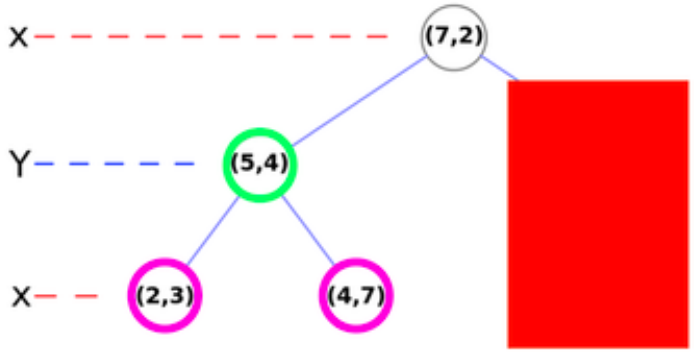




k-NN Search: K-d-Tree

- How to find the neighbours of ● ?
  - Find closest point in first level: (5,4) or (9,6)
    - (5,4) is closer – eliminate other branch
  - Find closest point in second level: (2,3) or (4,7)

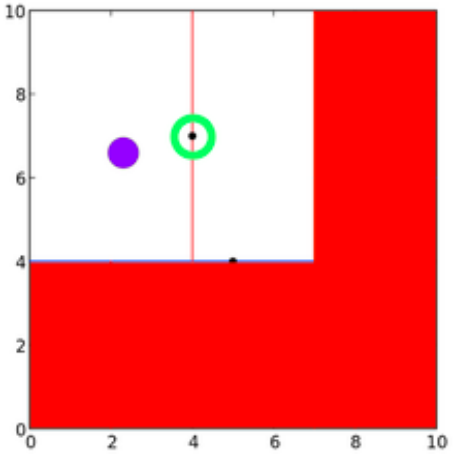


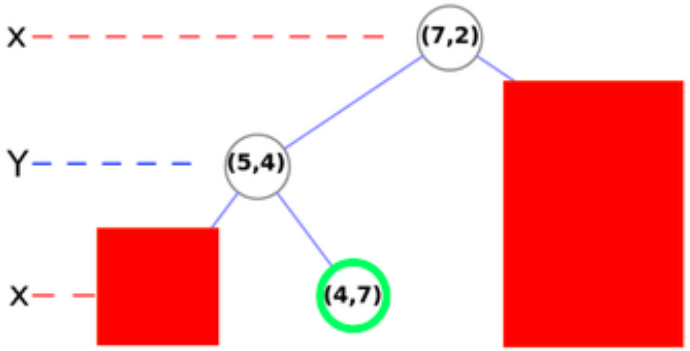


k-NN Search: K-d-Tree

- How to find the neighbours of ● ?
  - Find closest point in first level: (5,4) or (9,6)
    - (5,4) is closer – eliminate other branch
  - Find closest point in second level: (2,3) or (4,7)
    - (4,7) is closer – eliminate other branch

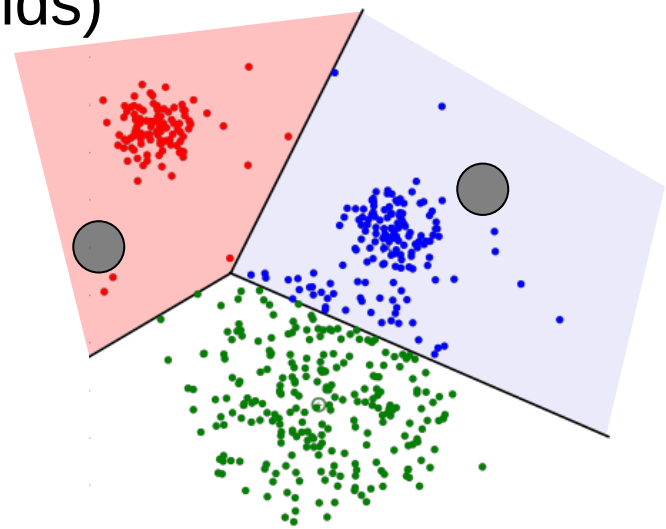
➔ Search in remaining region





k-NN Search: Clustering

- Using Vector Quantisation / Clustering
  - Obtain *prototypes* (cluster centroids)
    - e.g. using k-Means algorithm



  - For a new sample
    1. Find closest prototype (containing cluster)
    2. Then search for neighbours in the same cluster

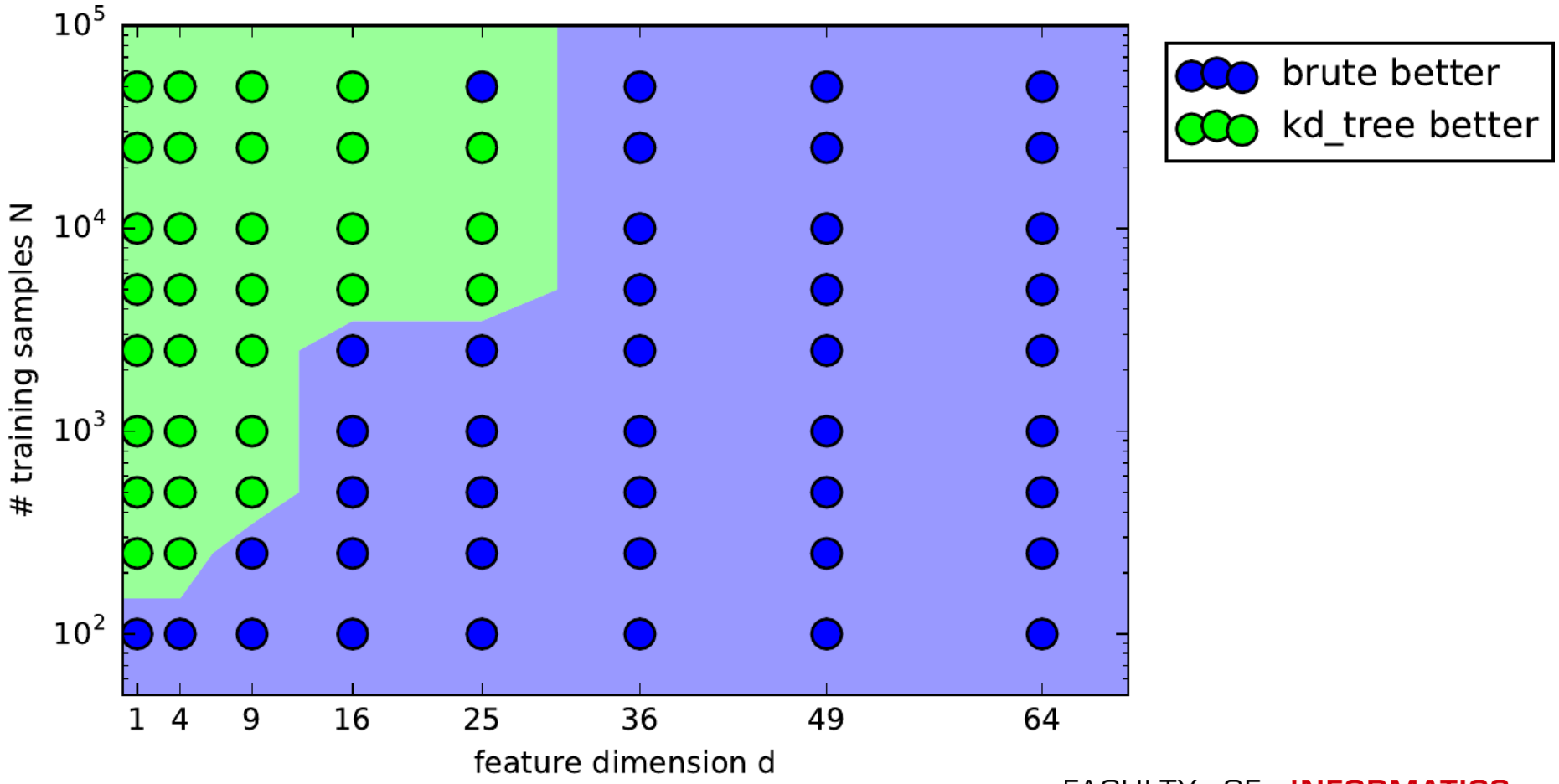
k-NN: other search strategies

- Advantage: can significantly reduce time
  - Practical experiments: up to 10 times
  - Important: need to consider both the time for the search (classification, **AND** for creating the tree/vector quantisation (this becomes your training time/model)

- *Disadvantages ?*
  - Does ***not*** pay off for small datasets
    - kd-Tree shown to perform worse with high d
  - May yield different predictions *(why?)*

k-NN: other search strategies

- Kd-Tree evaluation (**combined** training & classification)

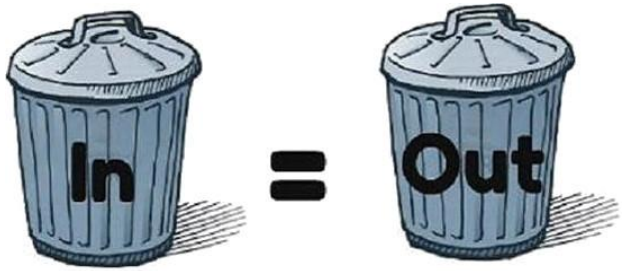


Outline

- Short recap
- K-nn (continued)
- Data preparation
- Evaluation (intro)
- Decision Trees
- Evaluation (continued)

Data Preparation

- Vital step for machine learning (supervised and unsupervised)
- ML algorithm will ***always*** give you a model
  - Quality of that model depends highly on the quality of the input data
- "Garbage in" ➔ "Garbage out"



- One major goal of data preparation:
  - Eliminate "wrong influence" of variables

Data Preparation

- Example data set from earlier (lung cancer)

gender	age	height	smoker	eye colour
male	19	170	yes	green
female	44	162	yes	grey
male	49	185	yes	blue
male	12	178	no	brown
female	37	165	no	brown
female	60	157	no	brown
male	44	190	no	blue
female	27	178	yes	brown
female	51	162	yes	green
female	81	168	yes	grey
male	22	184	yes	brown
male	29	176	no	blue

- Potential issues ?
  - Missing values
  - Quantitative (continuous) data with different scales
  - Data type not fitting (categorical vs numerical) data

Scaling / Normalisation / Standardisation

- Different variables may exhibit significantly different value ranges
  - E.g. a length variable measured in cm, inch, or meters
  - Different types of measurements: length, speed, temperature, …
  - Different types of measuring devices capturing different value ranges
  - …
  - *Why is this a (potential) problem?*

Scaling / Normalisation / Standardisation

- Some ML algorithms rely on *measuring the (numeric) distance* between samples

- ➔ There should be no impact by the value range

  – higher values in one attribute / variable would have unproportional effect on measure distance
  ➔ would dominate distance metric
  ➔ might thus dominate learning

Scaling / Normalisation / Standardisation

- Remove effects of different value ranges in each attribute/variable

- Common method in statistics and machine learning
  - With many different notations
  - Scaling, Normalisation, Standardisation, ...
  - Often used interchangely, different for each field ...

Scaling / Normalisation / Standardisation

- Z-score standardisation / normalization

- Each variable *x* is converted to have a mean of 0 and a standard deviation of 1
  - subtracting the mean
  - dividing by standard deviation

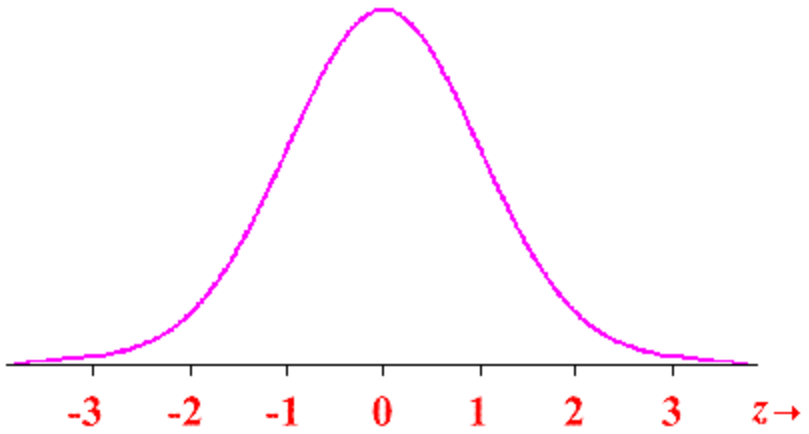
$$z_i = \frac{x_i - \mu}{\sigma}$$

Scaling / Normalisation / Standardisation

- Z-score standardisation / normalization

$$z_i = \frac{x_i - \mu}{\sigma}$$

- *What's the new value range after z-score?*



Scaling / Normalisation / Standardisation

- Min-Max scaling
  - Scale all variables to the same (fixed) range
  - Often between 0 and 1
  - Subtract minimum value for each variable
  - Divide by value range of each variable

$$z_i = \frac{x_i - \min(X)}{\max(X) - \min(X)}$$

  - *Multiply by new range (if different than 0..1)*

Scaling

- *Is scaling an issue for*
  - *k-nn?*
  - *Perceptron?*

Scaling / Normalisation / Standardisation

- Especially algorithms relying on distances
  - k-Nearest Neighbours
  - …
- Scaling not needed for algorithms that don't use distances, e.g.
  - Naïve Bayes
  - Decision trees
  - …

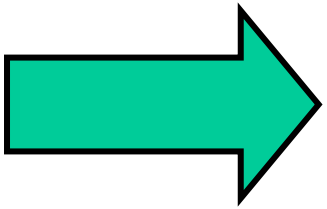
Scaling / Normalisation / Standardisation

- Especially algorithms relying on distances
  - k-Nearest Neighbours
  - ...

- *Caveat*
  - Many implementations already do this pre-processing implicitly
  - Check default settings carefully

One hot encoding (1-to-N Coding)

colour
brown
blue
green
grey
brown
green
blue



green	*blue*	*brown*	*grey*
0	0	1	0
0	1	0	0
1	0	0	0
0	0	0	1
0	0	1	0
1	0	0	0
0	1	0	0

Categorical data: animals data set

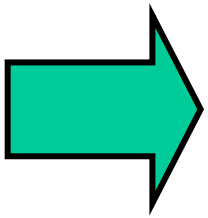
- Animal data set
  - Describes animal by some characteristics
  - Instances: cow, duck, bee, …
- Variables
  - Size (tiny, small, medium, big)
  - Number of legs (2, 4, 6, 8)
  - Feathers (yes/no)
  - Eggs (yes/no)

	size	*legs*	*feathers*	*eggs*
duck	small	2	yes	yes
dog	medium	4	no	no
spider	tiny	8	no	yes
ladybird	tiny	6	no	yes
cow	large	4	no	no
bee	tiny	6	no	no
sparrow	small	2	yes	yes

One hot encoding animal data set

- Replace categorical attribute “size” with four binary attributes

	size	*legs*
duck	small	2
dog	medium	4
spider	tiny	8
ladybird	tiny	6
cow	large	4
bee	tiny	6
sparrow	small	2

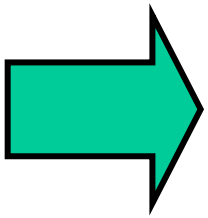


tiny	*small*	*med*	*large*	*legs*
0	1	0	0	2
0	0	1	0	0
1	0	0	0	8
1	0	0	0	6
0	0	0	1	4
1	0	0	0	6
0	1	0	0	2

Label encoding animal data set

- *N.b.: could also encode to numerical value: label encoding*
  - *E.g.: tiny=1, small=2, med=3, large=4* *(if sizes are equally distant, ratio scale!)*

	size	***legs***
duck	small	2
dog	medium	4
spider	tiny	8
ladybird	tiny	6
cow	large	4
bee	tiny	6
sparrow	small	2



size	***legs***
2	2
3	4
1	8
1	6
4	4
1	6
2	2

Categorical data: another example

- *Any other pre-processing needed?*

- Variable “legs”
  - If considered categorical: defined order ➔ ordinal data
  - Can compute similarity: 2 closer to 4 than to 6

	size	*legs*
duck	small	2
dog	medium	4
spider	tiny	8
ladybird	tiny	6
cow	large	4
bee	tiny	6
sparrow	small	2

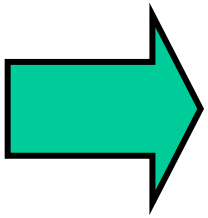
  - Numerical value, can compute distance directly
  - *Does the number of legs denote similarity?*
    - *Is this a ratio-scale variable?*

One hot encoding animal data set

- *Does number of legs denote similarity?*
  - i.e., is an animal with 2 legs more similar to one with 4, or with 6?
  - And, is one with 4 equally similar to the one with 2 and 6?
    - Dog to monkey vs. dog to spider
  - One with 6 equally similar to one with 4 and 8?
    - Bee to spider vs. bee to cow

One hot encoding animal data set

	size	*legs*
duck	small	2
dog	medium	4
spider	tiny	8
ladybird	tiny	6
cow	large	4
bee	tiny	6
sparrow	small	2



2 legs	*4 legs*	*6 legs*	*8 legs*
1	0	0	0
0	1	0	0
0	0	0	1
0	0	1	0
0	1	0	0
0	0	1	0
1	0	0	0

k-nn: distance function

- *What to do with categorical data?*

a) 1-n coding – then apply any distance function mentioned earlier

b) Definition of custom distance functions that applies to categorical data

k-nn: distance function

- Definition of custom distance functions
  - E.g. adapt hamming distance

    Hamming distance between two strings of equal length is the number of positions at which the corresponding symbols are different

    ➔ count number of different nominal values

	size	*legs*	*feathers*	*eggs*	*distance*
horse	large	4	no	no	
duck	**small**	**2**	**yes**	**yes**	4
dog	**medium**	4	no	no	1
spider	**tiny**	**8**	no	**yes**	3
ladybird	**tiny**	**6**	no	**yes**	3
cow	large	4	no	no	
bee	**tiny**	**6**	no	no	2
sparrow	**small**	**2**	**yes**	**yes**	4

k-nn: distance function

- Definition of custom distance functions
  - E.g. adapt hamming distance

    Hamming distance between two strings of equal length is the number of positions at which the corresponding symbols are different

    ➔ count number of different nominal values

  - Define custom distance for **each attribute**, and then aggregate e.g. via sum

Other data preparation aspects

- Missing values
  - For some samples, not all attribute values are known
    - Some ML algorithms can handle missing values
    - For others, we need a special treatment: data imputation

➔ *More in this in upcoming lectures*

Outline

- Short recap
- K-nn (continued)
- Data preparation
- Evaluation (intro)
- Decision Trees
- Evaluation (continued)

Evaluation

- When we use a machine learning model, we want to know how good it is (effectiveness)
  - To know how confident we can be in the predictions
  - To know which algorithm to use
  - ....
- ➔ *Model validation*

- Need to measure performance of an algorithm
  - Test on (labelled) data
  - Several different measures

- Orthogonal topic: efficiency, i.e. required runtime
  - *More on that later*

Evaluation (effectiveness)

- Binary classification (classes true/false)
- Table of confusion *(contingency table)*

		Actual value		
		true	false	
Test outcome	true	True positive (TP)	False positive (FP, Type I error)	***Precision*** $\frac{TP}{TP + FP}$
	false	False negative (FN, Type II error)	True negative (TN)	
		Recall Sensitivity $\frac{TP}{TP + FN}$		***Accur-acy*** $\frac{TP + TN}{\#\ samples}$

- *Examples of Type I & II errors?*
  - *Which is worse?*

Evaluation measures

	true	false
true	True positive (TP)	False positive (FP, Type I error)
false	False negative (FN, Type II error)	True negative (TN)

- Accuracy: # correctly predicted samples

$$\frac{TP + TN}{TP + FP + TN + FN} = \frac{TP + TN}{\#\ samples}$$

- Precision $\frac{TP}{TP + FP}$ ➔ *How to optimise?*
- Recall $\frac{TP}{TP + FN}$ ➔ *How to optimise?*
- *More measures in later lectures ...*

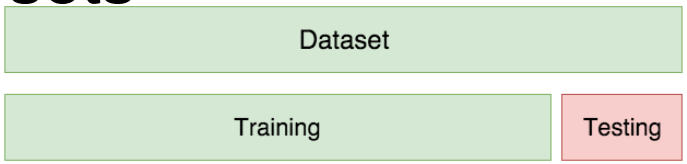
Evaluation: data

- Which data to do evaluation on?
- *The samples used for training?*
  - *Why not?*
  - Would not tell us how good the model works for unknown data
    - Which is however why we train a model in first place ...
  - *If we test on training data – we are biased*
    - Fully grown decision trees – will be 100% on training data
    - Perceptron on linear separable data: training data will be 100% correctly “predicted”
    - K-NN, Naïve Bayes: not necessarily 100% correct on training data. Still biased!
    - Similar for other algorithms, e.g. SVMs, ...
  - *Want to actually find out:* how will model perform on **unseen** data!

Training & Test set

- Really unseen data doesn't have labels
  - "Simulate" "unseen" data
  - ➔ Holdout method: keep part of your labelled data for testing

- Split data into training and test sets
  - E.g. ~80% training, 20% test
    - 66% - 33%
  - Sorted (linear), random, …

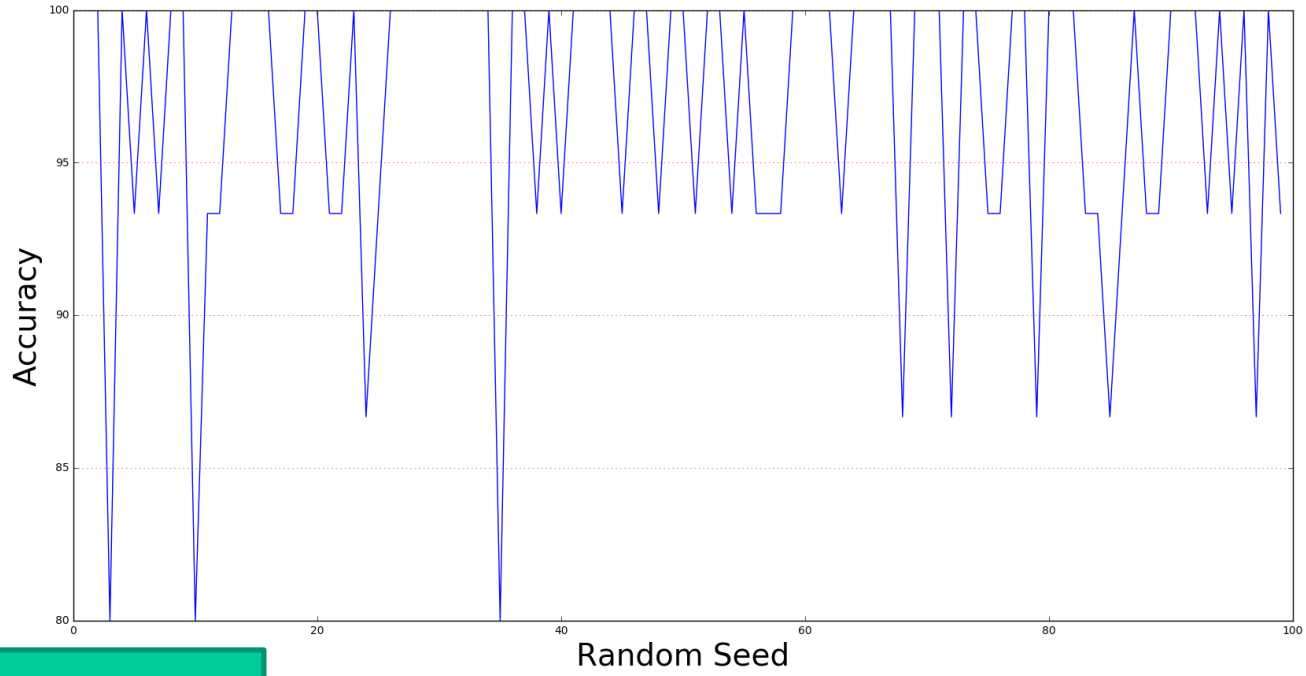


- Performance on test set is an estimate for generalisation ability of our model

- Results can vary a lot according to how the split is done

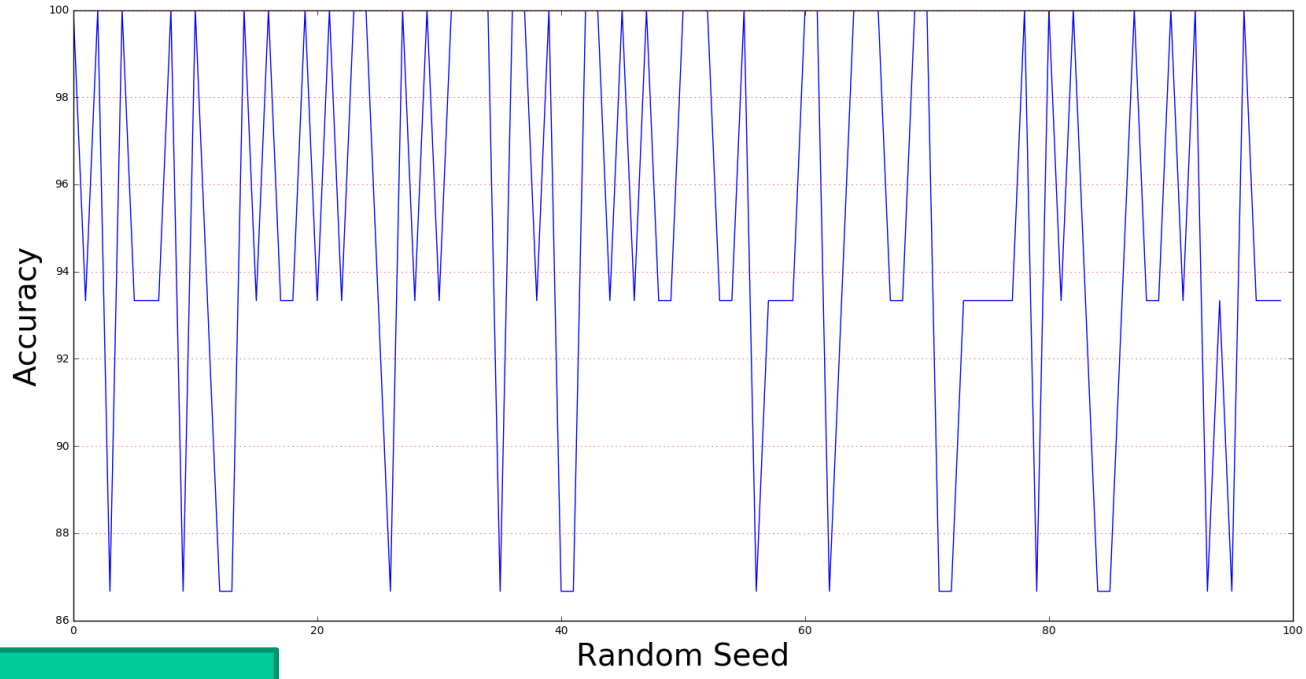
Holdout method

- Random influence in training/test split
  - Example: **Iris Data, *k-nn (k=15)***, random seed 1..100
    - Average Accuracy: 96.67 (+/- 9.8883) [range: 80-100]



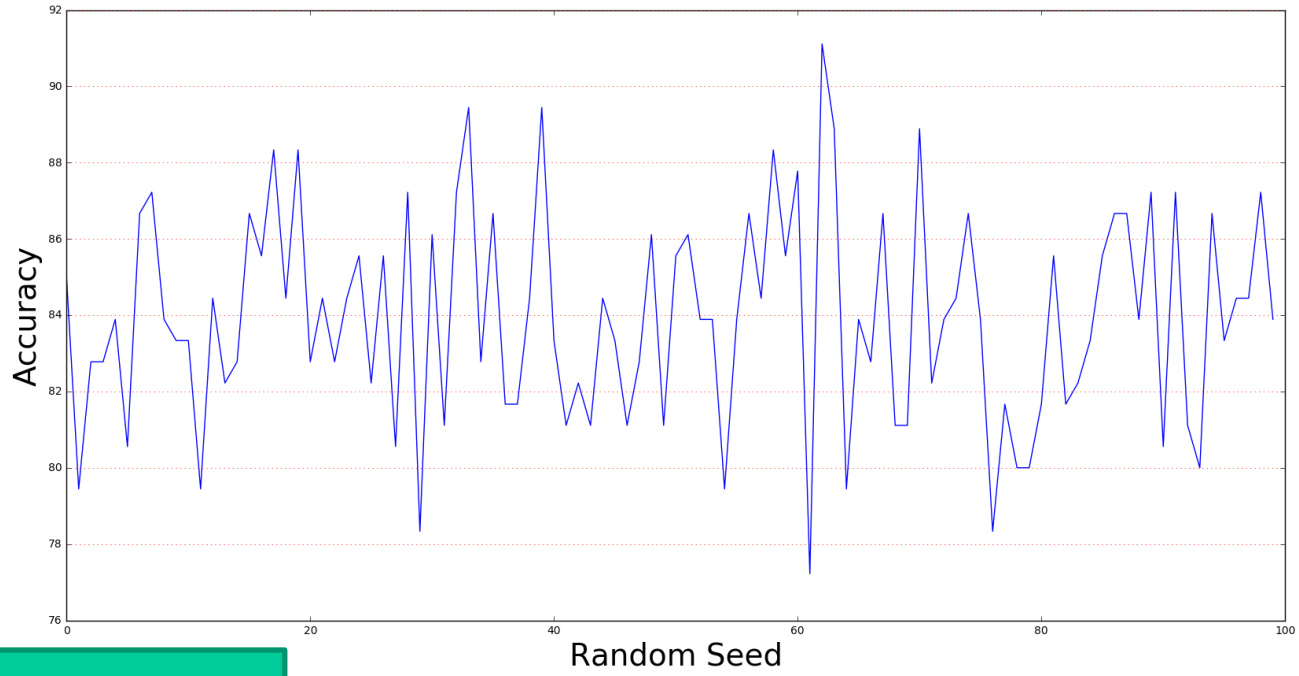
Holdout method

- Random influence in training/test split
  - Example: **Iris Data, *Naive Bayes***, random seed 1..100
    - Average Accuracy: 95.00 (+/- 9.6839) [range: 86.67-100]



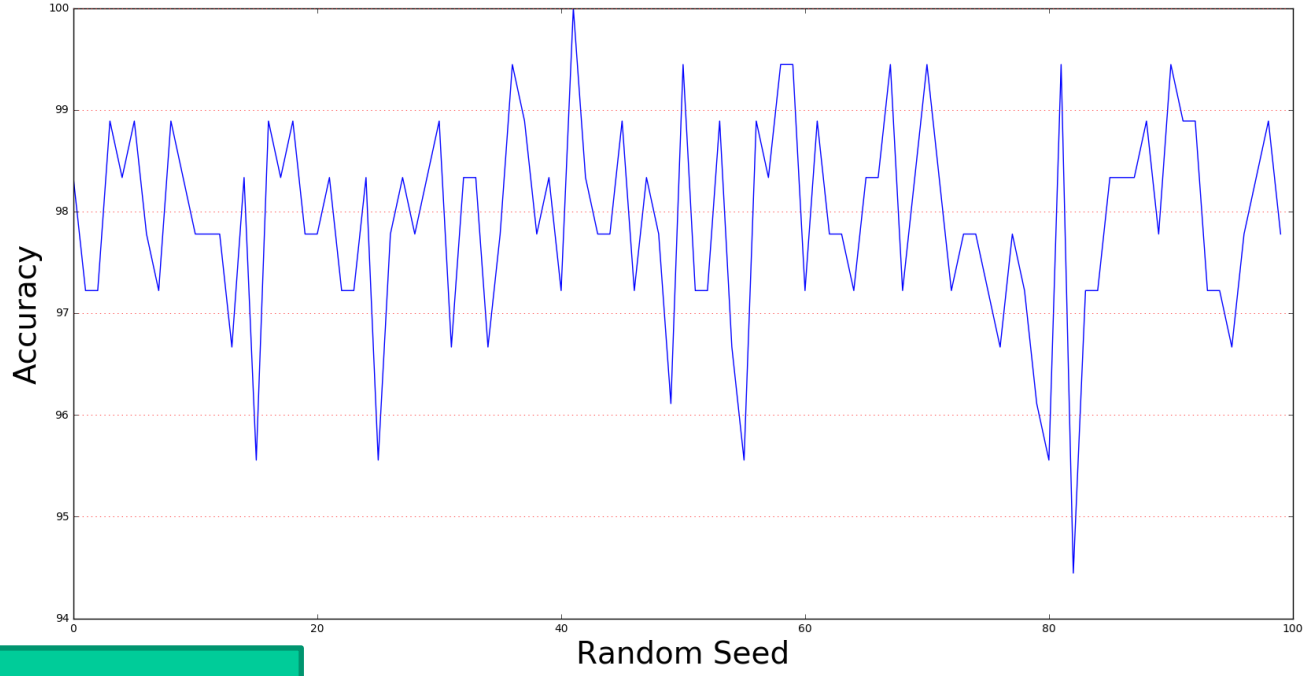
Holdout method

- Random influence in training/test split
  - Example: **Digit Data, *k-nn (k=15)***, random seed 1..100
    - Average Accuracy: 97.90 (+/- 2.0097) [range: 94.44-100]



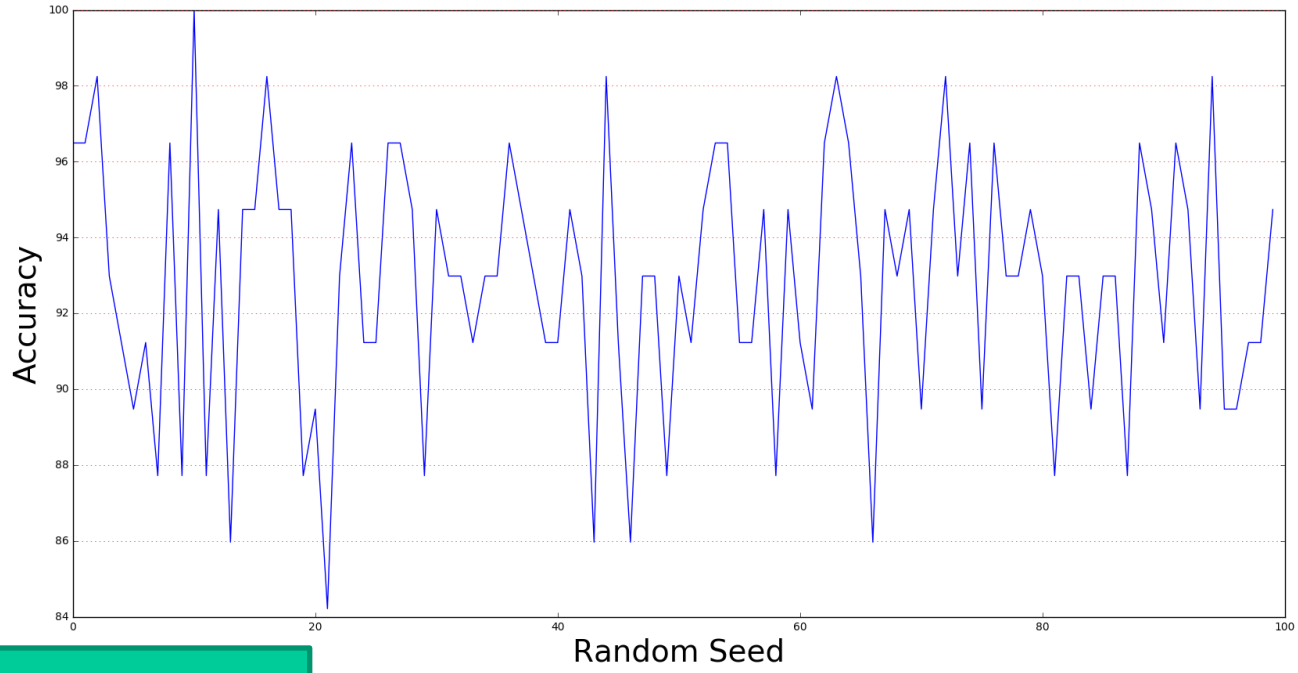
Holdout method

- Random influence in training/test split
  - Example: **Digit Data, *Naive Bayes***, random seed 1..100
    - Average Accuracy: 83.93 (+/- 5.6167) [range: 77.22-91.11]



Holdout method

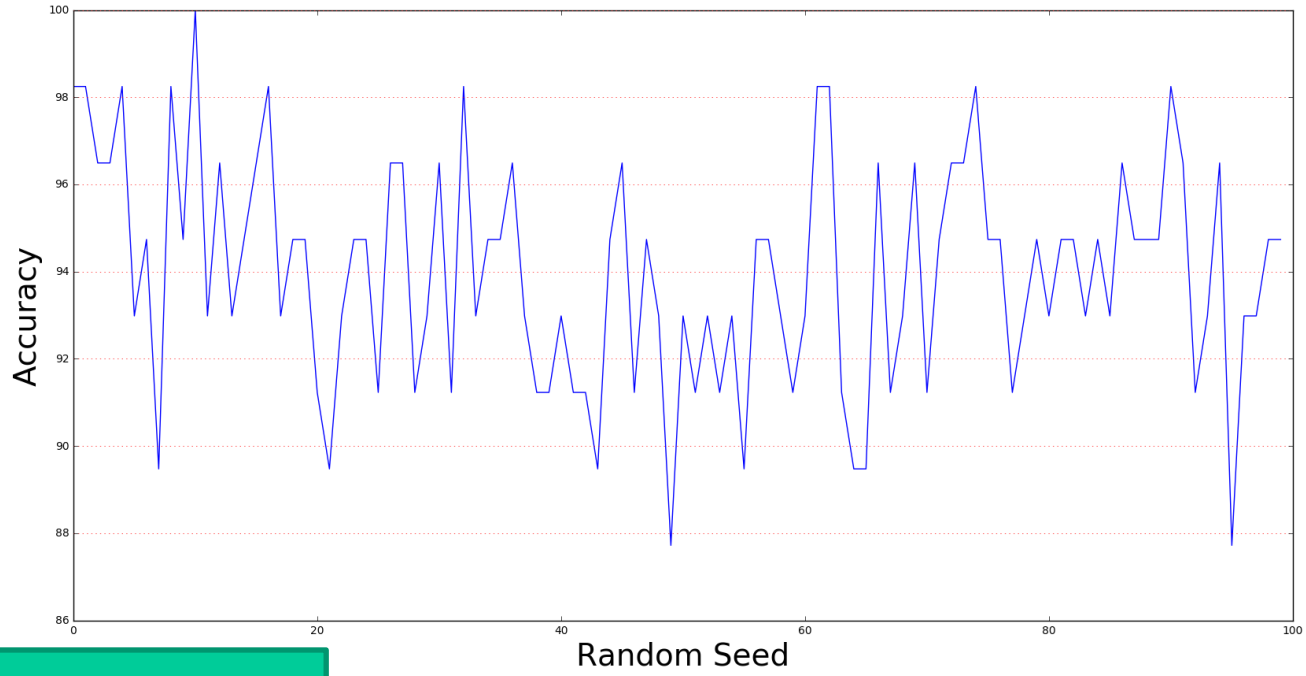
- Random influence in training/test split
  - Example: **Cancer Data, *k-nn (k=15)***, random seed 1..100
    - Average Accuracy: 92.81 (+/- 6.7401) [range: 84.21-100]



Holdout method

- Random influence in training/test split
  - Example: **Cancer Data, *Naive Bayes***, random seed 1..100
    - Average Accuracy: 93.96 (+/- 5.2355) [range: 87.719-100]

- Solution?



k-fold Cross validation

- Split data into e.g. 10 parts of equal sizes
  - This is called 10-fold cross validation
- Repeat 10 times:
  - use 9 parts for training (training set)
  - calculate performance on remaining part (test set)

- Estimate of performance is average (mean) of the validation set performances

k-fold Cross validation

- Estimate of performance is average (mean)
- In addition to mean, compute standard deviation
  - Indication on how stable the results are in the folds

    ➔ lower standard deviation is better ...

  - Standard deviation to be considered when comparing cross-validation performances from different classifiers

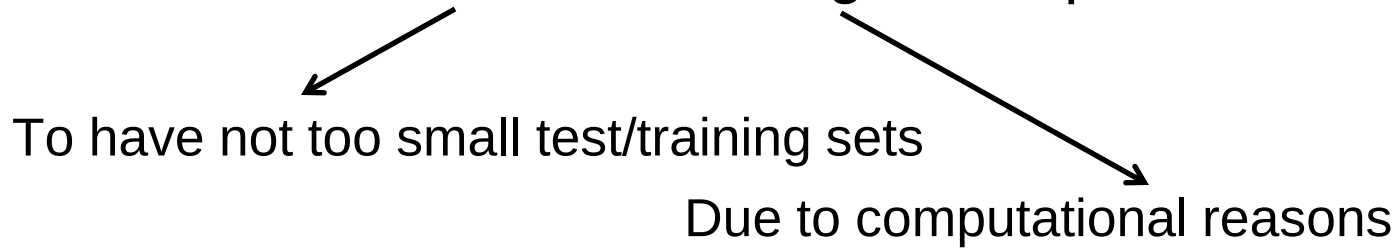
Classifier / Fold	1	2
1	91,80	86,7
2	82,30	87
3	84,40	87,1
4	93,00	85,7
5	81,60	86,8
6	87,40	86,4
7	82,40	87,2
8	92,10	86,5
9	91,90	86,5
10	87,40	86,5
Mean	87,4	86,6
Stdv	4,6	0,4

k-fold Cross validation

- Which classifier is better:
  - Average 87,4%, standard deviation 4,8%
    - *(87,4% 4,8)*
  - Average 86,6%, standard deviation 0,4%
    - *(86,6% 0,4)*
    - ➔ *More on that later: significance testing*

k-fold Cross validation

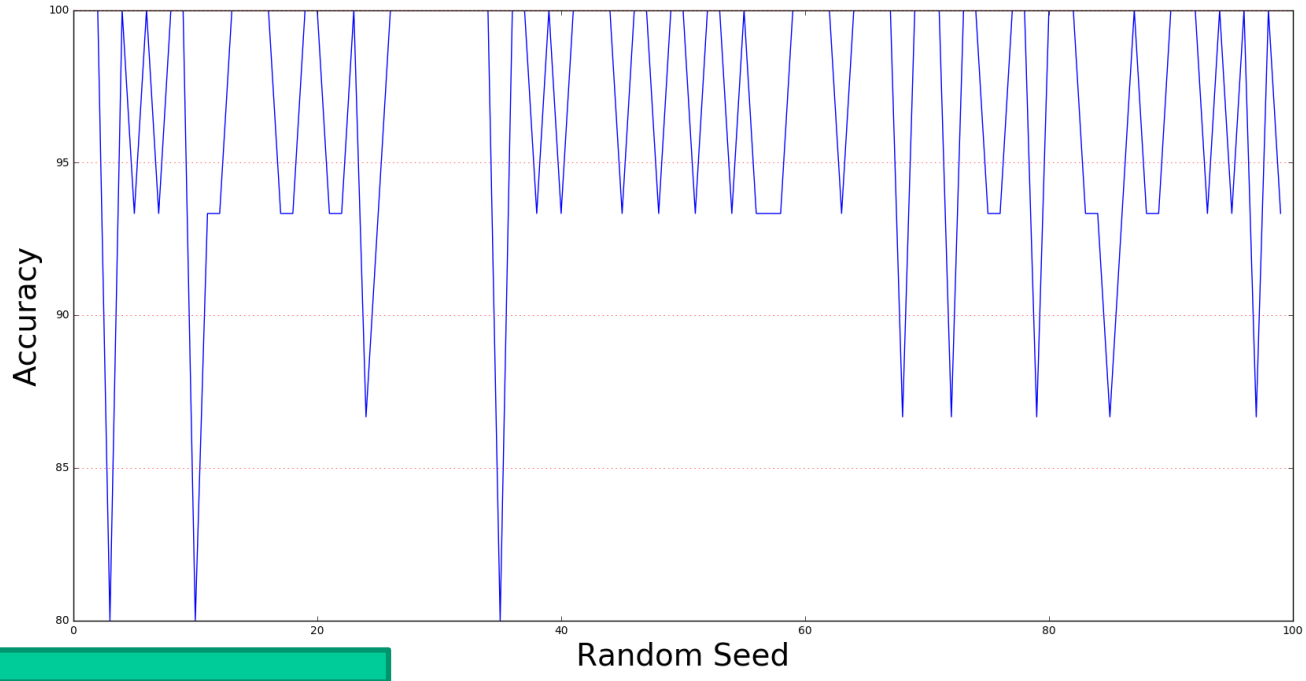
- Results obtained via cross-validation are generally much more reliable
  - Parameter of 10 often used
  - Fewer folds on *smaller and larger* sample size



- Number of folds increases runtime!
  - More-or-less linear with ***n***
  - Might not be that critical – why?
    - Can be parallelised

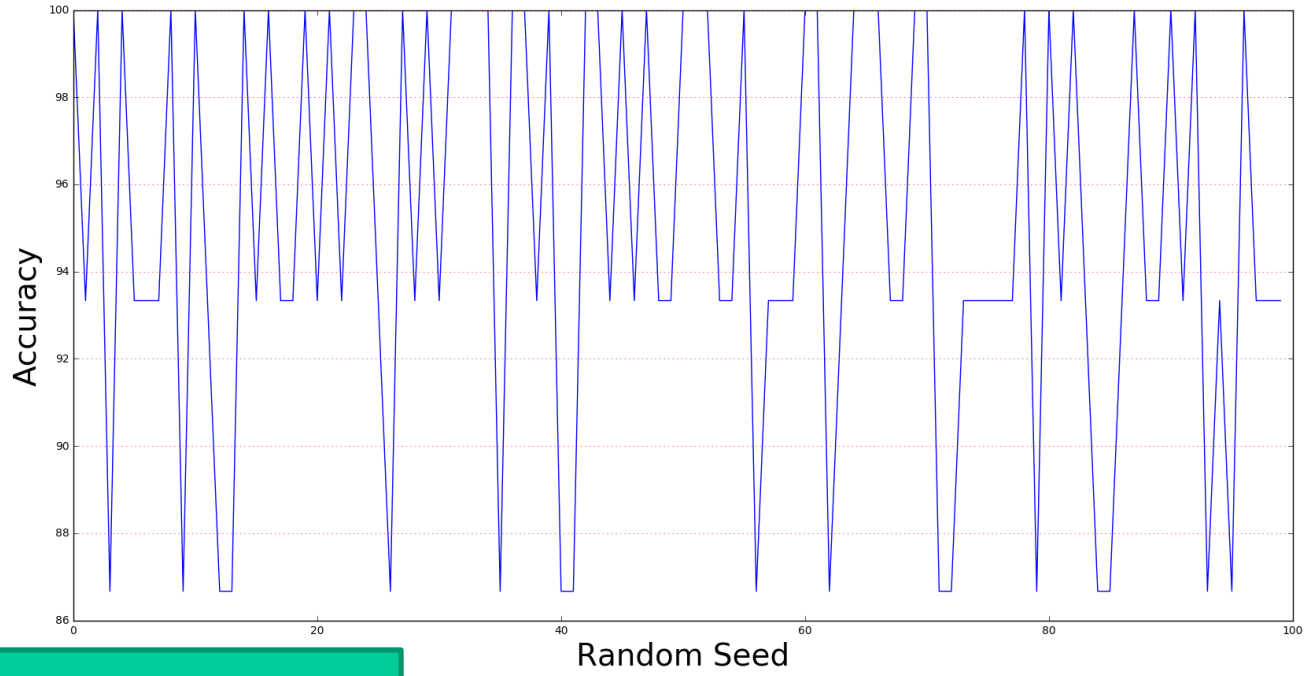
Cross validation

- Cross-validation
  - Smooth effects of random influence in training/test split
  - Example: **Iris Data, *k-nn (k=15)***, random seed 1..100
    - Average Accuracy: 96.67 (+/- 9.8883) [range: 80-100]
    - 10-fold CV: 96.67 (+/- 8.94)



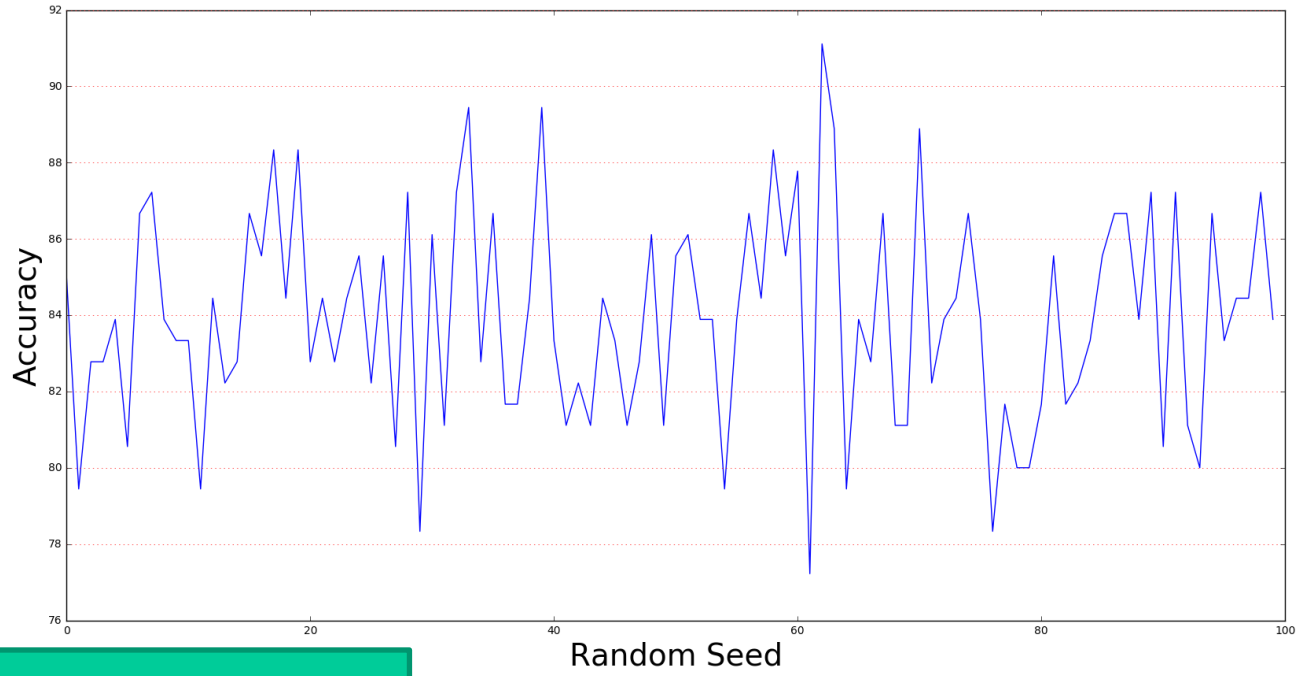
Cross validation

- Cross-validation
  - Smooth effects of random influence in training/test split
  - Example: **Iris Data, *Naive Bayes***, random seed 1..100
    - Average Accuracy: 95.00 (+/- 9.6839) [range: 86.67-100]
    - 10-fold CV: 95.33 (+/- 12.00)



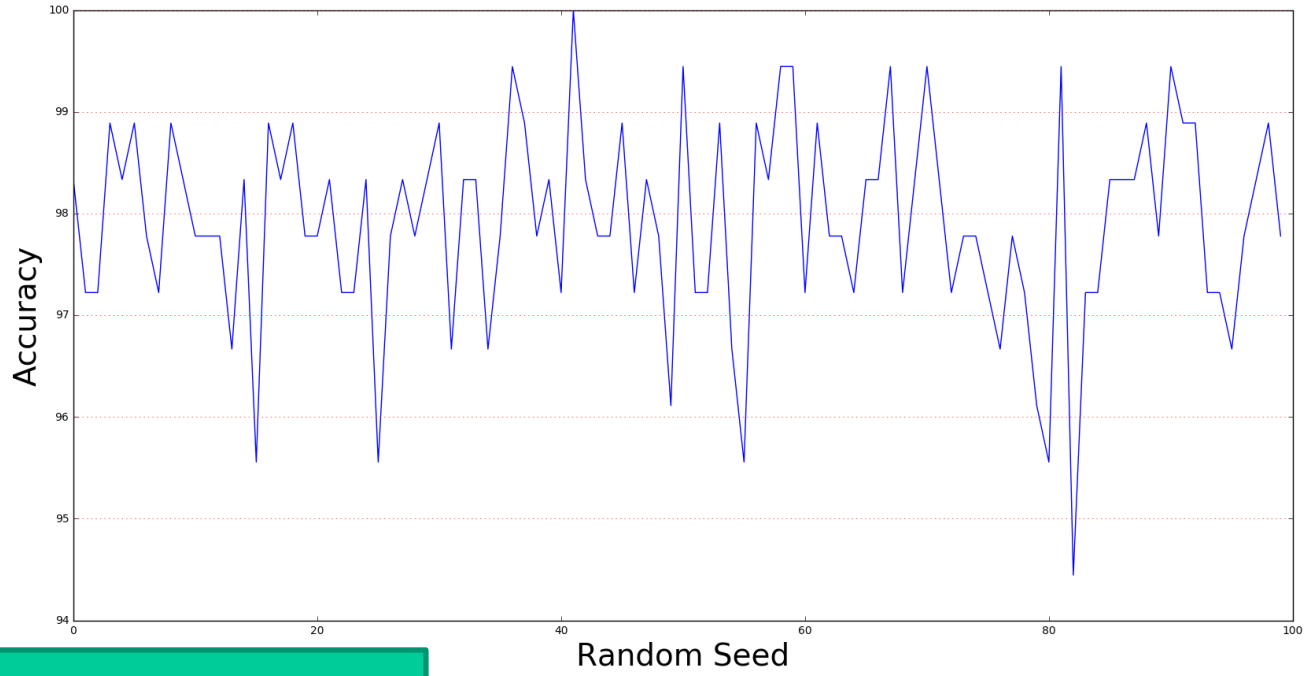
Cross validation

- Cross-validation
  - Smooth effects of random influence in training/test split
  - Example: **Digit Data, *k-nn (k=15)***, random seed 1..100
    - Average Accuracy: 97.90 (+/- 2.0097) [range: 94.44-100]
    - 10-fold CV: 97.99 (+/- 1.96)



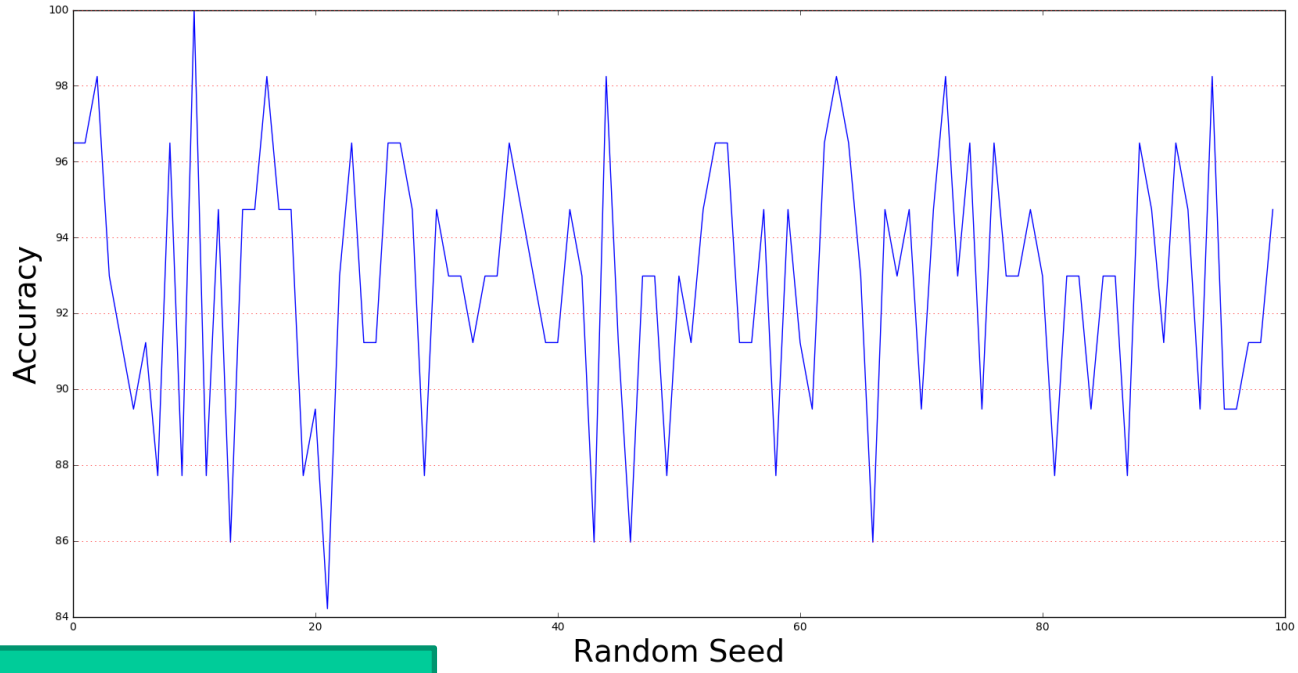
Cross validation

- Cross-validation
  - Smooth effects of random influence in training/test split
  - Example: **Digit Data, *Naive Bayes***, random seed 1..100
    - Average Accuracy: 83.93 (+/- 5.6167) [range: 77.22-91.11]
    - 10-fold CV: 84.03 (+/- 8.02)



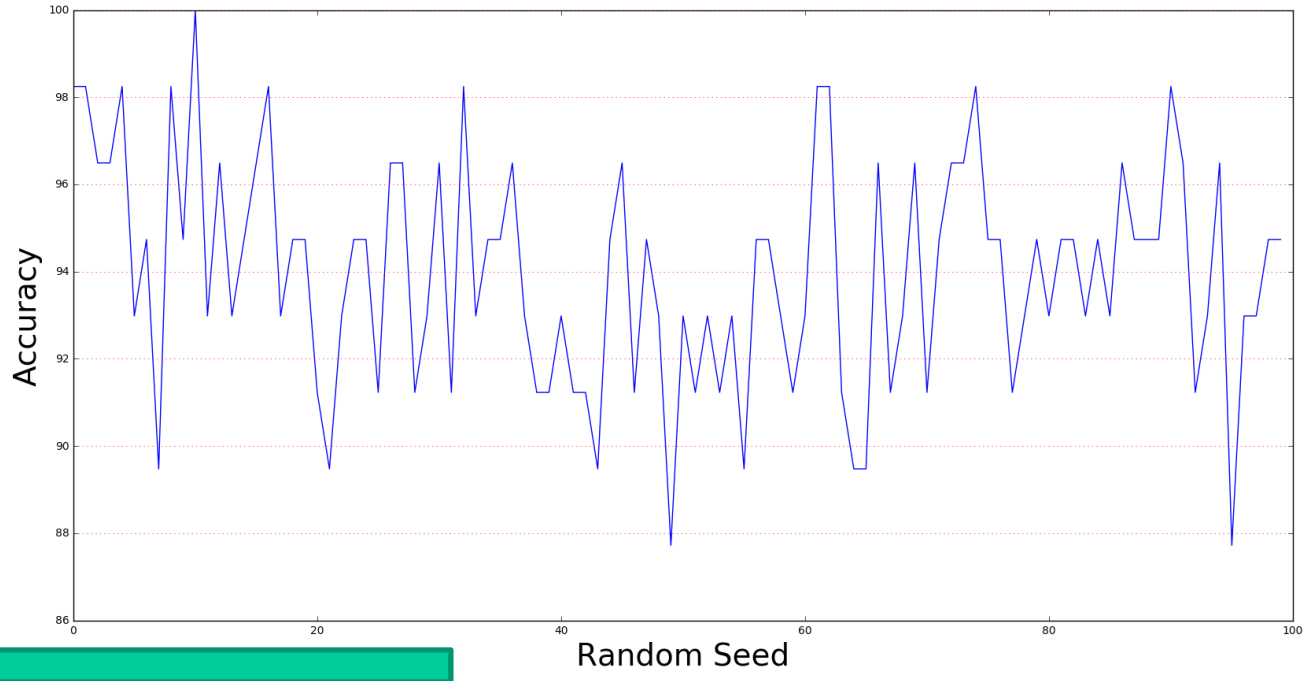
Cross validation

- Cross-validation
  - Smooth effects of random influence in training/test split
  - Example: **Cancer Data, *k-nn (k=15)***, random seed 1..100
    - Average Accuracy: 92.81 (+/- 6.7401) [range: 84.21-100]
    - 10-fold CV: 93.29 (+/- 8.1)



Cross validation

- Cross-validation
  - Smooth effects of random influence in training/test split
  - Example: **Cancer Data, *Naive Bayes***, random seed 1..100
    - Average Accuracy: 93.96 (+/- 5.2355) [range: 87.719-100]
    - 10-fold CV: 93.83 (+/- 4.33)



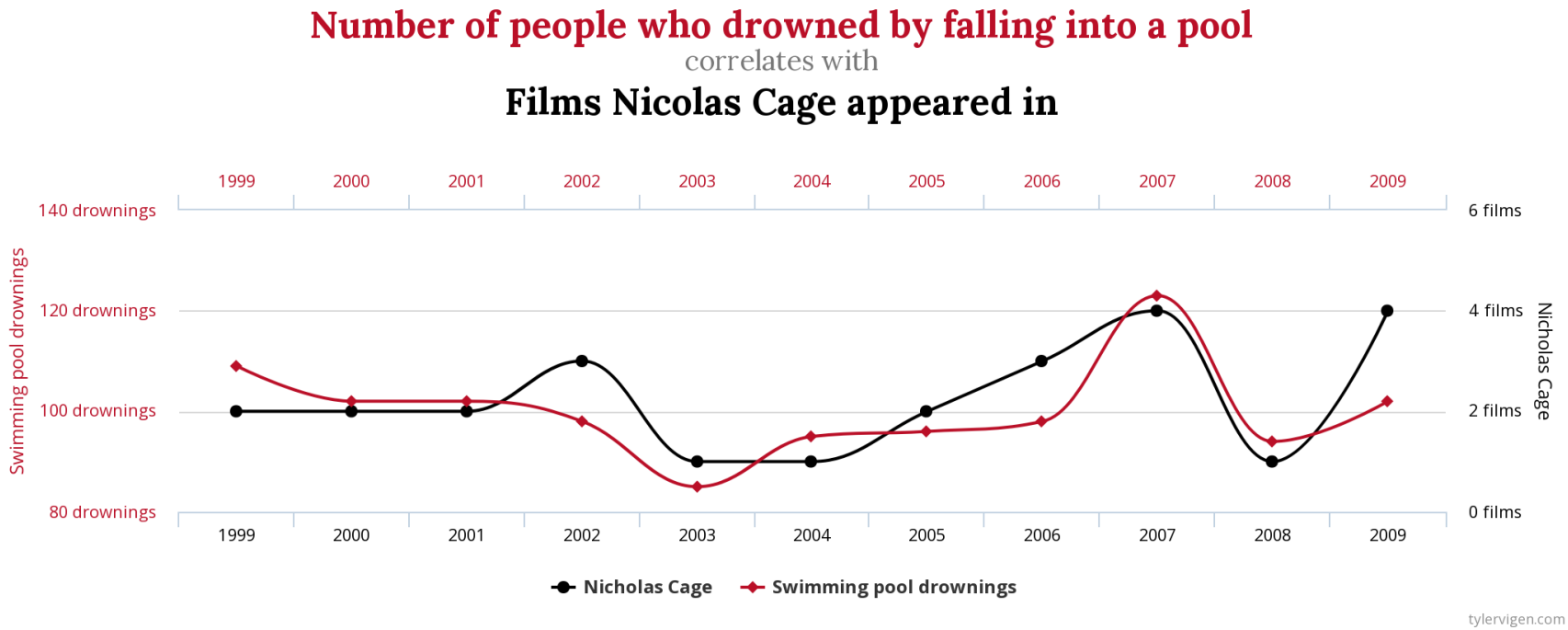
Evaluation: outlook

- Micro vs. macro averaging
  - Confusion matrix
  - Cost functions
- Other evaluation measures
- ROC curves
- Bootstrapping
- Significance testing

- Evaluation measures for regression

Meaningful?

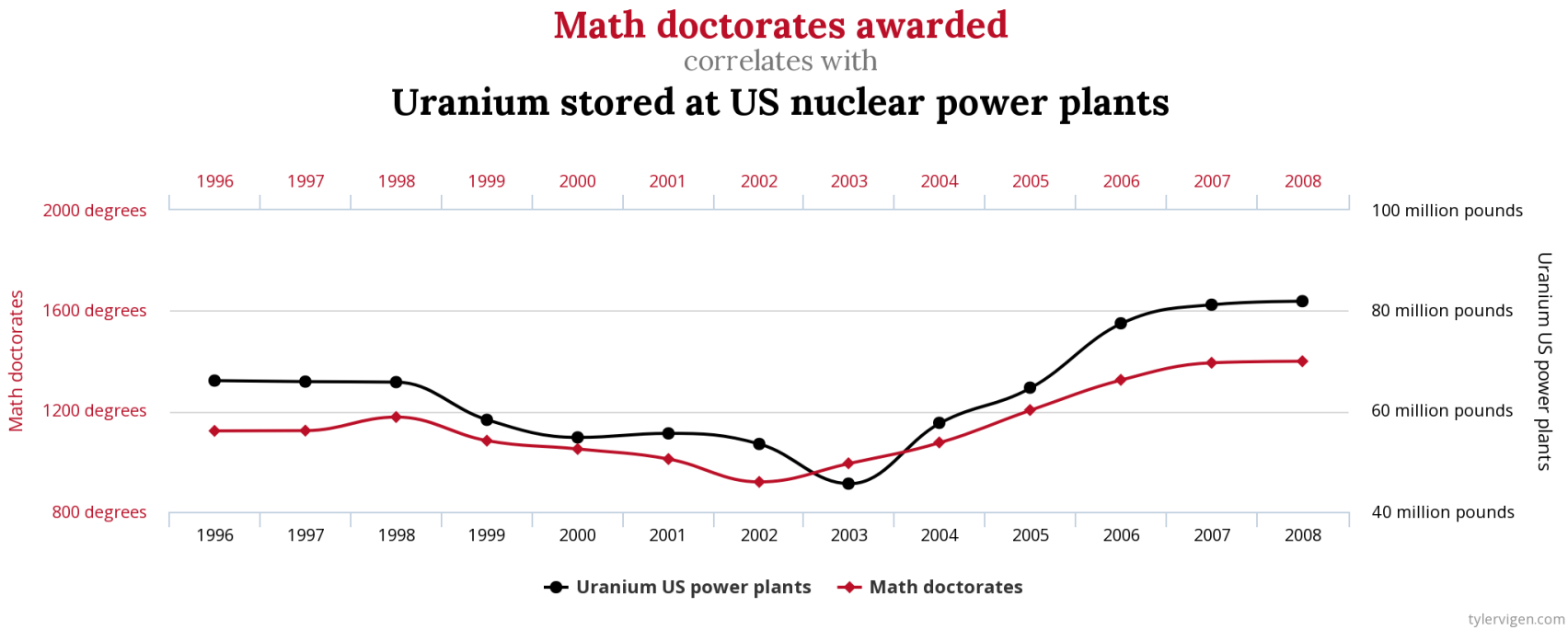
- Not every model is actually use/meaningful ☺



http://www.tylervigen.com/spurious-correlations

Meaningful?

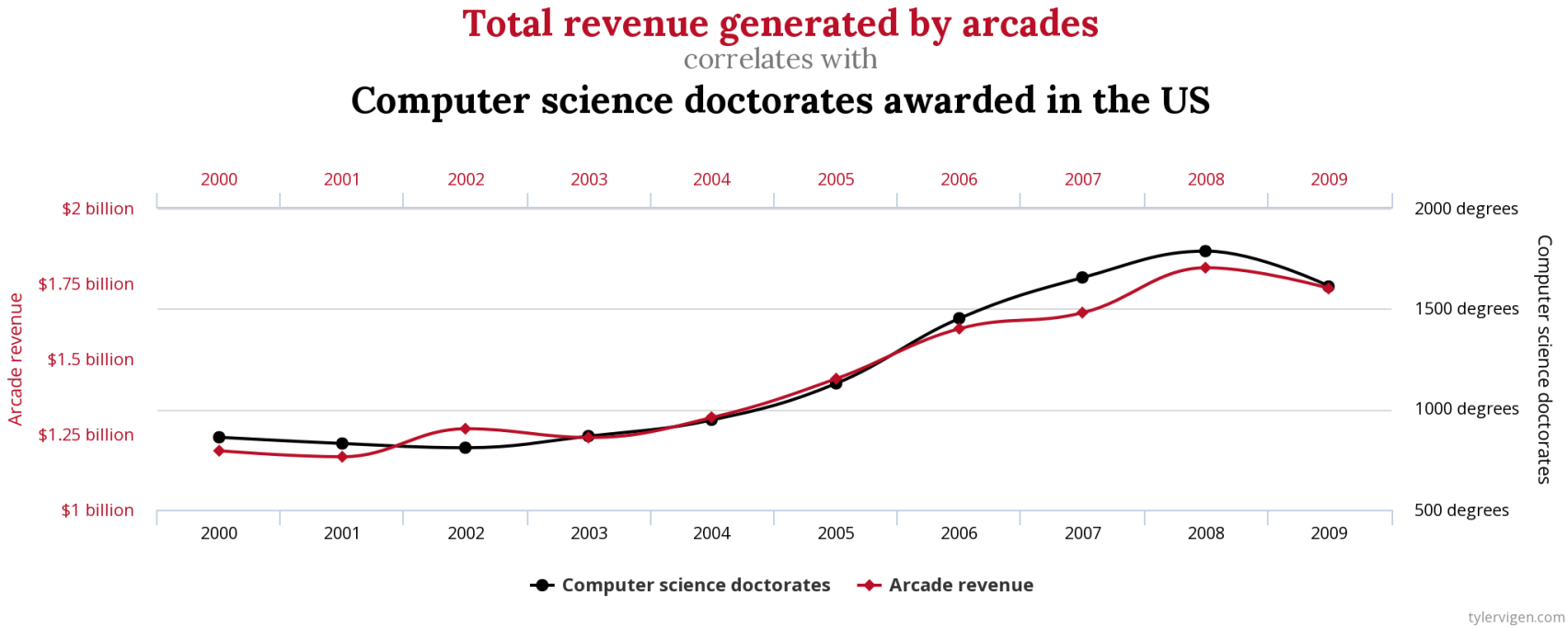
- Not every model is actually use/meaningful ☺



http://www.tylervigen.com/spurious-correlations

Meaningful?

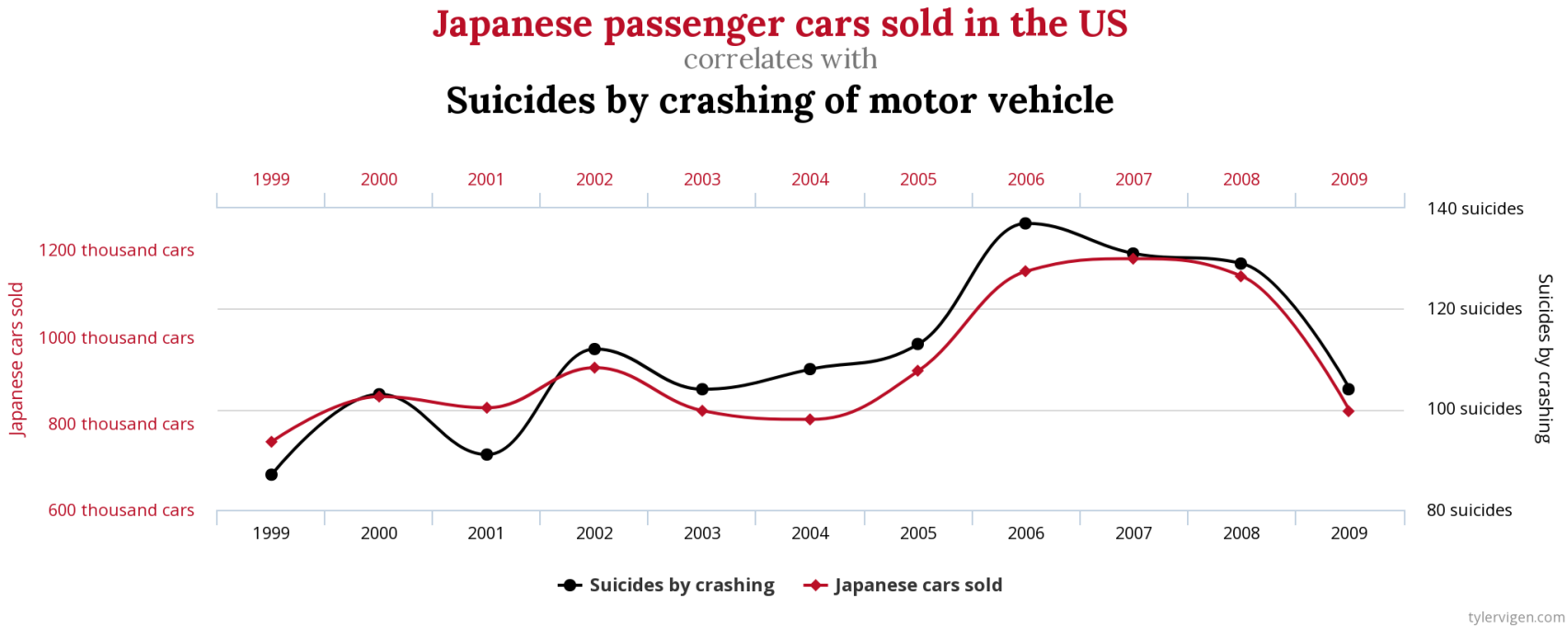
- Not every model is actually use/meaningful ☺



http://www.tylervigen.com/spurious-correlations

Meaningful?

- Not every model is actually use/meaningful ☺



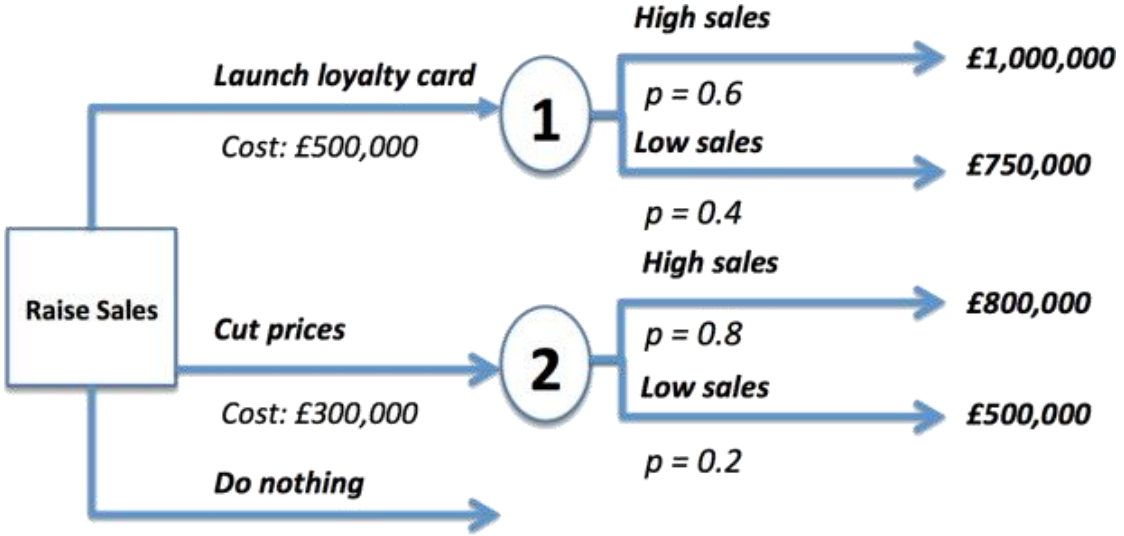
http://www.tylervigen.com/spurious-correlations

Outline

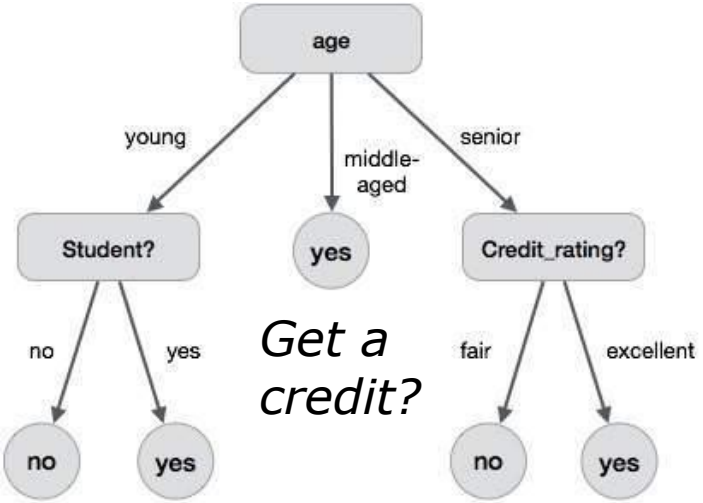
- Short recap
- K-nn (continued)
- Data preparation
- Evaluation (intro)
- Decision Trees
- Evaluation (continued)

Decision Trees

- In general: decision support tool
- Tree-like graph, flowchart
- Models decisions and their outcome
  - Potentially including probabilities, costs, …
  - Leaf nodes: events / outcomes
  - Non-leaf nodes: decisions



Decision Trees in ML



- Rather old, simple model
- "Flowchart-style"

- Easy to understand model & rules
  ➔ popular in decision support (management, …)

- Classification & regression
- Categorical (size: small/medium/large) ***and*** numerical (size in cm) data

- *Simple version?*
  1R – just one (root) node

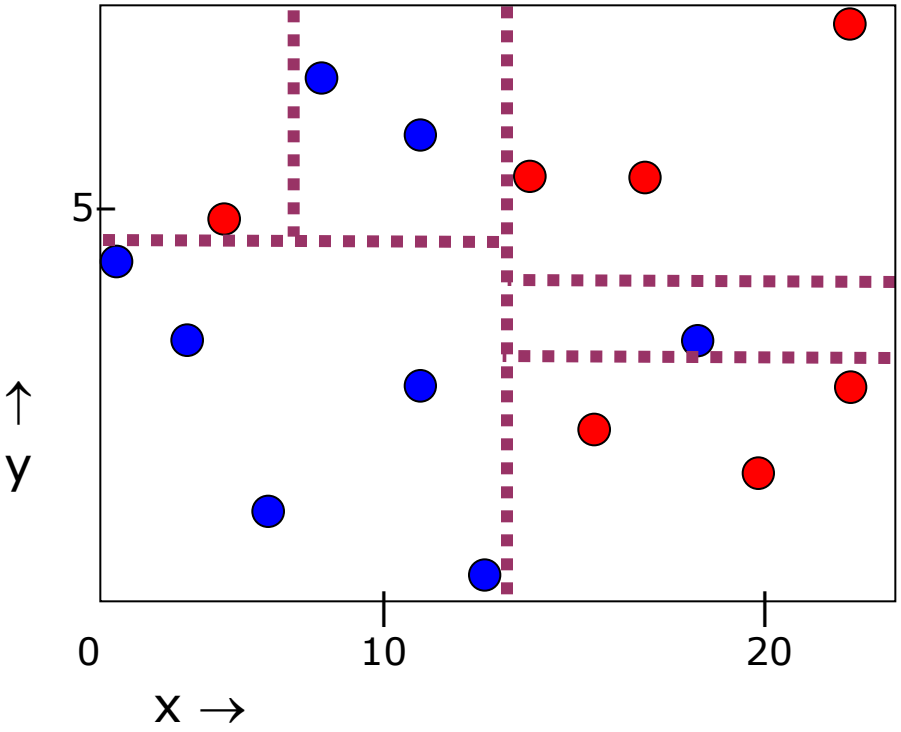
Decision Trees in ML

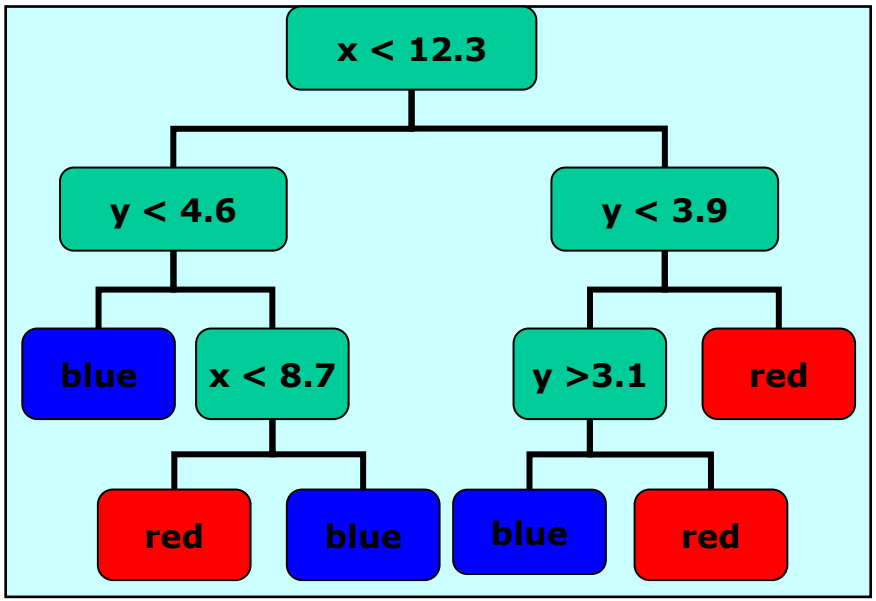
- *Can be manually modelled by domain experts*
- In ML: ***learn*** tree from training **data**
- Use tree to *predict* classes for unlabeled data
  - Leaf nodes: define outcome (class / regression value)
  - Inner nodes: rules, based on attributes and their values

- Training: find rules that recursively split data into sub-spaces
  - Until we have reached a stopping criterion
  - Different strategies for evaluating splits
  - *Well-known algorithms:* CART, ID3 (1986), C4.5 (1996)
- Classification: traverse through tree until reaching a leaf node
  - Compute prediction based on training data in that leaf node
    - E.g. majority voting of the class

Decision Trees: Example

- Numerical, 2-dimensional data (x, y), two (output) classes
- Training: find rules that recursively split data into sub-spaces (partition)
  - Until each sub-space contains samples from only one class





Decision Trees: Algorithm

- For each leaf node
  - If not all data from the same class
    - For ***each*** attribute
      - Identify possible splits of samples into subspaces
    - Compute ***best*** split (over all attributes!)
      - Based on a split goodness measure/criterion
  - Until data in all leaf nodes is from the same class

Decision Trees: Algorithm

- Popular measures to compute best split
  - Error rate
  - Information gain
  - Gini impurity (Gini index)
  - *...*

Decision Trees: Algorithm

- Popular measures to compute best split
  - Error rate
  - Information gain
  - *Gini impurity (Gini index)*

Decision Trees: Algorithm

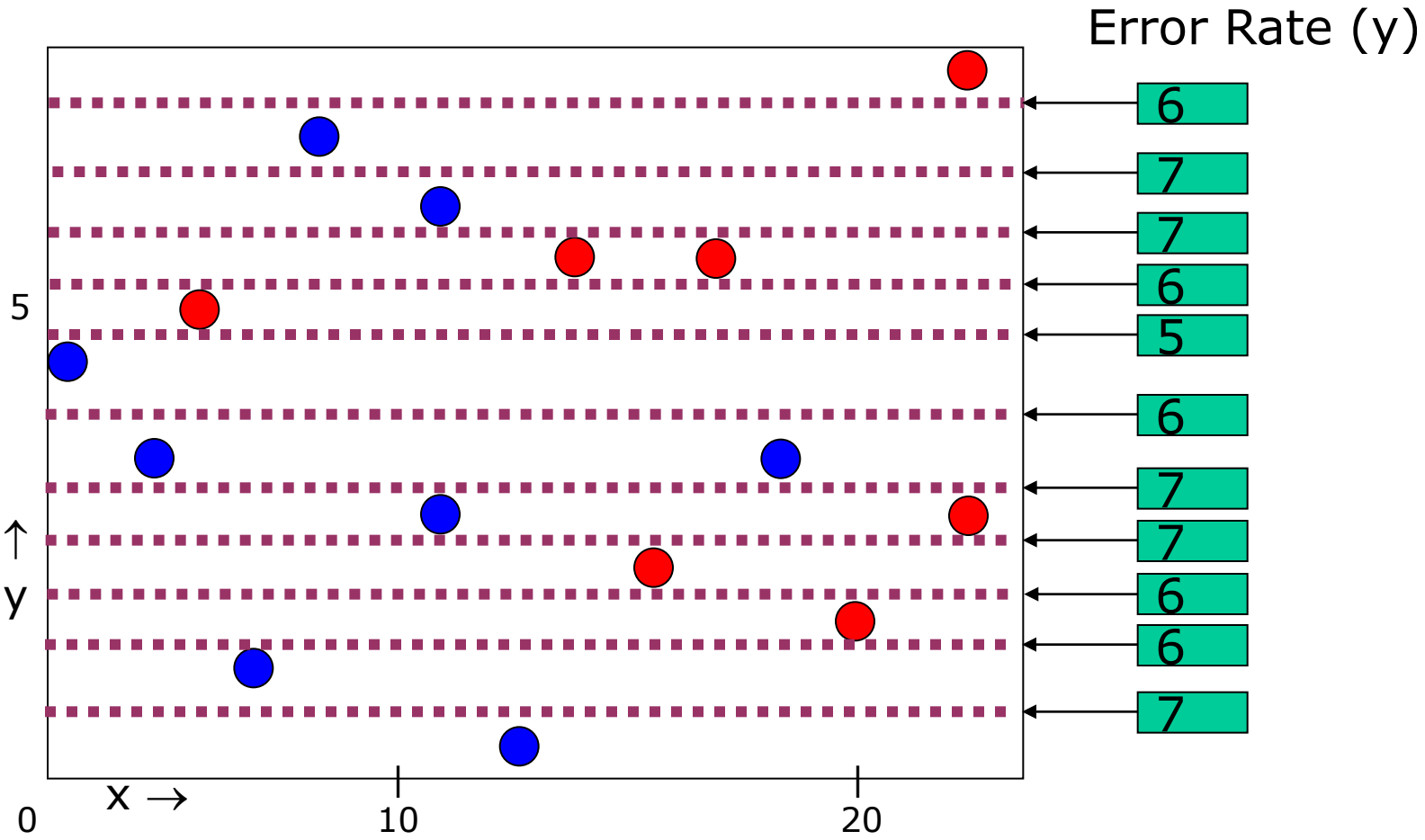
- Error rate
  - Related to accuracy (“complement“)
  - Absolute error rate: simply count the number of classification errors when performing a split
    - *How to express in TP/TN/FP/FN ?*

	true	false
true	True positive (TP)	False positive (FP, Type I error)
false	False negative (FN, Type II error)	True negative (TN)

  - The lower the error, the better
  - Absolute error rate vs. relative error rate
    - Relative error in relation to total number of samples (0..1)
    - Absolute error: 0..n
      - *Semantic difference?*
    - Decision trees: often absolute error rate used

Decision Trees: error rate

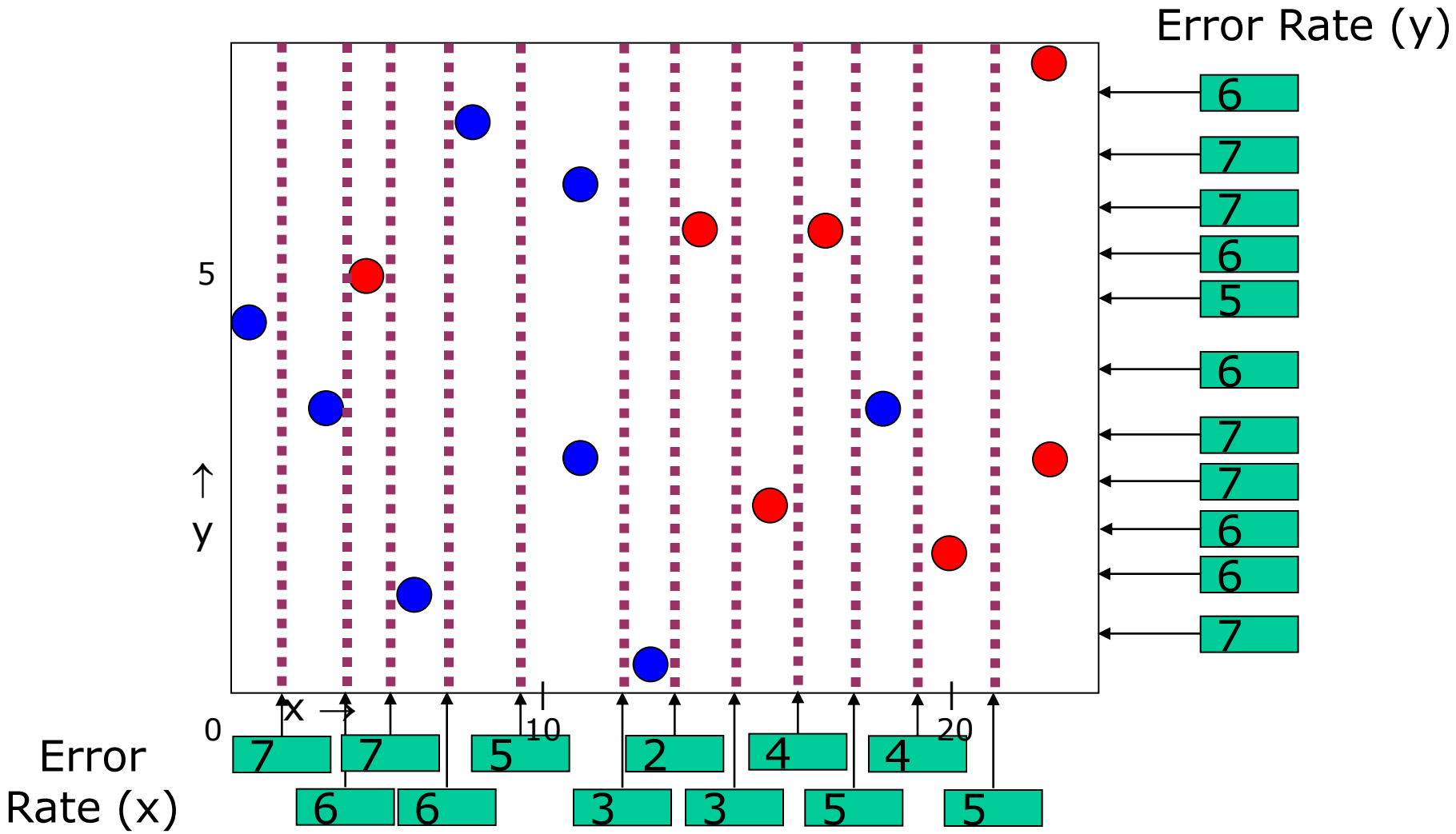
- 2-dimensional data (x, y), numerical values, two classes



- Find all possible splits (*how many in this example?*), compute error

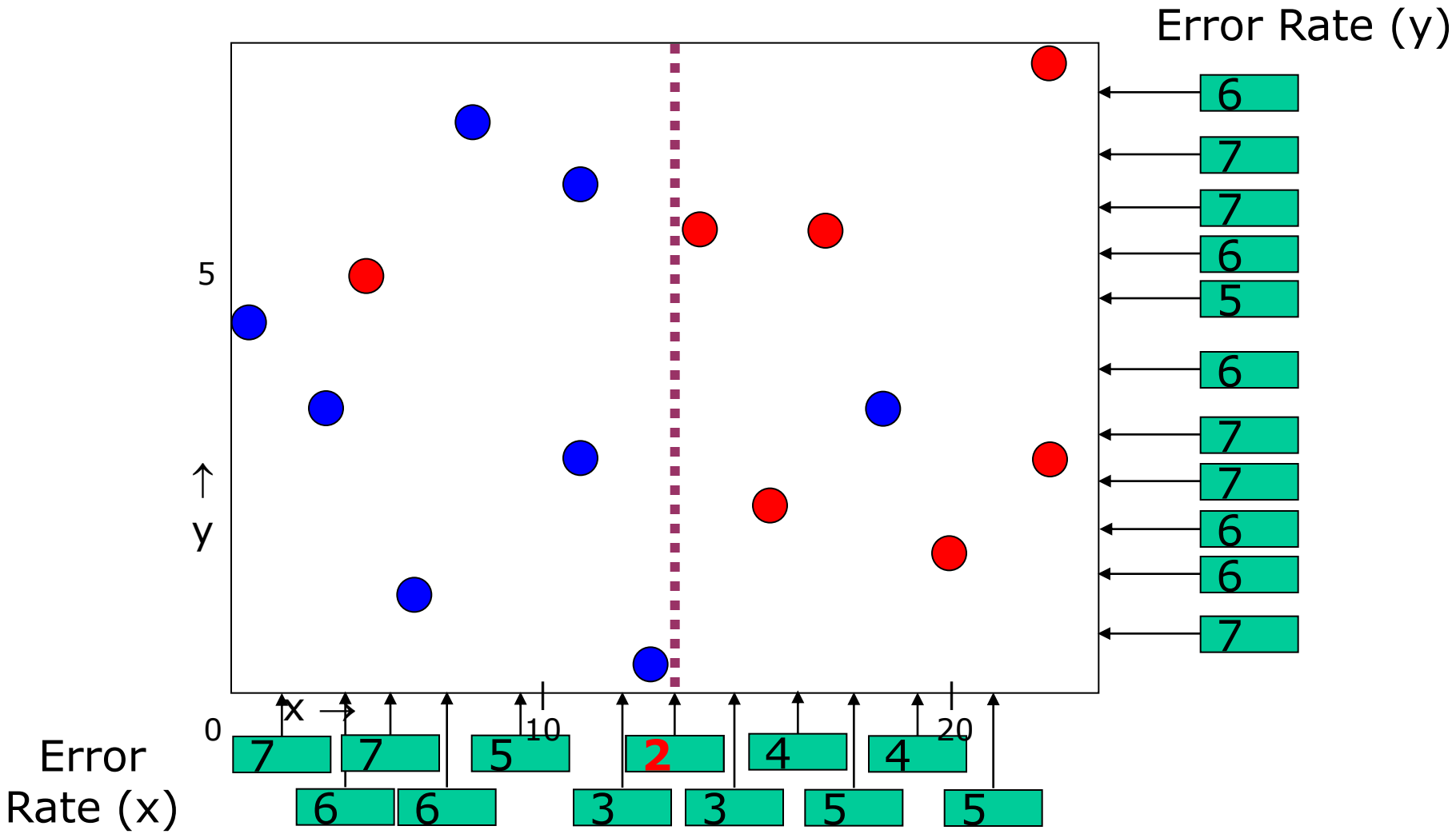
Decision Trees: error rate

- 2-dimensional data (x, y), numerical values, two classes

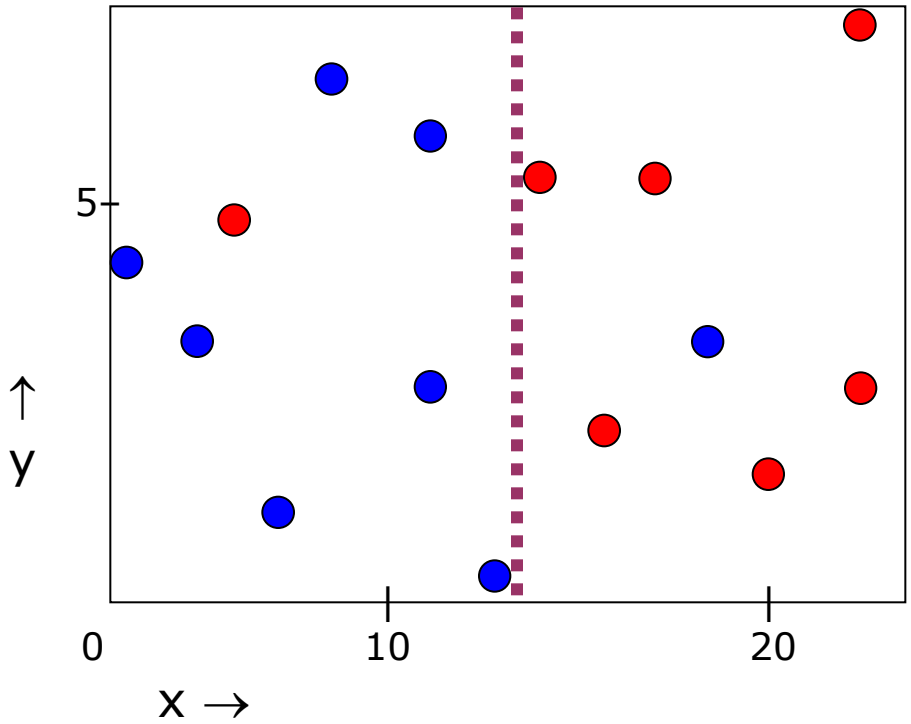


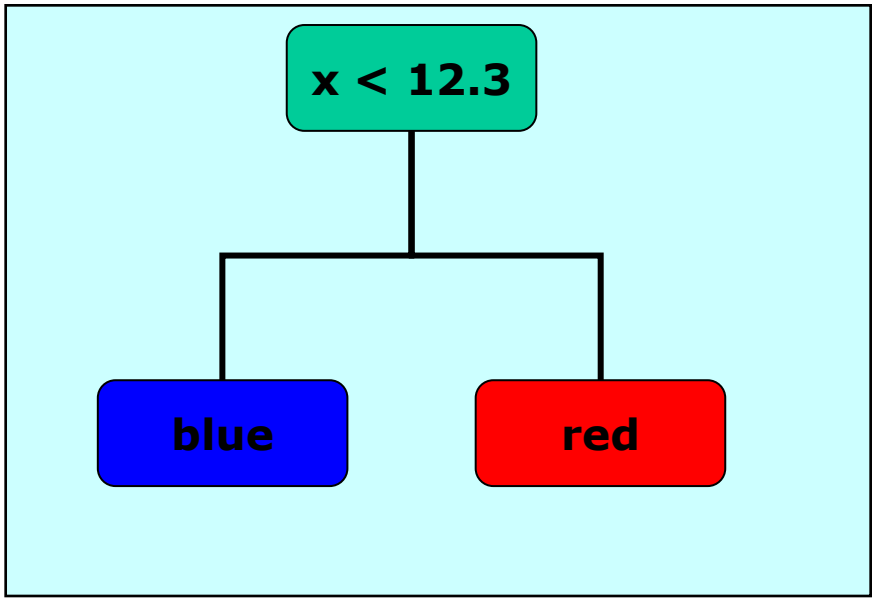
Decision Trees: error rate

- 2-dimensional data (x, y), numerical values, two classes



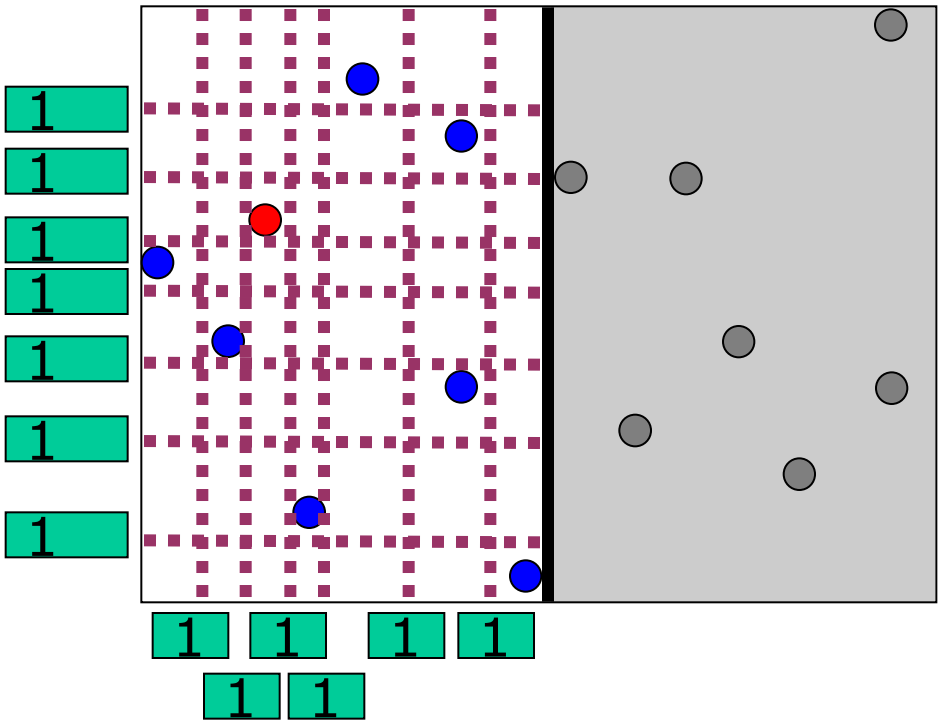
Decision Trees: error rate

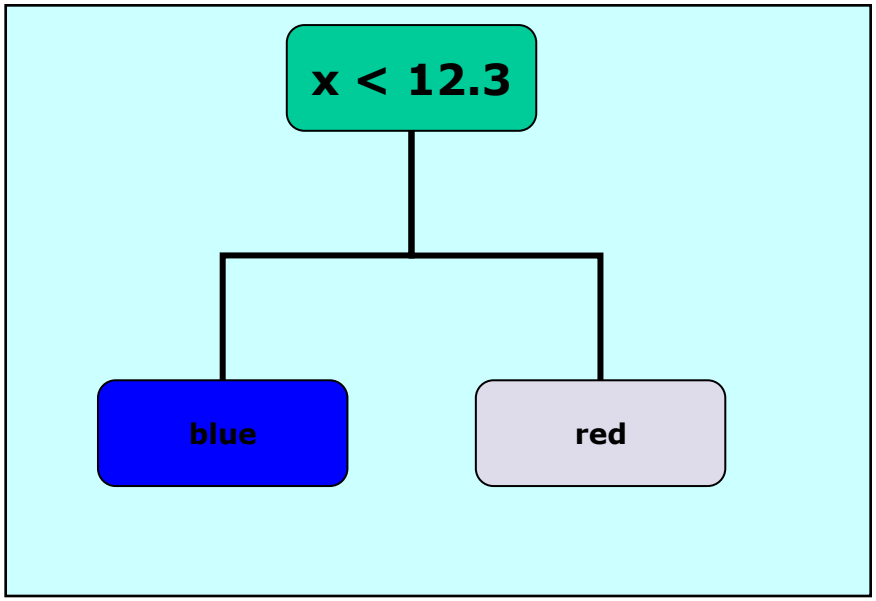




- Split for top-level (root) node found ✓
- *If we stop here – how is this classifier called?*
- *Next step(s)?*

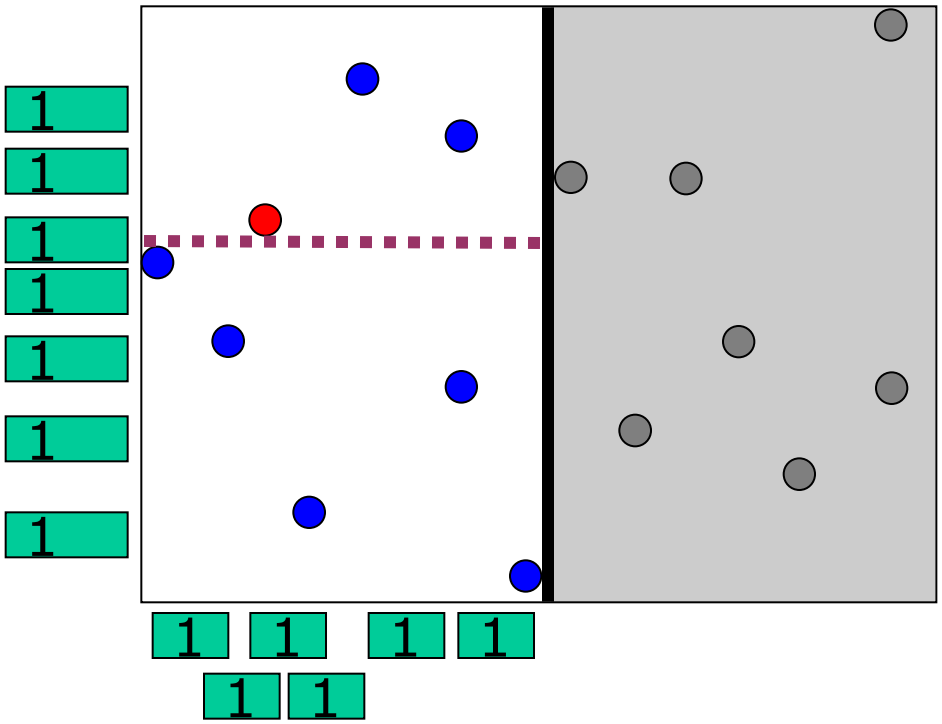
Decision Trees: error rate

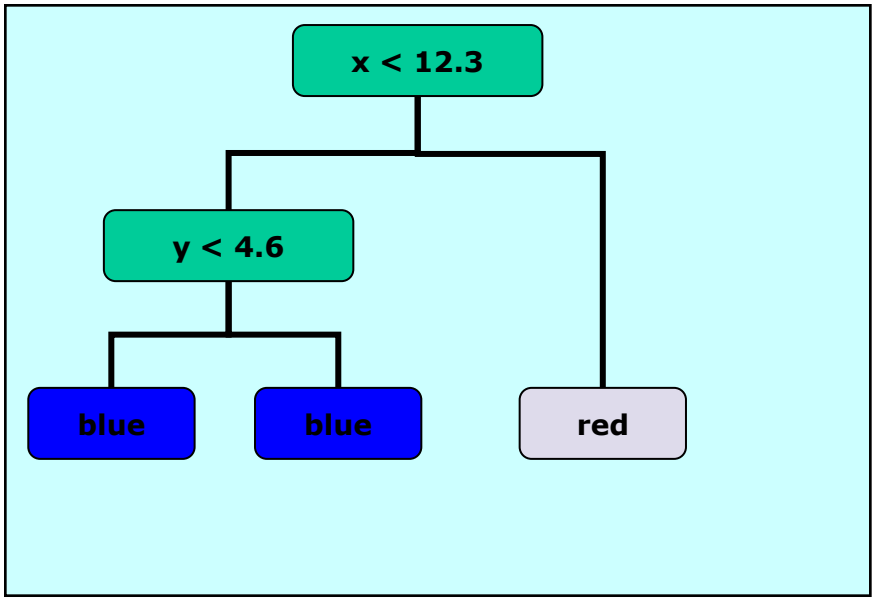




Which split?

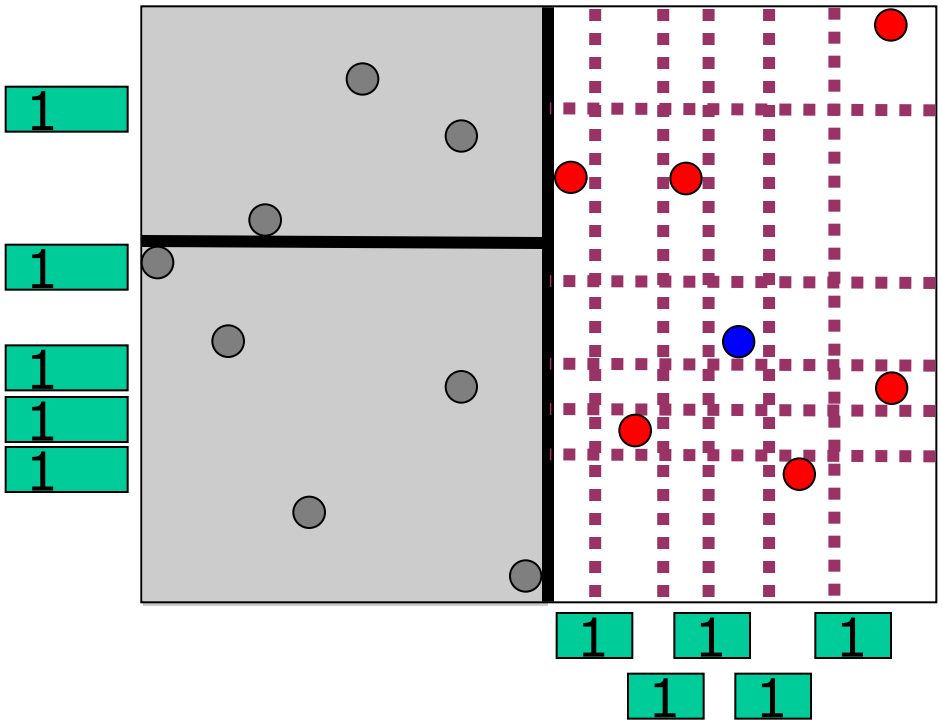
Decision Trees: error rate

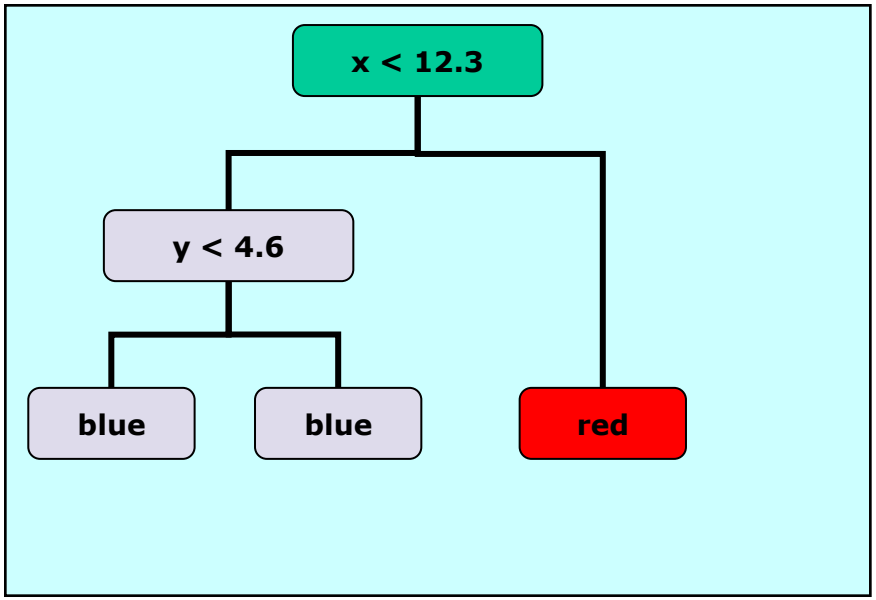




Decision for split in case of equality – e.g. random

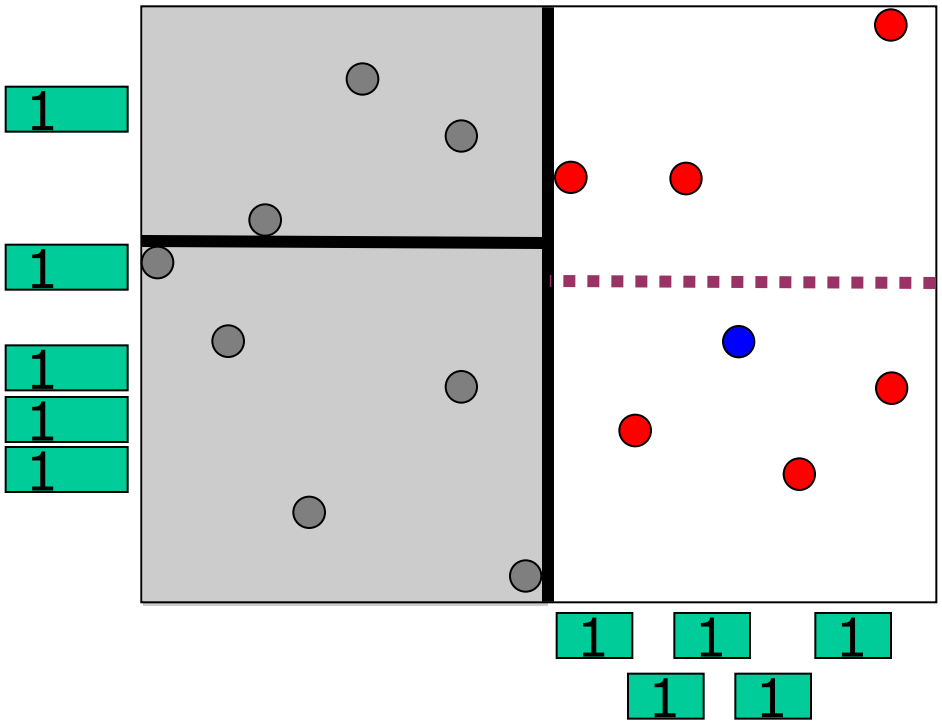
Decision Trees: error rate

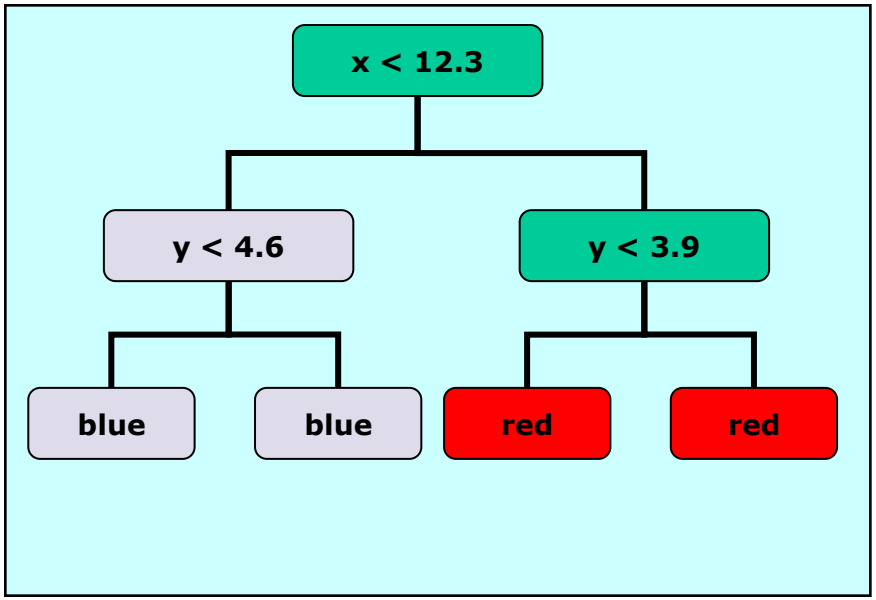




Which split?

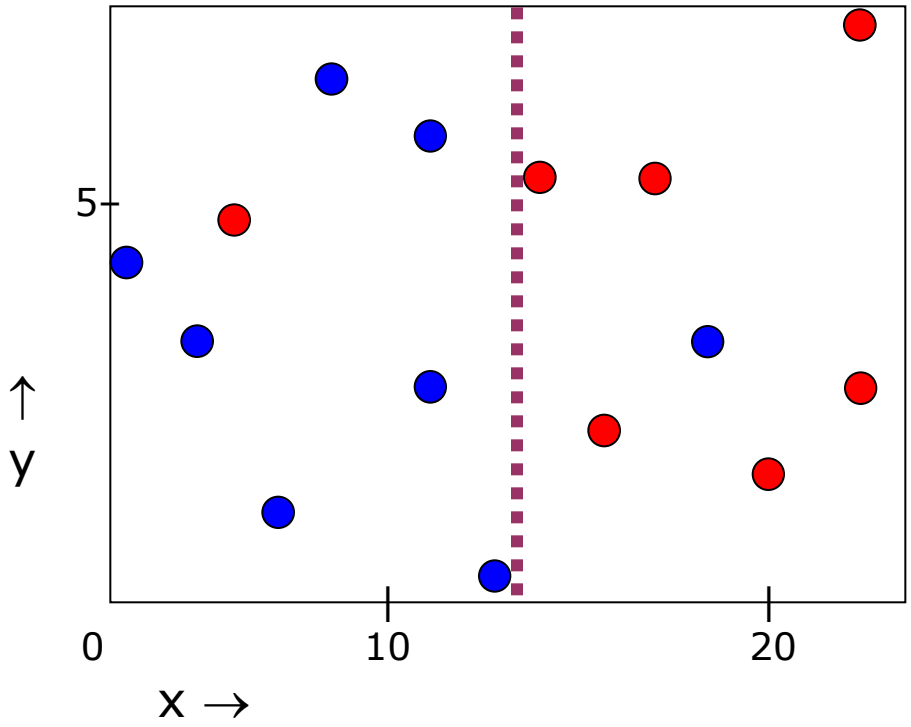
Decision Trees: error rate

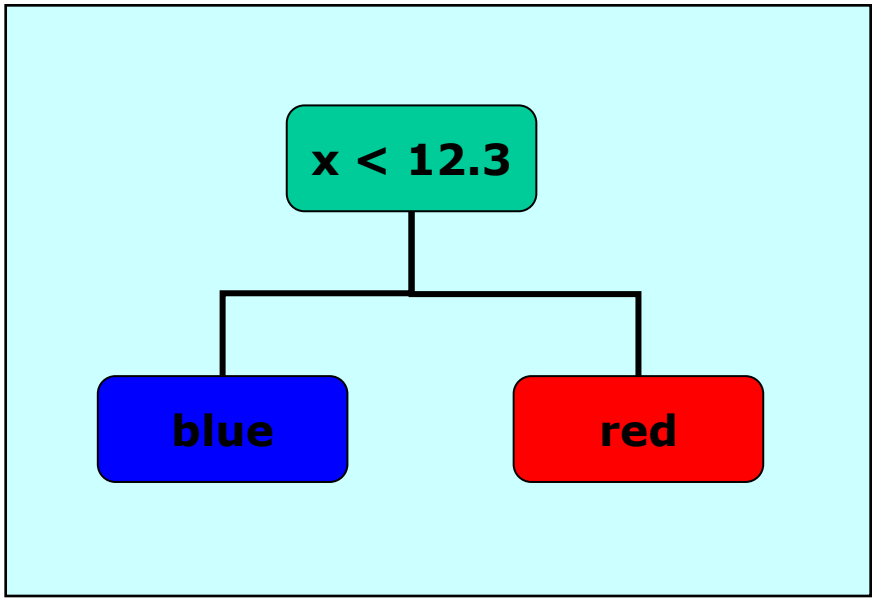




Decision Trees: error rate

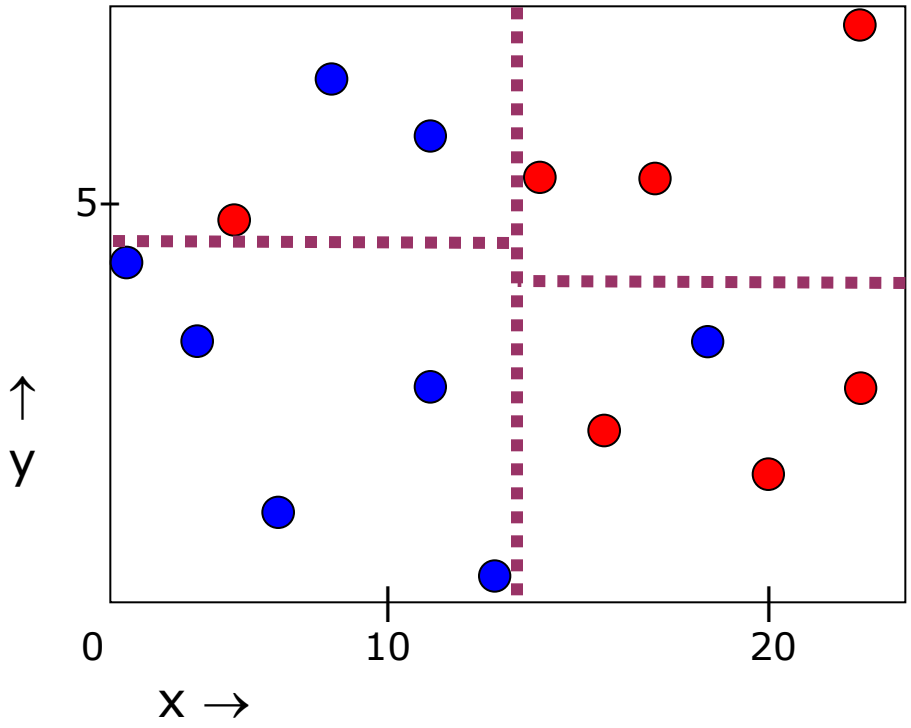
- Tree training, level 1

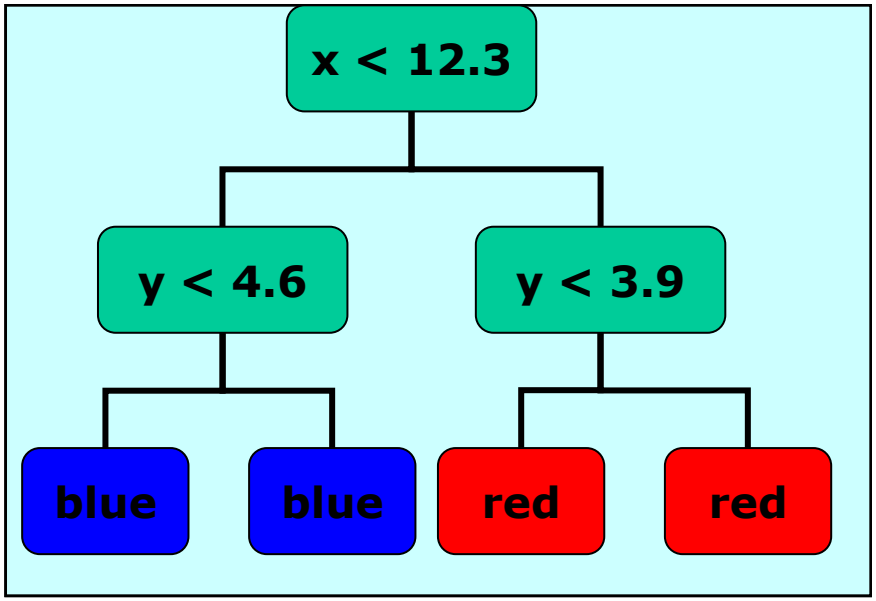




Decision Trees: error rate

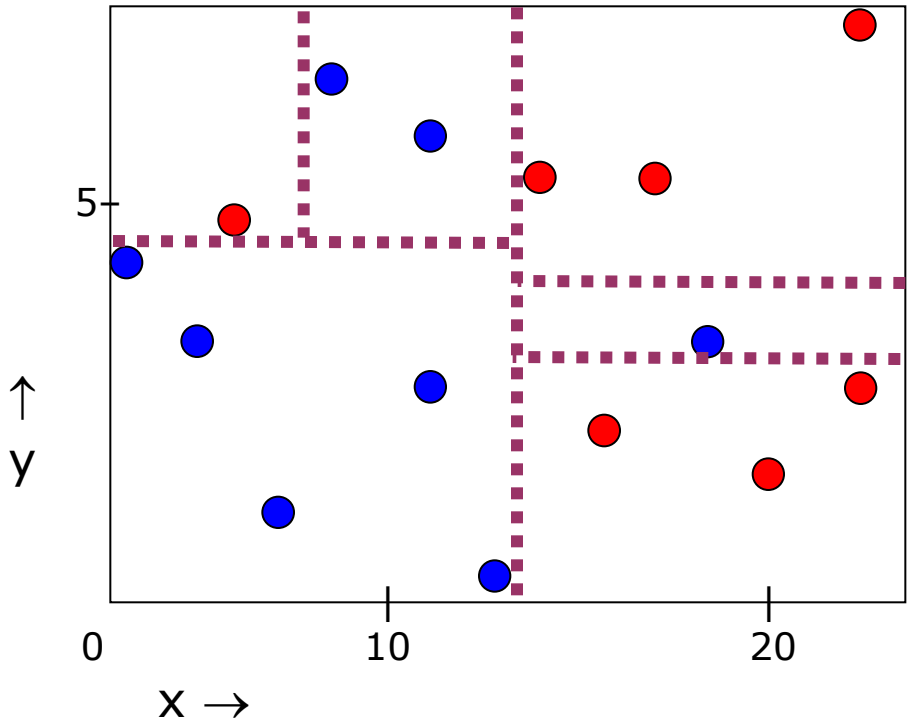
- Tree training, level 2

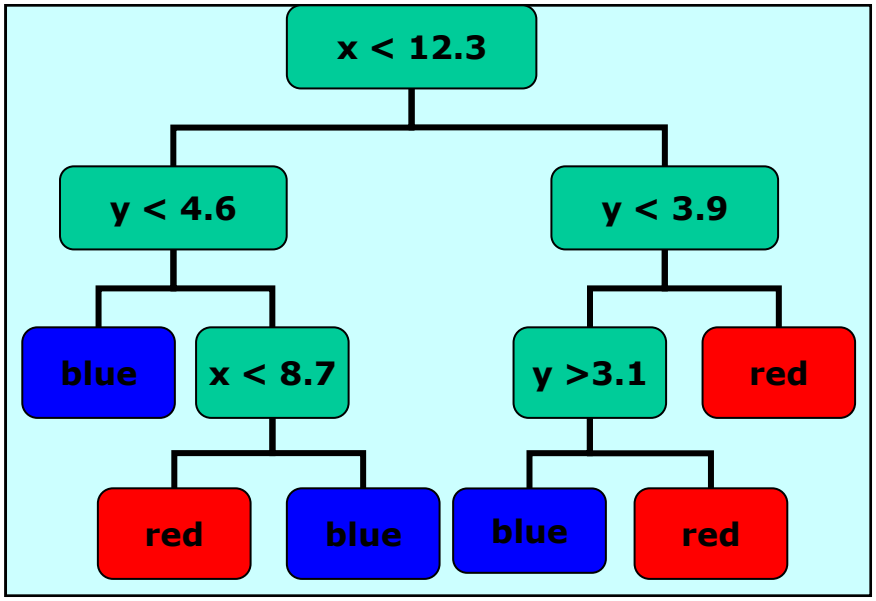




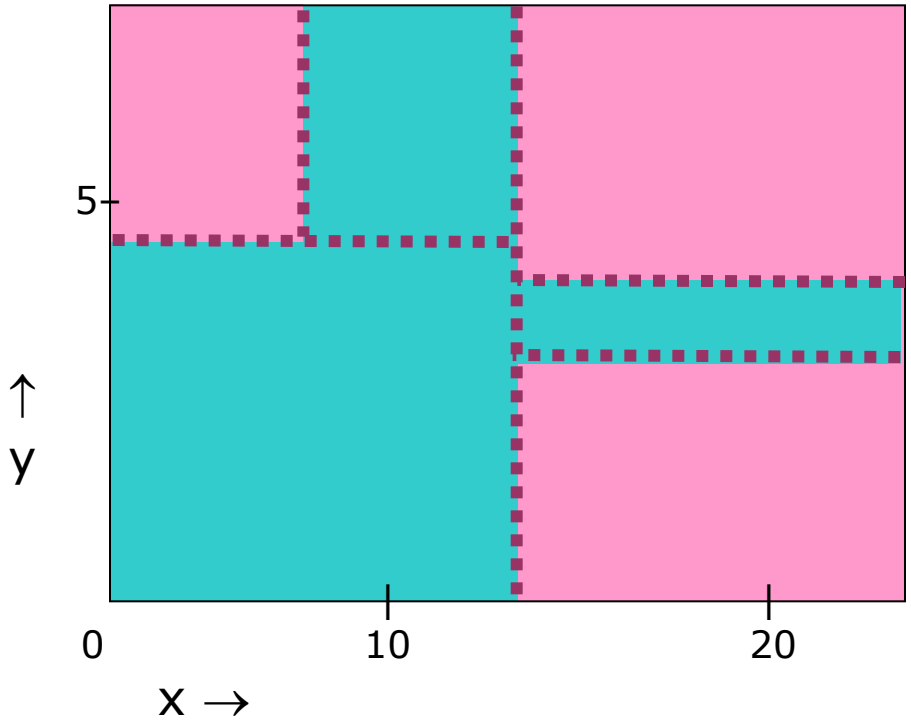
Decision Trees: error rate

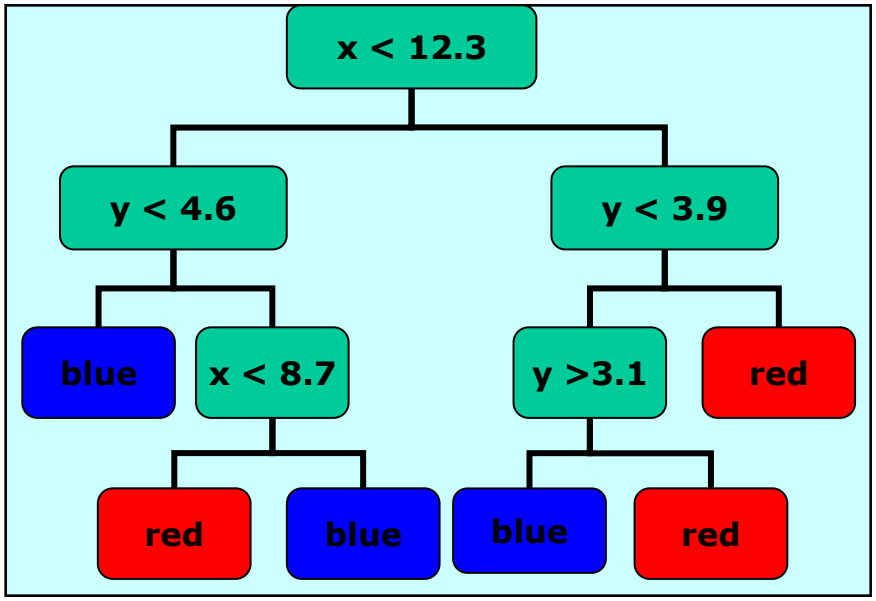
- Tree training, level 3: completely built tree





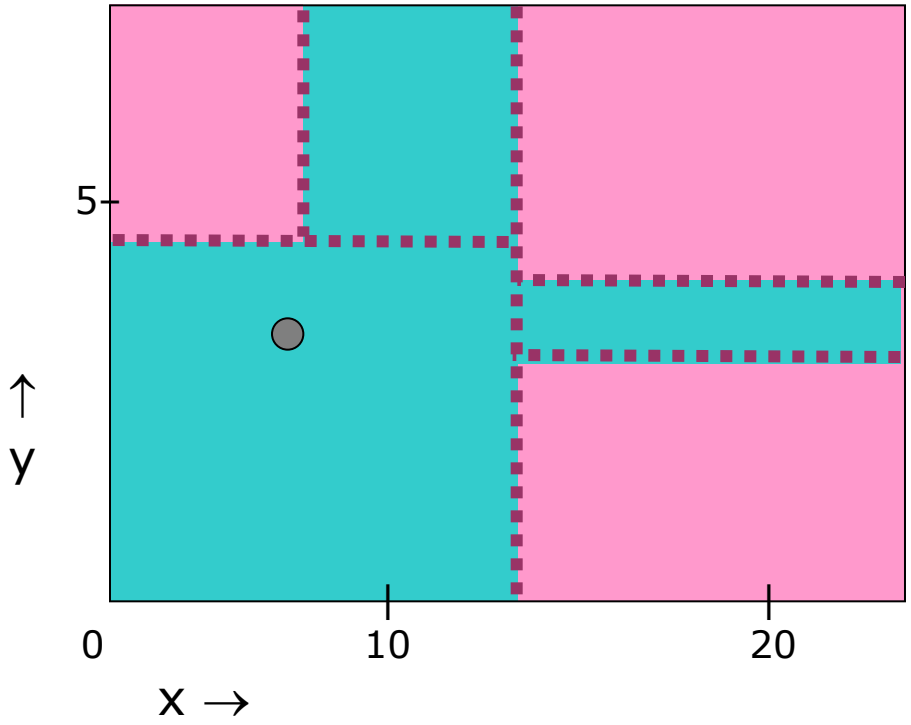
Decision Trees: Classification

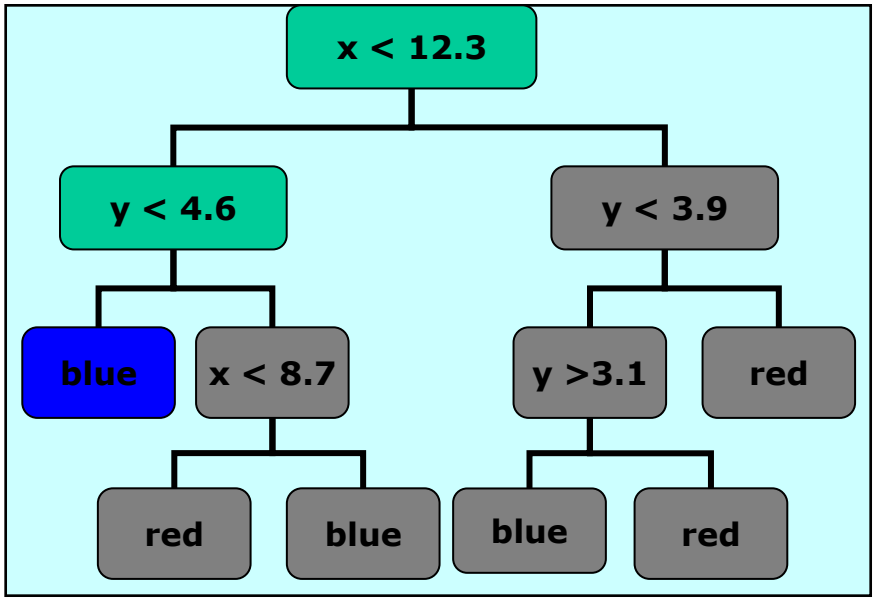




How to classify unknown items?

Decision Trees: Classification

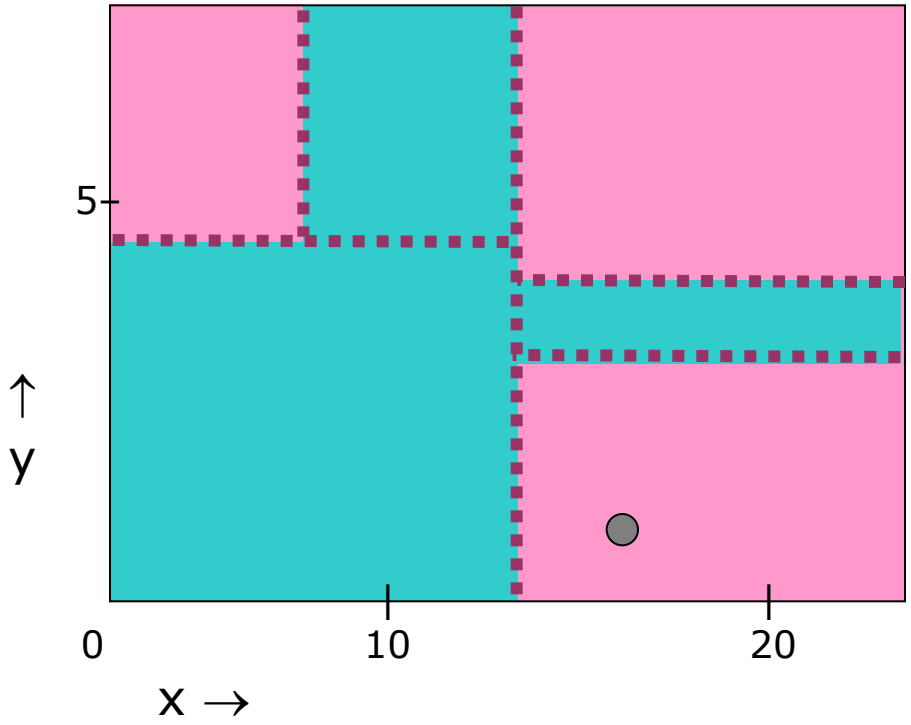


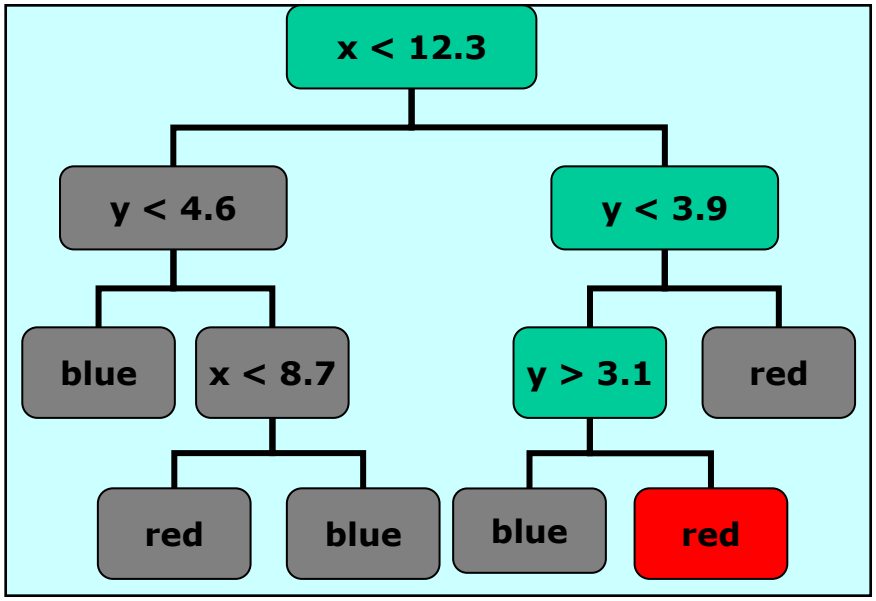


➔ Decend the tree until leaf-node

➔ Use majority of class in that leaf node

Decision Trees: Classification





➔ Decend the tree until leaf-node

➔ Use majority of class in that leaf node

Decision Trees: Algorithm

- Popular measures to compute best split
  - Error rate
  - Information gain
  - *Gini impurity (Gini index)*

Information Theory & Entropy

- Introduced by Claude Shannon (1948)
  - Original for compression & reliable communication
  - Applications in statistical inference, NLP, cryptography,…
- Entropy: # of bits needed for communication
  - Absolute limit for best lossless compression

- Measure of uncertainty
  - High probability – low entropy
- Concerned with measuring actual information vs. redundancy

What is „Information Entropy“?

- Entropy – measure of uncertainty

- ML: measure for the “impurity” of a set
  - High Entropy ➔ bad for prediction
  - High Entropy ➔ needs to be reduced

$$H(X) = \mathrm{E}(\mathrm{I}(\mathrm{X})) = \sum_{i=1}^{n} p(x_i)\, I(x_i) = -\sum_{i=1}^{n} p(x_i) \log_2 p(x_i)$$

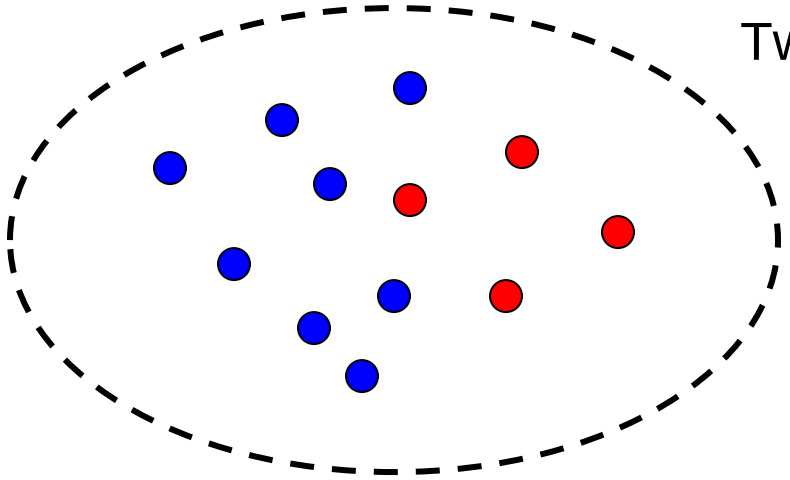
H … Entropy

E … Expected value

I(X) …information content of X

p(...) … probability function

Calculating *H(X)*: example



Two dimensional data, two classes

$$p(x_{red}) = \frac{4}{12} = 0.33$$

$$p(x_{blue}) = \frac{8}{12} = 0.67$$

$$H(X) = -\sum_{i=1}^{n} p(x_i) \log_2 p(x_i)$$

$$H(X) = -p(x_{red}) \log_2 p(x_{red}) - p(x_{blue}) \log_2 p(x_{blue})$$

$$H(X) = -\frac{1}{3} \times \log_2\left(\frac{1}{3}\right) - \frac{2}{3} \times \log_2\left(\frac{2}{3}\right)$$

$$= -\frac{1}{3} \times -1.58 - \frac{2}{3} \times -0.58$$

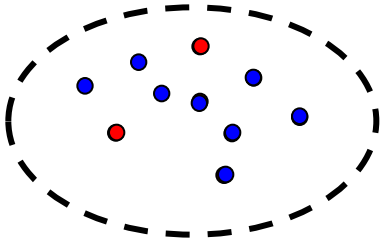
$$= 0.53 + 0.39$$

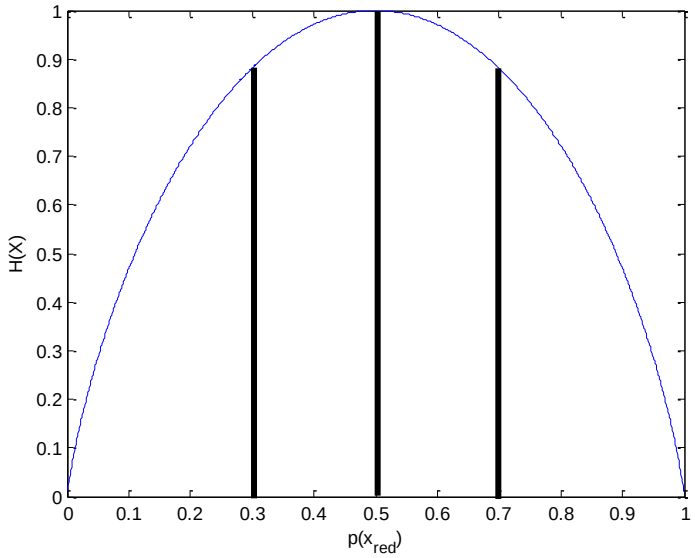
$$= 0.92$$

Remember:

$$\log_2(x) = \log(x) / \log(2)$$

H(X): Example values



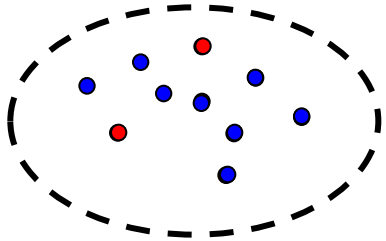


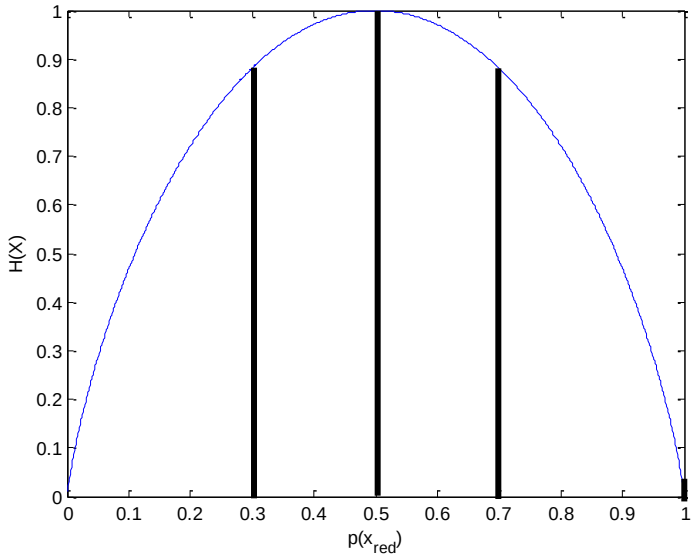
	$p(x_{red})$	$p(x_{blue})$	$H(X)$
I	0.5	0.5	?
II	0.3	0.7	?
III	0.7	0.3	?
IV	0	1	?

$$H(X) = -\sum_{i=1}^{n} p(x_i) \log_2 p(x_i)$$

Any guesses?

H(X): Example values



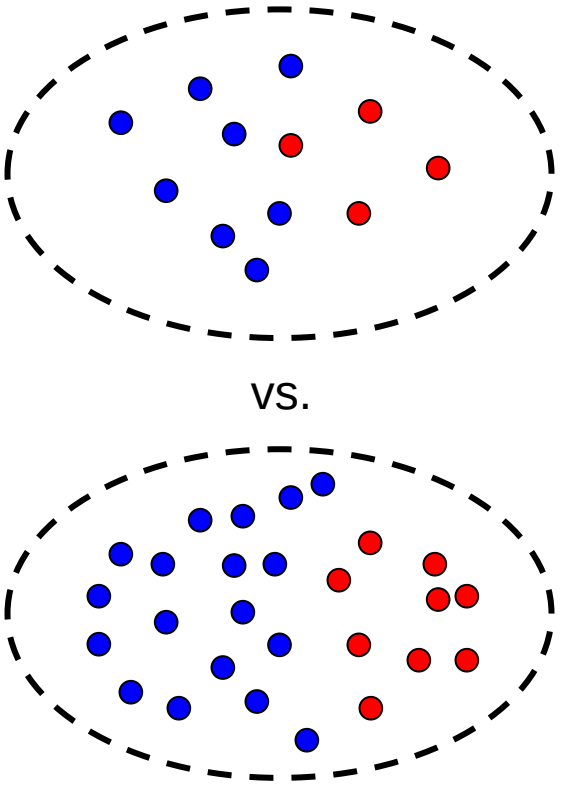


	$p(x_{red})$	$p(x_{blue})$	$H(X)$
I	0.5	0.5	?
II	0.3	0.7	?
III	0.7	0.3	?
IV	0	1	?

$$H(X) = -\sum_{i=1}^{n} p(x_i) \log_2 p(x_i)$$

H(X): Relative vs. absolute frequencies

What are the entropies of I and II?



vs.

$$p(x_{\text{red, I}}) = \frac{4}{12} = 0.33 \; ; \; p(x_{\text{blue, I}}) = \frac{8}{12} = 0.67$$

$$p(x_{\text{red, II}}) = \frac{9}{27} = 0.33 \; ; \; p(x_{\text{blue, II}}) = \frac{18}{27} = 0.67$$

Dataset	red	blue
I	8	4
II	18	9

$$\Rightarrow H(X_I) = H(X_{II})$$

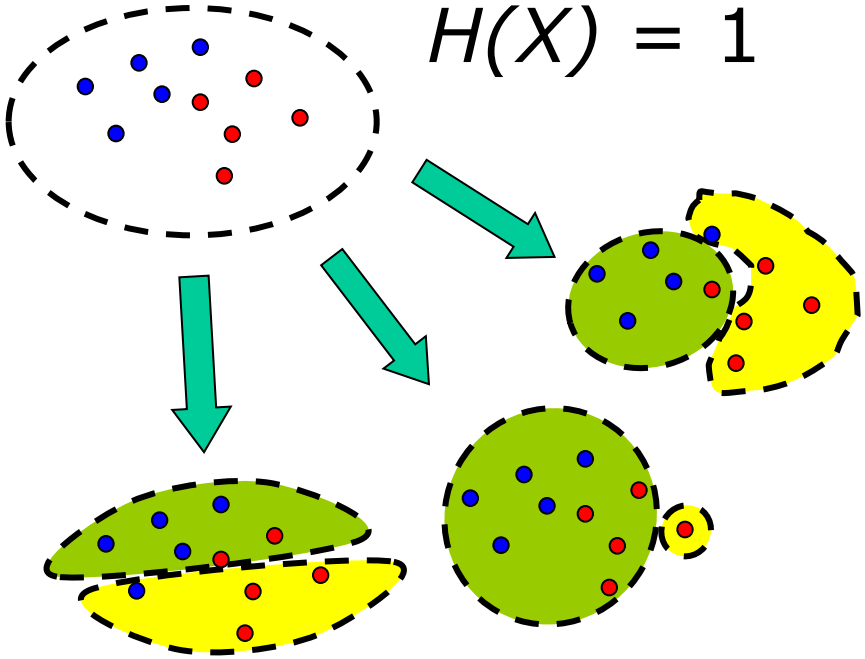
Only relative frequencies matter!

Information Gain

Given a set and a choice between possible subsets, which one is preferable?

Information Gain: Sets that minimize Entropy by largest amount

$$IG(X_A, X_B) = H(X) - p(x_A)H(X_A) - p(x_B)H(X_B)$$



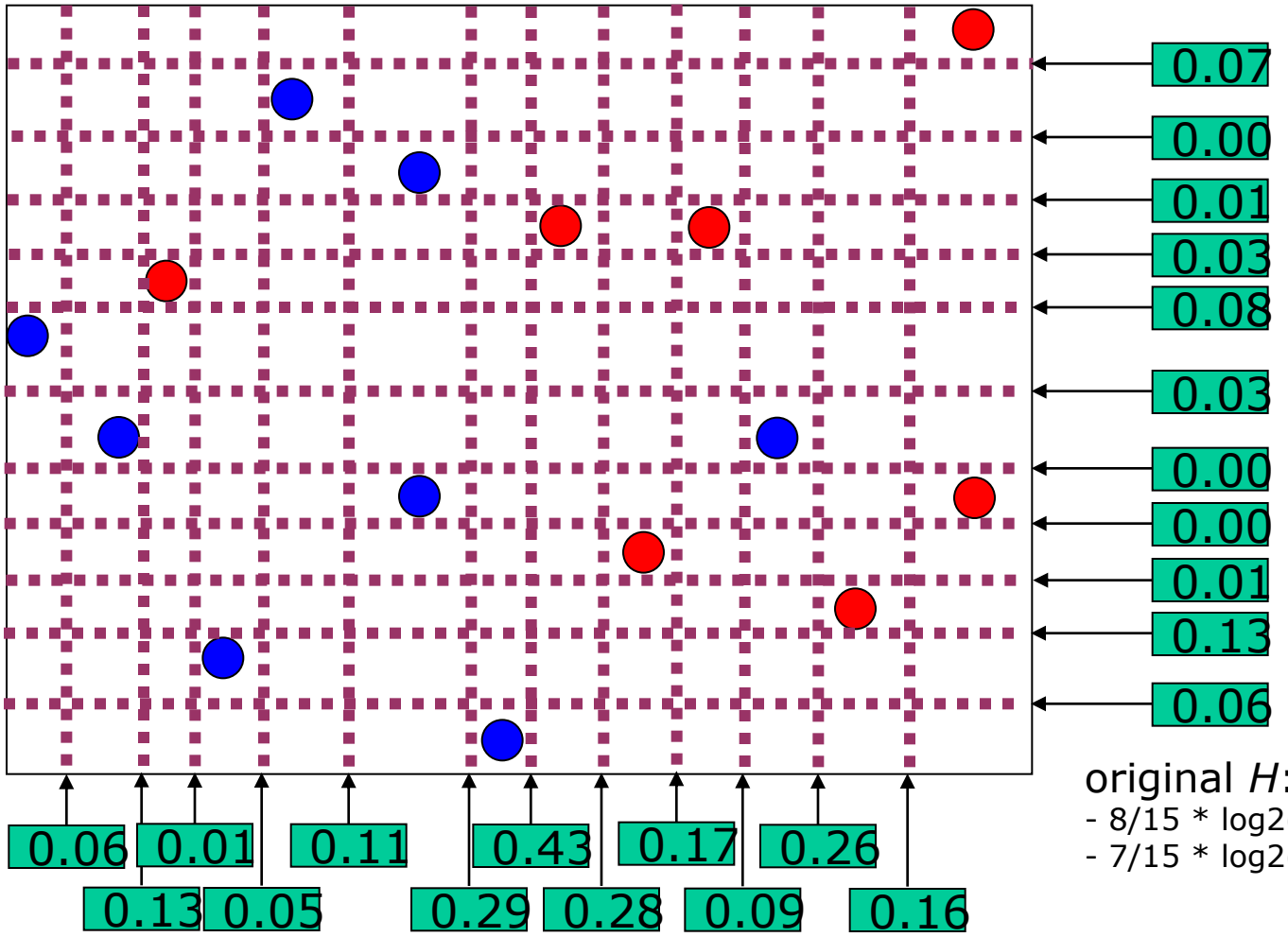
	A (green)	B (yellow)
Points	5	4
p(X.)	0.5	0.4
$p(x_{red})$	0.2	1.0
$p(x_{blue})$	0.8	0.25
H(X.)	0.72	0.81
IG	0.29 (1 - 0.5x0.72 - 0.5x0.89)	

Information Gain (Properties)

- Information Gain is
  - the amount by which the original Entropy can be reduced by splitting into subsets
  - *Min/max bounds of Information gain?*
  - at most as large as the Entropy of the undivided set
  - at least zero (if Entropy is not reduced)
- $0 \leq IG \leq H(X)$

Decision Trees: information gain

- 2-dimensional data (x, y), numerical values, two classes

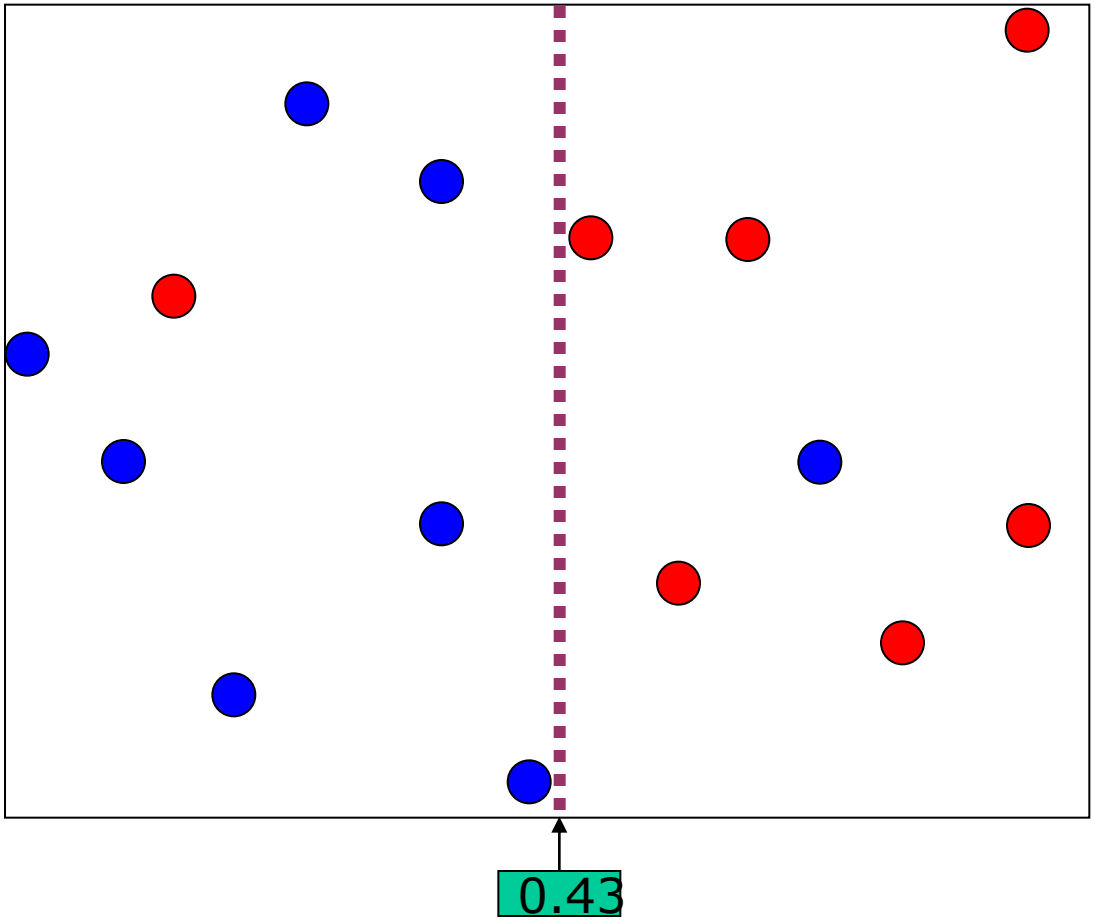


original *H*: 0.99679
- 8/15 * log2(8/15)
- 7/15 * log2(7/15)

Decision Trees: information gain

H(left) =
$-0.125\log_2 0.125 - 0.875\log_2 0.875 =$
$0.375 + 0.169 = 0.54356$

H(right) =
$-0.143\log_2 0.143 - 0.857\log_2 0.857 =$
$0.401+0.191 = 0.59167$



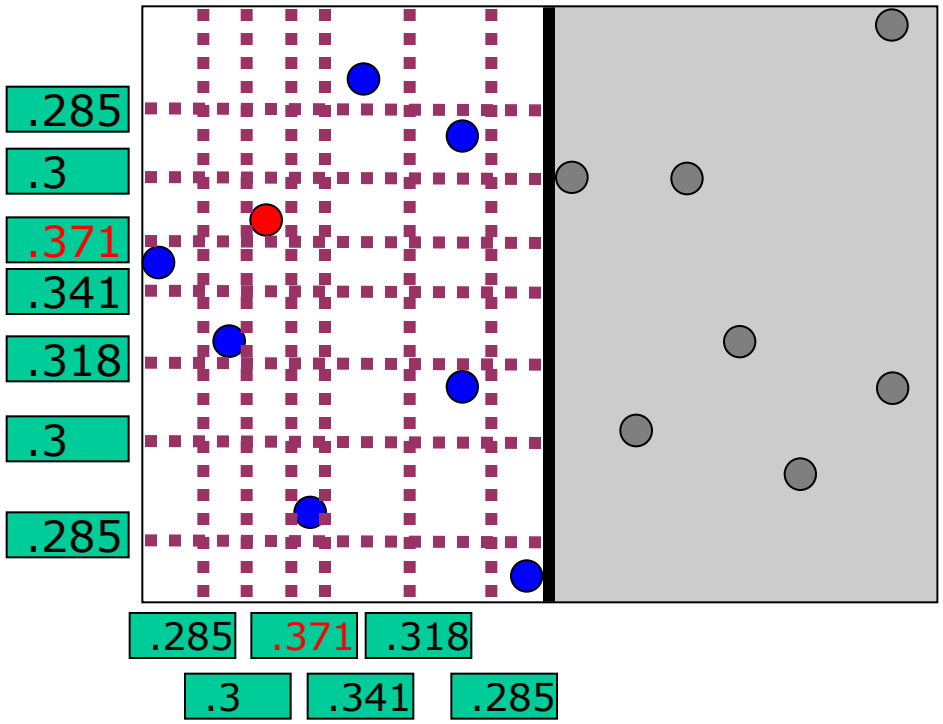
H(split) =
0.54356*8/15 +
0.59167*7/15 =
0.566011333

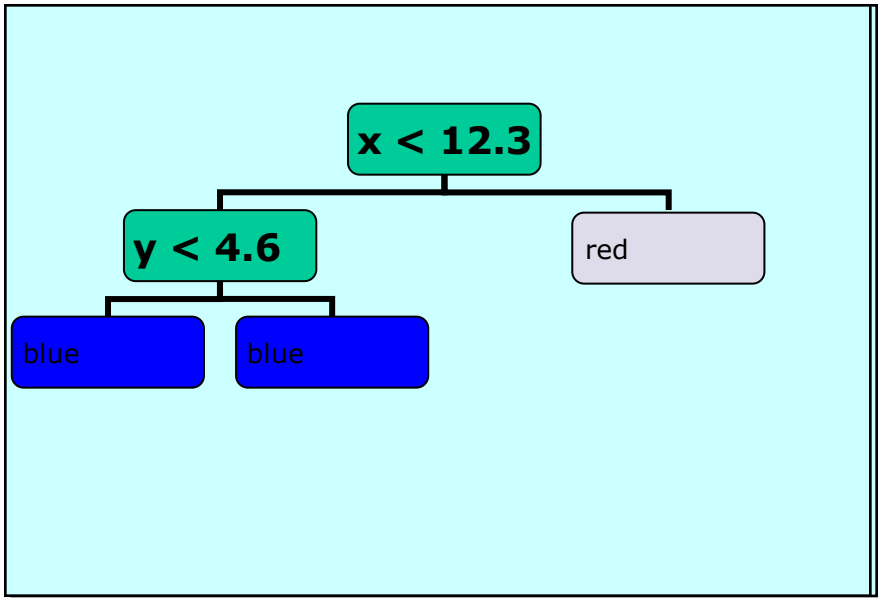
original *Entropy*:
H(x) = 0.99679

- 8/15 * log2(8/15)
- 7/15 * log2(7/15)

IG = 0.43078

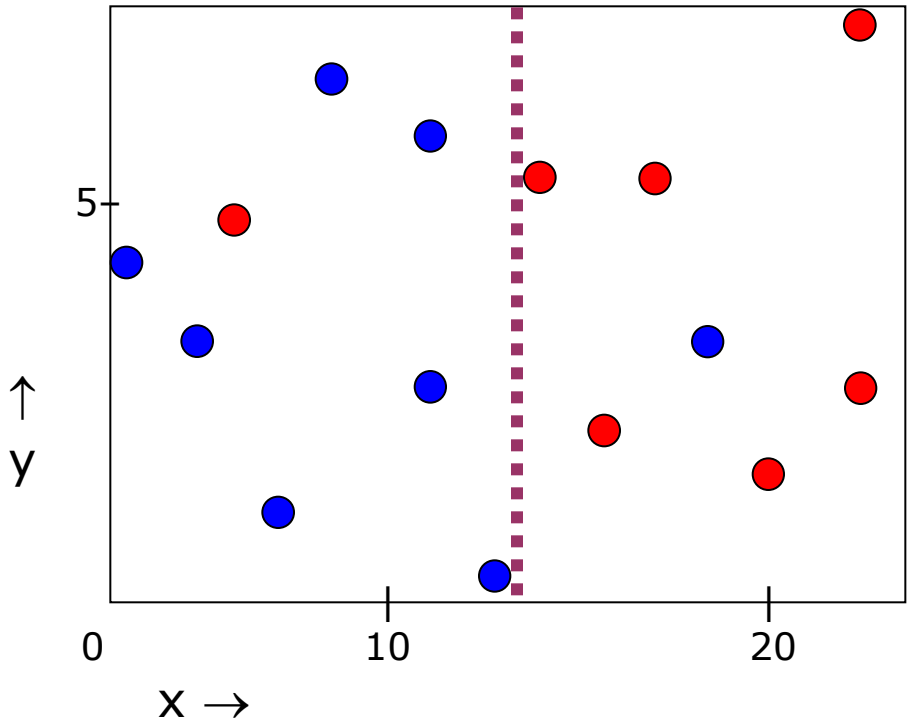
Decision Trees: information gain

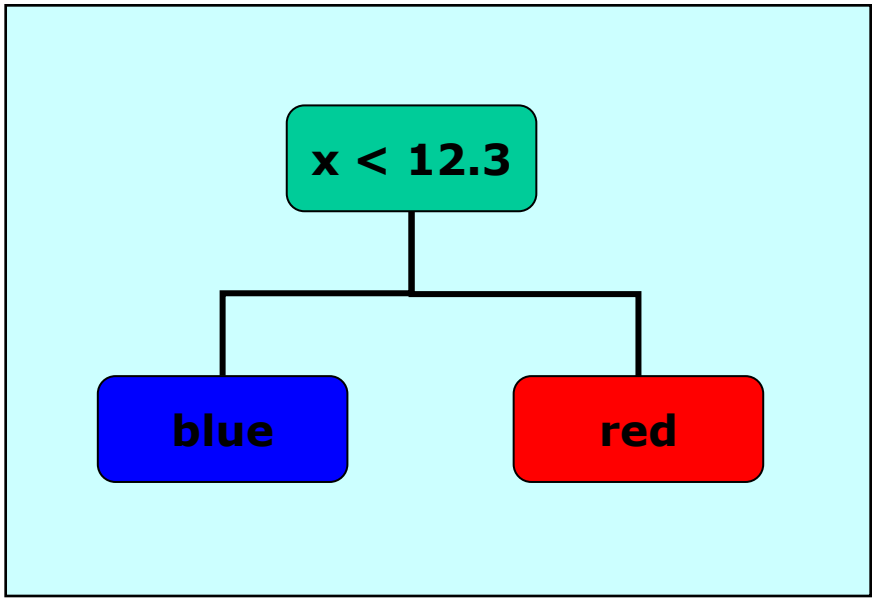




Decision Trees: information gain

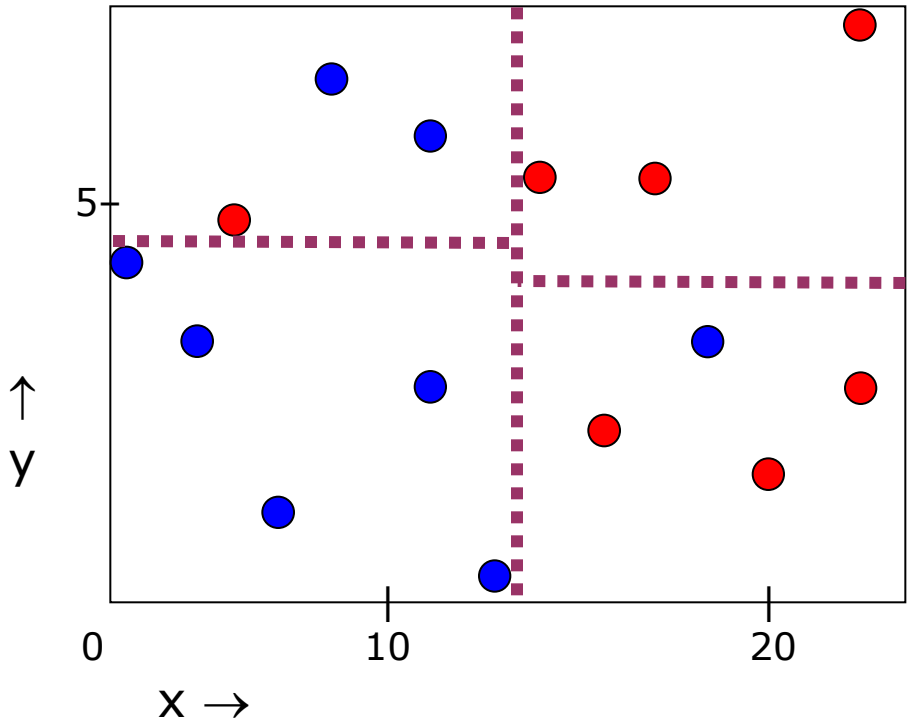
- Tree training, level 1

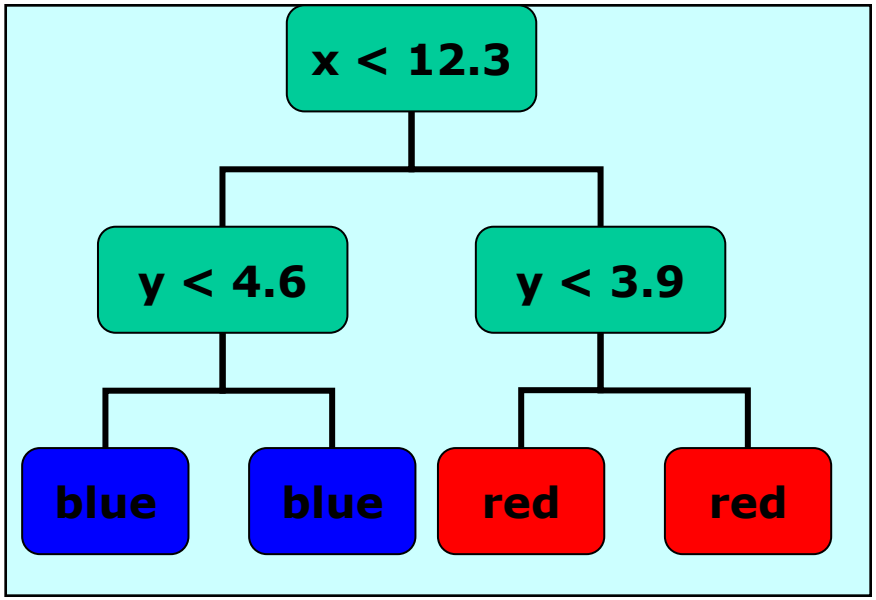




Decision Trees: information gain

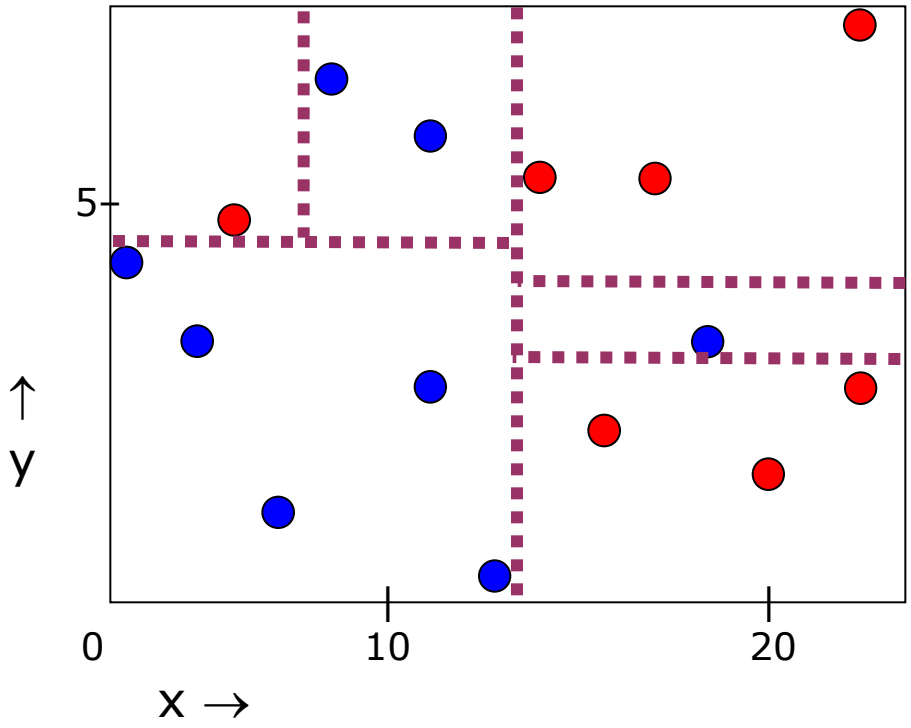
- Tree training, level 2

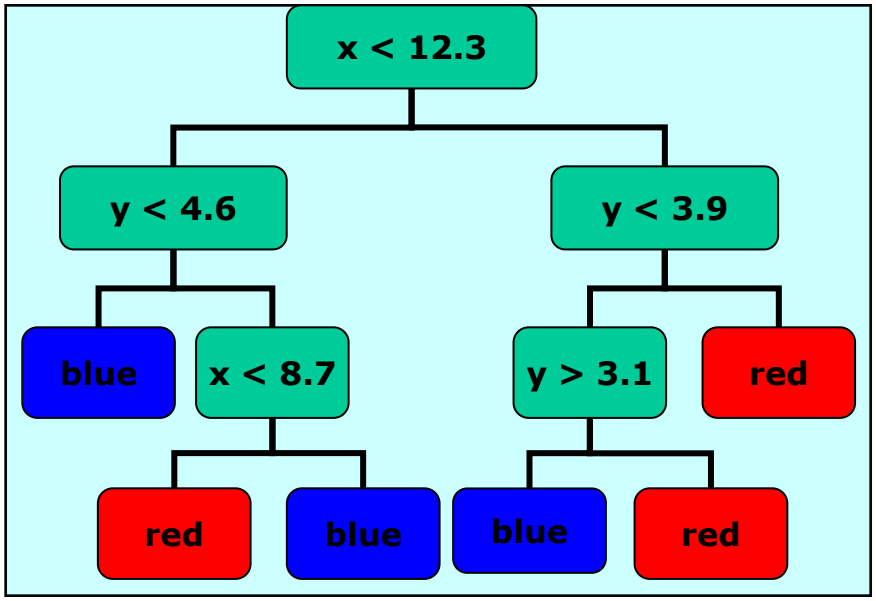




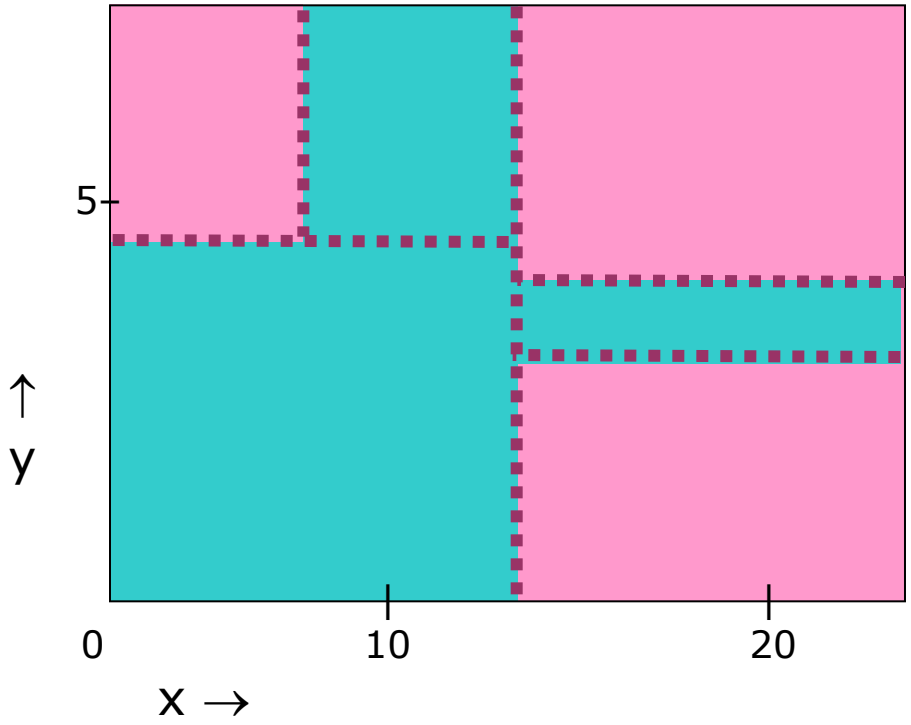
Decision Trees: information gain

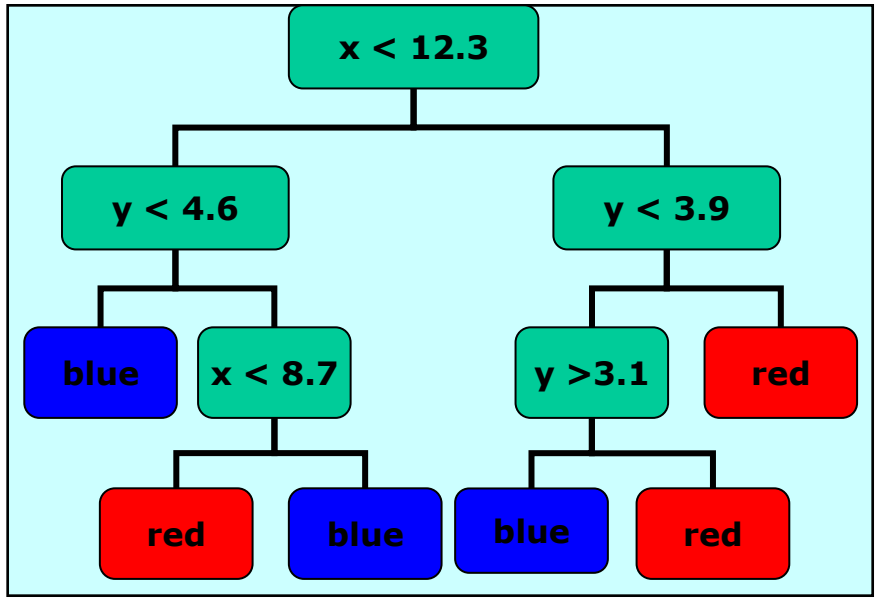
- Tree training, level 3: completely built tree





Decision Trees: Classification

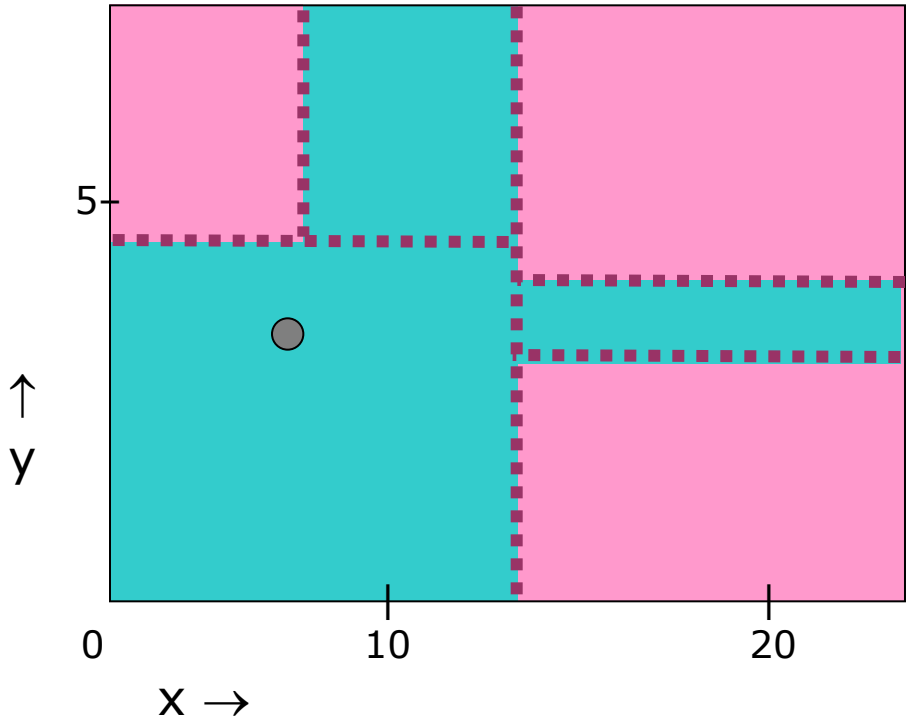


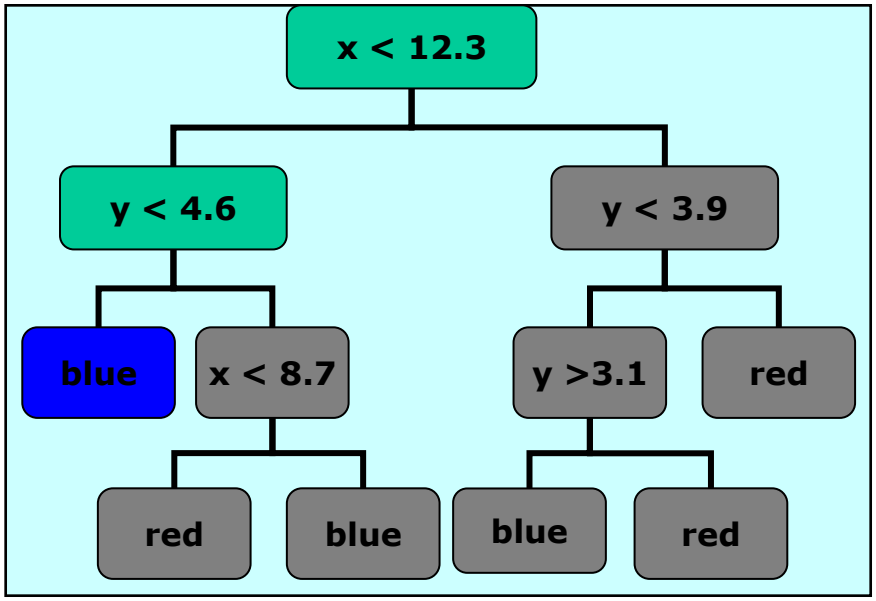


How to classify unknown items?

Same as for a tree built error rate!

Decision Trees: Classification

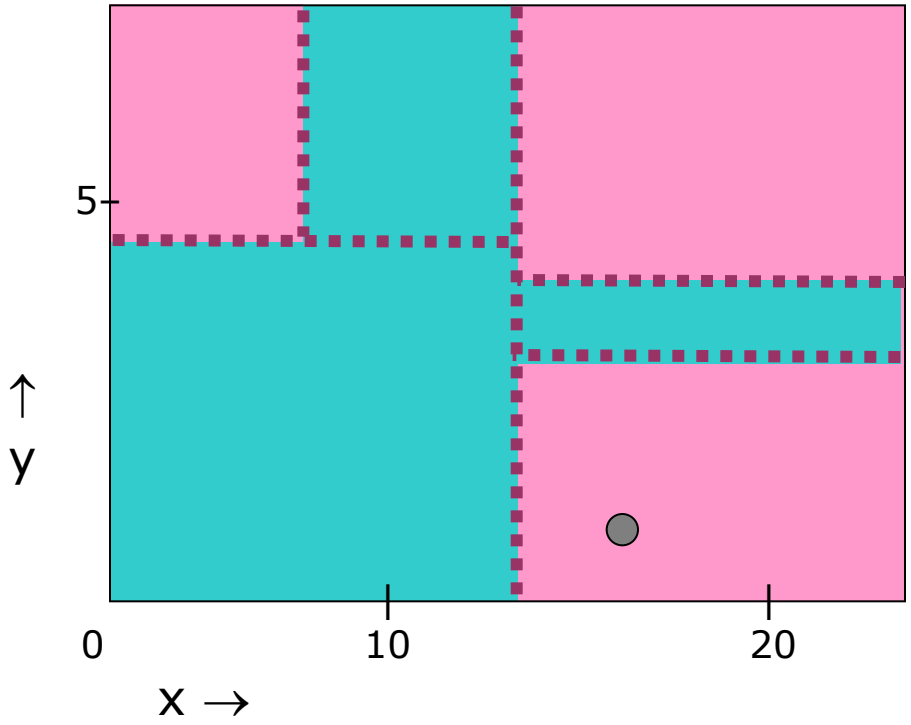


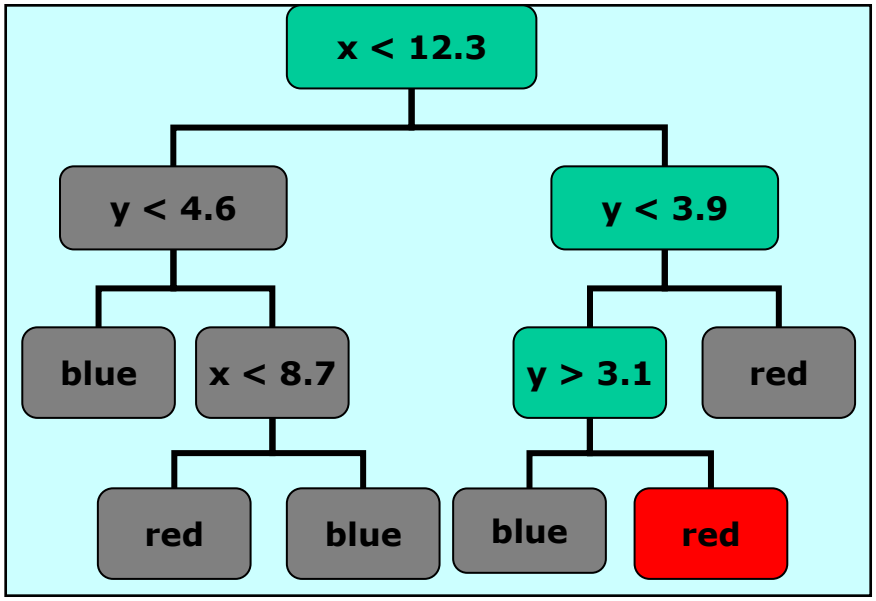


➔ Decend the tree until leaf-node

➔ Use majority of class in that leaf node

Decision Trees: Classification

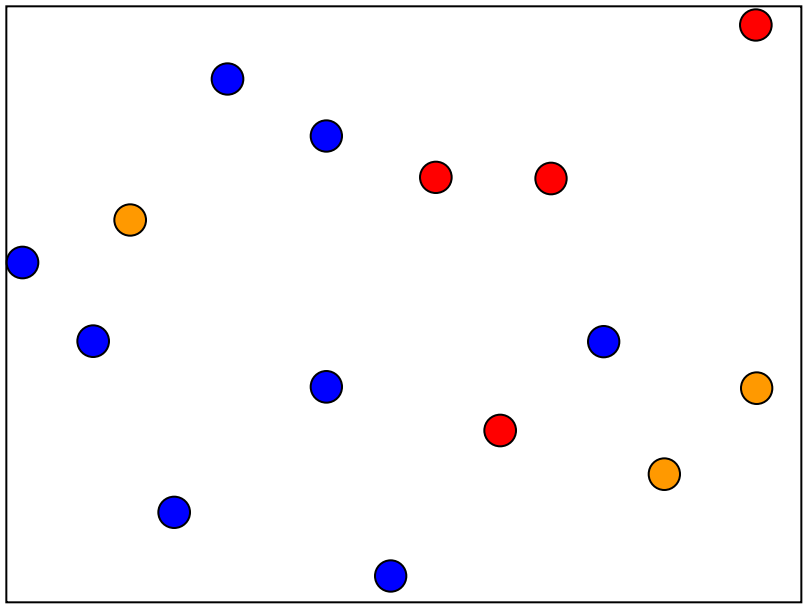




➔ Decend the tree until leaf-node

➔ Use majority of class in that leaf node

Decision Trees: More than 2 classes



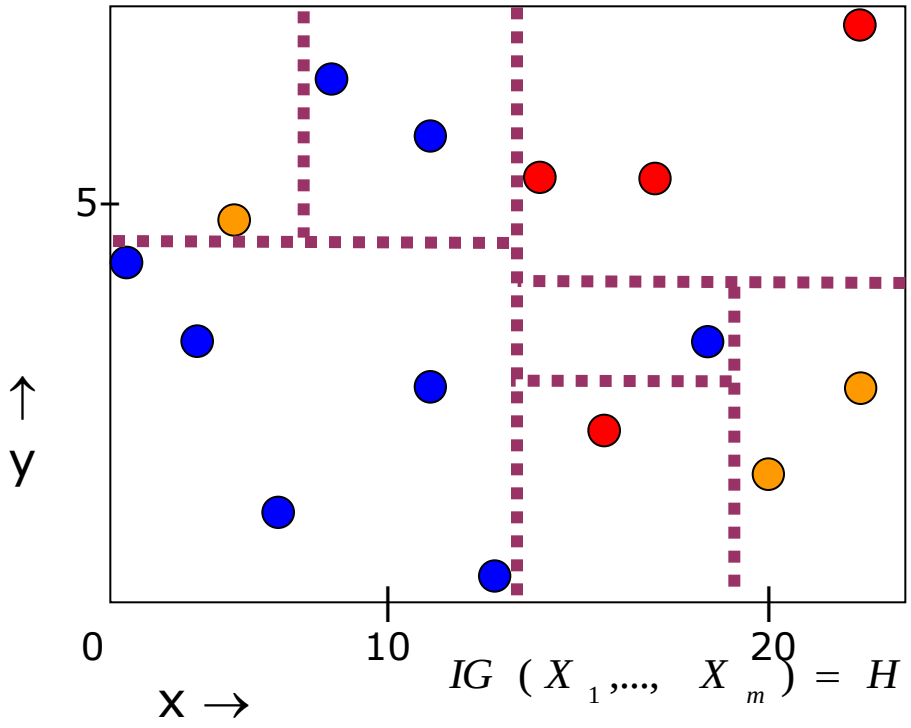
- *Conceptual changes?*

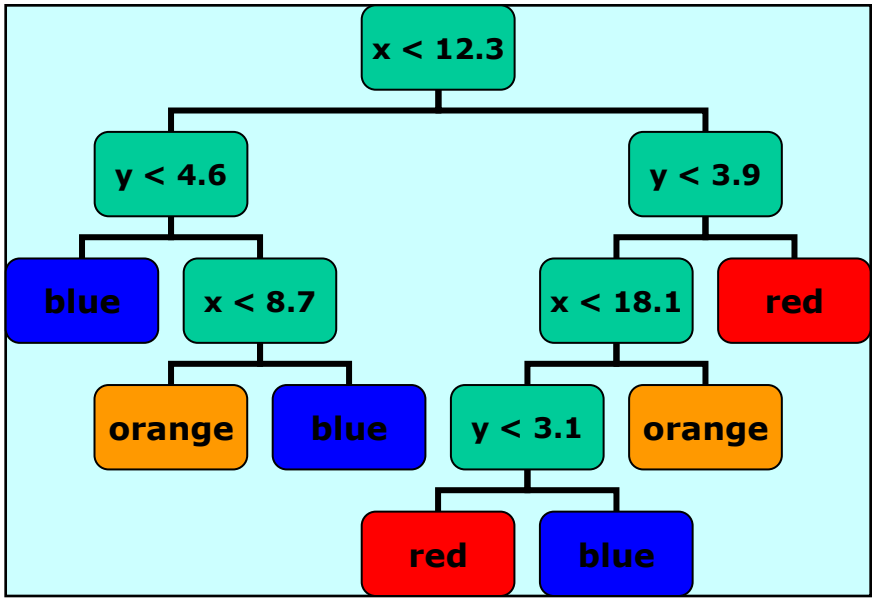
$$IG(X_1,\ldots,X_m) = H(X) - \sum_{j=1}^{m} p(x_j)\,H(X_j)$$

$$H(X) = -\sum_{i=1}^{n} p(x_i)\log_2 p(x_i)$$

$$H(X) = -p(x_{red})\log_2 p(x_{red}) - p(x_{blue})\log_2 p(x_{blue}) - p(x_{yellow})\log_2 p(x_{yellow})$$

Decision Trees: More than 2 classes

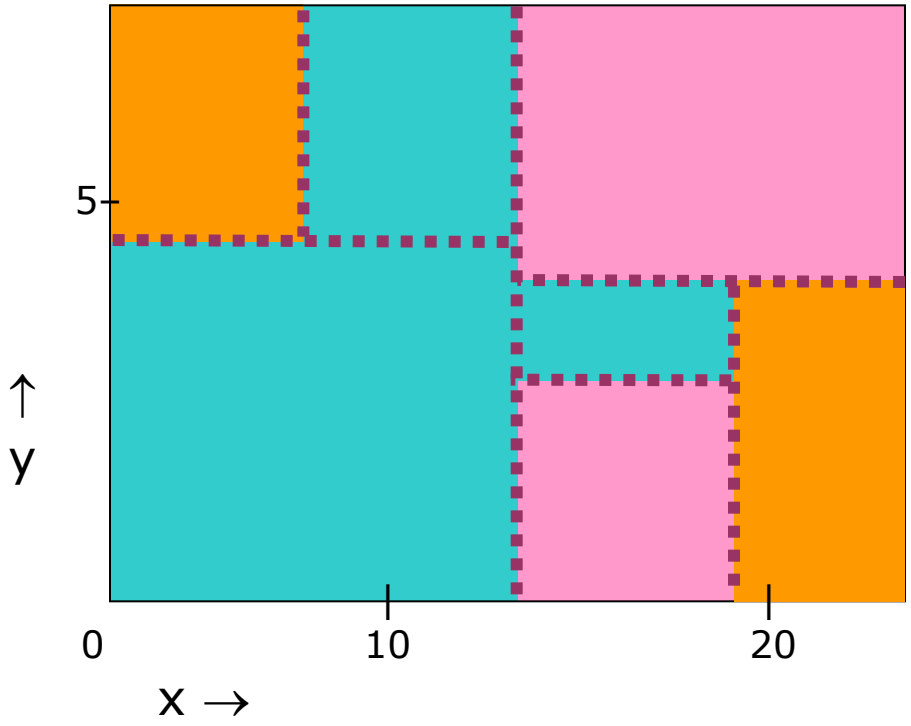


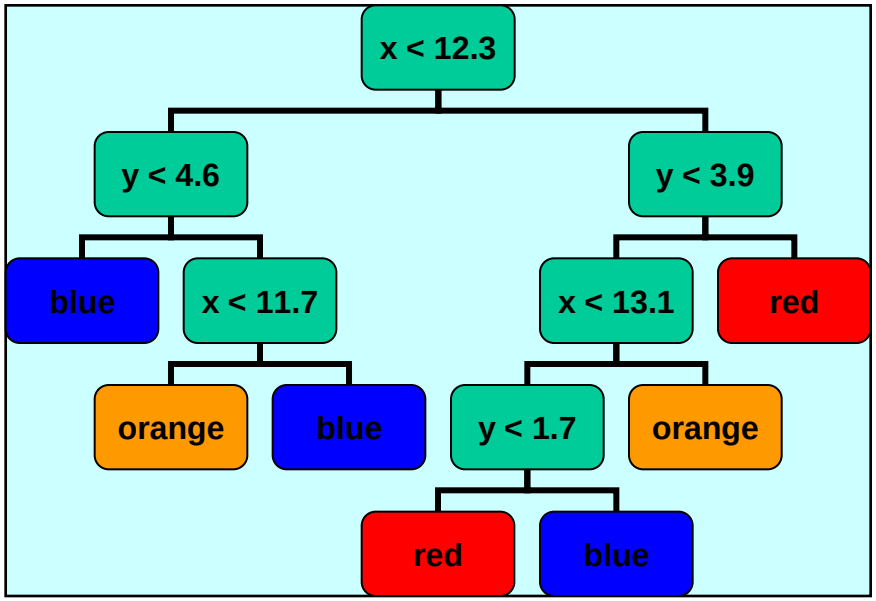


$$IG(X_1,\ldots,X_m) = H(X) - \sum_{j=1}^{n} p(x_j)\,H(X_j)$$

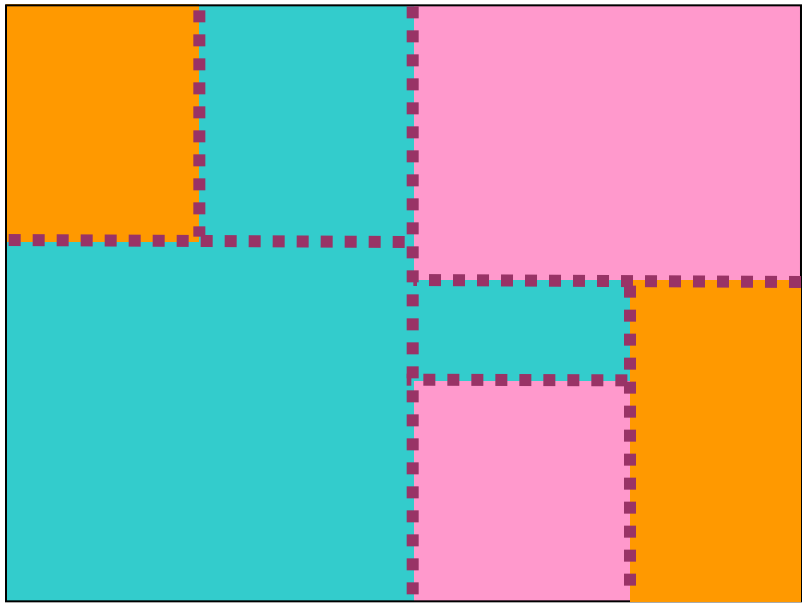
$$H(X) = -\sum_{i=1} p(x_i)\log_2 p(x_i)$$

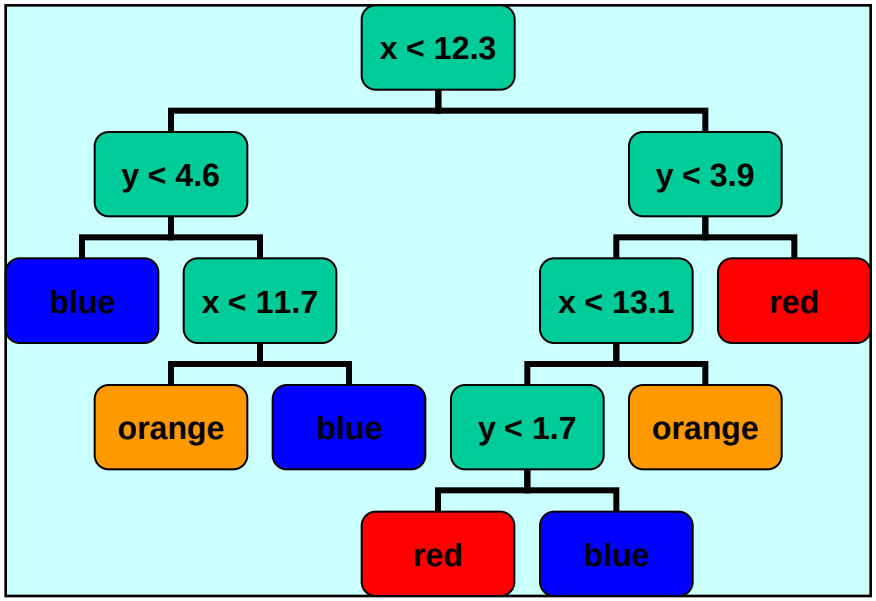
Decision Trees: More than 2 classes





Decision Trees: More than 2 classes



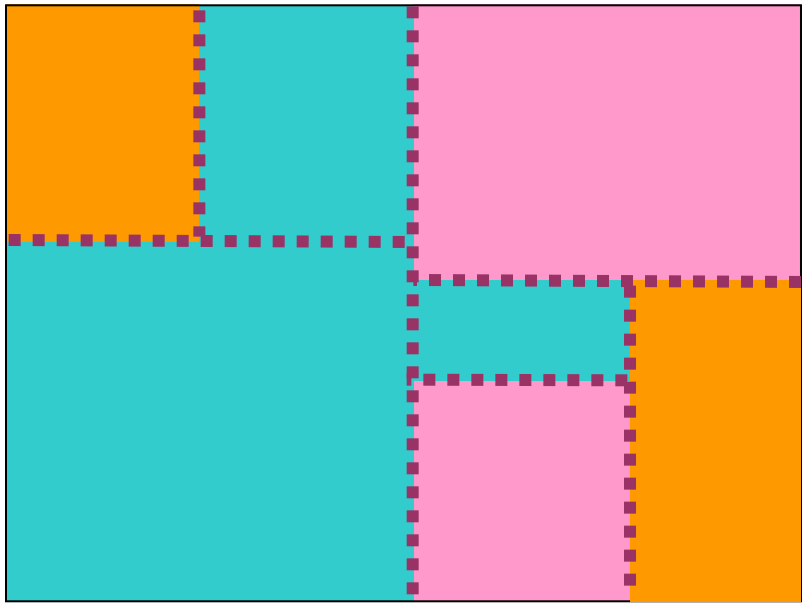


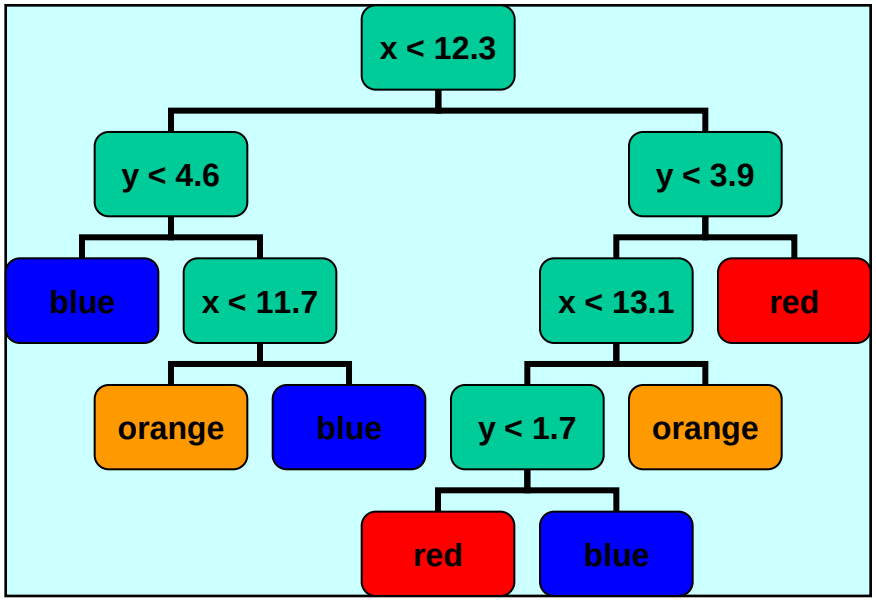
$$IG(X_1,\ldots,X_m) = H(X) - \sum_{j=1}^{n} p(x_j)\,H(X_j)$$

$$H(X) = -\sum_{i=1}^{m} p(x_i)\log_2 p(x_i)$$

Maximum value of Entropy?

Decision Trees: More than 2 classes





$$IG(X_1,\ldots,X_m) = H(X) - \sum_{j=1}^{n} p(x_j)\,H(X_j)$$

$$H(X) = -\sum_{i=1} p(x_i)\log_2 p(x_i)$$

$$H(X) = -p(x_{red})\log_2 p(x_{red}) - p(x_{blue})\log_2 p(x_{blue}) - p(x_{yellow})\log_2 p(x_{yellow})$$

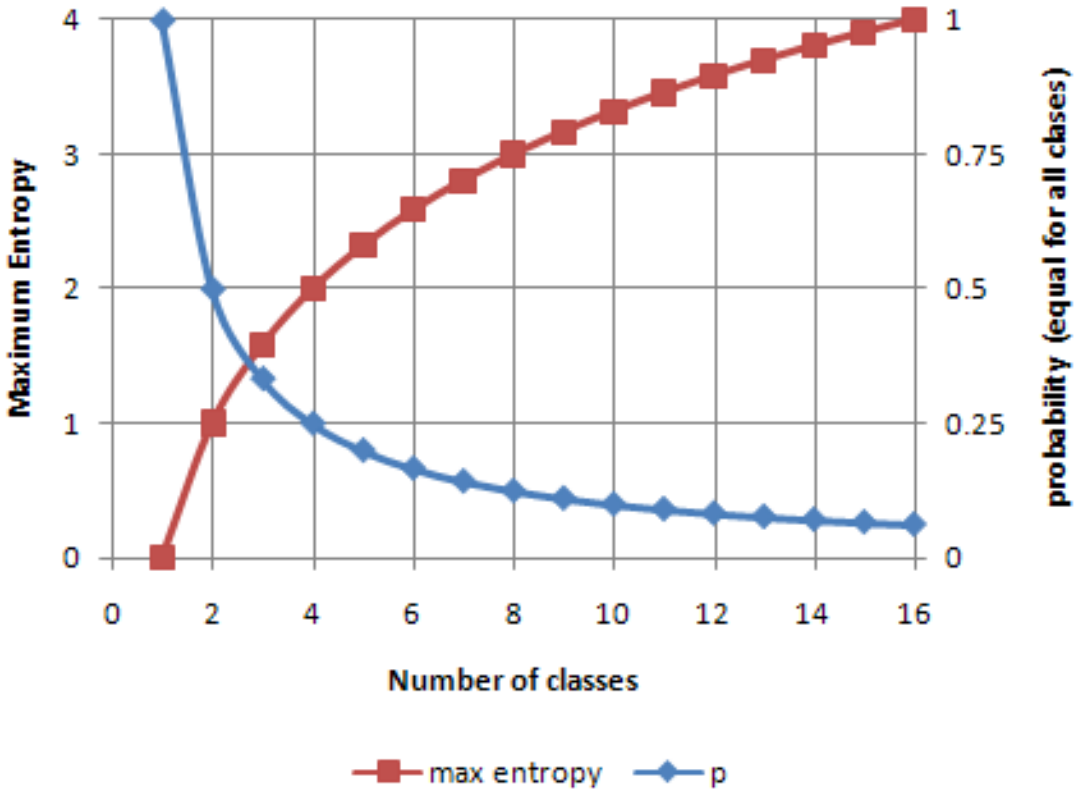
Maximum Entropy? $H(X) = \frac{1}{3}\times\log_2(\frac{1}{3}) - \frac{1}{3}\times\log_2(\frac{1}{3}) - \frac{1}{3}\times\log_2(\frac{1}{3}) = 1.5849$

Decision Trees: More than 2 classes

$$IG(X_1,\ldots,X_m) = H(X) - \sum_{j=1}^{m} p(x_j) H(X_j)$$

$$H(X) = -\sum_{i=1}^{n} p(x_i) \log_2 p(x_i)$$

Maximum Entropy?



Decision Trees: Algorithm

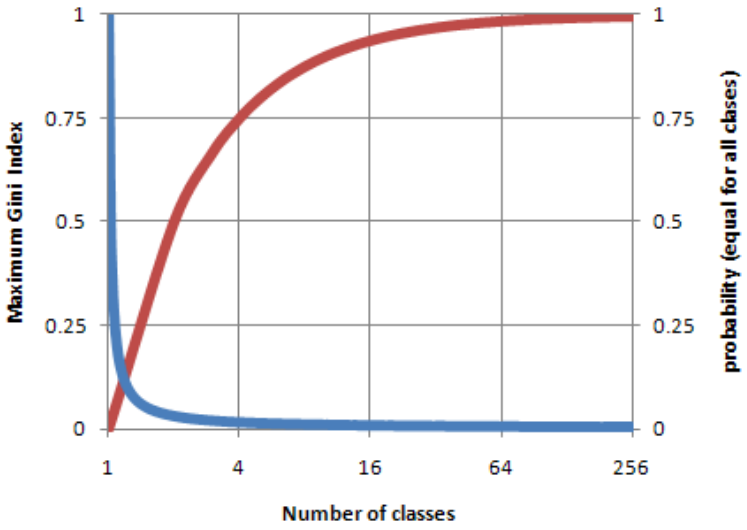
- Popular measures to compute best split
  - Error rate
  - Information gain
  - *Gini impurity (Gini index)*

Gini impurity (Gini index)

- Inequality among values of a distribution
  - Developed initial for income levels
  - How often a randomly chosen element from the set would be incorrectly labeled, if it was randomly labeled according to the distribution of labels in the subset

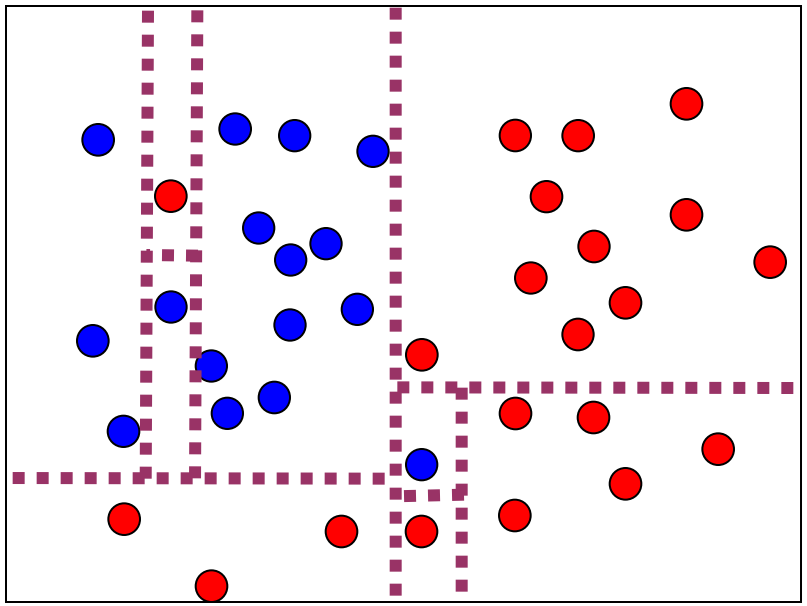
$$I_G(p) = \sum_{i=1}^{|C|} p_i(1 - p_i) = \sum_{i=1}^{|C|} (p_i - p_i^2) = \sum_{i=1}^{|C|} p_i - \sum_{i=1}^{|C|} p_i^2 = 1 - \sum_{i=1}^{|C|} p_i^2$$

- Value range?
  - Between 0 (only one class) and 1 (total inequality)

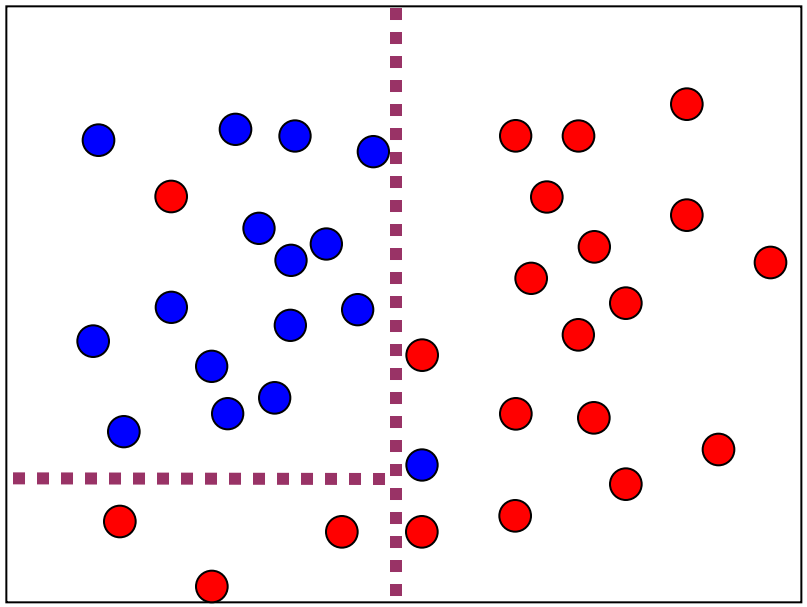


Decision Trees: Overfitting

- Fully grown trees are usually too complicated
  - *Why is that an issue?*
    - Generalisation
    - Understanding
  - Especially useful when there is noisy/"useless" data



Fully grown



Simplified

Decision Trees: Stopping Criteria

- *How to achieve simplified trees?*
  - Avoid fully growing trees! *How?*

    ➔ Alternative stopping criteria

  - Stop splitting a node when
    - ~~*Data in each node from only one class*~~
    - Absolute number of samples is low (< threshold)
    - Entropy is already relatively low (< threshold)
    - Information Gain is low (< threshold)
    - Depth of tree has reached a max value (> threshold)

  - Threshold values depend on data set

Decision Trees: Pruning

- *How to achieve simplified trees?*
  - "Cut back" complicated trees!

- "Pruning" means removing nodes from a tree after training has finished
  - Stopping criteria are sometimes referred to as "pre-pruning"

- Nodes/branches with no/little power to classify are removed
  - Reduces complexity of tree

Decision Trees: Pruning

- Simple bottom-up approach: reduced error pruning

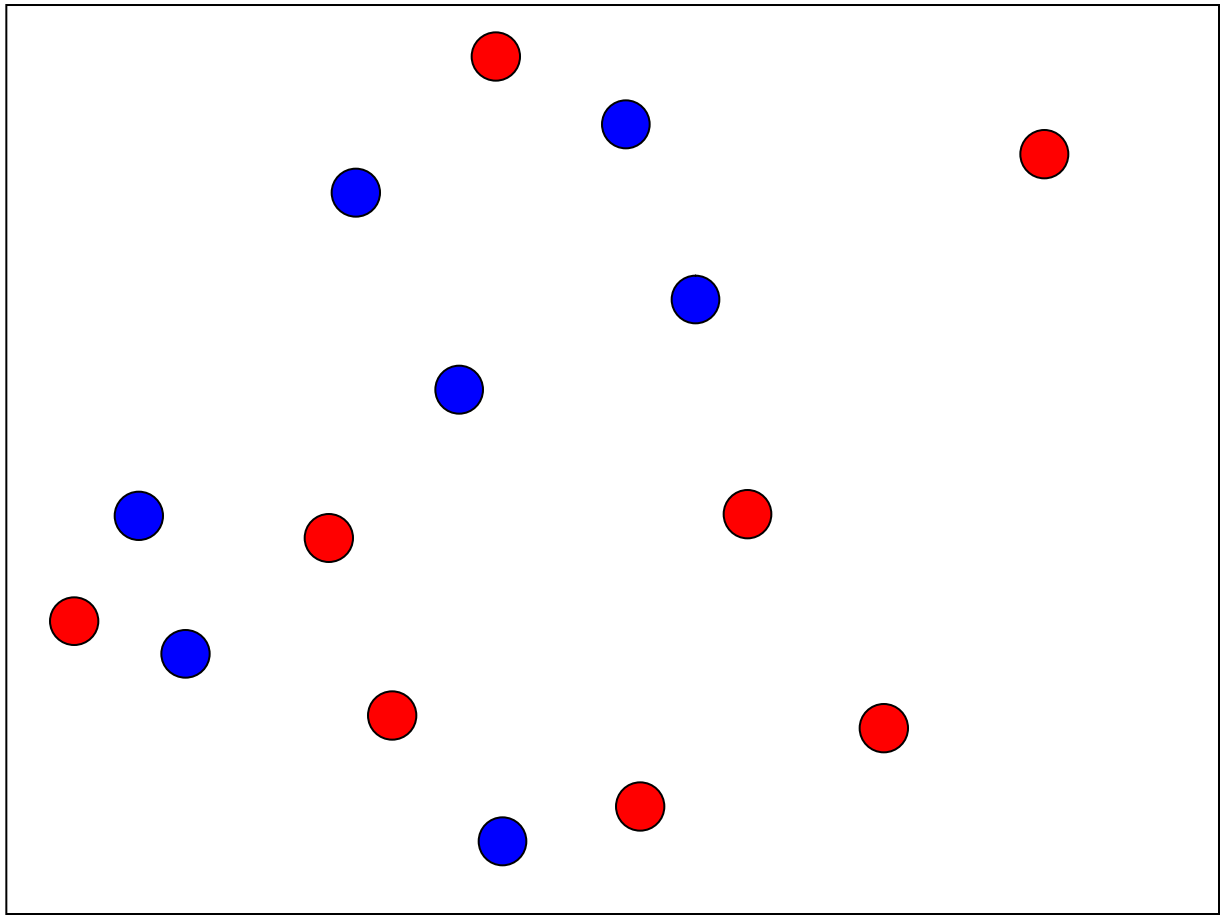
  1. Starting from leaves, remove a node from the tree
     1. Replace it with the majority class
  2. Evaluate the performance without the pruned node
  3. Repeat steps 1 & 2
     - Until no improvement is obtained from pruning

- Other approaches, e.g. cost complexity pruning (bottom up), Pessimistic Error Pruning (top-down) …
  - Generate a list of candidate trees T_0, T_1, … T_n
  - Take the best performing tree as the final decision tree

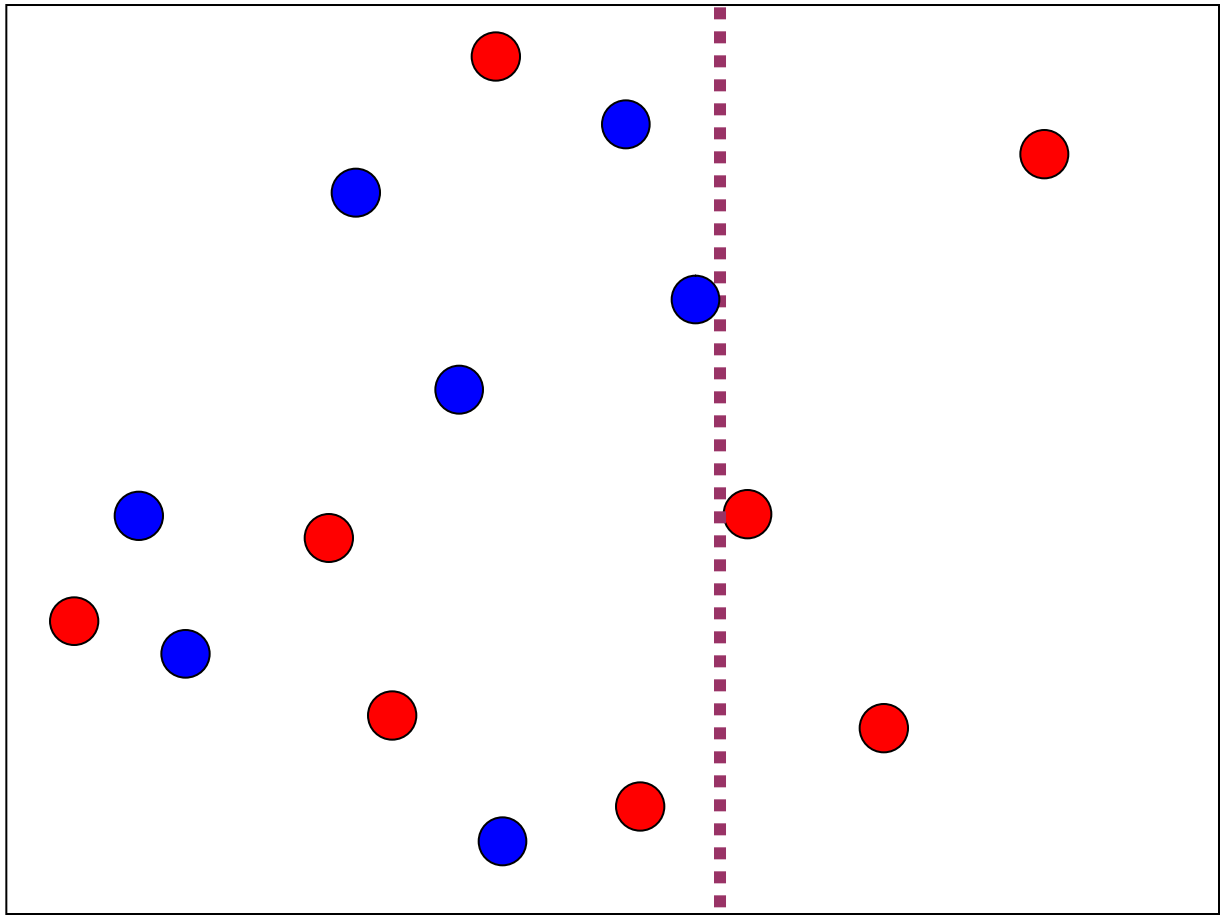
Decision Trees: Overfitting

- *What's the difference between pruning & pre-pruning?*
  - Effectiveness of final tree
    - Might be better for pruned trees
    - Or simply more fitted to training data?
  - Effort for pruning algorithm (runtime)

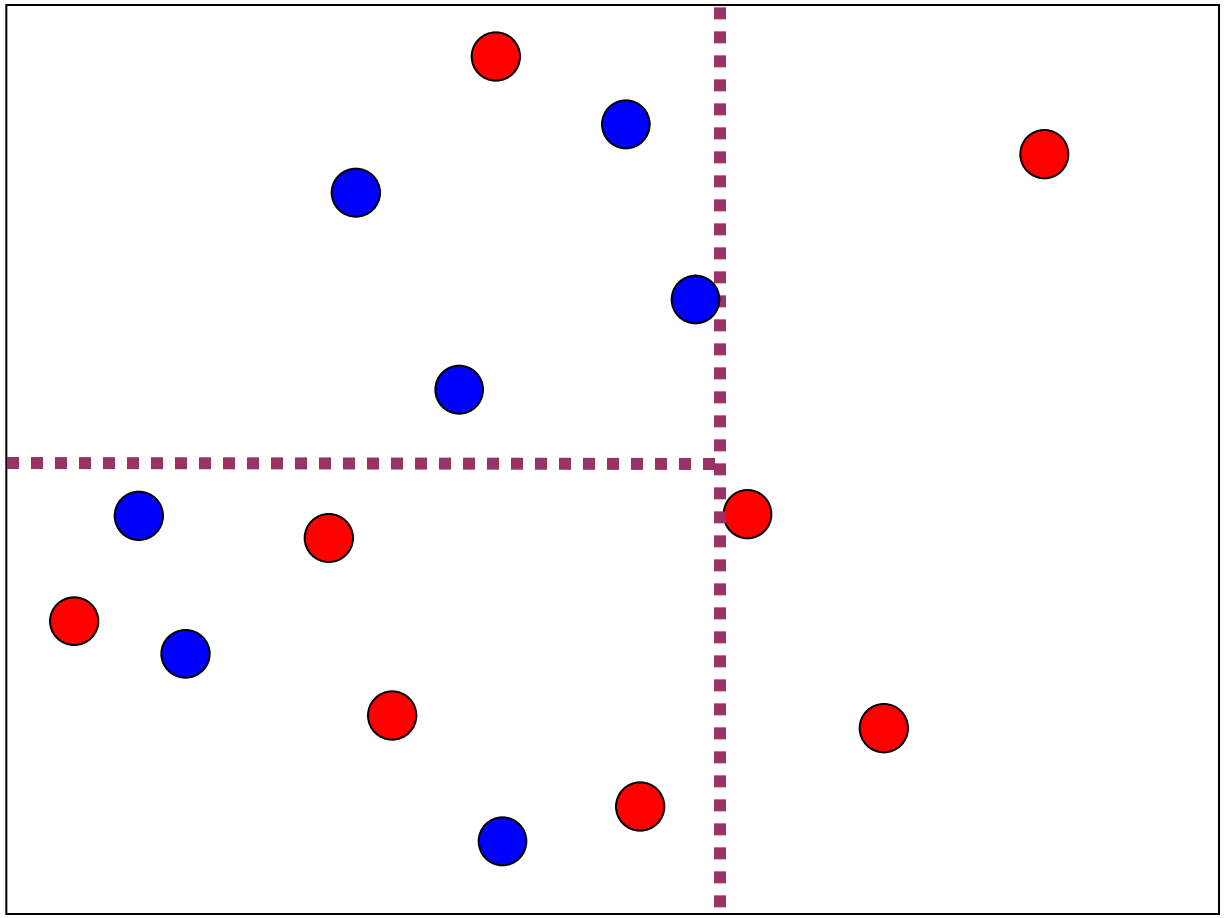
Decision Trees: Stability



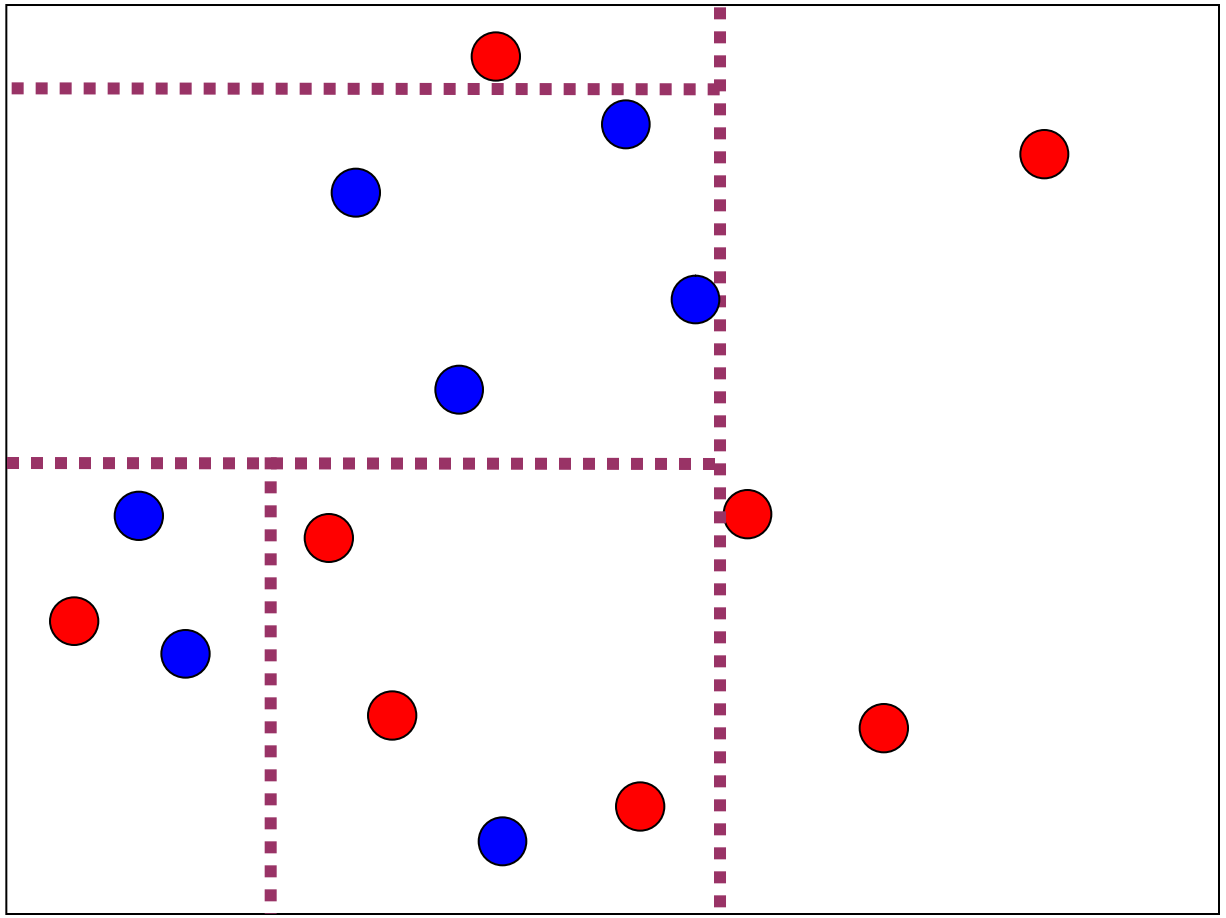
Decision Trees: Stability



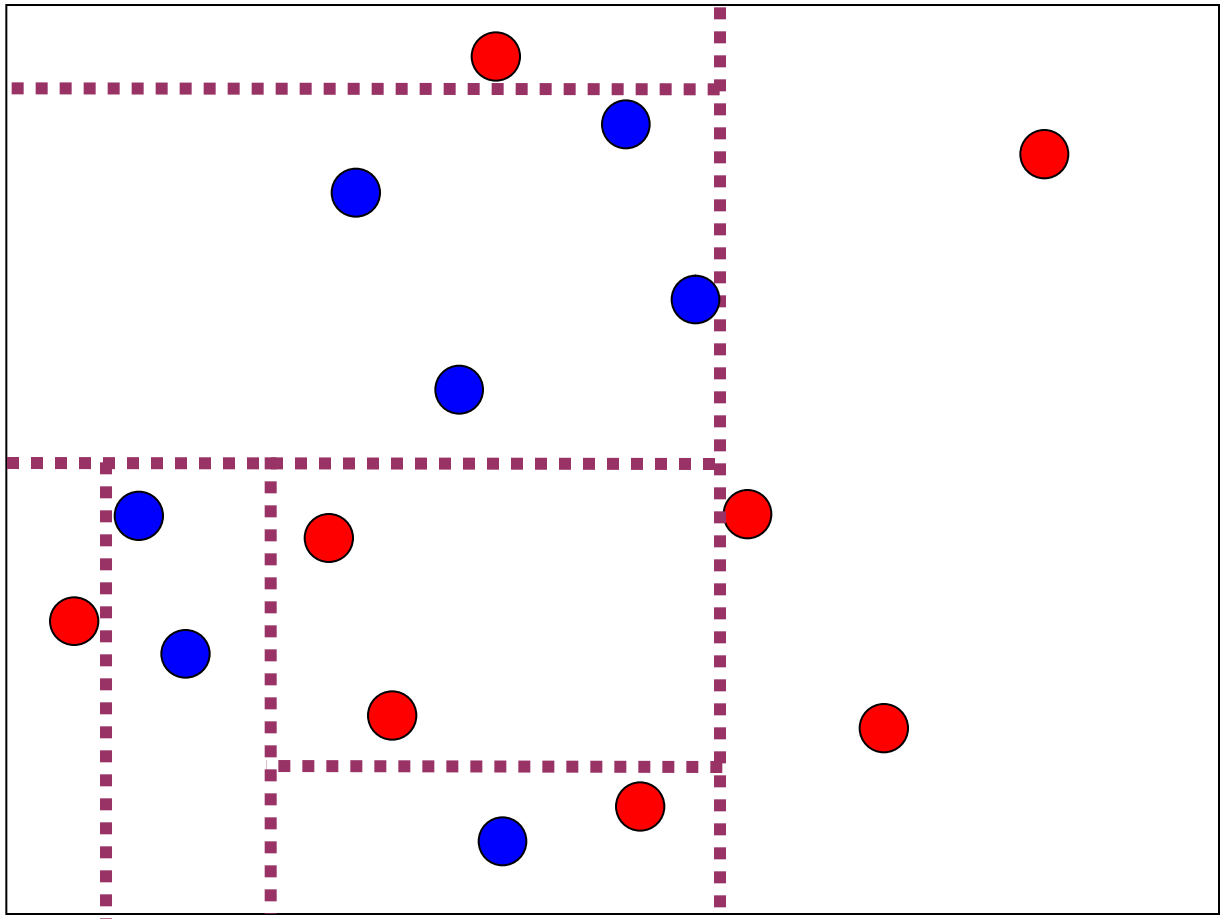
Decision Trees: Stability



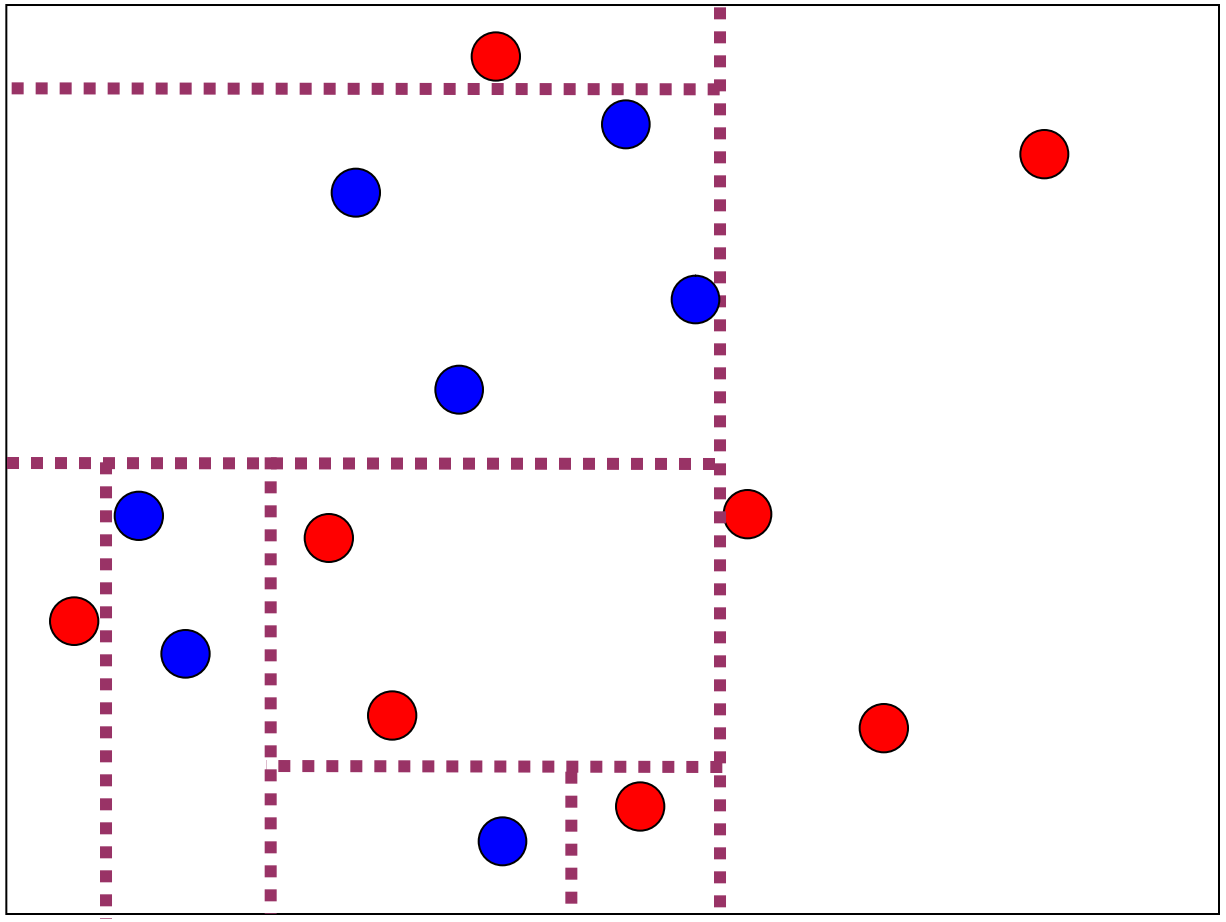
Decision Trees: Stability



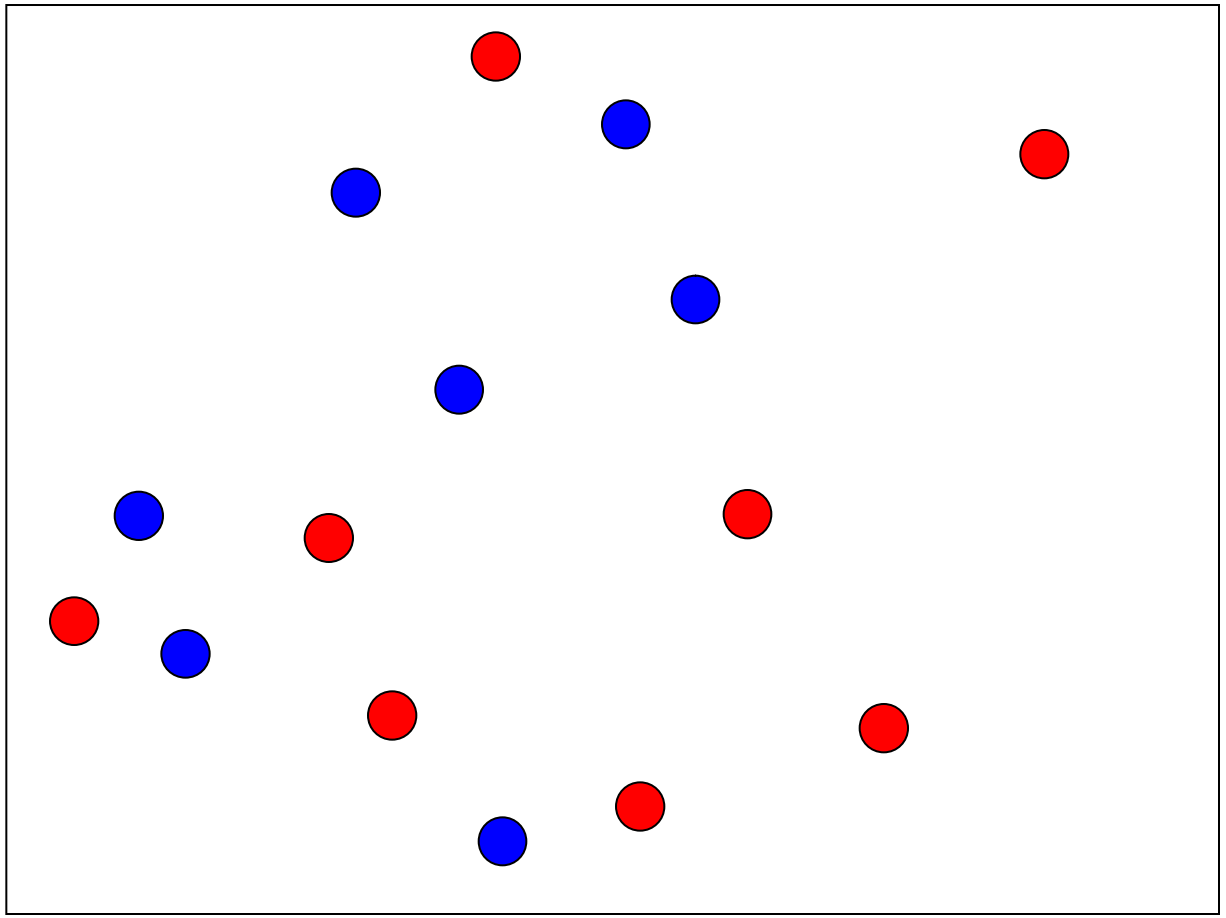
Decision Trees: Stability



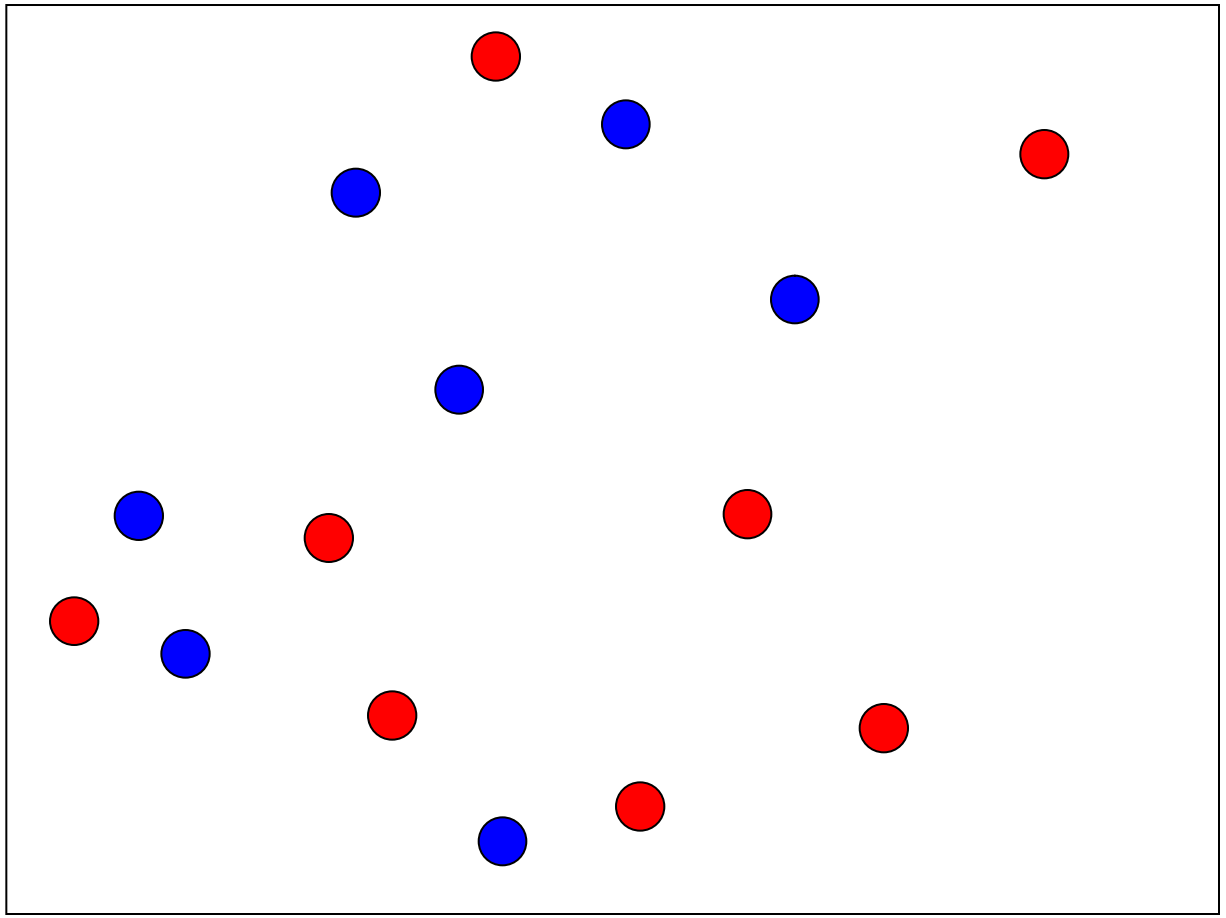
Decision Trees: Stability



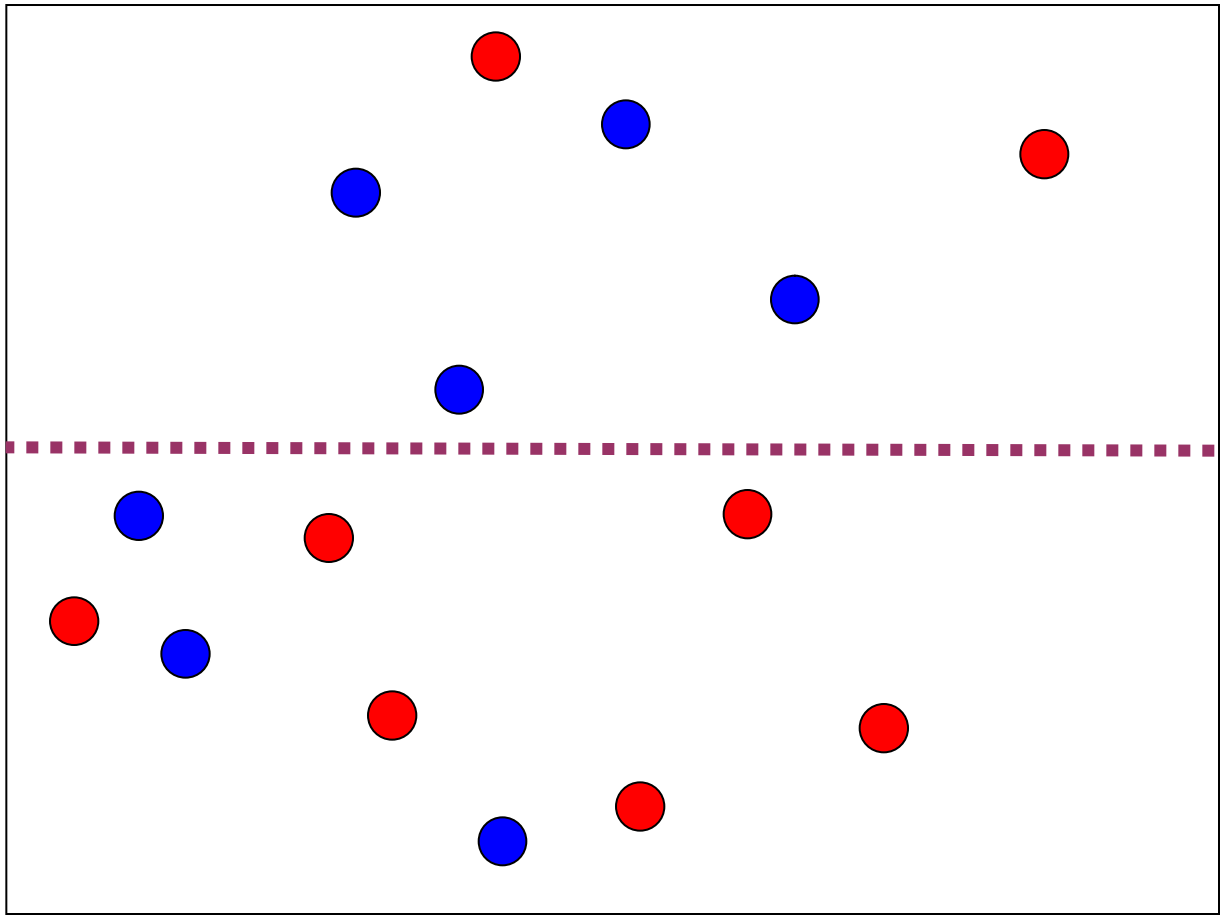
Decision Trees: Stability



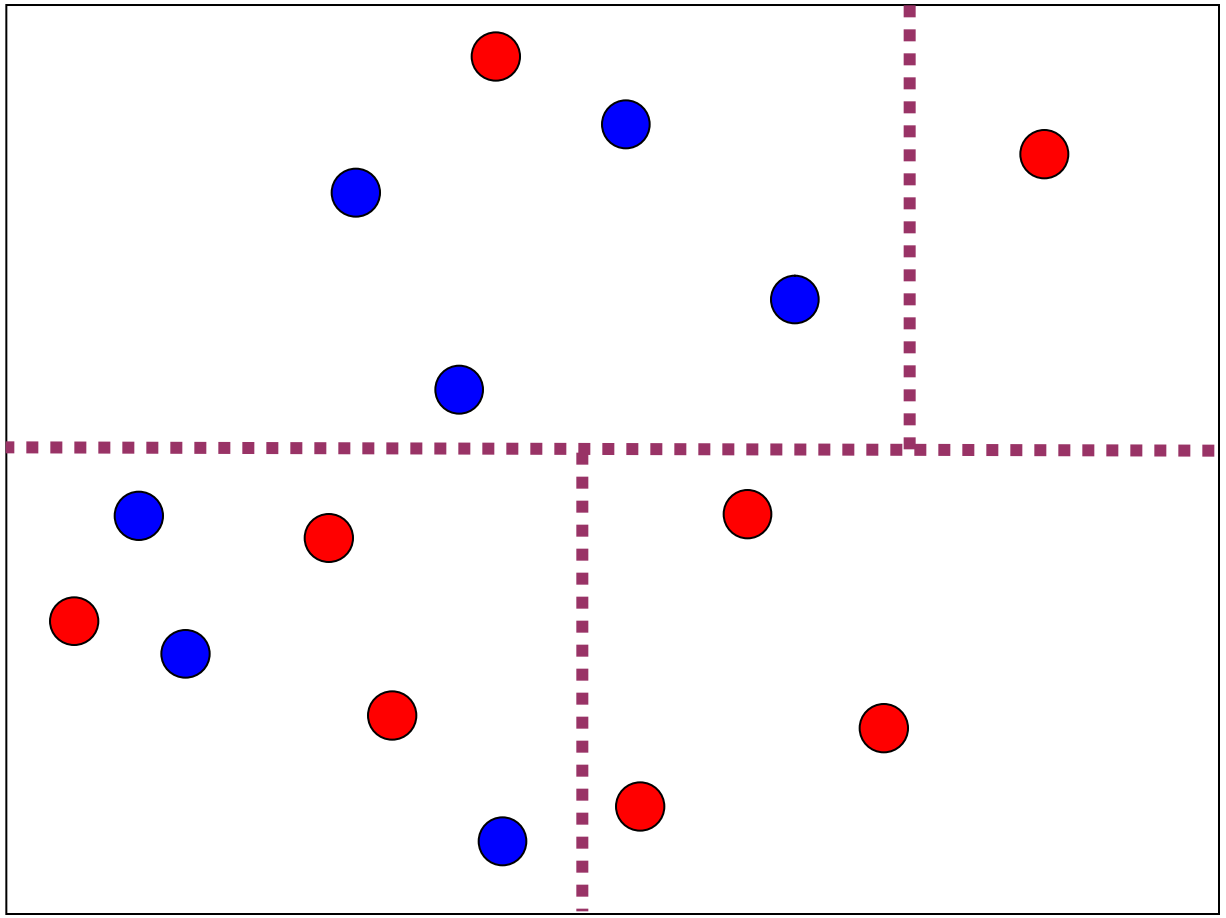
Decision Trees: Stability



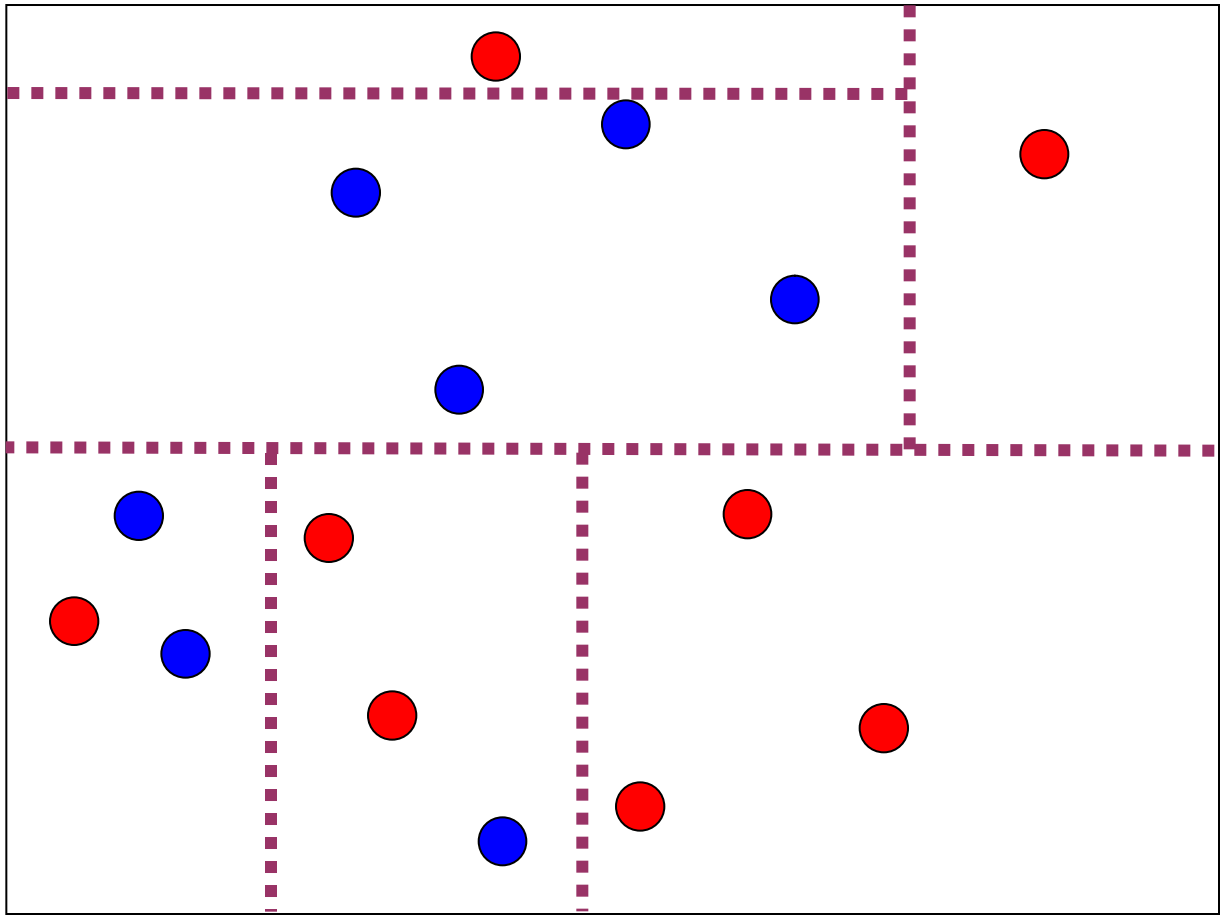
Decision Trees: Stability



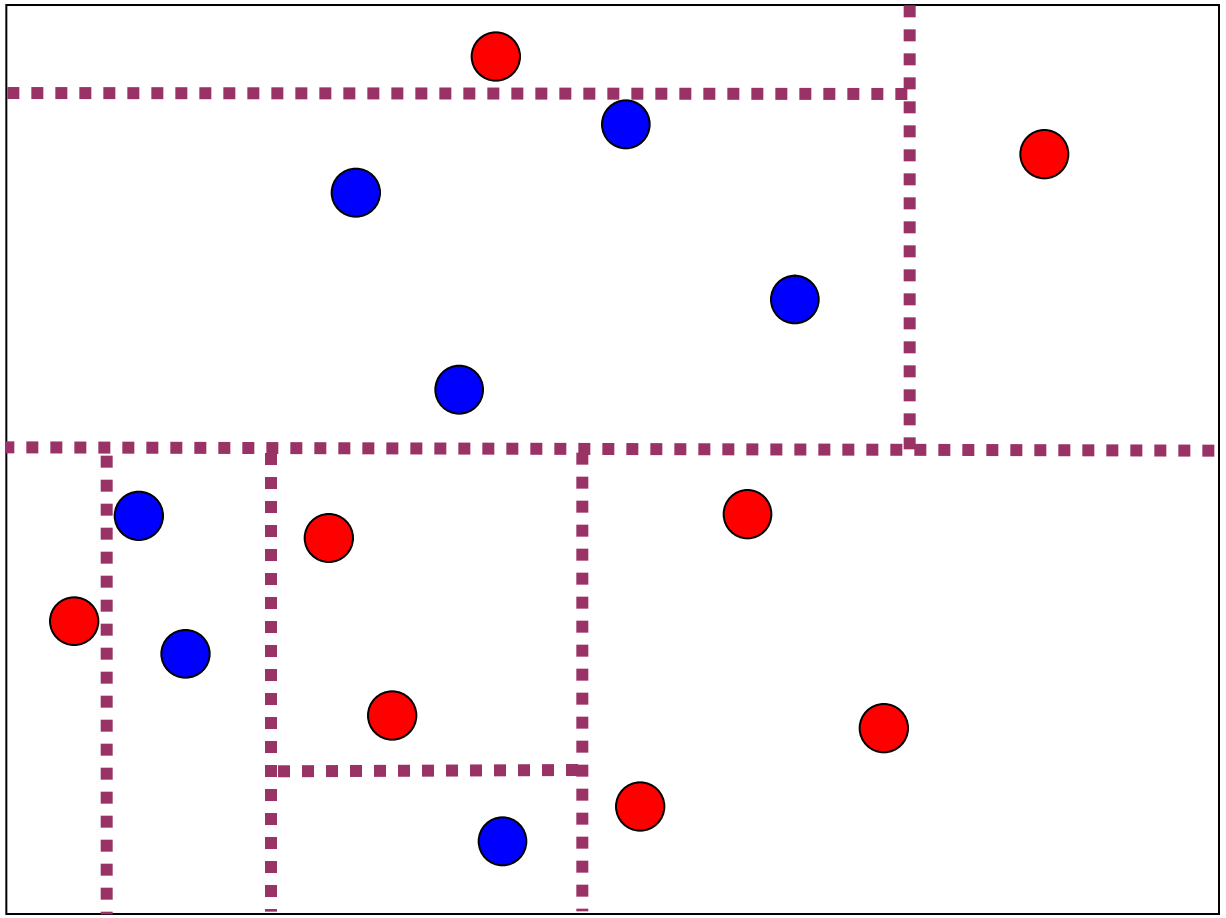
Decision Trees: Stability



Decision Trees: Stability



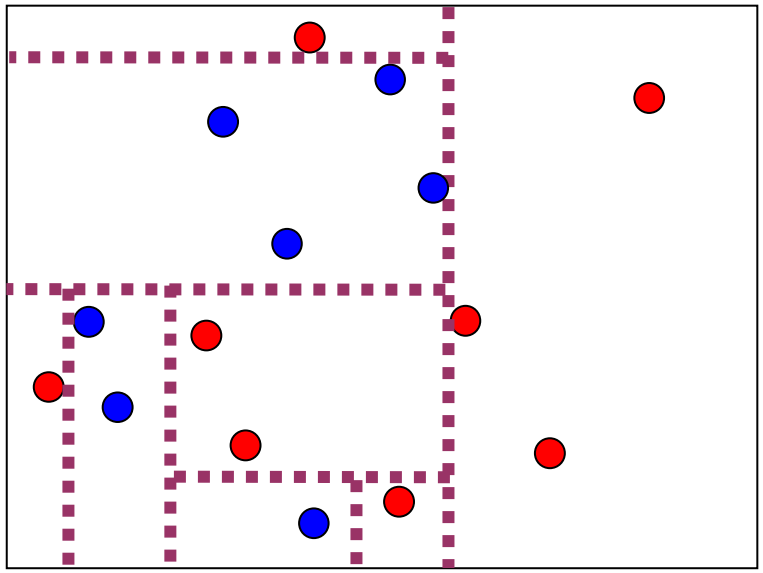
Decision Trees: Stability

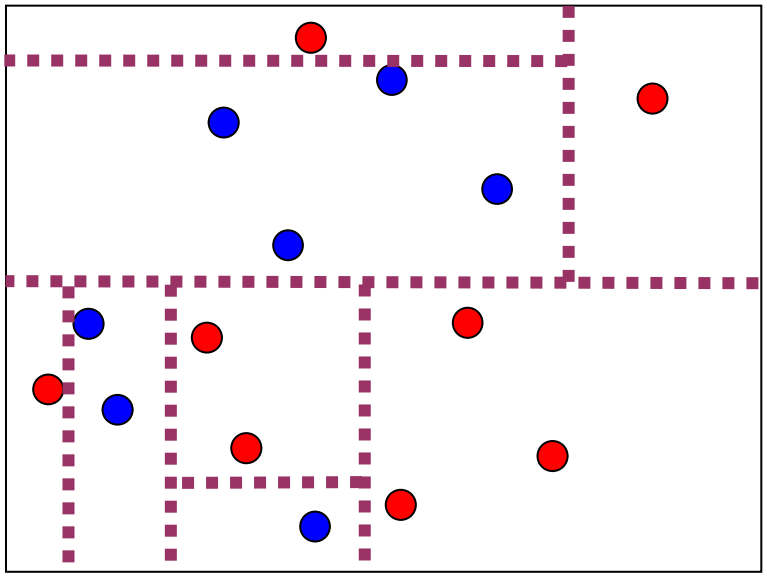


Decision Trees: Stability

Small changes in data

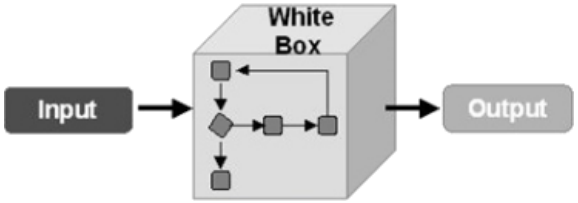
➔ potentially very different tree!





Decision Trees – properties

- Rather old model, well known
- Simple algorithm ➔ easy to understand
  - White box, rather than black-box
  - Used in many non-IT domains
  - Can be used to illustrate expert knowledge



- Various split criteria

- Problems with overfitting – (pre)pruning helps
- Problems with stability ➔ can be exploited!

Outline

- Short recap
- K-nn (continued)
- Data preparation
- Evaluation (intro)
- Decision Trees
- Evaluation (continued)

Confusion Matrix

- Matrix of classification results per class
  - Size (# classes) x (# classes)
- For each actual class plot the predicted classes
- Shows accuracy for single classes
- Indicates which classes are confused

Confusion Matrix: Example

	Grey	Black	Red	*Accuracy*
Grey	5	3	0	*0.625*
Black	2	3	1	*0.500*
Red	0	1	12	*0.920*
				0.740

- *How does the ideal matrix look like?*
  - Numbers only in the diagonal
  - In other cells: indicates misclassification

Confusion Matrix

- Important to analyse mistake patterns
  - *Which classes get mixed up?*

classified as											
a	b	c	d	e	f	g	h	i	j	k	genre
34	3	0	0	2	8	0	0	2	10	1	a = Country
9	39	0	1	1	4	0	0	0	5	1	b = Folk
0	2	47	0	1	4	1	0	1	4	0	c = Grunge
0	2	0	39	0	3	1	6	8	0	1	d = Hip-Hop
2	3	3	0	34	4	10	0	0	4	0	e = Metal
10	3	9	4	4	11	3	2	1	11	2	f = Pop
5	2	5	0	10	2	36	0	0	0	0	g = Punk Rock
2	0	0	10	0	3	0	40	2	1	2	h = R&B
0	1	0	7	0	1	0	2	45	0	4	i = Reggae
8	1	8	1	3	5	1	1	1	27	4	j = Slow Rock
1	0	0	0	0	1	0	1	3	2	52	k = Children's

Confusion Matrix

- Important to analyse mistake patterns
  - Which classes get mixed up

classified as											
a	b	c	d	e	f	g	h	i	j	k	genre
34	3	0	0	2	8	0	0	2	10	1	a = Country
9	39	0	1	1	4	0	0	0	5	1	b = Folk
0	2	47	0	1	4	1	0	1	4	0	c = Grunge
0	2	0	39	0	3	1	6	8	0	1	d = Hip-Hop
2	3	3	0	34	4	10	0	0	4	0	e = Metal
10	3	9	4	4	11	3	2	1	11	2	f = Pop
5	2	5	0	10	2	36	0	0	0	0	g = Punk Rock
2	0	0	10	0	3	0	40	2	1	2	h = R&B
0	1	0	7	0	1	0	2	45	0	4	i = Reggae
8	1	8	1	3	5	1	1	1	27	4	j = Slow Rock
1	0	0	0	0	1	0	1	3	2	52	k = Children's
47	69	65	63	62	23	69	76	71	42	77	Precision
57	65	78	65	57	18	6	67	75	45	87	Recall

Confusion Matrix: Example

	BigClass	SmallClass	*Accuracy*
BigClass	490	0	*100*
SmallClass	10	0	*0*
			0.98

Evaluation measures – averages

- Previous measures are ***micro-averaged***
- Do not indicate issues with imbalanced classes
- Alternative: macro-averaged measures
  - Compute precision, recall, ... ***per class***
  - Average class-results

Evaluation – micro average

	BigClass	SmallClass	*Accuracy*
BigClass	490	0	*100*
SmallClass	10	0	*0*
			0.98

- Accuracy:

$$\frac{TP + TN}{TP + FP + TN + FN} = \frac{TP + TN}{\#\ samples}$$

Evaluation – macro average

	BigClass	SmallClass	*Accuracy*
BigClass	490	0	*100*
SmallClass	10	0	*0*
			0.5

- Accuracy:

$$\frac{1}{|C|}\sum_{i=1}^{|C|}\frac{TP_i + TN_i}{TP_i + FP_i + TN_i + FN_i}$$

Performance per class

- Important to consider when
  - imbalanced classes
  - Performance of a particular class is more important

- *Examples ?*
  - Health prediction
  - Classify sensitive documents, …
  - Spam filter
  - Identify malicious software

Costs of misclassification

- Cost / loss functions
  - Measures per class with weighted averages
  - Higher weight to classes where errors are more severe

  ➔ Requires expert knowledge to identify weights

Effectiveness & Efficiency

- Effectiveness: quality of classification
  - Accuracy, precision, recall, F1, …
- Efficiency: computational efficiency (speed, runtime) of a classification
- Performance: often used as *synonym* for either *effectiveness OR efficiency* !

Effectiveness & Efficiency

- What is more important?

- Trade-off between effectiveness & efficiency
- Differentiate between efficiency on
  - Training (learning) a model
  - Classification

- Efficiency is more relevant if model needs to be (re-)trained frequently

Evaluation: outlook

- Other evaluation measures
- ROC curves
- Cross-validation & Bootstrapping
- Significance testing

- Evaluation measures for regression

Questions?