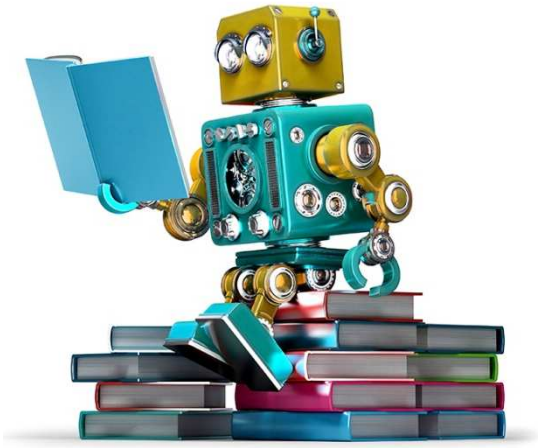
MACHINE LEARNING



MODULE-V CHAPTER 8
INSTANCE BASED LEARNING,

BY

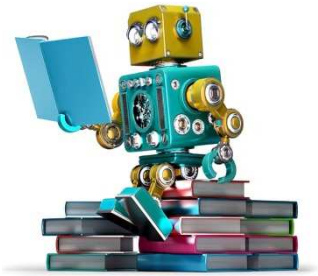
HARIVINOD N

VIVEKANANDA COLLEGE OF ENGINEERING TECHNOLOGY, PUTTUR

Module 5 - Outline

Chapter 8: Instance Based Learning

1. **Introduction**
2. K-nearest neighbor Learning
3. Locally Weighted regression
4. Radial basis functions
5. Case based reasoning
6. Summary



Introduction

- all learning methods presented so far construct a general explicit description of the target function when examples are provided
- **Instance-based learning:**
  - examples are simply stored
  - generalizing is postponed until a new instance must be classified
  - in order to assign a target function value, its relationship to the previously stored examples is examined
  - sometimes referred to as **lazy learning**

Introduction

- **advantages:**
  - instead of estimating for the whole instance space, local approximations to the target function are possible
  - especially if target function is complex but still decomposable
- **disadvantages:**
  - classification costs are high

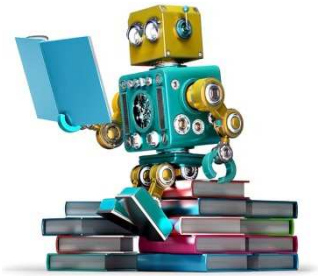
    efficient techniques for indexing examples are important to reduce computational effort
  - typically all attributes are considered when attempting to retrieve similar training examples

    if the concept depends only on a few attributes, the truly most similar instances may be far away

Module 5 - Outline

Chapter 8: Instance Based Learning

1. Introduction
2. **K-nearest neighbor Learning**
3. Locally Weighted regression
4. Radial basis functions
5. Case based reasoning
6. Summary



K-nearest neighbor learning
(For classification and regression)

MACHINE LEARNING

- most basic instance-based method
- **assumption:**
  - instances correspond to a point in a n-dimensional space $\Re^n$
  - thus, nearest neighbors are defined in terms of the standard **Euclidean Distance**

$$d(x_i, x_j) \equiv \sqrt{\sum_{r=1}^{n} (a_r(x_i) - a_r(x_j))^2}$$

where an instance x is described by $< a_1(x), a_2(x), ..., a_n(x) >$

- target function may be either discrete-valued or real-valued

K-nearest neighbor learning

MACHINE LEARNING

- **discrete-valued target function:**
  - $f : \Re^n \rightarrow V$ where V is the finite set $\{v_1, v_2, ..., v_s\}$
  - the target function value is the most common value among the k nearest training examples

$$\hat{f}(x_q) \leftarrow \underset{v \in V}{argmax} \sum_{i=1}^{k} \delta(v, f(x_i))$$

where $\delta(a, b) = (a == b)$

- **continuous-valued target function:**
  - algorithm has to calculate the mean value instead of the most common value
  - $f : \Re^n \rightarrow \Re$

$$\hat{f}(x_q) \leftarrow \frac{\sum_{i=1}^{k} f(x_i)}{k}$$

KNN Algorithm for classification

MACHINE LEARNING

Training algorithm:

- For each training example $\langle x, f(x) \rangle$, add the example to the list *training_examples*

Classification algorithm:

- Given a query instance x_q to be classified,
  - Let $x_1 \ldots x_k$ denote the k instances from *training_examples* that are nearest to x_q
  - Return

$$\hat{f}(x_q) \leftarrow \underset{v \in V}{\operatorname{argmax}} \sum_{i=1}^{k} \delta(v, f(x_i))$$

where $\delta(a, b) = 1$ if $a = b$ and where $\delta(a, b) = 0$ otherwise.

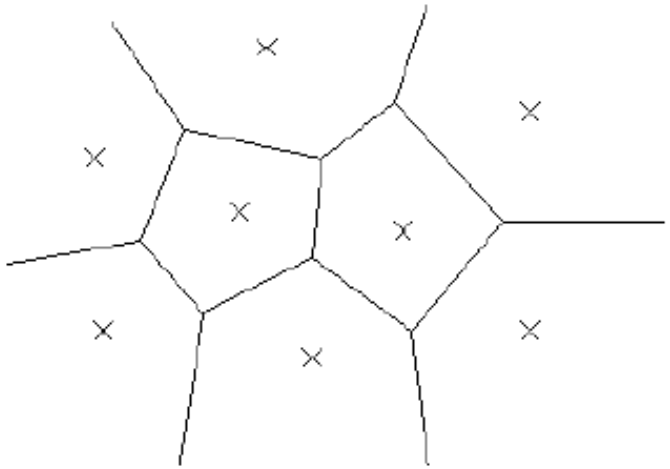
TABLE 8.1

The k-NEAREST NEIGHBOR algorithm for approximating a discrete-valued function $f : \Re^n \rightarrow V$.

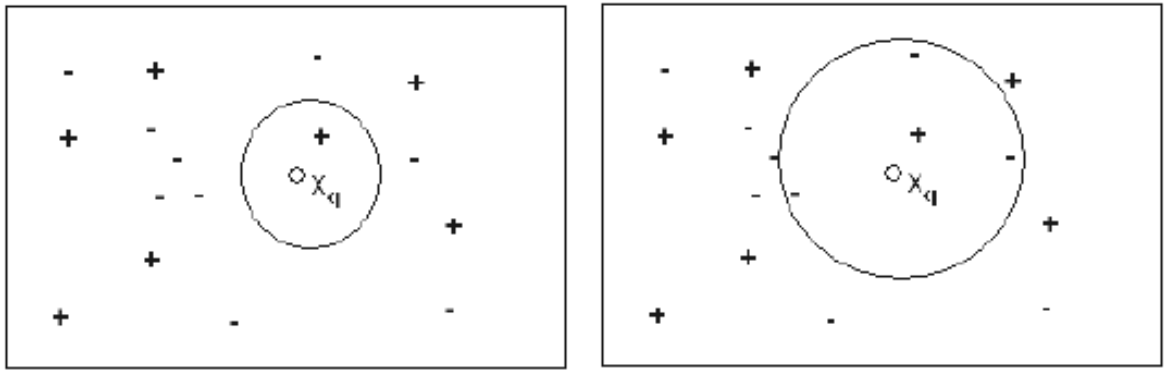
K-NN Hypothesis Space

MACHINE LEARNING

- **no explicit** hypothesis is formed
- decision surface is a combination of convex polyhedra surrounding each of the training examples
- for each training example, the polyhedron indicates the set of possible query points x_q whose classification is completely determined by this training example (**Voronoi diagram**)



K-nearest neighbor learning



- e.g. instances are points in a two-dimensional space where the target function is boolean-valued
  - 1-nearest neighbor: x_q is classified positive
  - 4-nearest neighbor: x_q is classified negative

Distance Weighted Nearest Neighbor

MACHINE LEARNING

- contribution of each of the k nearest neighbors is weighted accorded to their distance to x_q
  - **discrete-valued target functions**

$$\hat{f}(x_q) \leftarrow \underset{v \in V}{argmax} \sum_{i=1}^{k} w_i \delta(v, f(x_i))$$

where $w_i \equiv \frac{1}{d(x_q, x_i)^2}$ and $\hat{f}(x_q) = f(x_i)$ if $x_q = x_i$

  - **continuous-valued target function:**

$$\hat{f}(x_q) \leftarrow \frac{\sum_{i=1}^{k} w_i f(x_i)}{\sum_{i=1}^{k} w_i}$$

Remarks on K-NN

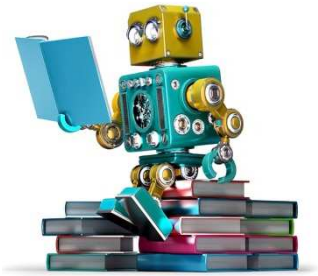
MACHINE LEARNING

- highly effective inductive inference method for many practical problems provided a sufficiently large set of training examples
- robust to noisy data
- weighted average smoothes out the impact of isolated noisy training examples
- **inductive bias of k-nearest neighbors**
  - assumption that the classification of x_q will be similar to the classification of other instances that are nearby in the Euclidean Distance
- **curse of dimensionality**
  - distance is based on all attributes
  - in contrast to decision trees and inductive logic programming
  - solutions to this problem
    - attributes can be weighted differently
    - eliminate least relevant attributes from instance space

Module 5 - Outline

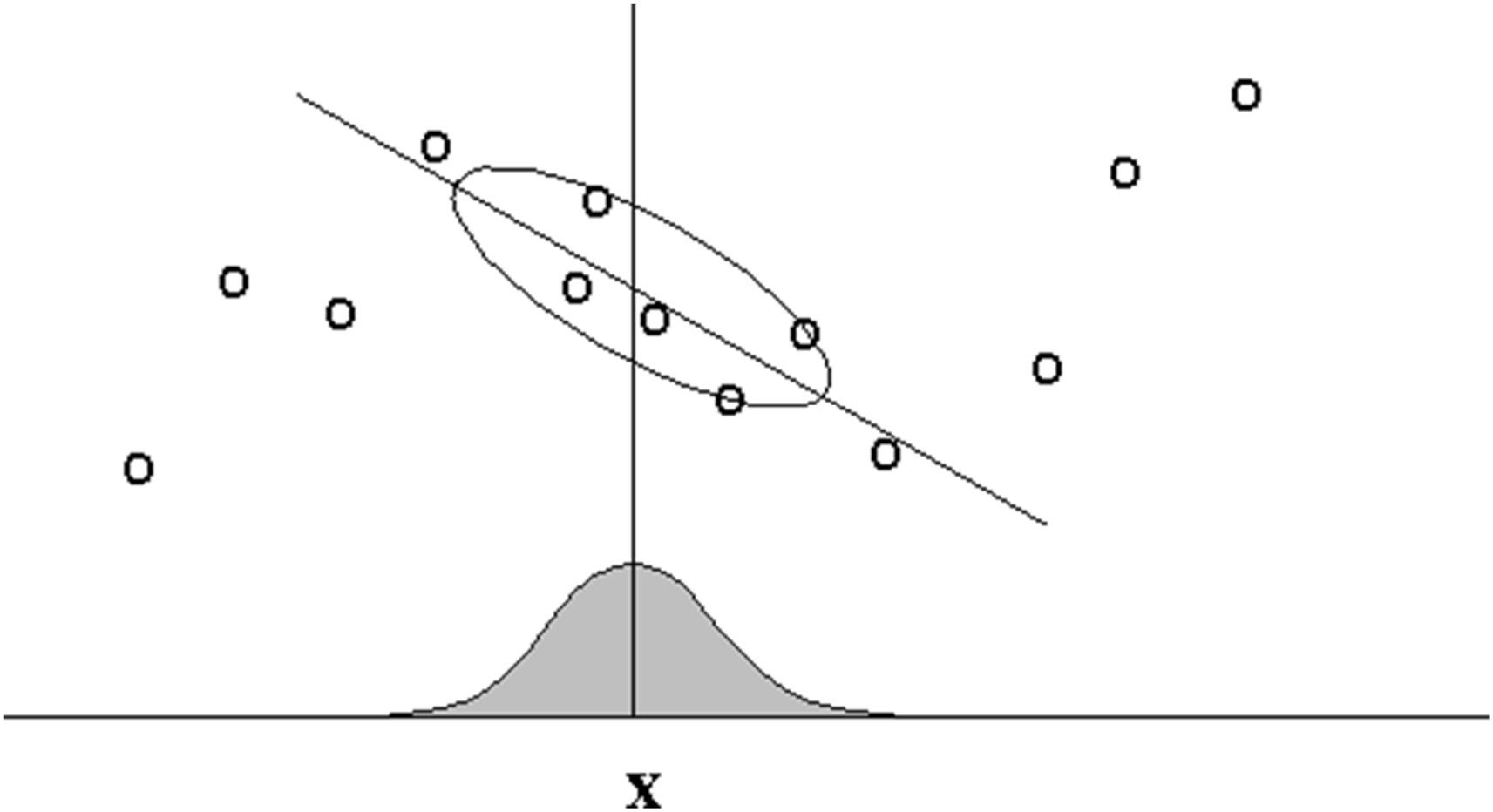
Chapter 8: Instance Based Learning

1. Introduction
2. K-nearest neighbor Learning
3. **Locally Weighted regression**
4. Radial basis functions
5. Case based reasoning
6. Summary



Locally Weighted Regression

- a note on terminology:
  - *Regression* means approximating a real-valued target function
  - *Residual* is the error $\hat{f}(x) - f(x)$ in approximating the target function
  - *Kernel function* is the function of distance that is used to determine the weight of each training example. In other words, the kernel function is the function K such that $w_i = K(d(x_i, x_q))$
- nearest neighbor approaches can be thought of as approximating the target function at the single query point x_q
- locally weighted regression is a generalization to this approach, because it constructs an explicit approximation of f over a local region surrounding x_q


X

Locally weighted linear regression

- target function is approximated using a **linear function**

$$\hat{f}(x) = w_0 + w_1 a_1(x) + ... + w_n a_n(x)$$

- methods like **gradient descent** can be used to calculate the coefficients $w_0, w_1, ..., w_n$ to minimize the error in fitting such linear functions
- ANNs require a global approximation to the target function
- here, just a local approximation is needed

$\Rightarrow$ the error function has to be redefined

Locally weighted linear regression

- possibilities to redefine the error criterion E
  1. Minimize the squared error over just the k nearest neighbors

$$E_1(x_q) \equiv \frac{1}{2} \sum_{x \in k \text{ nearest neigbors}} (f(x) - \hat{f}(x))^2$$

  2. Minimize the squared error over the entire set D, while weighting the error of each training example by some decreasing function K of its distance from x_q

$$E_2(x_q) \equiv \frac{1}{2} \sum_{x \in D} (f(x) - \hat{f}(x))^2 \cdot K(d(x_q, x))$$

  3. Combine 1 and 2

$$E_3(x_q) \equiv \frac{1}{2} \sum_{x \in k \text{ nearest neighbors}} (f(x) - \hat{f}(x))^2 \cdot K(d(x_q, x))$$

Locally weighted linear regression

- choice of the error criterion
  - E_2 is the most esthetically criterion, because it allows every training example to have impact on the classification of x_q
  - however, computational effort grows with the number of training examples
  - E_3 is a good approximation to E_2 with constant effort

$$\Delta w_j = \eta \sum_{x \in k \text{ nearest neighbors}} K(d(x_q, x))(f(x) - \hat{f}(x))a_j$$

- Remarks on locally weighted linear regression:
  - in most cases, constant, linear or quadratic functions are used
  - costs for fitting more complex functions are prohibitively high
  - simple approximations are good enough over a sufficiently small subregion of X

Algorithm

MACHINE LEARNING

1. Read the Given data Sample to **X** and the curve (linear or non linear) to **Y**
2. Set the value for Smoothening parameter or Free parameter say **τ**
3. Set the bias /Point of interest set **X0** which is a subset of **X**
4. Determine the weight matrix using :

$$w(x, x_o) = e^{-\frac{(x-x_o)^2}{2\tau^2}}$$

5. Determine the value of model term parameter β using :

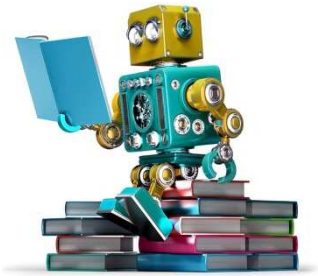
$$\hat{\beta}(x_o) = (X^T W X)^{-1} X^T W y$$

6. Prediction = $x_0 * \beta$

Module 5 - Outline

Chapter 8: Instance Based Learning

1. Introduction
2. K-nearest neighbor Learning
3. Locally Weighted regression
4. **Radial basis functions**
5. Case based reasoning
6. Summary



Radial basis function

One of the approach to function approximation

that is closely related to

distance-weighted regression and

artificial neural networks

is learning with radial basis functions

Radial basis function (for regression)

MACHINE LEARNING

- closely related to distance-weighted regression and to ANNs
- learned hypotheses have the form

$$\hat{f}(x) = w_0 + \sum_{u=1}^{k} w_u \cdot K_u(d(x_u, x))$$

where

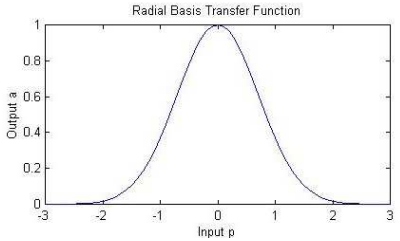
- each x_u is an instance from X and
- $K_u(d(x_u, x))$ decreases as $d(x_u, x)$ increases and
- k is a user-provided constant

- though $\hat{f}(x)$ is a global approximation to $f(x)$, the contribution of each of the K_u terms is localized to a region nearby the point x_u

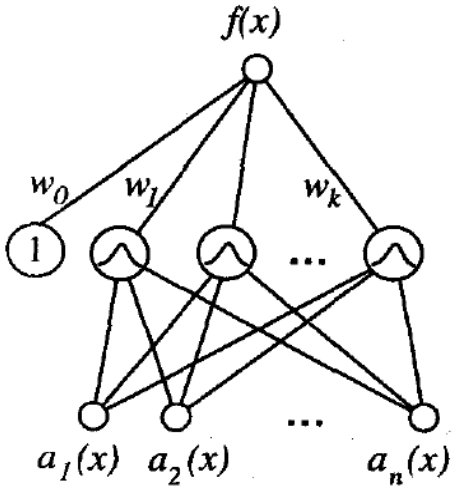
Radial basis function

- it is common to choose each function $K_u(d(x_u, x))$ to be a Gaussian function centered at x_u with some variance σ^2

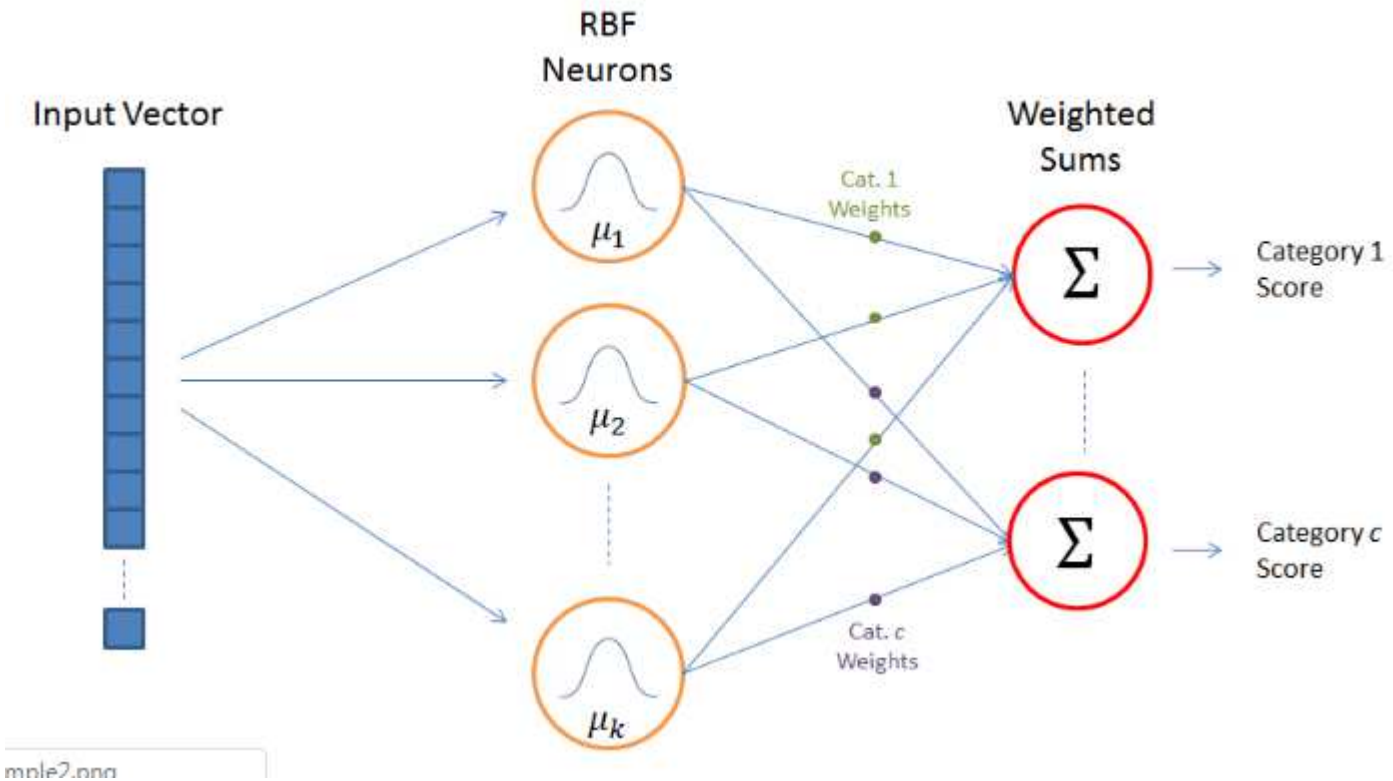
$$K_u(d(x_u, x)) = e^{\frac{1}{2\sigma_u^2} d^2(x_u, x)}$$



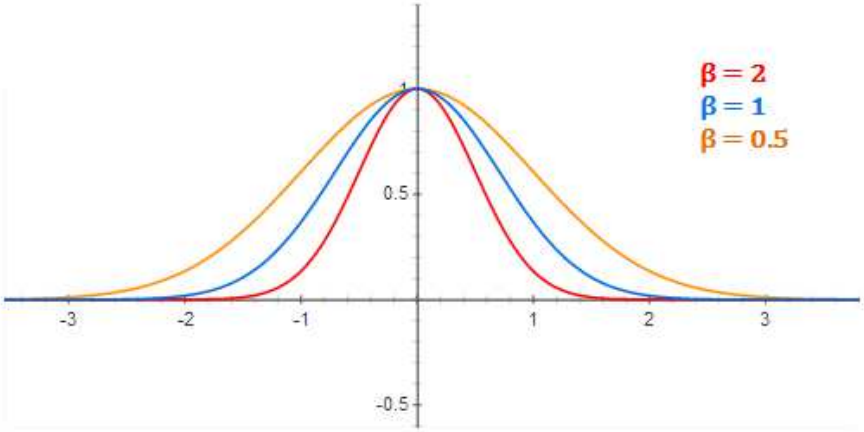
- the function of $\hat{f}(x)$ can be viewed as describing a two-layer network where the first layer of units computes the various $K_u(d(x_u, x))$ values and the second layer a linear combination of the results



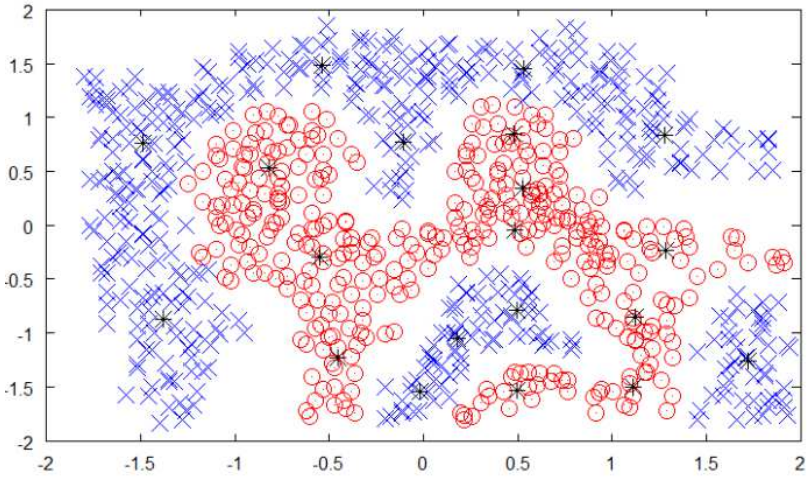
RBF NN

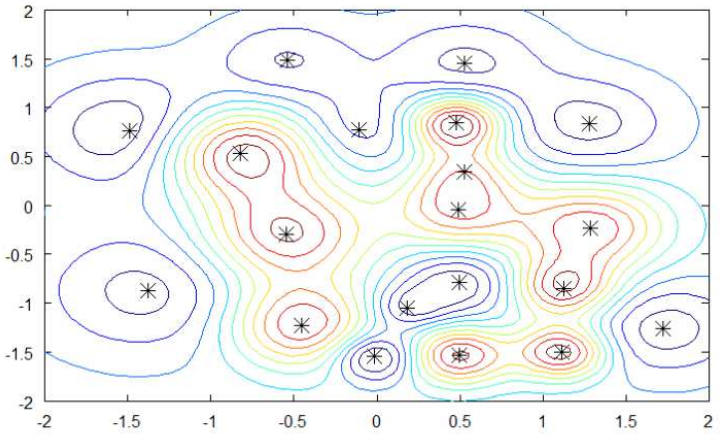

RBF
Neurons
Input Vector
Weighted
Sums
Cat. 1
Weights
μ_1
μ_2
μ_k
Σ
Σ
Category 1
Score
Category c
Score
Cat. c
Weights

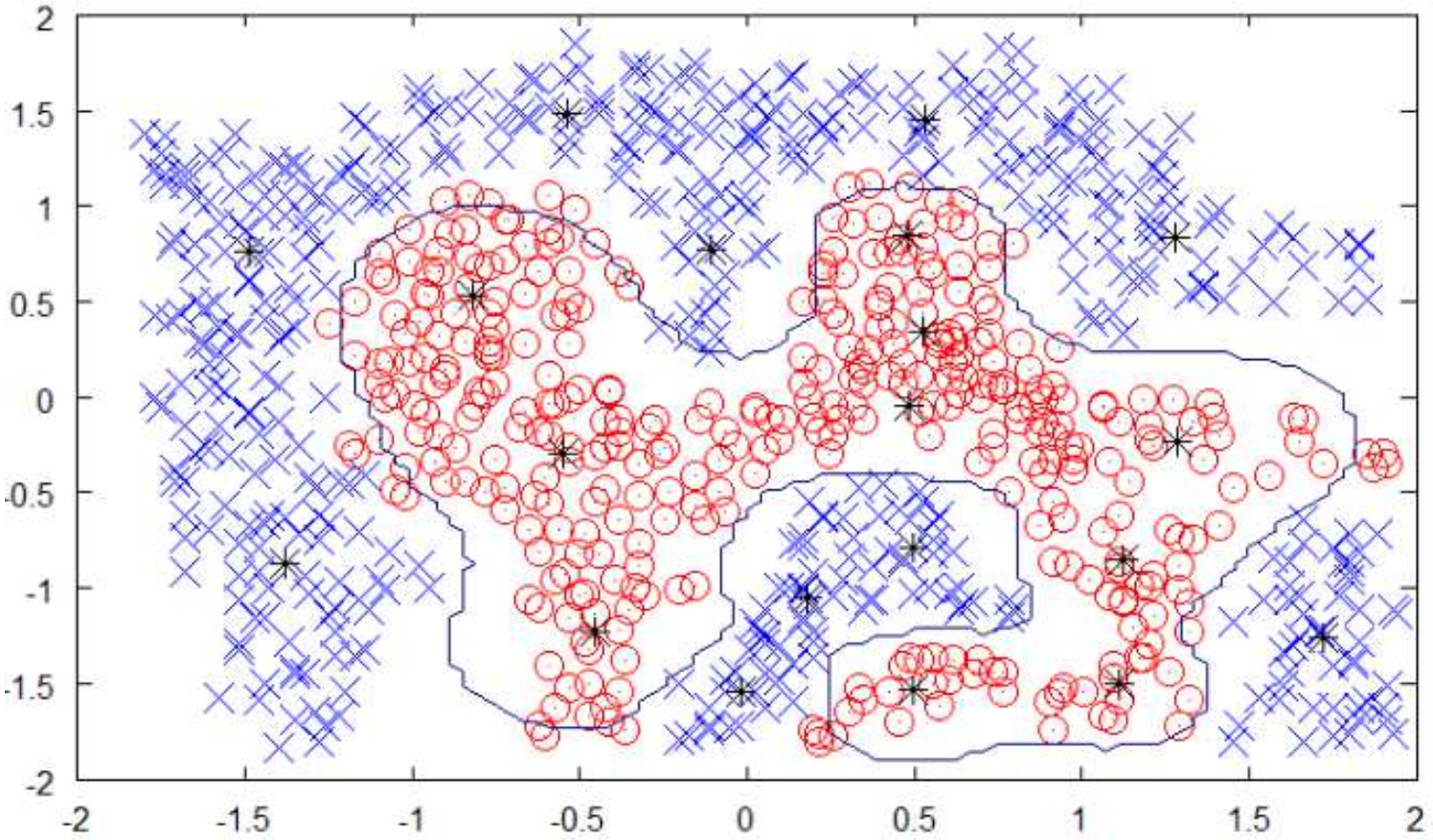
$$\varphi(x) = e^{-\beta\|x-\mu\|^2}$$



RBF Neuron activation for different values of beta



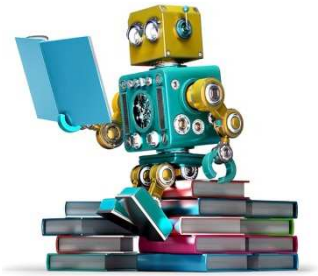



2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
-2
-1.5
-1
-0.5
0
0.5
1
1.5
2

Module 5 - Outline

Chapter 8: Instance Based Learning

1. Introduction
2. K-nearest neighbor Learning
3. Locally Weighted regression
4. Radial basis functions
5. **Case based reasoning**
6. Summary



Case Based Reasoning

- Instance-based methods such as k-NN, locally weighted regression share three key properties.
  1. They are lazy learning methods

     They defer the decision of how to generalize beyond the training data until a new query instance is observed.
  2. They classify new query instances by analyzing similar instances while ignoring instances that are very different from the query.
  3. Third, they represent instances as real-valued points in an n-dimensional Euclidean space.
- Case-based reasoning (CBR) is a learning paradigm based on the first two of these principles, but not the third.

Case Based Reasoning

- In CBR, instances are typically represented using more rich symbolic descriptions, and the methods used to retrieve similar instances are correspondingly more elaborate.
  - CBR has been applied to problems such as conceptual design of mechanical devices based on a stored library of previous designs (Sycara et al. 1992),
  - reasoning about new legal cases based on previous rulings (Ashley 1990),
  - solving planning and scheduling problems by reusing and combining portions of previous solutions to similar problems (Veloso 1992).

Case Study

- The CADET system (Sycara et al. 1992)
  - employs case based reasoning to assist in the conceptual design of simple mechanical devices such as water faucets.
  - It uses a library containing approximately 75 previous designs and
  - design fragments to suggest conceptual designs to meet the specifications of new design problems.
  - Complete Case study - Self study

Summary

- instance-based learning simply stores examples and postpones generalization until a new instance is encountered
- able to learn discrete- and continuous-valued conepts
- noise in the data is allowed (smoothed out by weighting distances)
- **Inductive Bias of k-nearest neighbors**: classification of an instance is similar to the classification of other instances nearby in the Euclidean Distance
- Locally Weighted Regression forms a local approximation of the target function