

# Machine Learning

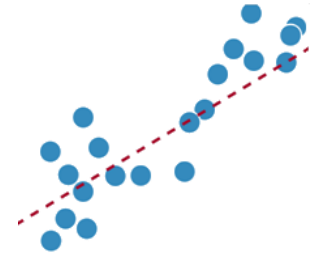
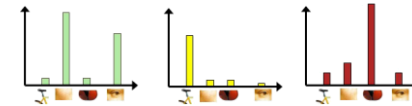
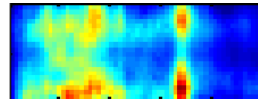
Rudolf Mayer  
October 25<sup>th</sup>, 2017

- Short recap
- Decision Trees – continued
- Evaluation
- Random Forests
- Evaluation, continued
- Data preparation
- SVM, intro
- SVM advancedselection, Ensemble learni,  
MLP/Neural Networks, Model ng, Significance  
testing, Feature selection

# Recap – Lecture 1

- ML Definitions & setting

- supervised, unsupervised, regression, classification reinforcement, ..
- Feature extraction



- Data types & preparation

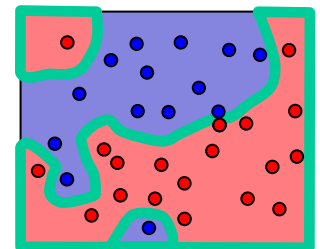
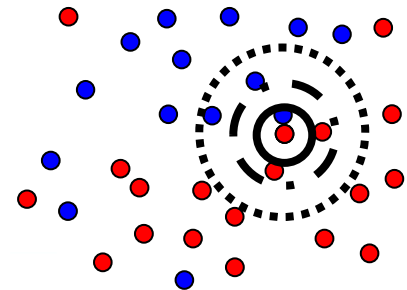
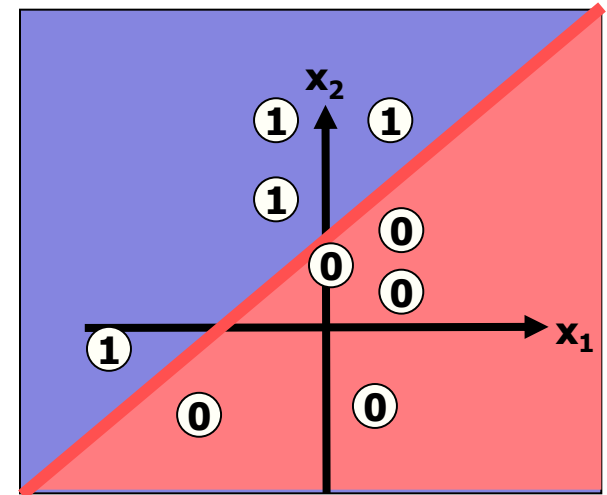
- Categorical (e.g. size: tiny/small/large) vs. numerical (size in cm)
- Scaling/Normalisation, 1-n coding, ..., **missing values/imputation, .. → more today**

- Outlook on evaluation

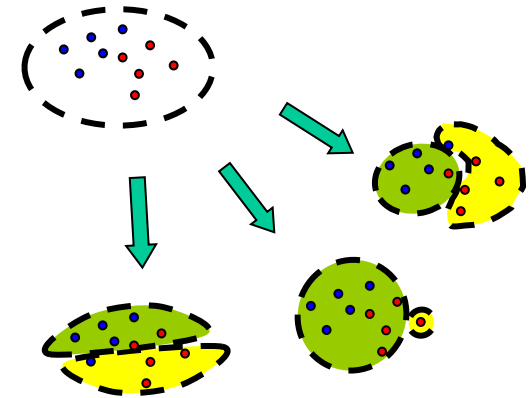
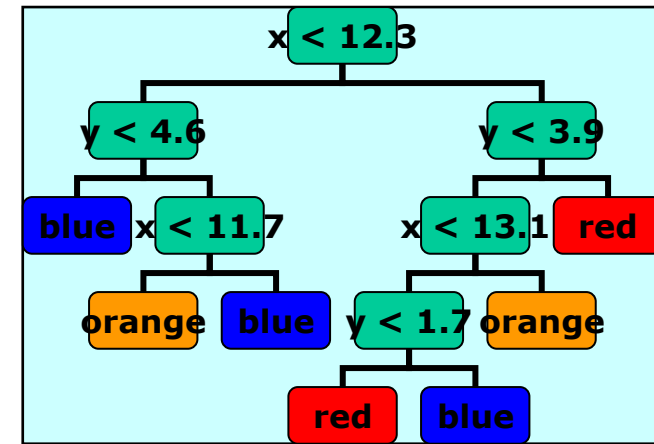
- Accuracy, Training/test set, ...
- **more today**

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

- Perceptron (1950s)
  - Linear separation
    - by linear combination of inputs
  - Learning weights & bias
  
- k-NN Classification
  - Searching for k-closest neighbours
  - Classification follows majority
  - Lazy learner
  - Optimisations for finding neighbours



- Decision Tree Learning
  - Finding optimal split
  - Different criteria for optimality
  - Binary & multiple classes
  - Overfitting & (pre)pruning
  - Stability
- Binary / n-ary trees
- Categorical & numerical data
- Stability



- Short recap
- **Decision Trees – continued**
- Evaluation
- Random Forests
- Evaluation, continued
- Data preparation
- SVM, intro

# Decision Trees – more examples

.....

- Previous example: 2D data, x & y axis (Cartesian space)
- Input data can be of any dimensionality
  - E.g. in 3D space of numerical data: planes dividing the space along x, y or z axis
- Splits not limited to binary (i.e. > two branches)
- Input data does not have to be numerical
  - ➔ decision trees also work on categorical data
- There can be more than two classes

# Miles Per Gallon Data Set

.....

<b>cylinders</b>	<b>displacement</b>	<b>horse power</b>	<b>weight</b>	<b>acceleration</b>	<b>Model year</b>	<b>maker</b>	<b>MpG</b>
4	low	low	low	high	75-78	Asia	good
6	medium	medium	medium	medium	70-74	America	bad
4	medium	medium	medium	low	75-78	Europe	bad
8	high	high	high	low	70-74	America	bad
6	medium	medium	medium	medium	70-74	America	bad
4	low	medium	low	medium	70-74	Asia	bad
4	low	medium	low	low	70-74	Asia	bad
8	high	high	high	low	75-78	America	bad
8	high	high	high	low	70-74	America	bad
8	high	medium	high	high	79-83	America	good
8	high	high	high	low	75-78	America	bad
4	low	low	low	low	79-83	America	good
6	medium	medium	medium	high	75-78	America	bad
4	medium	low	low	low	79-83	America	good
4	low	low	medium	high	79-83	America	good
8	high	high	high	low	70-71	America	bad
4	low	medium	low	medium	75-78	Europe	good
5	medium	medium	medium	medium	75-78	Europe	bad

18/40 Records subsample (similar in UCI Machine learning repository)



# Miles Per Gallon Data Set

cylinders	displacement	horse power	weight	acceleration	Model year	maker	MpG
4	low	low	low	high	75-78	Asia	good
6	medium	medium	medium	medium	70-74	America	bad
4	medium	medium	medium	low	75-78	Europe	bad
8	high	high	high	low	70-74	America	bad
6	medium	medium	medium	medium	70-74	America	bad
4	low	medium	low	medium	70-74	Asia	bad
4	low	medium	low	low	70-74	Asia	bad
8	high	high	high	low	75-78	America	bad
8	high	high	high	low	70-74	America	bad
8	high	medium	high	high	79-83	America	good
8	high	high	high	low	75-78	America	bad
4	low	low	low	low	79-83	America	good
6	medium	medium	medium	high	75-78	America	bad
4	medium	low	low	low	79-83	America	good
4	low	low	medium	high	79-83	America	good
8	high	high	high	low	70-71	America	bad
4	low	medium	low	medium	75-78	Europe	good
5	medium	medium	medium	medium	75-78	Europe	bad

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

12 samples class bad (2/3), 6 samples good (1/3)

# Miles Per Gallon Data Set

.....

cylinders	displacement	horse power	weight	acceleration	Model year	maker	MpG
4	low	low	low	high	75-78	Asia	good
6	medium	medium	medium	medium	70-74	America	bad
4	medium	medium	medium	low	75-78	Europe	bad
8	high	high	high	low	70-74	America	bad
6	medium	medium	medium	medium	70-74	America	bad
4	low	medium	low	medium	70-74	Asia	bad
4	low	medium	low	low	70-74	Asia	bad
8	high	high	high	low	75-78	America	bad
8	high	high	high	low	70-74	America	bad
8	high	medium	high	high	79-83	America	good
8	high	high	high	low	75-78	America	bad
4	low	low	low	low	79-83	America	good
6	medium	medium	medium	high	75-78	America	bad
4	medium	low	low	low	79-83	America	good
4	low	low	medium	high	79-83	America	good
8	high	high	high	low	70-71	America	bad
4	low	medium	low	medium	75-78	Europe	good
5	medium	medium	medium	medium	75-78	Europe	bad

Entropy of data set:

$$\begin{aligned}
 & - 1/3 \times \log_2 1/3 - 2/3 \times \log_2 2/3 = - 1/3 \times \log(1/3)/\log(2) - 2/3 \times \log(2/3)/\log(2) \\
 & = - 1/3 \times -1,59946 - 2/3 \times -0,58496 \\
 & = 0,918295834
 \end{aligned}$$

# Miles Per Gallon Data Set

<b>cylinders</b>	<b>displacement</b>	<b>horse power</b>	<b>weight</b>	<b>acceleration</b>	<b>Model year</b>	<b>maker</b>	<b>MpG</b>
4	low	low	low	high	75-78	Asia	good
4	low	low	low	low	79-83	America	good
4	low	medium	low	medium	75-78	Europe	good
4	medium	low	low	low	79-83	America	good
4	low	low	medium	high	79-83	America	good
4	medium	medium	medium	low	75-78	Europe	bad
4	low	medium	low	medium	70-74	Asia	bad
4	low	medium	low	low	70-74	Asia	bad
5	medium	medium	medium	medium	75-78	Europe	bad
6	medium	medium	medium	medium	70-74	America	bad
6	medium	medium	medium	medium	70-74	America	bad
6	medium	medium	medium	high	75-78	America	bad
8	high	medium	high	high	79-83	America	good
8	high	high	high	low	75-78	America	bad
8	high	high	high	low	70-74	America	bad
8	high	high	high	low	75-78	America	bad
8	high	high	high	low	70-74	America	bad
8	high	high	high	low	70-71	America	bad

Split on first attribute – cylinders

- Sort data set by cylinders & MpG (output variable)

# Miles Per Gallon Data Set

<i>cylinders</i>	displacement	horse power	weight	acceleration	Model year	maker	MpG
4	low	low	low	high	75-78	Asia	good
4	low	low	low	low	79-83	America	good
4	low	medium	low	medium	75-78	Europe	good
4	medium	low	low	low	79-83	America	good
4	low	low	medium	high	79-83	America	good
4	medium	medium	medium	low	75-78	Europe	bad
4	low	medium	low	medium	70-74	Asia	bad
4	low	medium	low	low	70-74	Asia	bad
5	medium	medium	medium	medium	75-78	Europe	bad
6	medium	medium	medium	medium	70-74	America	bad
6	medium	medium	medium	medium	70-74	America	bad
6	medium	medium	medium	high	75-78	America	bad
8	high	medium	high	high	79-83	America	good
8	high	high	high	low	75-78	America	bad
8	high	high	high	low	70-74	America	bad
8	high	high	high	low	75-78	America	bad
8	high	high	high	low	70-74	America	bad
8	high	high	high	low	70-71	America	bad

Split on first attribute – cylinders

- Sort data set by cylinders & MpG (output variable)
- **Identify subsets: 4 distinct values → 4 sets**

# Miles Per Gallon Data Set

<i>cylinders</i>	displacement	horse power	weight	acceleration	Model year	maker	MpG
4	low	low	low	high	75-78	Asia	good
4	low	low	low	low	79-83	America	good
4	low	medium	low	medium	75-78	Europe	good
4	medium	low	low	low	79-83	America	good
4	low	low	medium	high	79-83	America	good
4	medium	medium	medium	low	75-78	Europe	bad
4	low	medium	low	medium	70-74	Asia	bad
4	low	medium	low	low	70-74	Asia	bad
5	medium	medium	medium	medium	75-78	Europe	bad
6	medium	medium	medium	medium	70-74	America	bad
6	medium	medium	medium	medium	70-74	America	bad
6	medium	medium	medium	high	75-78	America	bad
8	high	medium	high	high	79-83	America	good
8	high	high	high	low	75-78	America	bad
8	high	high	high	low	70-74	America	bad
8	high	high	high	low	75-78	America	bad
8	high	high	high	low	70-74	America	bad
8	high	high	high	low	70-71	America	bad

Split on first attribute – cylinders

- 4 distinct values – split in 4 sets
- Compute IG – compute entropy for each subset

# Miles Per Gallon Data Set

.....

cylinders	displacement	horse power	weight	acceleration	Model year	maker	MpG
4	low	low	low	high	75-78	Asia	good
4	low	low	low	low	79-83	America	good
4	low	medium	low	medium	75-78	Europe	good
4	medium	low	low	low	79-83	America	good
4	low	low	medium	high	79-83	America	good
4	medium	medium	medium	low	75-78	Europe	bad
4	low	medium	low	medium	70-74	Asia	bad
4	low	medium	low	low	70-74	Asia	bad

5 samples class good (5/8), 3 samples class bad (3/8)

$$\begin{aligned}
 H(X_{\text{cylinders}=4}) &= -5/8 \times \log_2(5/8) - 3/8 \times \log_2(3/8) \\
 &= (-5/8 \log(5/8) \log(2)) + (-3/8 \log(3/8) \log(2)) \\
 &= 0,954434003
 \end{aligned}$$

# Miles Per Gallon Data Set

.....

cylinders	displacement	horse power	weight	accelleration	Model year	maker	MpG
5	medium	medium	medium	medium	75-78	Europe	bad

1 sample class bad

$$H(X_{\text{cylinders}=5}) = 0$$

# Miles Per Gallon Data Set

.....

cylinders	displacement	horse power	weight	accelleration	Model year	maker	MpG
6	medium	medium	medium	medium	70-74	America	bad
6	medium	medium	medium	medium	70-74	America	bad
6	medium	medium	medium	high	75-78	America	bad

3 sample class bad

$$H(X_{\text{cylinders}=6}) = 0$$



# Miles Per Gallon Data Set

.....

cylinders	displacement	horse power	weight	acceleration	Model year	maker	MpG
8	high	medium	high	high	79-83	America	good
8	high	high	high	low	75-78	America	bad
8	high	high	high	low	70-74	America	bad
8	high	high	high	low	75-78	America	bad
8	high	high	high	low	70-74	America	bad
8	high	high	high	low	70-71	America	bad

1 sample class good (1/6), 5 samples class bad (5/6),

$$\begin{aligned}
 H(X_{\text{cylinders}=8}) &= -1/6 \times \log_2(1/6) - 5/6 \times \log_2(5/6) \\
 &= (-1/6 \log(1/6) \log(2)) + (-5/6 \log(5/6) \log(2)) \\
 &= 0,650022422
 \end{aligned}$$

# Miles Per Gallon Data Set

cylinders	displacement	horse power	weight	acceleration	Model year	maker	MpG
4	low	low	low	high	75-78	Asia	good
4	low	low	low	low	79-83	America	good
4	low	medium	low	medium	75-78	Europe	good
4	medium	low	low	low	79-83	America	good
4	low	low	medium	high	79-83	America	good
4	medium	medium	medium	low	75-78	Europe	bad
4	low	medium	low	medium	70-74	Asia	bad
4	low	medium	low	low	70-74	Asia	bad
5	medium	medium	medium	medium	75-78	Europe	bad
6	medium	medium	medium	medium	70-74	America	bad
6	medium	medium	medium	medium	70-74	America	bad
6	medium	medium	medium	high	75-78	America	bad
8	high	medium	high	high	79-83	America	good
8	high	high	high	low	75-78	America	bad
8	high	high	high	low	70-74	America	bad
8	high	high	high	low	75-78	America	bad
8	high	high	high	low	70-74	America	bad
8	high	high	high	low	70-71	America	bad

Entropy of split:

$$\begin{aligned}
 H(X_{cyl}) &= p(x_{cyl=4})H(X_{cyl=4}) + p(x_{cyl=5})H(X_{cyl=5}) + p(x_{cyl=6})H(X_{cyl=6}) + p(x_{cyl=8})H(X_{cyl=8}) \\
 &= \frac{8}{18} \times 0,95443 + \frac{1}{18} \times 0 + \frac{3}{18} \times 0 + \frac{6}{18} \times 0,6500
 \end{aligned}$$

# Miles Per Gallon Data Set

cylinders	displacement	horse power	weight	acceleration	Model year	maker	MpG
4	low	low	low	high	75-78	Asia	good
4	low	low	low	low	79-83	America	good
4	low	medium	low	medium	75-78	Europe	good
4	medium	low	low	low	79-83	America	good
4	low	low	medium	high	79-83	America	good
4	medium	medium	medium	low	75-78	Europe	bad
4	low	medium	low	medium	70-74	Asia	bad
4	low	medium	low	low	70-74	Asia	bad
5	medium	medium	medium	medium	75-78	Europe	bad
6	medium	medium	medium	medium	70-74	America	bad
6	medium	medium	medium	medium	70-74	America	bad
6	medium	medium	medium	high	75-78	America	bad
8	high	medium	high	high	79-83	America	good
8	high	high	high	low	75-78	America	bad
8	high	high	high	low	70-74	America	bad
8	high	high	high	low	75-78	America	bad
8	high	high	high	low	70-74	America	bad
8	high	high	high	low	70-71	America	bad

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

# Miles Per Gallon Data Set

cylinders	displacement	horse power	weight	acceleration	Model year	maker	MpG
4	low	low	low	high	75-78	Asia	good
4	low	low	low	low	79-83	America	good
4	low	medium	low	medium	75-78	Europe	good
4	medium	low	low	low	79-83	America	good
4	low	low	medium	high	79-83	America	good
4	medium	medium	medium	low	75-78	Europe	bad
4	low	medium	low	medium	70-74	Asia	bad
4	low	medium	low	low	70-74	Asia	bad
5	medium	medium	medium	medium	75-78	Europe	bad
6	medium	medium	medium	medium	70-74	America	bad
6	medium	medium	medium	medium	70-74	America	bad
6	medium	medium	medium	high	75-78	America	bad
8	high	medium	high	high	79-83	America	good
8	high	high	high	low	75-78	America	bad
8	high	high	high	low	70-74	America	bad
8	high	high	high	low	75-78	America	bad
8	high	high	high	low	70-74	America	bad
8	high	high	high	low	70-71	America	bad

Information Gain:  $IG(X_A, X_B) = H(X) - p(x_A)H(X_A) - p(x_B)H(X_B)$   
 $0,918295834 - 8/18 \times 0,954434003 - 6/18 \times 0,650022422$   
 $= 0,277428803$

# Miles Per Gallon Data Set

cylinders	displacement	horse power	weight	acceleration	Model year	maker	MpG
4	low	low	low	high	75-78	Asia	good
4	low	low	low	low	79-83	America	good
4	low	medium	low	medium	75-78	Europe	good
4	low	low	medium	high	79-83	America	good
4	low	medium	low	medium	70-74	Asia	bad
4	low	medium	low	low	70-74	Asia	bad
4	medium	low	low	low	79-83	America	good
5	medium	medium	medium	medium	75-78	Europe	bad
6	medium	medium	medium	medium	70-74	America	bad
6	medium	medium	medium	medium	70-74	America	bad
6	medium	medium	medium	high	75-78	America	bad
4	medium	medium	medium	low	75-78	Europe	bad
8	high	medium	high	high	79-83	America	good
8	high	high	high	low	75-78	America	bad
8	high	high	high	low	70-74	America	bad
8	high	high	high	low	75-78	America	bad
8	high	high	high	low	70-74	America	bad
8	high	high	high	low	70-71	America	bad
























Split on second attribute – displacement

- 3 distinct values – split in 3 sets
- Compute IG – compute entropy for each subset

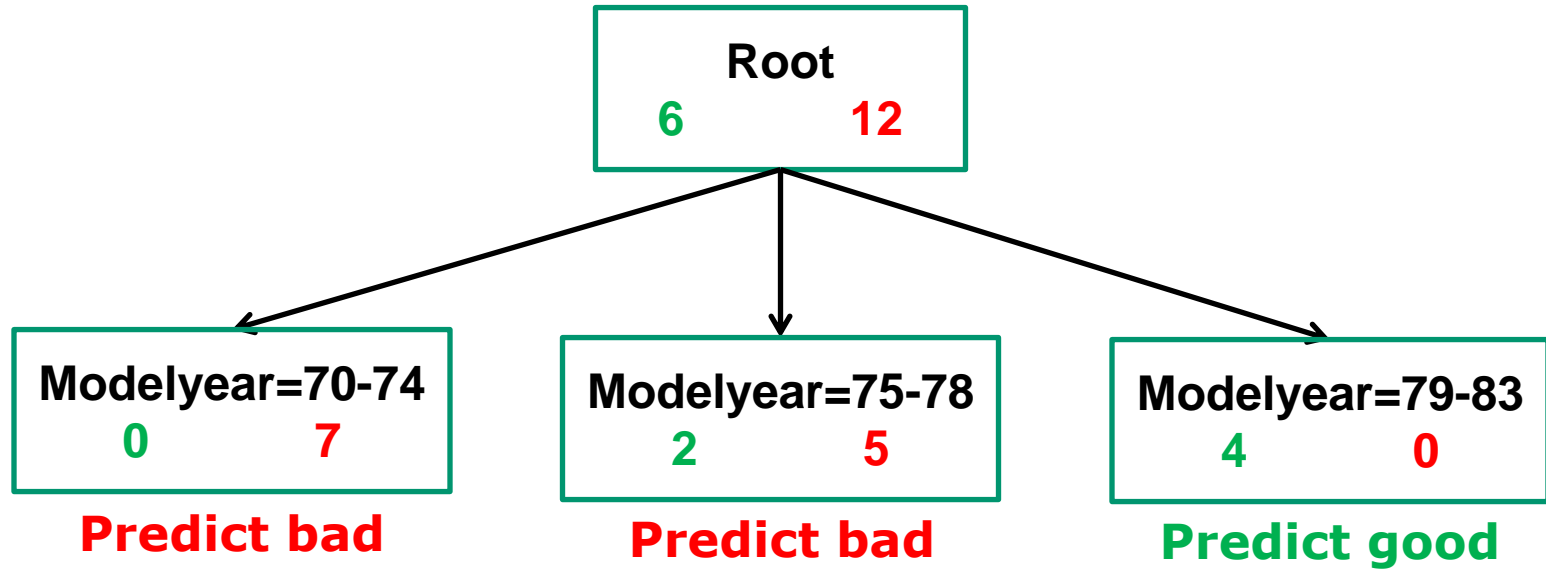
# Training the tree

## • Build a decision tree

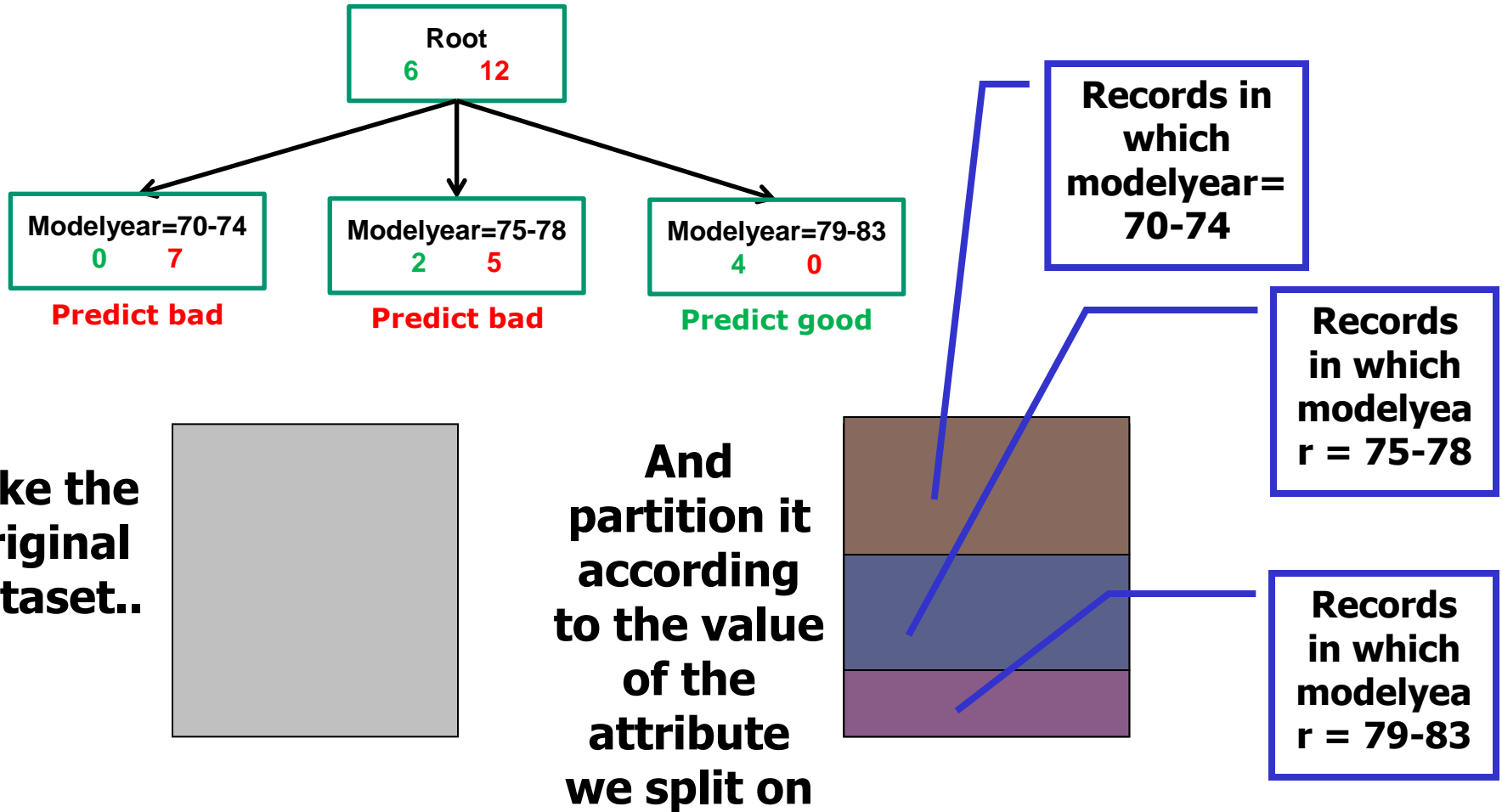
1. Identify splits
2. Compute IGs
3. Select attribute with highest IG  
→ Model Year

Attribute	Attr.Value	good	bad	Distribution	Entropy	Info Gain
Full Dataset	*	6	12		0.9183	
Attribute	Attr.Value	good	bad	Distribution	Entropy	Info Gain
cylinders	4	5	3		0.9544	
	5	0	3		0.	
	6	0	3		0.	
	8	1	3		0.8113	
Split					<b>0.6045</b>	<b>0.3138</b>
displacement	low	4	2		0.9183	
	medium	1	5		0.65	
	high	1	5		0.65	
Split					<b>0.7394</b>	<b>0.1788</b>
horse power	low	4	0		0.	
	medium	2	7		0.7642	
	high	0	5		0.	
Split					<b>0.3821</b>	<b>0.5362</b>
weight	low	4	2		0.9183	
	medium	1	5		0.65	
	high	1	5		0.65	
Split					<b>0.7394</b>	<b>0.1788</b>
acceleration	low	2	7		0.7642	
	medium	1	4		0.7219	
	high	3	1		0.8113	
Split					<b>0.7629</b>	<b>0.1554</b>
Model year	70-74	0	7		0.	
	75-78	2	5		0.8631	
	79-83	4	0		0.	
Split					<b>0.3357</b>	<b>0.5826</b>
maker	Asia	1	2		0.9183	
	America	4	8		0.9183	
	Europe	1	2		0.9183	
Split					0.9183	0.

# First level of Decision Tree

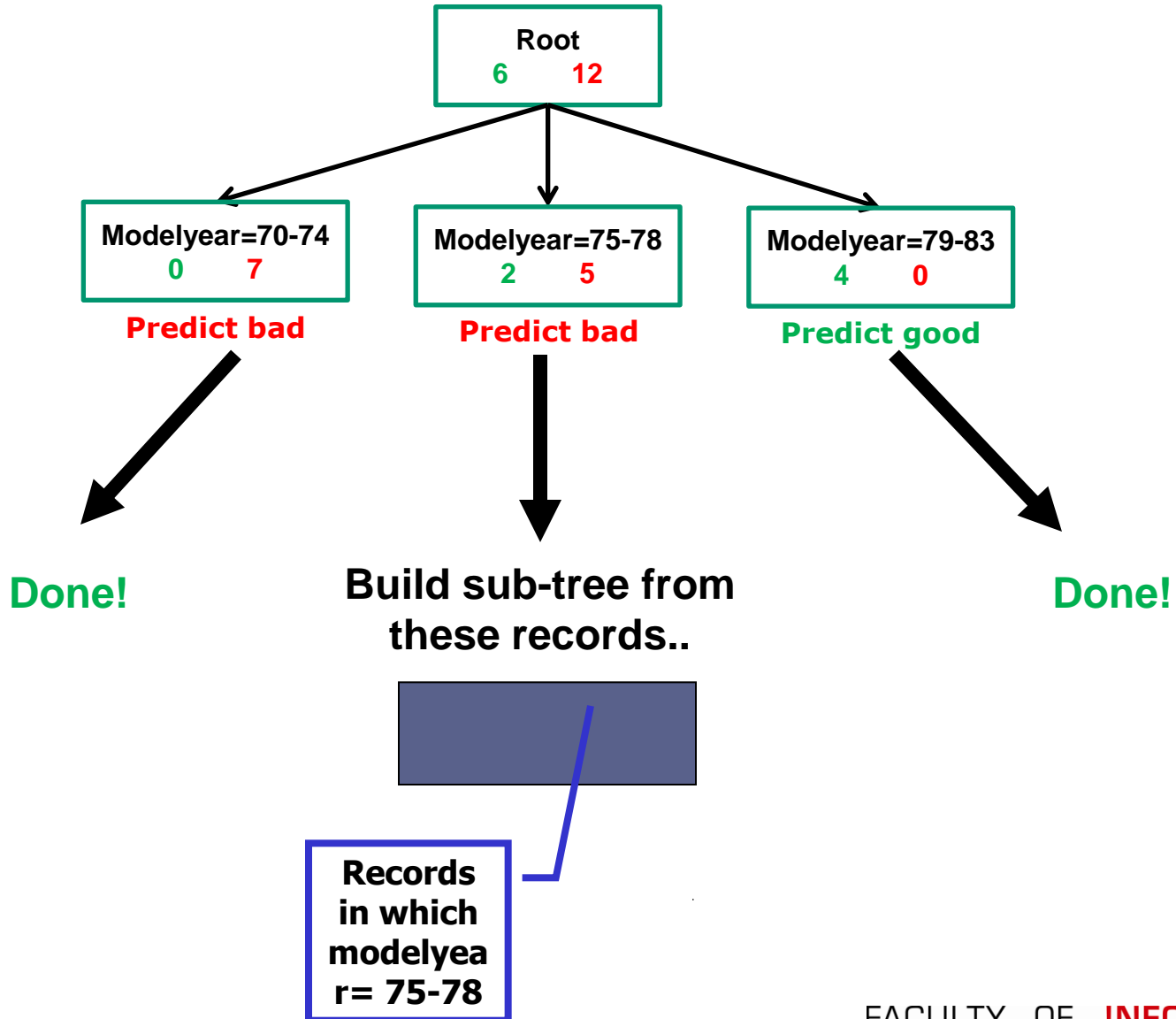


# Recursion Step






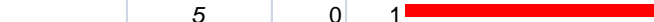







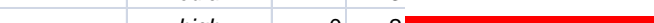


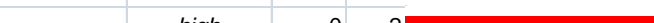
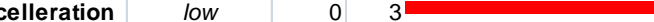

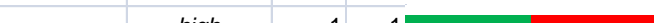

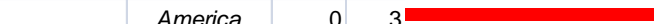

# Recursion Step



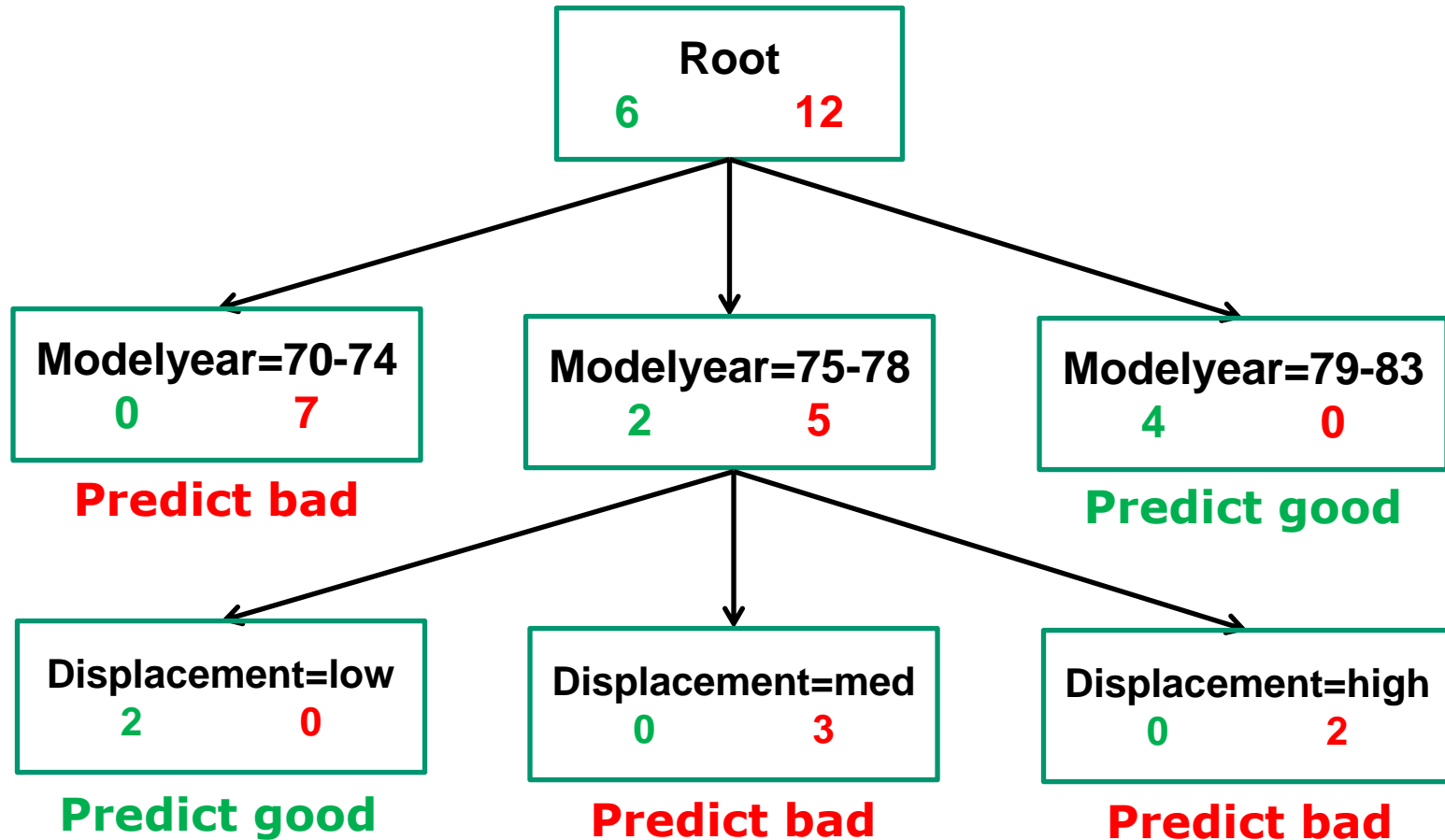
# Second level of tree

- Only one node from first level needs expansion

- Identify splits  
(on all attributes except *modelyear*)
- Compute IGs
- Select attribute with highest IG  
→ displacement OR weight

Attribute	Attr.Value	good	bad	Distribution	Entropy	Info Gain
cylinders	4	2	1		0.9183	
	5	0	1		0	
	6	0	1		0	
	8	0	2		0	
Split					0.3936	0.5247
displacement	low	2	0		0	
	medium	0	3		0	
	high	0	2		0	
Split					0	0.9183
horse power	low	1	0		0	
	medium	1	3		0.8113	
	high	0	2		0	
Split					0.4636	0.4547
weight	low	2	0		0	
	medium	0	3		0	
	high	0	2		0	
Split					0	0.9183
accelleration	low	0	3		0	
	medium	1	1		1.	
	high	1	1		1.	
Split					0.5714	0.3469
maker	Asia	1	0		0	
	America	0	3		0	
	Europe	1	2		0.9183	
Split					0.3936	0.5247

# Second level of tree



## 'Play golf/tennis' data set

Outlook	Temperature	Humidity	Windy	Play?
sunny	85	85	false	Don't Play
sunny	80	90	true	Don't Play
overcast	83	78	false	Play
rain	70	96	false	Play
rain	68	80	false	Play
rain	65	70	true	Don't Play
overcast	64	65	true	Play
sunny	72	95	false	Don't Play
sunny	69	70	false	Play
rain	75	80	false	Play
sunny	75	70	true	Play
overcast	72	90	true	Play
overcast	81	75	false	Play
rain	71	80	true	Don't Play

- *Solve it at home as an exercise !*
- Discussion next lecture

- For each leaf node
  - If not all data from the same class (*or other stopping criterion*)
    - For **each** attribute
      - Identify possible splits of samples into (two or more) subspaces
    - Compute **best** split (over all attributes!)
      - Based on a split goodness measure/criterion
  - Until data in all leaf nodes is *pure* (same class)
    - Or cannot be distinguished (*When can this happen?*)
    - *Or other stopping criterion fulfilled (e.g. maximum depth)*

- For ***each*** attribute
  - Identify possible splits of samples into (two or more) subspaces
    - categorical variables? (e.g. size with values “small” / “medium” / “large”)
      - *By each variable value, i.e. split into 3 sub-branches*
      - *Or one value vs. other values: small vs. rest, medium vs rest, large vs. rest (split into 2 sub-branches)*
      - *Difference?*
    - numerical variables? (e.g. size in centimeters)
      - *sort values & split between each pair of values*
      - ➔ *How many candidate splits?*

- Short recap
- Decision Trees – continued
- **Evaluation**
- Random Forests
- Evaluation, continued
- Data preparation
- SVM, intro

- 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*)**

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

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

	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

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

– Inverse: Error rate

- Precision  $\frac{TP}{TP + FP}$

- Recall  $\frac{TP}{TP + FN}$

- F-Measure: trade-off between precision and recall; F1:

$$\frac{2 * (precision * recall)}{(precision + recall)}$$

- Value ranges?*

- 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 split

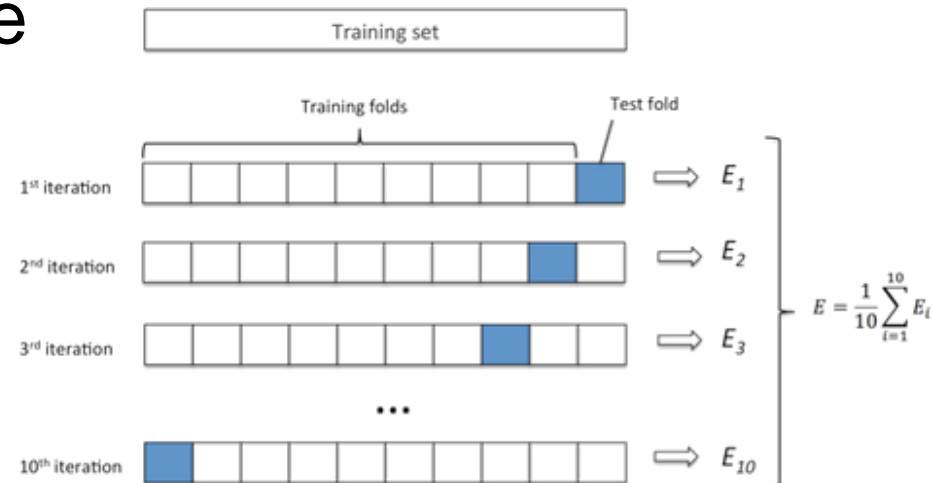
.....

- Really unseen data doesn't have labels
  - “Simulate” “unseen” data
  - “*Holdout method*”
  - Split labelled data into training and test (validation) sets
    - E.g. ~80% training, 20% test, 66% - 33%
    - Linear (first 80%), randomised, ...
- Performance on test set is an estimate for generalisation ability of our model
- Results can vary a **lot** according to how split is done!
  - ➔ *Cross validation*

# 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

- 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

↙  
To have not too small test/training sets

↘  
Due to computational reasons

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



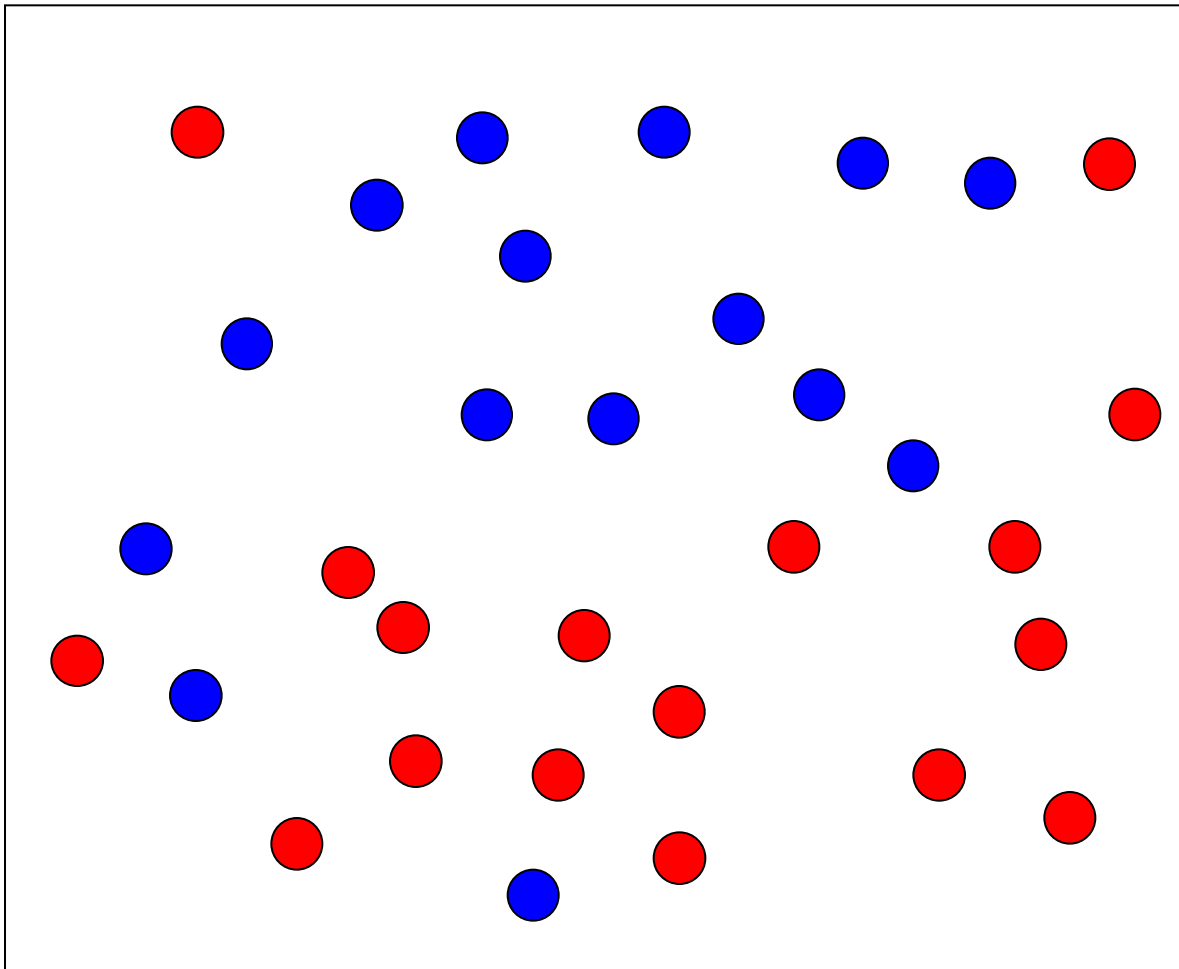
# Leave- $p$ -out Cross validation

.....

- A type of ***exhaustive*** cross-validation
  - Use  $p$  observations in test (validation) set
    - Remaining samples are in training set
  - Repeated **for all combinations** to cut  $p$  samples
  - Quickly becomes computationally infeasible
    - 100 samples,  $p=30$ 
      - $3 \times 10^{25}$  combinations!
- Special case:  $p=1$ , leave-one-out cross validation
  - Test/validation set contains one sample
  - **Number of combinations?**
    - $n$

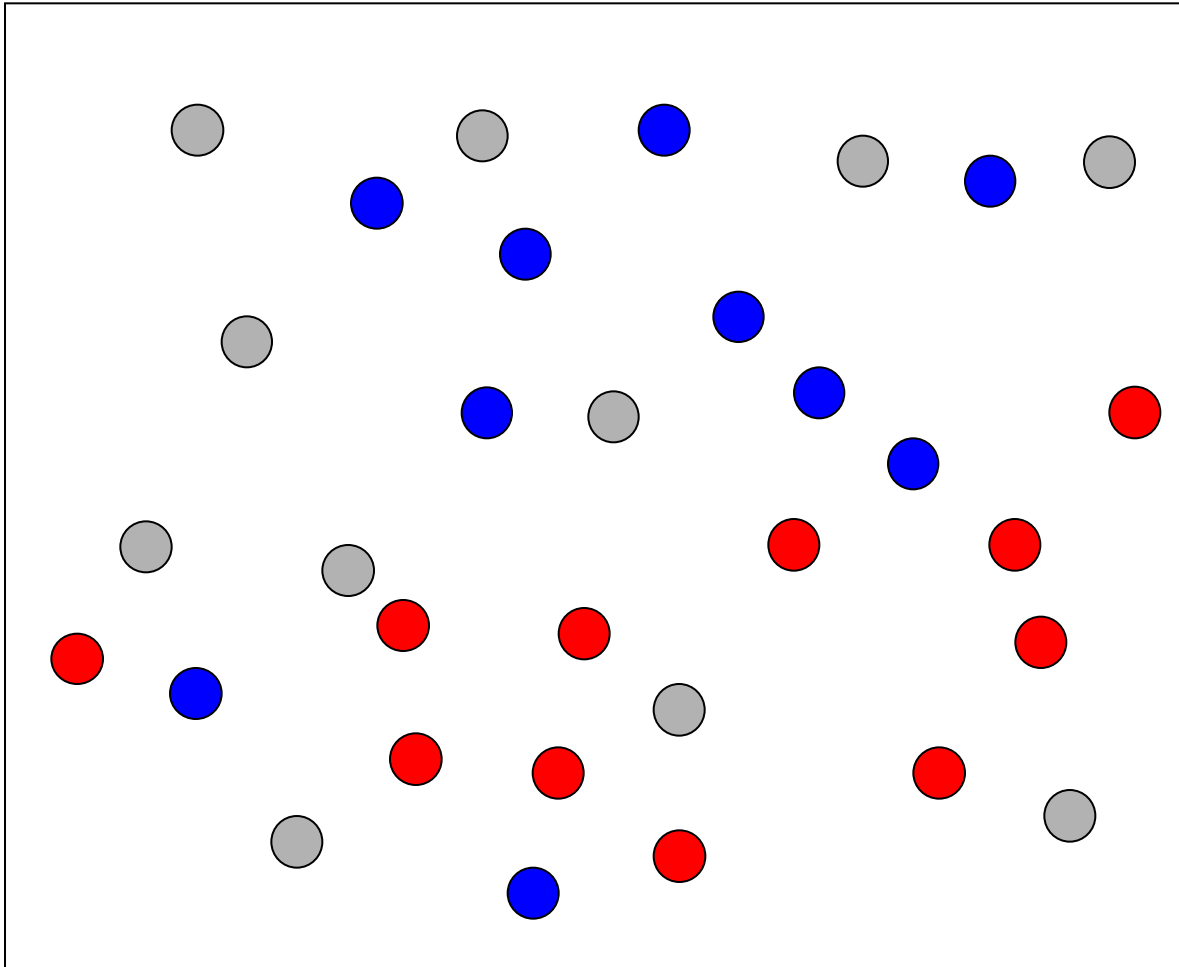
- A bootstrap sample is a random subset of the data sample
  - Test set is also random sample
- Data points may be selected repeatedly
  - i.e. selection with replacement
- An arbitrary number of bootstrap samples may be used
- Bootstrapping is an alternative to cross validation and holdout method (training-test split)

# Example: Bootstrapping



# Example: Bootstrapping

.....



- Short recap
- Decision Trees – continued
- Evaluation
- **Random Forests**
- Evaluation, continued
- Data preparation
- SVM, intro

- Combination of Decision Trees and bootstrapping concepts
- Proposed ~1995 by Leo Breiman & Adele Cutler
- Basic Idea & Name: a large number of decision trees is “grown in the forest”, each on a different bootstrap sample

- For each tree: use bootstrap sample
- For each tree, only a random number of the original variables is available
  - i.e. small selection of columns
  - much smaller than original number
  - Change at each tree node!
- Grow trees to maximal extend
  - No stopping
  - No pruning
  - (they are rather small anyhow)

# Example: Tree in Random Forests

.....

Input					Output
12	A	0.1	501	red	I
8	B	1.2	499	red	II
9	B	1.1	504	blue	II
15	A	1.8	480	green	II
2	C	1.0	511	red	I
-2	C	0.7	512	green	II
7	C	0.4	488	cyan	I
7	A	0.6	491	cyan	I
10	A	1.5	500	cyan	I
0	C	0.3	505	blue	II
9	B	1.9	502	blue	II



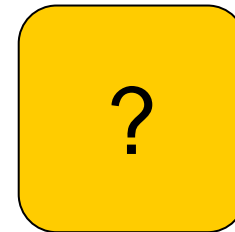
# Example: Tree in Random Forests

Input					Output
12	A	0.1	501	red	I
8	B	1.2	499	red	II
15	A	1.8	480	green	II
2	C	1.0	511	red	I
7	C	0.4	488	cyan	I
7	A	0.6	491	cyan	I
0	C	0.3	505	blue	II

Bootstrap sample

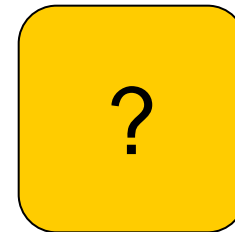
# Example: Tree in Random Forests

Input					Output
12	A	0.1	501	red	I
8	B	1.2	499	red	II
15	A	1.8	480	green	II
2	C	1.0	511	red	I
7	C	0.4	488	cyan	I
7	A	0.6	491	cyan	I
0	C	0.3	505	blue	II



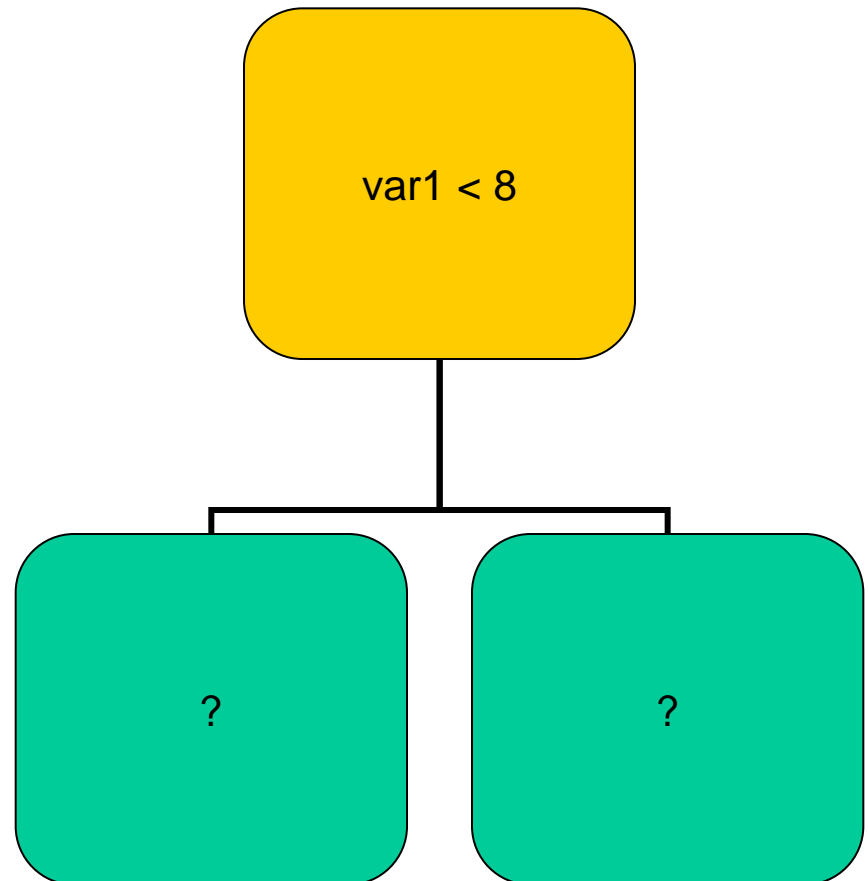
# Example: Tree in Random Forests

Input				Output	
12	A	0.1	501	red	I
8	B	1.2	499	red	II
15	A	1.8	480	green	II
2	C	1.0	511	red	I
7	C	0.4	488	cyan	I
7	A	0.6	491	cyan	I
0	C	0.3	505	blue	II



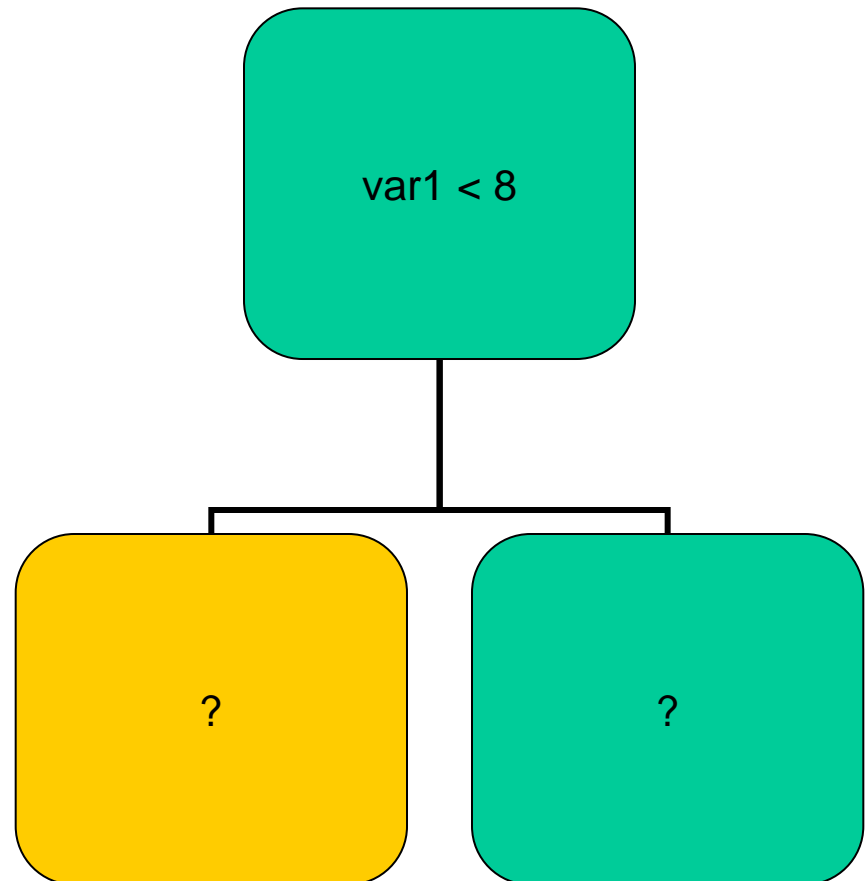
# Example: Tree in Random Forests

Input					Output
12	A	0.1	501	red	I
8	B	1.2	499	red	II
15	A	1.8	480	green	II
2	C	1.0	511	red	I
7	C	0.4	488	cyan	I
7	A	0.6	491	cyan	I
0	C	0.3	505	blue	II



# Example: Tree in Random Forests

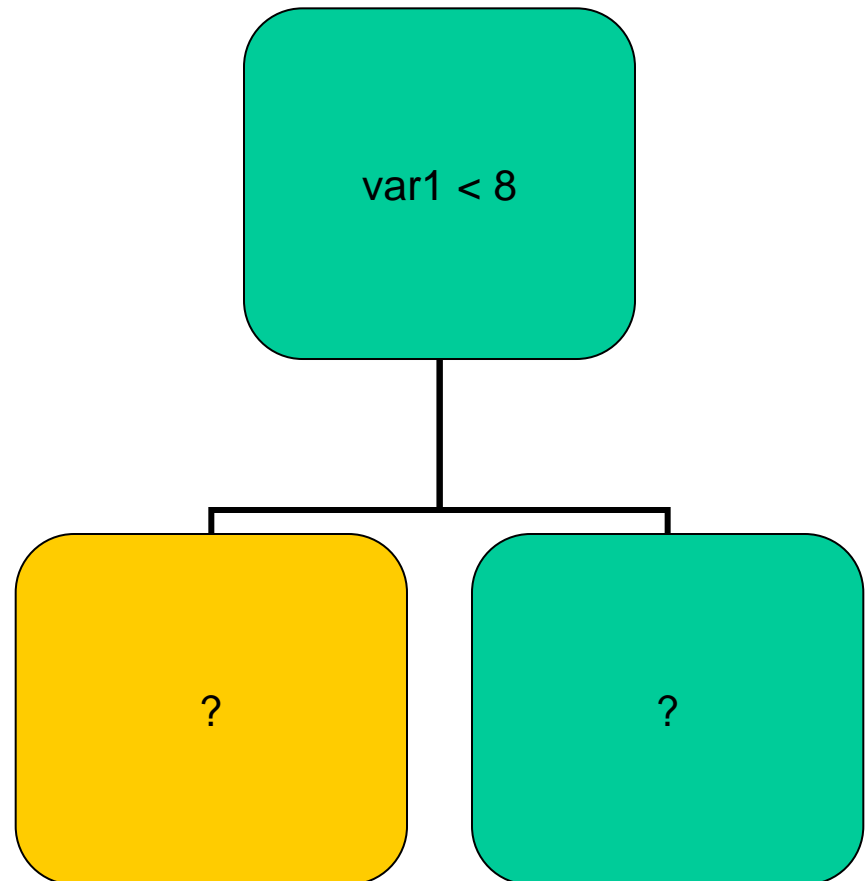
Input					Output
12	A	0.1	501	red	I
8	B	1.2	499	red	II
15	A	1.8	480	green	II
2	C	1.0	511	red	I
7	C	0.4	488	cyan	I
7	A	0.6	491	cyan	I
0	C	0.3	505	blue	II



# Example: Tree in Random Forests

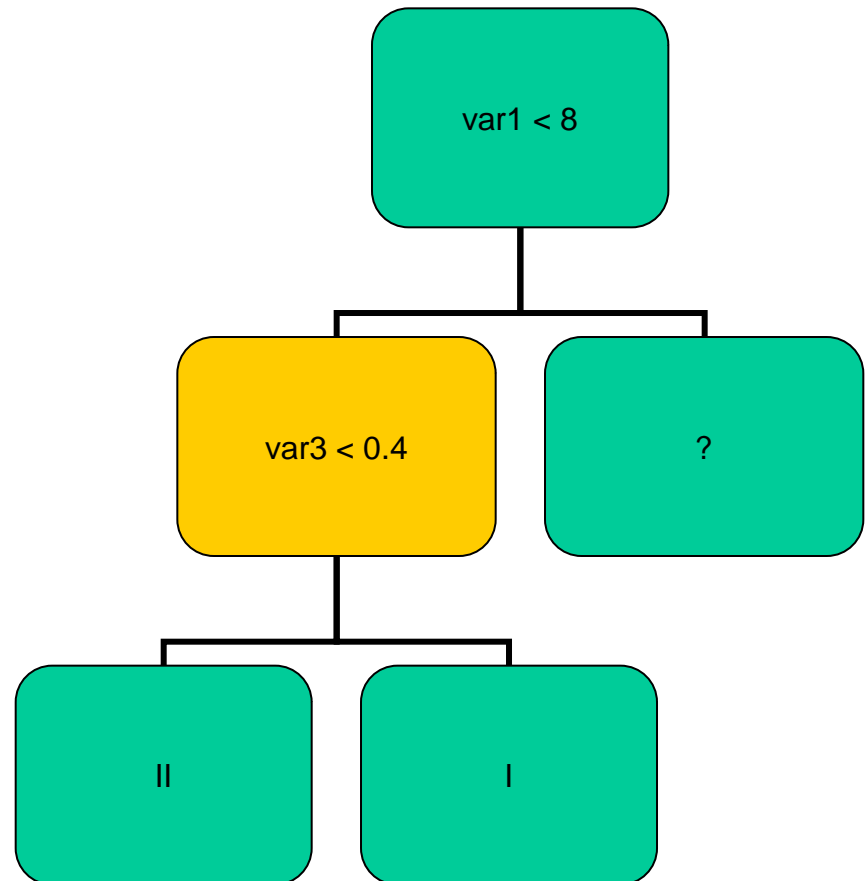
!! Select new attributes at each tree node !!

Input					Output
12	A	0.1	501	red	I
8	B	1.2	499	red	II
15	A	1.8	480	green	II
2	C	1.0	511	red	I
7	C	0.4	488	cyan	I
7	A	0.6	491	cyan	I
0	C	0.3	505	blue	II



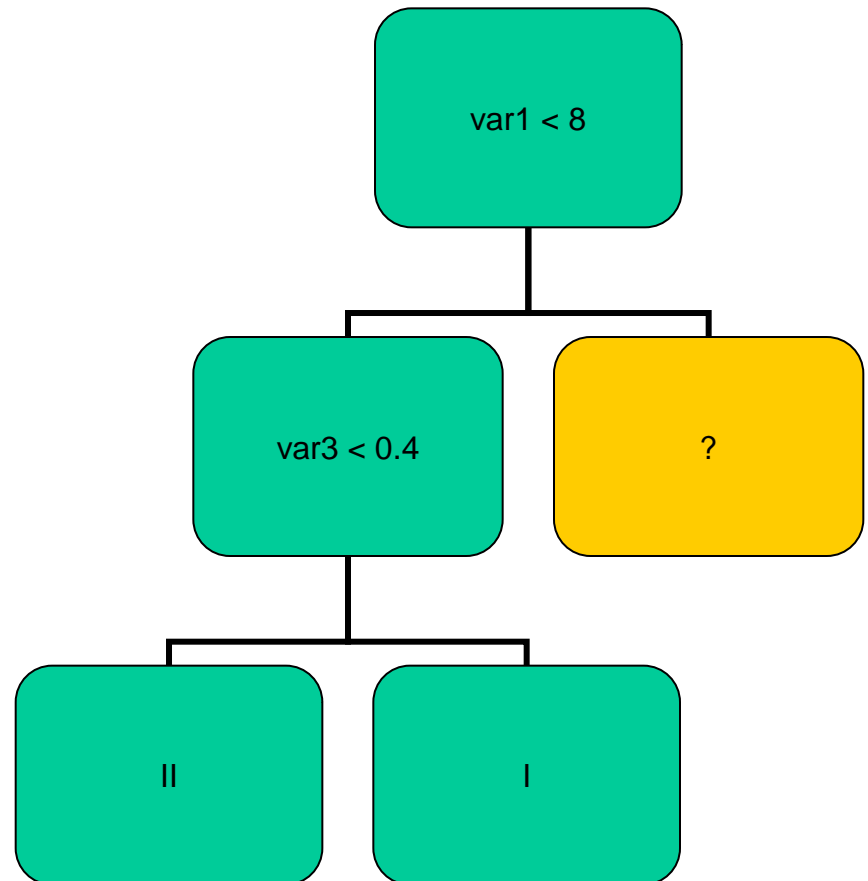
# Example: Tree in Random Forests

Input					Output
12	A	0.1	501	red	I
8	B	1.2	499	red	II
15	A	1.8	480	green	II
2	C	1.0	511	red	I
7	C	0.4	488	cyan	I
7	A	0.6	491	cyan	I
0	C	0.3	505	blue	II



# Example: Tree in Random Forests

Input					Output
12	A	0.1	501	red	I
8	B	1.2	499	red	II
15	A	1.8	480	green	II
2	C	1.0	511	red	I
7	C	0.4	488	cyan	I
7	A	0.6	491	cyan	I
0	C	0.3	505	blue	II

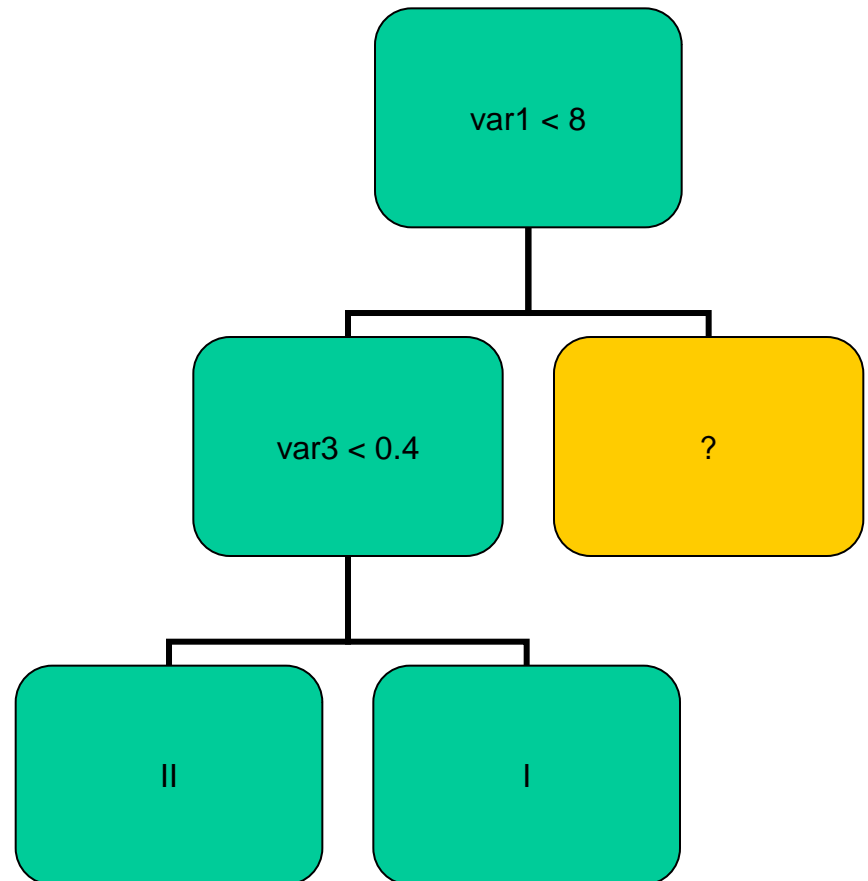




# Example: Tree in Random Forests

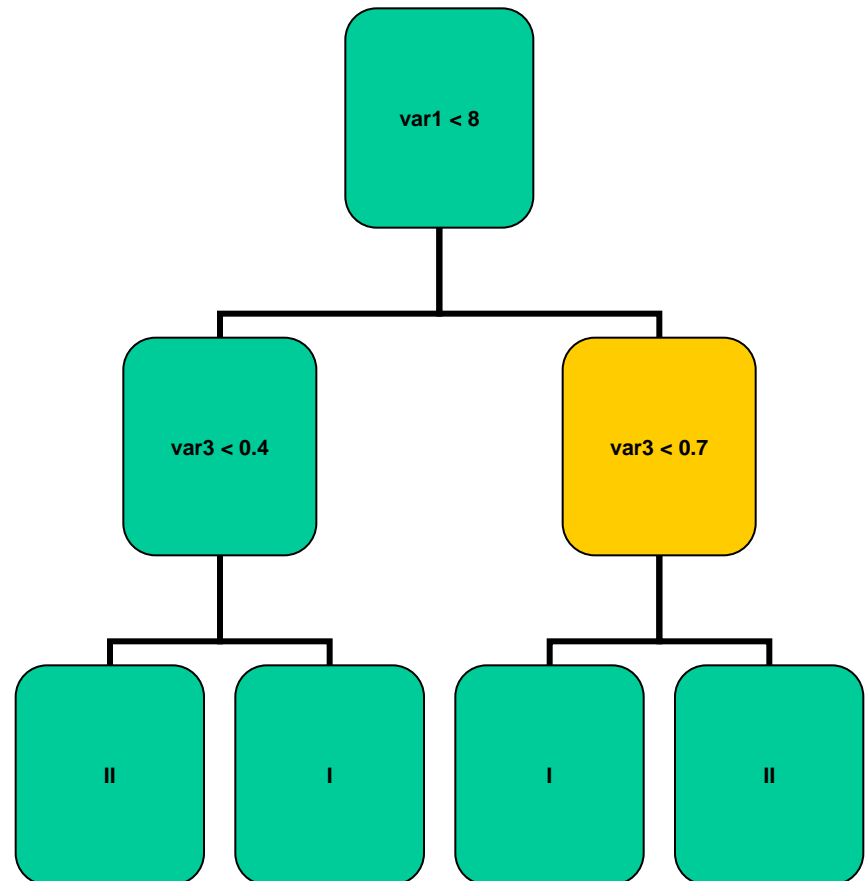
!! Select new attributes at each tree node !!

Input					Output
12	A	0.1	501	red	I
8	B	1.2	499	red	II
15	A	1.8	480	green	II
2	C	1.0	511	red	I
7	C	0.4	488	cyan	I
7	A	0.6	491	cyan	I
0	C	0.3	505	blue	II



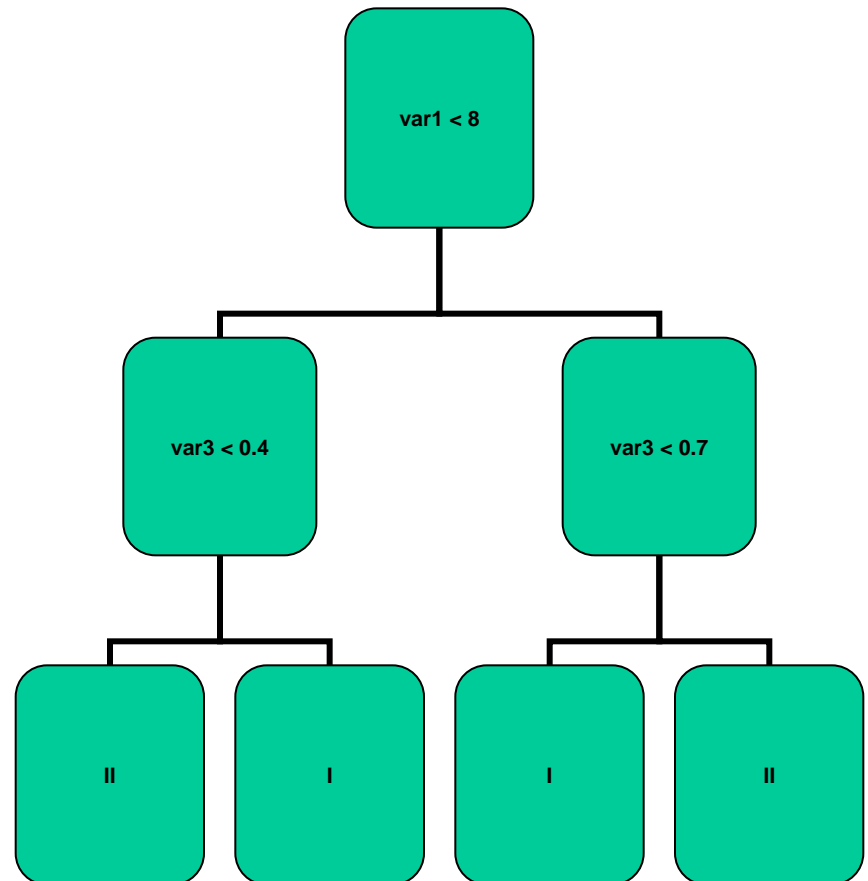
# Example: Tree in Random Forests

Input					Output
12	A	0.1	501	red	I
8	B	1.2	499	red	II
15	A	1.8	480	green	II
2	C	1.0	511	red	I
7	C	0.4	488	cyan	I
7	A	0.6	491	cyan	I
0	C	0.3	505	blue	II



# Example: Tree in Random Forests

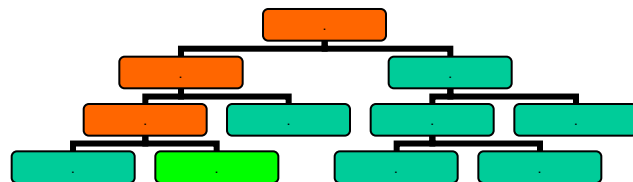
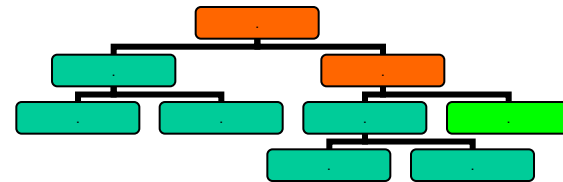
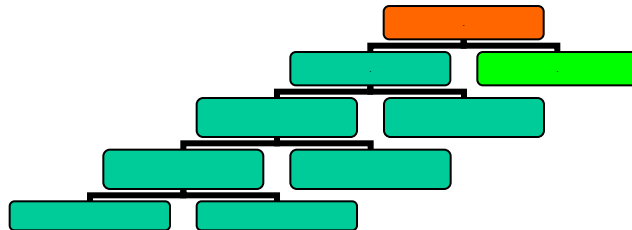
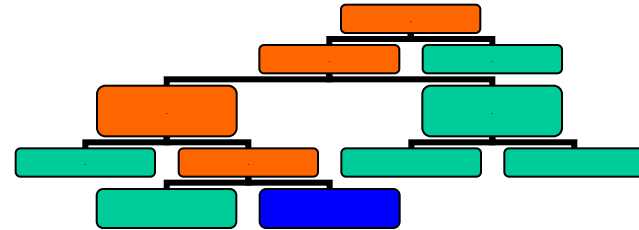
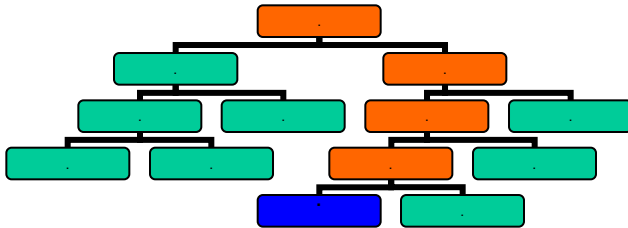
Input					Output
12	A	0.1	501	red	I
8	B	1.2	499	red	II
15	A	1.8	480	green	II
2	C	1.0	511	red	I
7	C	0.4	488	cyan	I
7	A	0.6	491	cyan	I
0	C	0.3	505	blue	II



- Train a number of trees
  - Tens, hundreds – or sometimes even more
- Classify new data by majority voting of the individual trees
  - Count which class is predicted by most trees

# Classification with Random Forests

.....



3 votes for I (Green)  
2 votes for II (Blue)  
=> classify as I

- Only few parameters (number of trees, number of variables for split)
  - Good default values, rather robust
- Still mostly simple concepts
- Very high accuracy for many data sets
- No over-fitting when selecting large number of trees
- Becomes slow with higher number of trees
  - Can be parallelised

- Short recap
- Decision Trees – continued
- Evaluation
- Random Forests
- **Evaluation, continued**
- Data preparation
- SVM, intro

- 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

.....

	<b>Grey</b>	<b>Black</b>	<b>Red</b>	<b><i>Accuracy</i></b>
<b>Grey</b>	5	3	0	<i>0.625</i>
<b>Black</b>	2	3	1	<i>0.500</i>
<b>Red</b>	0	1	12	<i>0.920</i>
				<b><i>0.740</i></b>

- Ideally: numbers only in the diagonal
  - In other cells: indicates misclassification

# Confusion Matrix

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

classified as											genre
a	b	c	d	e	f	g	h	i	j	k	
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											genre
a	b	c	d	e	f	g	h	i	j	k	
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

.....

	<b>BigClass</b>	<b>SmallClass</b>	<b><i>Accuracy</i></b>
<b>BigClass</b>	490	0	100
<b>SmallClass</b>	10	0	0
			0.98

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

# Evaluation – macro average

.....

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

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

- 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



- Cost / loss functions
  - Measures per class with weighted averages
  - Higher weight to classes where errors are more severe
  - ➔ Requires expert knowledge to identify weights

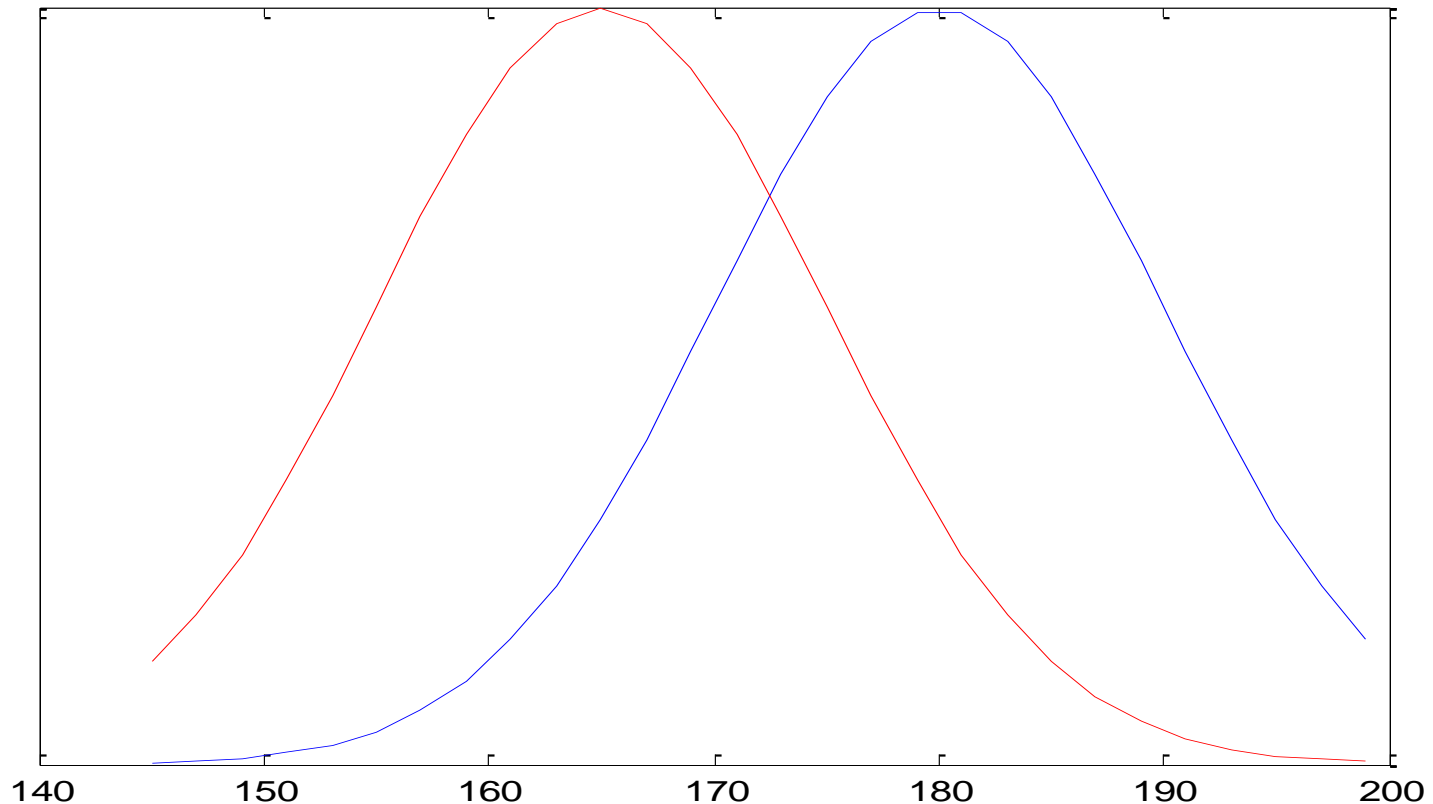
- Effectiveness: quality of classification
  - Accuracy, precision, recall, F1, ...
- Efficiency: computational efficiency (speed, runtime) of a classification
- Performance: often used as *synonym* for either *effectiveness OR efficiency* !

- 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

- Need to know how “good” a classifier is
  - ➔ Train model on (labelled) training data
  - ➔ Test on (labelled) test data
  - ➔ Measure performance on test data
- Performance
  - Several different measures ...

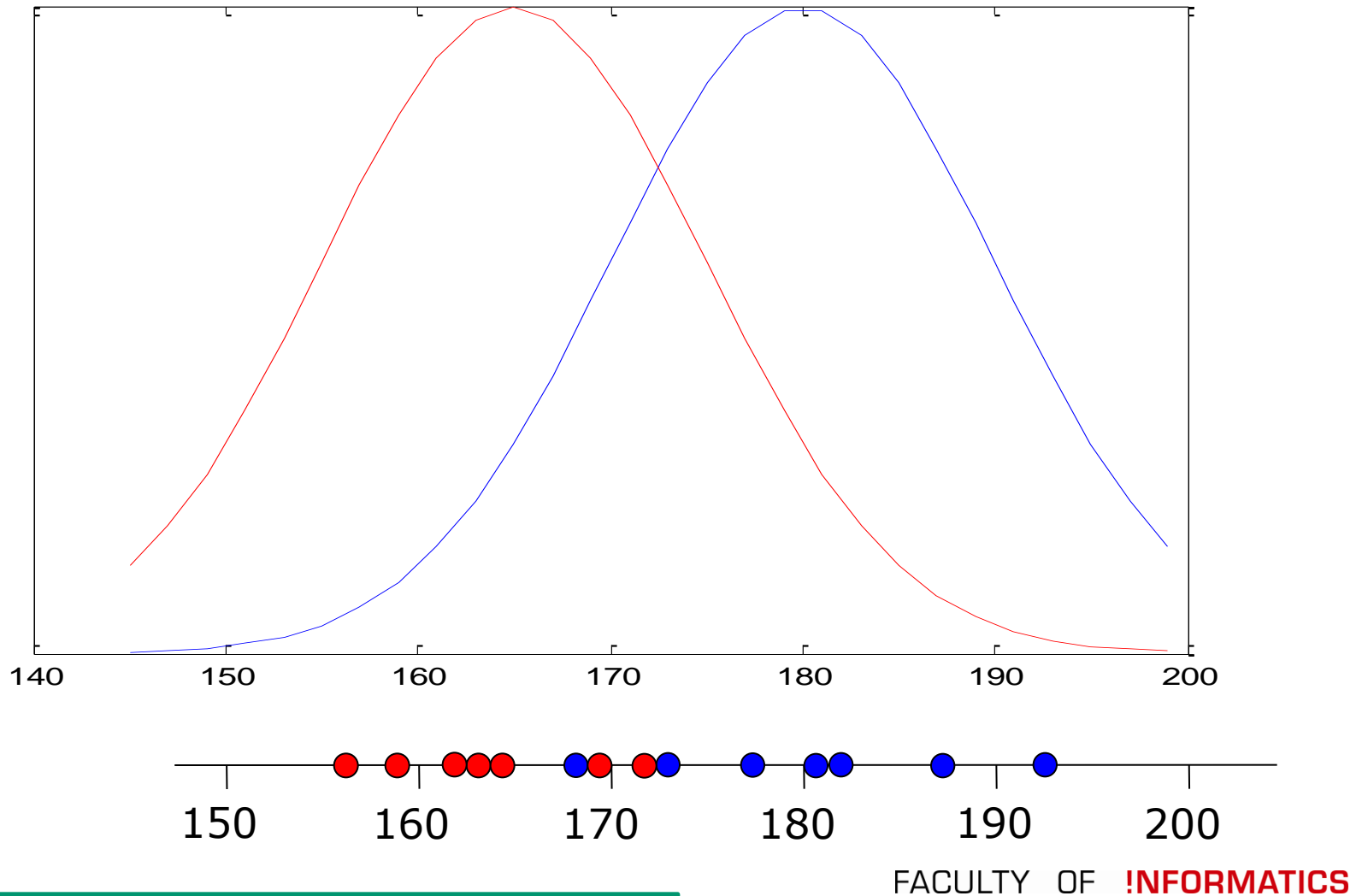
- Overfitting: model is trained too specific to learning examples
  - Examples of classifiers?
- Generalisation: ability of model to perform well on the general problem
  - i.e. the real distribution that generated the training data

- Distributions of two classes (Gaussian, different mean)



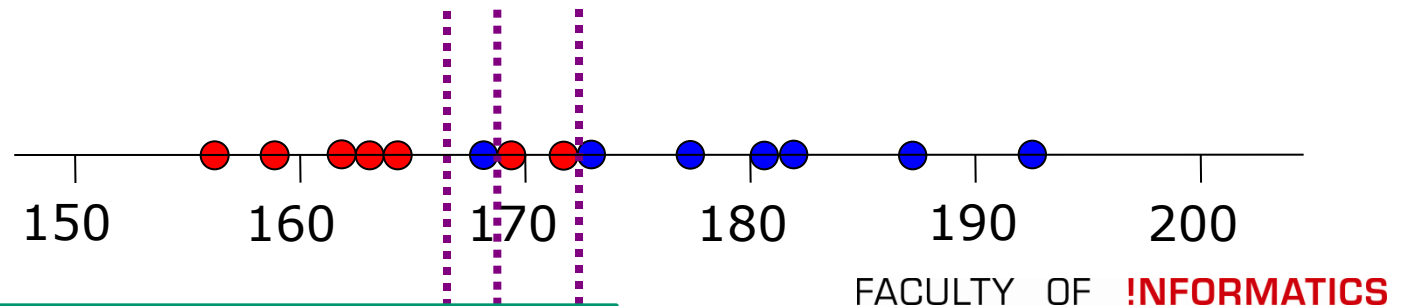
# Overfitting & Generalisation

- Points drawn from that distributions



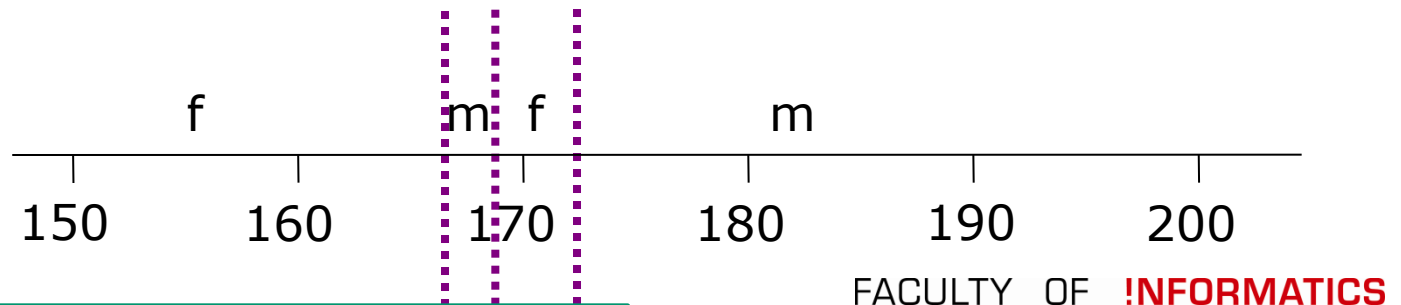
# Overfitting & Generalisation

- Train e.g. a k-nn with  $k=1$
- Assign each point to the closest neighbour  
 → decision boundaries half-way between points of different class

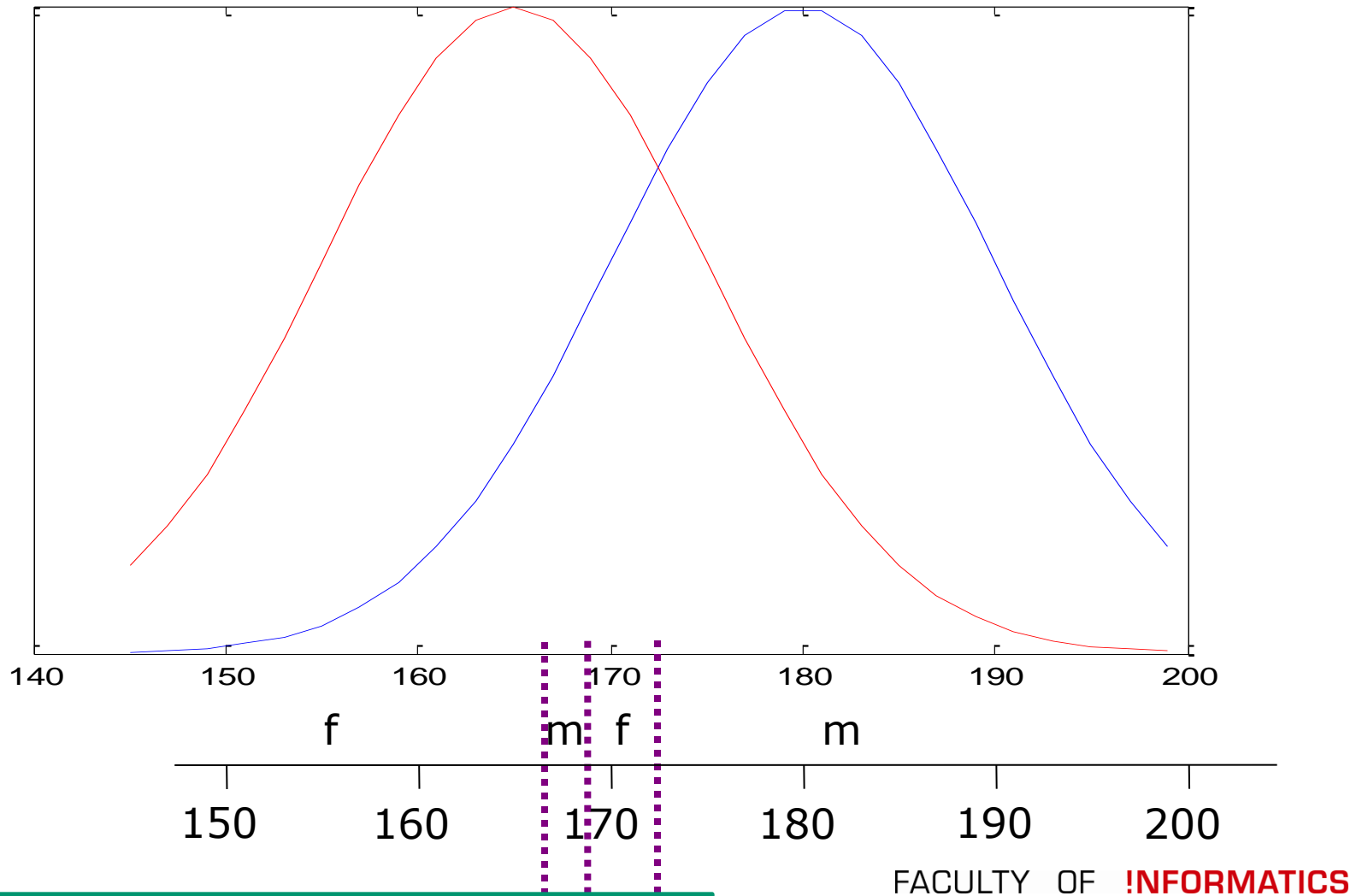




- Train e.g. a k-nn with  $k=1$
- Assign each point to the closest neighbour  
 → decision boundaries half-way between points of different class
- Classify according to closest point

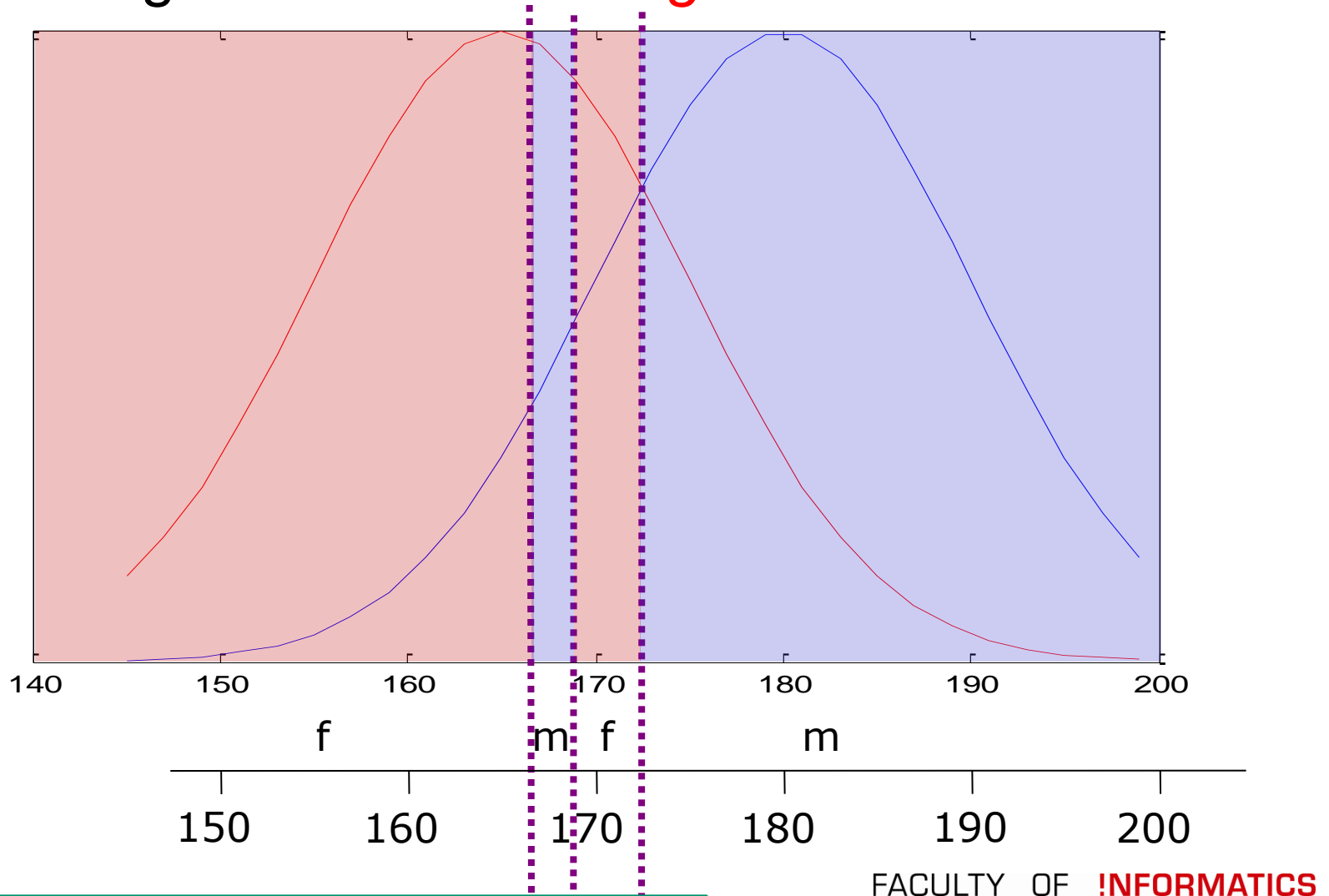


- Train e.g. a k-nn with  $k=1$



# Overfitting & Generalisation

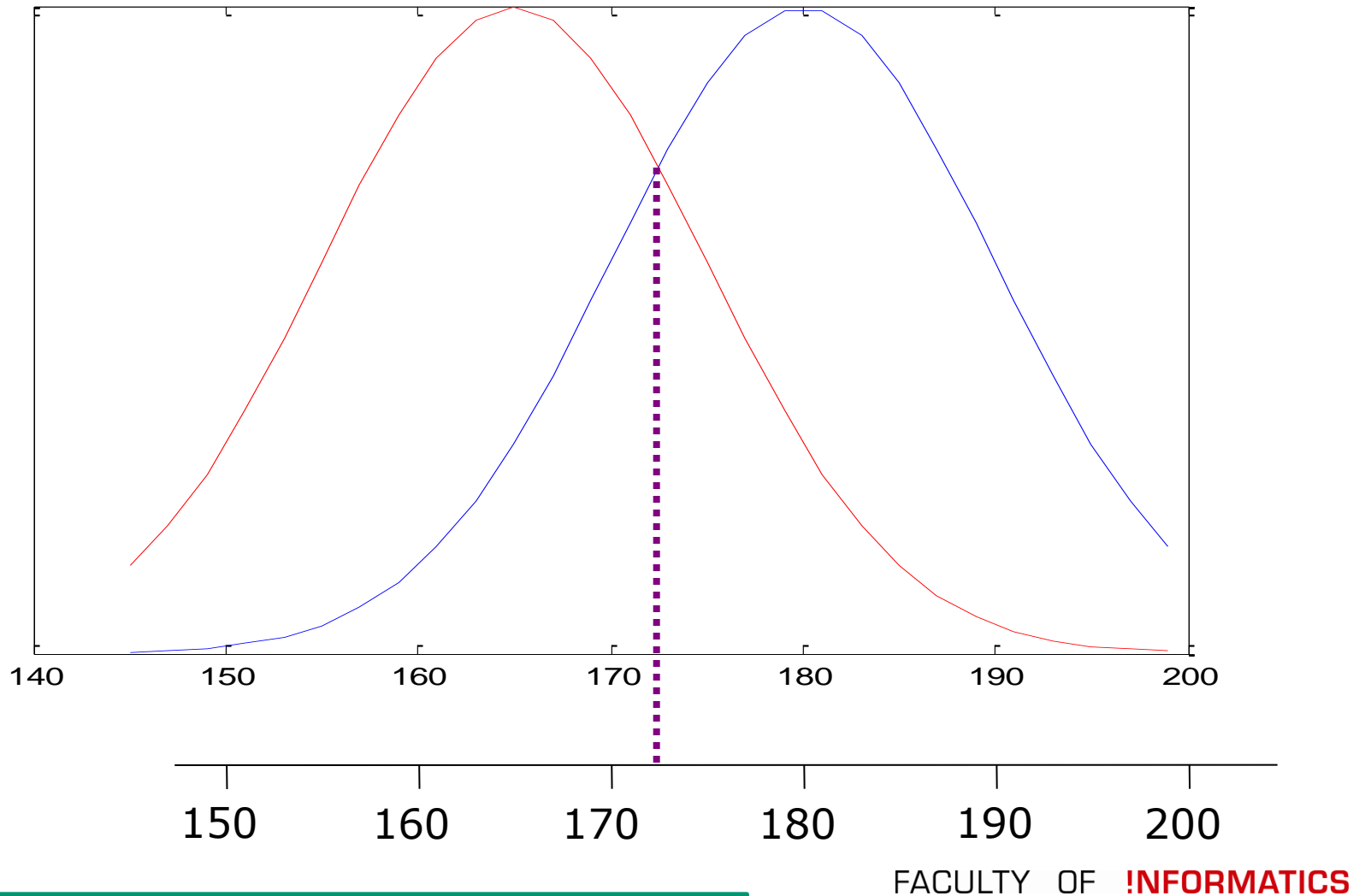
- Train e.g. a k-nn with  $k=1$ : *good classifier?*



- Bayes Optimal Classifier:
  - Simple probabilistic classifier
    - Classification by taking the most likely output value for a given input
    - I.e. the highest probability
  - Probabilities normally not known ...
    - Estimate probability densities based on samples

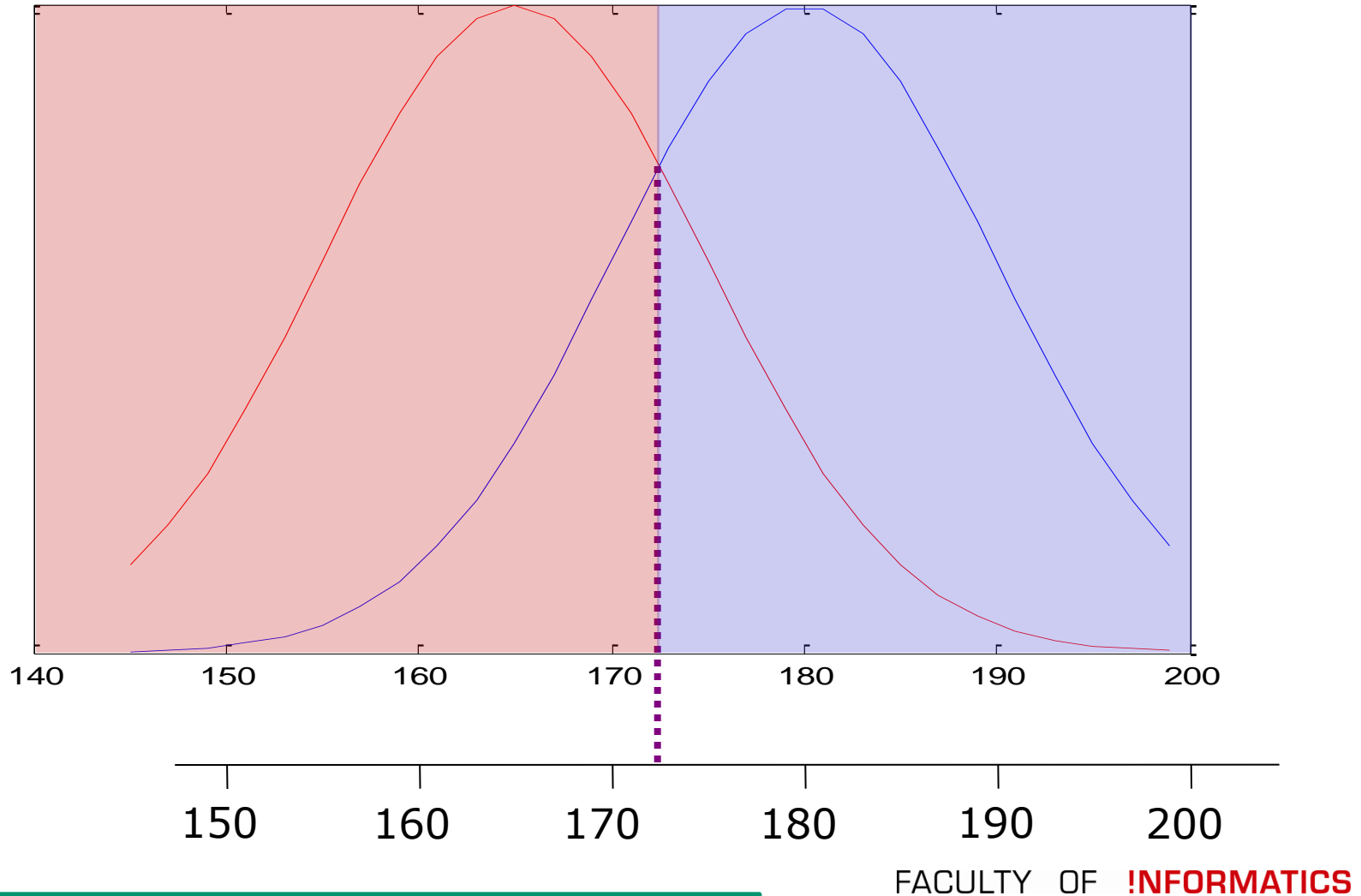
# Overfitting & Generalisation

Bayes optimal classifier: estimation of probability density function

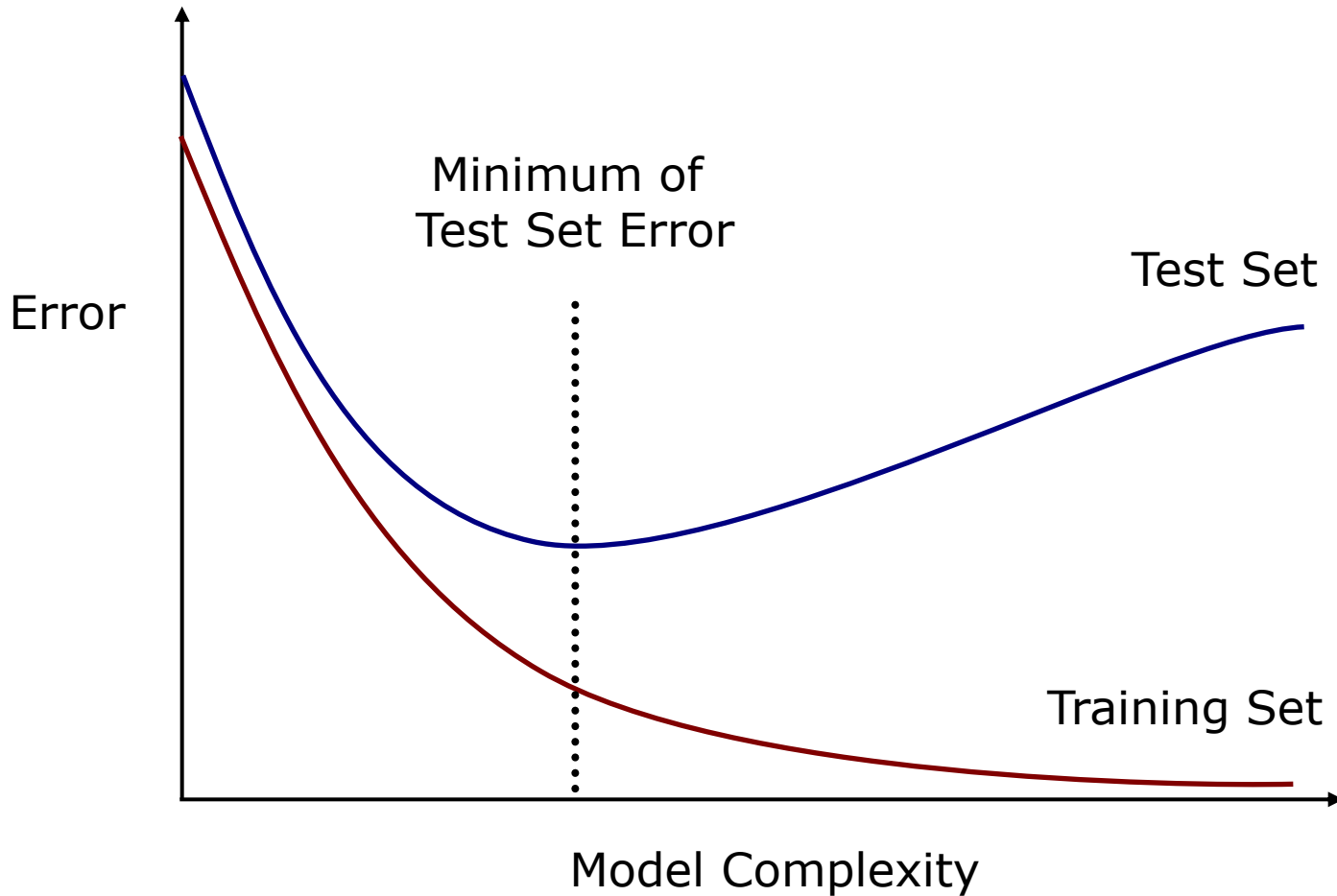


# Overfitting & Generalisation

Bayes optimal classifier: estimation of probability density function



# Trade-off complexity vs. generalization

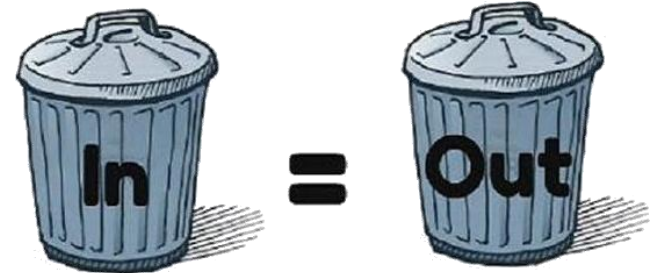


- Table of confusion → accuracy, precision, ...
- Cross-validation
- Bootstrapping
- Micro vs. macro averaging
  - Confusion matrix
  - Cost functions
- Overfitting & Generalisation
- Evaluation measures for regression
- ROC curves, Area under curve, kappa statistic
- Significance testing
- Bias & Variance



- Short recap
- Decision Trees – continued
- Evaluation
- Random Forests
- Evaluation, continued
- **Data preparation**
- SVM, intro

- 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”

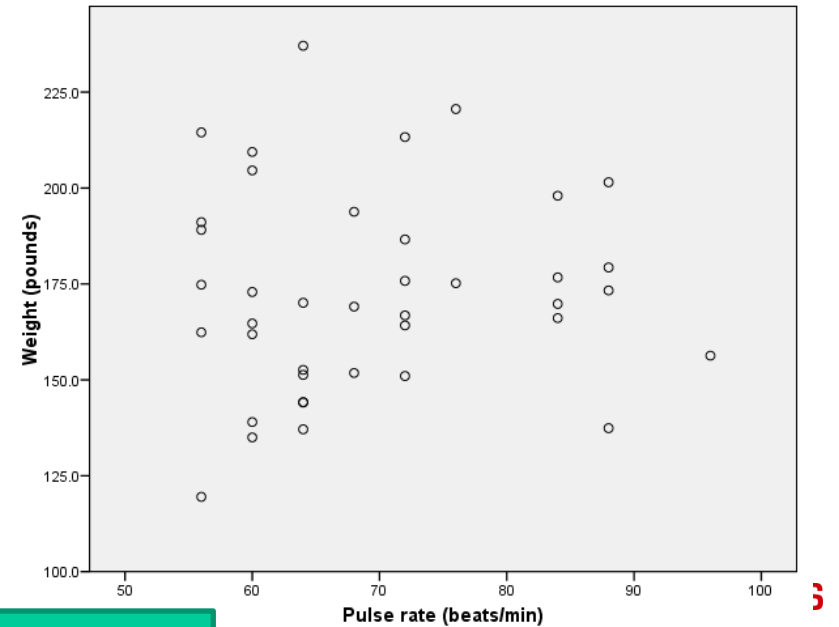
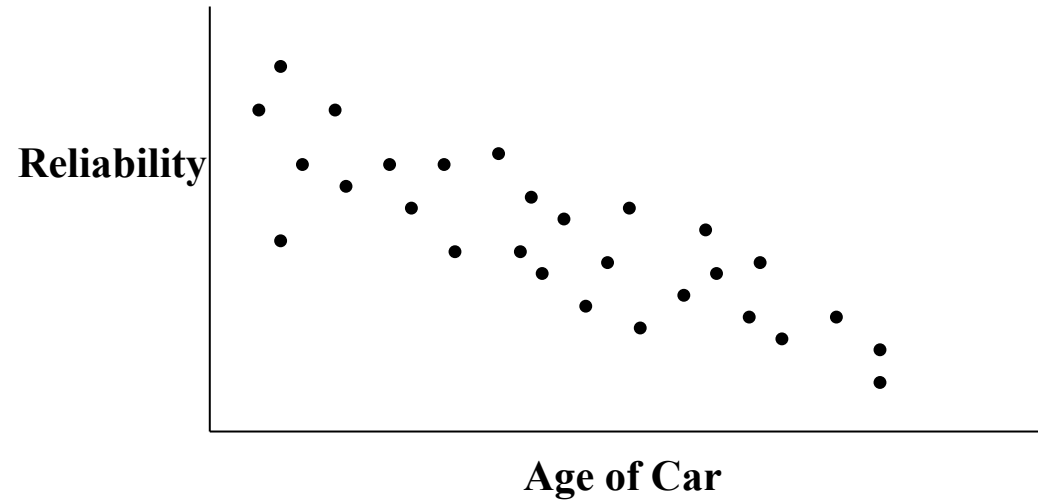
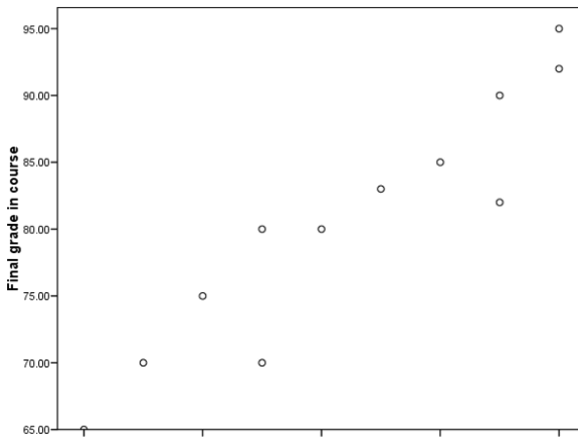
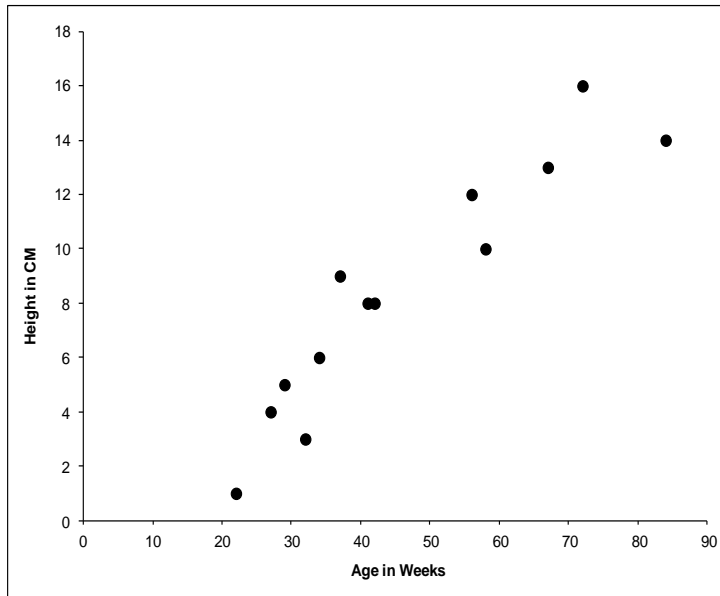


- One major goal of data preparation:
  - Eliminate “wrong influence” of variables
- Already discussed
  - 1-n / one hot encoding, scaling/normalisation

- Data set might contain input variables that directly depend on each other
  - Might have unproportional weight on output prediction
  - Might be beneficial to treat such variables
- ➔ Finding dependencies with (pair wise) analysis of correlation important pre-processing step
- E.g. Pearson correlation coefficient for linear dependence

# Correlation – examples

*direct / indirect / none*

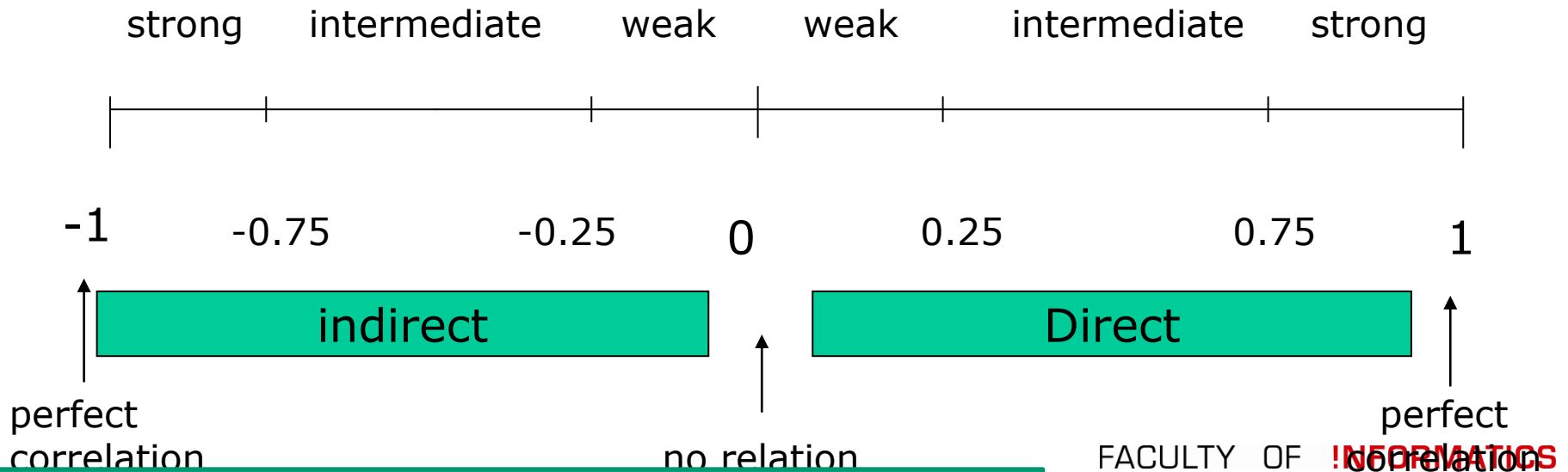


# Pearson Correlation Coefficient

- Denoted as  $r$  or  $\rho$   $r = \frac{\text{cov}(X, Y)}{\sigma_X \sigma_Y} =$

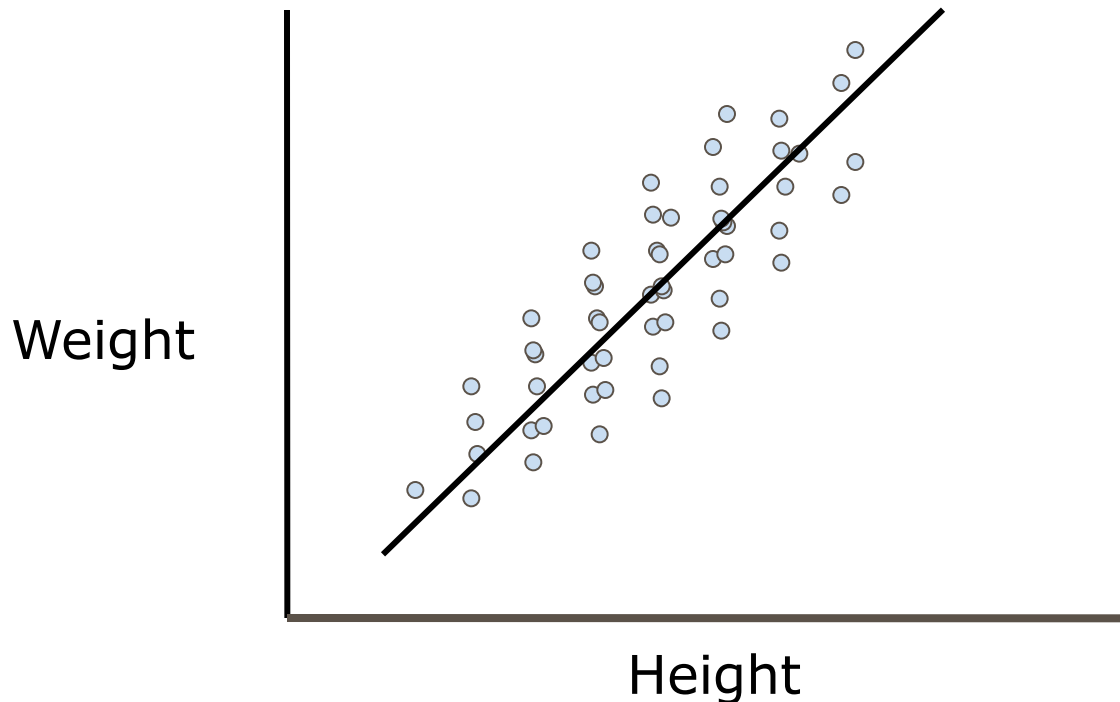
$$= \frac{\sum (x - \bar{x})(y - \bar{y})}{\sqrt{(\sum (x - \bar{x})^2)} \sqrt{(\sum (y - \bar{y})^2)}}$$

$\text{cov}(X, Y)$  = covariance  
 $\sigma(X)$  = standard deviation of X  
 $\bar{x} = \mu_X$  = mean of X



# Correlation Coefficient – examples

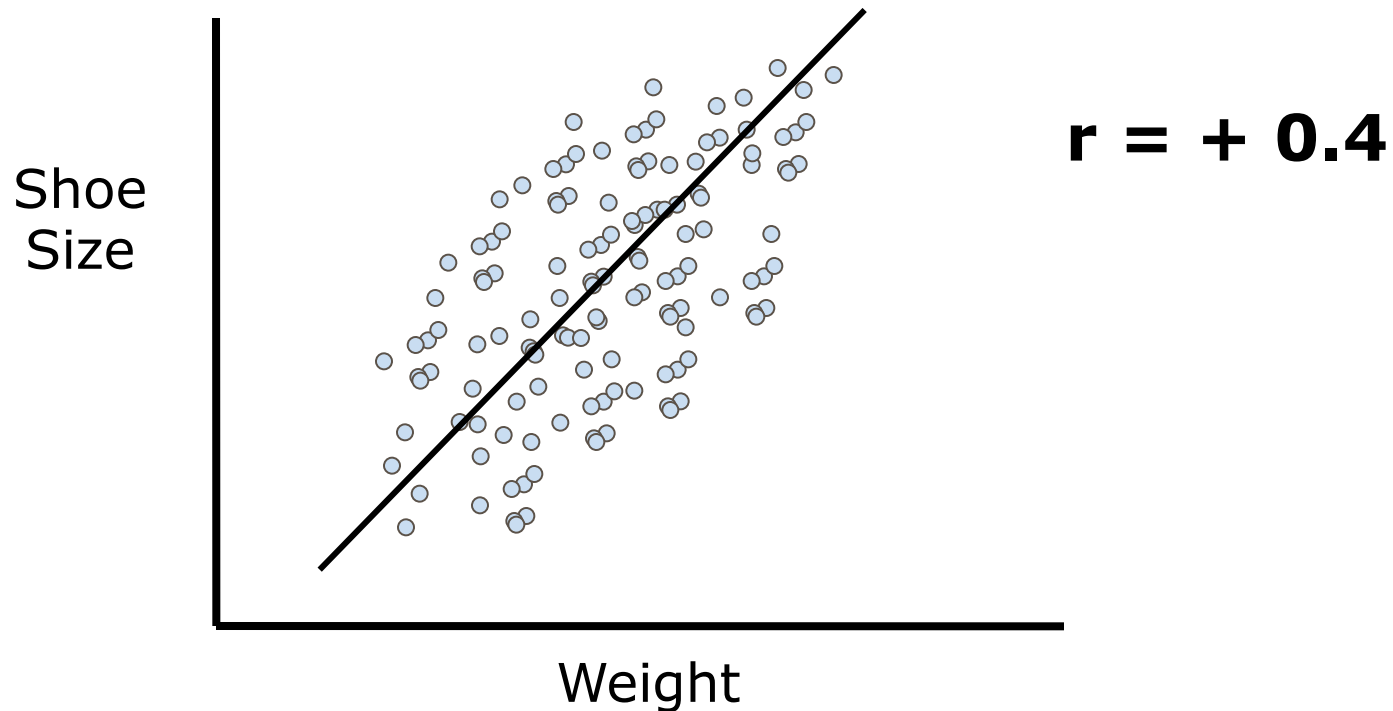
- High Degree of positive (direct) correlation



$$r = +.80$$

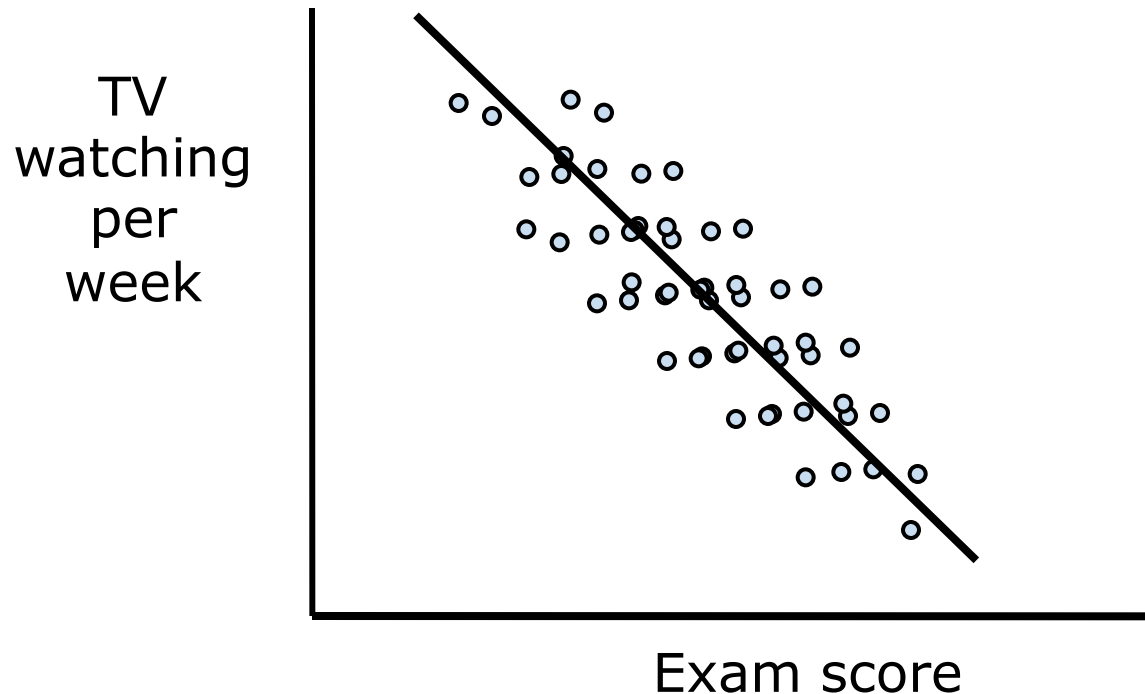
# Correlation Coefficient – examples

- Moderate Positive Correlation



# Correlation Coefficient – examples

- Moderate Negative Correlation

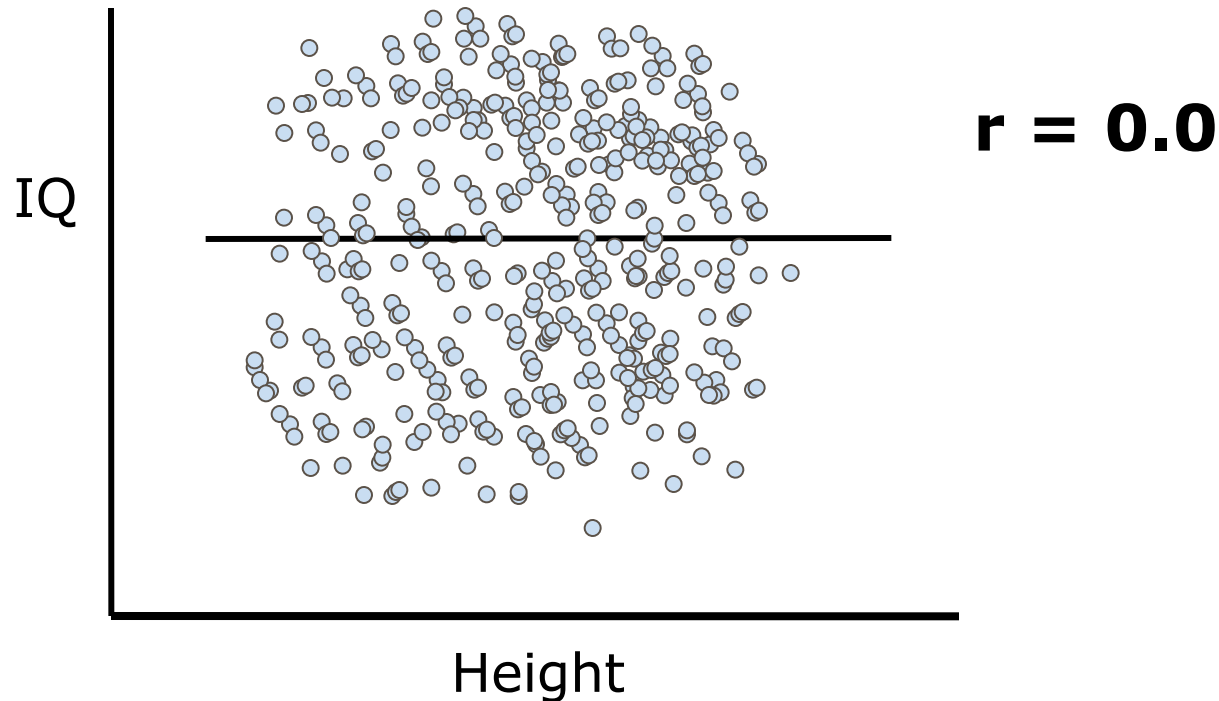


$$r = -.80$$

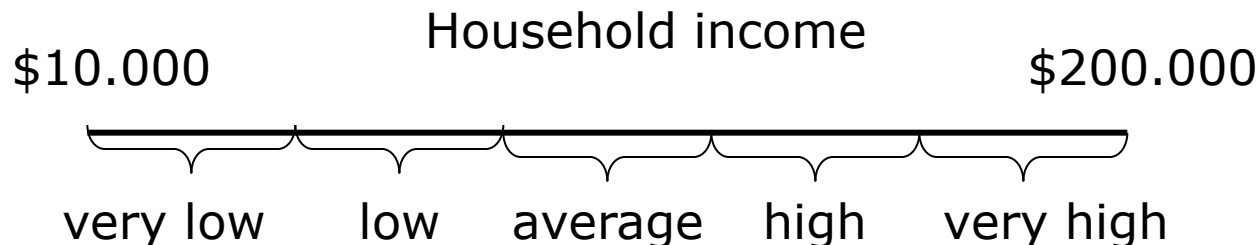


# Correlation Coefficient – examples

- No Correlation



- Classification requires categorical output
  - continuous output = regression
- Classification methods can be applied by discretising output variable
  - Loss of prediction precision
  - More classes → higher precision
    - But also more difficult to learn ...



- For some samples, not all attribute values are known
  - E.g. not captured by typist, broken sensors, change in capturing method/experiment design, ...
- Some ML algorithms can handle missing values
  - k-NN: skip distance computation for attributes that have missing value
  - *Naive Bayes: omit values from probability computation*

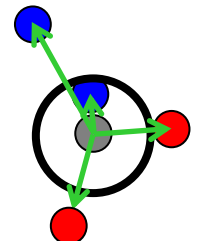
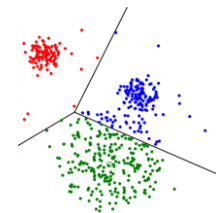
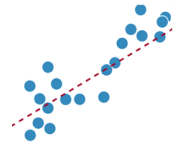
gender	age	smoker	eye colour	distance
male	19	yes	green	-
female	44	yes	grey	2 + 25
male	77	yes	grey	1 + 68
male	21	yes	green	0 + 2



gender	age	smoker	eye colour	distance
male	19	yes	green	-
?	44	yes	grey	1 + 25
male	77	yes	?	0 + 68
male	21	yes	green	0 + 2

- For some samples, not all attribute values are known
- Some ML algorithms can handle missing values
- Solutions for other algorithms
  - Deletion of sample
    - Bad when only few labelled samples
    - Not always an option when you have missing values in the test set... !
  - Imputation
    - Substitute missing value

- Substitution of a missing value
- Different methods
  - Mean value of the attribute (*computed from other samples*)
  - Random selection of value from another sample
  - Regression – using other attributes to predict
  - Clustering – values of cluster centroid
  - Nearest Neighbour – value of closest sample (*computed by similarity of other attributes*)



- *When is imputation useful?*
- Most useful when the number of labelled samples (w/o missing values) is small
  - *Relatively easy to identify*
- When samples with missing values contain otherwise important information
  - *Difficult to identify*

- How to test for the effectiveness of data imputation?
- Example experiment setup:
  - Experiment #1
    1. Delete samples with missing values
    2. Split into training & test set
    3. Compute performance on test set
  - Experiment #2
    1. Perform data imputation
    2. Split into training & test set
    3. Compute performance on test set
  - *Compare results from Exp #1 & #2?*

gender	age	smoker	eye colour
male	19	no	green
?	44	yes	grey
male	77	no	?
male	21	yes	green
female	51	?	green
female	81	yes	grey

gender	age	smoker	eye colour
male	19	no	green
male	21	yes	green
female	81	yes	grey

Train

Test

gender	age	smoker	eye colour
male	19	no	green
male	44	yes	grey
male	77	no	green
male	21	yes	green
female	51	yes	green
female	81	yes	grey

Train

Test

# Data Imputation

gender	age	smoker	eye colour
male	19	no	green
?	44	yes	grey
male	77	no	?
male	21	yes	green
female	51	?	green
female	81	yes	grey

Train

Test

gender	age	smoker	eye colour
Male	77	No	?
Male	21	Yes	green
Female	81	Yes	grey

gender	age	smoker	eye colour
male	19	no	green
?	44	yes	grey
female	51	?	green

- Experiment setup:

- Split data into training & test set

- Experiment #1

1. Delete samples with missing values in training set?
2. Compute performance on test set?

- Experiment #2

1. Perform data imputation
2. Compute performance on test set

- Compare results from Exp #1 & #2!

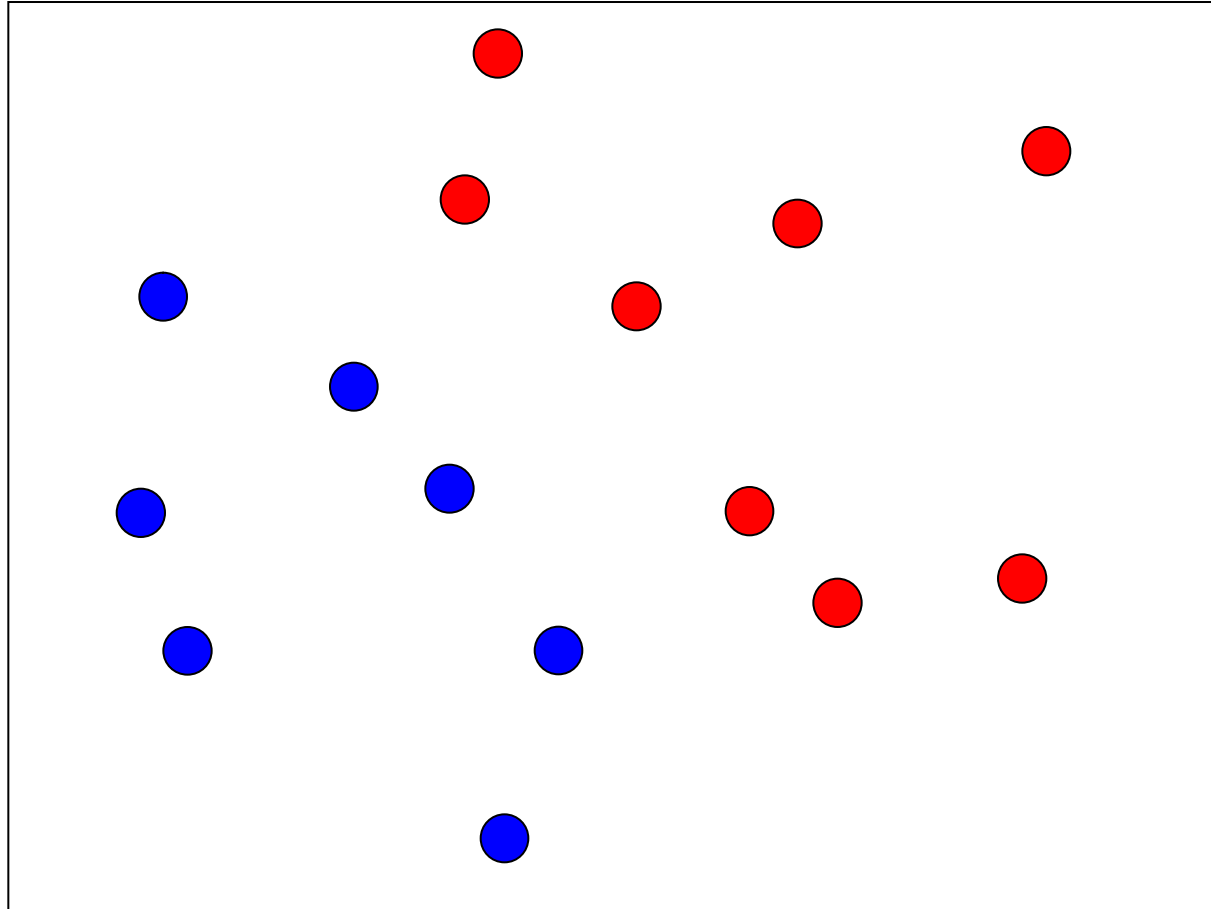


- Short recap
- Decision Trees – continued
- Evaluation
- Random Forests
- Evaluation, continued
- Data preparation
- SVM, intro

- Concept introduced by Vladimir Vapnik in 1960s
  - Heavily used & researched in last decade(s)
- Rather sophisticated mathematical model
- Also known as *maximum margin* classifier
- Basic concepts
  - Linear separation
  - Optimisation of hyperplane
  - Soft margin & kernel function
    - When linear separation is not possible

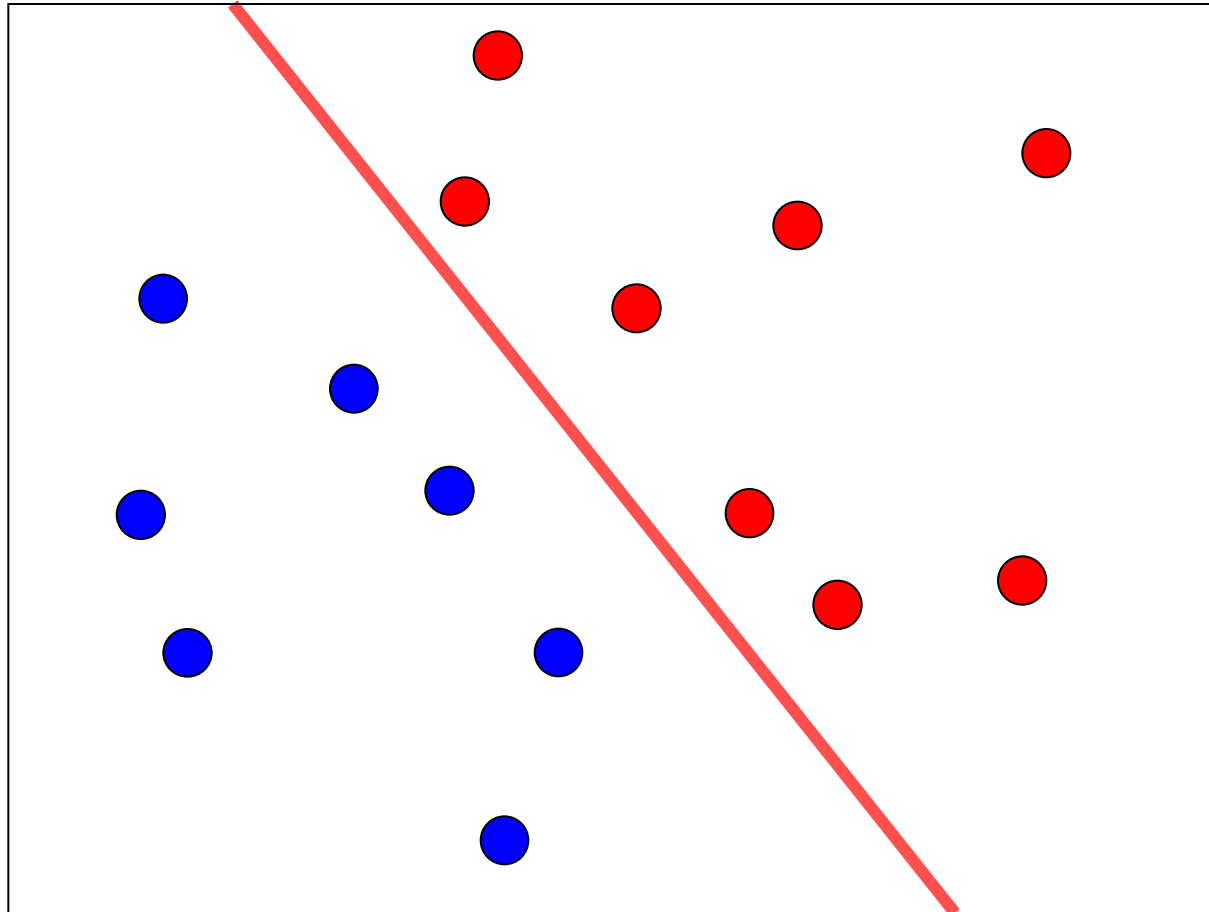
# Linear separation

.....



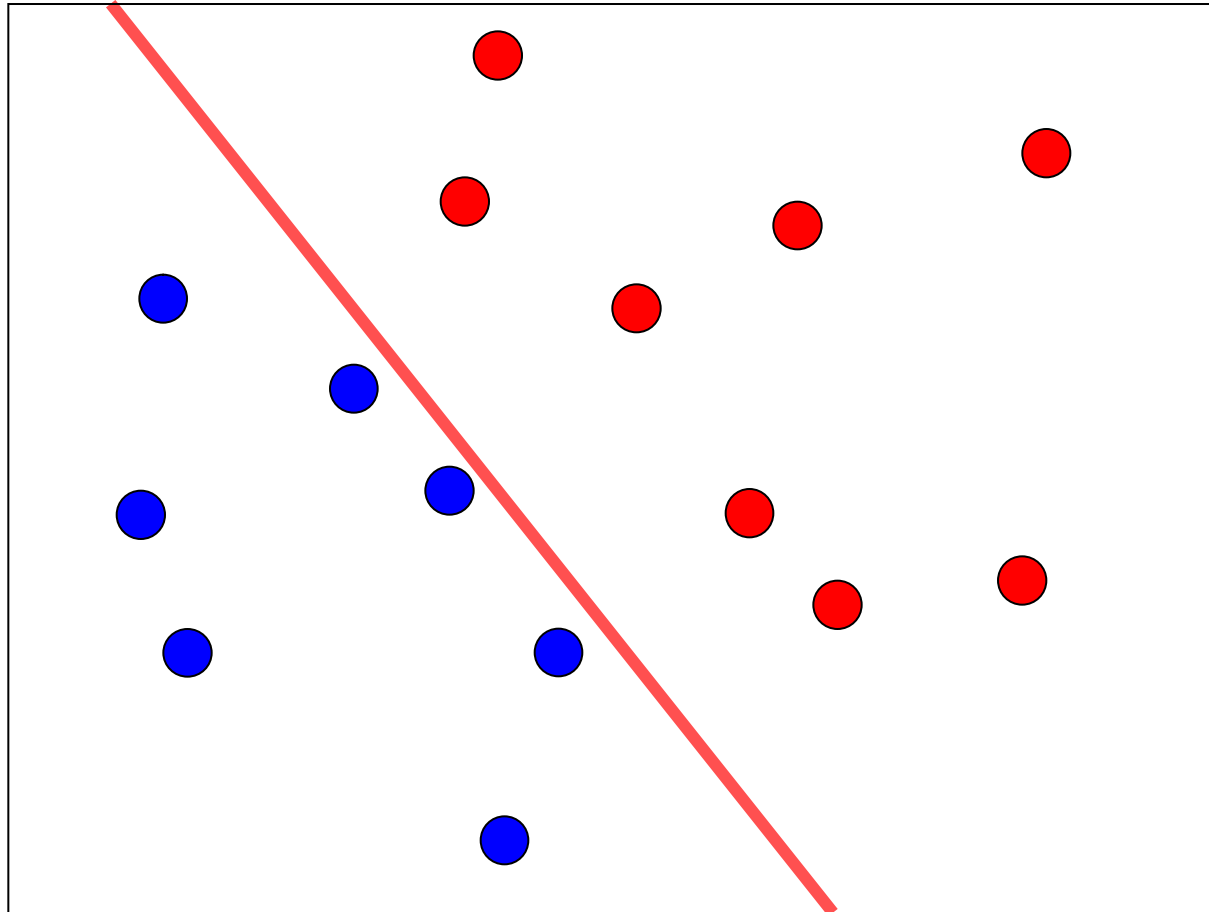
# Linear separation

- Separating line #1



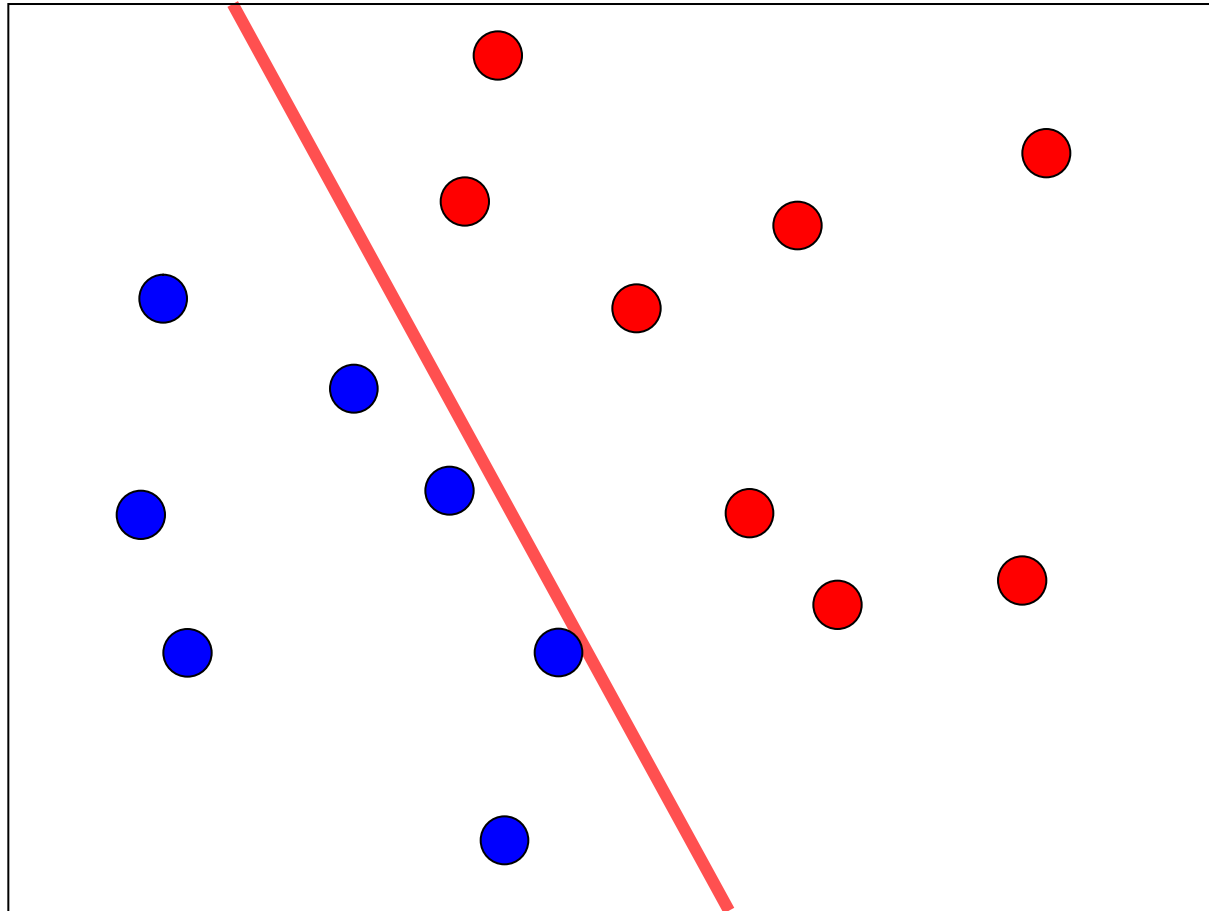
# Linear separation

- Separating line #2



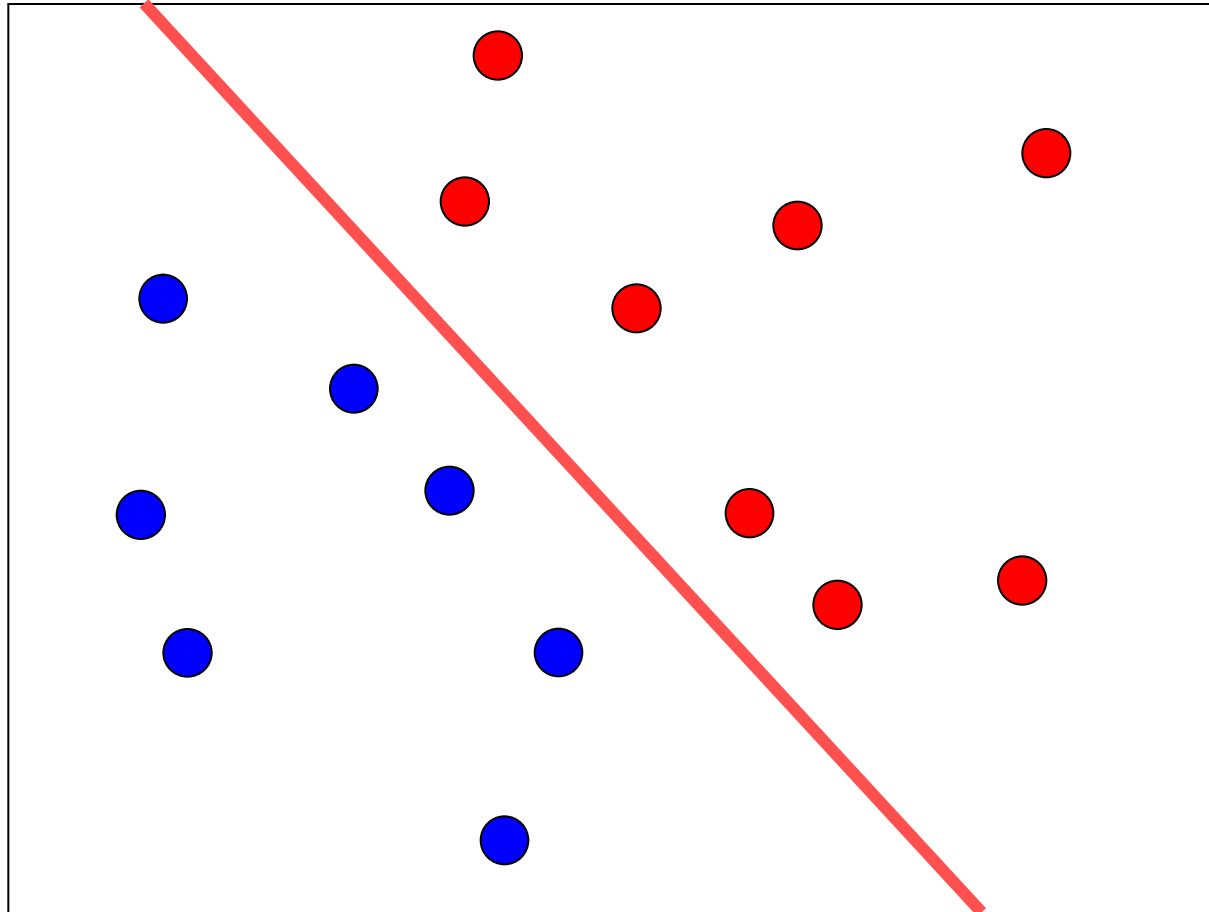
# Linear separation

- Separating line #3



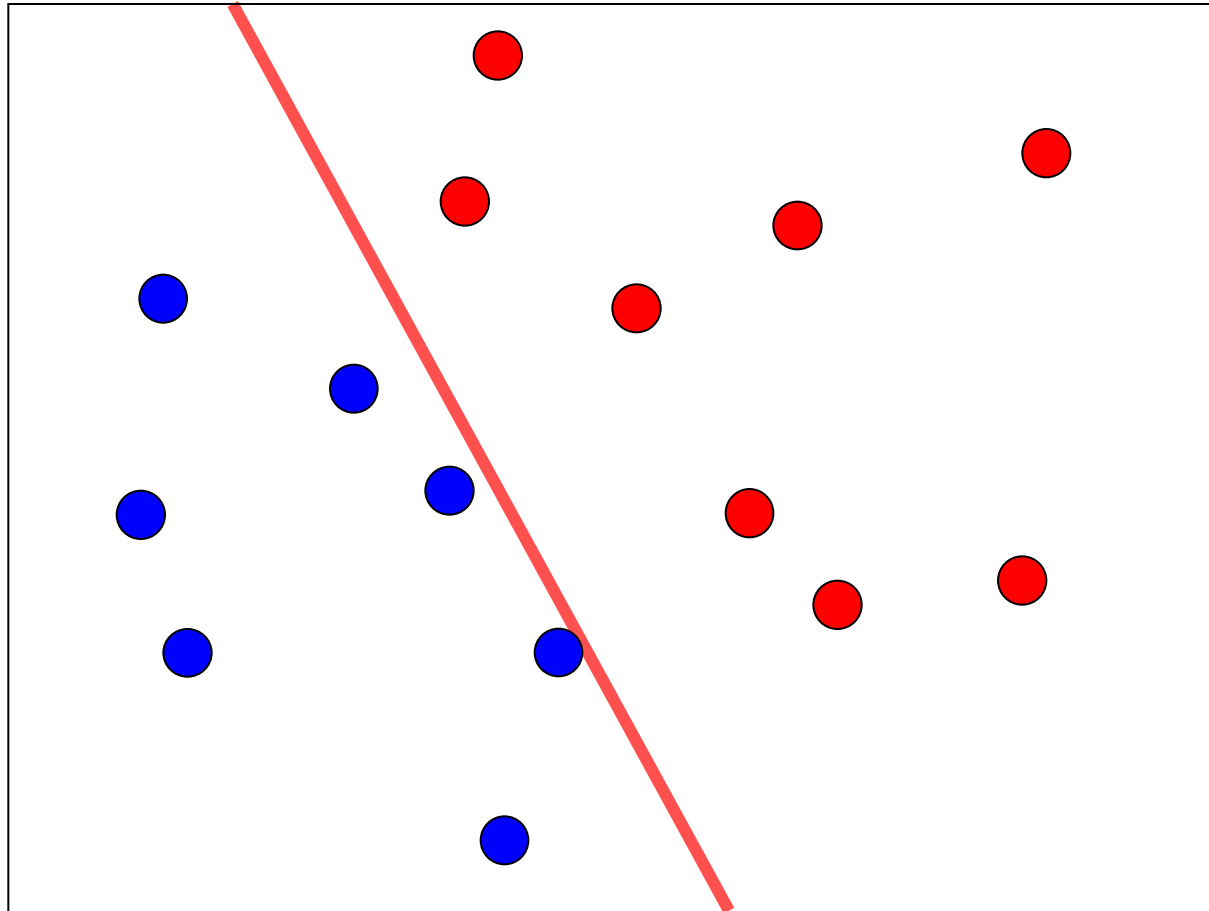
# Linear separation

- Separating line #4



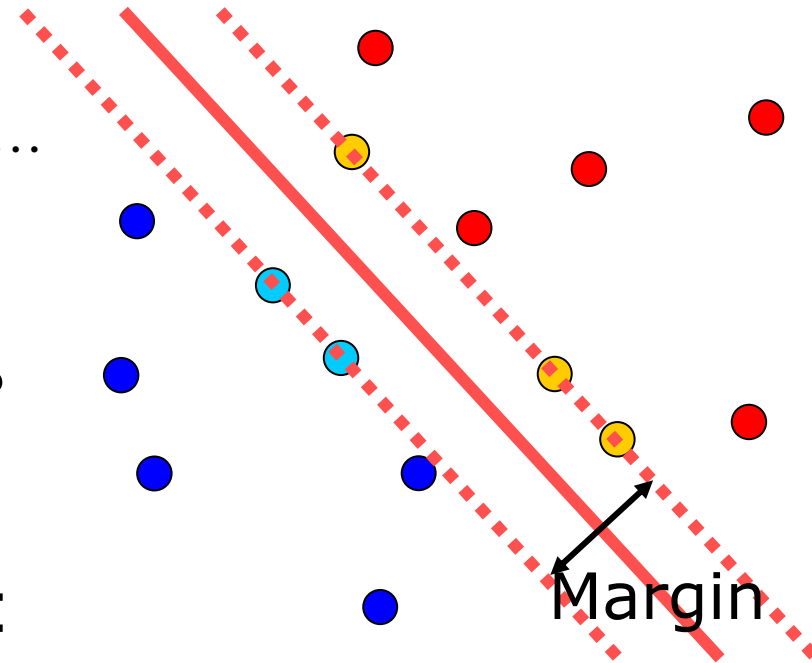
# Linear separation

- Separating line #5





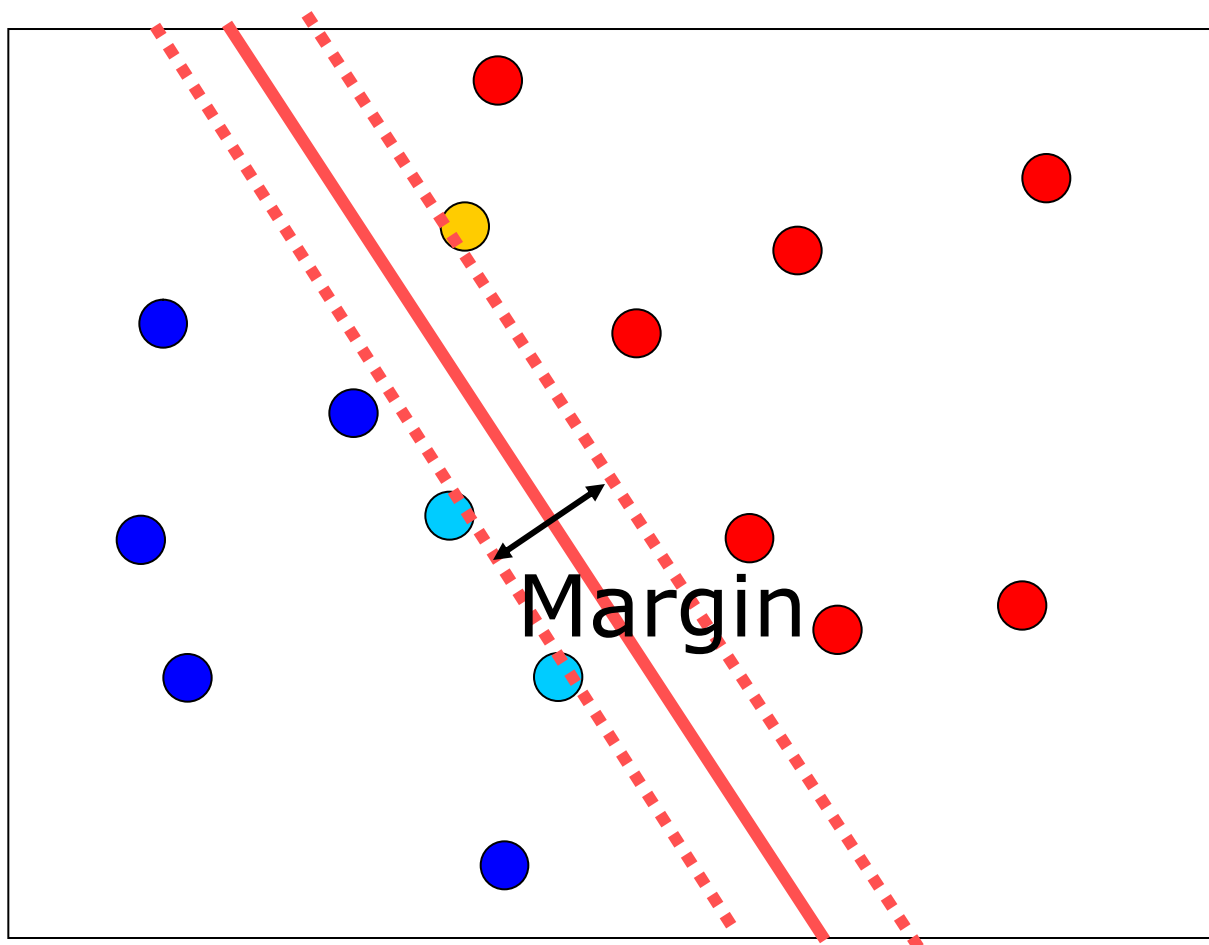
- *Which other classifier(s) use linear separation?*
- *What's the difference to SVMs?*
- Optimization
  - maximisation of margin separating items
- *Later: soft margin, kernels, ...*



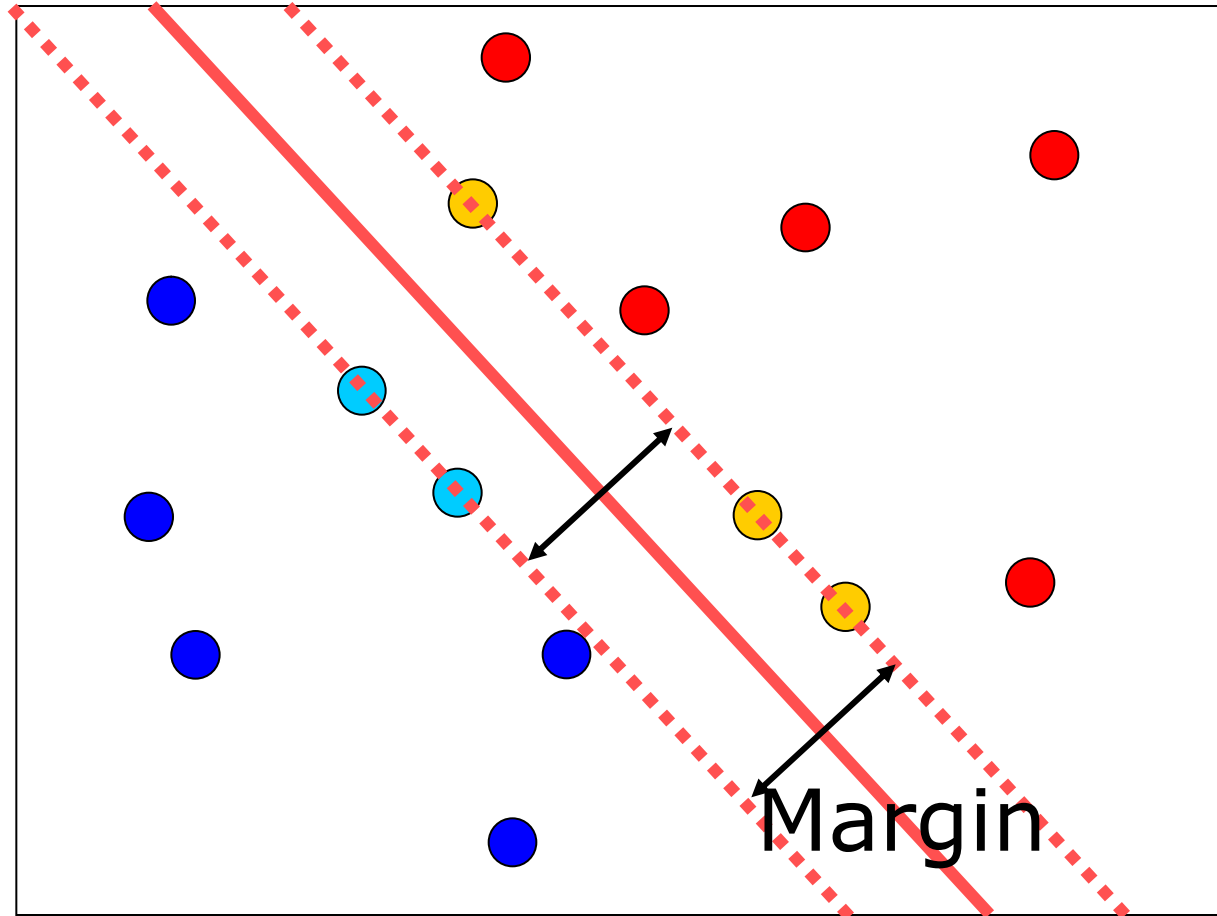
- All separations are valid
  - Which separation is the best?
- **Margin** of a linear classifier: width that boundary could be increased to
  - before hitting any data-point
- **Support Vectors** are those data-points that the margin pushes up against
- *What's the minimum number of support vectors?*

# Largest Margin

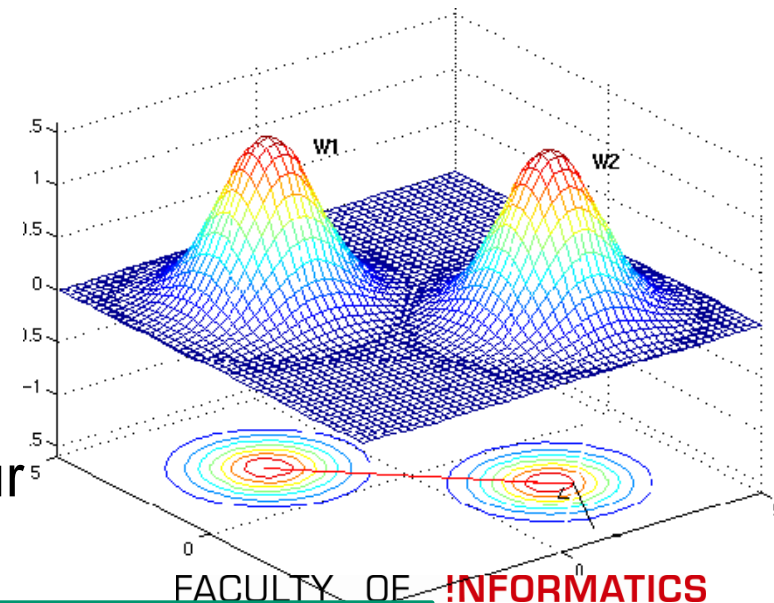
.....



# Largest Margin

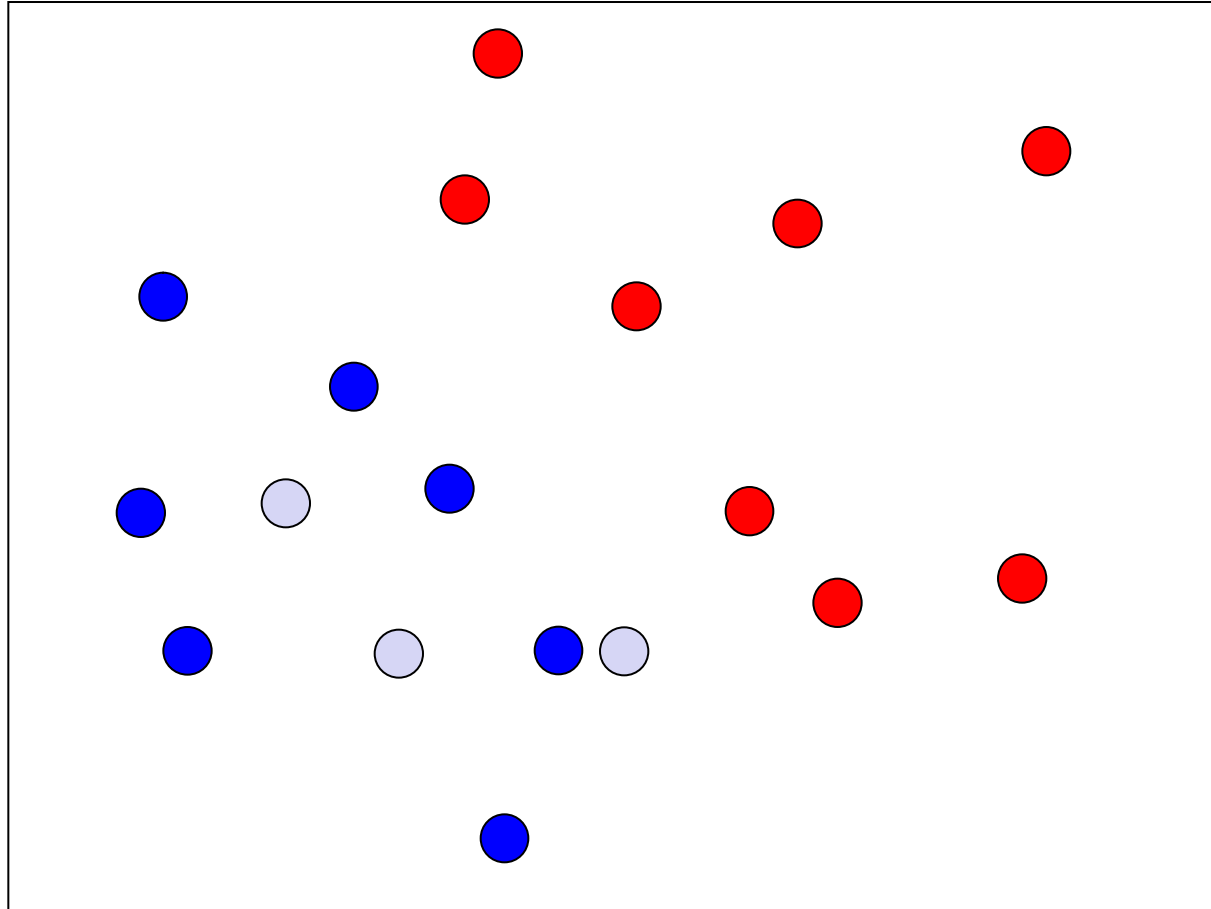


- *Which separation/margin is the best?*
- Claim: **bigger margin is better**
- Intuitive illustration example
  - Assumption in previous dataset: samples are drawn from probability distribution
  - E.g. two Gaussians with different means (& variances)
  - Now, draw more samples from these distributions to increase our data set (training/testing)



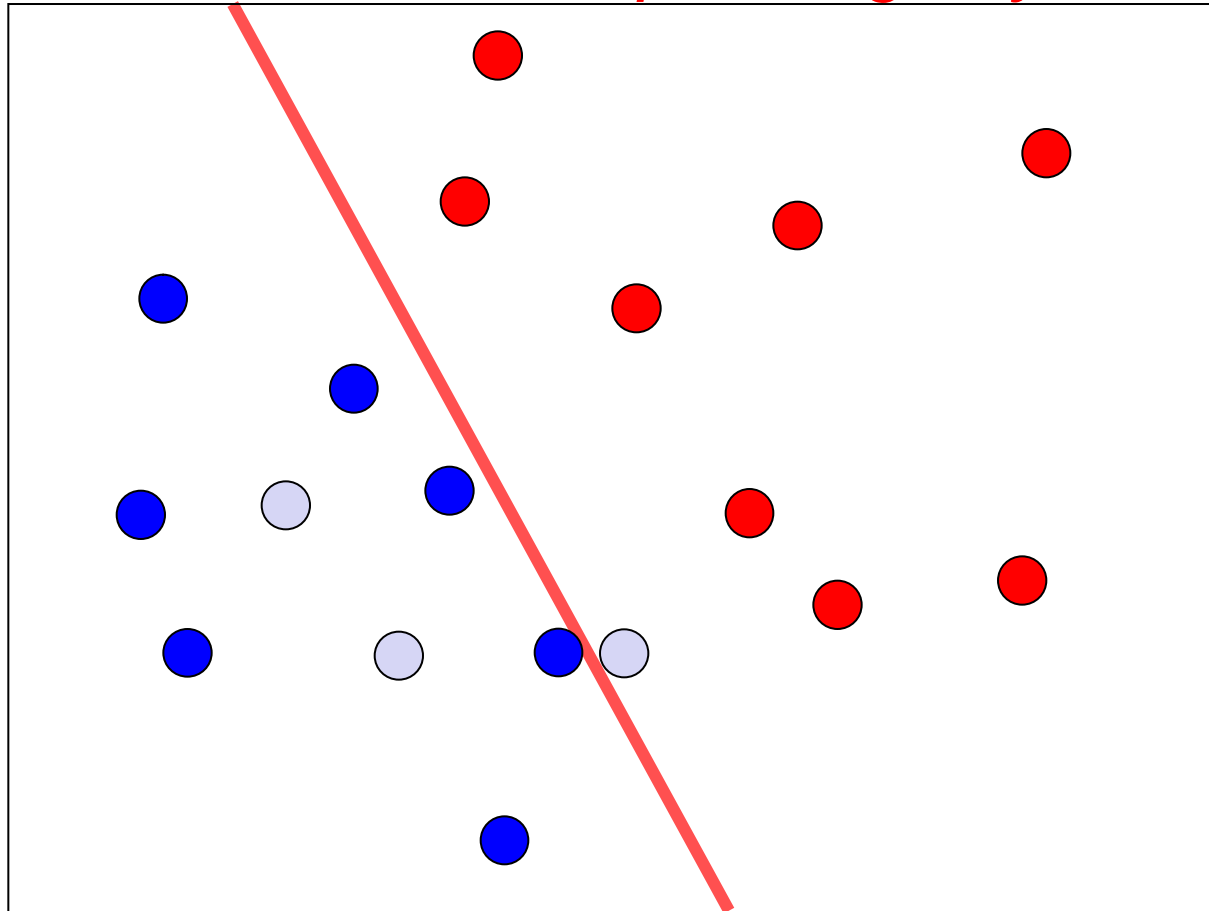
# Linear separation

- Draw more samples from the distribution



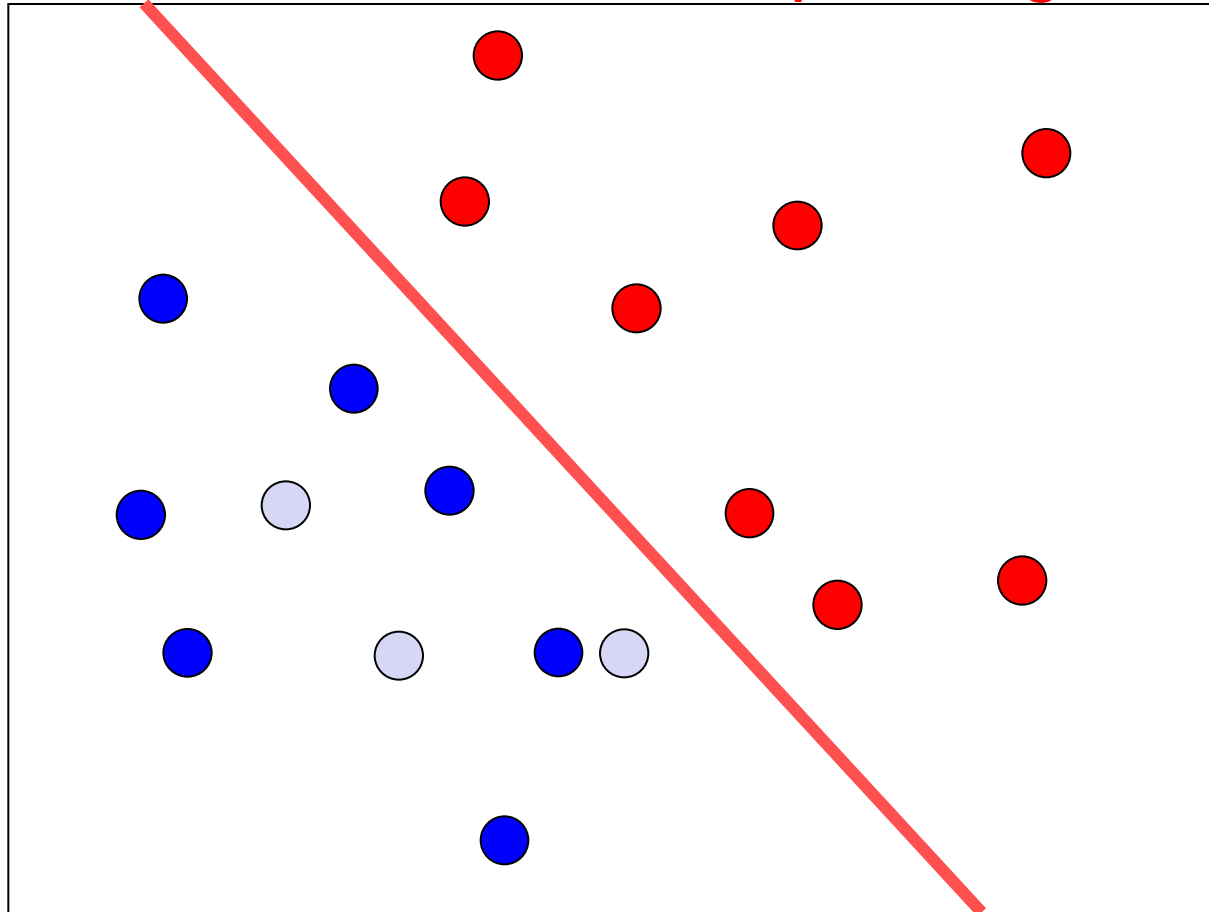
# Linear separation

- Draw more samples from the distribution  
 → *Line #3 not separating anymore*



# Linear separation

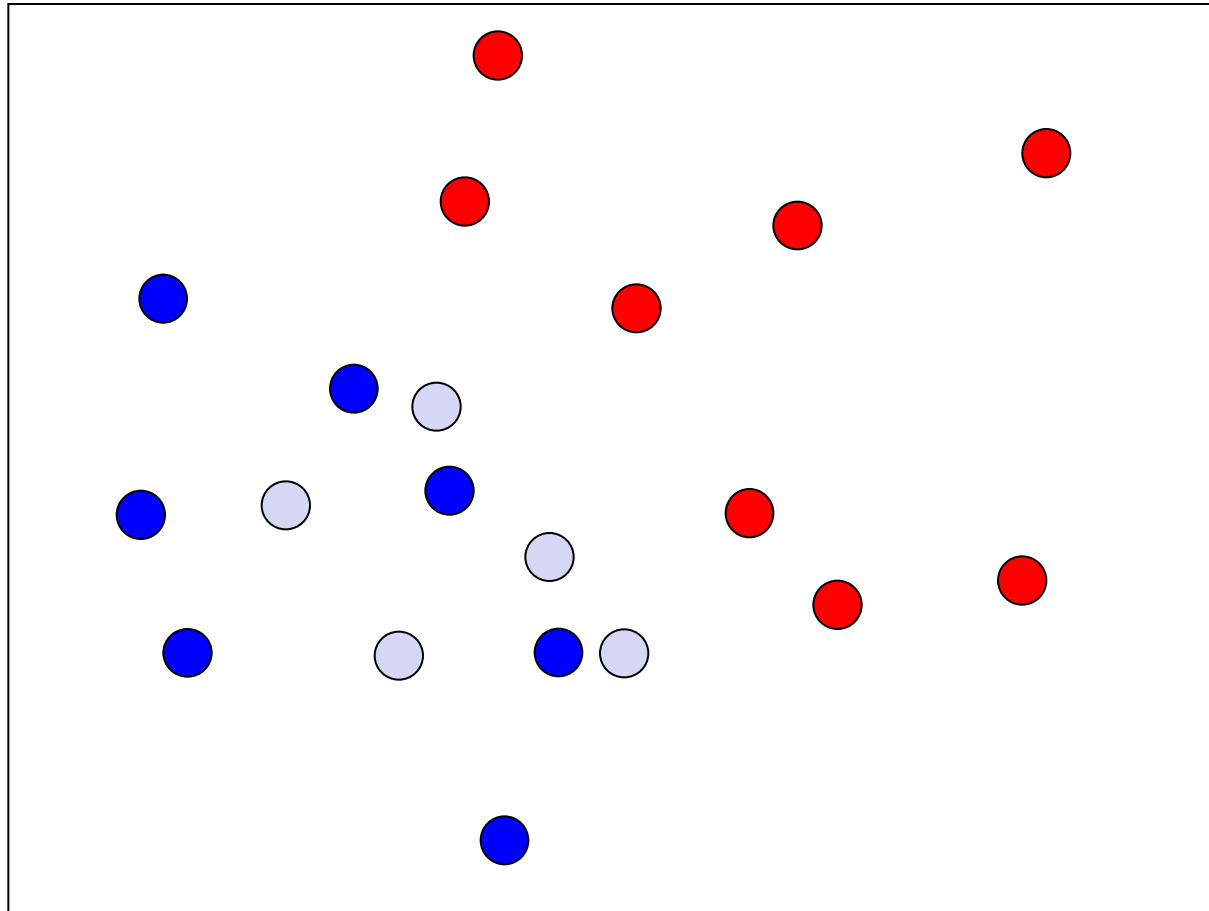
- Draw more samples from the distribution  
 → *Line #4 still separating*





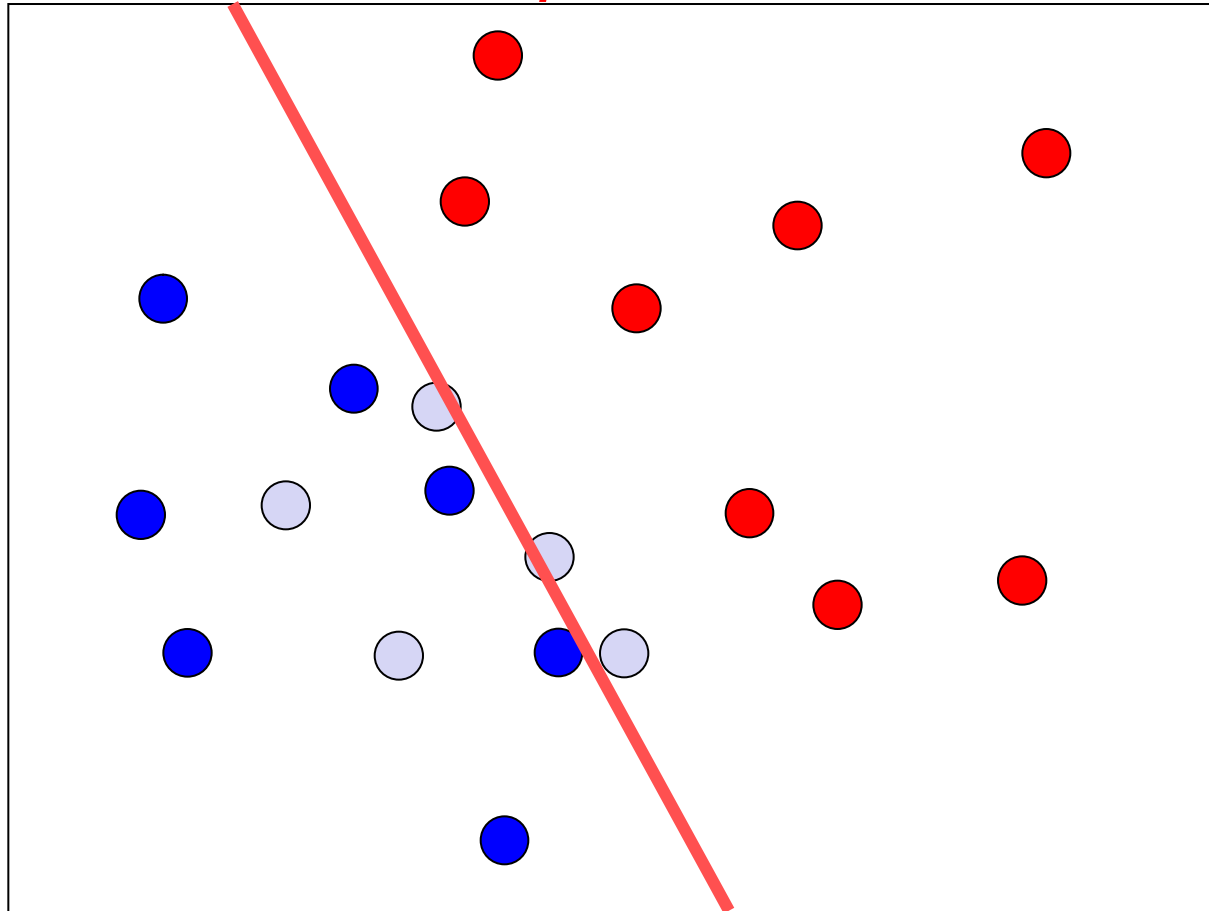
# Linear separation

- Draw even more samples from the distribution



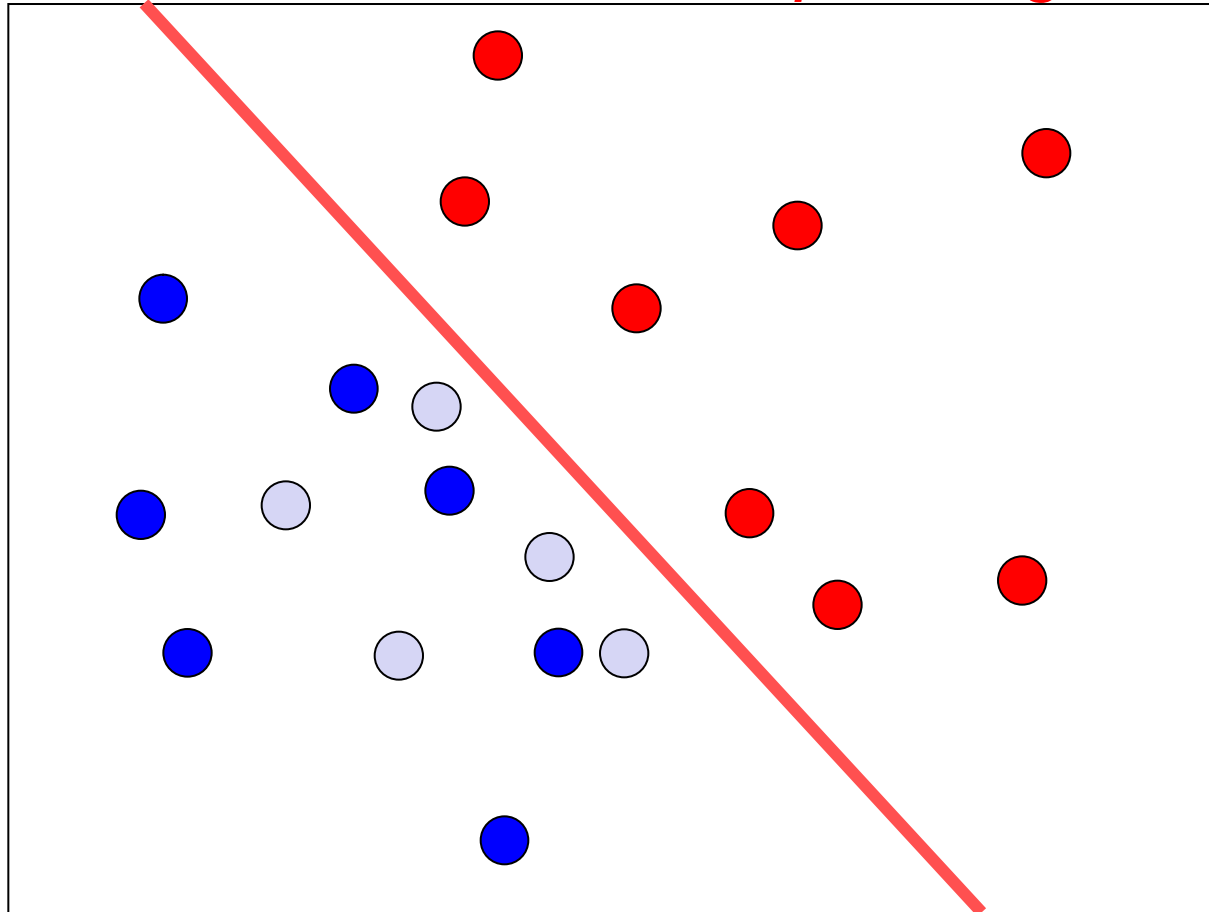
# Linear separation

- Draw even more samples from the distribution  
 → *Line #3 separates even worse*

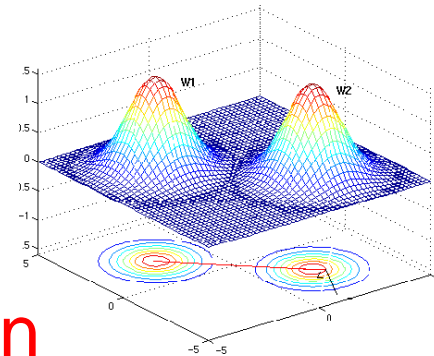


# Linear separation

- Draw even more samples from the distribution  
 → *Line #4 still separating*

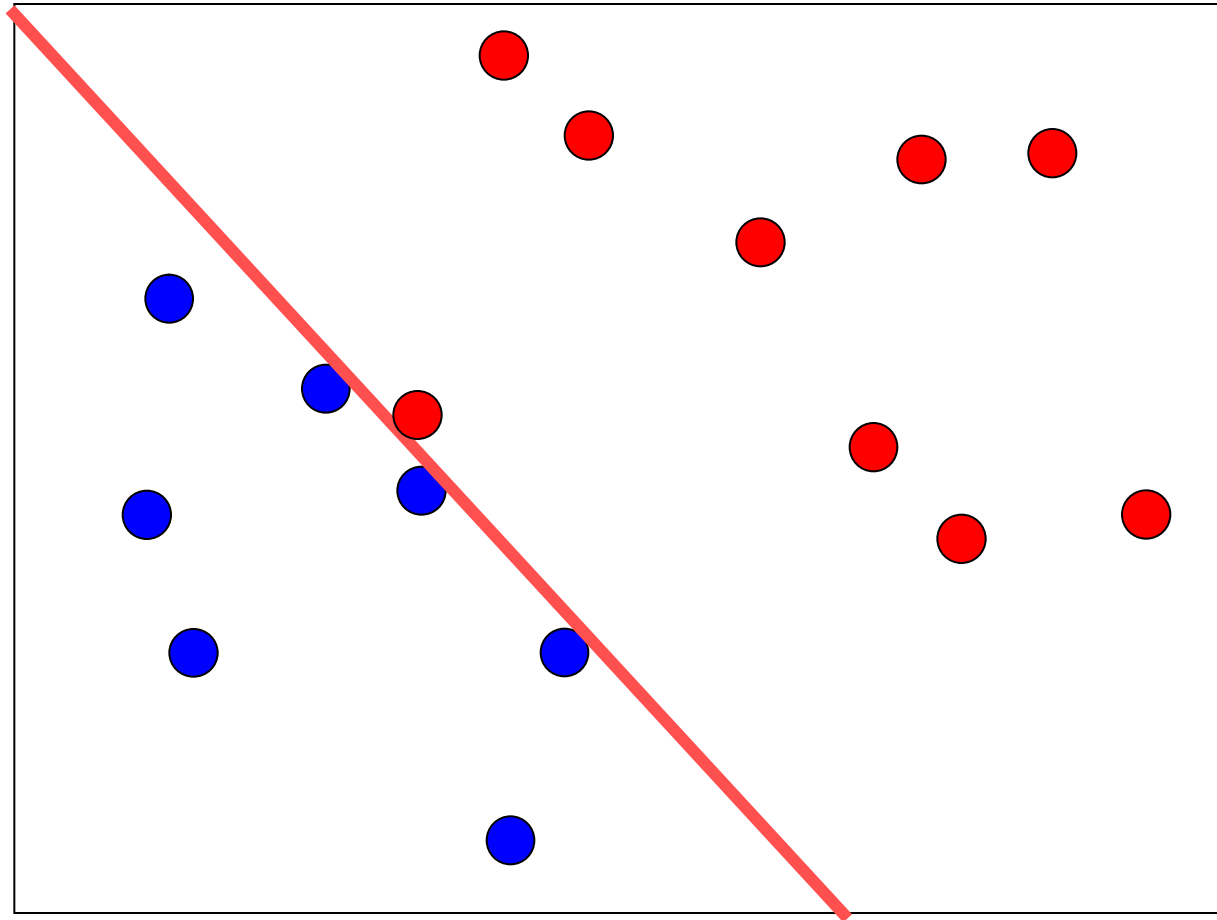


- *Which separation/margin is the best?*
- Claim: **bigger margin is better**
- Intuitive demonstration
  - The bigger the margin  $\rightarrow$  the better is the separating plane fitting to slightly different data
  - I.e. less “overfitting”, more generalization
- Next lecture: how to optimise the margin
  - Using Lagrange multipliers, quadratic programming,



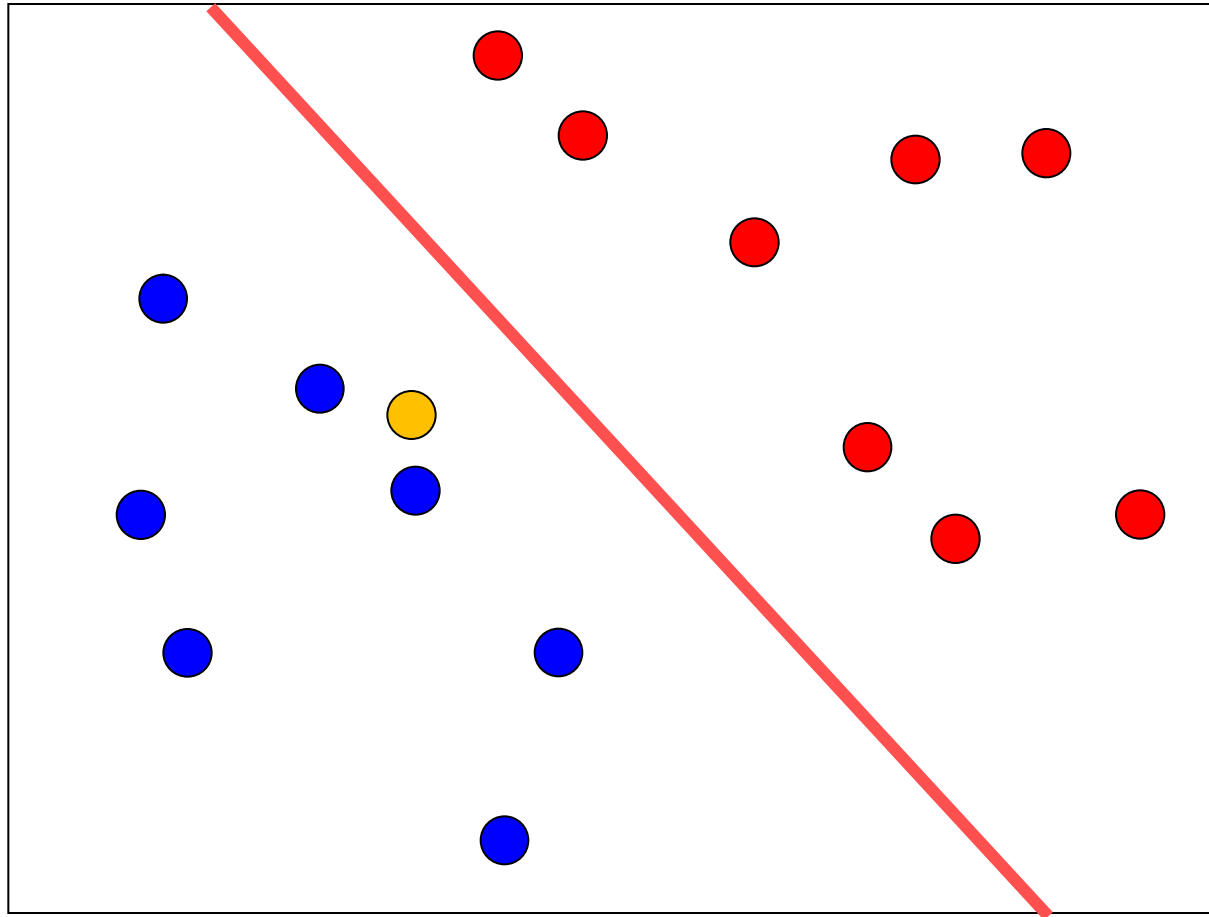
- Short recap
- Decision Trees – continued
- Evaluation
- Random Forests
- Evaluation, continued
- Data preparation
- SVM, intro: soft margin

- SVMs optimise the decision boundary ...
  - ... but still rely on ***linear separation*** !
- 
1. Sometimes linear separation not possible
  2. Sometimes, linear separation would lead to a badly generalising model
    - When?



# Bad generalisation

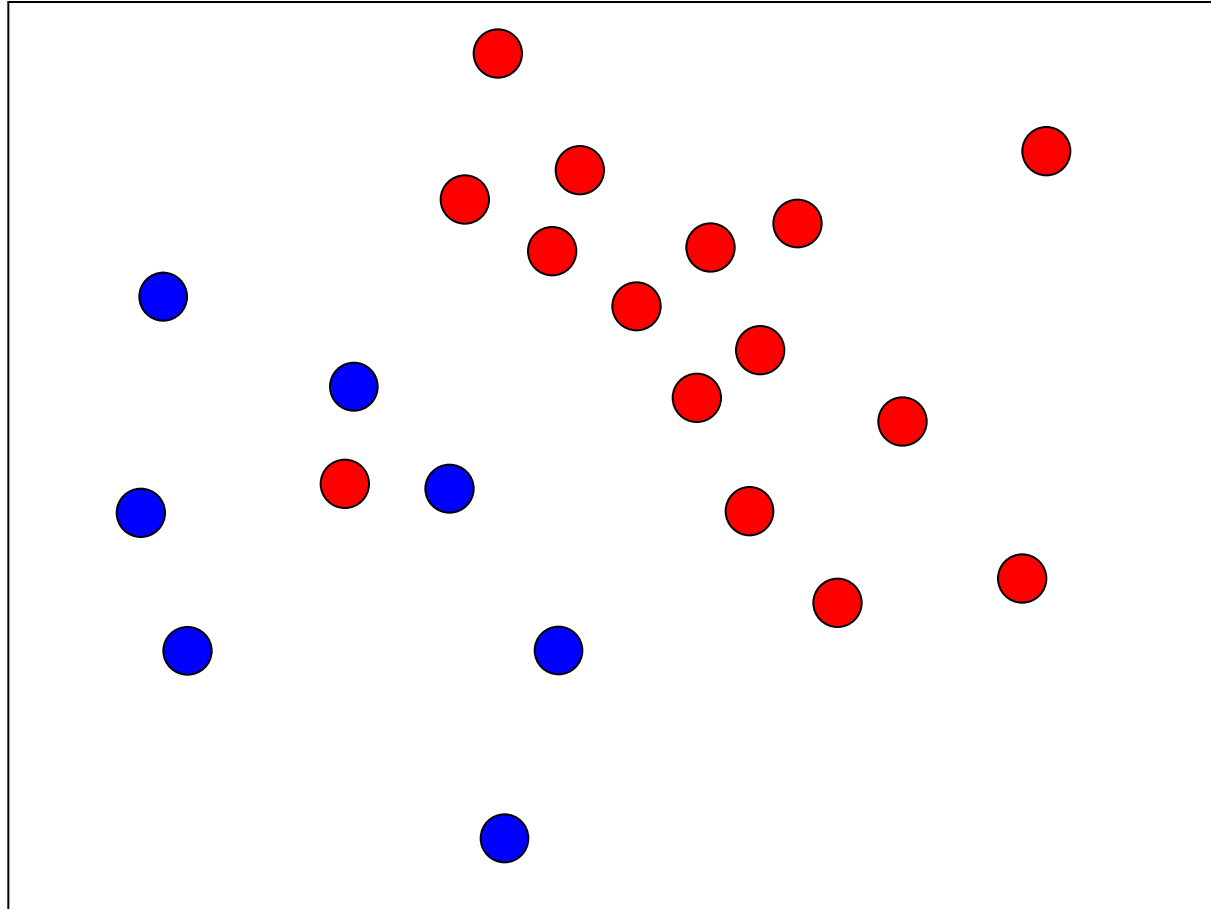
- *Wider margin might be better*





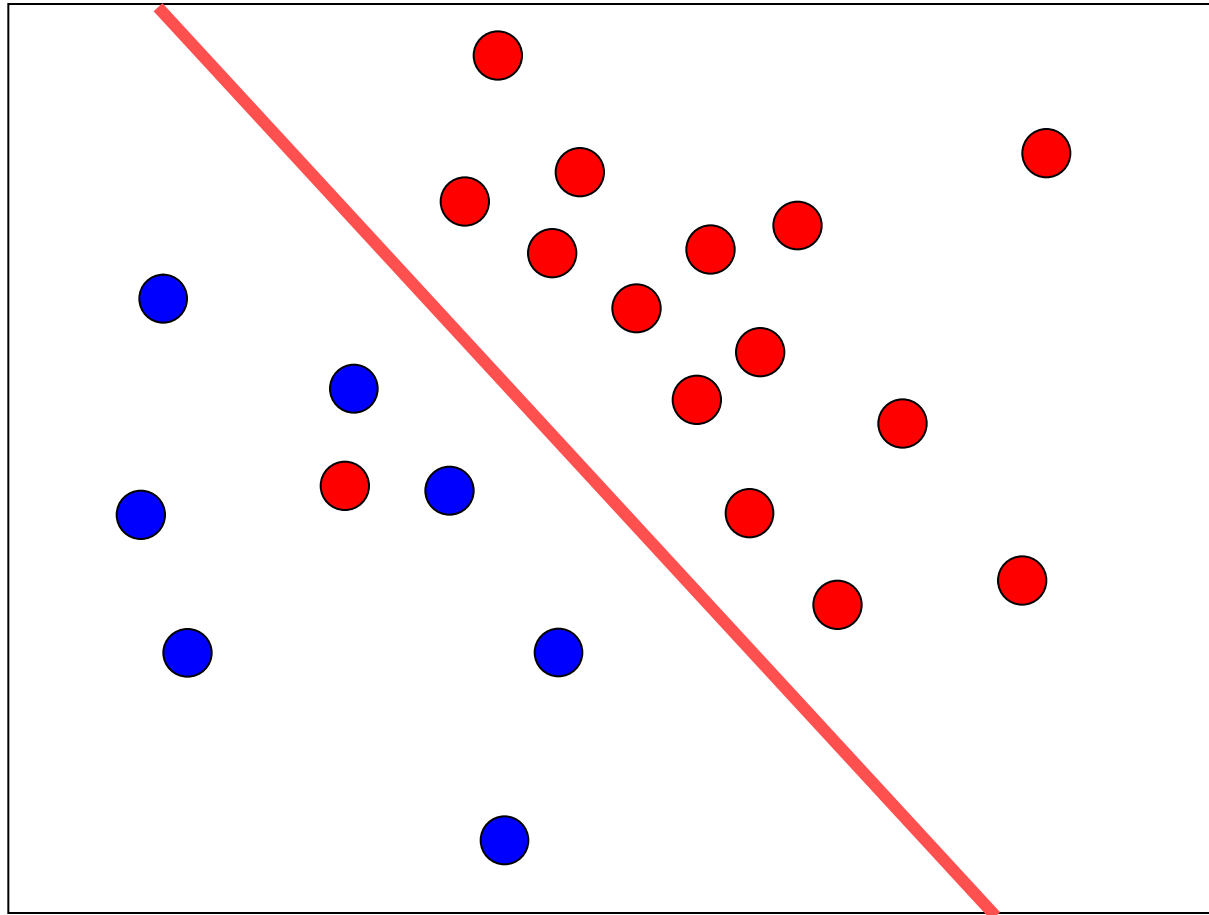
# Linear separation not possible

.....



# Linear separation not possible

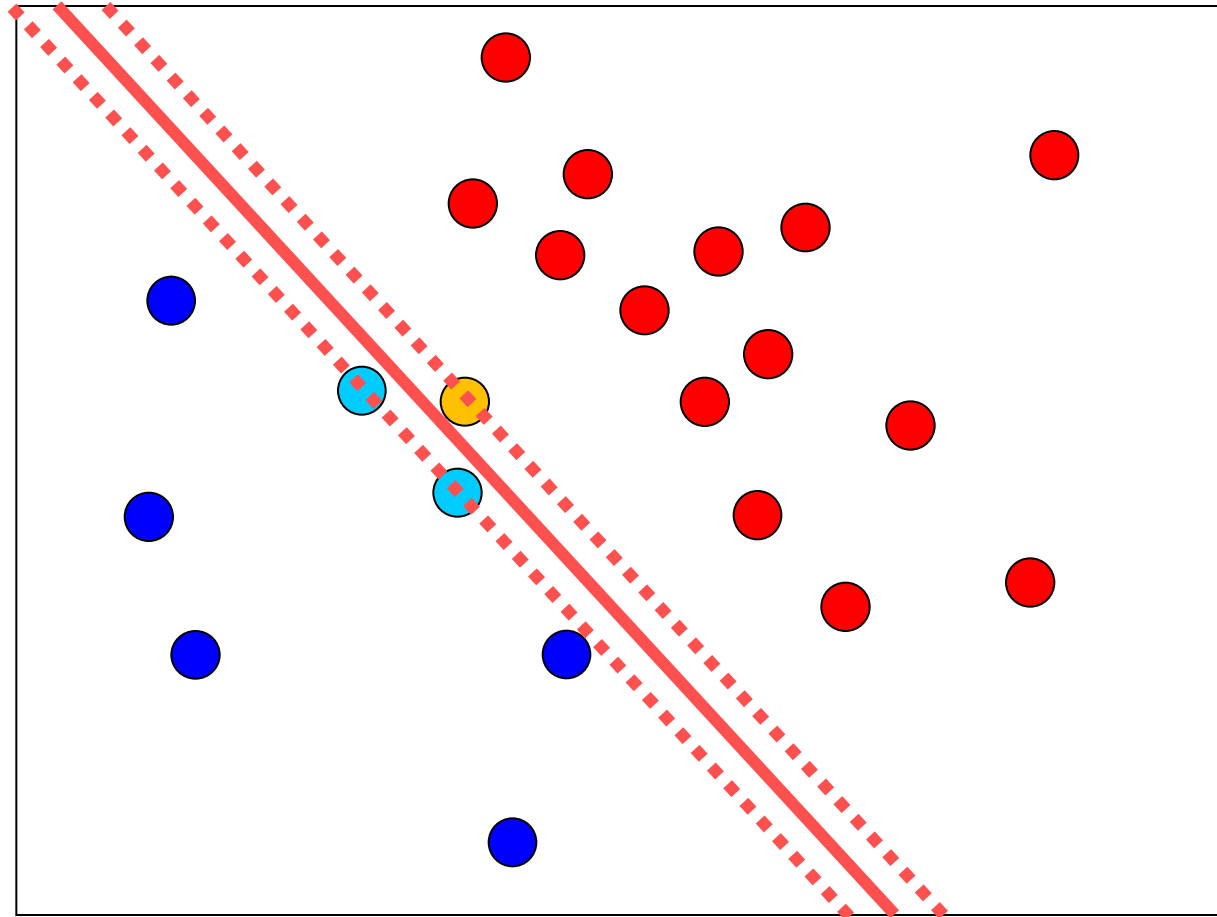
- *“Acceptable” hyperplane could still be found*

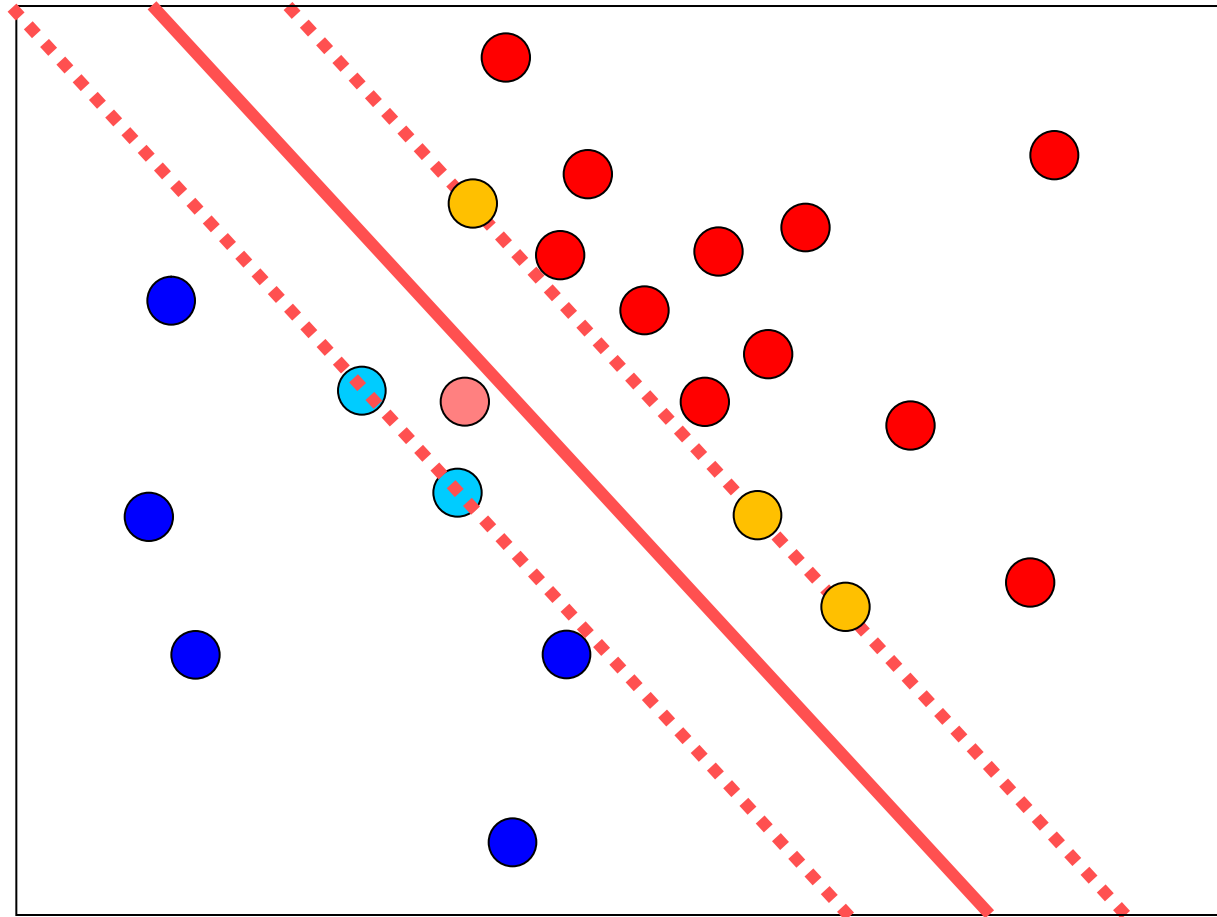


- Sometimes linear separation not possible, or
- Linear separation would lead to a badly generalising model

→ Soft margin

- Hyper plane that splits “as cleanly as possible/desirable”
- While maximising margin





- Introduction of slack variables
  - penalises misclassification
  - adapt constraint

$$y_i (\mathbf{w} \cdot \mathbf{x}_i - b) \geq 1 - \xi_i \quad \text{for all } i$$

- Penalise non-zero  $\xi_i$

$$\rightarrow \min \quad \|\mathbf{w}\| + C \sum_{i=1}^n \xi_i$$

- Using Lagrange multipliers, similar to “hard-margin” case

- *Implications?*

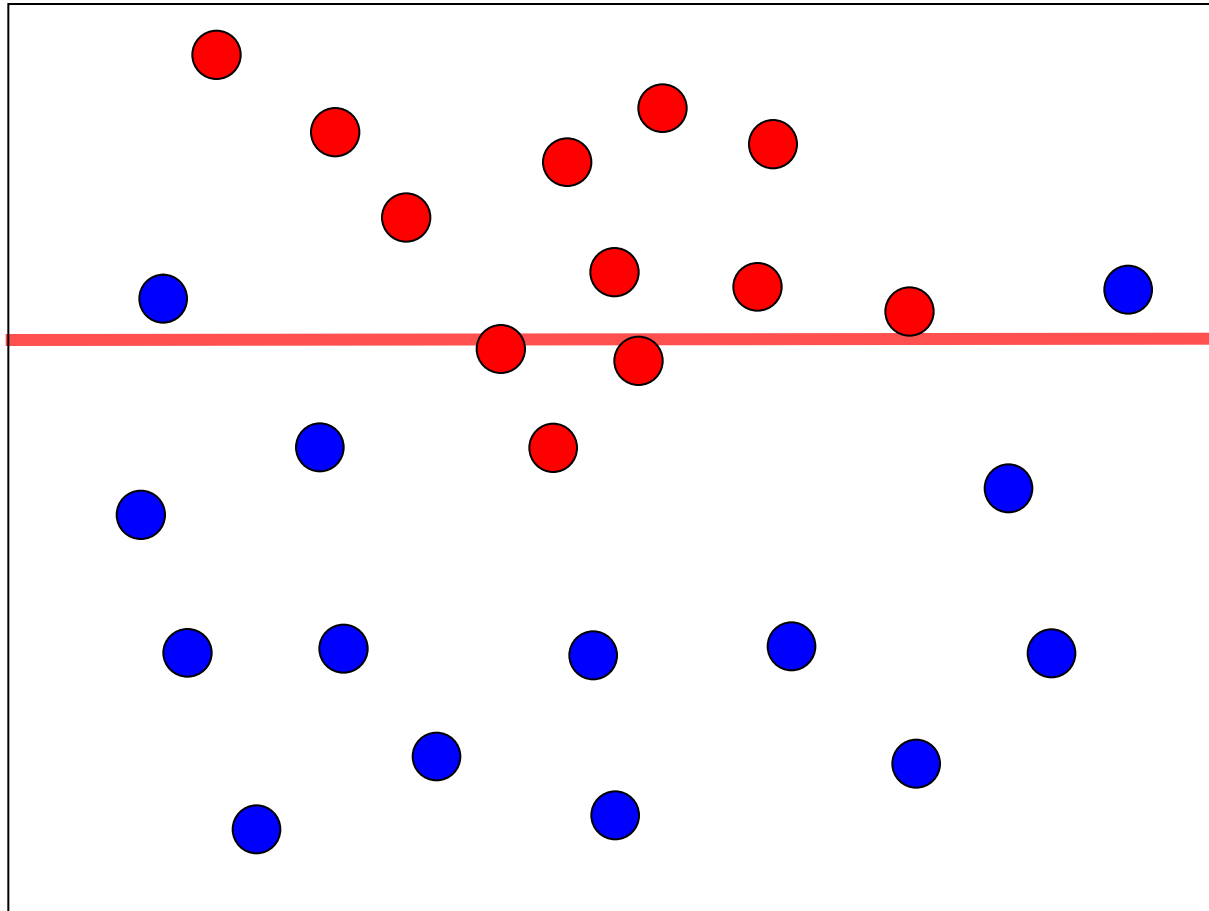
- Introduction of slack variables for optimisation problem
  - penalises misclassification
- *Implications ?*
  - Not necessarily 100% classification accuracy on training set
  - Even if linearly separable
- Optimisation: trade off between large margin and small error penalty (controlled by  $C$ )
  - Trade-off between fitting to training data and general model
- $C$  becomes part of optimisation problem
  - Sometimes called the “complexity parameter”

- Short recap
- Decision Trees – continued
- Evaluation
- Random Forests
- Evaluation, continued
- Data preparation
- SVM, intro: outlook on kernels



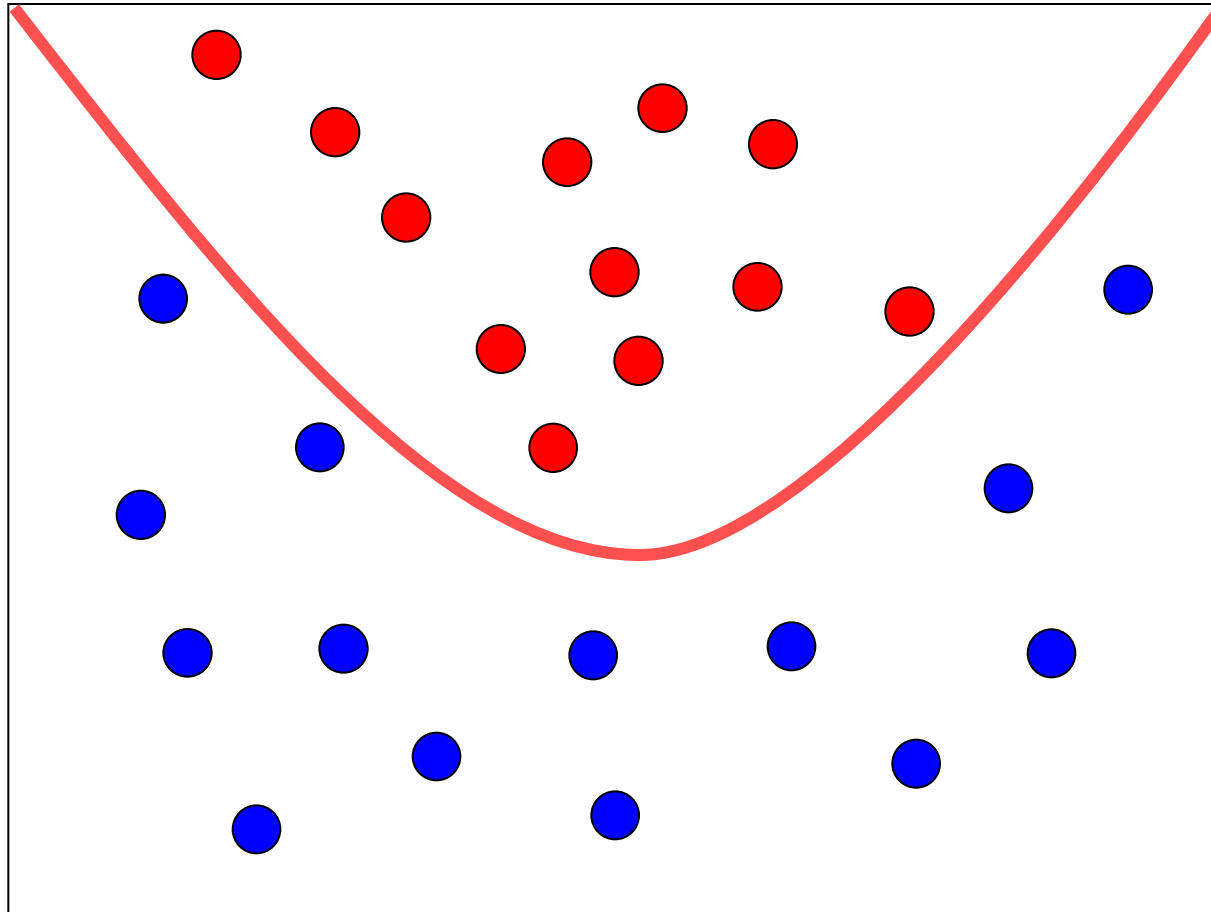
# Non-linearly separable data

- “Acceptable” hyperplane can ***not*** be found



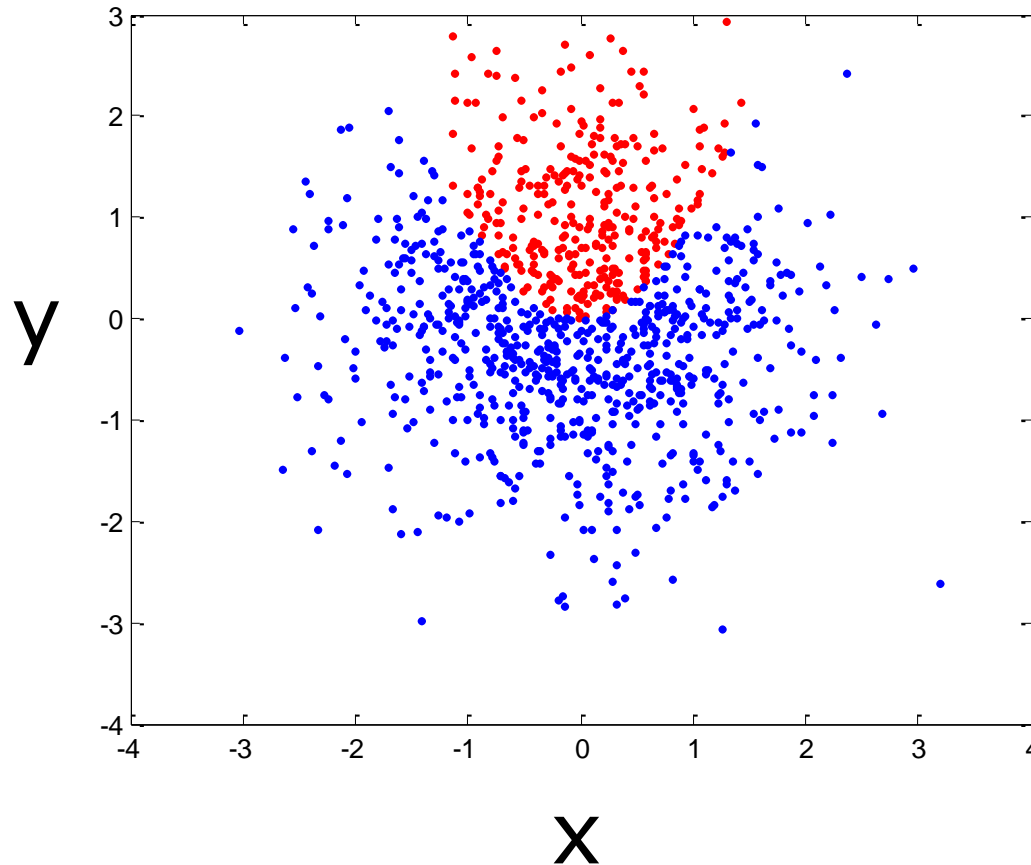
# Non-linearly separable data

- Data would be easily separable by a polynomial

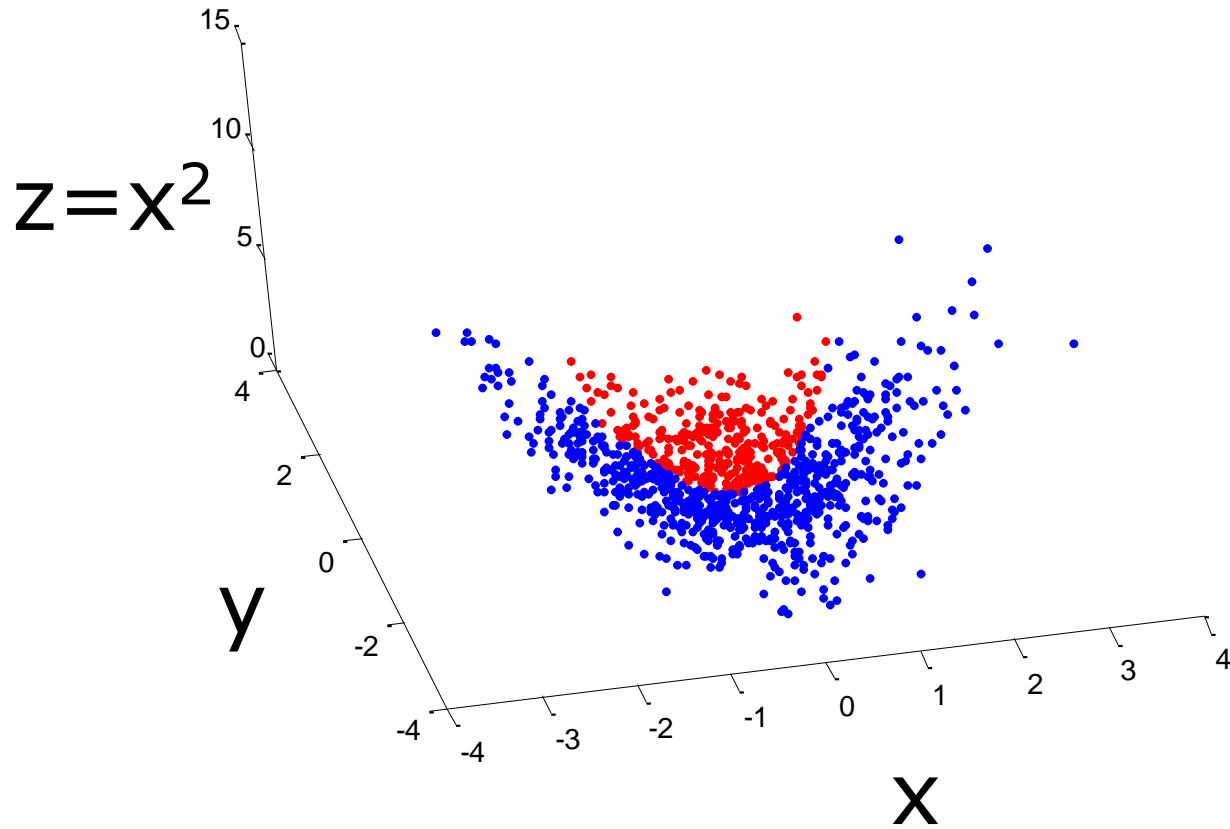


# Non-linearly separable data

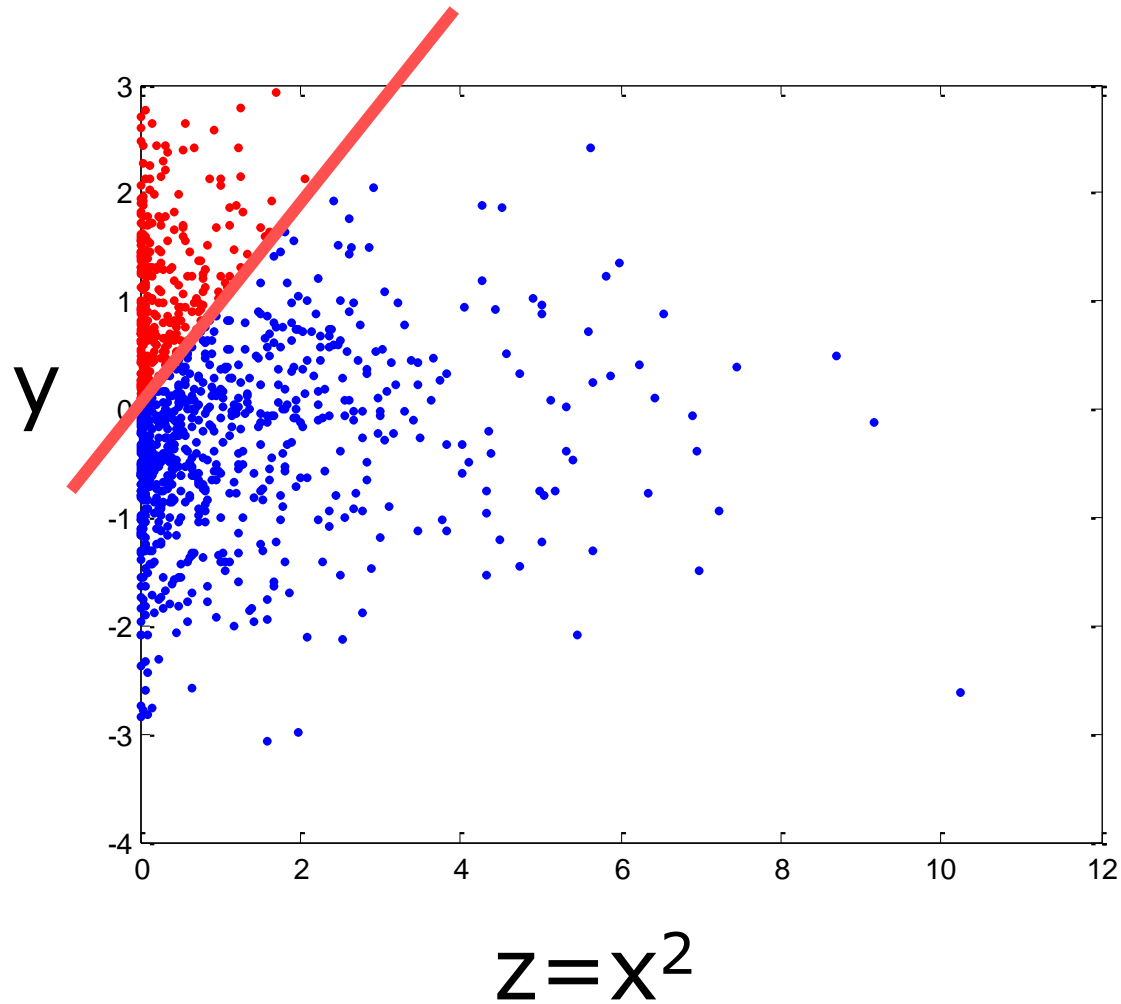
.....



# New coordinate $z=x^2$



# New coordinate $z=x^2$



- Kernel – projection into high dimensional space
  - Kernel perceptron & difference to SVMs
  - “Kernel trick”

# Questions?