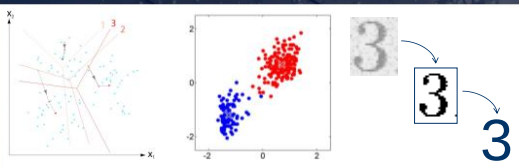
SINTEF



INF4300
Learning from data
Classification and clustering

Asbjørn Berge 20-10-2010

SINTEF

Plan for today

- What is clustering and classification?
- Clustering example: K-means algorithm
- Classification example: Support vector machines

SINTEF

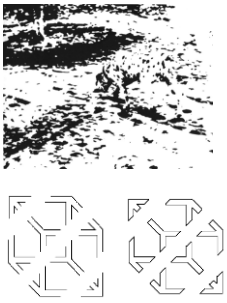
Recommended reading after this lecture

- Pattern recognition introduction
  - R.C. Gonzales and R.E. Woods: *Digital Image Processing, 3rd ed*, 2008. Prentice Hall. ISBN: 978-0-13-168728-8. Chapter 12, 12.2.3 very cursory
- Clustering
  - T. Hastie, R. Tibshirani and J. H. Friedman: *The Elements of Statistical Learning*, 2001. Springer Verlag. ISBN: 978-0-38-7952840. Ch. 14.3
- Support Vector Machines
  - C.J.C. Burges. *A tutorial on support vector machines for pattern recognition*. Data Mining and Knowledge Discovery, 2(2):955-974, 1998. http://citeseer.nj.nec.com/burges98tutorial.html

SINTEF

Copying human vision is hard

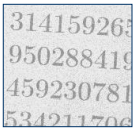
- Humans are incredible *pattern recognizers*
  - In fact the brain will usually suggest patterns even in noise
  - Many optical illusions can be explained by your brain choosing the "closest match" to something familiar
  - Can we copy this innate ability to *complete* patterns with computers?

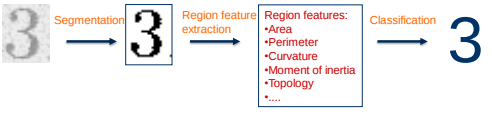


SINTEF

What is pattern recognition?

- Informal definition: Based on measurements from images or image regions, create a system that finds the closest match to something we have seen before. Usually referred to as classification.





SINTEF

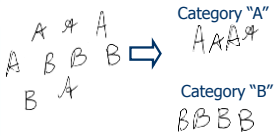
What you already know

- Features
  - Measurements of some property with the objects we study
  - You understand why and how to extract features from images
  - Several features measured for each object makes a vector
  - Feature vectors can be visualized with scatterplots
- Basic linear algebra
  - Some simple matrix manipulations should not scare you
  - You understand how to translate algebra into code

SINTEF

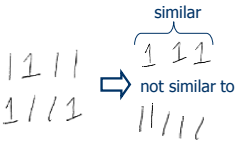
Classification and clustering

Supervised classification



- Predefined categories
  - Result is assignment to one
- Training on previously seen examples
  - Prior knowledge can be applied

Unsupervised classification

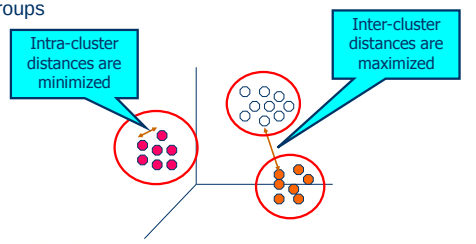


- Define quality measure
  - Similarity / dissimilarity
- No predefined categories
  - No knowledge of category

SINTEF

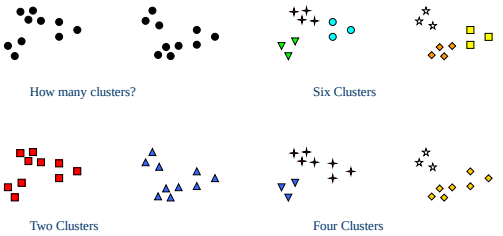
What is Cluster Analysis?

- Finding groups of objects such that the objects in a group will be similar (or related) to one another and different from (or unrelated to) the objects in other groups



SINTEF

Notion of a Cluster can be Ambiguous



SINTEF

Types of Clusterings

- A clustering is a set of clusters
- Important distinction between hierarchical and partitional sets of clusters
- Partitional Clustering
  - A division data objects into non-overlapping subsets (clusters) such that each data object is in exactly one subset
  - Soft partitioning allows objects to participate in several subsets (clusters)
- Hierarchical clustering
  - A set of nested clusters organized as a hierarchical tree

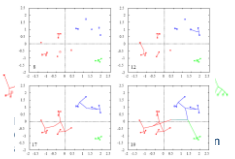
SINTEF

Hierarchical Clustering

hclust

Consider a sequence of partitions of the *n* samples into *c* clusters

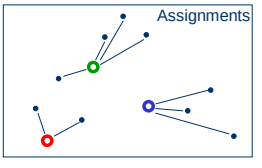
- The first is a partition into *n* cluster, each one containing exactly one sample
- The second is a partition into *n-1* clusters, the third into *n-2*, and so on, until the *n-th* in which there is only one cluster containing all of the samples
- At the level *k* in the sequence, *c = n-k+1*.



SINTEF

Partition clustering

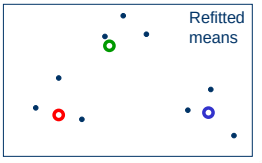
- Assume we want *k* classes.
- Assume we start with randomly located cluster centers
- *n* datapoints into *k* classes means ~n^k allocations to test⇨iterative algoritm



General algorithm alternates:

Assignment step: Assign each datapoint to the closest cluster.

Refitting step: Move each cluster center to the center of gravity of the data assigned to it.al



SINTEF

k-means Clustering

kmeans

- Each cluster is associated with a centroid (center point)
- Each point is assigned to the cluster with the closest centroid
- Number of clusters, *k*, must be specified
- The basic algorithm is very simple

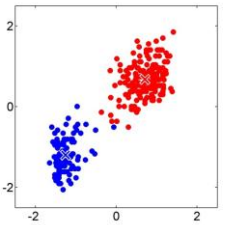
1: Select K points as the initial centroids.
2: **repeat**
3: Form K clusters by assigning all points to the closest centroid.
4: Recompute the centroid of each cluster.
5: **until** The centroids don't change

SINTEF

K-means Algorithm

Step 4:

If the clusters don't change; $\mu_k^{(i+1)} \approx \mu_k^{(i)}$ (or prespecified number of iterations *i* reached), *terminate*, else reassign - increase iteration *i* and goto step 2.



SINTEF

Responsibilities

- *Responsibilities* assign data points to clusters

$r_{nk} \in \{0, 1\}$

such that

$\sum_k r_{nk} = 1$

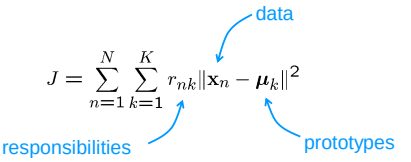
- Example: 5 data points and 3 clusters

$$(r_{nk}) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix}$$

Christopher M. Bishop

SINTEF

K-means Cost Function



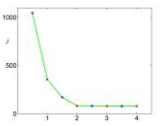
SINTEF

Minimizing the Cost Function

- E-step: minimize J w.r.t. r_{nk}
  - assigns each data point to nearest prototype
- M-step: minimize J w.r.t $\boldsymbol{\mu}_k$
  - gives

$$\boldsymbol{\mu}_k = \frac{\sum_n r_{kn} \mathbf{x}_n}{\sum_n r_{kn}}$$

  - each prototype set to the mean of points in that cluster
- Convergence guaranteed since there is a finite number of possible settings for the responsibilities



SINTEF

Quality of K-means clustering

- Most common measure is Sum of Squared Error (*SSE*)
  - For each point, the error is the distance to the nearest cluster
  - To get *SSE*, we square these errors and sum them.

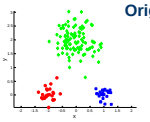
$$SSE = \sum_{i=1}^{K} \sum_{x \in C_i} dist^2(m_i, x)$$

  - *x* is a data point in cluster C_i and m_i is the representative point for cluster C_i
    - can show that m_i corresponds to the center (mean) of the cluster
  - Given two clusterings, we can choose the one with the smallest error
  - One easy way to reduce SSE is to increase *K*, the number of clusters
    - A good clustering with smaller *K* can have a lower SSE than a poor clustering with higher *K*

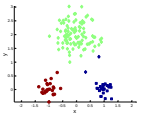
SINTEF

Two different K-means Clusterings

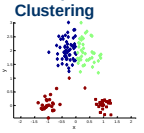
Original Points



Optimal Clustering

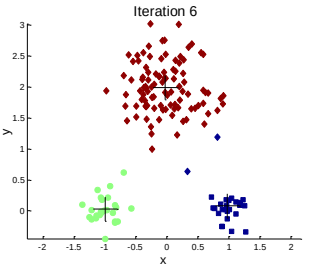


Sub-optimal Clustering



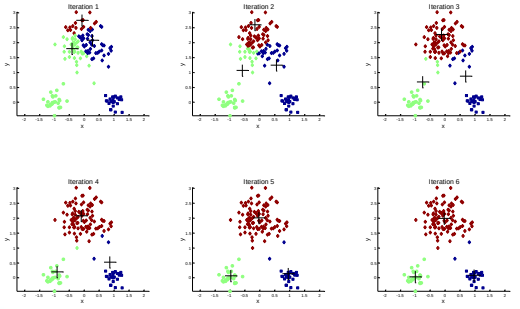
SINTEF

Importance of Choosing Initial Centroids



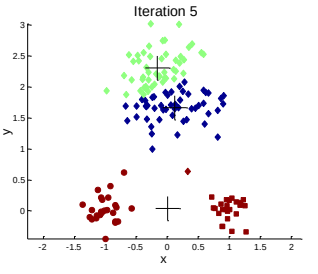
SINTEF

Importance of Choosing Initial Centroids



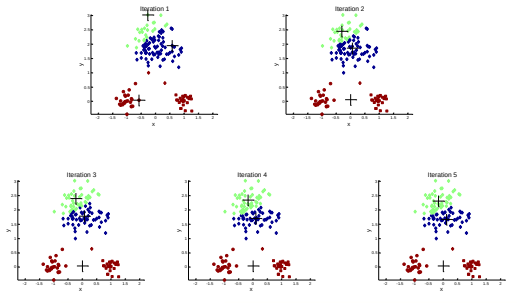
SINTEF

Importance of Choosing Initial Centroids …



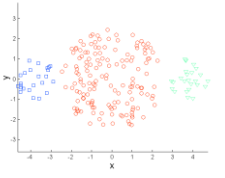
SINTEF

Importance of Choosing Initial Centroids …

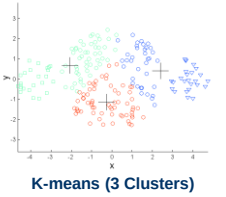


SINTEF

Limitations of K-means: Differing Sizes



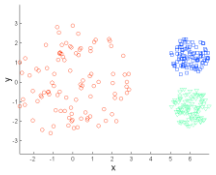
Original Points



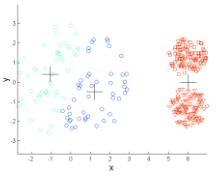
K-means (3 Clusters)

SINTEF

Limitations of K-means: Differing Density



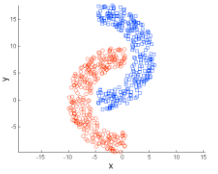
Original Points



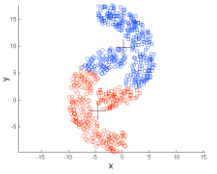
K-means (3 Clusters)

SINTEF

Limitations of K-means: Non-globular Shapes



Original Points



K-means (2 Clusters)

SINTEF

Problems with Selecting Initial Points

- If there are *K* 'real' clusters then the chance of selecting one centroid from each cluster is small.
  - Chance is relatively small when *K* is large
  - If clusters are the same size, *n*, then

$$P = \frac{\text{number of ways to select one centroid from each cluster}}{\text{number of ways to select } K \text{ centroids}} = \frac{K!n^K}{(Kn)^K} = \frac{K!}{K^K}$$

  - For example, if $K = 10$, then probability = $10!/10^{10} = 0.00036$
  - Sometimes the initial centroids will readjust themselves in 'right' way, and sometimes they don't

SINTEF

Solutions to Initial Centroids Problem

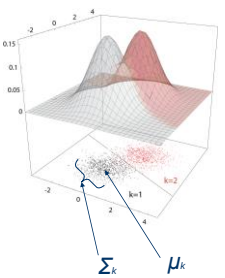
- Multiple runs
  - Helps, but probability is not on your side
- Sample and use other clustering methods to determine initial centroids
- Select more than k initial centroids and then select among these initial centroids
  - Select most widely separated
- Improved k-means with cluster merge and split (ISODATA)

SINTEF

Classification

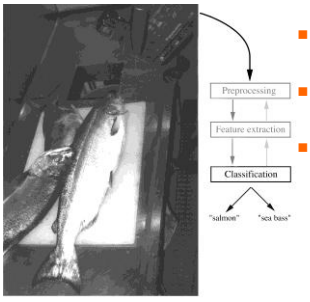
We have *k* classes, and some initial *training* data

Task is now not to assign data into the class which it is most similar, but learning what is *typical* for each of the classes, and base a rule on that



SINTEF

Fish sorting example

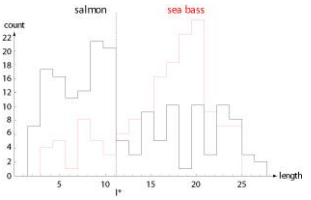


- Task: create a computer system that sorts fish on conveyor belt according to species
- Create a rule that enables us to decide "salmon " or "sea bass" with *minimum error*
- Steps in the process
  - Capture image
  - Isolate fish
  - Take measurements
  - Make Decision

SINTEF

Fish sorting – fish length

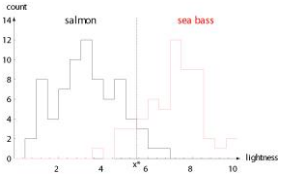
- General knowledge : sea bass is usually longer than salmon
- Define length of fish as a *feature,* and decide according to a *treshold* on length
- Rule : fish = ”salmon” if *length* < *l**
- Set threshold *l** such that it minimizes error



SINTEF

Fish sorting – scale lightness

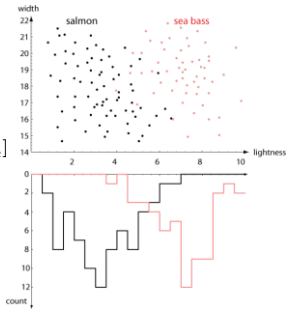
- Average lightness of scales is also different between species
- Does a treshold based on this lightness classify better?
- Rule : fish = ”sea bass” if *lightness* > *x**



SINTEF

Fish sorting

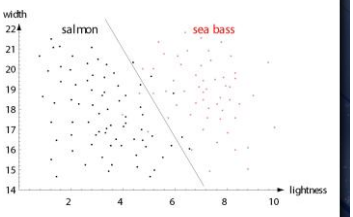
- Both features are inadequate
- Combine the features, creating a feature vector (n-tuple) $\mathbf{x} = [x_1 \quad x_2]$
- Scatterplot replaces histograms
- Features have complementary information



SINTEF

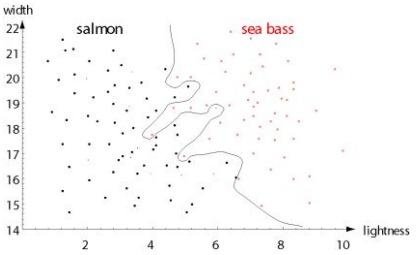
Fish sorting – linear decision boundary

- Create a rule that uses the feature vector to decide
- For example a line, a *decision boundary*, and decide ”salmon” if a point is to the left of this boundary



SINTEF

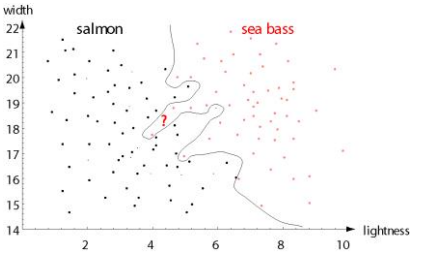
The goal was to minimize error?



A complex rule that classifies all *known* points correctly can *always* be found

SINTEF

Classifying new data with the perfect rule



The new data is most likely ”salmon”, but classifies as ”sea bass”

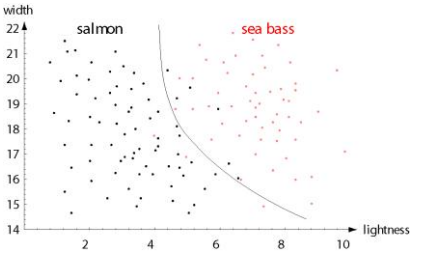
SINTEF

The goal of classification

- Classify *new* observations with *minimum error*
  - We *train* on available data, with the goal of minimizing error
  - Evaluate (*test*) classification on an "*unknown*" dataset, measure *generalization*
  - We may face a tradeoff *generalization* ↔ *error*, when reducing the complexity of a classifier to generalize better

SINTEF

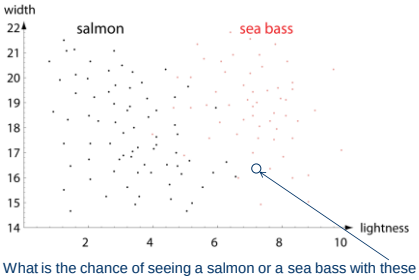
A reasonable tradeoff



SINTEF

How to design this rule?

Find probability estimates in data space

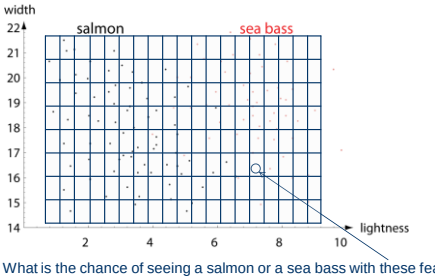


What is the chance of seeing a salmon or a sea bass with these features?

SINTEF

How to design this rule?

Bin and count, and divide by total

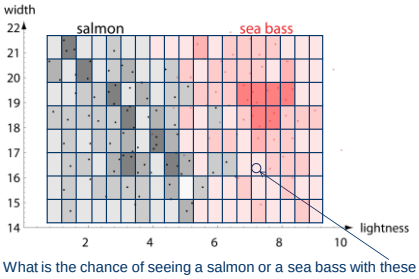


What is the chance of seeing a salmon or a sea bass with these features?

SINTEF

How to design this rule?

A histogram, chance is equal to bin value

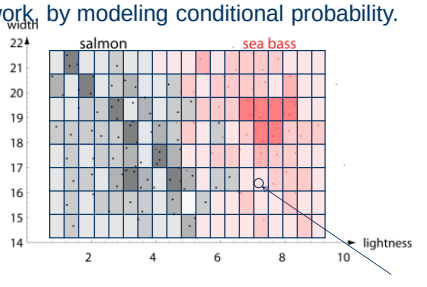


What is the chance of seeing a salmon or a sea bass with these features?

SINTEF

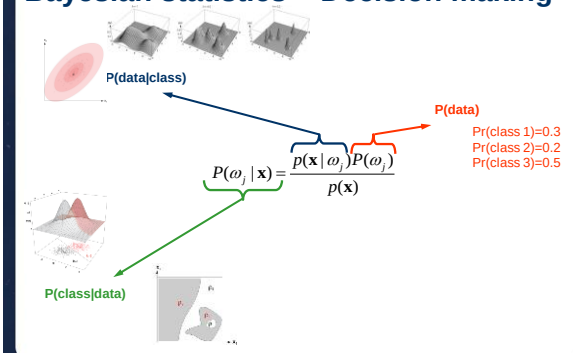
How to design this rule?

This is in essence how statistical classifiers work, by modeling conditional probability.



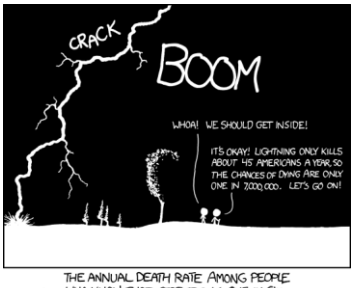
SINTEF

Bayesian statistics – Decision making



SINTEF

xkcd explains conditional probability



THE ANNUAL DEATH RATE AMONG PEOPLE WHO KNOW THAT STATISTIC IS ONE IN SIX.

SINTEF

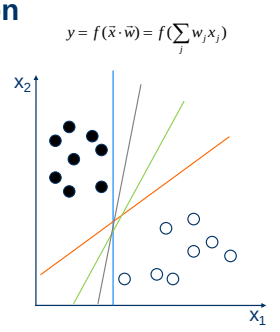
Another approach to classification

- The first approach illustrated in this lecture is based on modeling the data in each class and using Bayes rule
  - Examples of this classification approach is
    - K-nearest neighbor
    - Statistical classifiers (usually called Gaussian)
- Decision is made by comparing probabilities of the models and defining a rule as a «boundary» where the probabilities are equal.
- What if we instead model the boundary directly?

SINTEF

Linear classification

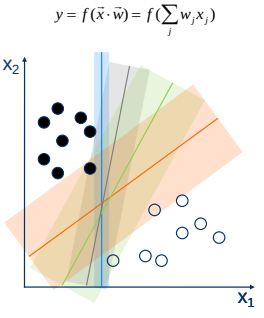
What is the best classification rule in form of a straight line?



SINTEF

Classifier margin

The margin of a classifier is the width you can increase the boundary before you hit a data point.



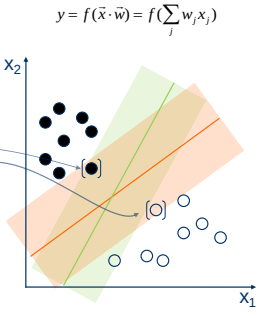
SINTEF

Support vectors

Large margin = good
Maximum margin = best

Slightly complex maths to prove, but feels intuitively right*

Support vectors are those datapoints that the margin pushes up against.



*Look up"Vapnik-Chernovenkis Dimension", and you will find a proof that a related proposition is a good idea.

SINTEF

Specifying a line and a margin

How to represent this in mathematical terms?

$y = f(\vec{x}\cdot\vec{w}) = f(\sum_j w_j x_j)$

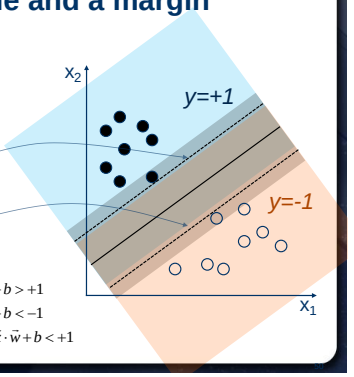
And in *m* input dimensions?

•Positive plane
$\{\vec{x} : \vec{x}\cdot\vec{w}+b=+1\}$

•Negative plane
$\{\vec{x} : \vec{x}\cdot\vec{w}+b=-1\}$

•Classifier from this

$$y = f(\vec{x}\cdot\vec{w}) = \begin{cases} +1, \text{if } \vec{x}\cdot\vec{w}+b > +1 \\ -1, \text{if } \vec{x}\cdot\vec{w}+b < -1 \\ \text{trouble, if } -1 < \vec{x}\cdot\vec{w}+b < +1 \end{cases}$$



SINTEF

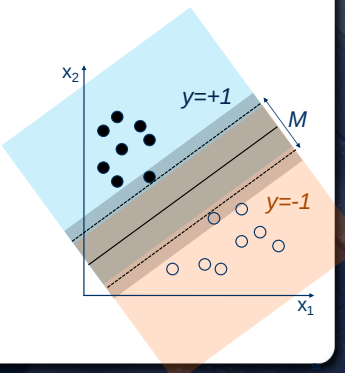
Margin width

$\{\vec{x} : \vec{x}\cdot\vec{w}+b=+1\}$

Remember, we're looking for the maximum margin.

Lets call the margin width *M*, and calculate it from our positive and negative planes. Remember that the normal vector of a line is the gradient. I.e., in this case *w*

We want to describe M in terms of w (and b).



SINTEF

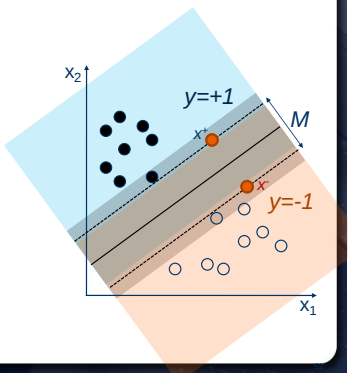
Margin width

•Positive plane
$\{\vec{x} : \vec{x}\cdot\vec{w}+w_0=+1\}$

•Negative plane
$\{\vec{x} : \vec{x}\cdot\vec{w}+w_0=-1\}$

•*w* is orthogonal to the planes
• Let x^- be *any point* on the negative plane.
• Let x^+ be the closest point to x^- on the positive plane.
•Claim

$\vec{x}^+ = \vec{x}^- + \lambda\vec{w}$



SINTEF

Margin width

•We have

$\{\vec{x} : \vec{x}\cdot\vec{w}+b=+1\}$

$\{\vec{x} : \vec{x}\cdot\vec{w}+b=-1\}$

$\vec{x}^+ = \vec{x}^- + \lambda\vec{w}$

$\Rightarrow$

$|\vec{x}^+ - \vec{x}^-| = M$

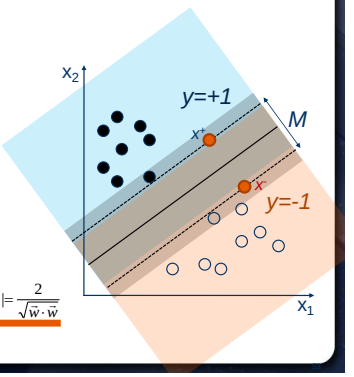
•Substitution

$(\vec{x}^- + \lambda\vec{w})\cdot\vec{w}+b=1$

$\vec{x}^-\cdot\vec{w} + \lambda\vec{w}\cdot\vec{w}+b=1$

$-1+\lambda\vec{w}\cdot\vec{w}=1$

$\lambda = \frac{2}{\vec{w}\cdot\vec{w}} \Rightarrow M = |\vec{x}^+ - \vec{x}^-| = |\lambda\vec{w}| = \frac{2}{\sqrt{\vec{w}\cdot\vec{w}}}$



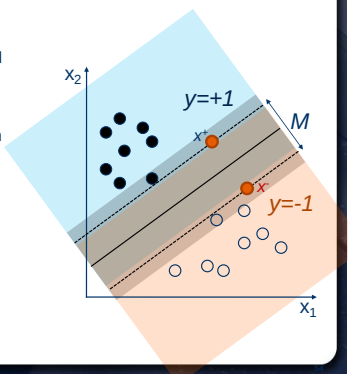
SINTEF

Margin width

So, given a guess of ***w*** and ***b*** we can
•Check if all datapoints are in the correct half-planes
•Compute the margin width

So now we just need to write a program that chooses the best w and b from all possible!

How?



SINTEF

Margin width

Assume ***R*** datapoints, each ***(x_k,y_k)***, where ***y_k=+/- 1***

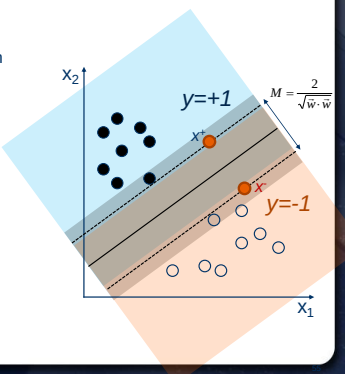
We will have ***R*** constraints

$w\cdot x_k + b \geq 1 \text{ if } y_k = 1$

$w\cdot x_k + b \leq -1 \text{ if } y_k = -1$

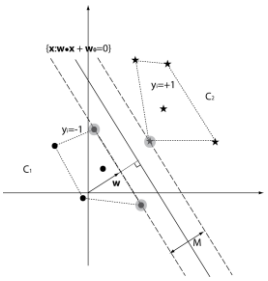
and quadratic criterion for optimization

$\text{Minimize } w\cdot w$



SINTEF

Find the ”optimal” linear classifier



- Hyperplane

$$H(w, w_0) = \{x : w \cdot x + w_0 = 0\}$$

- Distance from data point to plane

$$\delta(x, H) = \frac{1}{\|w\|}(w \cdot x + w_0 = 0)$$

- Maximize the distance to the closest datapoint

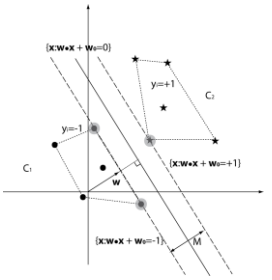
$$\max_{w, w_0} \min_{\forall i} \delta(x, H) = \frac{1}{\|w\|}(w \cdot x + w_0 = 0)$$

Precisely what we are looking for – but not very practical!

56

SINTEF

Clever tricks for solving the problem



- Any positive multiple of *(w,w0)* defines precisely the same hyperplane H
- Thus we can arbitrarily redefine the least distance to the closest data point

$$\min_{\forall i} |w \cdot x + w_0| = 1$$

- Implicitly, this means that the distances from the hyperplane to the closest points are

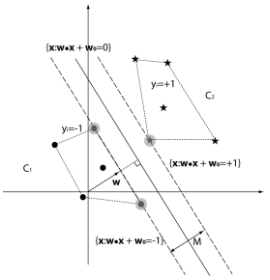
$$\pm 1/\|w\|$$

- i.e. ***M=2/||w||***
- Furthermore, we are looking for a classifier – so a reasonable request for our optimization problem is to demand that all points be on the correct side of the hyperplane, i.e.

$$y_i(w \cdot x + w_0) \geq 1 \forall i$$

SINTEF

....and it reduces to a simple(?) quadratic optimization problem



- The optimization problem can be written

$$\min_{w, w_0} \frac{1}{2} w^T w$$

$$\text{s.t. } y_i(w \cdot x_i + w_0) \geq 1 \ \forall i$$

- However, this problem has the same number of constraints as data points!
- # parameters to optimize = feature space dimension
- Solution: Rewrite to the dual which is slightly simpler to solve

58

SINTEF

Constrained optimization in 20 seconds

If x_0 is a solution to

$$\min_x f(x) \text{ subject to } g_i(x) \leq 0 \text{ for } i = 1, \ldots, n$$

there must exist a set of positive α_i [1] such that the following is satisfied

$$\begin{cases} \frac{\partial}{\partial x}(f(x) + \sum_i \alpha_i g_i(x)) \Big|_{x = x_0} \\ g_i(x) \leq 0 \text{ for } i = 1, \ldots, n \end{cases}$$

[1] or else we would be allowed to violate the constraint we imply by $g(x)<=0$

59

SINTEF

Solving technical details

- Primal problem on Lagrangian form

$$L_p = \frac{1}{2} w^T w - \sum_{i=1}^{n} \alpha_i (y_i (w \cdot x_i + w_0) - 1)$$

- *Q & D explanation of the dual*: We want to minimize L_p w.r.t. **w** and w_0 and maximize w.r.t. α_i
- We are looking for [illegible] i.e. derivatives of **w** and w_0 are 0, so a[illegible]g must hold at the solution:

The data only occurs as an inner product in this optimization! More on this later.

- Substitute **w** into primal proble[illegible]d optimization problem is changed to maximizing

$$L_d = -\frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j x_i \cdot x_j + \sum_{i=1}^{n} \alpha_i ; \sum_{i=1}^{n} \alpha_i y_i = 0, \alpha_i \geq 0$$

60

SINTEF

Dual optimization problem

The following quadratic program ready to be thrown to your favourite QP-solver

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

s.t.

$$\alpha_i \geq 0 \ \forall i, \sum_{i=1}^{n} \alpha_i y_i = 0$$

SINTEF

Quadratic Programming

Find ar… $\mathbf{u}^T R\mathbf{u}$ Quadratic criterion

Subjec… linear …y …ts

And su… e additional linear equality …ints

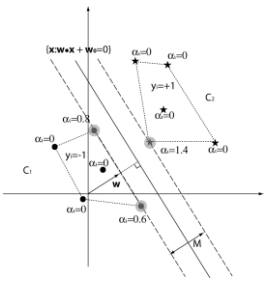
$$a_{(n+e)1}u_1 + a_{(n+e)2}u_2 + \ldots + a_{(n+e)m}u_m = b_{(n+e)}$$

Ignore all this, the numerical computing nerds have it under control!

(And apparently, they are having loads of fun constantly optimizing their solution!)

SINTEF

Interpretation of optimal α



- We defined the hyperplane when deriving the dual

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

- Many α are zero because:

- $\alpha_i (y_i (w \cdot x_i + w_0) > 1 \Rightarrow \downarrow \alpha_i \Rightarrow \uparrow L_p$

- Only α on the margin contribute to finding the hyperplane $\boldsymbol{w}$ (these are called *support vectors*)
- Classification of a new data point $\boldsymbol{x^*}$ can be found from the support vectors only without explicitly forming $\boldsymbol{w}$

- $w \cdot x^* + w_0 = \sum_{i:\alpha_i>0} \alpha_i y_i x_i \cdot x^* + w_0$

$$f(x^*) = \text{sgn}\left(\sum_{i:\alpha_i>0} \alpha_i y_i x_i \cdot x^* + w_0\right)$$

- Fun(?) mnemonic : physical interpretation of α as forces with sign y on a plane sheet along the normal vector. Optimal solution for α satisfies force and torque equilibrium

63

SINTEF

But, this classifier is not very useful?!

- Correct! In real world problems classes are usually overlapping, and the best boundary might not even be linear
- Two hacks allow the SVM ideas to be useful anyway
- Hack #1, *soft margin:*
  - Allow a certain amount of training points to violate constraint (slack variables in the QP)
  - Thus nonseparable classes can be used in the training data
- Hack #2, *kernels:*
  - Since the data occurs in the QP in form of an inner product, we may measure the distance in some other space of our choice, resulting in the fitting of a nonlinear boundary

64

SINTEF

Problem: data is never perfect

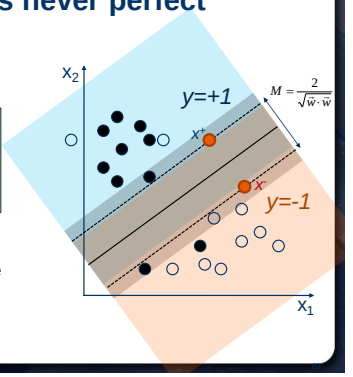
Different clusters usually overlap, or the problem would be *super simple.*

Cannot be expressed as a quadratic program! (Solving will be *slow*)

Does not differentiate errors between near misses and far off

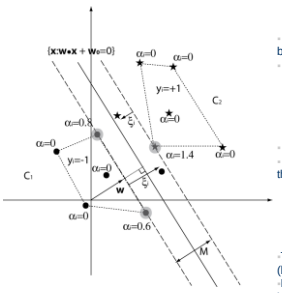
min w•w + C(#train errors)

NO! Two things to minimize at the same time usually creates a mess.



SINTEF

Hack #1: The soft margin



- Allow some datapoints to violate the constraint by an amount ξ_i
- Primal problem becomes

$$\min_{w,w_0} \frac{1}{2} w^T w + C\sum_{i=1}^{n} \xi_i$$

$$s.t.$$

$$y_i(w \cdot x_i + w_0) \geq 1 - \xi_i, \xi_i \geq 0 \forall i$$

- C is a *cost of misclassification* when training
- The dual is just adjusted for this slack giving the QP an upper bound on each α

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

$$s.t.$$

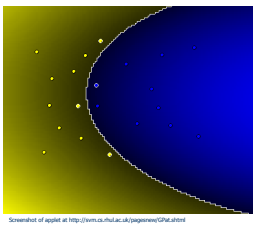
$$0 \leq \alpha_i \leq C \forall i$$

$$\sum_{i=1}^{n}\alpha_i y_i = 0$$

- The equation for the hyperplane stays the same (but use only the points where $\alpha < C$)
- But, no clear interpretation of α anymore, points inside the margin take the limit value C (due to ξ nonnegativity constraint), and thus are support vectors as well!

SINTEF

Hack #2: Kernels



Screenshot of applet at http://svm.cs.rhul.ac.uk/pagesnew/GPat.shtml

- Recall that in L_d the data only occurs as an inner product

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

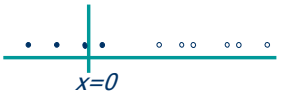
- So, presumably, there is nothing wrong with transforming to another basis for evaluation of this inner product!

$$\max_{\alpha} -\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j y_i y_j \phi(x_i) \cdot \phi(x_j) + \sum_{i=1}^{n}\alpha_i$$

SINTEF

More general decision boundaries

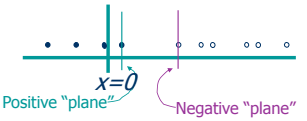
Suppose a simple 1-D dataset



SINTEF

More general decision boundaries

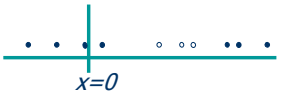
The best classifier found is hardly a surprise



SINTEF

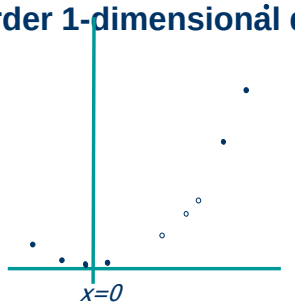
Harder 1-dimensional dataset

Points are not linearly separable. Now what?



SINTEF

Harder 1-dimensional dataset

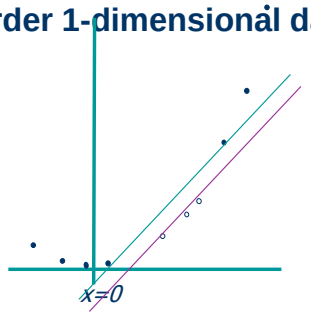


$$\mathbf{z}_k = (x_k, x_k^2)$$

Transform the data points from 1-dim to 2-dim by some nonlinear basis function (called **Kernel functions**)

SINTEF

Harder 1-dimensional dataset



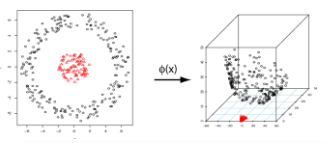
$$\mathbf{z}_k = (x_k, x_k^2)$$

These points are linearly separable now!

Boundary can be found by QP

SINTEF

Data transformation



$$\begin{array}{ccc} \mathbb{R}^2 & \rightarrow & \mathbb{R}^3 \\ (x, y) & \rightarrow & (x^2, y^2, \sqrt{2}xy) \end{array}$$

General SVM approach:

- Transform to a higher dimensional space, and solve a linear problem
- Usually, the space where your problem is linearly solvable is *much* higher dimension, and thus hard work to evaluate

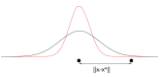
Plots made by J. Wickman

77

SINTEF

Popular kernels

- Polynomial
  $K(x_i, x_j) = (1 + x_i \cdot x_j)^p$
- Gaussian RBF
  $K(x_i, x_j) = \exp[-\| x_i - x_j \|^2 / (2\sigma^2)]$
- Sigmoid (not PD for all parameter choices)
  $K(x_i, x_j) = \tanh(\kappa x_i^T x_j + \theta)^p$
- Specialized kernels for measuring string distance and so on is popular in text classification and related fields
- Furthermore, combinations of kernels by sum or product produces valid kernels, these are usually called composite kernels



SINTEF

Kernelized SVMs

- Change the dual optimization problem to

$$\max_{\alpha} -\frac{1}{2}\sum_{i=1}^{n}\sum_{i=1}^{n}\alpha_i\alpha_j y_i y_j K(x_i, x_j) + \sum_{i=1}^{n}\alpha_i$$
$$0 \le \alpha_i \le C \quad \forall i$$
$$\sum_{i=1}^{n}\alpha_i y_i = 0$$

- The resulting classifier is changed accordingly

$$f(x^*) = \mathrm{sgn}\Big(\sum_{i:0<\alpha_i<C}\alpha_i y_i K(x_i, x^*) + w_0\Big)$$

SINTEF

Typical classifier design with SVMs

- Scale data to a *{0..1}* domain
  - Treat your QP solver nice by giving it smaller inner products to eat
- Choose kernel
  - Actually quite hard, as there does not seem to be any theoretical guidelines here. RBF is regarded as a fairly robust choice.
- Adjust parameters
  - Grid search guided by crossvalidation is usually your only option for parameter choices
  - Rule of thumb[1] when using Gaussian RBFs; search C={2^{-5}, 2^{3}, ..., 2^{15}} and σ ={2^{-15}, 2^{-13}, ..., 2^{3}}

[1] http://www.csie.ntu.edu.tw/~cjlin/papers/guide/guide.pdf

SINTEF

Pros / cons of SVMs

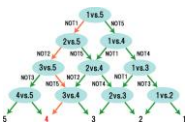
- Advantages
  - Easy to apply on any problem!
  - Powerful implementations freely available[1]
  - Kernels can extend classifier into non-numerical tasks
  - Regularization built into cost function
  - Scales relatively well to high dimension
  - Parameter tuning is straightforward
- Drawbacks
  - Choice of kernel is difficult and problem dependent
  - Slack variables is a heuristic approach to dealing with non-separability
  - Kernelization turns the classifier into a black-box
  - The scale of the optimization problem is directly related to training set size

[1] http://www.csie.ntu.edu.tw/~cjlin/libsvm/
http://svmlight.joachims.org/

SINTEF

Multiple classes?

- Although the optimization problem can be rewritten to handle multiple classes, several authors seem to agree that a one-vs-one approach is the best approach for extending the classifier
  - Solve the task by searching a tree of classifiers or by voting
- Several other approaches have been proposed, but all can be described as a tree of classification tasks



SINTEF

What did you learn today?

- A very simple, but in practice very useful general algorithm for clustering data, the K-means
- The general idea of using optimization to design a classification rule, Support Vector Machines
- At the exam you will be expected to be able to describe these two algorithms
- The exercise for this lecture will introduce you to computer tools that hide a lot of the details covered