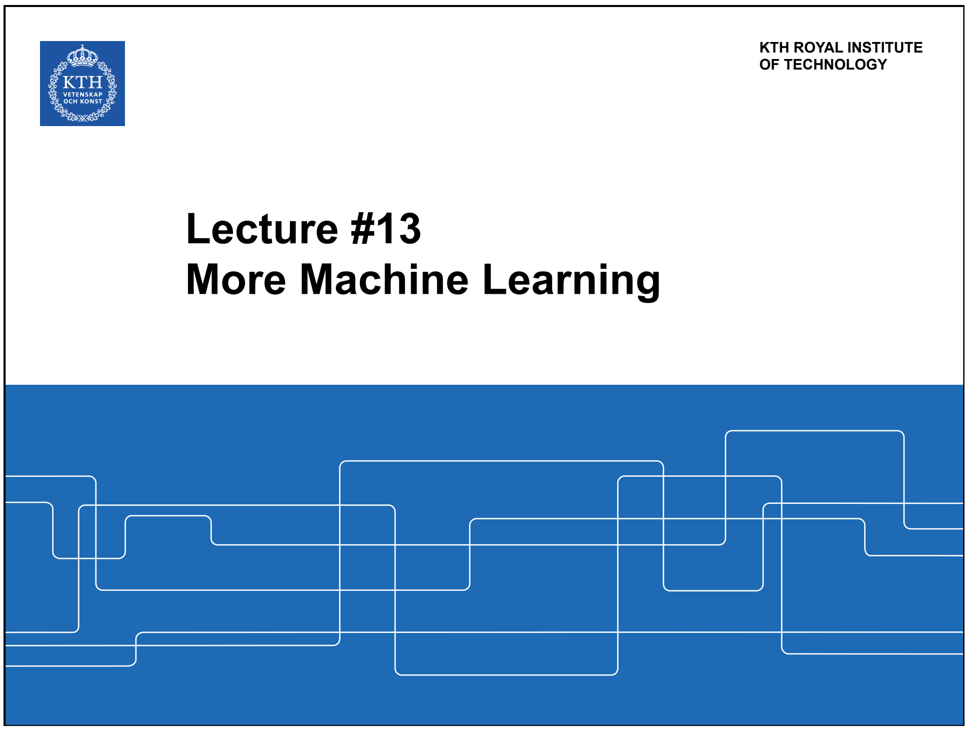
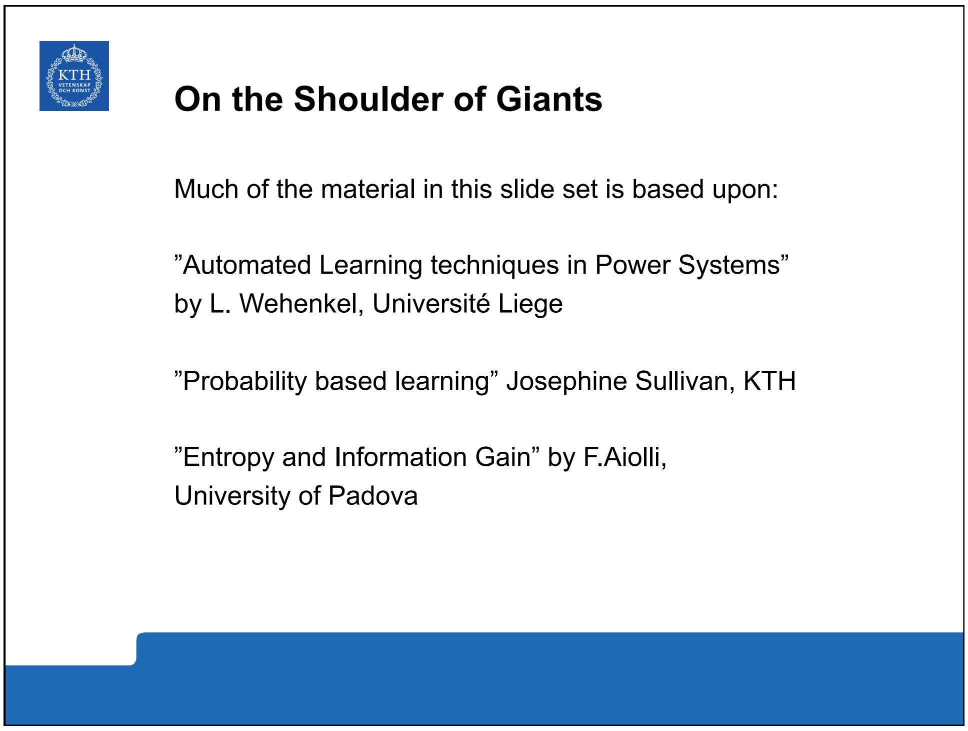


KTH ROYAL INSTITUTE
OF TECHNOLOGY

Lecture #13
More Machine Learning



On the Shoulder of Giants

Much of the material in this slide set is based upon:

”Automated Learning techniques in Power Systems” by L. Wehenkel, Université Liege

”Probability based learning” Josephine Sullivan, KTH

”Entropy and Information Gain” by F.Aiolli, University of Padova

Contents

Repeating from last time

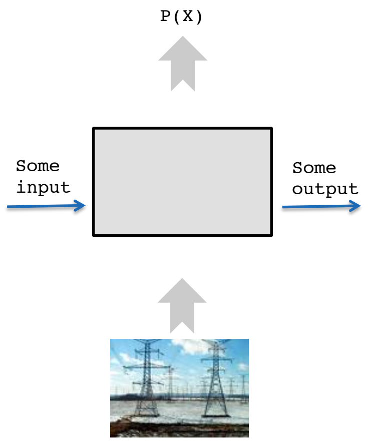
Artificial Neural Networks

K-Nearest Neighbour

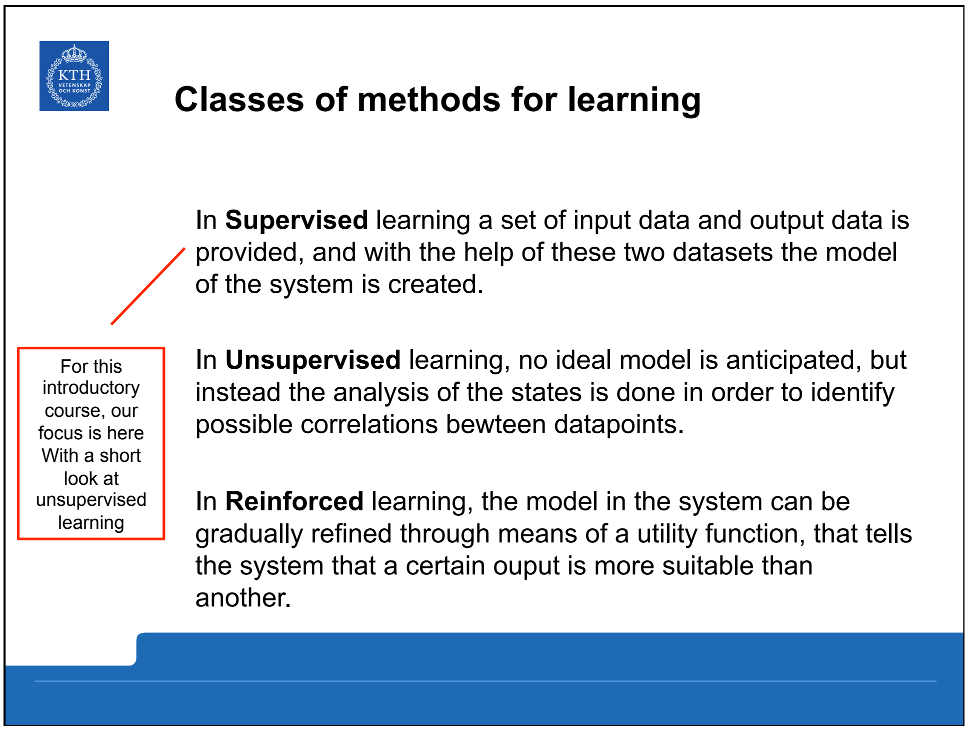
Power Systems Analysis – An automated learning approach

Understanding states in the power system is established through observation of inputs and outputs without regard to the physical electrotechnical relations between the states.

Adding knowledge about the electrotechnical rules means adding heuristics to the learning.



Given a set of examples (the learning set (LS)) of associated input/output pairs, derive a general rule representing the underlying input/output relationship, which may be used to explain the observed pairs and/or predict output values for any new unseen input.



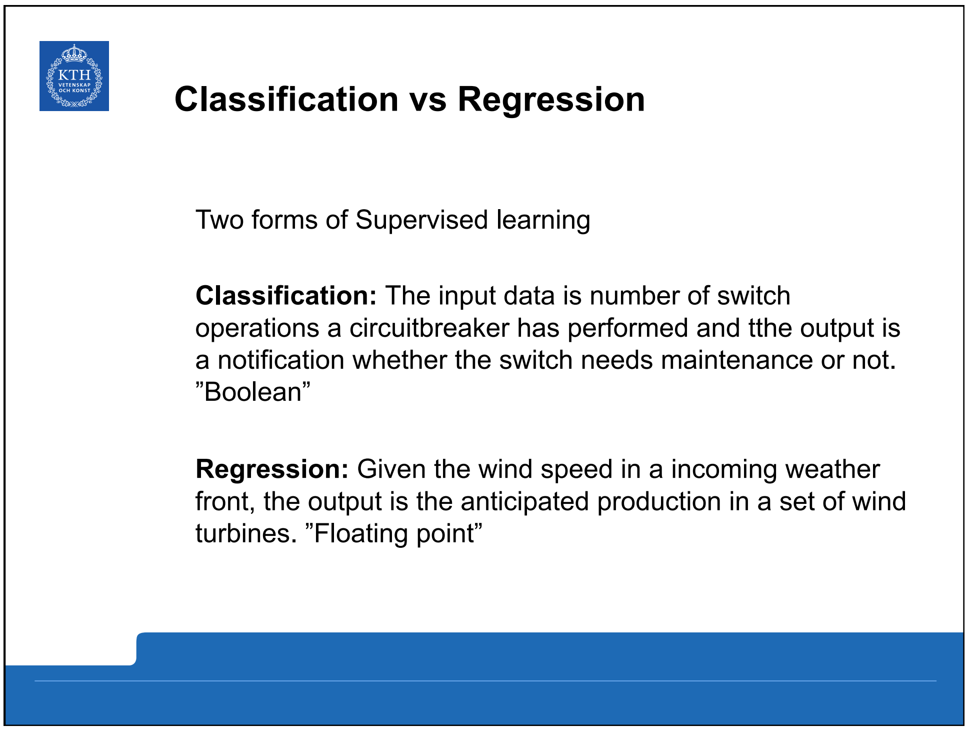
Classes of methods for learning

For this introductory course, our focus is here With a short look at unsupervised learning

In **Supervised** learning a set of input data and output data is provided, and with the help of these two datasets the model of the system is created.

In **Unsupervised** learning, no ideal model is anticipated, but instead the analysis of the states is done in order to identify possible correlations bewteen datapoints.

In **Reinforced** learning, the model in the system can be gradually refined through means of a utility function, that tells the system that a certain ouput is more suitable than another.



Classification vs Regression

Two forms of Supervised learning

Classification: The input data is number of switch operations a circuitbreaker has performed and tthe output is a notification whether the switch needs maintenance or not. ”Boolean”

Regression: Given the wind speed in a incoming weather front, the output is the anticipated production in a set of wind turbines. ”Floating point”

Supervised learning - a preview

In the scope of this course, we will be studying three forms of supervised learning.

- Decision Trees

  Overview and practical work on exercise session.

- Artificial Neural Networks

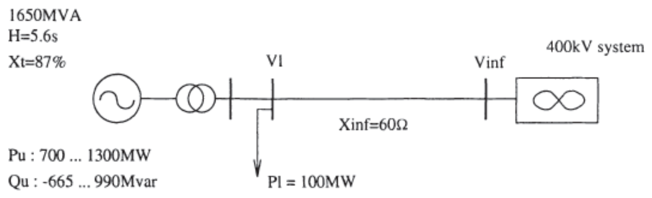
  Overview only, no practical work.

- Statistical methods – k-Nearest Neighbour

  Overview and practical work on exercise session. Also included in Project Assignment

  kNN algorithm can also be used for unsupervised clustering.

Example from Automatic Learning techniques in Power Systems



One Machine Infinite Bus (OMIB) system

- Assuming a fault close to the Generator will be cleared within 155 ms by protection relays
- We need to identify situations in which this clearing time is sufficient and when it is not
- Under certain loading situations, 155 ms may be too slow.

Source: Automatic Learning techniques in Power Systems, L. Wehenkel

OMIB – further information

In the OMIB system the following parameters influence security

- Amount of active and reactive power of the generator (
- Amount of load nearby the generator (Pl)
- Voltage magnitudes at the load bus and at the infinite bus Short-circuit reactance Xinf, representing the effect of variable topology in the large system represented by the infinite bus.

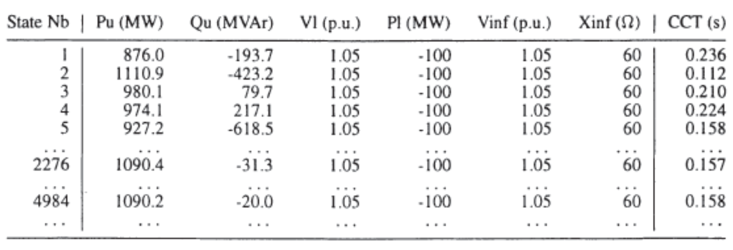
In the example, Voltages at generator and Infinite bus are assumed similar and constant for simplicity

Source: Automatic Learning techniques in Power Systems, L. Wehenkel

Our database of objects with attributes

In a simulator, we randomly sample values for P_u and Q_u creating a database with 5000 samples (objects) and for each object we have a set of attributes $(P_u, Q_u, V_1, P_1, V_{inf}, X_{inf}, CCT)$ as per below.

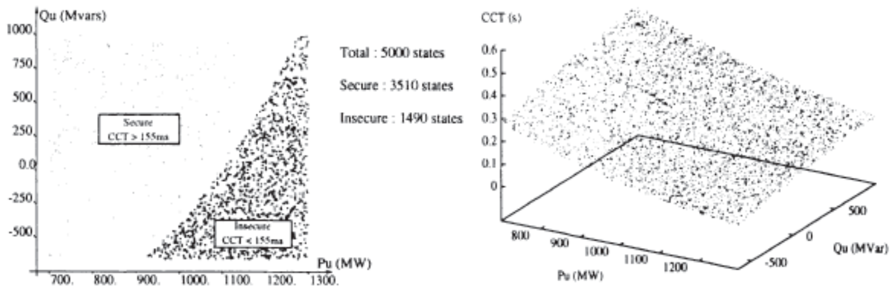
Table 1.1. Sample of OMIB operating states



State Nb	Pu (MW)	Qu (MVAr)	Vl (p.u.)	Pl (MW)	Vinf (p.u.)	Xinf (Ω)	CCT (s)
1	876.0	-193.7	1.05	-100	1.05	60	0.236
2	1110.9	-423.2	1.05	-100	1.05	60	0.112
3	980.1	79.7	1.05	-100	1.05	60	0.210
4	974.1	217.1	1.05	-100	1.05	60	0.224
5	927.2	-618.5	1.05	-100	1.05	60	0.158
...	...	...	...	...	...	...	...
2276	1090.4	-31.3	1.05	-100	1.05	60	0.157
...	...	...	...	...	...	...	...
4984	1090.2	-20.0	1.05	-100	1.05	60	0.158
...	...	...	...	...	...	...	...

Source: Automatic Learning techniques in Power Systems, L. Wehenkel

Plot of database content



Source: Automatic Learning techniques in Power Systems, L. Wehenkel

How to measure information content

Entropy $\mathbf{H}$ is a measure of *Unpredictability*.

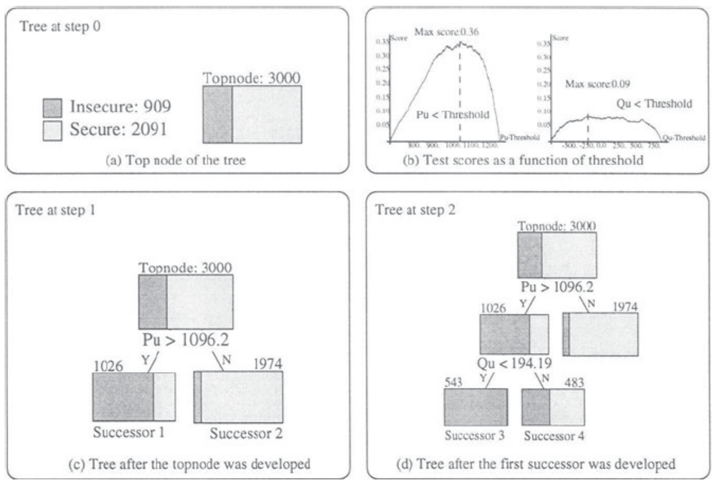
Defined as:

$$-\sum p_i \log p_i$$

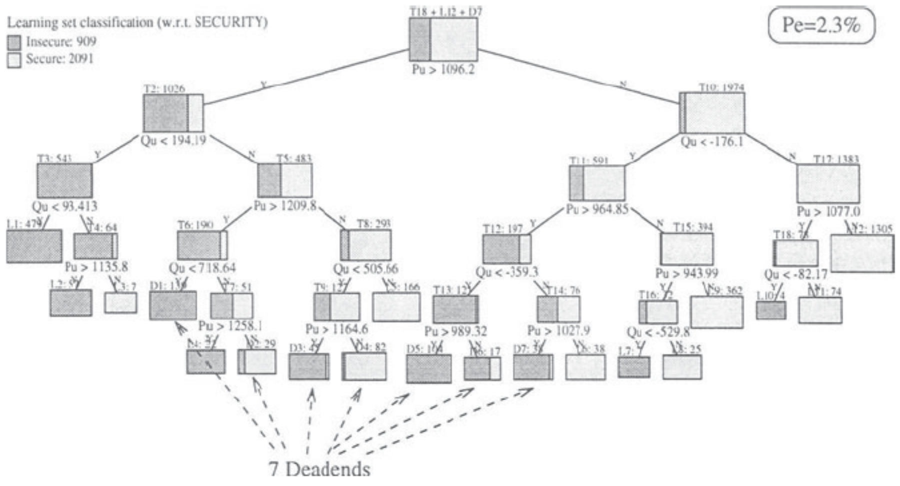
Where

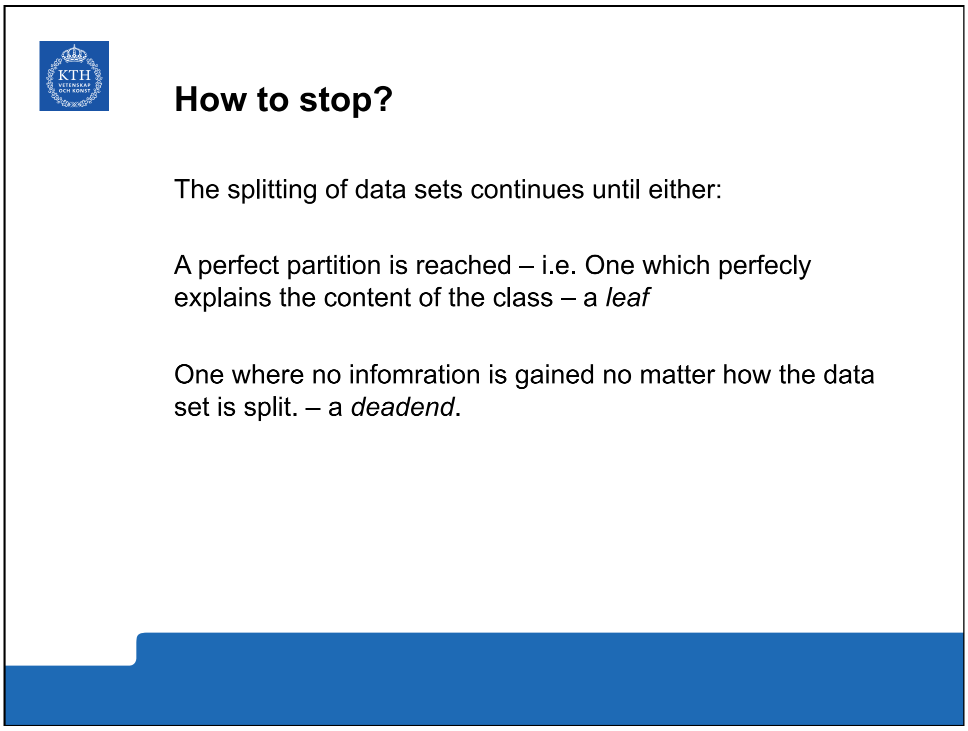
p_i is the probability of event i

Gradual expansion of the Decision Tree



Complete Decision Tree



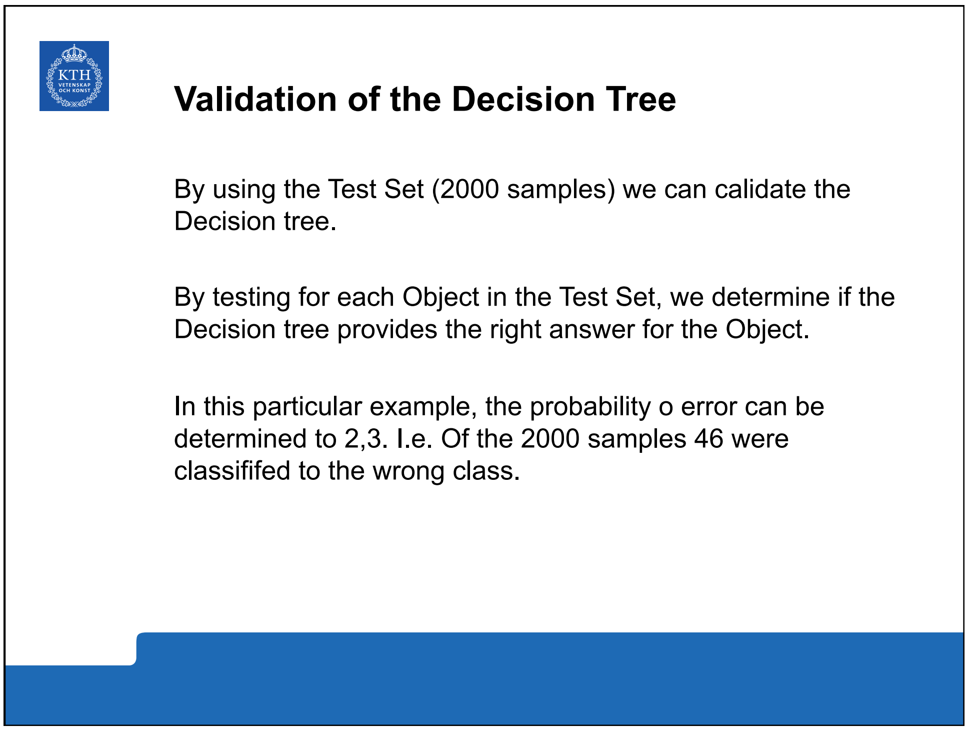


How to stop?

The splitting of data sets continues until either:

A perfect partition is reached – i.e. One which perfecly explains the content of the class – a *leaf*

One where no infomration is gained no matter how the data set is split. – a *deadend*.

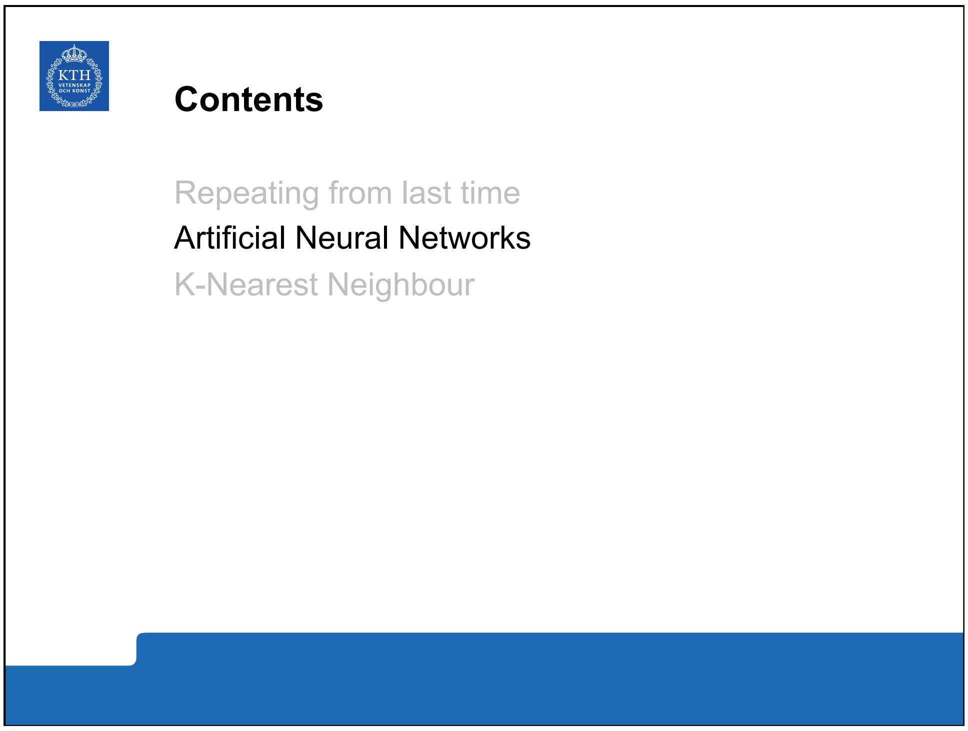


Validation of the Decision Tree

By using the Test Set (2000 samples) we can calidate the Decision tree.

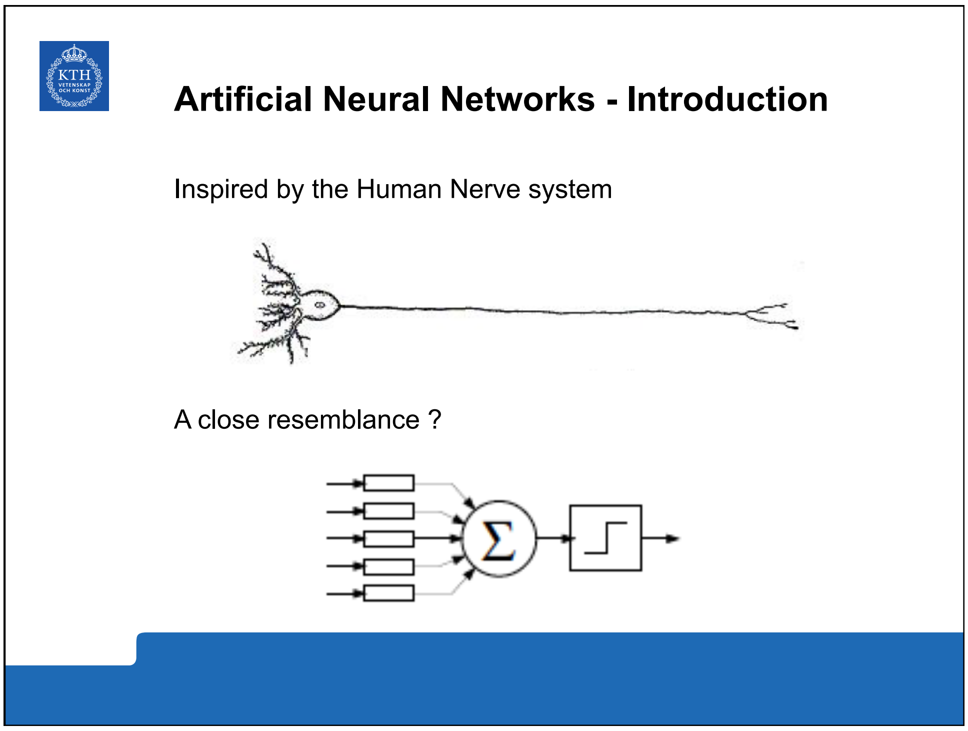
By testing for each Object in the Test Set, we determine if the Decision tree provides the right answer for the Object.

In this particular example, the probability o error can be determined to 2,3. I.e. Of the 2000 samples 46 were classifed to the wrong class.



Contents

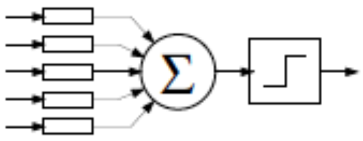
Repeating from last time
Artificial Neural Networks
K-Nearest Neighbour



Artificial Neural Networks - Introduction

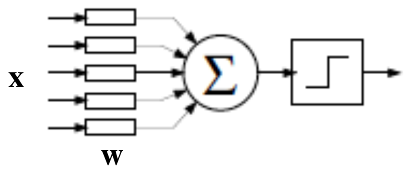
Inspired by the Human Nerve system

A close resemblance ?



The Perceptron – Artifical Neuron

- The perceptron takes inputs x
- The inputs are weighted w
- The Perceptron sums the values of the inputs
- Provides as output a threshold function based on the sum
- **Linear** perceptron provides sum as output
- Non-linear provide a output function.

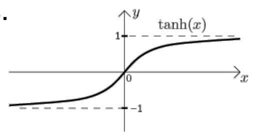


$$A_j(\bar{x}, \bar{w}) = \sum_{i=0}^{n} x_i w_{ji}$$

Non Linear Perceptrons

Linear Perceptrons provide output as sum of weighted inputs.

Non-linear perceptrons normally use a threhold function for the output, to limit the extreme values.

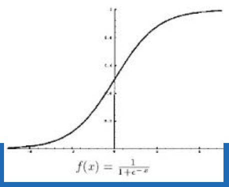


Some examples are: tanh(x)

Signmoidal function

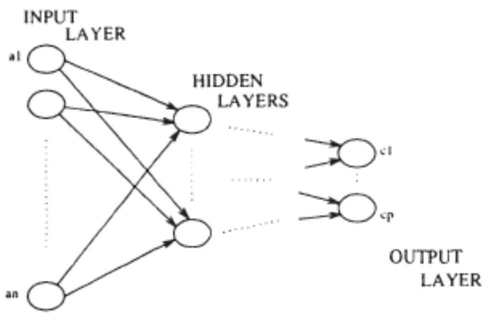
$$O_j(\bar{x}, \bar{w}) = \frac{1}{1+e^{A_j(\bar{x}, \bar{w})}} \qquad A_j(\bar{x}, \bar{w}) = \sum_{i=0}^{n} x_i w_{ji}$$

0 -> 0,5
<< -1 -> 0
>>1 -> 1



Artificial Neural Network

Multi Layer Perceptrons (MLP)
A network of interconnected Perceptrons in several layers
First layer recives input, forwards to second layer etc.
Normally one hidden layer is sufficient to create good mappings



Where is the ”learning” in ANN

Given an input vector $a(o)$ (attributes of an object)

For a classification problem

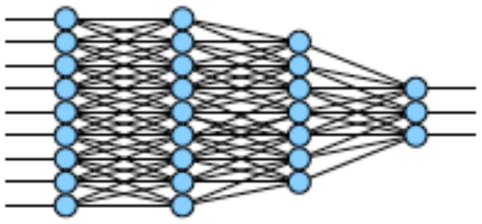
We want to assign it to a class C_i

For a regression problem

We want it to approximate a value y

We have to tune the weights of the inputs of the perceptrons

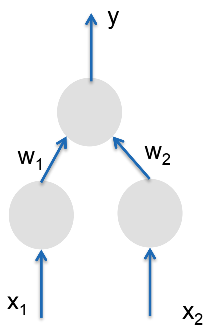
So how to tune the weights in this …?



10s of perceptrons, 100s of links, 1000s of input values…

Backpropagation algorithm

Trivial case



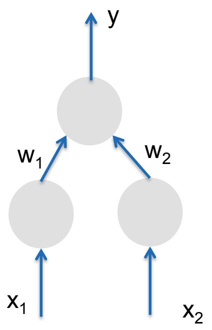
Remember, we are discussing **supervised** learning.

This means we have a sets of the following form: (x_1,x_2,t)

Input attributes and a correct target value t, that we want to achieve.

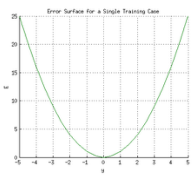
Backpropagation algorithm

Trivial case



The Least Squares Error

$$E = (t - y)^2$$



For a linear Perceptron

$$y = x_1 w_1 + x_2 w_2$$

Find minima of $E(y)$ w.r.t (w_1, w_2)

Finding minima - gradient descent

In the general case, we want to find the minima of the Error function with regards to the weights

$$E_j(\bar{x}, \bar{w}, d) = \left(O_j(\bar{x}, \bar{w}) - d_j\right)^2$$

If we can find the derivative of the error function, we can use that to find a suitable adjustment of the weights

$$\Delta w_{ji} = -\eta \frac{\partial E}{\partial w_{ji}}$$

$$\frac{\partial E}{\partial O_j} = 2(O_j - d_j)$$

………..

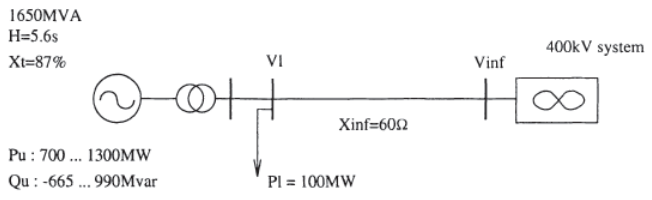
$$\Delta w_{ji} = -2\eta(O_j - d_j)O_j(1 - O_j)x_i$$

Where are we now?

1. We have created a ANN with a ”suitable number of layers” and perceptrons
2. We have chosen a threshold function for the perceptrons
3. We have allocated random weights to all links
4. We use our Training set to tune the weights using the Backdrop algorithm.
5. In the Backgrop algorithm we had to determine the minima of the error function and derived a formula on how to adjust weights to reach the minima.
6. We adjust the weights, try a new test set and run the whole process again.

And this was for a ANN with one layer…..

Example from Automatic Learning techniques in Power Systems

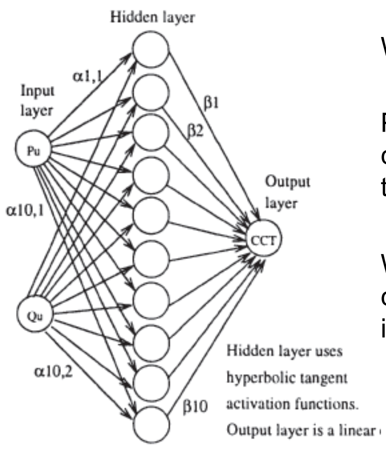


One Machine Infinite Bus (OMIB) system

- Assuming a fault close to the Generator will be cleared within 155 ms by protection relays
- We need to identify situations in which this clearing time is sufficient and when it is not
- Under certain loading situations, 155 ms may be too slow.

Source: Automatic Learning techniques in Power Systems, L. Wehenkel

Initial ANN for the OMIB problem



Weights are random

Perceptrons use linear combination of inputs and tanh function

We want to calculate the clearing time (CCT), i.e. This is a **Regression** problem

$$\text{Output}_i(\text{state}) = \tanh(\alpha_{i,1} Pu(\text{state}) + \alpha_{i,2} Qu(\text{state}) + \theta_i),$$

Output and Error function

The Output function is:

$$\text{CCT}_{\text{MLP}}(\text{state}) = \sum_{i=1\ldots10} \beta_i \tanh(\alpha_{i,1} Pu(\text{state}) + \alpha_{i,2} Qu(\text{state}) + \theta_i),$$

The error function is:

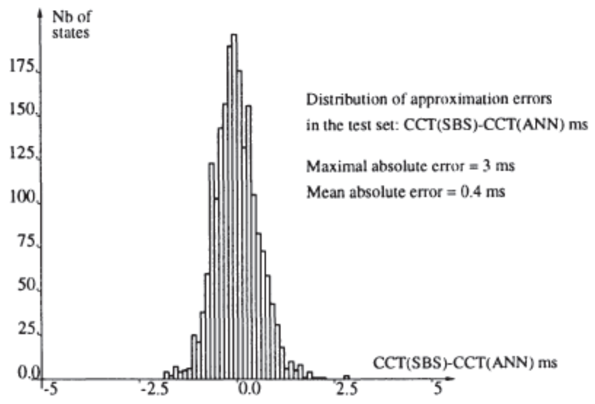
$$SE = N^{-1} \sum_{\text{state} \in \boldsymbol{LS}} |\text{CCT}(\text{state}) - \text{CCT}_{\text{MLP}}(\text{state})|^2,$$

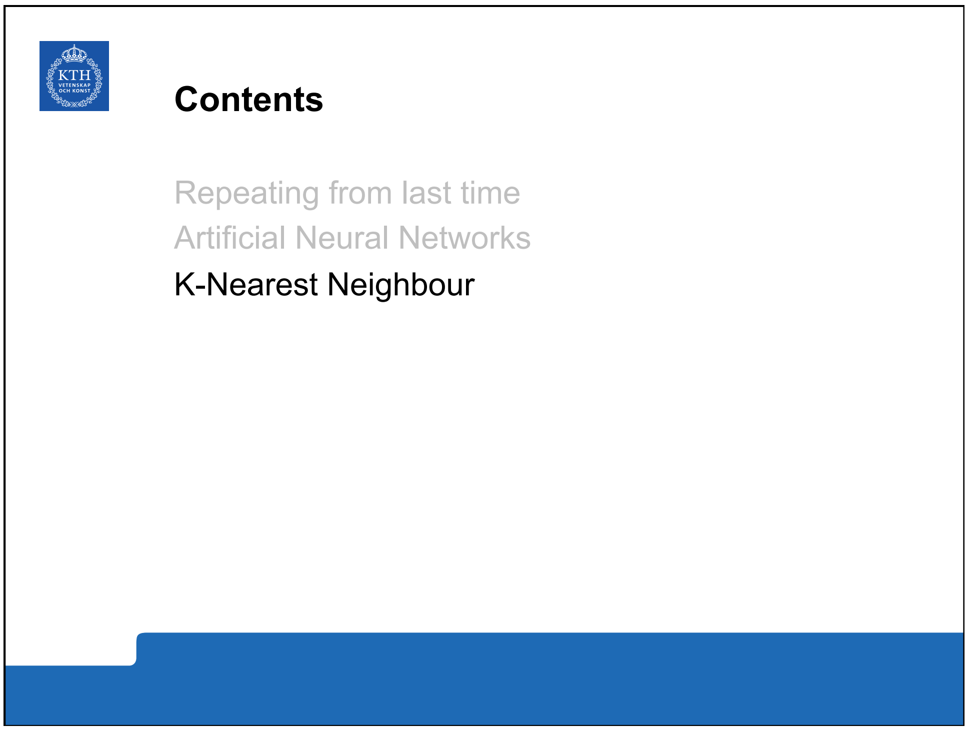
The final ANN structure is

After 46 iterations

$$
\begin{aligned}
\mathrm{CCT}_{\mathrm{MLP}} = \; & -0.602710\tanh(0.000194Pu - 0.00034Qu - 0.93219)\\
& -0.401320\tanh(0.000822Pu - 0.00020Qu - 0.76681)\\
& +0.318249\tanh(0.000239Pu - 0.00050Qu - 0.29351)\\
& -0.287230\tanh(0.002004Pu - 0.00034Qu - 1.20080)\\
& +0.184522\tanh(0.000131Pu - 0.00057Qu - 0.03152)\\
& +0.177701\tanh(0.001799Pu - 0.00011Qu - 2.08190)\\
& -0.150720\tanh(0.001530Pu - 0.00056Qu - 1.68040)\\
& +0.142678\tanh(0.002152Pu - 0.00046Qu - 1.72280)\\
& -0.067897\tanh(0.001910Pu - 0.00051Qu - 1.71343)\\
& -0.056020\tanh(0.000202Pu - 0.00085Qu - 0.39876)
\end{aligned}
$$

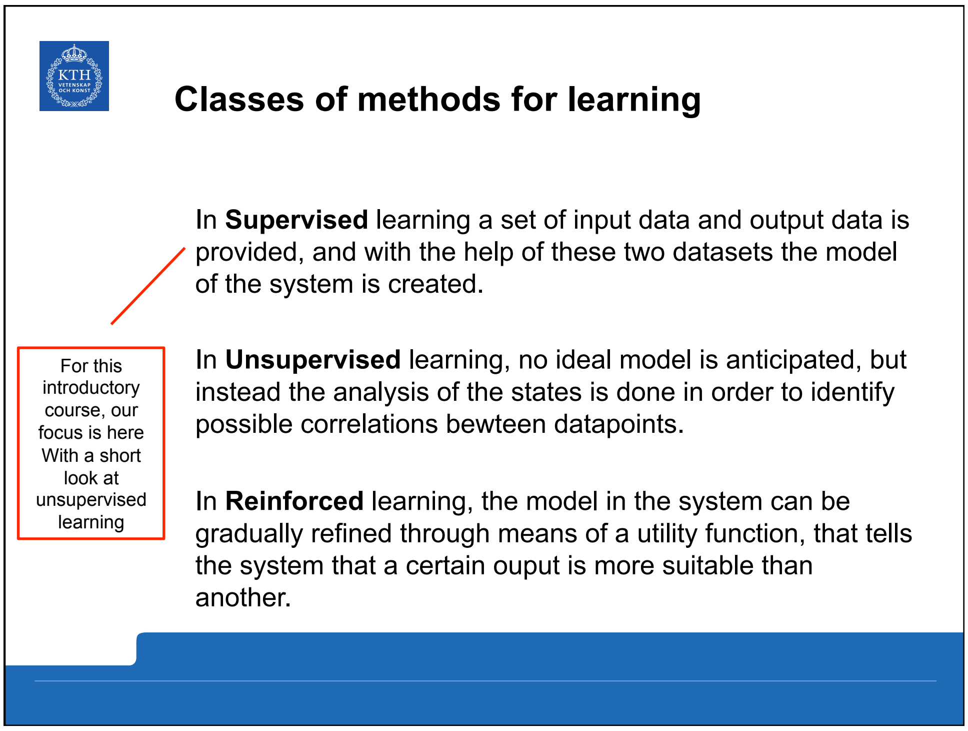
Error estimation with Test set





Contents

Repeating from last time
Artificial Neural Networks
K-Nearest Neighbour



Classes of methods for learning

In **Supervised** learning a set of input data and output data is provided, and with the help of these two datasets the model of the system is created.

For this introductory course, our focus is here With a short look at unsupervised learning

In **Unsupervised** learning, no ideal model is anticipated, but instead the analysis of the states is done in order to identify possible correlations bewteen datapoints.

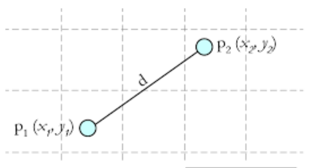
In **Reinforced** learning, the model in the system can be gradually refined through means of a utility function, that tells the system that a certain ouput is more suitable than another.

The k Nearest Neighbour algorithm

The Nearest Neighbour algorithm is a way to classify objects with attributes to its nearest neighbour in the Learning set.

In k–Nearest Neighbour, the k nearest neighbours are considered.

”Nearest” is measured as distance in Euclidean space.



Euclidean distance (d) $= \sqrt{(x_2-x_1)^2 + (y_2-y_1)^2}$

Lazy vs. Eager learning

In Eager learning, the Training set is pre-classified. All objects in the Learning set are clustered with regards to their neighbours.

In Lazy learning, only when a new object is input to the algorithm is the distance calculated.

K-means clustering

K-means clustering involves creating clusters of data

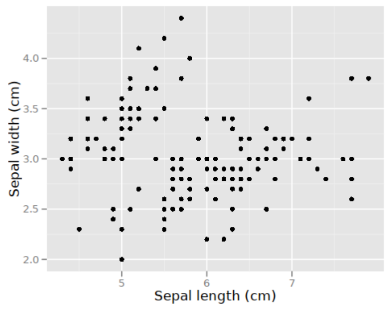
It is iterative and continues until no more clusters can be created

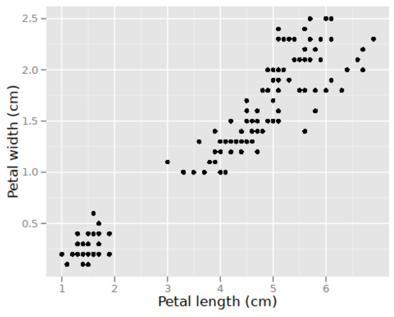
It requires the value of k to be defined at start.

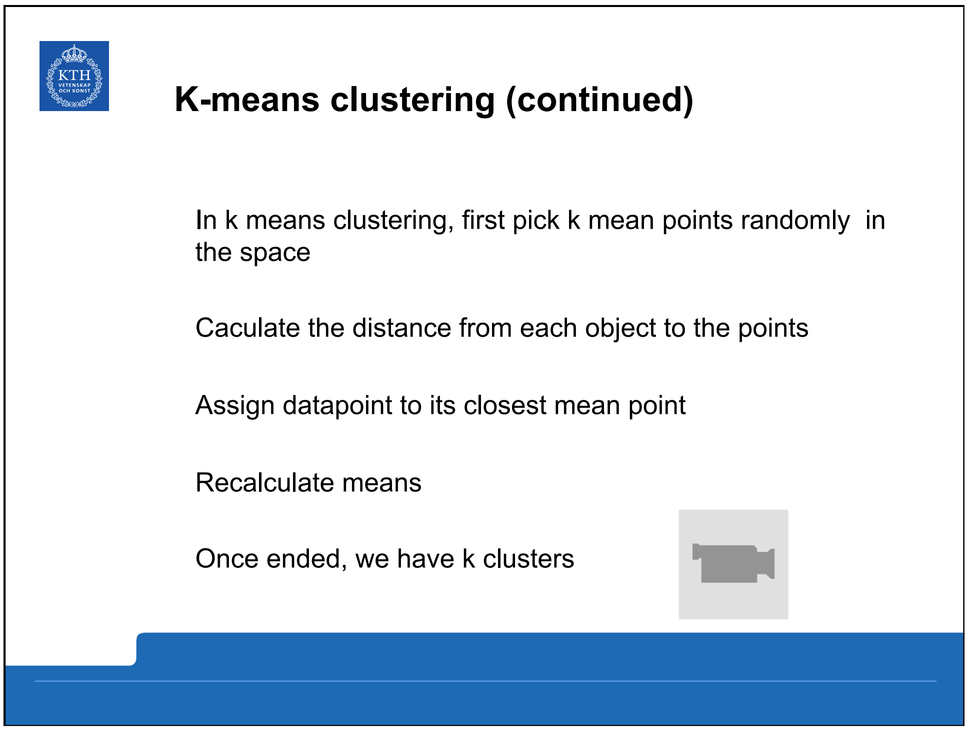
Consider for instance a table like the following:

Sepal length	Sepal width	Petal length	Petal width
5.1	3.5	1.4	0.2
4.9	3.0	1.4	0.2
4.7	3.2	1.3	0.2
…	…	…	…

Plotted the data looks something like







K-means clustering (continued)

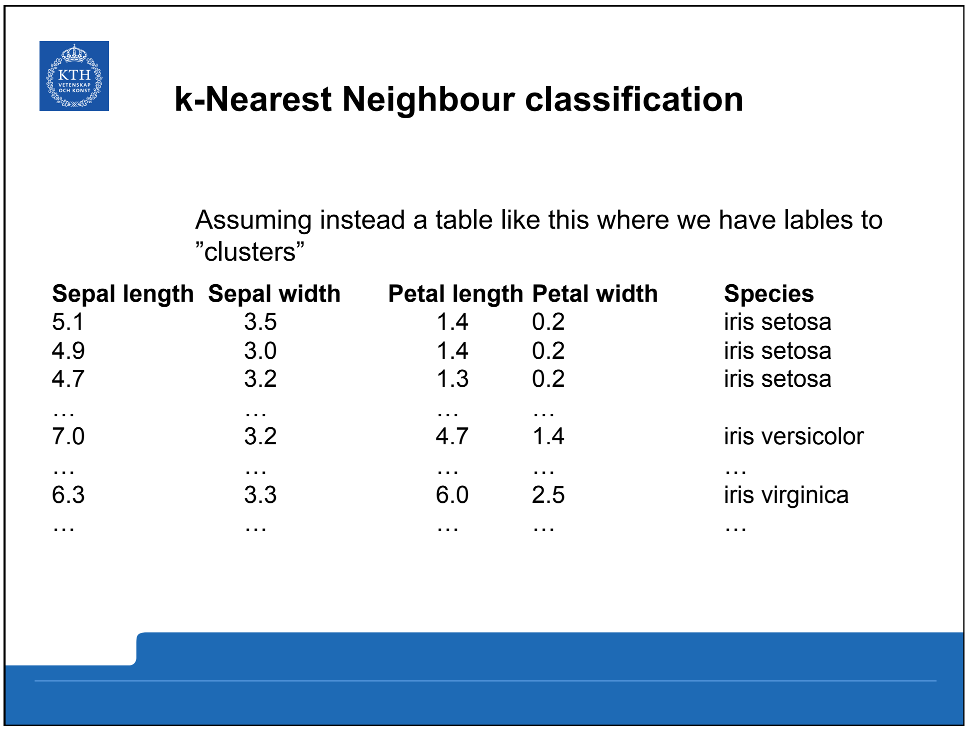
In k means clustering, first pick k mean points randomly in the space

Caculate the distance from each object to the points

Assign datapoint to its closest mean point

Recalculate means

Once ended, we have k clusters



k-Nearest Neighbour classification

Assuming instead a table like this where we have lables to ”clusters”

Sepal length	**Sepal width**	**Petal length**	**Petal width**	**Species**
5.1	3.5	1.4	0.2	iris setosa
4.9	3.0	1.4	0.2	iris setosa
4.7	3.2	1.3	0.2	iris setosa
…	…	…	…	
7.0	3.2	4.7	1.4	iris versicolor
…	…	…	…	…
6.3	3.3	6.0	2.5	iris virginica
…	…	…	…	…

K-Nearest Neighbour algorithm

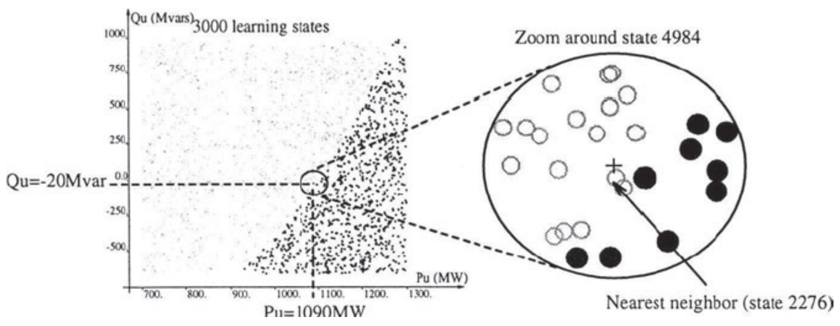
Given a new set of measurements, perform the following test:

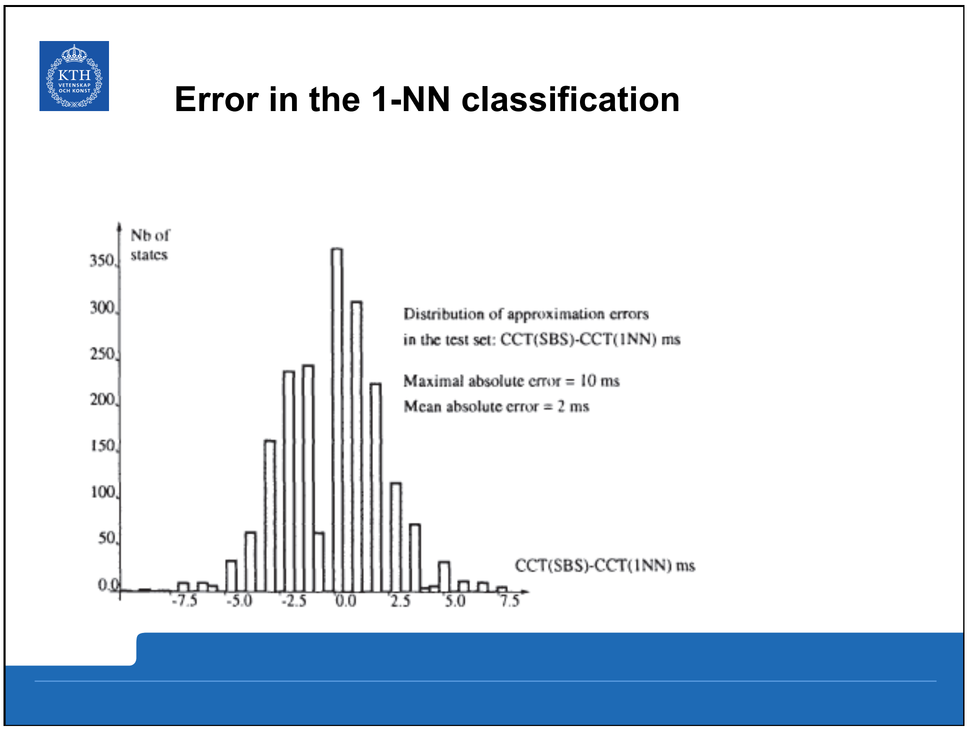
Find (using Euclidean distance, for example), the k nearest entities from the training set. These entities have known labels. The choice of k is left to us.

Among these k entities, which label is most common? That is the label for the unknown entity.

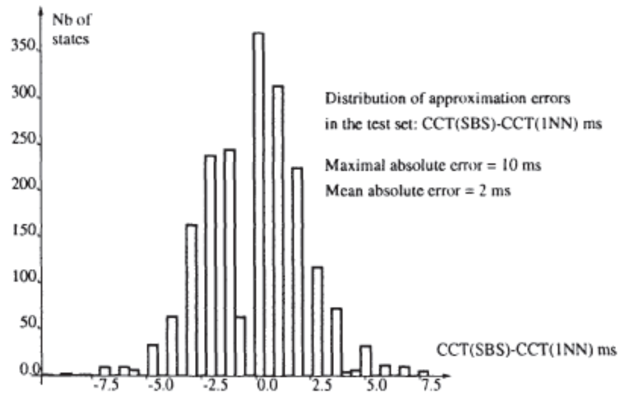
In the OMIB example database

Sample 4984, and its neighbours





Error in the 1-NN classification



Distribution of approximation errors
in the test set: CCT(SBS)-CCT(1NN) ms

Maximal absolute error = 10 ms
Mean absolute error = 2 ms