Machine Learning

Rudolf Mayer

(mayer@ifs.tuwien.ac.at)

November 27th, 2019

Outline

- Short recap
- Random Forests
- Evaluation
- Support Vector Machines

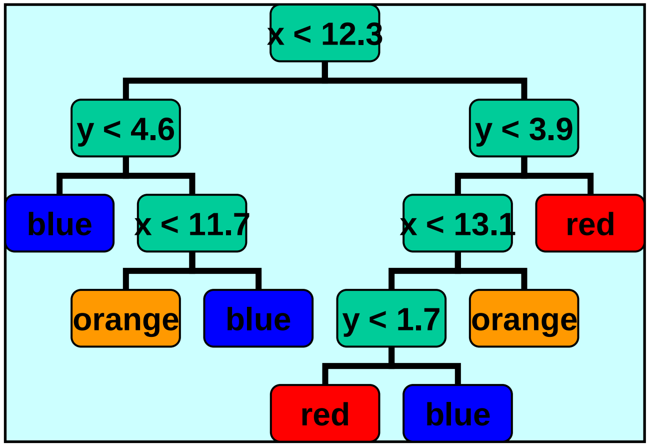
Exercises

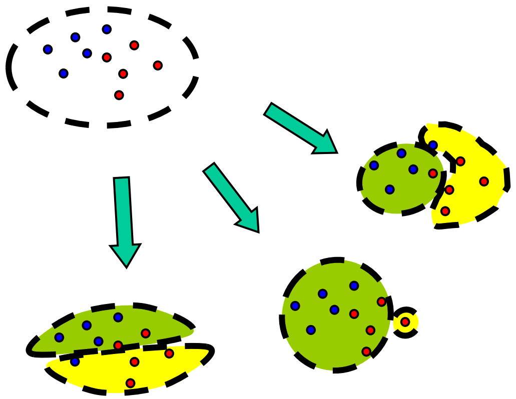
- Exercise 1 (Regression) discussion
  - 28.11. – 6.12. (apply if you haven't done it yet)
  - Everyone needs to attend

- Exercise 2 (Classification & Exercise 3
  - Need to present either Exercise 2 or Exercise 3
  - You need to attend the whole slot (~2 – 2,5 hours)
  - All group members should be present
  - Submission: Report for both Exercise 2 & 3
    - In addition a presentation for the one that you present
      - (has a different DL)
  - Dates: December 16 – 20 resp. January 27 - 31

Short Recap

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





Short Recap

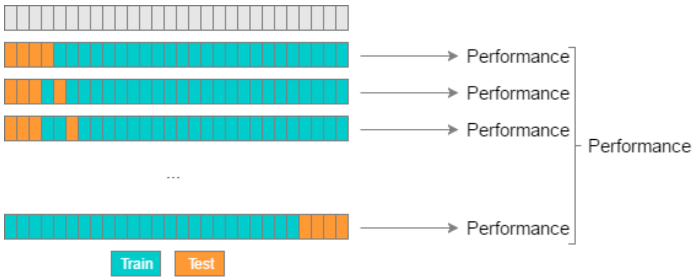
- Evaluation
  - Confusion Matrix

					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

  - Micro vs. Macro averaging

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

  - Leave-p-out cross-validation



  - Bootstrapping

Outline

- Short recap
- Random Forests
- Evaluation
- Support Vector Machines

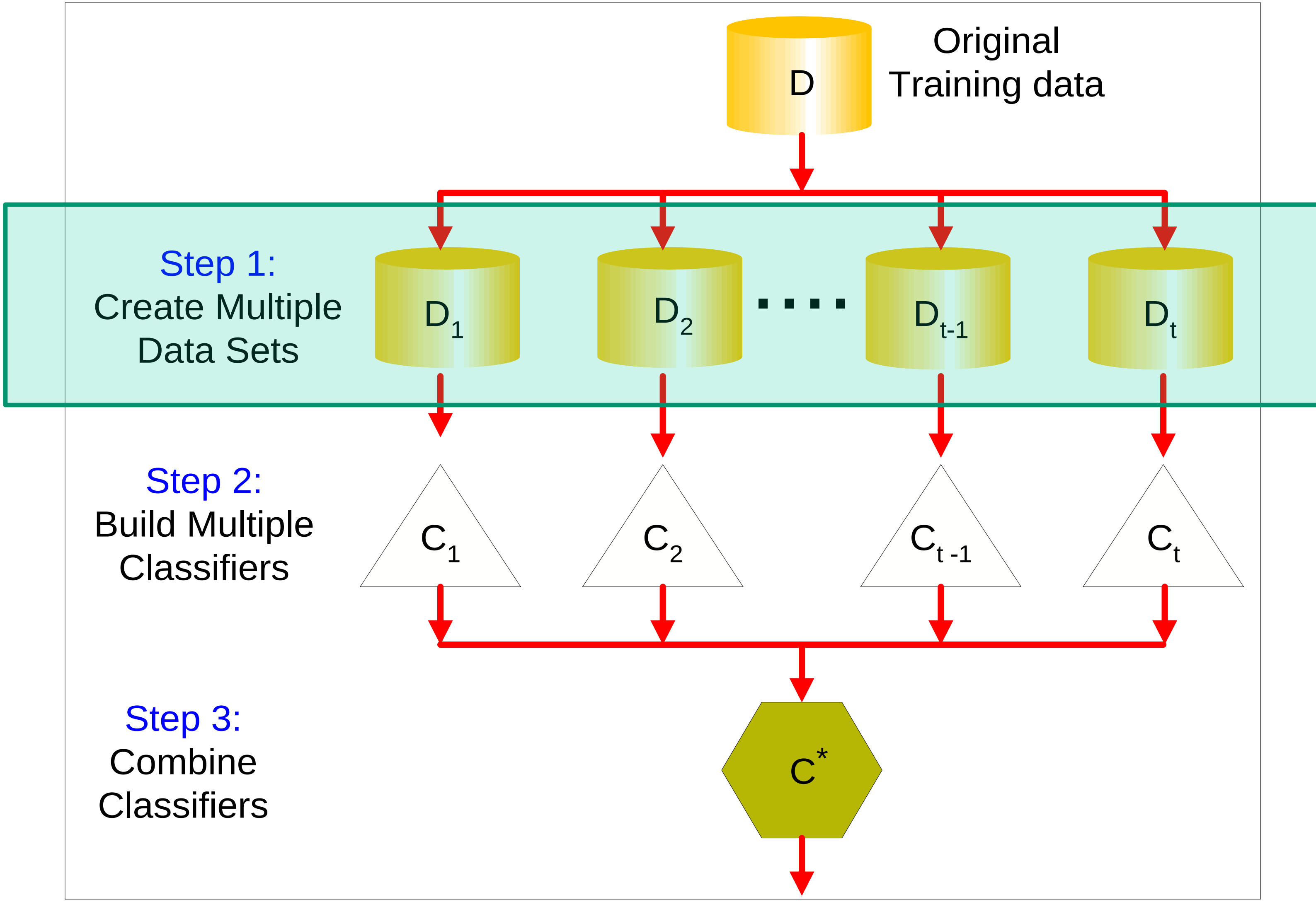
Random Forests

- Combination of **Decision Tree** 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
  - Utilises instability of Decision Trees

Random Forests



Random Forests

- 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 extent
  - No stopping, no pruning

Example: Tree in Random Forests

Input Output

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

Example: Tree in Random Forests

Input Output

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

Bootstrap sample

Example: Tree in Random Forests

Input Output

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

?

Example: Tree in Random Forests

Input Output

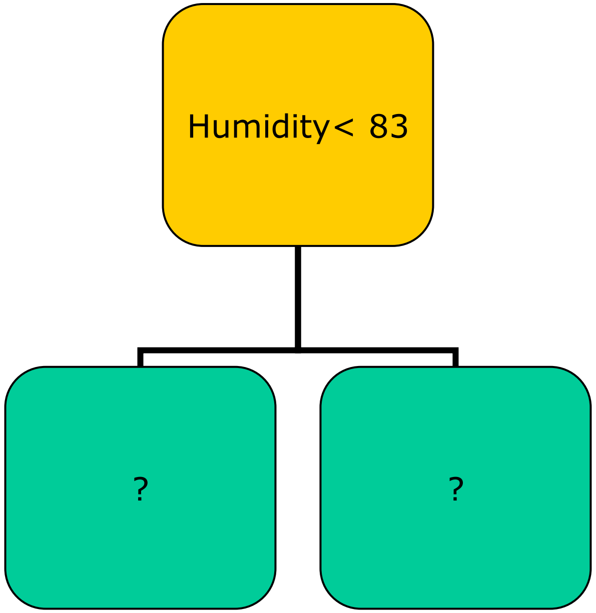
Outlook	Temperature	Humidity	Windy	Play?
sunny	85	85	false	Don't Play
overcast	83	78	false	Play
rain	68	80	false	Play
sunny	72	95	false	Don't Play
sunny	69	70	false	Play
rain	75	80	false	Play
overcast	72	90	true	Play
rain	71	80	true	Don't Play

?

Example: Tree in Random Forests

Input Output

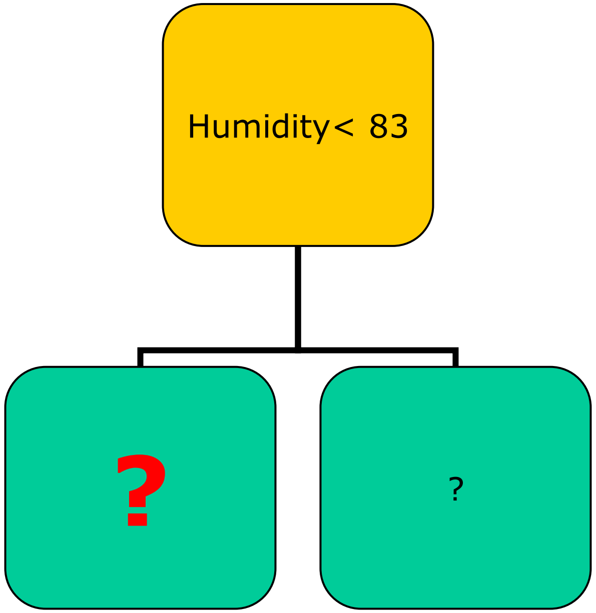
Outlook	Temperature	Humidity	Windy	Play?
sunny	85	85	false	Don't Play
overcast	83	78	false	Play
rain	68	80	false	Play
sunny	72	95	false	Don't Play
sunny	69	70	false	Play
rain	75	80	false	Play
overcast	72	90	true	Play
rain	71	80	true	Don't Play



Example: Tree in Random Forests

Input Output

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

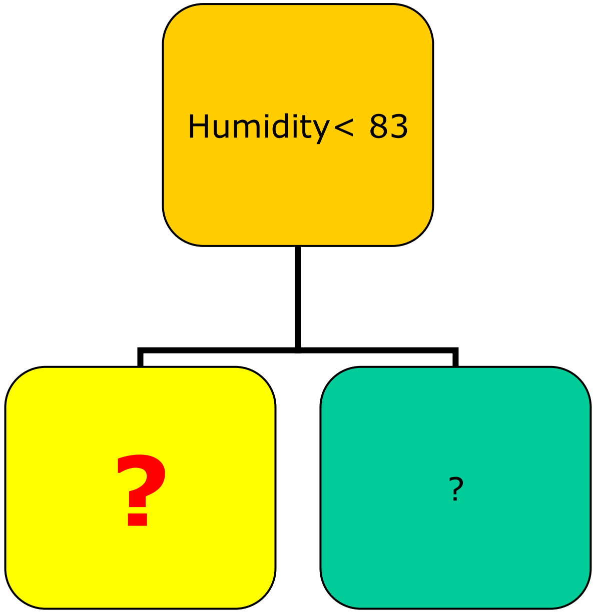


Example: Tree in Random Forests

Input Output

!! Select new attributes at each tree node !!

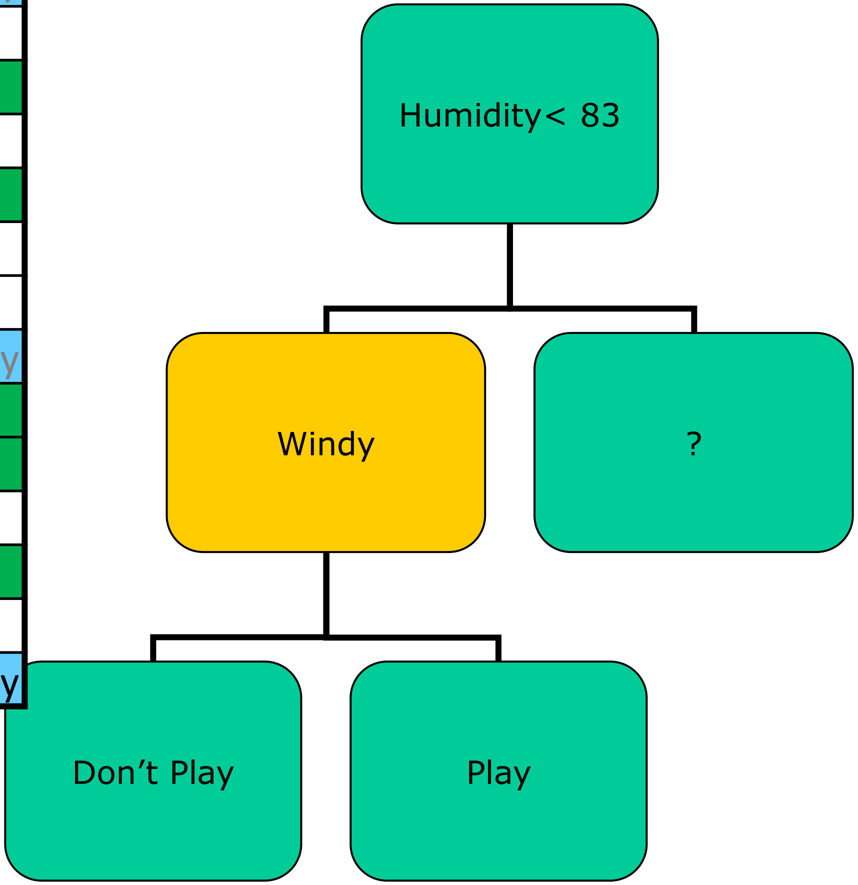
Outlook	Temperature	Humidity	Windy	Play?
sunny	85	85	false	Don't Play
overcast	83	78	false	Play
rain	68	80	false	Play
sunny	72	95	false	Don't Play
sunny	69	70	false	Play
rain	75	80	false	Play
overcast	72	90	true	Play
rain	71	80	true	Don't Play



Example: Tree in Random Forests

Input Output

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

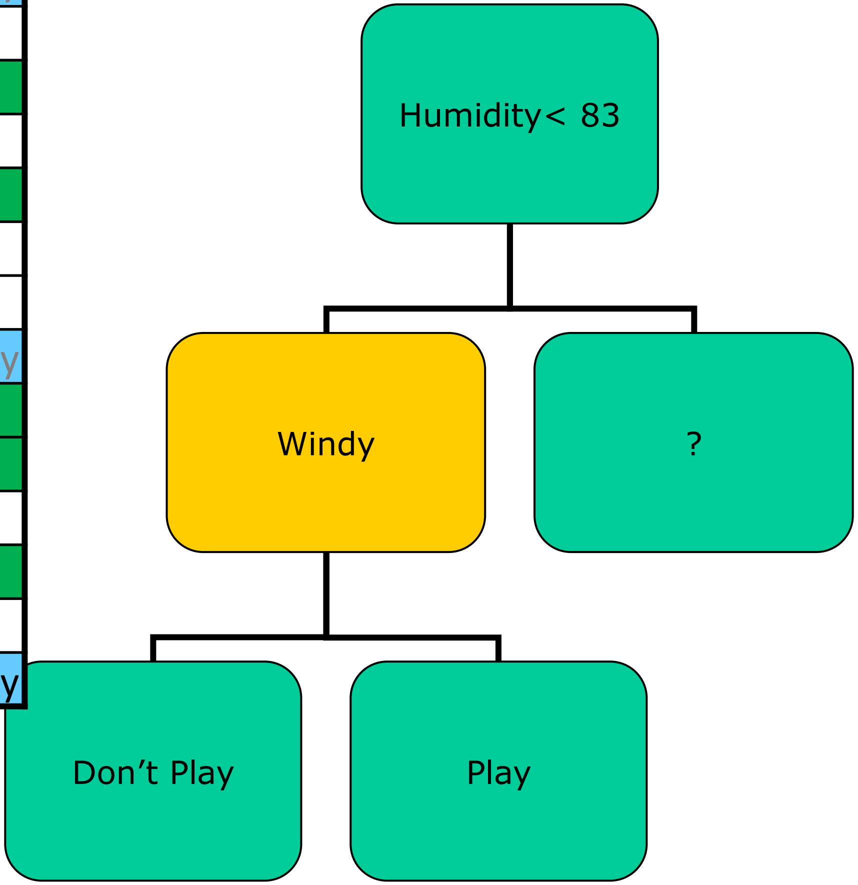


FACULTY OF !NFORMATICS

Example: Tree in Random Forests

Input Output

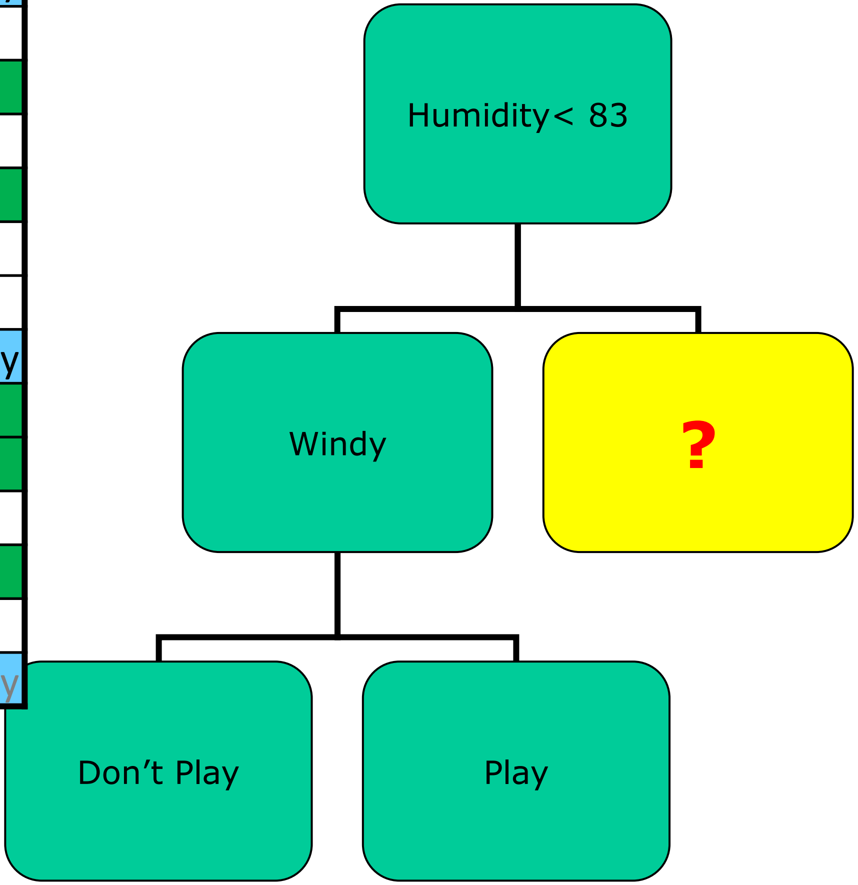
Outlook	Temperature	Humidity	Windy	Play?
sunny	85	85	false	Don't Play
overcast	83	78	false	Play
rain	68	80	false	Play
sunny	72	95	false	Don't Play
sunny	69	70	false	Play
rain	75	80	false	Play
overcast	72	90	true	Play
rain	71	80	true	Don't Play



Example: Tree in Random Forests

Input Output

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

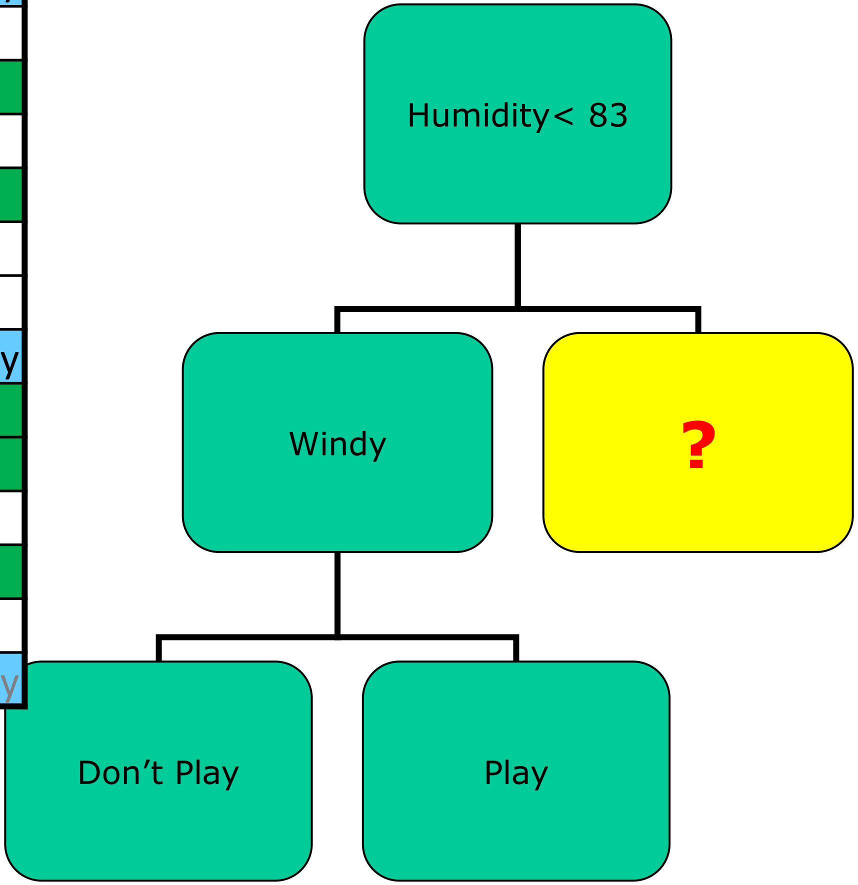


Example: Tree in Random Forests

Input Output

!! Select new attributes at each tree node !!

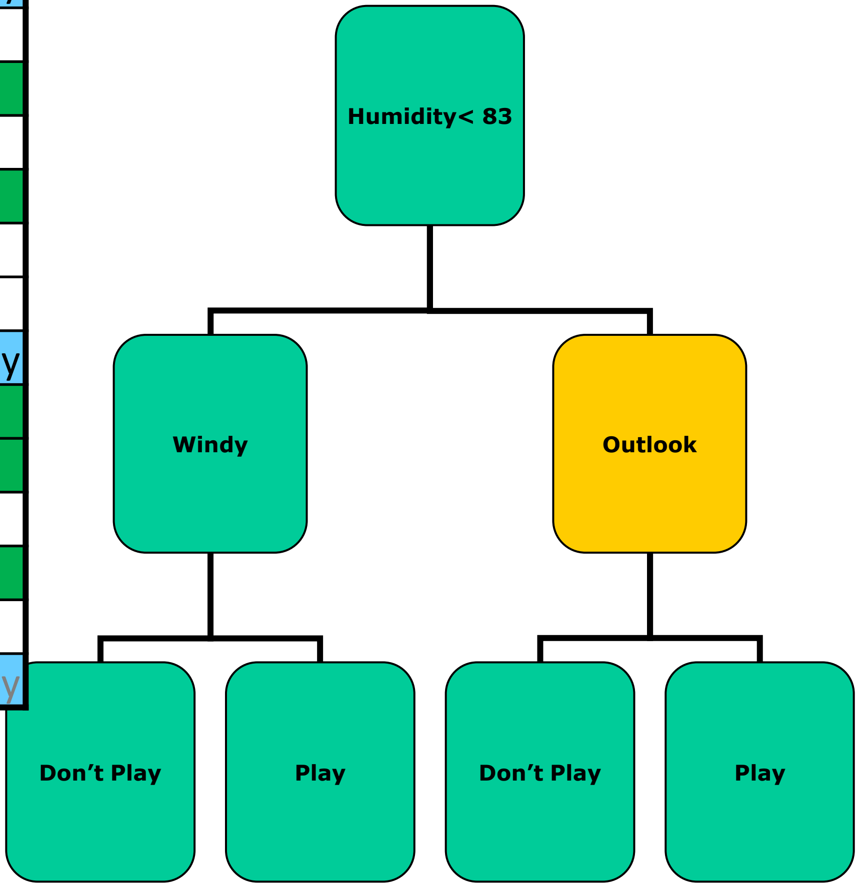
Outlook	Temperature	Humidity	Windy	Play?
sunny	85	85	false	Don't Play
overcast	83	78	false	Play
rain	68	80	false	Play
sunny	72	95	false	Don't Play
sunny	69	70	false	Play
rain	75	80	false	Play
overcast	72	90	true	Play
rain	71	80	true	Don't Play



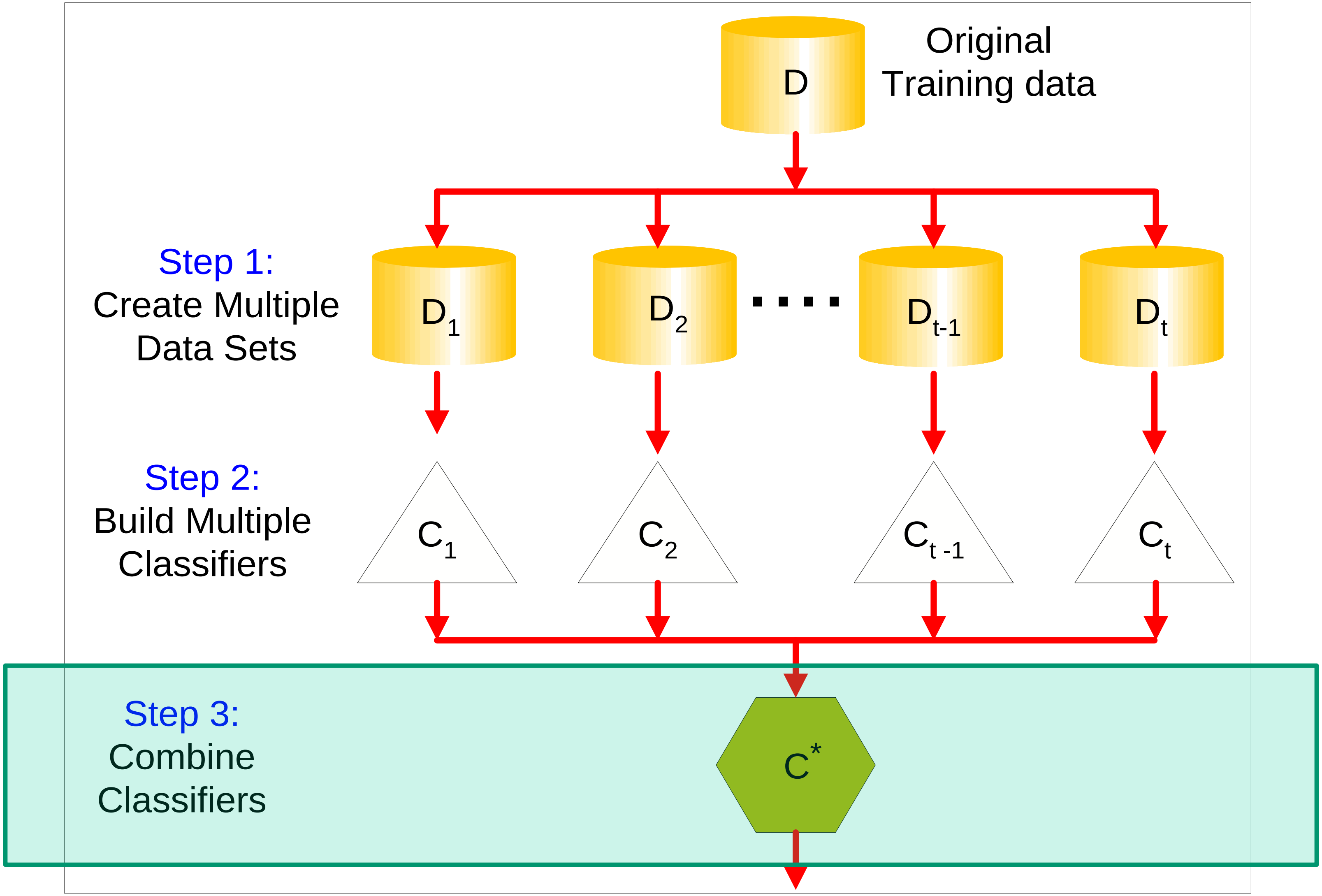
Example: Tree in Random Forests

Input Output

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



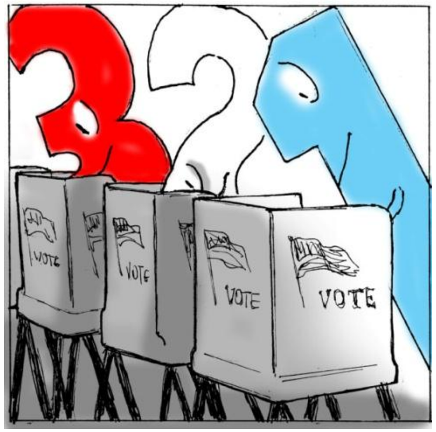
Random Forests



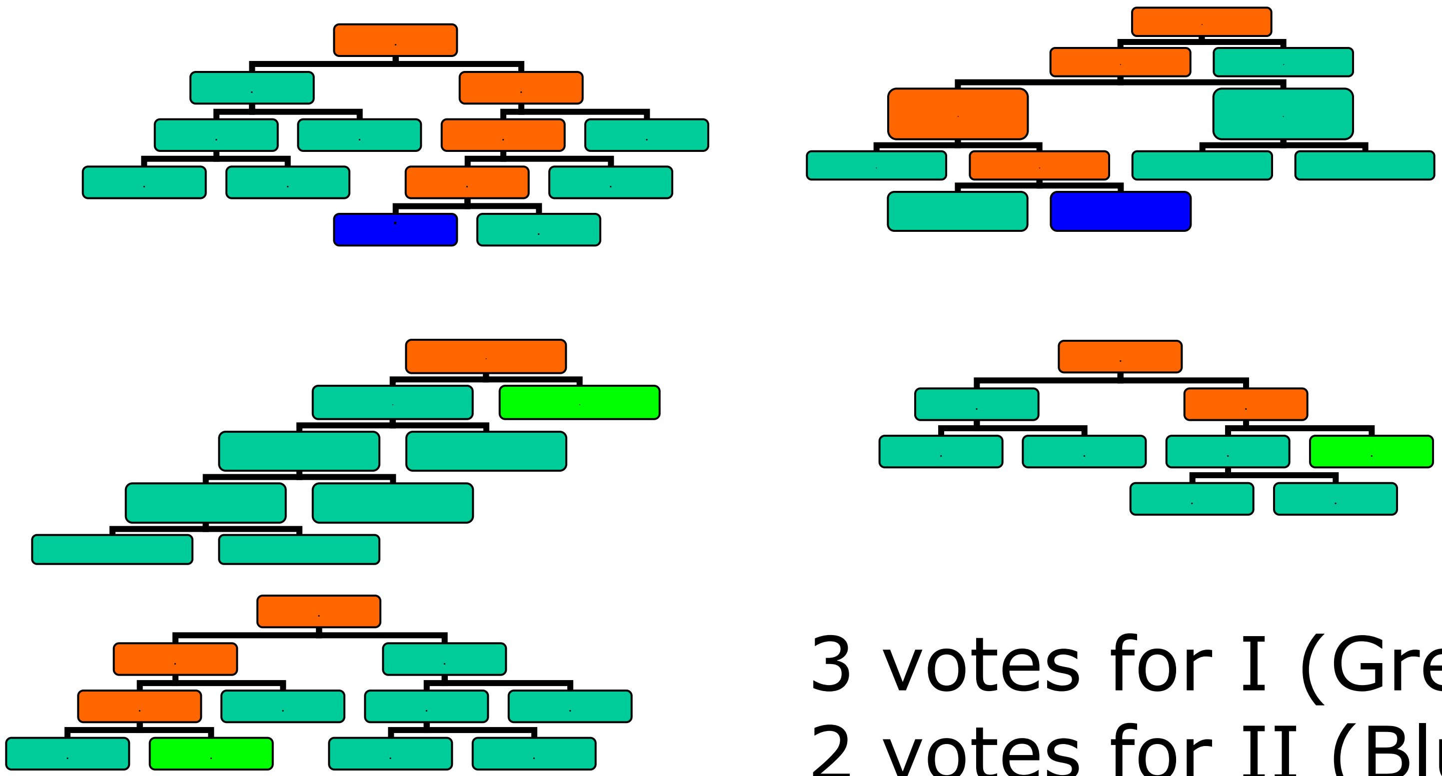
Random Forests

- 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



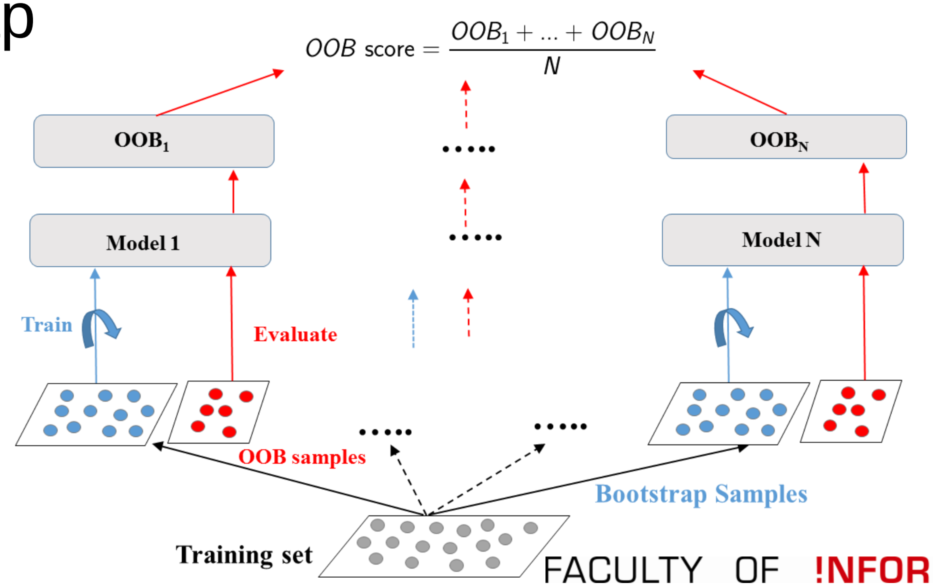
Classification with Random Forests



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

Random Forests – Evaluation

- *How to evaluate performance of Random Forests?*
  - Holdout, cross-validation, etc..
- Alternatively: out-of-bag error / estimate (OOB)
  - General method for bootstrapped algorithms
  - Compute mean error on each training sample $x_{i,}$, using only trees that did not have x_i in their bootstrap sample



Properties of Random Forests

- 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 slower with increasing number of trees
  - Maybe not so critical – ***why?***
  - Can be parallelised

Outline

- Short recap
- Random Forests
- Evaluation
- Support Vector Machines

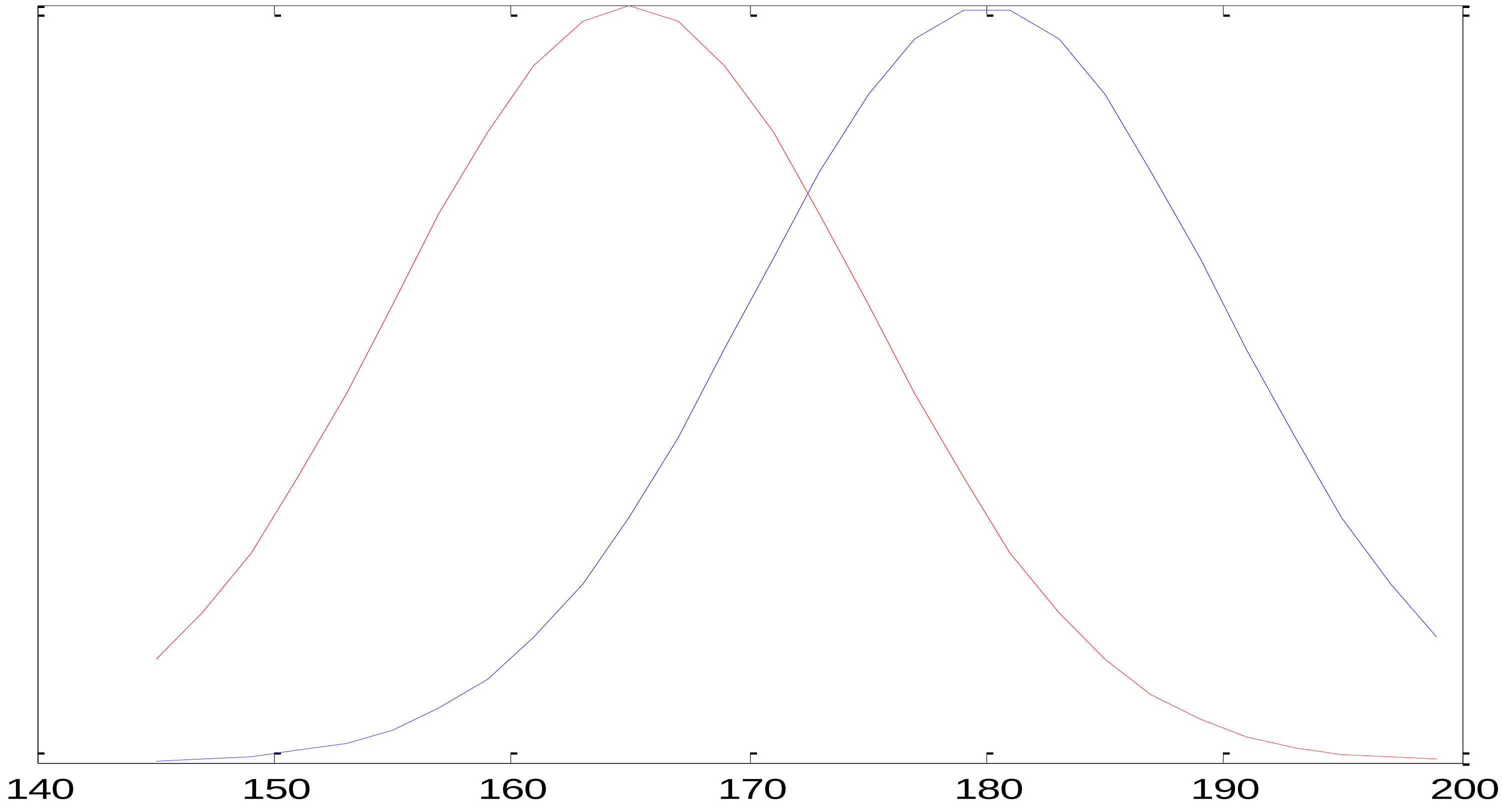
Overfitting & Generalisation

- 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

Overfitting & Generalisation

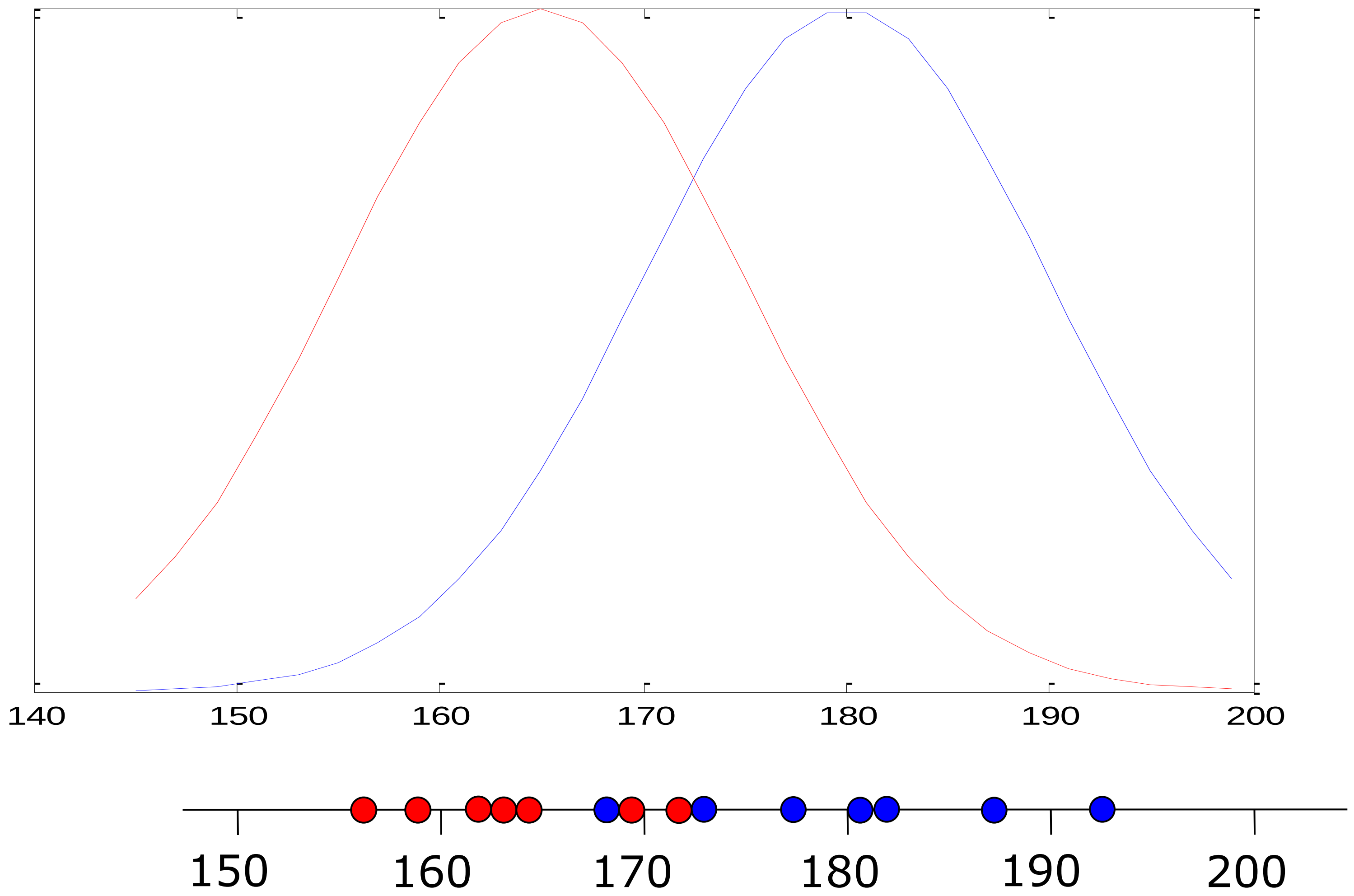
- Distributions of two classes (Gaussian, different mean)



FACULTY OF !INFORMATICS

Overfitting & Generalisation

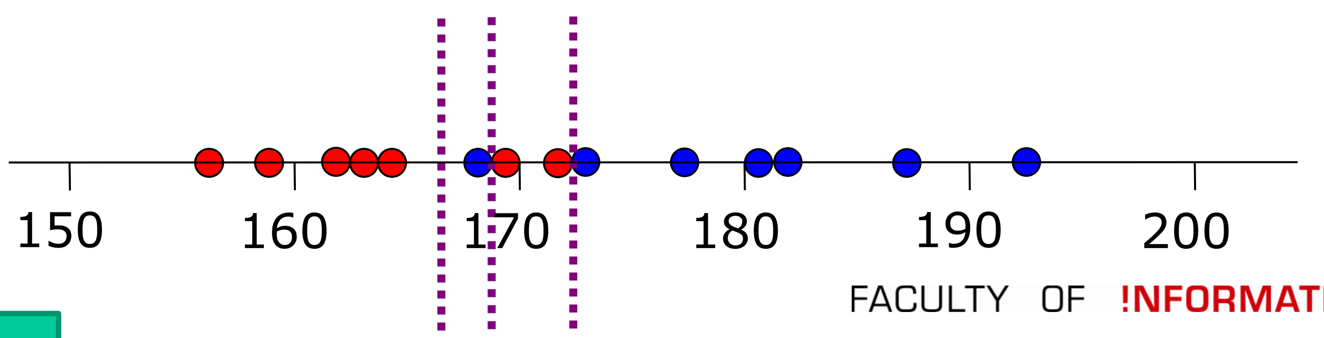
- Points drawn from that distributions



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

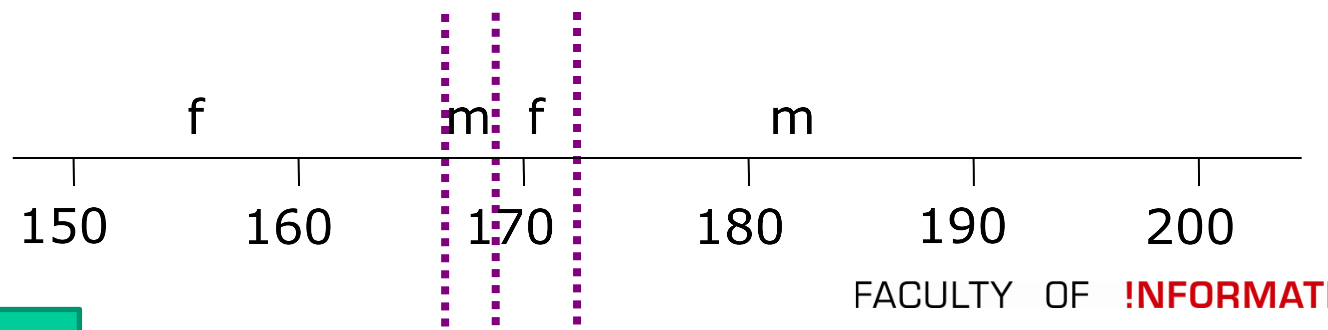


Overfitting & Generalisation

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

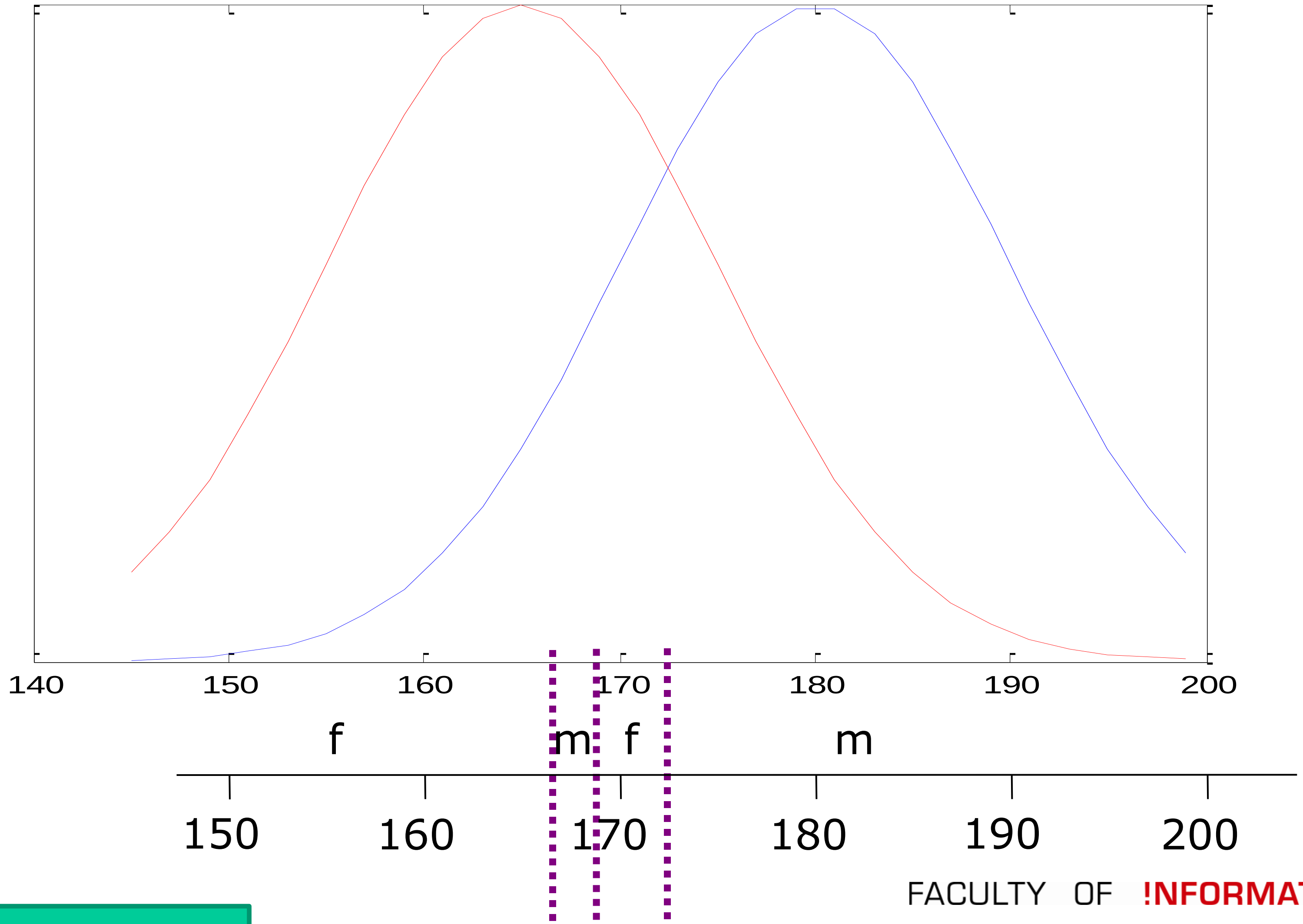
- Assign each point to the closest neighbour
- ➔ decision boundaries half-way between points of different class

- Classify according to closest point



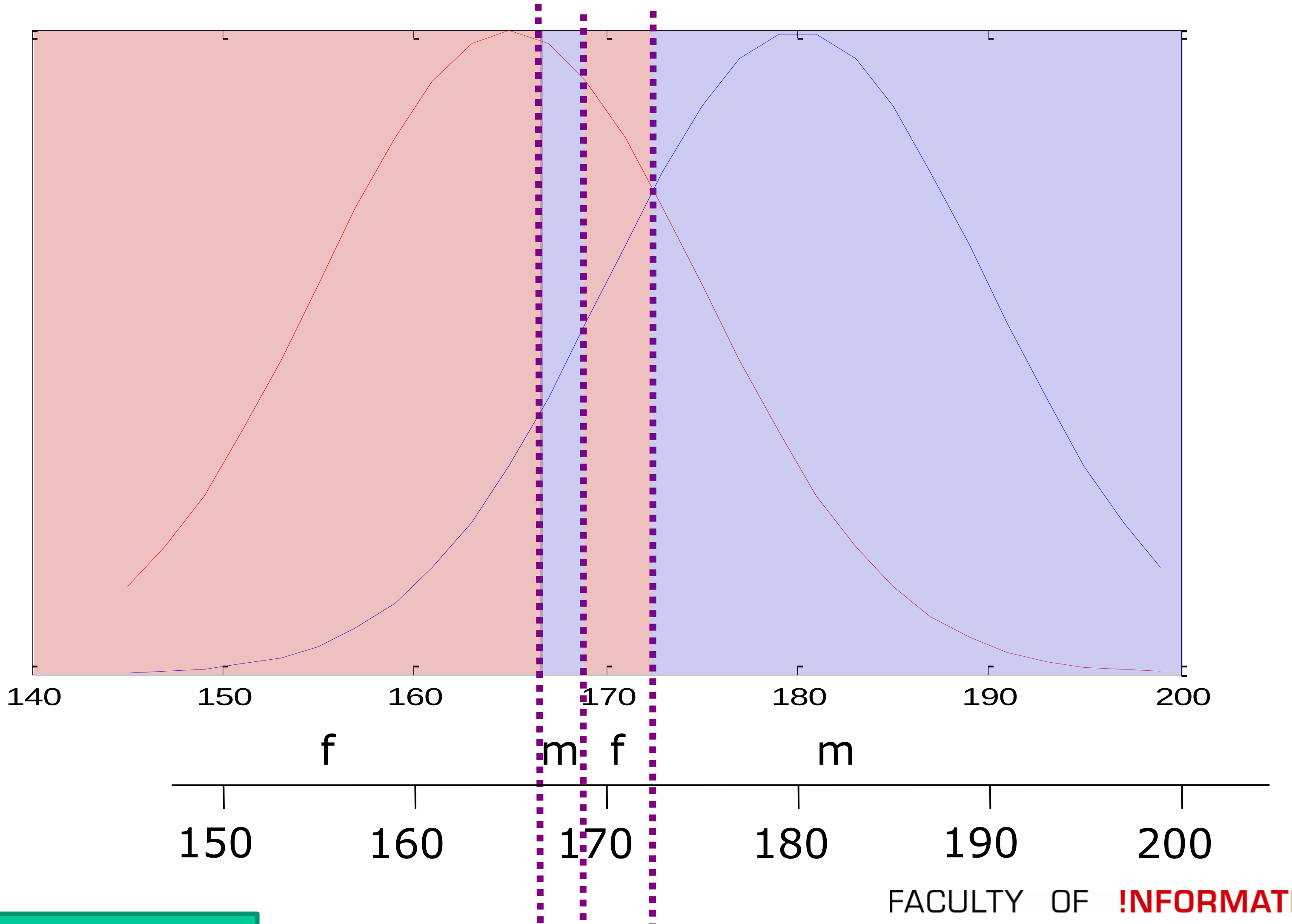
Overfitting & Generalisation

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



Overfitting & Generalisation

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

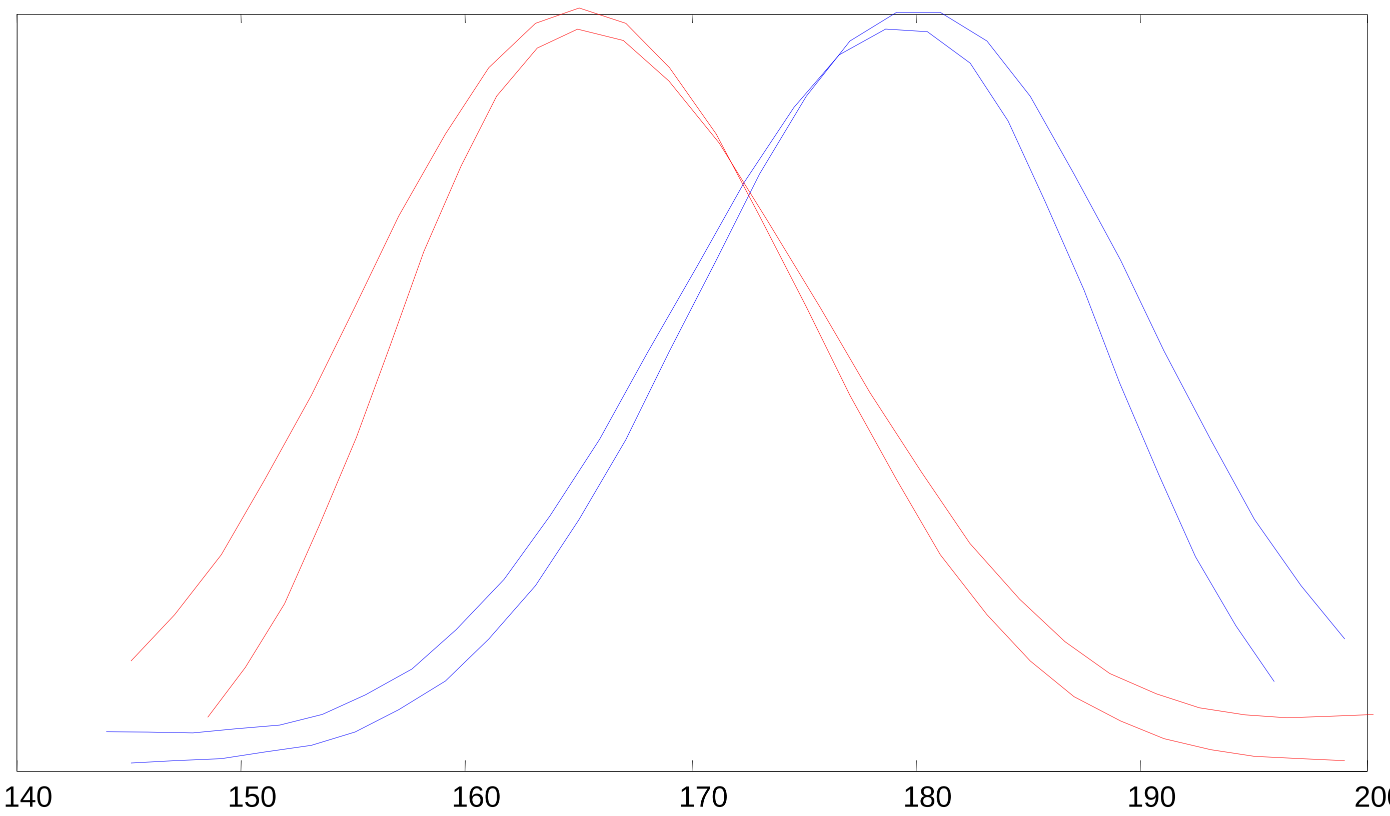


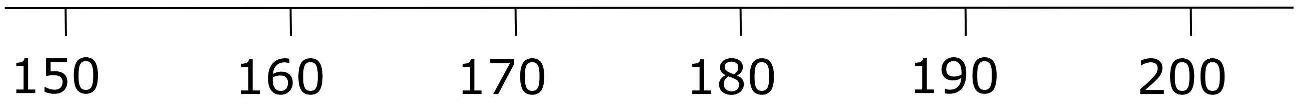
Overfitting & Generalisation

- 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
    - C.f.: similar to what Naive Bayes does
      - But by assessing all attribute values together, not independently

Overfitting & Generalisation

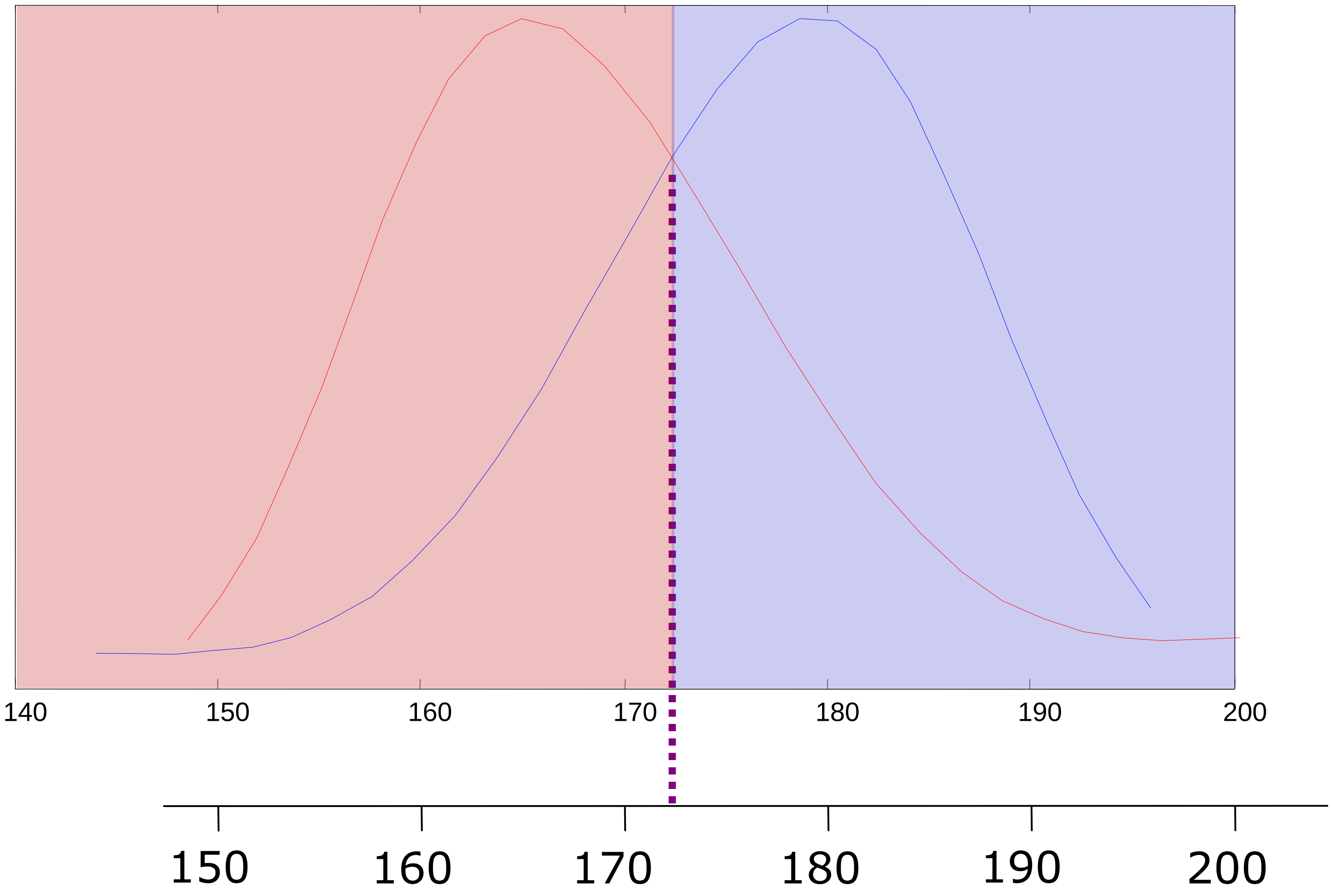
Bayes optimal classifier: estimation of probability density function



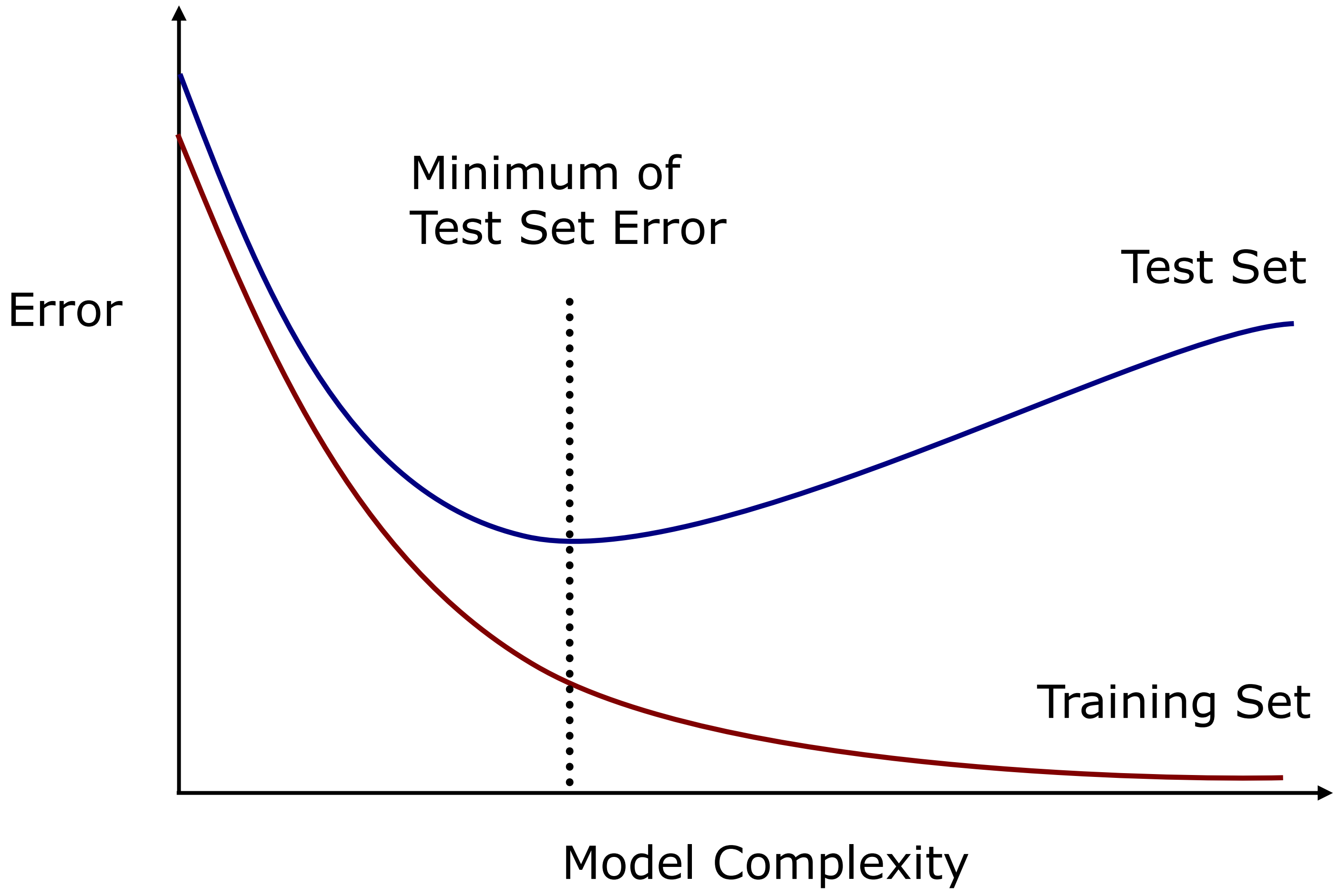


Overfitting & Generalisation

Bayes optimal classifier: estimation of probability density function



Trade-off complexity vs. generalization



Bias & Variance

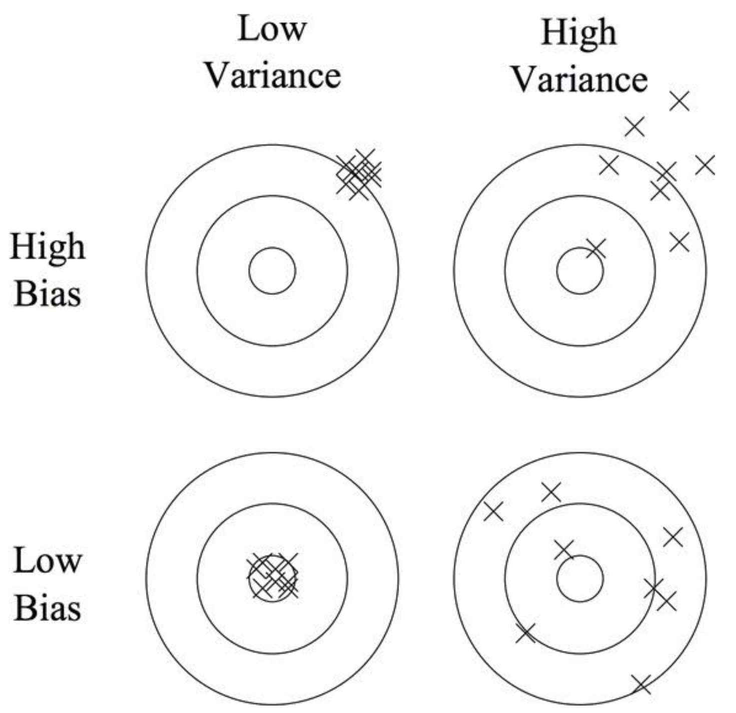
- *Errors: difference prediction ←→ actual output value*
- **Bias** and **variance** component

- Bias: errors from erroneous assumptions in learning algorithm
  – High bias: model misses relevant relations
  – "Model not complex enough"
- Variance: error stems from sensitivity to small fluctuations in training set
  – Small fluctuations → large difference in model
  – "Models noise"; "instable"
  – *Example?*

Bias & Variance

- Bias is the algorithm's tendency to consistently learn the wrong thing by not taking into account all the information in the data (underfitting)

- Variance is the algorithm's tendency to learn random things irrespective of the real signal by fitting highly flexible models that follow the error/noise in the data too closely (overfitting)



Bias & Variance

- Low bias: models are usually more complex
  - Represent the training set more accurately
    - Might also represent noise

- High bias: simpler models that don't tend to overfit
  - May underfit, failing to capture important regularities

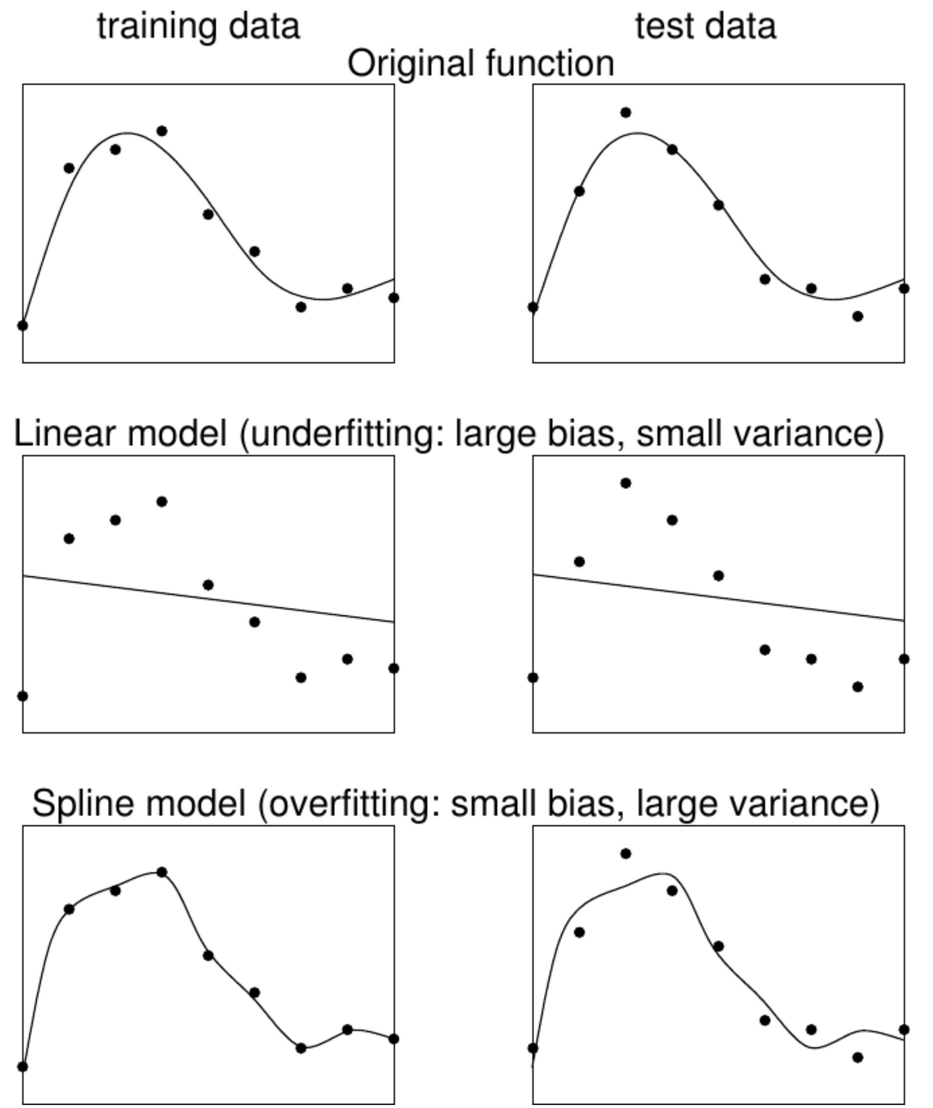
Bias & Variance

- High Variance
  - If different training sets lead to (very) different classifiers / decision boundaries
  - Learning methods able to represent training set well
    - Risk of overfitting to noisy or unrepresentative training data

- “Measure for prediction consistency”
- Variance is the “memory capacity”
  - To which detail characteristics of training data can be remembered

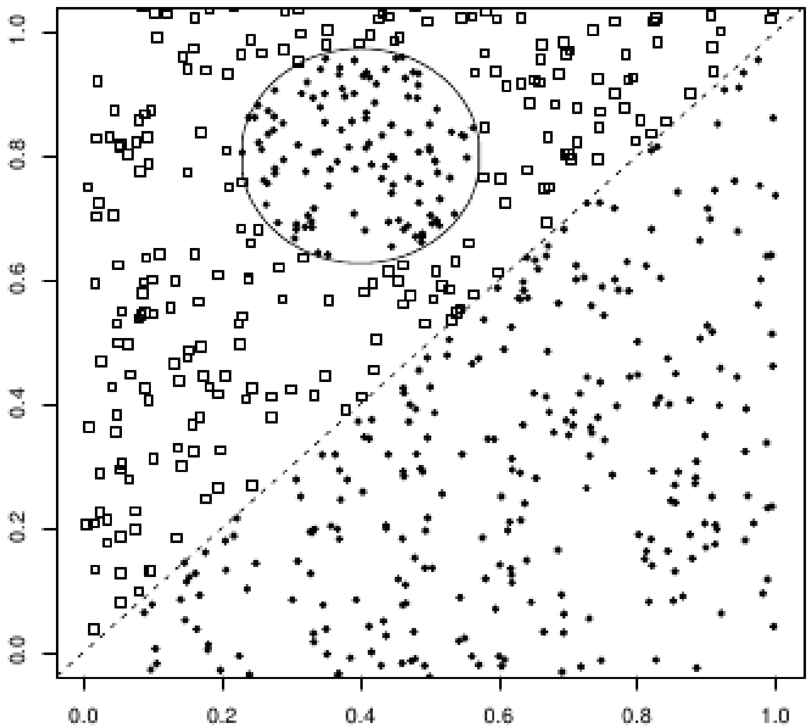
Bias & Variance

- Example for continous (regression) case



Bias & Variance

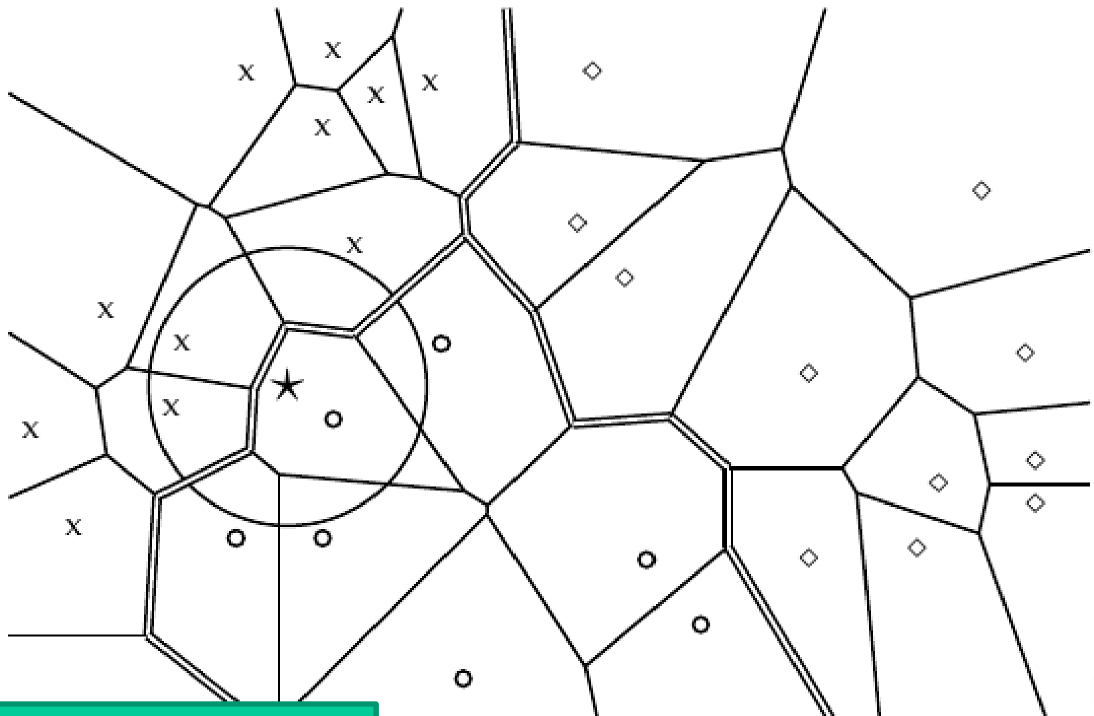
- Linear classifiers: high bias on non-linear problems
  - (Almost) regardless of training data (low variance)



Bias & Variance

- k-nn (non-linear) classifier – low bias (with small k)
- Changes in training set can influence decision boundary greatly

  ➔ high variance *(low stability)*

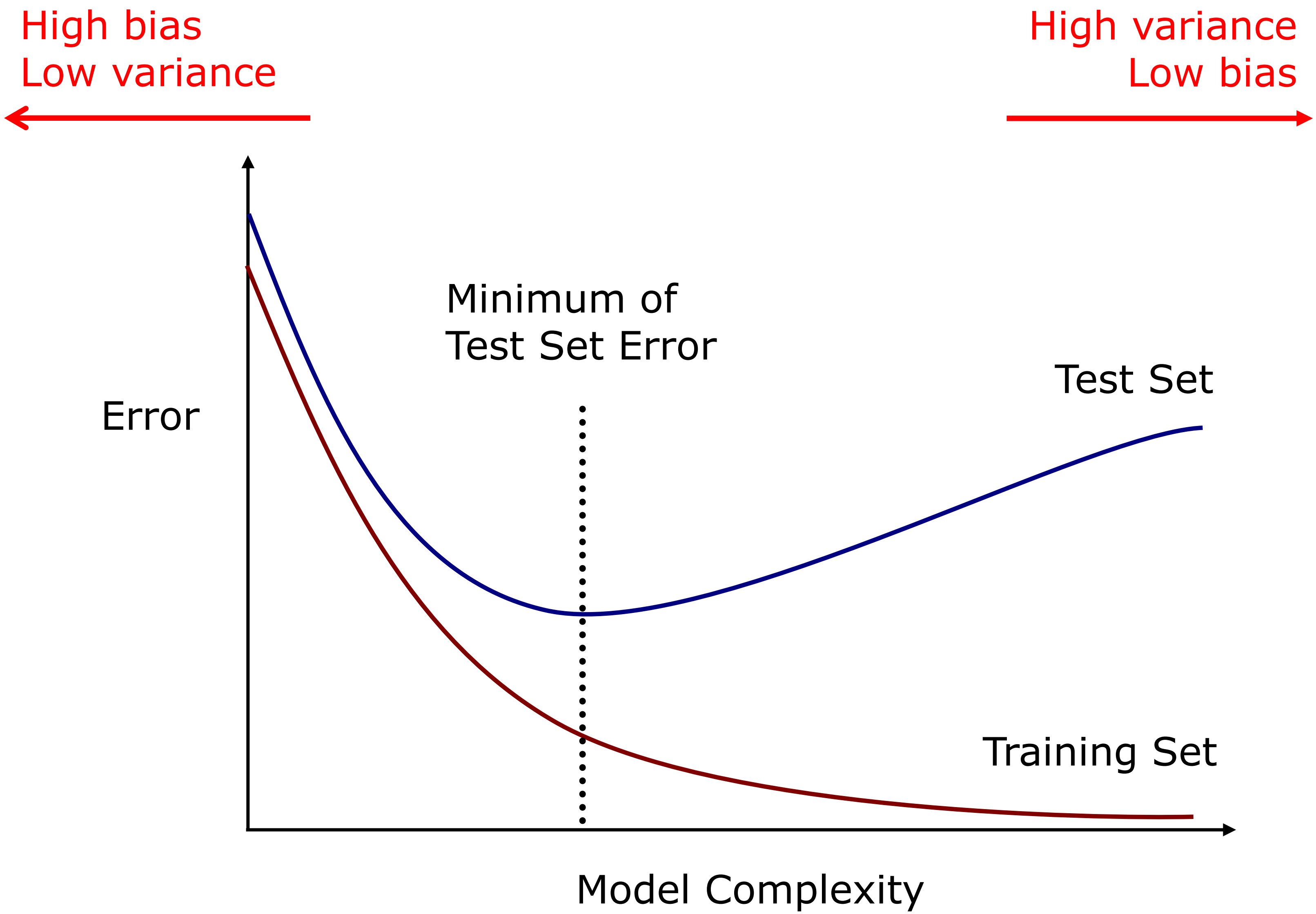


Bias & Variance

- Bias–variance tradeoff
  - Would want model that
  - Accurately captures regularities in training data (low bias)

    AND
  - Generalises well to unseen data (low variance)
  - Typically impossible to do both simultaneously!
    - *cf. precision-recall tradeoff*

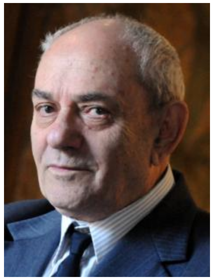
Bias & Variance

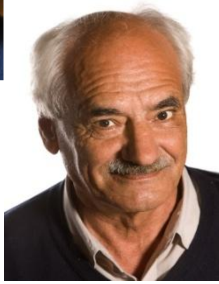


Outline

- Short recap
- Random Forests
- Evaluation
- Support Vector Machines

Support Vector Machines

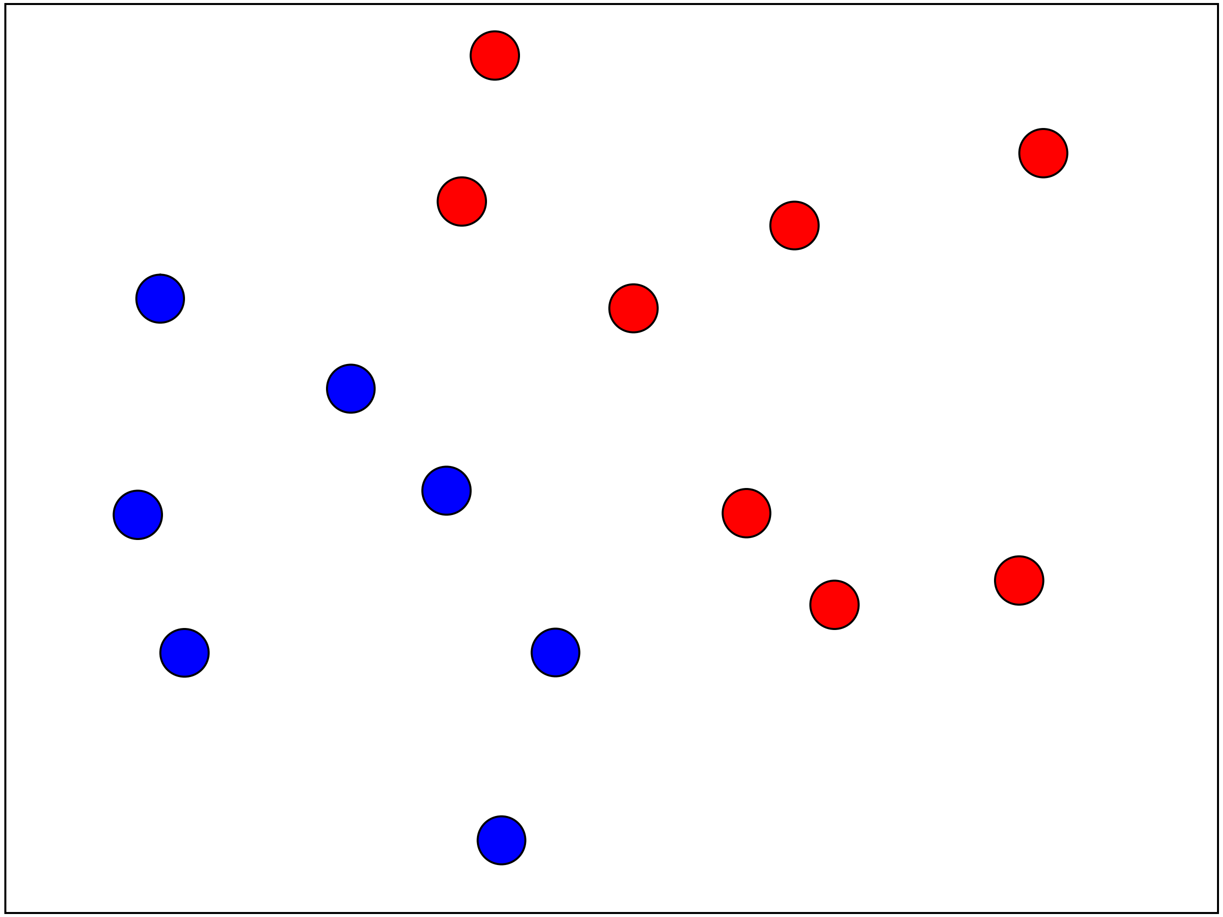




- Introduced by Vladimir Vapnik & Alexey Chervonenkis in early 1960s (!)
  - Kernel-trick added in 1992

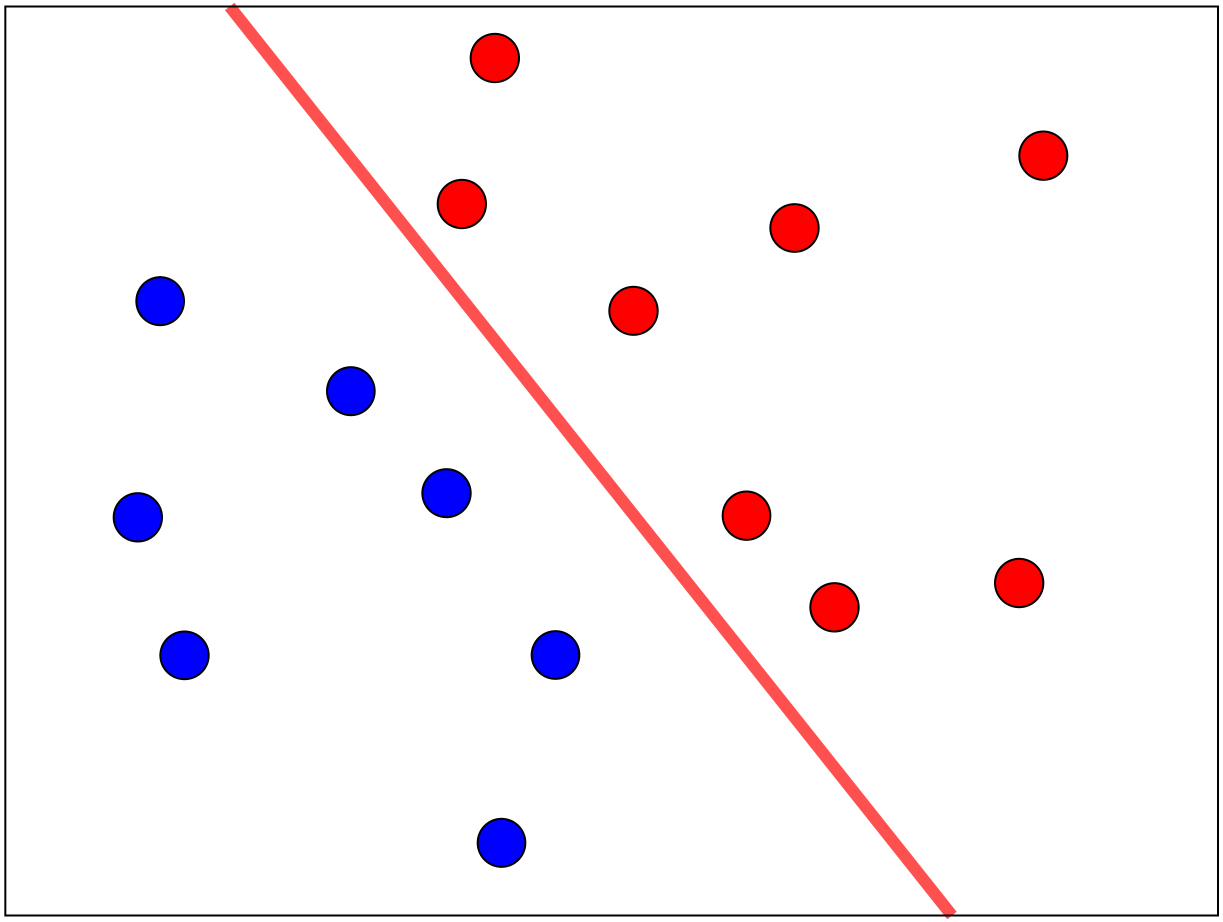
- Also known as **maximum margin** classifiers
- Heavily used & researched in last decade(s)
- Rather sophisticated mathematical model
- Basic concepts
  - Linear separation
  - Optimisation of hyperplane
  - Soft margin & kernel function
    - When linear separation is not possible

Linear separation



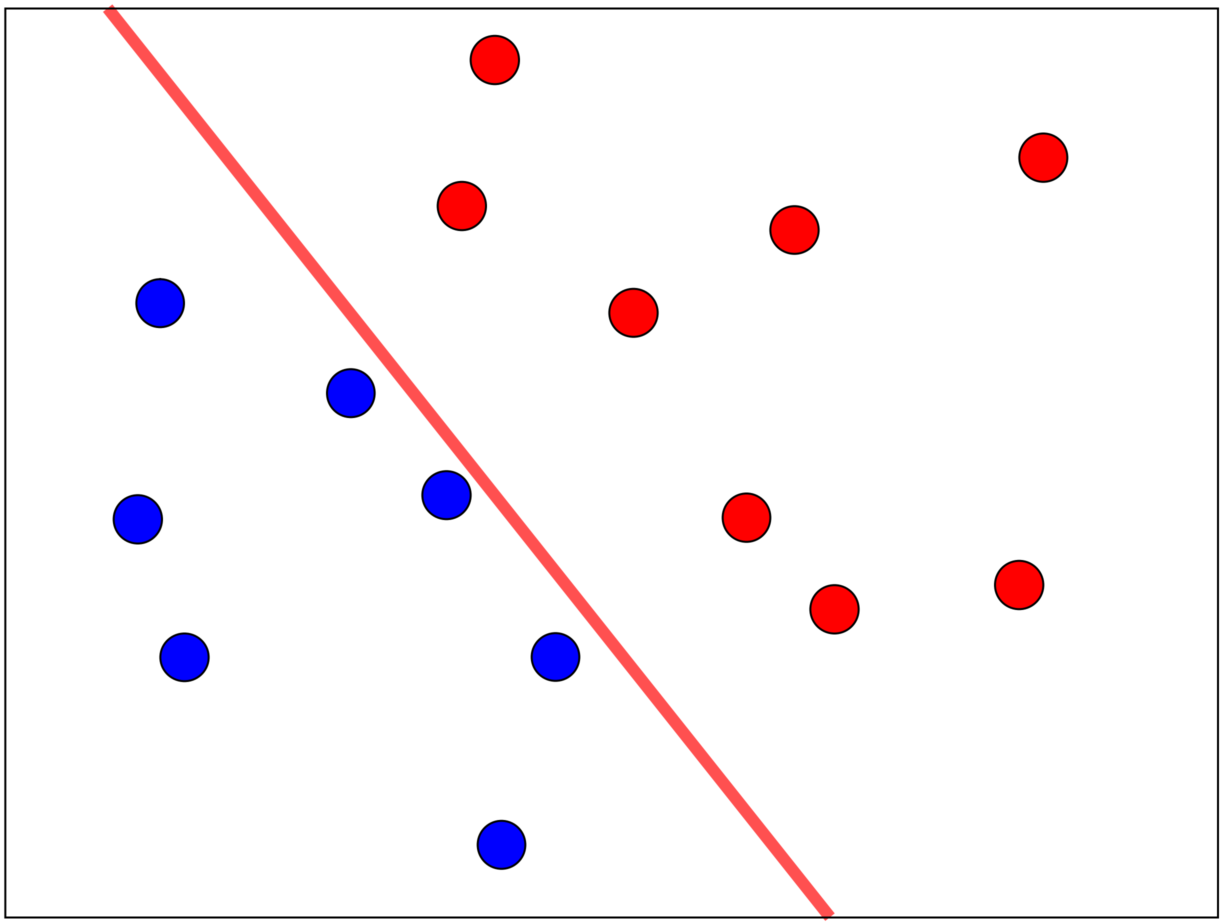
Linear separation

- Separating line #1



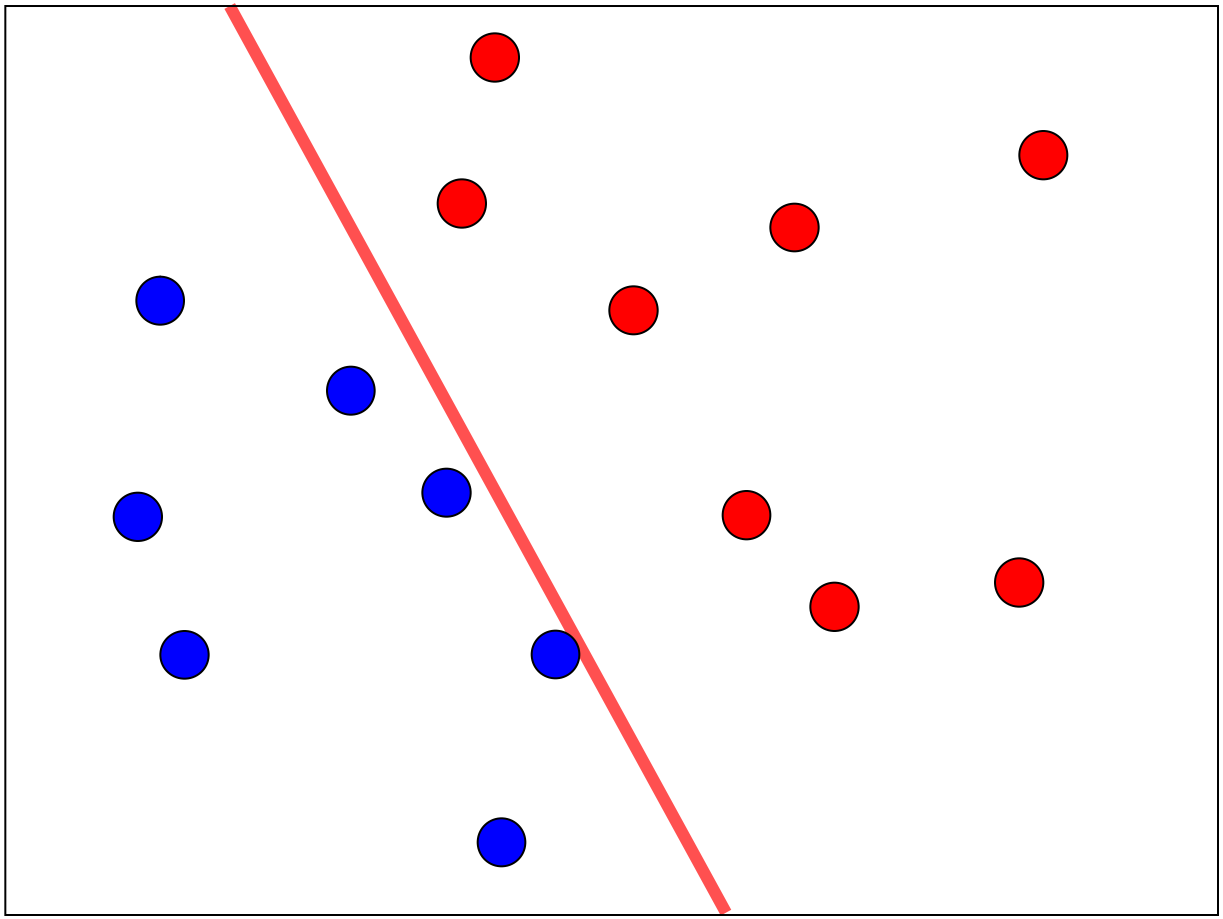
Linear separation

- Separating line #2



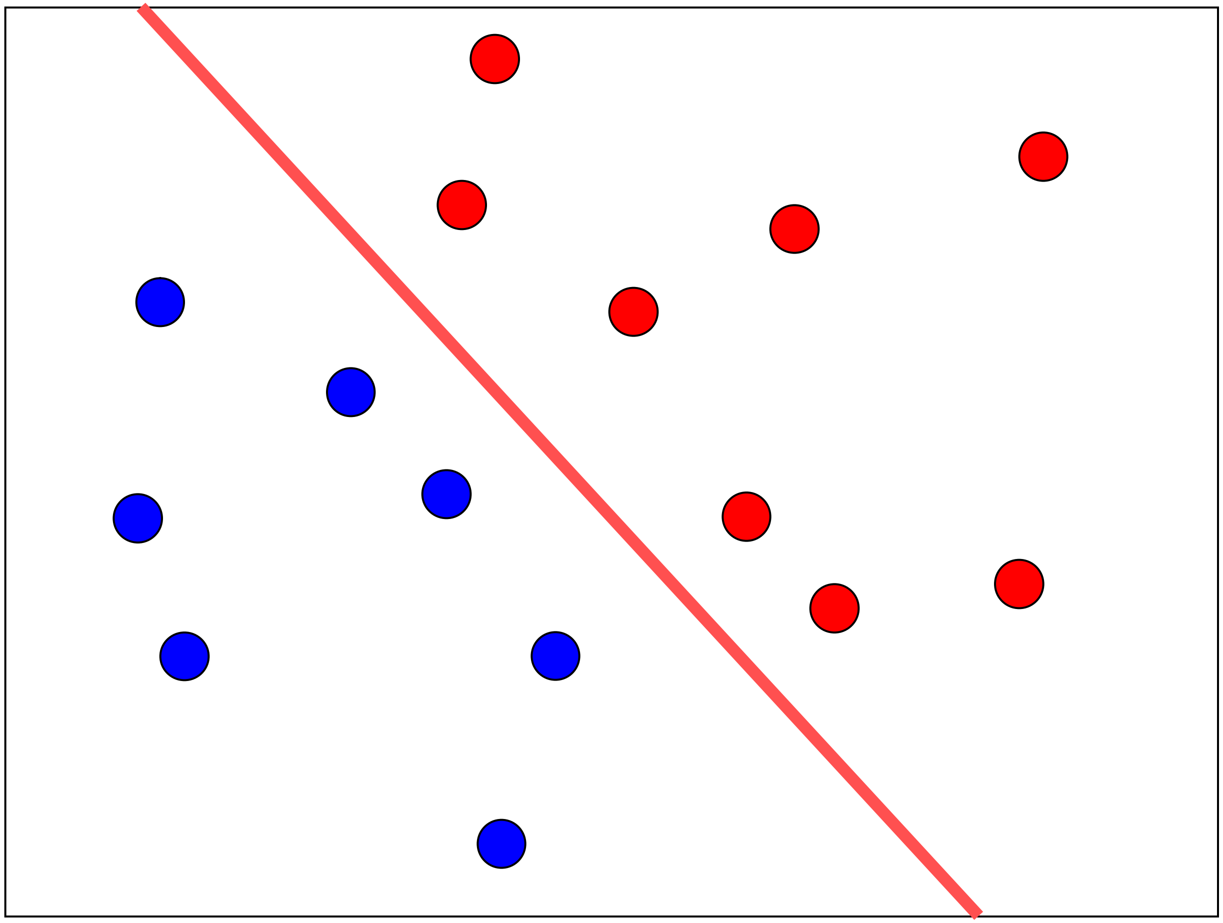
Linear separation

- Separating line #3



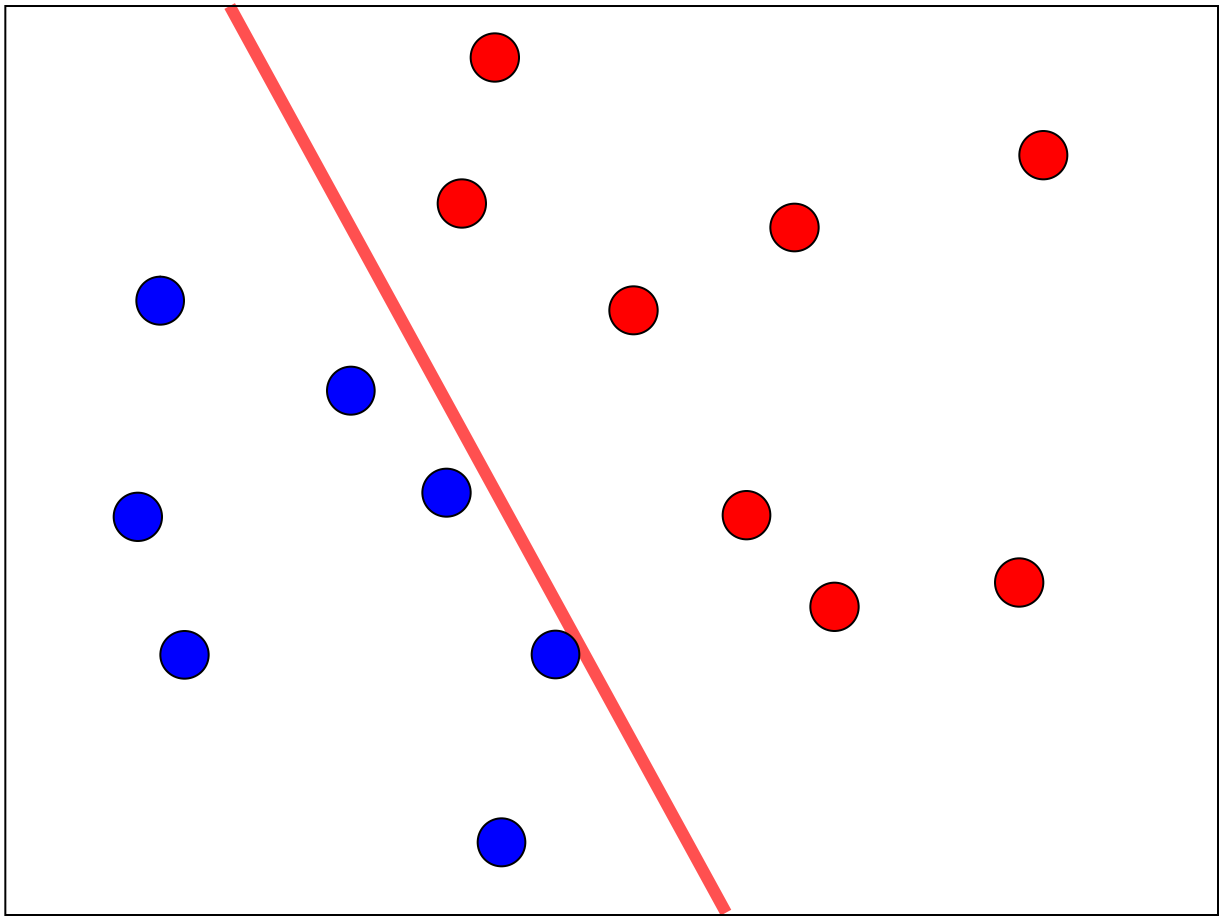
Linear separation

- Separating line #4



Linear separation

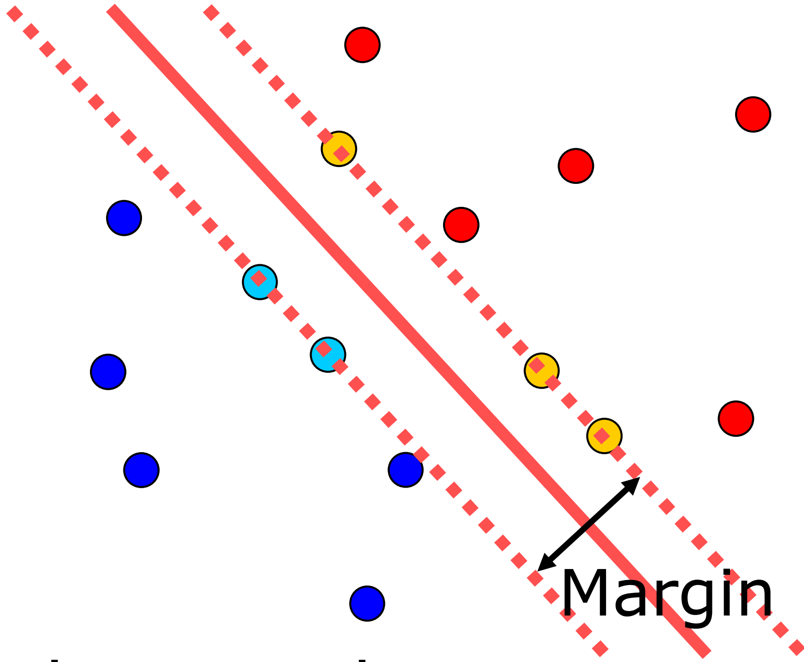
- Separating line #5



Linear Separation

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

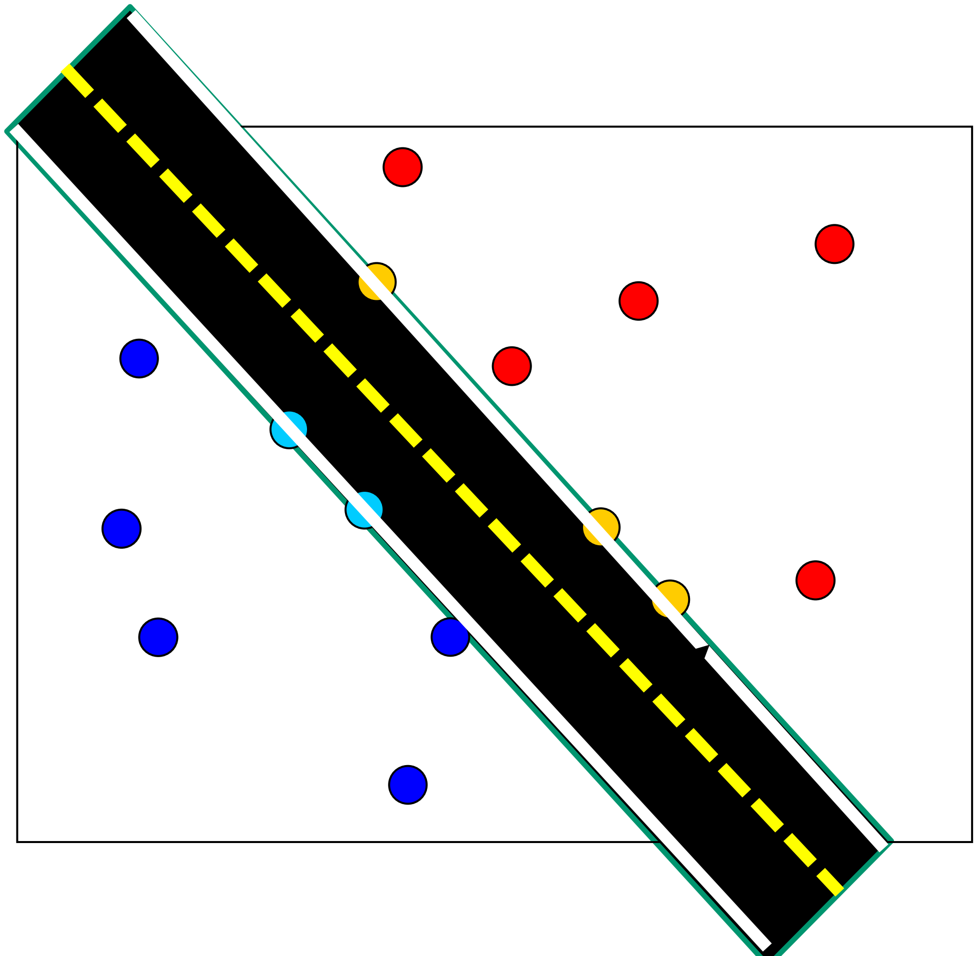
Largest Margin



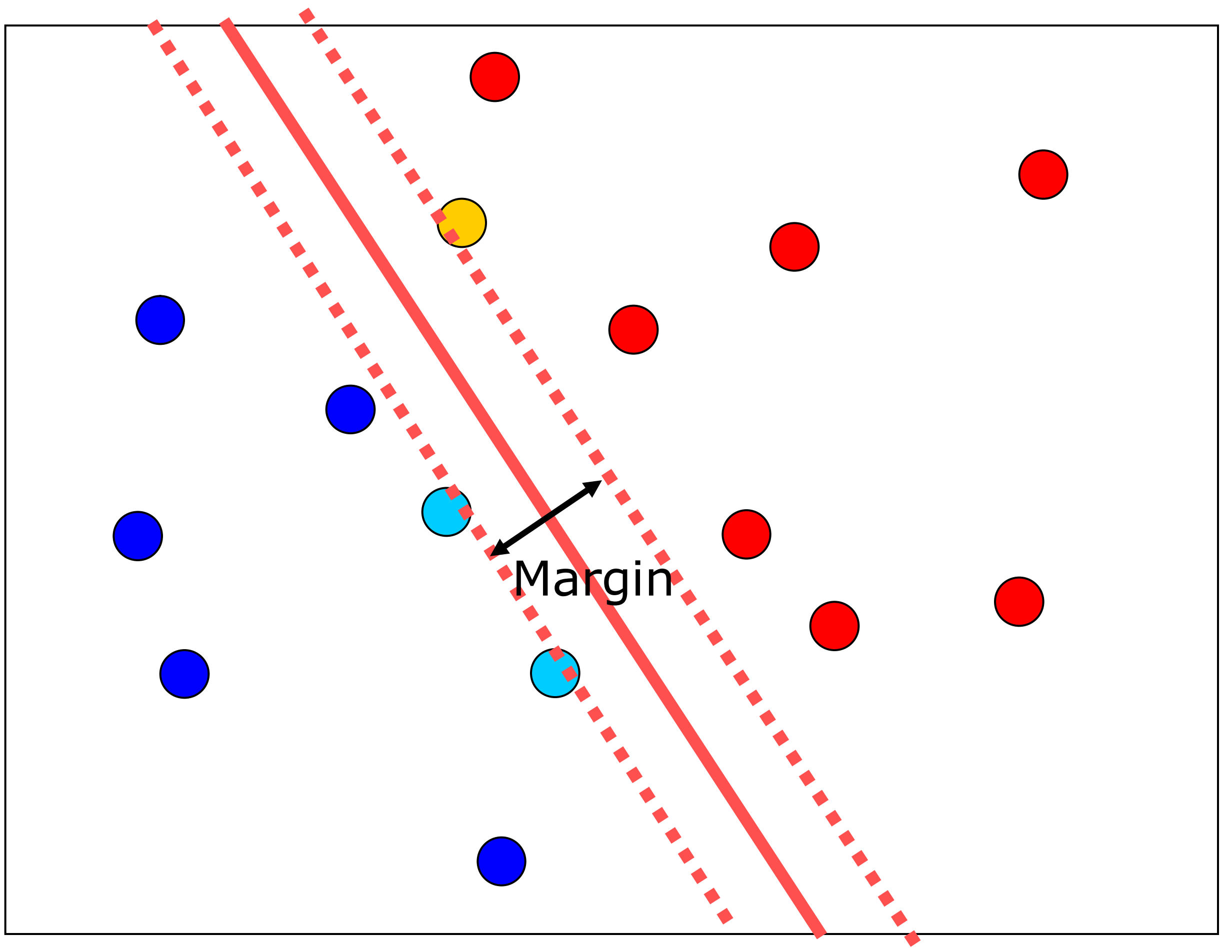
- All separations are valid
  - Which separation is the best?
- **Margin** of a linear classifier: width that boundary could be increased to
  - before hitting any data-point
- **Support Vectors** are those data-points that the margin pushes up against
- *What's the minimum number of support vectors?*

Margin – a simple analogy

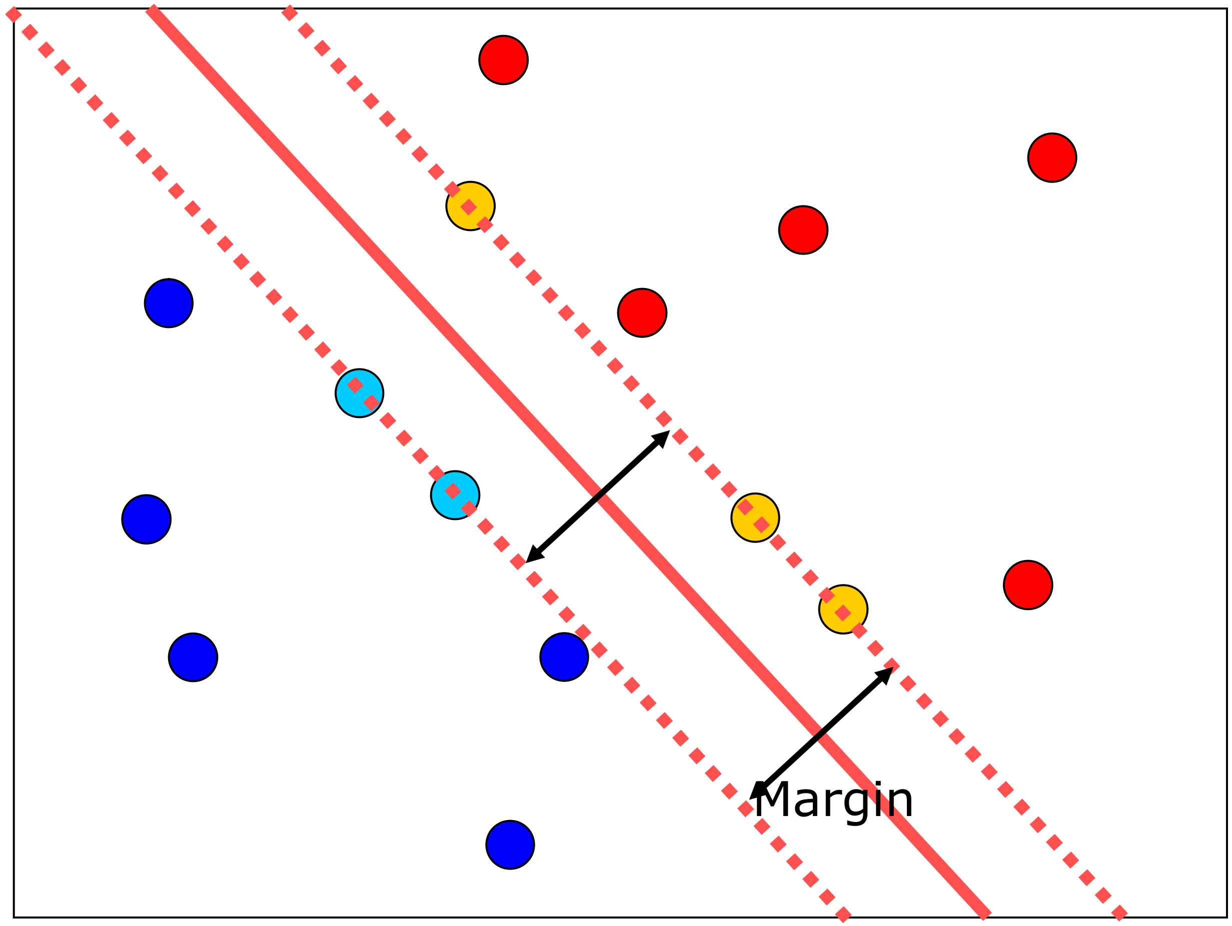
- Margin is a road
  - Decision boundary is the *median separator*
  - Support vectors are *reflector posts*
  - Margin = width of the road



Largest Margin



Largest Margin

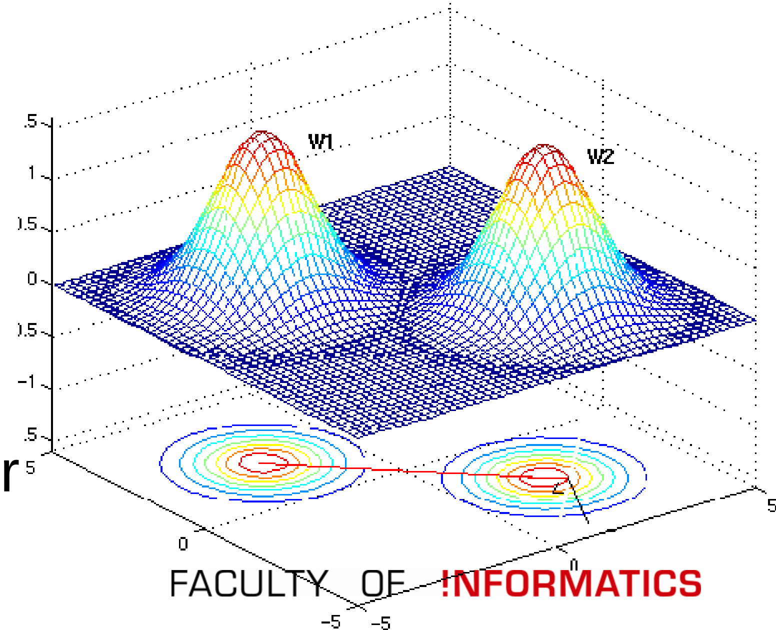


Largest Margin

- *Which separation/margin is the best?*

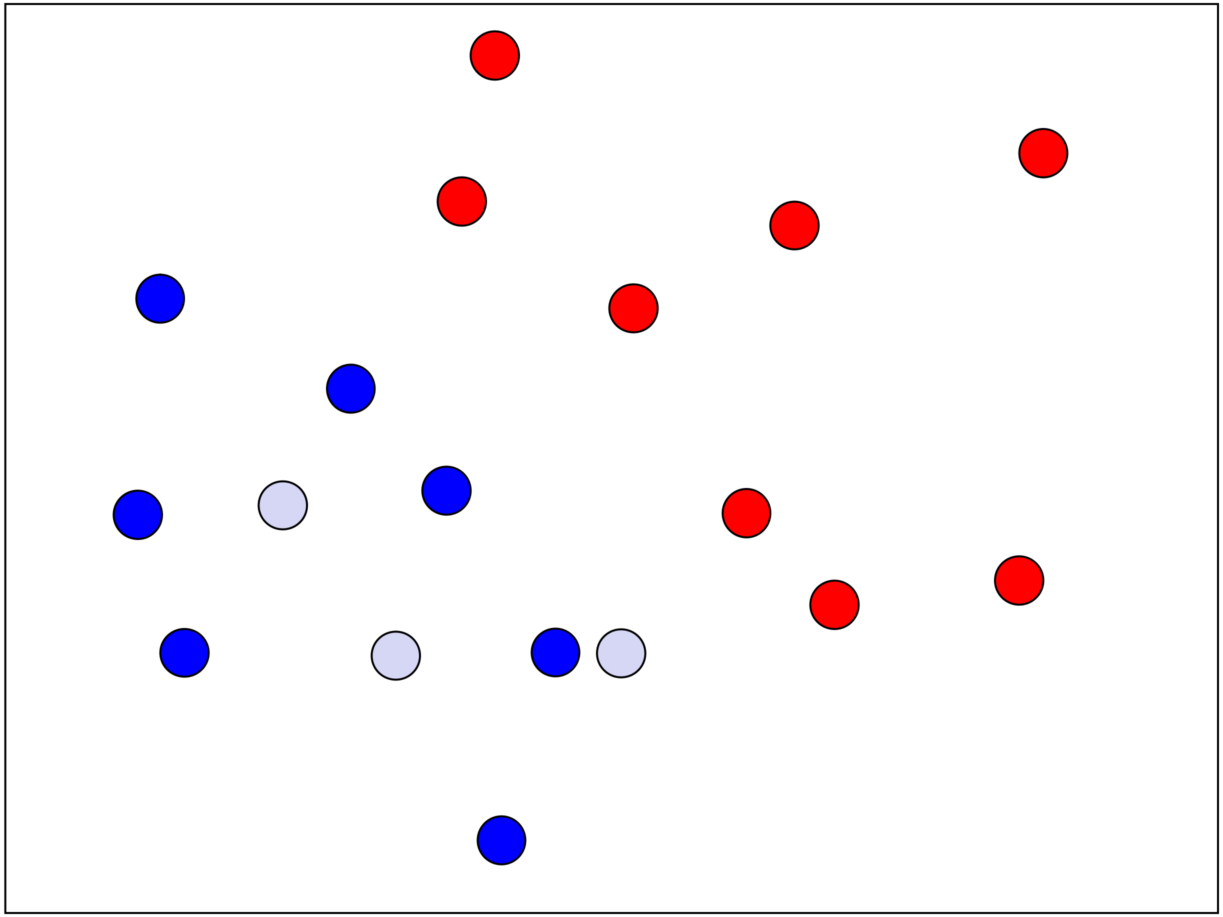
- Claim: **bigger** margin **is better**

- 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)



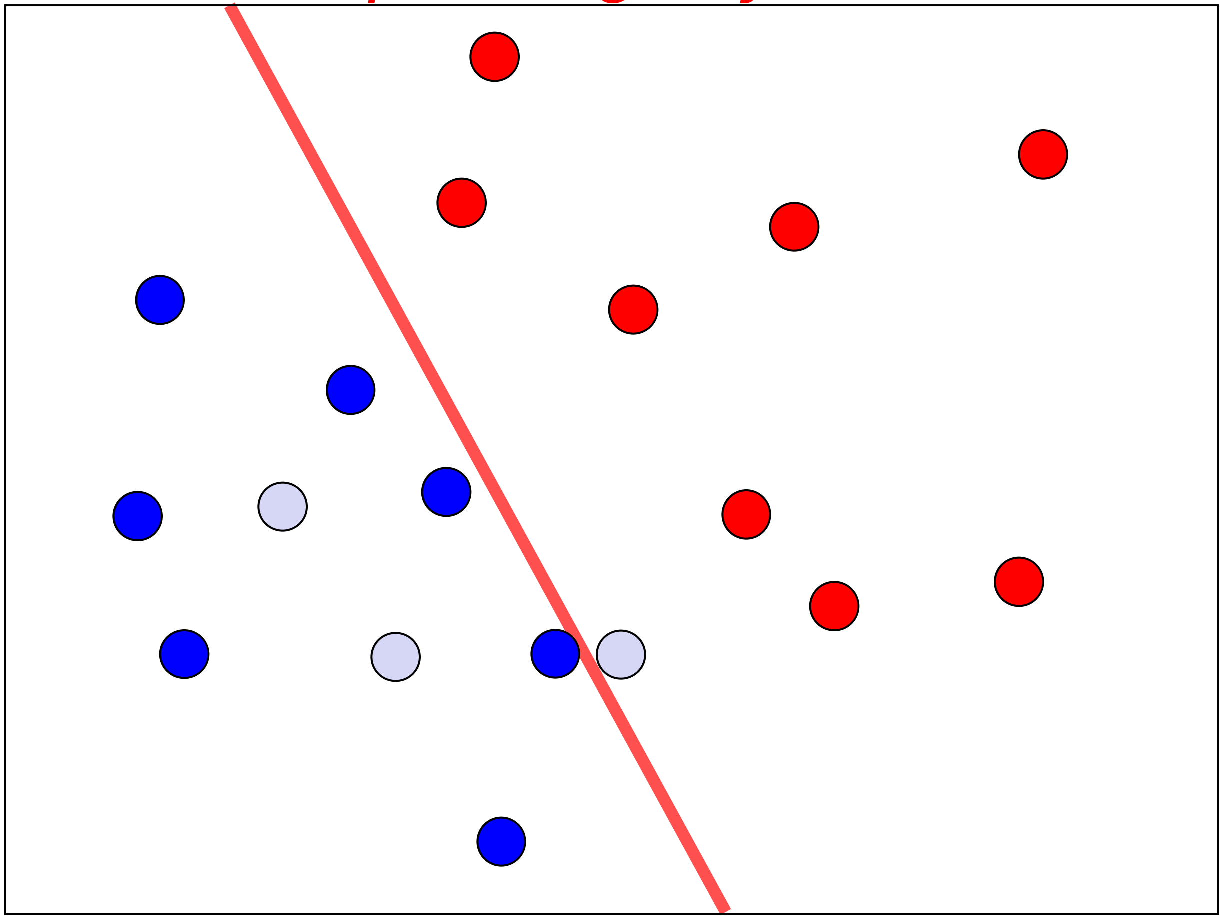
Linear separation

- Draw more samples from the distribution



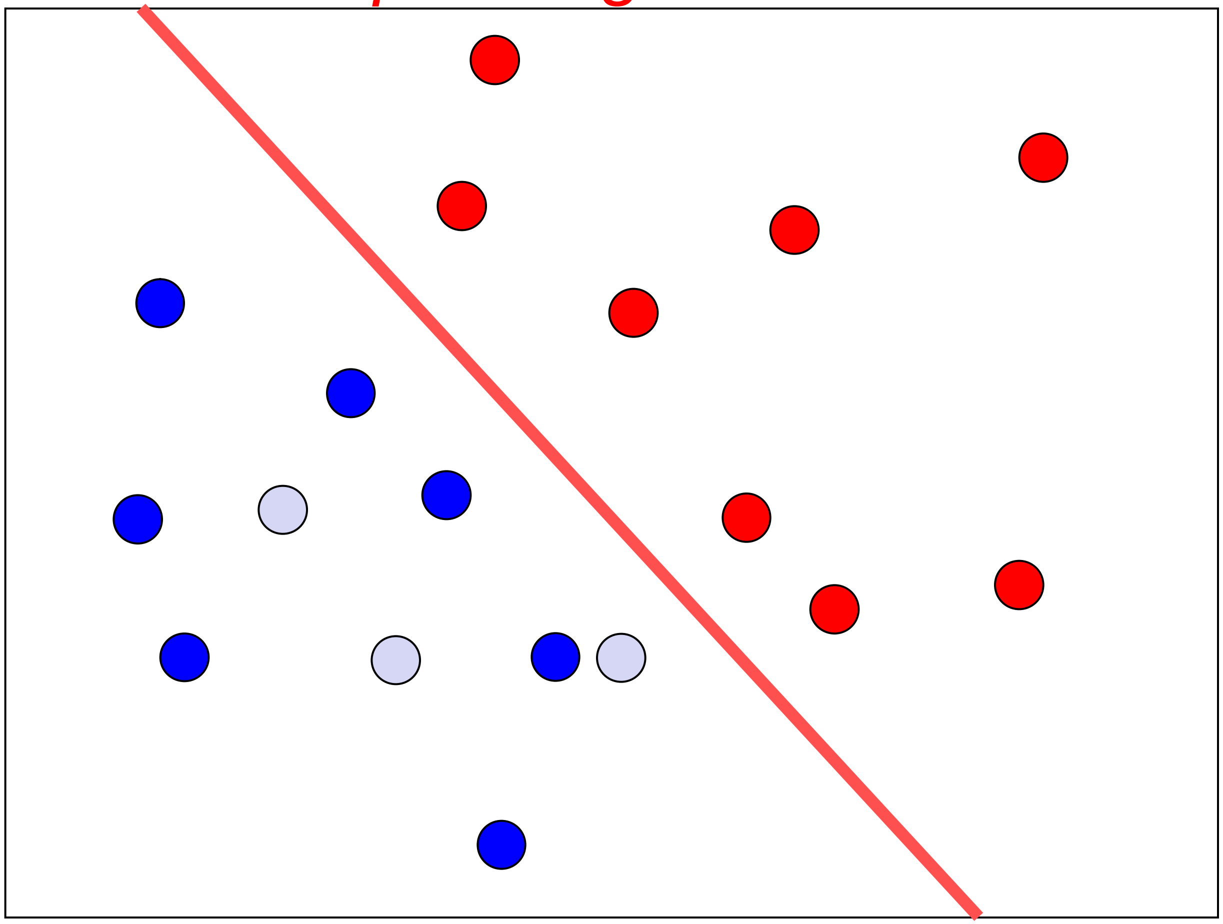
Linear separation

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



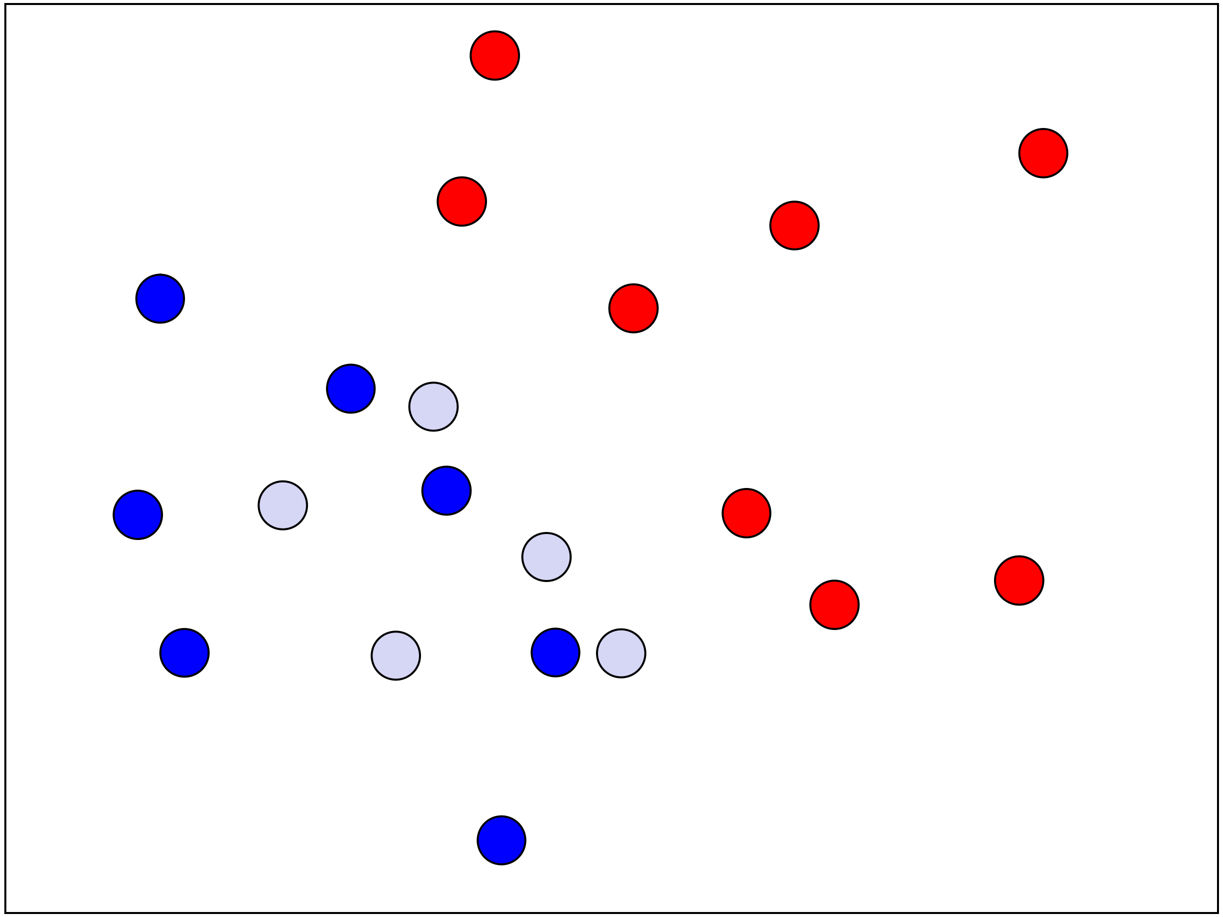
Linear separation

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



Linear separation

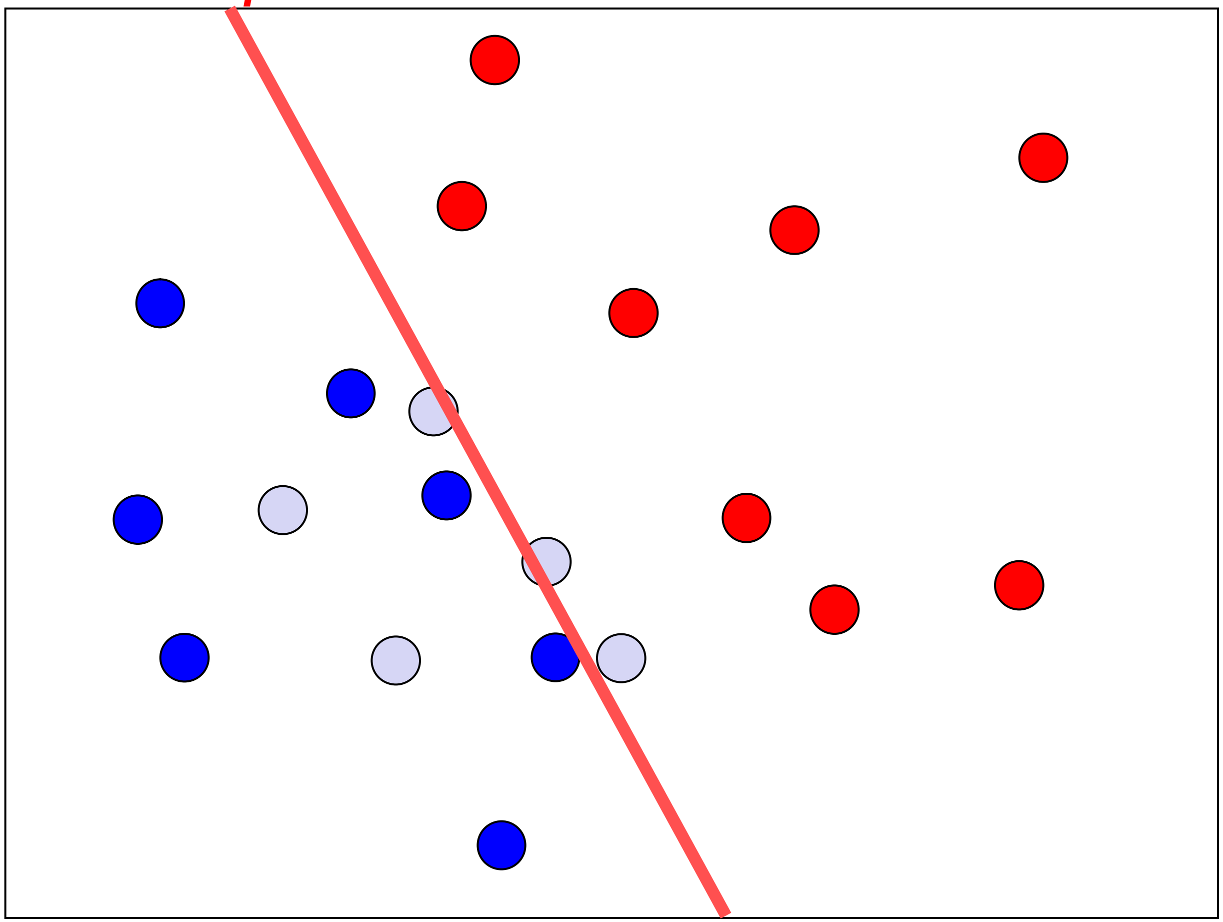
- Draw even more samples from the distribution



Linear separation

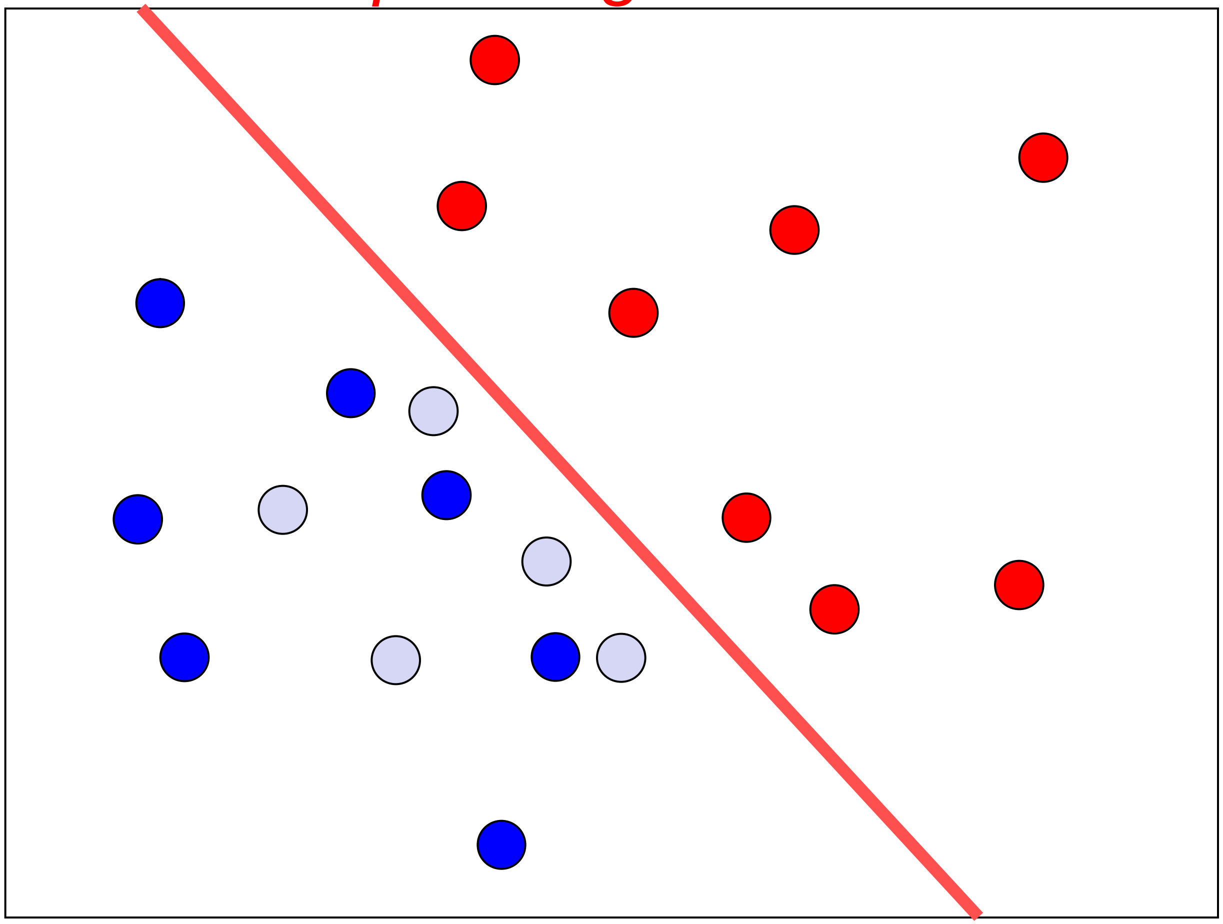
- Draw even more samples from the distribution

➔ *Line #3 separates even worse*



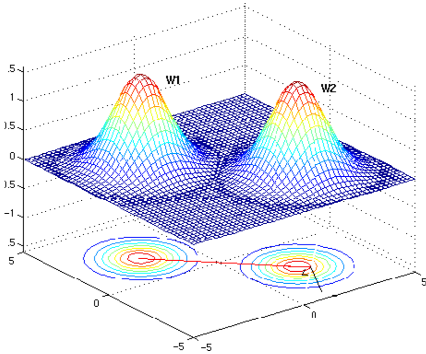
Linear separation

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



Largest Margin

- *Which separation/margin is the best?*
- Claim: **bigger** margin **is better**
- Intuitive demonstration
  - The bigger the margin ➔ the better is the separating plane fitting to slightly different data
  - I.e. less “overfitting”, more generalization



- Later: how to optimise the margin
  - Using Lagrange multipliers, quadratic programming, …

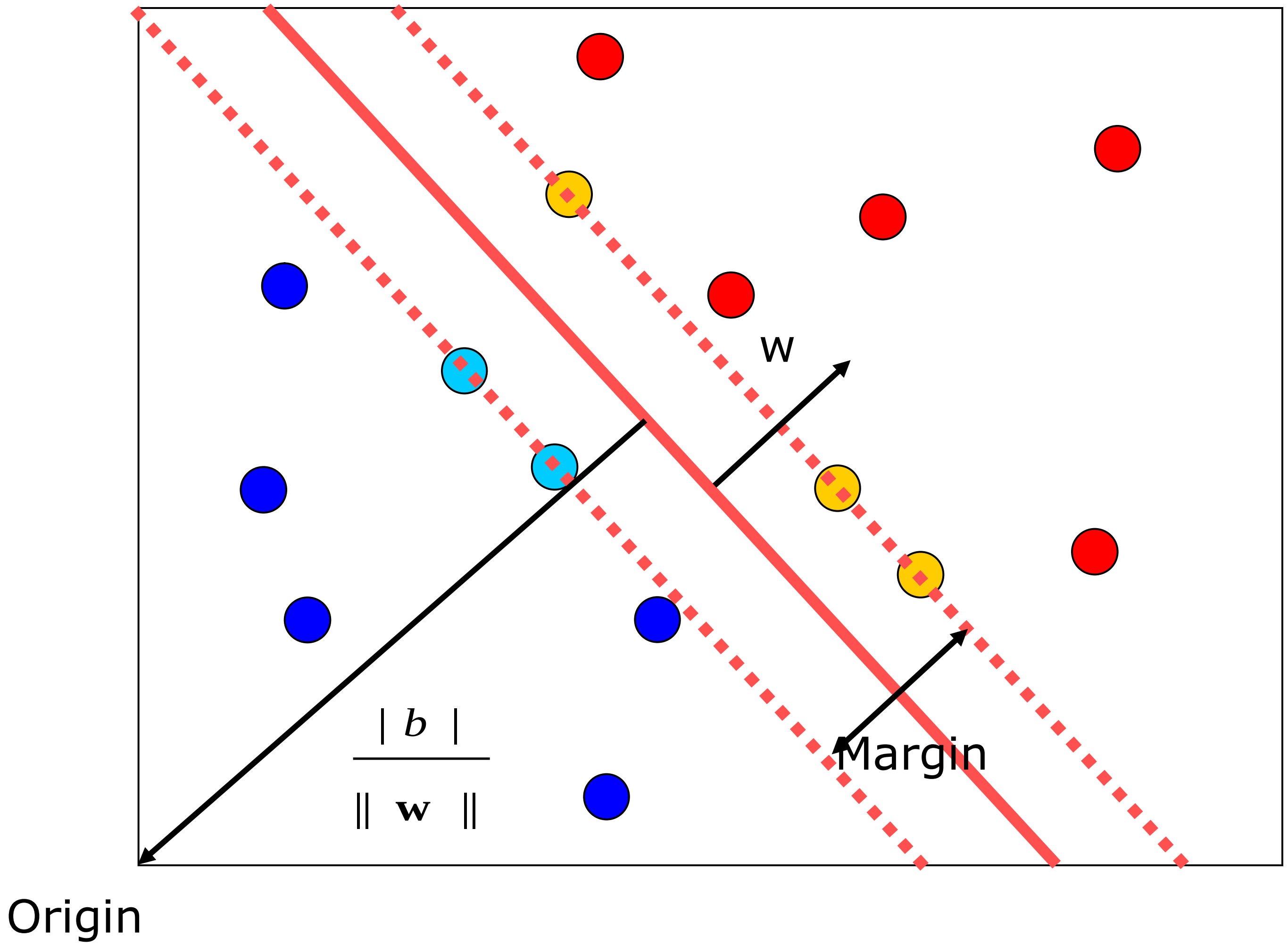
Finding largest margin

- Given set of points $\mathbf{x}_i$ (n-dimensional)
  - belonging to binary class: $y_i = 0$ or $y_i = 1$

- Hyperplane dividing points
  - Satisfying: $\mathbf{w} \cdot \mathbf{x}_i + b = 0$
    - **w** is normal to the hyperplane (weight vector)
    - b determines offset from origin (intercept/bias)
    - $\frac{|b|}{\|\mathbf{w}\|}$ perpendicular (normal) distance from origin

  - In general: *infinite number of possibilities*

Finding largest margin



Finding largest margin

- Recap *Perceptron*: Activation $a = \sum_{i=1}^{n} w_i x_i$

  Classification $f(x) = \begin{cases} 1 \text{ if } a \geq 0 \\ 0 \text{ if } a < 0 \end{cases}$

- SVM: often ≥ +1 / < -1
- Classification function: $f(\vec{x}) = sign(\vec{w}^T \vec{x}_i + b)$
  - Alternative formulation: $f(x) = \begin{matrix} 1 \text{ if } a \geq 1 \\ 0 \text{ if } a < -1 \end{matrix}$
- Hyperplane dividing points in -1 & +1
  - Satisfying: $\mathbf{w} \cdot \mathbf{x}_i + b = 0$

    with $\vec{a} \bullet \vec{b} = \vec{a}^T \vec{b}$

    $\Rightarrow \vec{w}^T \vec{x}_i = -b$

Finding largest margin

- Task: find hyperplane with *largest margin*

  ➔ minimise $\|\mathbf{w}\|$ *(will show why)*

  – Given the *constraints*

  $\mathbf{w} \cdot \mathbf{x}_i + b \geq +1$ for $\mathbf{y}_i = +1$ (1)

  $\mathbf{w} \cdot \mathbf{x}_i + b \leq -1$ for $\mathbf{y}_i = -1$ (2)

  – Constraints (1) and (2) combined:

  $y_i\,(\mathbf{w} \cdot \mathbf{x}_i + b) - 1 \geq 0$ for all i

  - *if $y_i = +1$:* $\mathbf{w} \cdot \mathbf{x}_i + b - 1 \geq 0$ ➔ $\mathbf{w} \cdot \mathbf{x}_i + b \geq 1$
  - *if $y_i = -1$:* $(-1)\,(\mathbf{w} \cdot \mathbf{x}_i + b) - 1 \geq 0$ ➔ $(-1)\,(\mathbf{w} \cdot \mathbf{x}_i + b) \geq 1$

    ➔ $\mathbf{w} \cdot \mathbf{x}_i + b \leq -1$

Finding largest margin

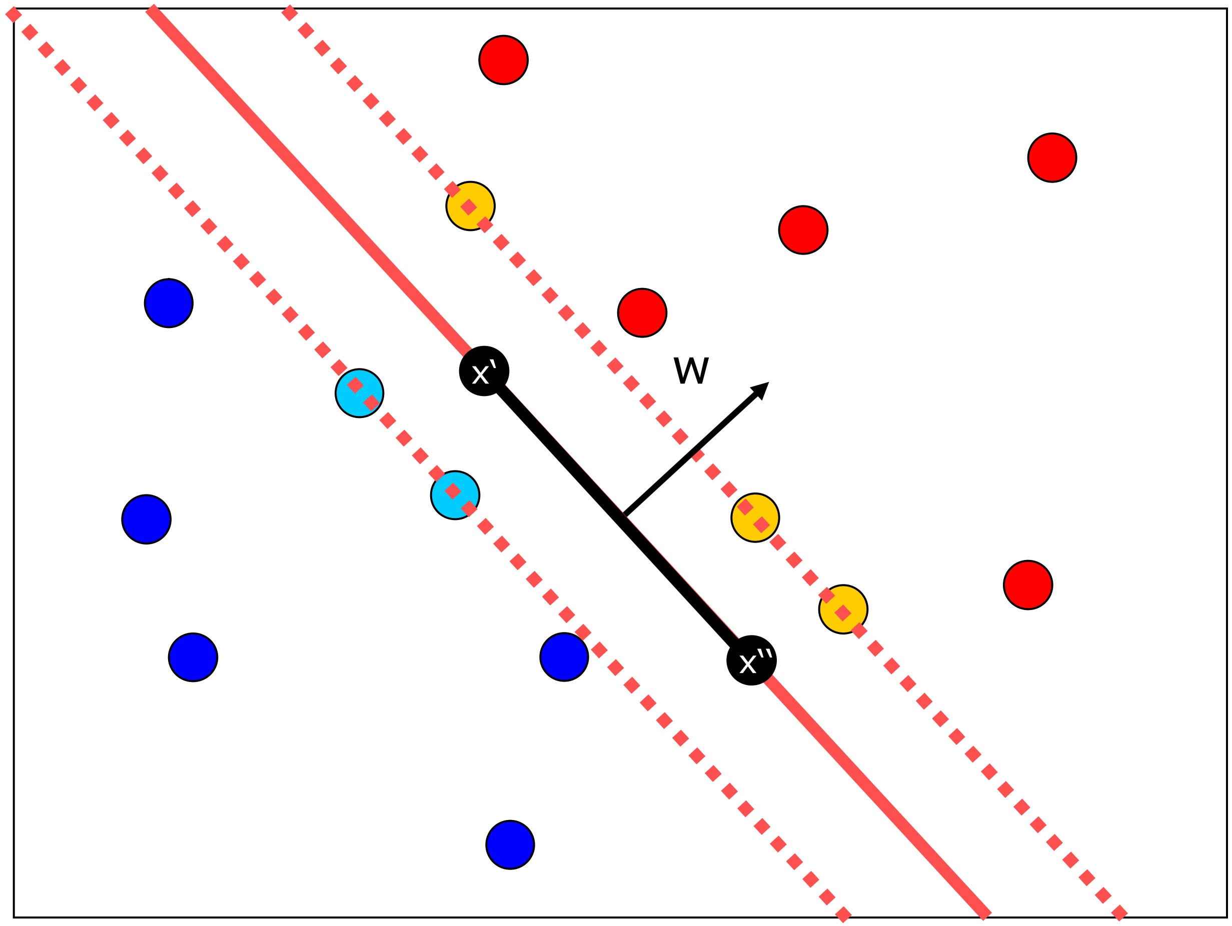
- Hyperplane dividing points in -1 & +1
  - Satisfying: $\mathbf{w} \cdot \mathbf{x}_i + b = 0$
    - ➔ ***w** is normal to the hyperplane*

- Demonstration: pick 2 points x‘, x‘‘on the plane, thus
  - $\mathbf{w} \cdot \mathbf{x'} + b = 0$ and $\mathbf{w} \cdot \mathbf{x''} + b = 0$

  ➔ $\mathbf{w} \cdot (\mathbf{x'} - \mathbf{x''}) = 0$

  ➔ **w** is normal to the plane
    - as it is normal to any vector on the plane
    - and x‘x‘‘ is a vector on the plane

Finding largest margin



Finding largest margin

- Can decompose margin into two components

- d_+: shortest distance from hyperplane to closest *positive* sample ($y_i = +1$)
- d_-: shortest distance from hyperplane to closest *negative* sample ($y_i = -1$)

- Width of margin: $d_+ + d_-$

Finding largest margin

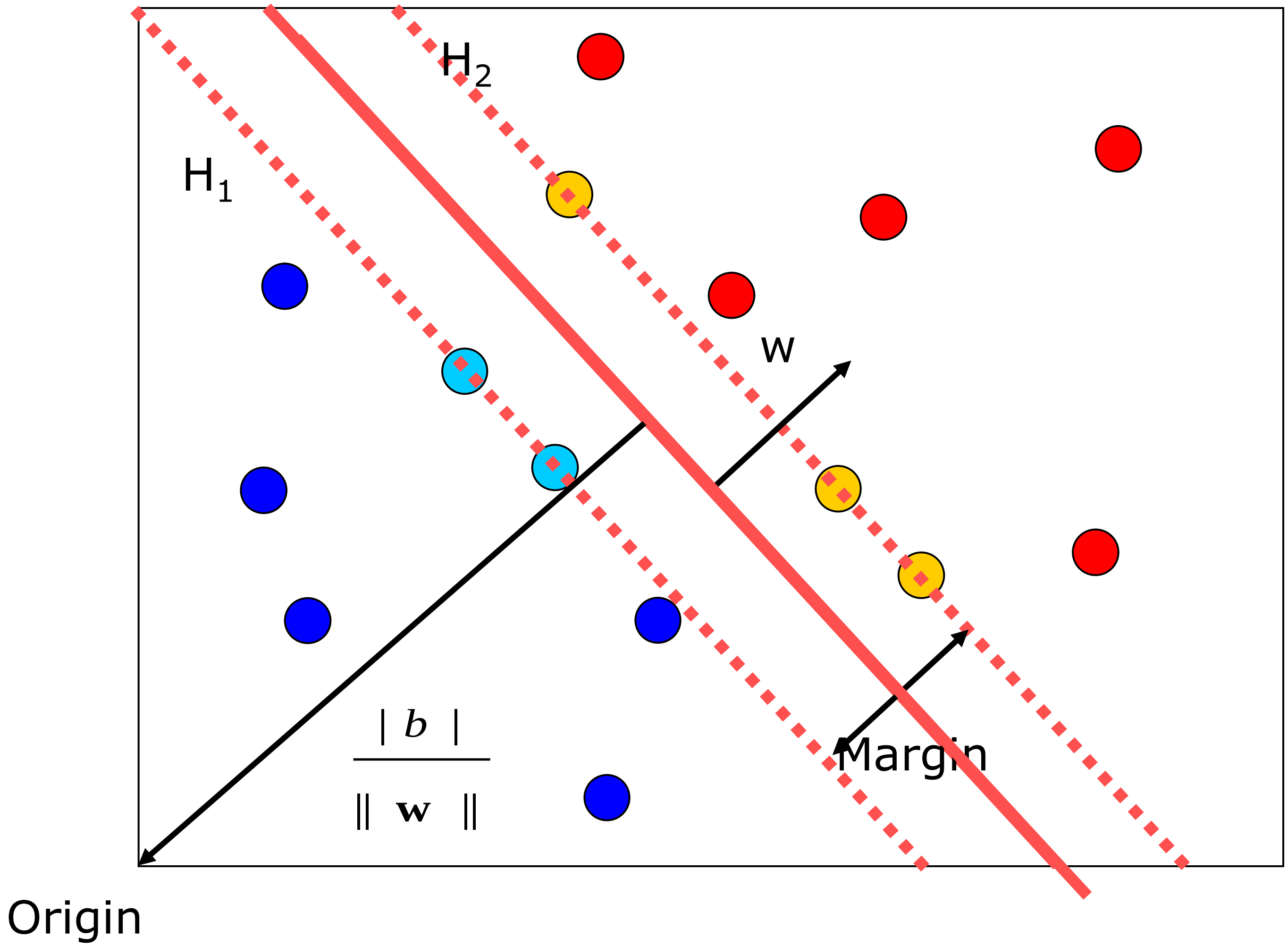
- Points for which equality in (1) holds
  - Points on hyperplane H_1: $\mathbf{w} \cdot \mathbf{x}_i + b = 1$
  - Distance from origin: $\frac{|1 - b|}{\|\mathbf{w}\|}$

- Points for which equality in (2) holds
  - Points on hyperplane H_2: $\mathbf{w} \cdot \mathbf{x}_i + b = -1$
  - Distance from origin: $\frac{|-1 - b|}{\|\mathbf{w}\|}$

- Optimisation: find H_1 and H_2, described by

  $\mathbf{w} \cdot \mathbf{x}_i + b = 1$ and $\mathbf{w} \cdot \mathbf{x}_i + b = -1$

Finding largest margin



Finding largest margin

- *How to find the largest margin?*
- Analytical solution, e.g. via quadratic programming and Langrage multipliers
  - E.g. SMO (Sequential Minimal Optimisation) algorithm
    - 1998 by John Platt (Microsoft Research)
    - Breaks the optimisation in a series of sub-problems
    - Used e.g. in WEKA, LibSVM, ...
- Heuristic Optimisation
  - Sub-gradient descent, coordinate descent
    - Sub-gradient when there are many training examples
    - Coordinate descent when the dimensionality is high
- *Difference in the two approaches?*

Largest Margin: Analytical

- After quite some math, using Lagrange Multipliers

$$L_p = \sum_{1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j x_i \cdot x_j$$

- and *Convex* quadratic programming, we obtain

$$\mathbf{w} = \sum_{i=1}^{n} \alpha_i y_i x_i$$

$$-b = \mathbf{w} \cdot \mathbf{x}_i - y_i$$

  – Depends on dot-product of $\mathbf{x}_i$
  – Quadratic programming complexity increases with n^2
- “Quite some Math“: tutorial in TUWEL
  – Discussed in a later lecture (time permitting)

Outline

- Short recap
- Random Forests
- Evaluation
- Support Vector Machines: soft margin

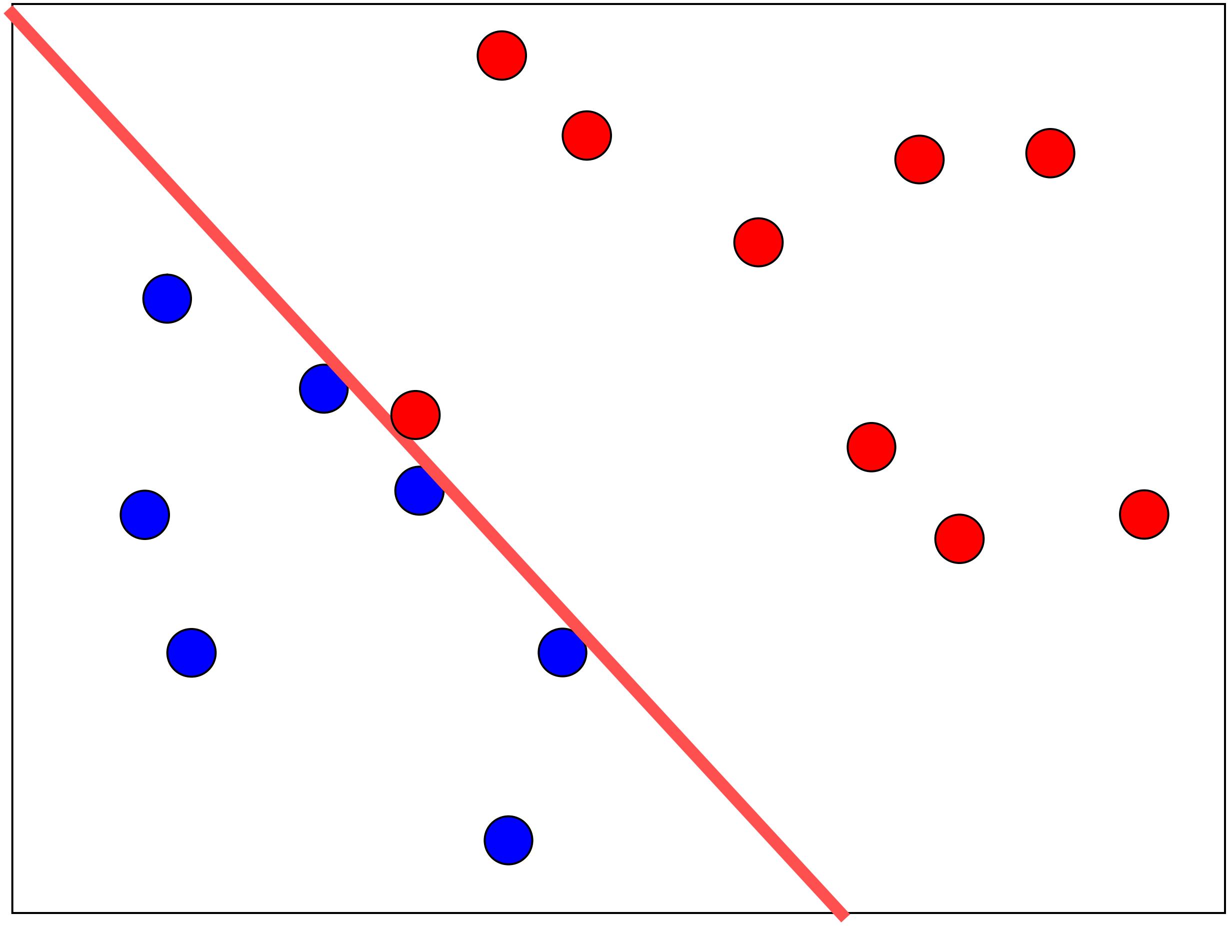
Soft margin

- SVMs optimise the decision boundary ...
- ... but still rely on ***linear separation*** !

1. Sometimes linear separation not possible

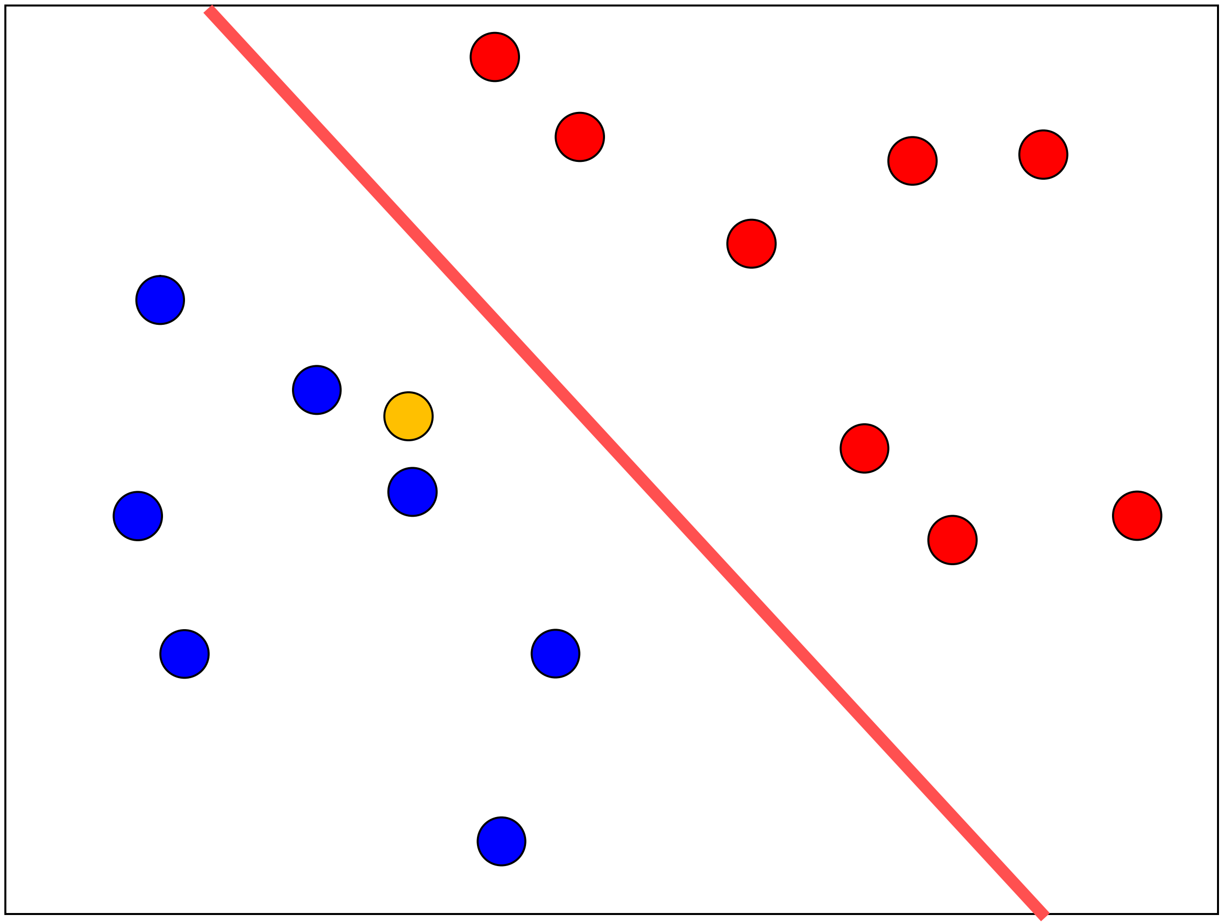
2. Sometimes, linear separation would lead to a badly generalising model
   – When?

Bad generalisation

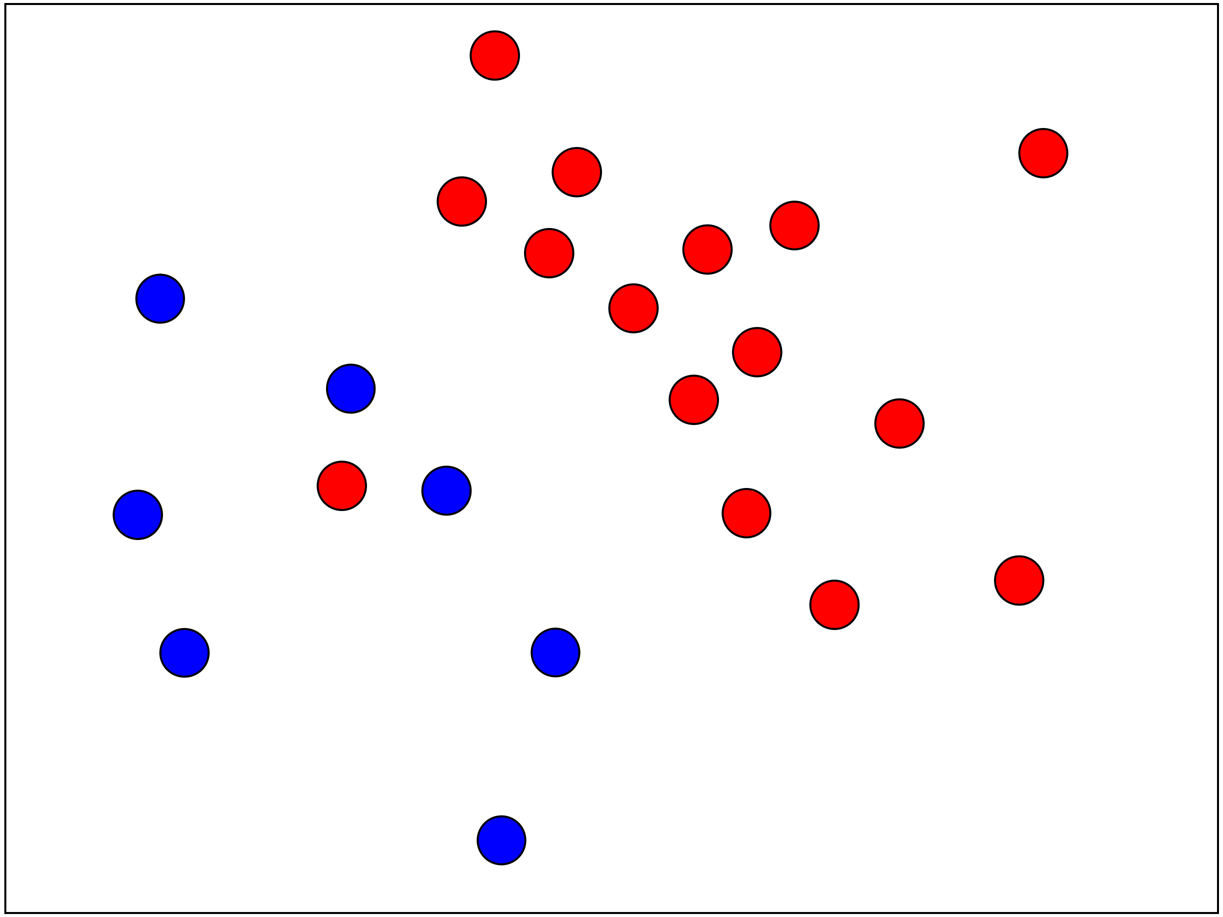


Bad generalisation

- *Wider margin might be better*

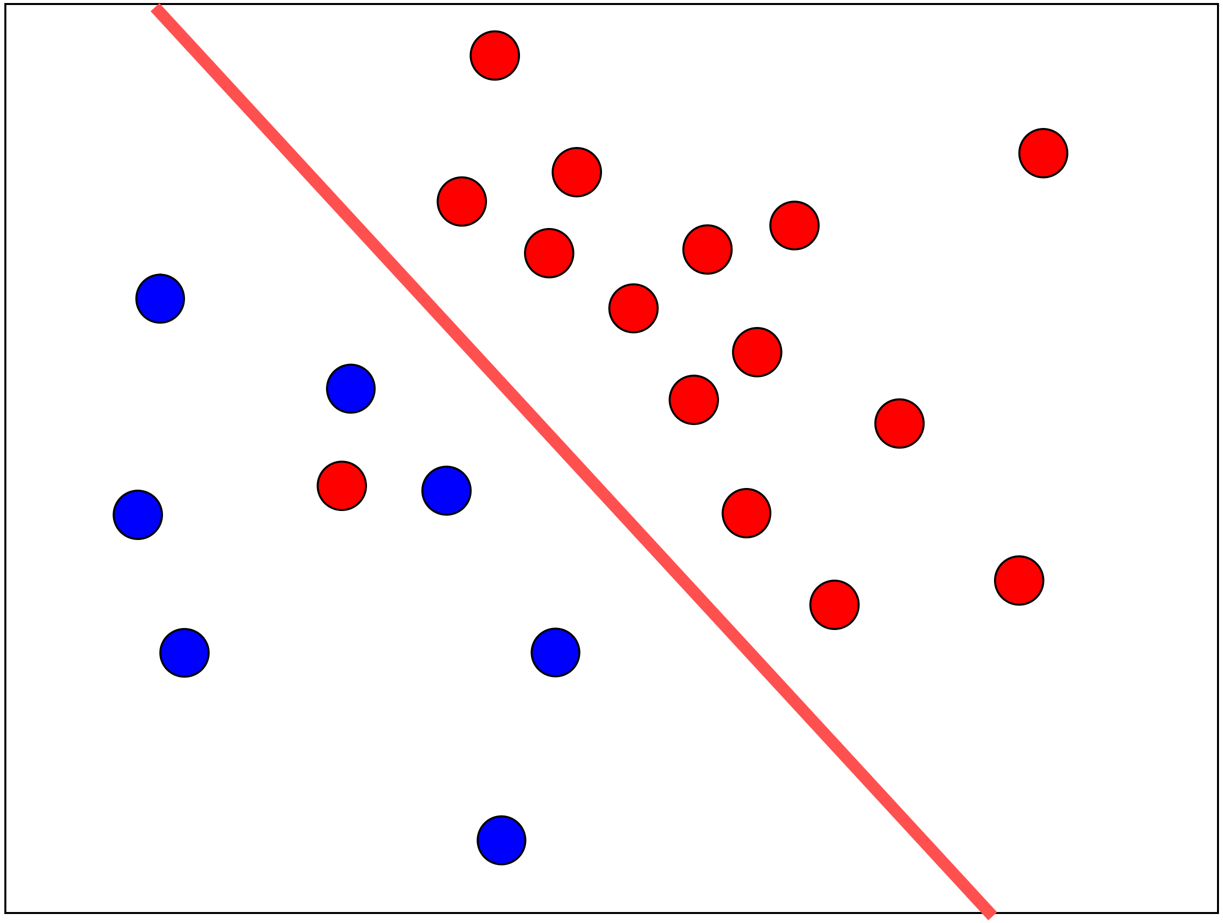


Linear separation not possible



Linear separation not possible

- *“Acceptable” hyperplane could still be found*



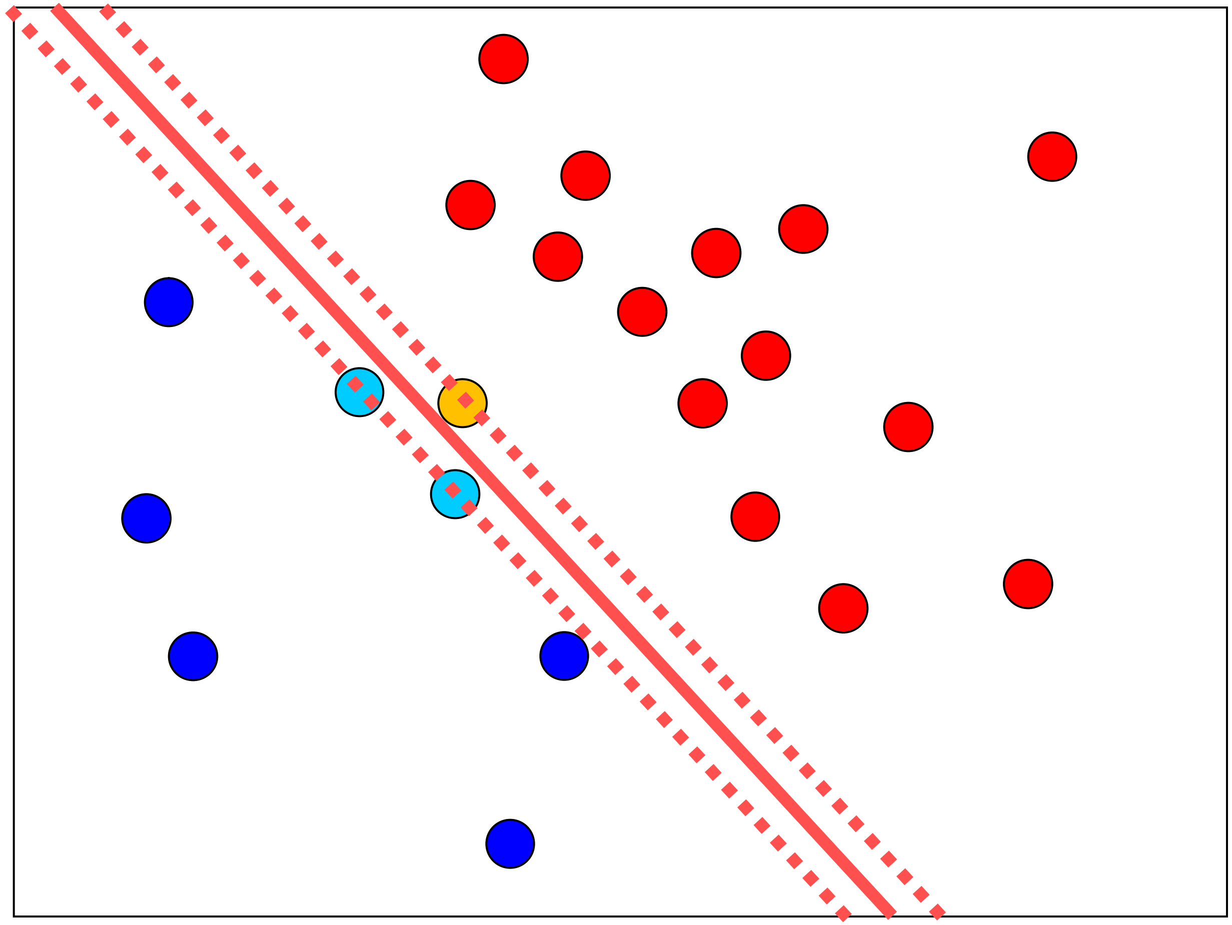
Soft margin

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

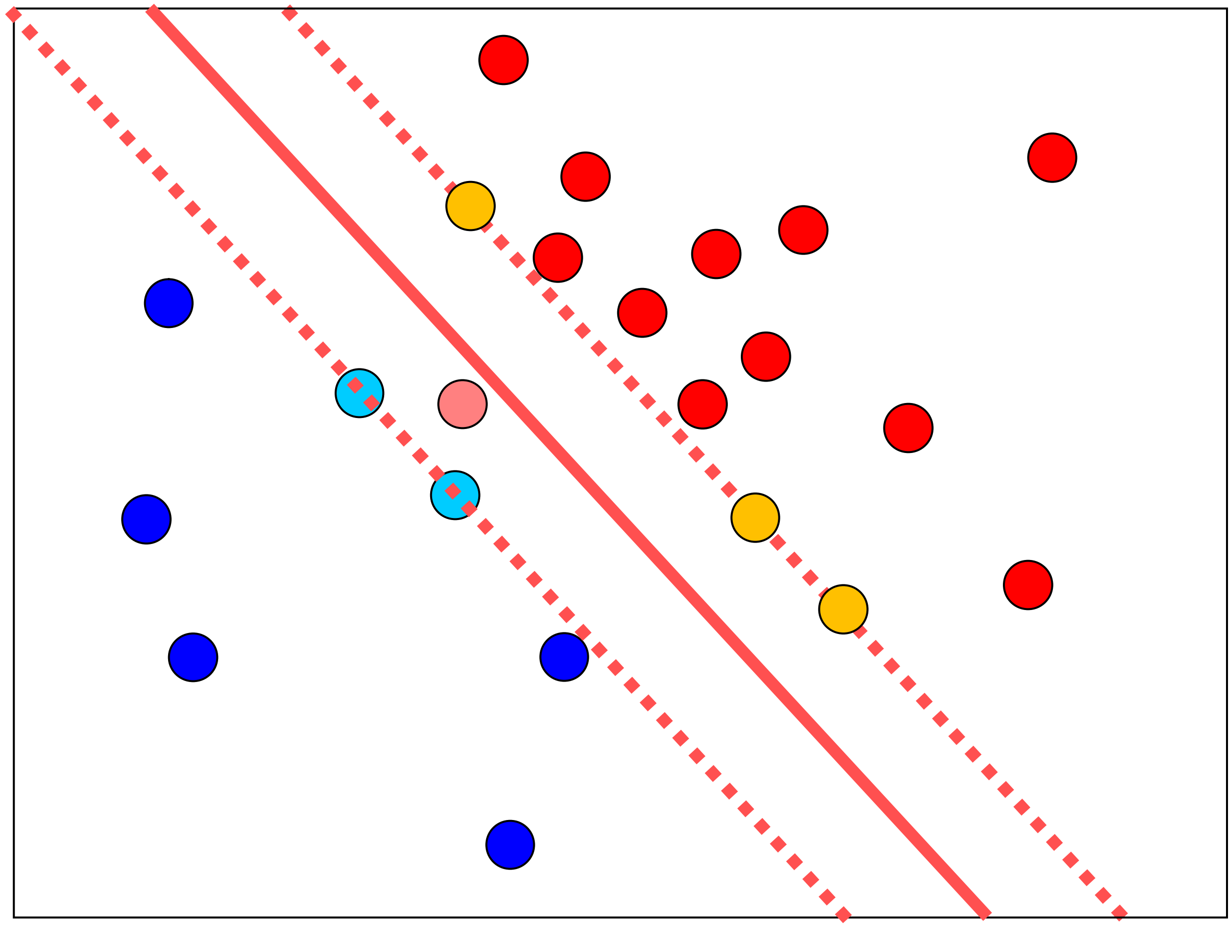
➔ Soft margin

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

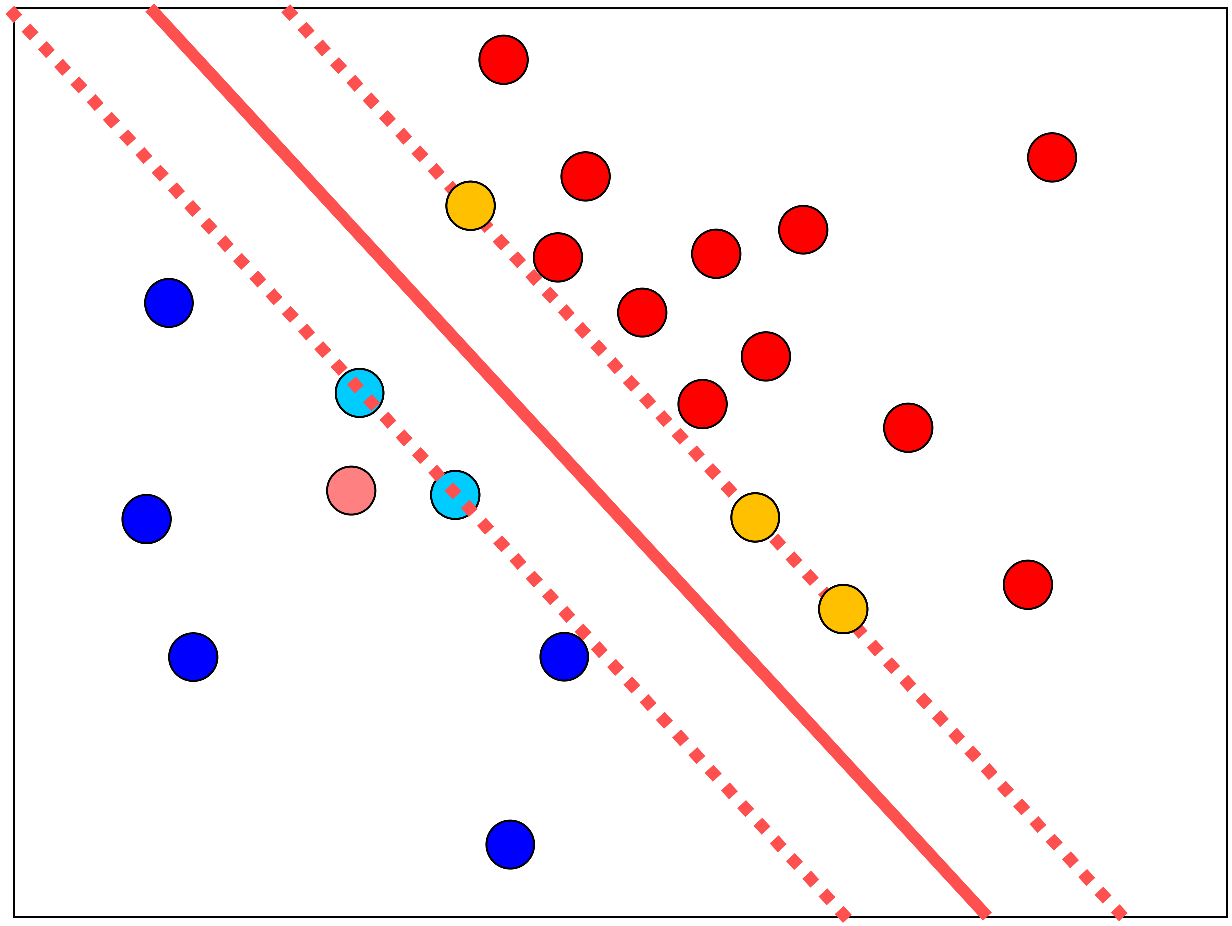
Hard Margin ...



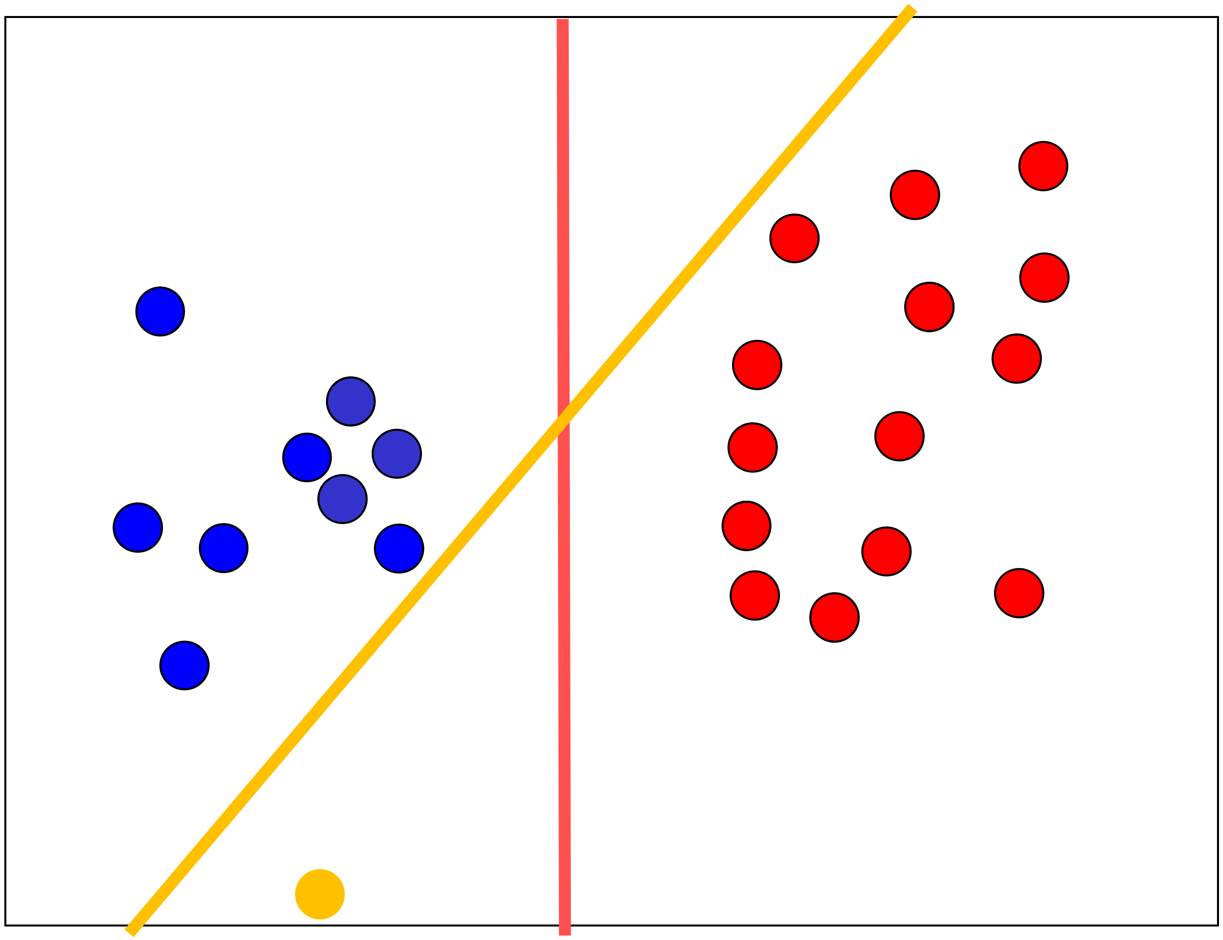
Soft Margin: for larger margin



Soft Margin: to find separation



Soft Margin: larger margin with outliers



Soft Margin

- Introduction of slack variables
  - penalises misclassification
  - adapt constraint

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

- Penalise non-zero ξi

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

  - How to solve it?
    - *Similar to "hard-margin" case*

- *Implications?*

Soft Margin

- 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

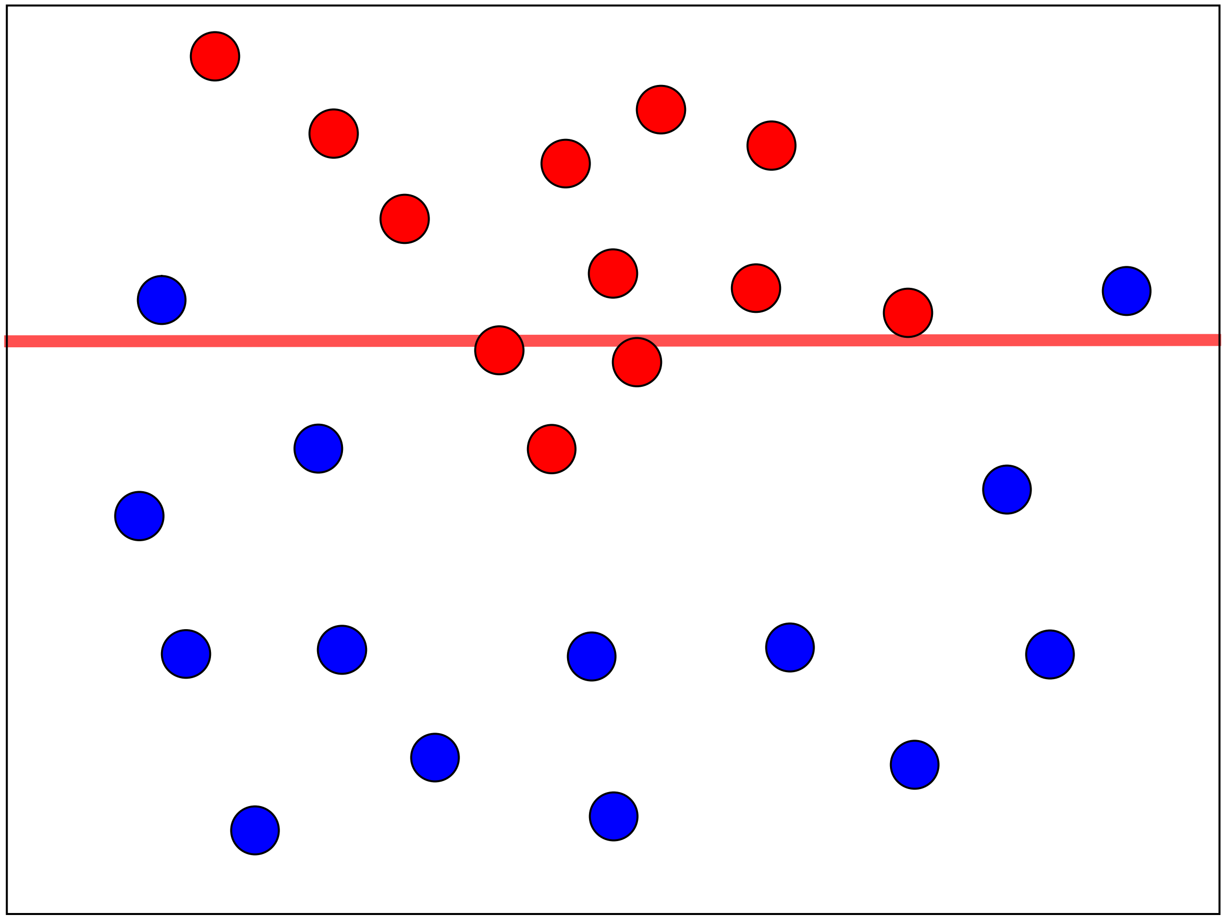
- C becomes part of optimisation problem
  - Sometimes called the “complexity parameter”
  - Large C: penalising errors more

Outline

- Short recap
- Random Forests
- Evaluation
- Support Vector Machines: kernels

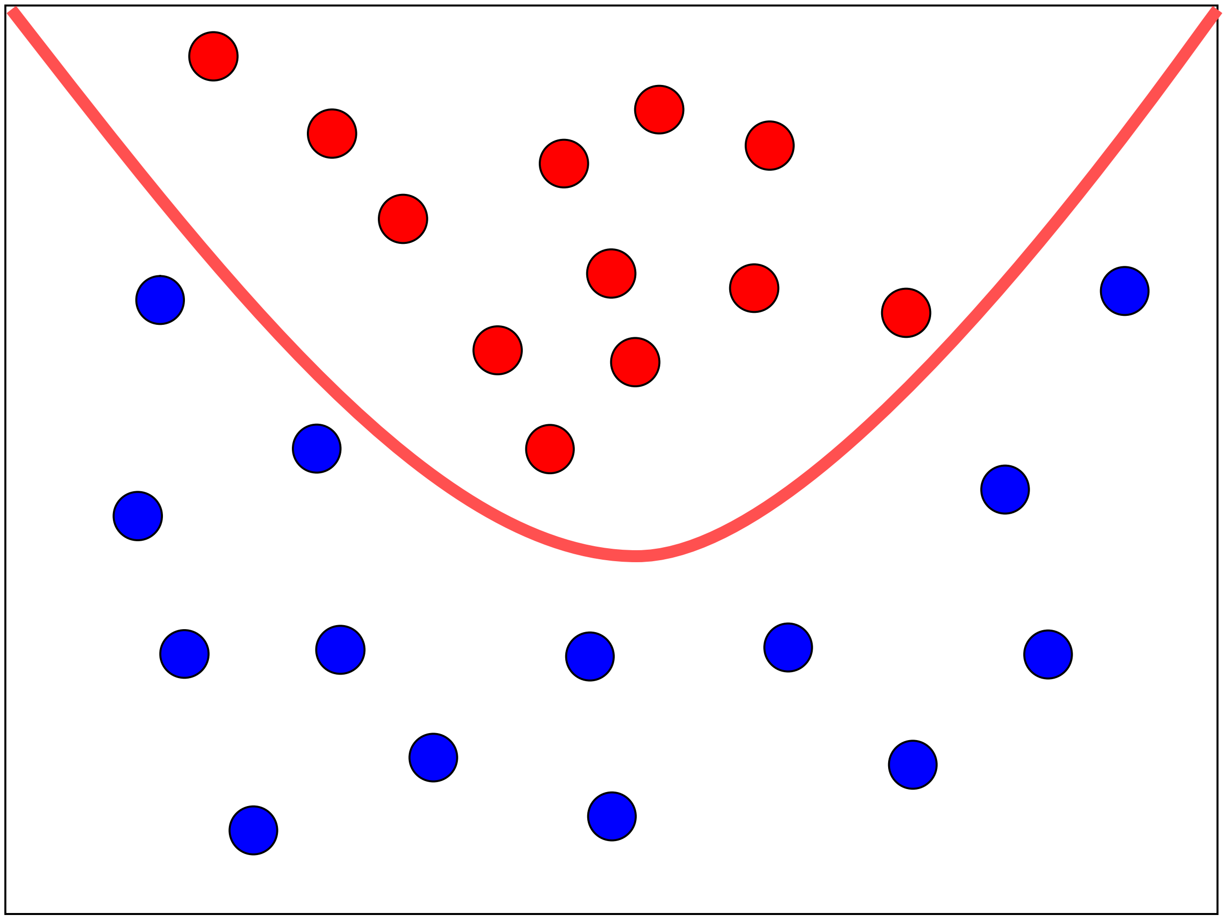
Non-linearly separable data

- "Acceptable" hyperplane can *not* be found

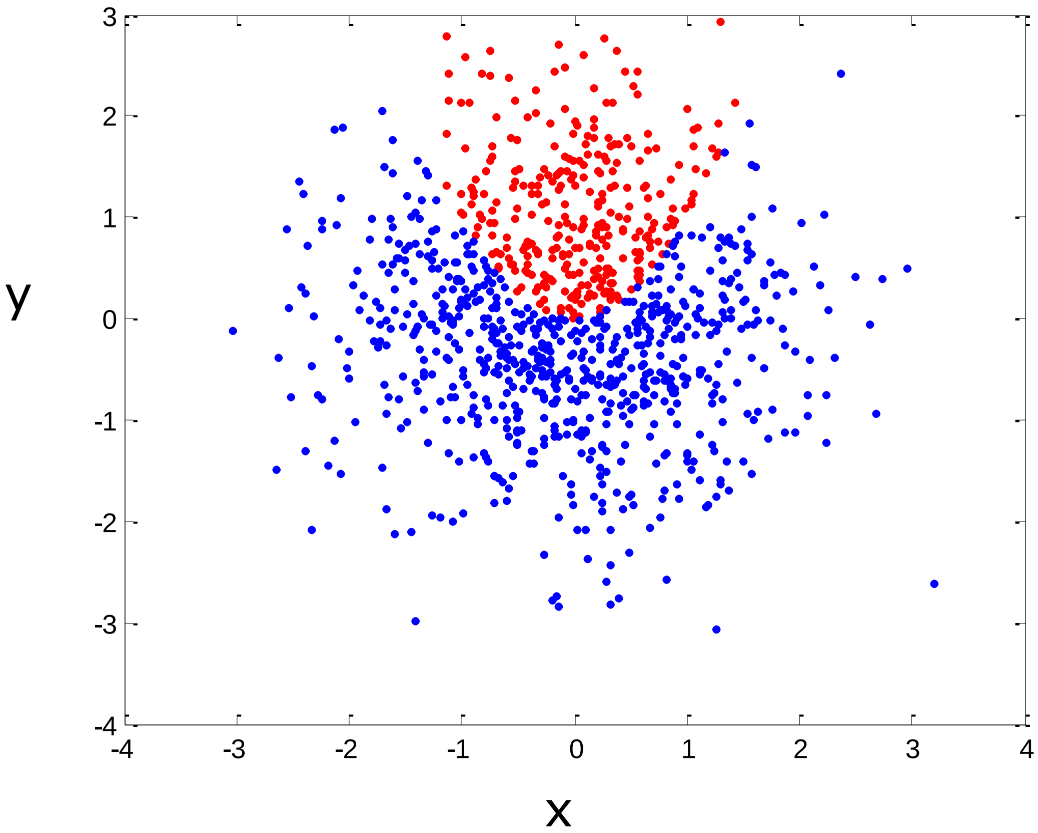


Non-linearly separable data

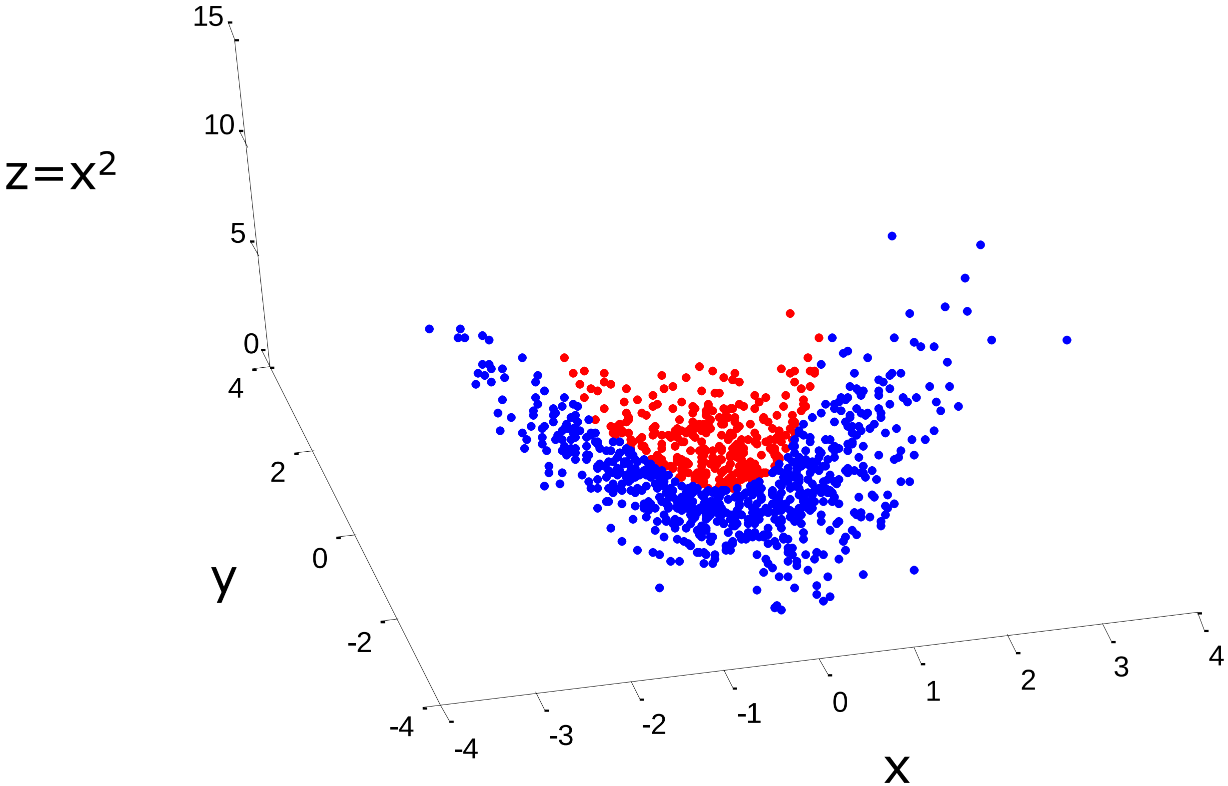
- Data would be easily separable by a polynom



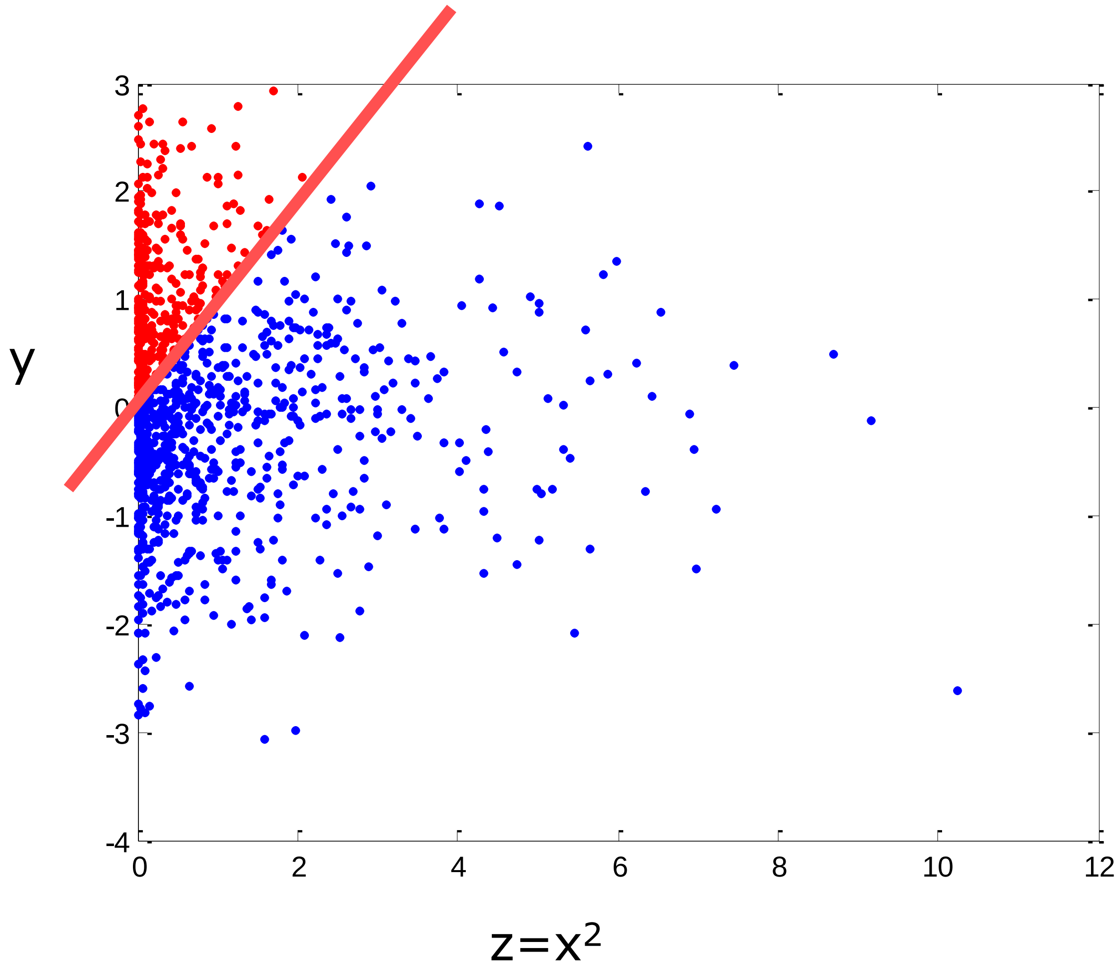
Non-linearly separable data



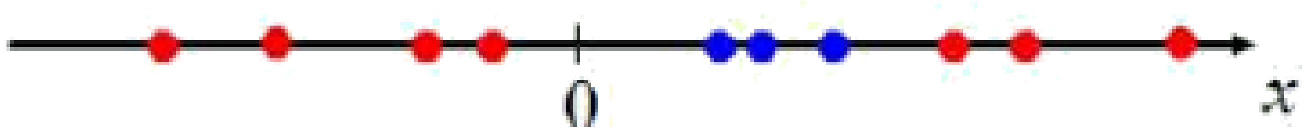
New coordinate $z=x^2$



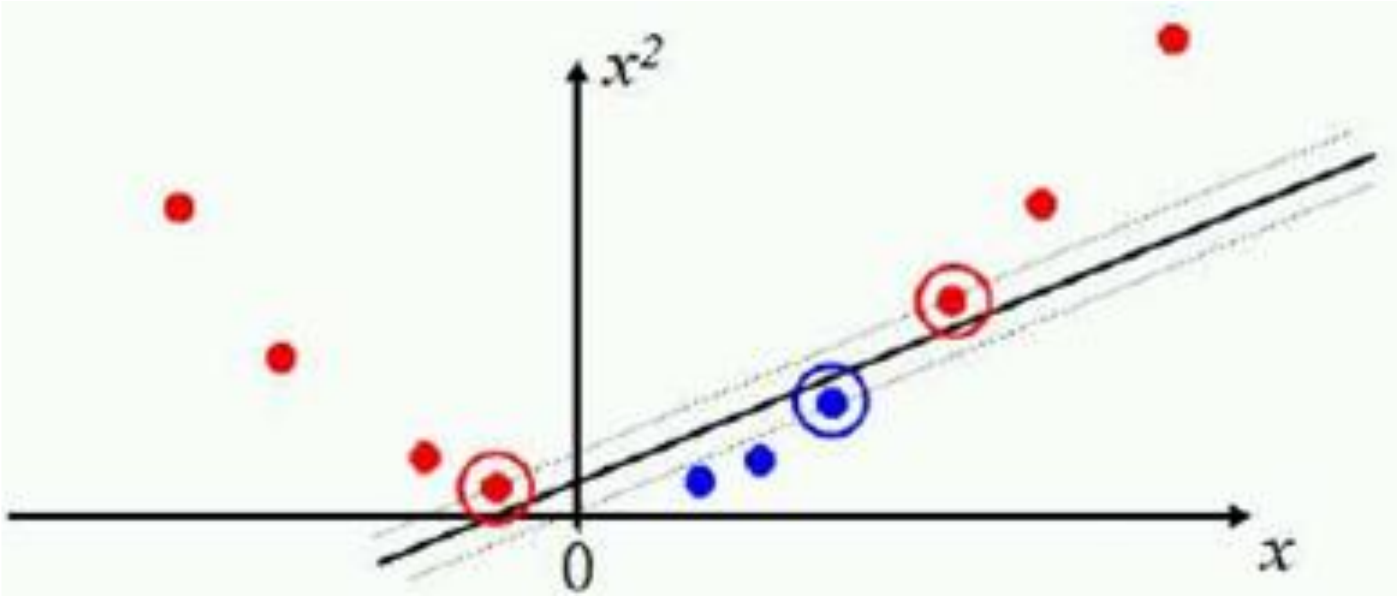
New coordinate $z=x^2$



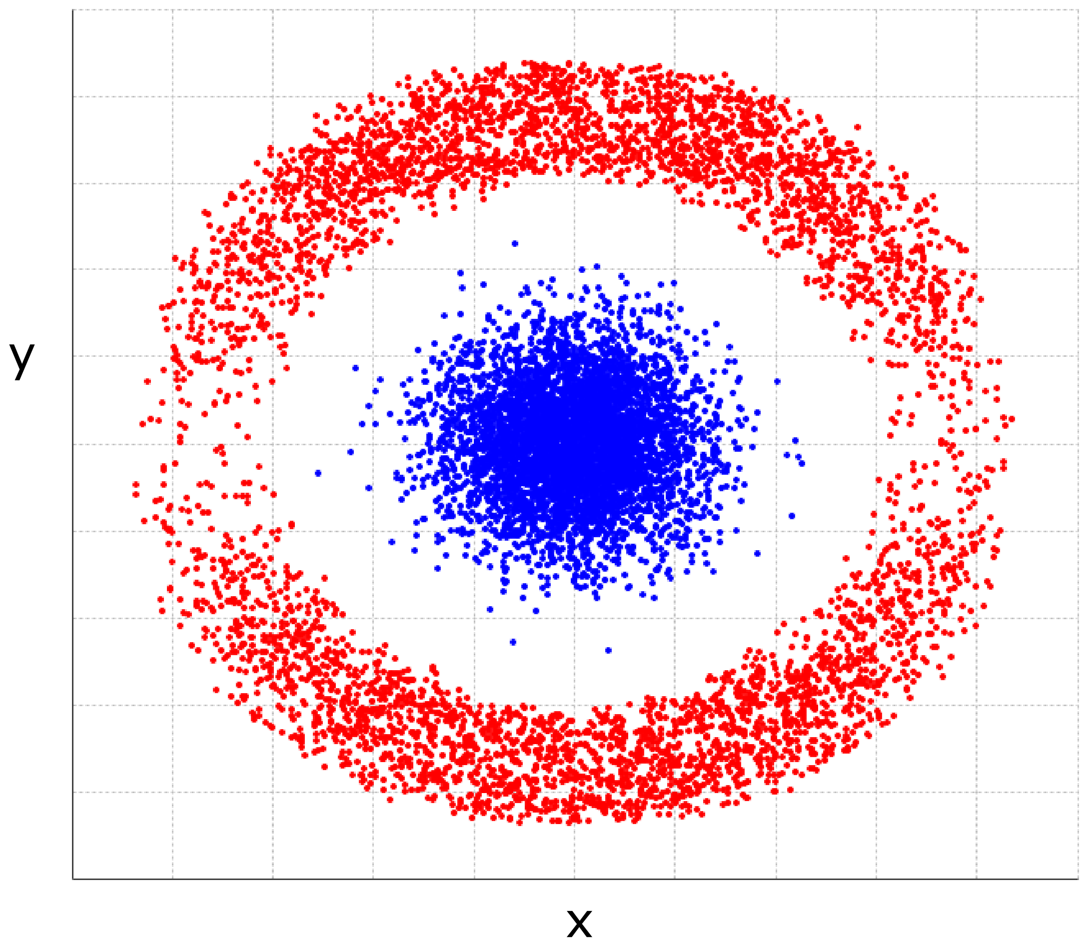
Non-linearly separable data



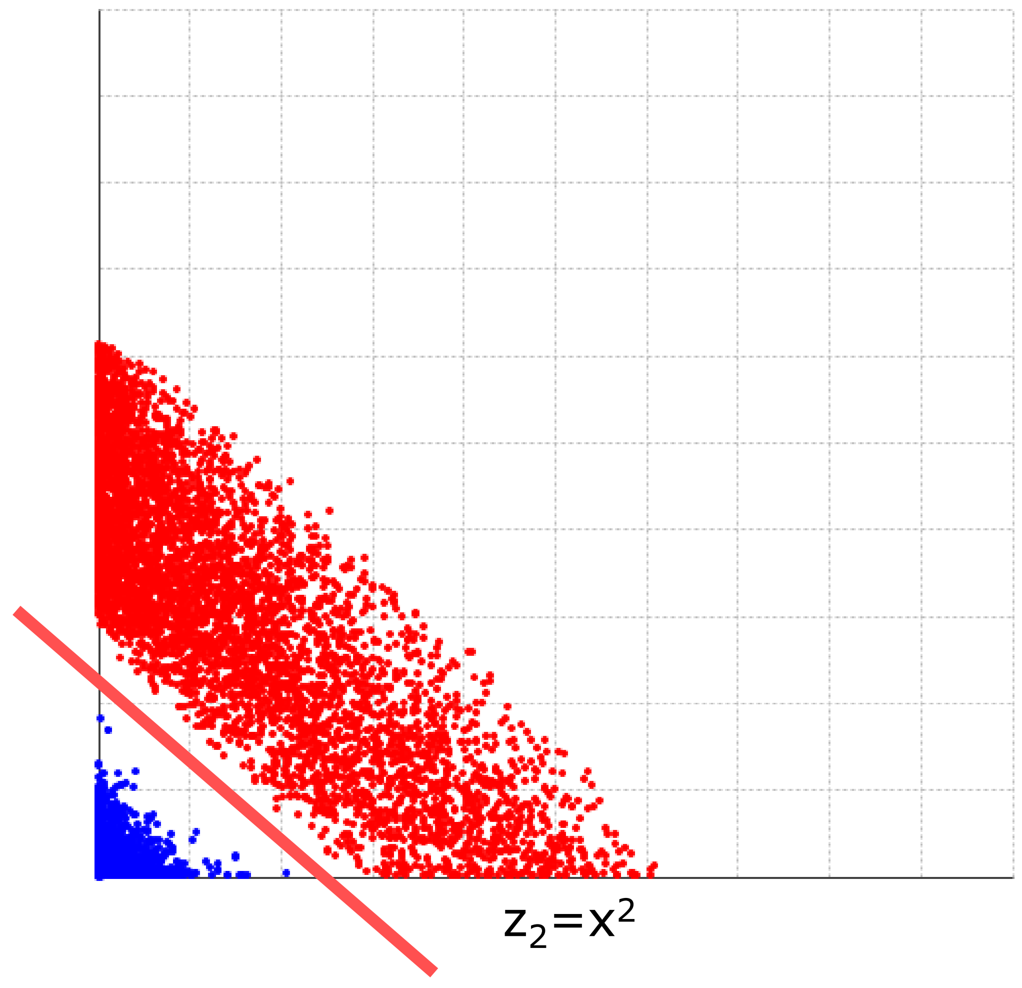
New coordinate $y=x^2$



Non-linearly separable data

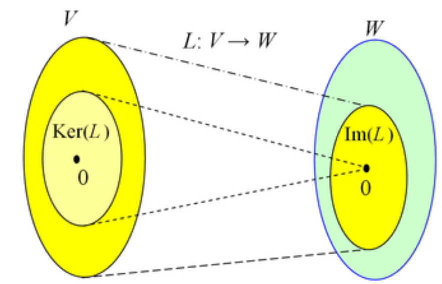


New coordinates $z_1=x^2$ & $z_2=y^2$



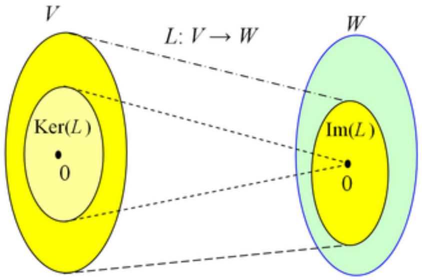
SVM: Kernels

- Projection of data into higher dimensional space
  - Data may be separable in this space
- Projection: *multiplication* of vectors with kernel matrix



- Often: projected space has dimensionality equal to number of training vectors
- Kernel matrix determines shape of possible separators
  - In previous example: polynomial (quadratic)

Common Kernels



- Quadratic
  - e.g. $k(x,y) = (x \cdot y)^2$
- General Polynomial (arbitrary degree)
  - $k(x,y) = (x \cdot y)^d$ (homogenous)
  - $k(x,y) = (x \cdot y + 1)^d$ (inhomogenous)
- Gaussian $k(x,y) = \exp\left(\frac{\|x - y\|^2}{2\sigma^2}\right)$
- Radial Basis Function $k(x,y) = \exp(-\gamma \|x - y\|^2), \gamma > 0$
- Sigmoid $k(x,y) = \tanh(\kappa x \cdot y + c)$

  $for\ some\ (not\ every)\ \kappa > 0, c < 0$

Best suited Kernel

- Best kernel depends on the data and its underlying distribution
- How to chose
  - Kernel family?
  - Exact parameters?
  - Model selection:
    - Train several models with different kernels
    - Chose best performing

- Linear kernels work well with sparse data (e.g. Text)

Kernel Trick

- Other ML algorithms could work with projected (high dimensional) data
  – So why bother with (rather) complicated SVM?

- Working with higher dimensional data is problematic: increased computational complexity

Kernel Trick

- Working with high dimensional data is problematic (computational complexity)

- *Outlook:* SVMs depend only on *dot-product* between vectors
  - Kernel Trick: replace dot-product with kernel function
  - This is computationally inexpensive (relatively)

$$Lp = \sum_{1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j x_i \cdot x_j$$

Outline

- Short recap
- Random Forests
- Evaluation
- Support Vector Machines: multi-class

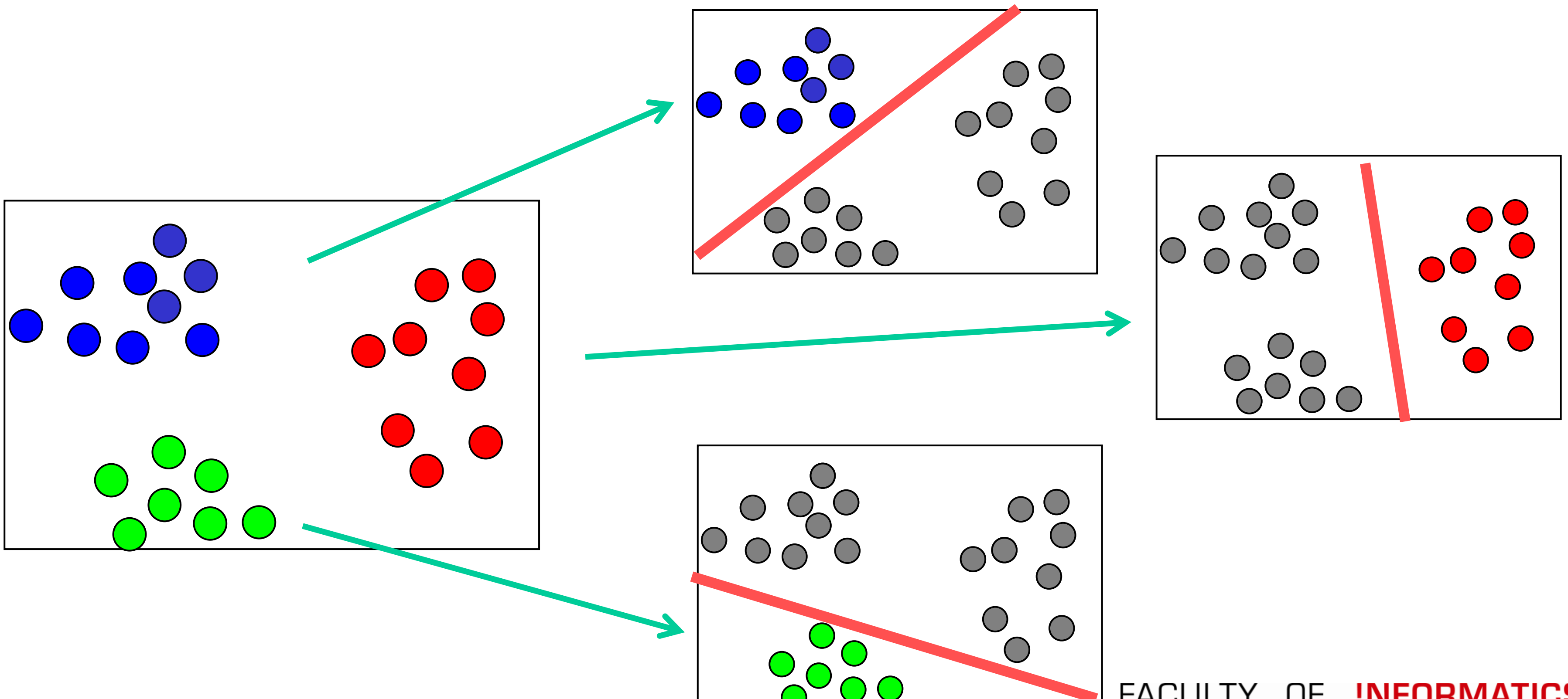
Multi-class SVM

- Standard SVM algorithm just solves binary problems (i.e. two classes)
- Multi-class problem (three or more classes)

- Reduce (single) multi-class problem to multi binary problems
  - One vs. all ➔ chose the classifier with the greatest margin
  - One vs. One ➔ chose class selected most often

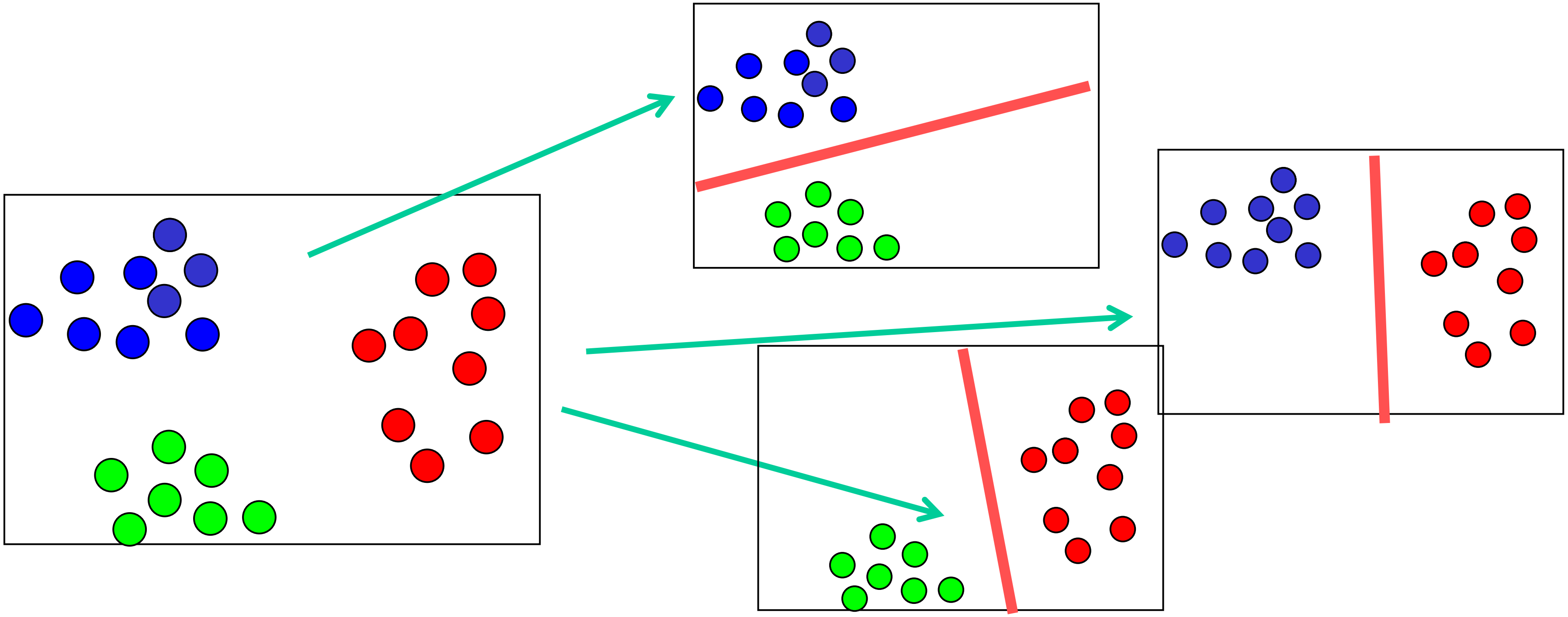
Multi-class SVM

- One vs. All
  - Build binary classifiers that distinguish between class *i* and the rest (i = 1, ..., #numClasses)
  - Classifier with highest output *f(x)* is the winner



Multi-class SVM

- One vs. One
  - Build binary classifier for each pair of classes
  - Class with highest number of votes wins

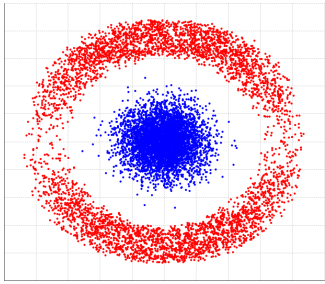


  - *How many classifiers?*
    - Needs |C|(|C|-1)/2 classifiers (but smaller)

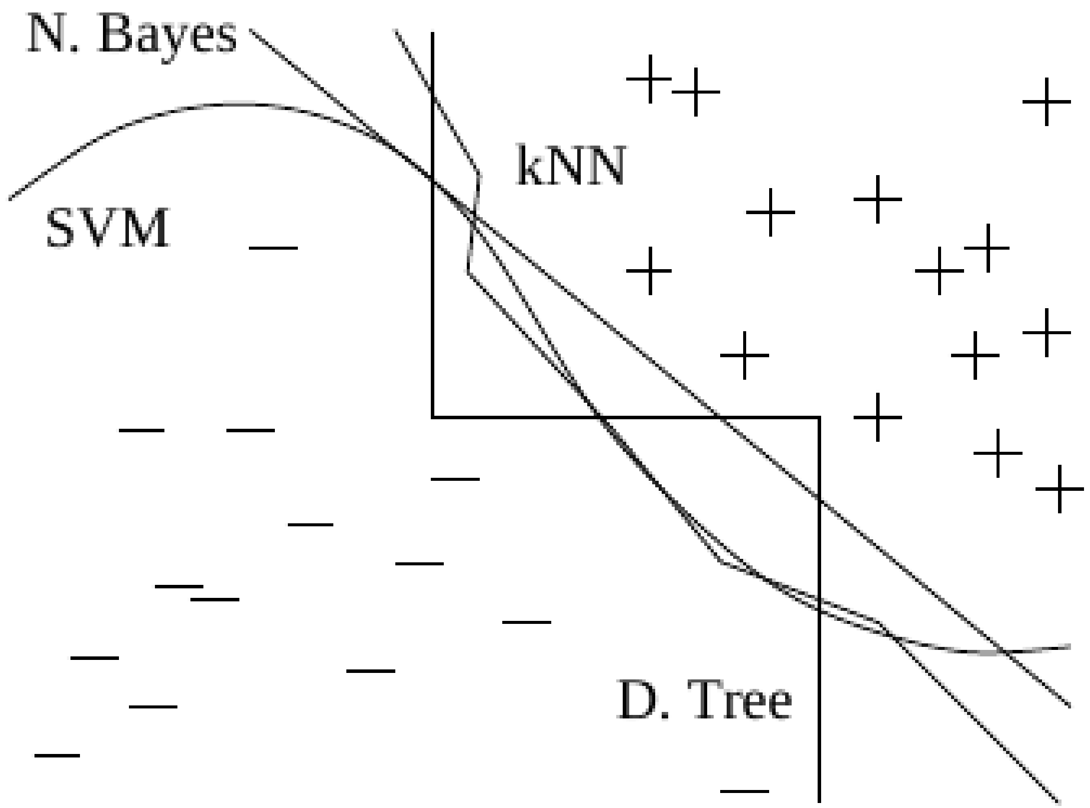
Properties of SVM

- High classification accuracy
  - Good generalisation, even though decision boundary in original space is complex
- Linear kernels: Good for sparse, high dimensional data, e.g. text mining

- Much research has been directed at SVM, and it’s foundations ➔ solid background
- Implementation available in open-source software (e.g. libSVM)



Comparison of decision boundaries



Questions ?