



BITS Pilani
Pilani Campus

Machine Learning

AIML CLZG565

Instance-based Learning

Raja vadhana P
Assistant Professor,
BITS - CSIS

Disclaimer and Acknowledgement



- The content for these slides has been obtained from books and various other source on the Internet
- I here by acknowledge all the contributors for their material and inputs.
- I have provided source information wherever necessary
- I have added and modified the content to suit the requirements of the course

Source: Slides of Prof. Chetana, Prof.Seetha, Prof.Sugata, Prof.Monali, Prof. Raja vadhana , Prof.Anita from BITS Pilani , CS109 and CS229 stanford lecture notes and many others who made their course materials freely available online.

Course Plan

M1	Introduction & Mathematical Preliminaries
M2	Machine Learning Workflow
M3	Linear Models for Regression
M4	Linear Models for Classification
M5	Decision Tree
M6	Instance Based Learning
M7	Support Vector Machine
M8	Bayesian Learning
M9	Ensemble Learning
M10	Unsupervised Learning
M11	Machine Learning Model Evaluation/Comparison

Instance Based Learning

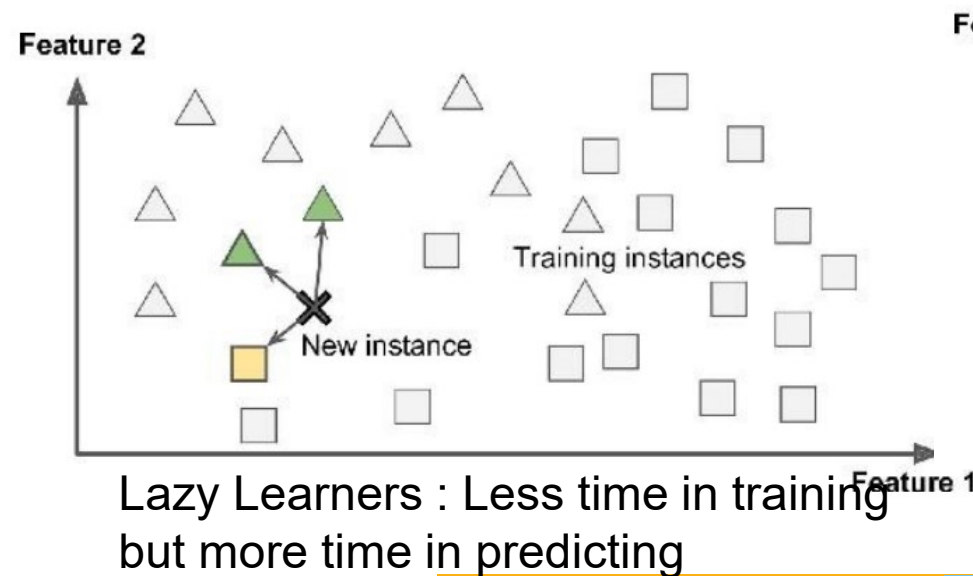
Introduction to Machine Learning



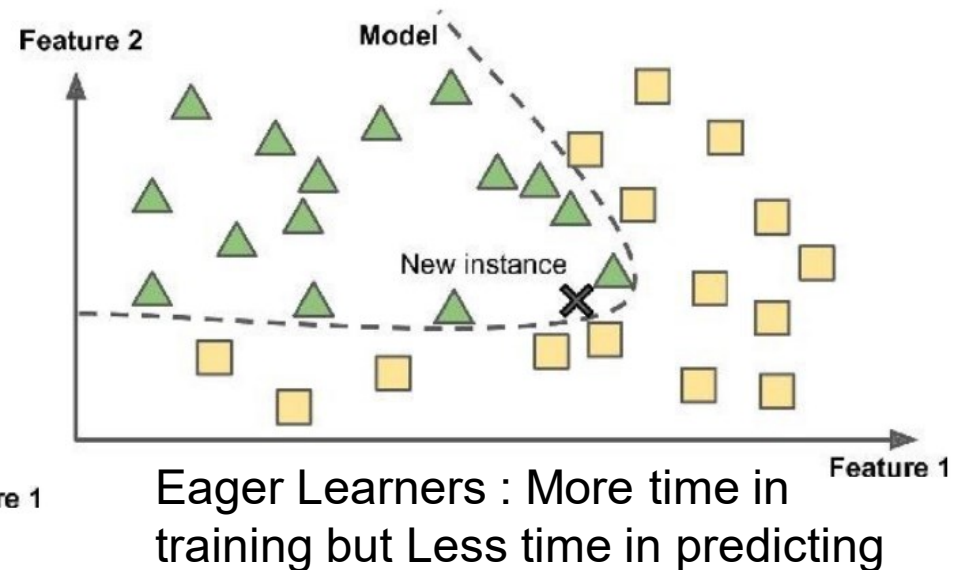
Types: Based on how & when the training data is used

- Instance Based Learning: Compare new data points to known data points
- Model Based learning : Detect patterns in the training data and build a predictive model

Instance
Based



Model
Based



Instance Based Learning

Model-based learning techniques

Use the input data

$$\begin{bmatrix} x_{1,0} & x_{1,1} & \dots & x_{1,n} \\ x_{2,0} & x_{2,1} & \dots & x_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ x_{m,0} & x_{m,1} & \dots & x_{m,n} \end{bmatrix} \text{ and } \begin{bmatrix} y_1 \\ y_2 \\ \dots \\ y_m \end{bmatrix}$$



To learn a set of parameters

$$\begin{bmatrix} \theta_0 & \theta_1 & \dots & \theta_n \end{bmatrix}$$



Which yield a **generalized** function

$$f(x) = \theta_0 x_0 + \theta_1 x_1 + \theta_2 x_2 + \dots + \theta_n x_n$$



Capable of predicting values or classes on new input data

$$f(x_i) = 39$$

$$f(x_j) = 1$$

Instance-based learning techniques

Store the input data

$$\begin{bmatrix} x_{1,0} & x_{1,1} & \dots & x_{1,n} \\ x_{2,0} & x_{2,1} & \dots & x_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ x_{m,0} & x_{m,1} & \dots & x_{m,n} \end{bmatrix} \text{ and } \begin{bmatrix} y_1 \\ y_2 \\ \dots \\ y_m \end{bmatrix}$$



When asked to predict a new value (a query)

$$y_i = ?$$



Search for similar data points previously stored

$$\begin{bmatrix} x_{4,1} & x_{4,2} & \dots & x_{4,n} \\ x_{9,1} & x_{9,1} & \dots & x_{9,n} \\ x_{15,1} & x_{15,1} & \dots & x_{15,n} \end{bmatrix} \text{ and } \begin{bmatrix} y_4 \\ y_9 \\ y_{15} \end{bmatrix}$$



And use them to generate your prediction

$$y_i = \frac{y_4 + y_9 + y_{15}}{3}$$

Source Credit : <https://www.jeremyjordan.me/k-nearest-neighbors/>

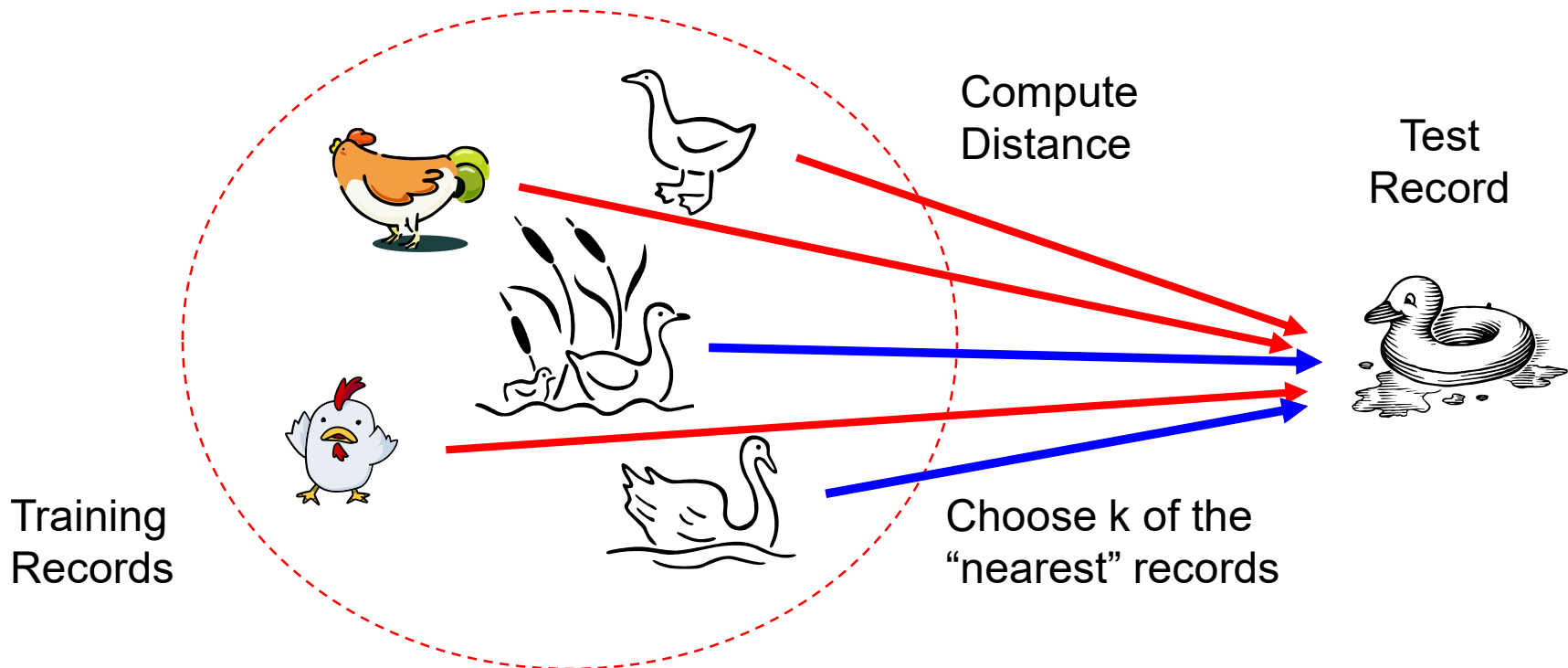
Instance Based Learning



Nearest Neighbor Approach

Basic idea:

- If it walks like a duck, quacks like a duck, then it's probably a duck



Instance Based Learning

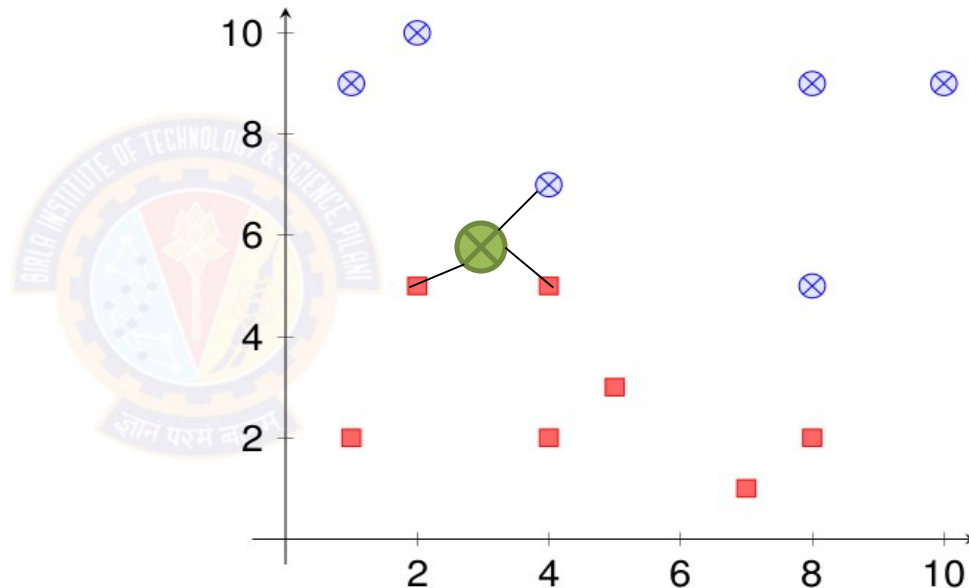


Nearest Neighbor Approach : Notations

K-nearest neighbor:

- Given x_q , take **vote** among its k nearest neighbors (if **discrete-valued target function**)
- Take **mean(or median)** of f values of k nearest neighbors (if **real-valued**) $f^*(x_q) = \sum_{i=1}^k f(x_i)/k$

x_1	x_2	y
1	9	white
10	9	white
4	7	white
4	5	pink
5	3	pink
8	9	white
4	2	pink
2	5	pink
7	1	pink
2	10	white
8	5	white
1	2	pink
8	2	pink



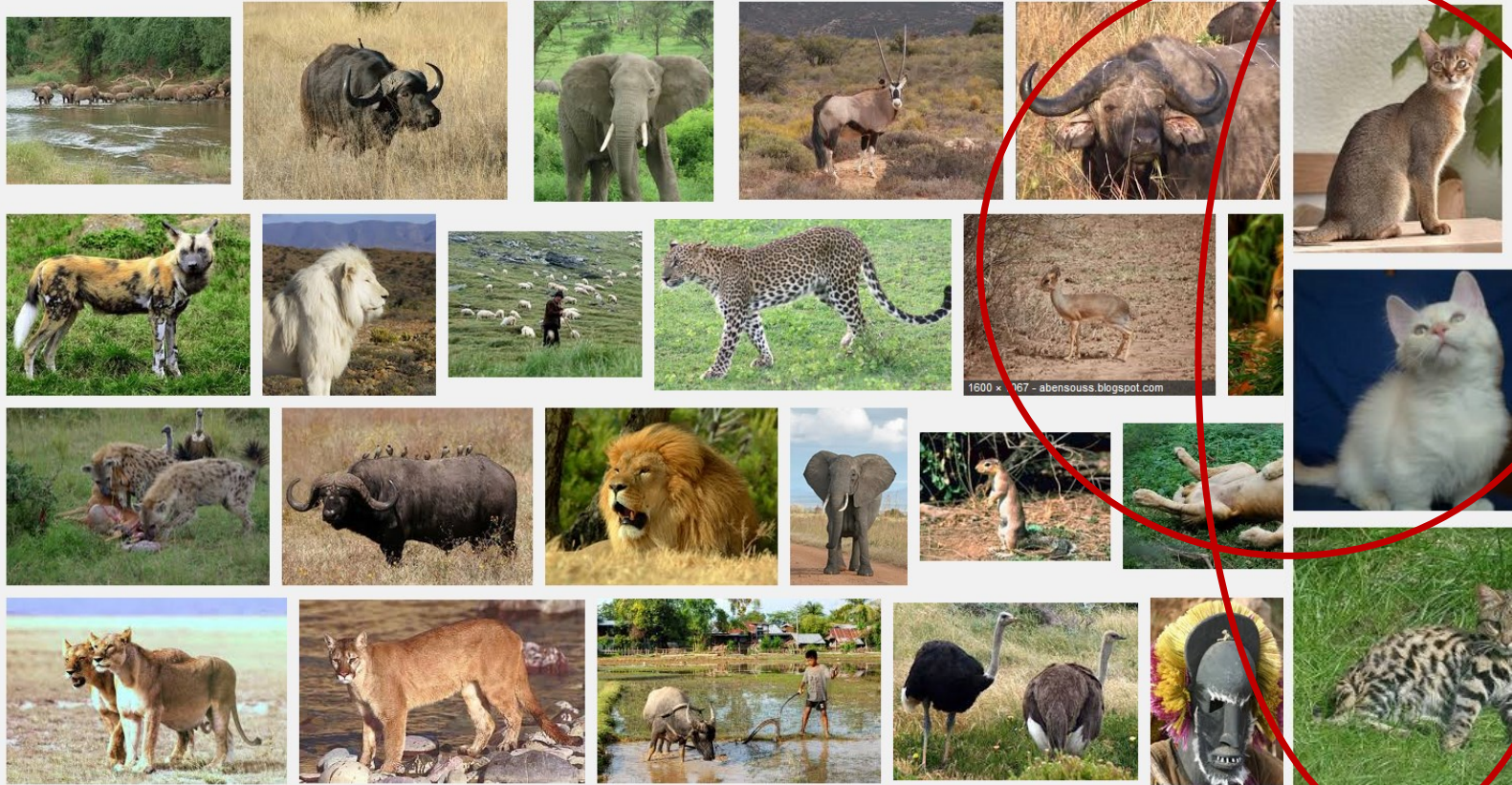
Interpretation : Assume that the Instances represented as points in a Euclidean space

innovate **achieve** **lead**

2 votes for cat,
1 each for Buffalo, Deer, Lion

Cat wins...
k = 5 Neighborhood Region

Return the most Frequent Label $k = 3$ votes for "cat"

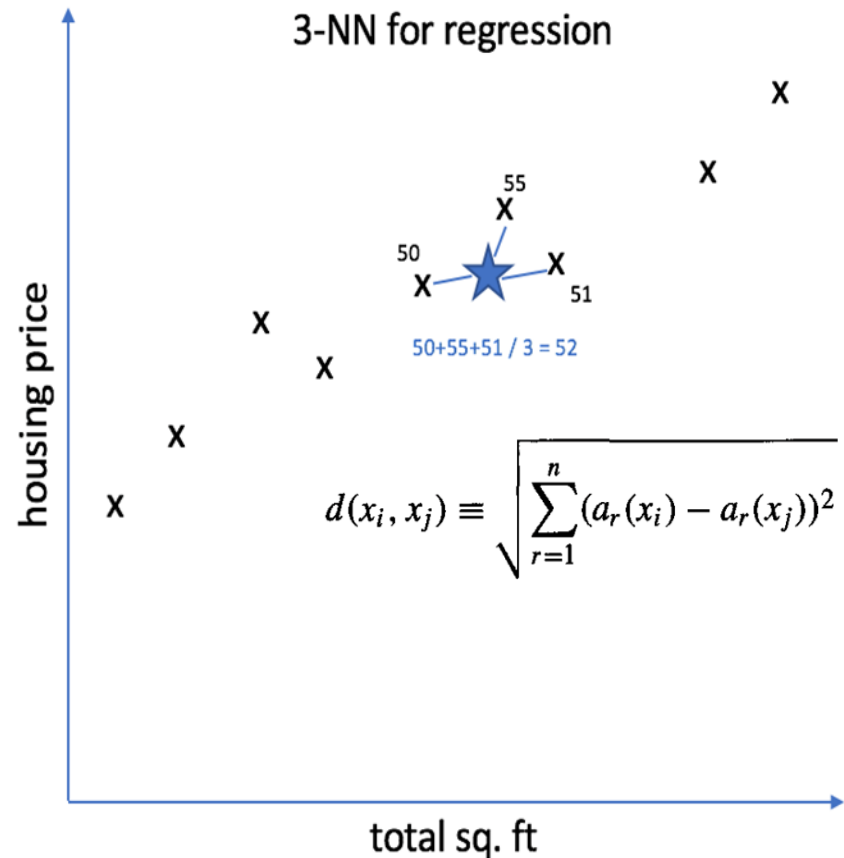


Instance Based Learning



K-Nearest Neighbor Regression: Example

Size in feet ² (x)	Price (\$) in 1000's (y)
2104	460
1416	232
1534	315
1852	178
...	...
...	...



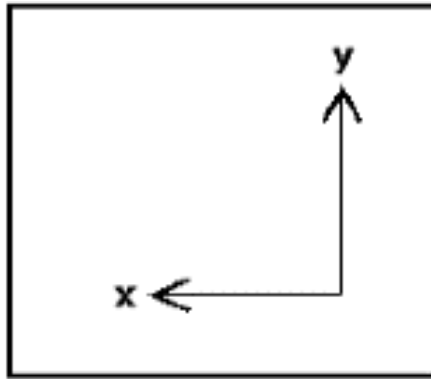
Source Credit : <https://www.jeremyjordan.me/k-nearest-neighbors/>

Measures of Dis/similarity

Distances

- Manhattan Distance

$$|X_1 - X_2| + |Y_1 - Y_2|$$

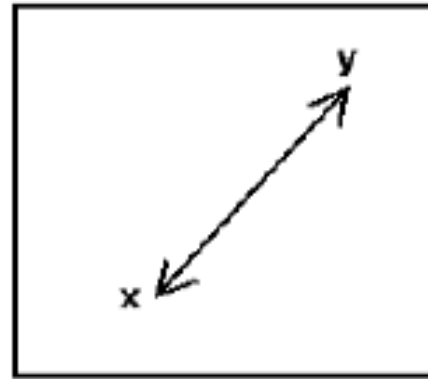


Manhattan

- works if the points are arranged in the form of a grid
- if the input variables are not similar in type (such as age, gender, height, etc.)

- Euclidean Distance

$$\sqrt{(y_1 - y_2)^2 + (x_1 - x_2)^2 + \dots}$$



Euclidean

- Euclidean is commonly used on dense, continuous variables.
- There every dimension matters, and a 20 dimensional space can be challenging**
- if the input variables are of similar in type (such as width, height, depth etc.)

Measures of Dis/similarity

Distances : Special Cases of Minkowski

- $h = 1$: **Manhattan** (city block, L_1 norm) distance
 - E.g., the Hamming distance: the number of bits that are different between two binary vectors

$$d(i, j) = |x_{i1} - x_{j1}| + |x_{i2} - x_{j2}| + \dots + |x_{ip} - x_{jp}|$$

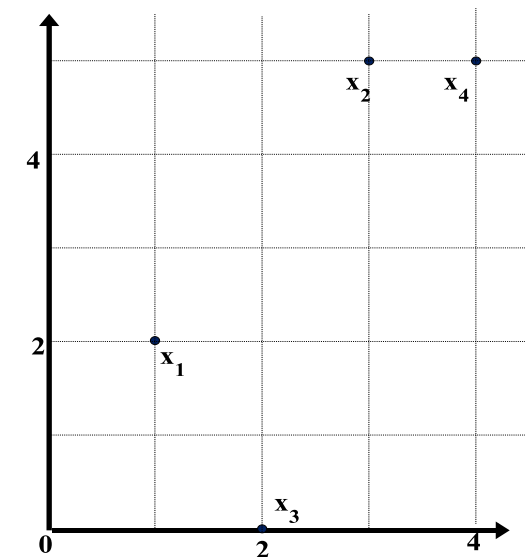
- $h = 2$: (L_2 norm) **Euclidean** distance

$$d(i, j) = \sqrt{(|x_{i1} - x_{j1}|^2 + |x_{i2} - x_{j2}|^2 + \dots + |x_{ip} - x_{jp}|^2)}$$

- $h \rightarrow \infty$. “**supremum**” (L_{\max} norm, L_{∞} norm) distance.
 - This is the maximum difference between any component (attribute) of the vectors

$$d(i, j) = \lim_{h \rightarrow \infty} \left(\sum_{f=1}^p |x_{if} - x_{jf}|^h \right)^{\frac{1}{h}} = \max_f |x_{if} - x_{jf}|$$

point	attribute 1	attribute 2
x1	1	2
x2	3	5
x3	2	0
x4	4	5



Measures of Dis/similarity

Distances : Special Cases of Minkowski

(Dissimilarity Matrices)

Manhattan (L_1)

L	x1	x2	x3	x4
x1	0			
x2	5	0		
x3	3	6	0	
x4	6	1	7	0

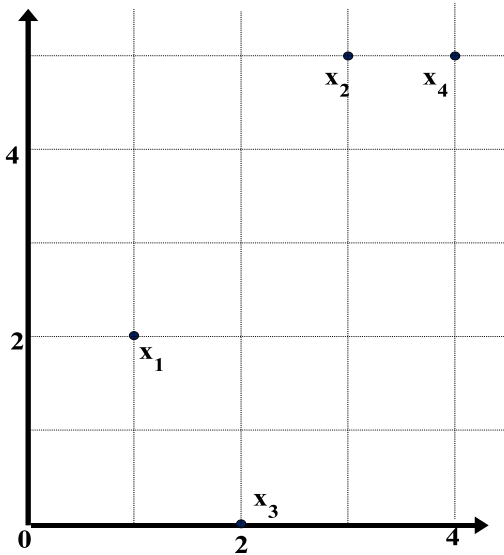
Euclidean (L_2)

L2	x1	x2	x3	x4
x1	0			
x2	3.61	0		
x3	2.24	5.1	0	
x4	4.24	1	5.39	0

Supremum

L_∞	x1	x2	x3	x4
x1	0			
x2	3	0		
x3	2	5	0	
x4	3	1	5	0

point	attribute 1	attribute 2
x1	1	2
x2	3	5
x3	2	0
x4	4	5



Measures of Dis/similarity

Distances : Attributes of Ordinal Type

- A database may contain all attribute types
 - Nominal, symmetric binary, asymmetric binary, numeric, ordinal

An ordinal variable can be discrete or continuous

Order is important, e.g., rank

Can be treated like interval-scaled

replace x_{if} by their rank

map the range of each variable onto $[0, 1]$ by replacing i -th object in the f -th variable by

$$z_{if} = \frac{r_{if} - 1}{M_f - 1}$$

Treat Z as numeric attribute

Measures of Dis/similarity

Distances : Attributes of Mixed Type – Gower's Distance

- A database may contain all attribute types
 - Nominal, symmetric binary, asymmetric binary, numeric, ordinal

$$d(i, j) = \frac{\sum_{c=1}^n \omega_c \delta_{ij}^{(c)} d_{ij}^{(c)}}{\sum_{c=1}^n \omega_c \delta_{ij}^{(c)}}$$

$d(i, j)$ = dissimilarity between row i and row j

c = the cth column

n = number of columns in the dataset

ω_c = weight of cth column = $\frac{1}{\text{nrows in dataset}}$

$\delta_{ij}^c = \begin{cases} 0 & \text{if column c is missing in row i or j} \\ 0 & \text{if column c is asymmetric binary and both} \\ & \text{values in row i and j are 0} \\ 1 & \text{otherwise} \end{cases}$

$d_{ij}^c(\text{categorical}) = \begin{cases} 0 & \text{if i and j are equal in column c} \\ 1 & \text{otherwise} \end{cases}$

$d_{ij}^c(\text{continuous/ordinal}) = \frac{|\text{row i in column c} - \text{row j in column c}|}{\max(\text{column c}) - \min(\text{column c})}$

Measures of Dis/similarity

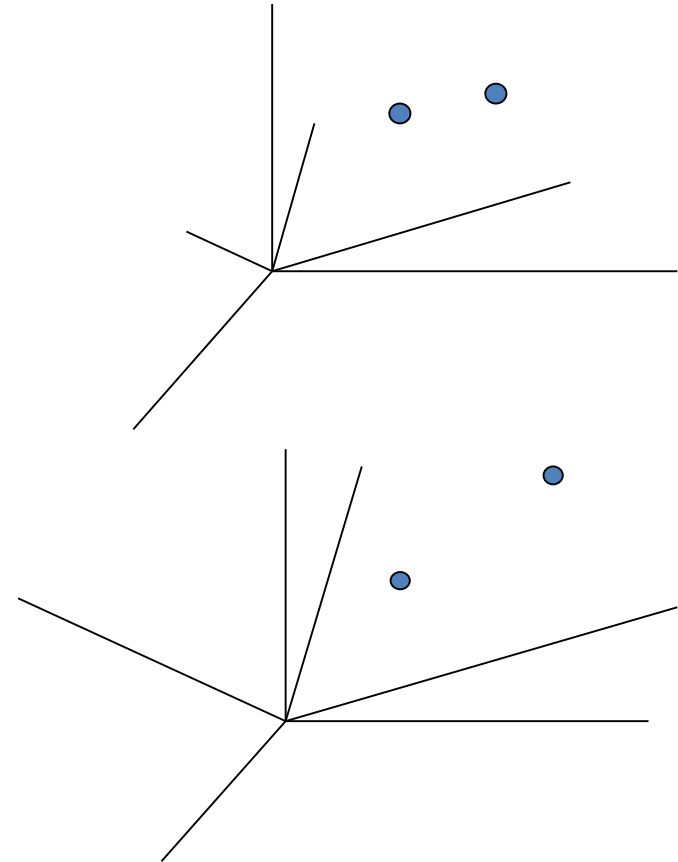
Curse of Dimensionality

- It uses all attributes to classify instances
- If the target concept depends on only a few of the many available attributes, then the instances that are truly most "similar" may well be a large distance apart.
- **Curse of dimensionality**: nearest neighbor is easily misled when instance space is high-dimensional

Solution 1: weigh the attributes differently (use cross-validation to determine the weights)

Solution 2: eliminate the least relevant attributes (again, use cross-validation to determine which attributes to eliminate)

Stretching the axes in the Euclidean space, shortening the axes that correspond to less relevant attributes, and lengthening the axes that correspond to more relevant attributes



Instance Based Learning



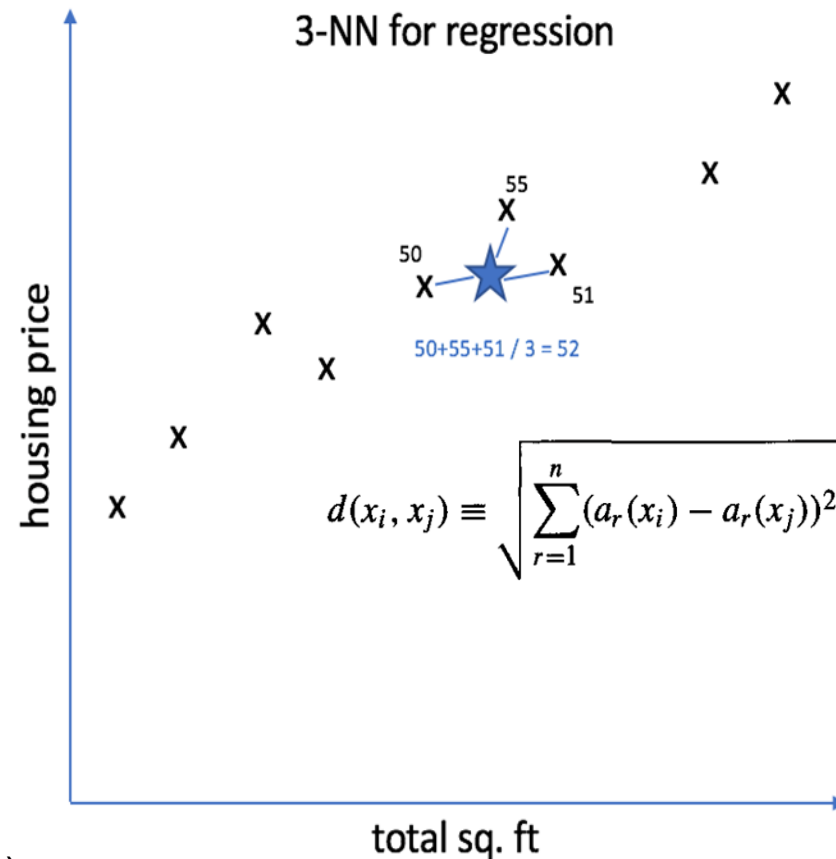
K-Nearest Neighbor Regression: Example

Size in feet ² (x_1)	No.of.Bed rooms (x_2)	Price (\$) in 1000's (y)
2104	4	460
1416	2	232
1534	3	315
1852	2	178
...

min value of x_1

Min-Max scaler/Normalization

Feature scaling of features x_i consists of rescaling the range of features to scale the range in $[0, 1]$ or $[-1, 1]$ (Do not apply to x_0)




Source Credit : <https://www.jeremyjordan.me/k-nearest-neighbors/>

Instance Based Learning



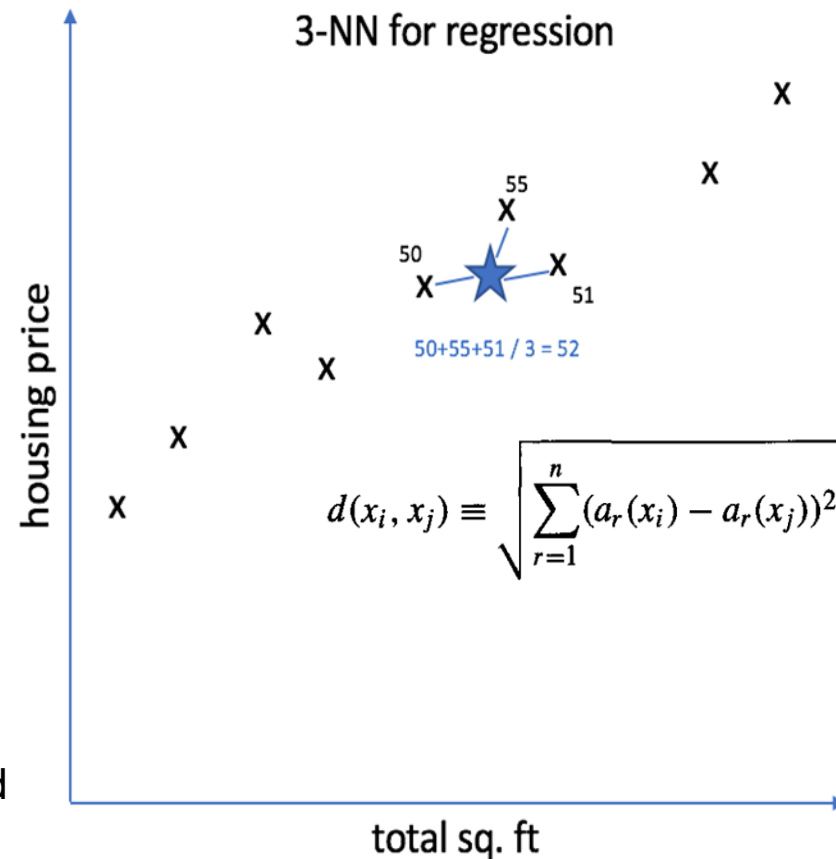
K-Nearest Neighbor Regression: Example

Size in feet ² (x_1)	No.of.Bed rooms (x_2)	Price (\$) in 1000's (y)
2104	4	460
1416	2	232
1534	3	315
1852	2	178
...


 ← Average value of x_1

$$Z = \frac{x - \mu}{\sigma}$$

The Standard Scaler assumes data is normally distributed within each feature and scales them such that the distribution centered around 0, with a standard deviation of 1



Source Credit : <https://www.jeremyjordan.me/k-nearest-neighbors/>

Instance Based Learning



K-Nearest Neighbor : Algorithm

Approximating a discrete-valued function $f : \mathbb{R}^n \rightarrow V$.

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 $x_1 \dots x_k$ denote the k instances from *training_examples* that are nearest to x_q
 - Return

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

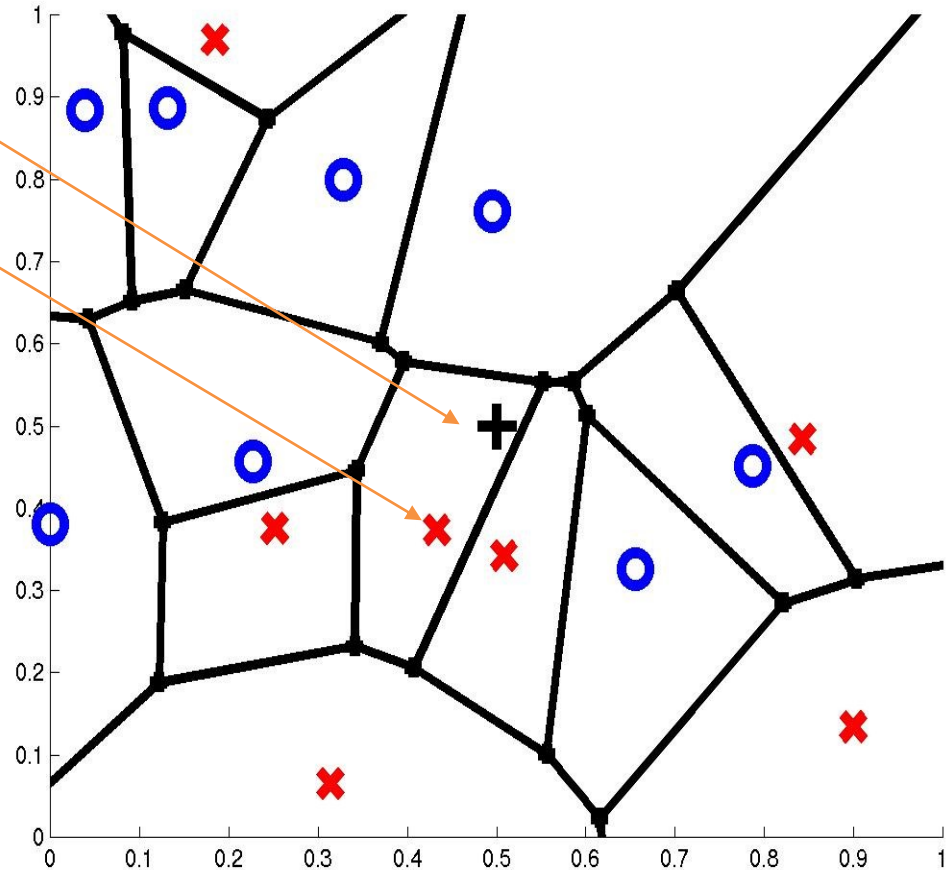
where $\delta(a, b) = 1$ if $a = b$ and where $\delta(a, b) = 0$ otherwise.

Approximate a real-valued target function $f : \mathbb{R}^n \rightarrow \mathbb{R}$

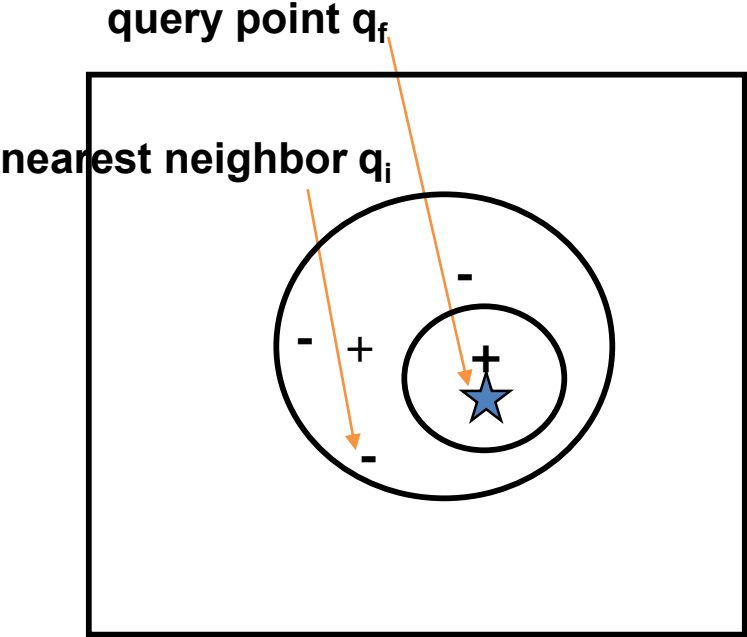
$$\hat{f}(x_q) \leftarrow \frac{\sum_{i=1}^k f(x_i)}{k}$$

Voronoi Diagram

- Nearest neighbour approach induces a **Voronoi tessellation**/partition of the input space (all test points falling in a cell will get the label of the training input in that cell)
- For any sample, the nearest sample is determined by the closest Voronoi cell edge

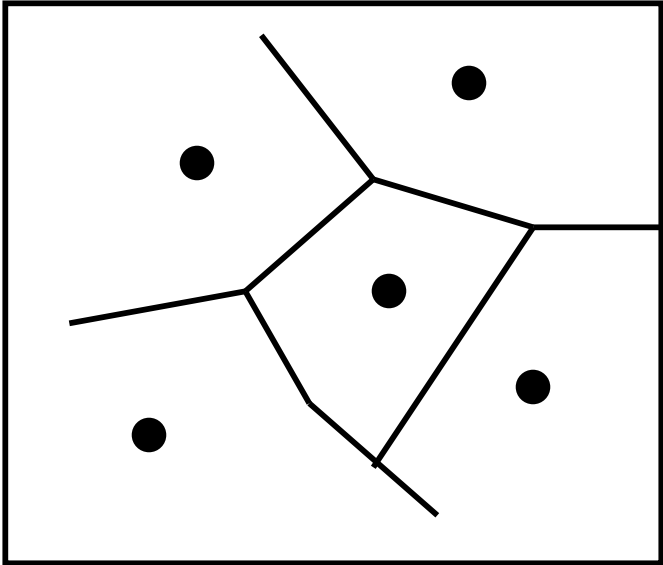


Voronoi Diagram



★ : query, x_q

1-NN: +
5-NN: -



Decision Surface
for 1-NN

Instance Based Learning



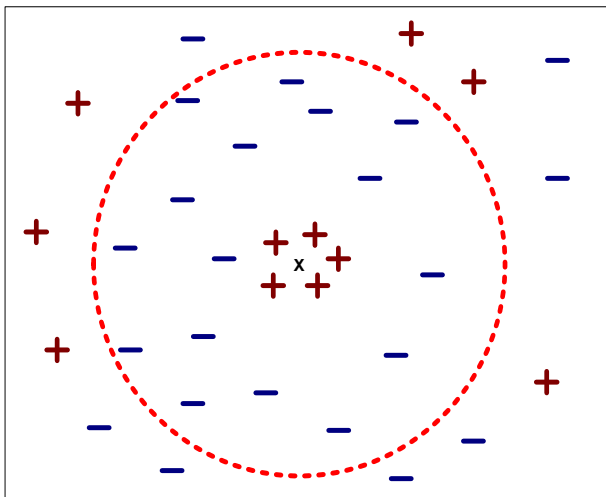
K-Nearest Neighbor : Hyper parameter “K”

Choosing the value of k:

Rule of thumb:

$$K = \text{sqrt}(N)$$

N: number of training points



K-Nearest Neighbor : Hyper parameter “K”

Choosing the value of k:

- If k is too small, sensitive to noise points
- If k is too large, neighborhood may include points from other classes

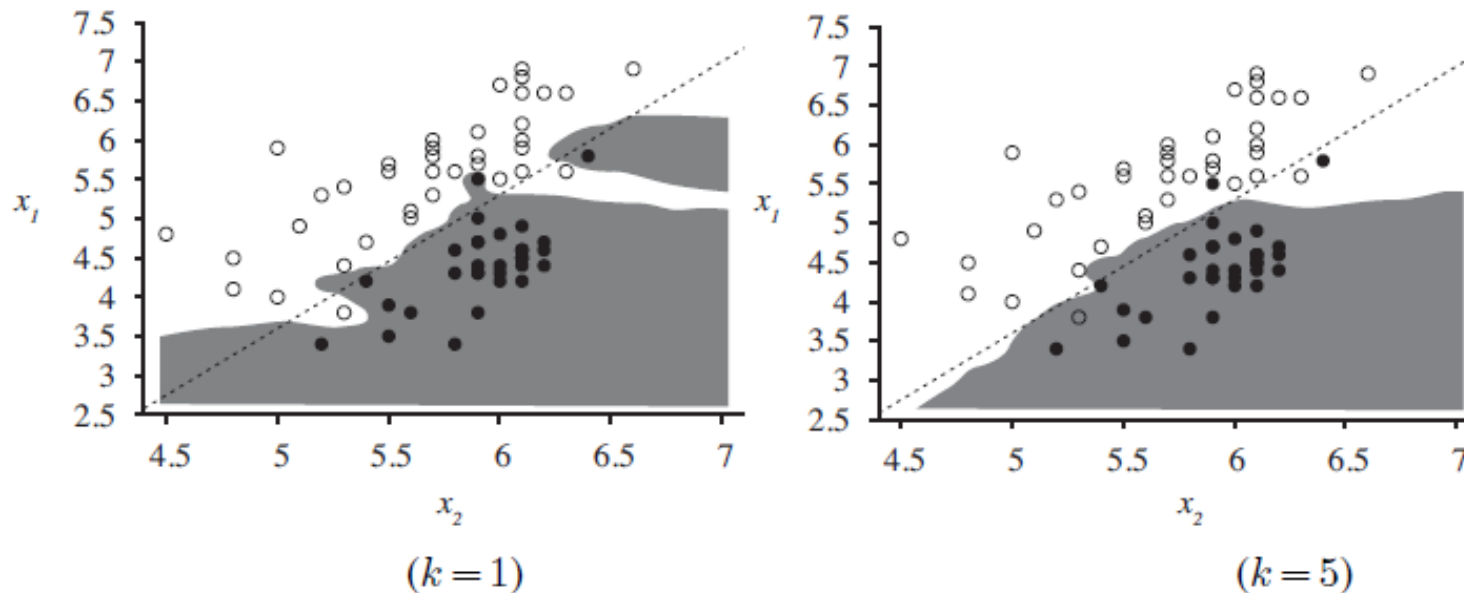
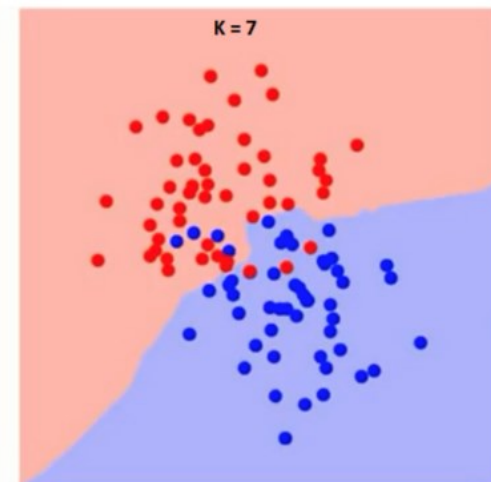
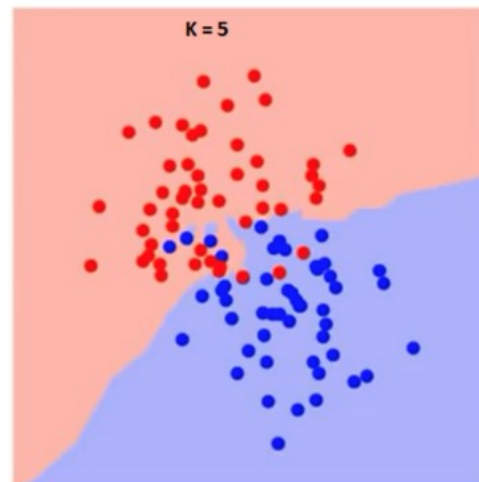
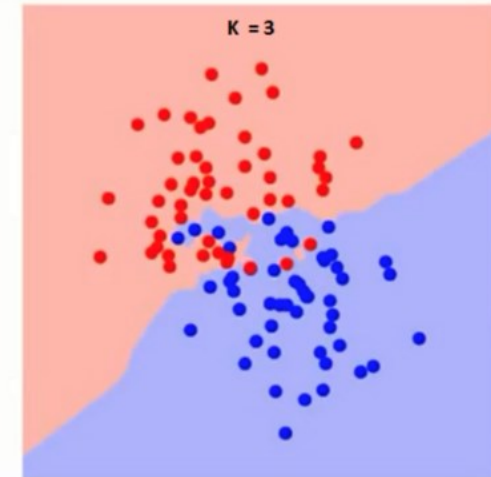
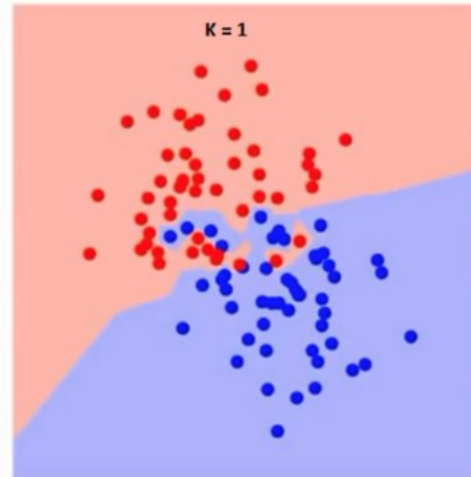


Figure 18.26 (a) A k -nearest-neighbor model showing the extent of the explosion class for the data in Figure 18.15, with $k = 1$. Overfitting is apparent. (b) With $k = 5$, the overfitting problem goes away for this data set.

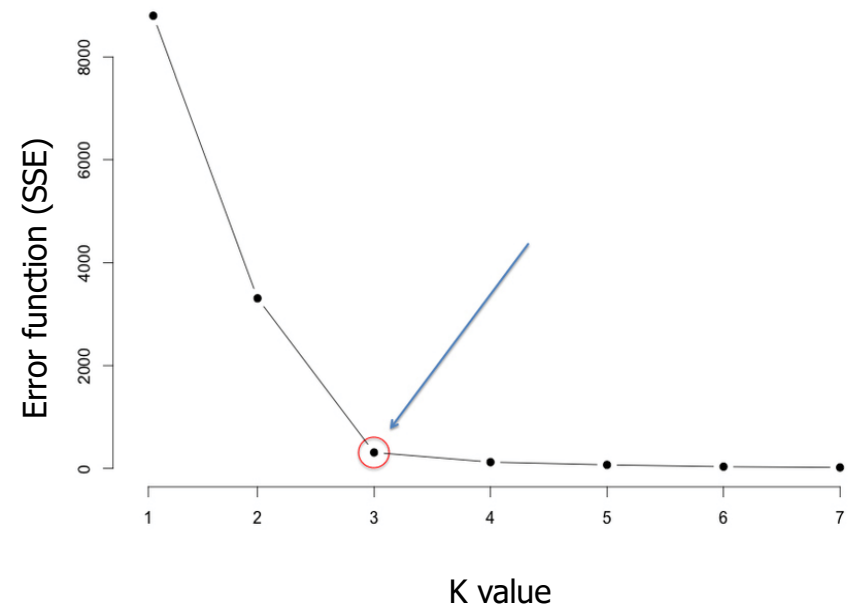
K-Nearest Neighbor : Hyper parameter “K” : Elbow Method

- Compute sum of squares error (SSE) or any other error function for varying values of K (1 to a reasonable X) and plot against K
- In the plot, the elbow (see pic) gives the value of K beyond which the error function plot almost flattens
- As K approaches the total number of instances in the set, error function drops down to ‘0’



K-Nearest Neighbor : Hyper parameter “K” : Elbow Method

- Compute sum of squares error (SSE) or any other error function for varying values of K (1 to a reasonable X) and plot against K
- In the plot, the elbow (see pic) gives the value of K beyond which the error function plot almost flattens
- As K approaches the total number of instances in the set, error function drops down to ‘0’



Instance Based Learning

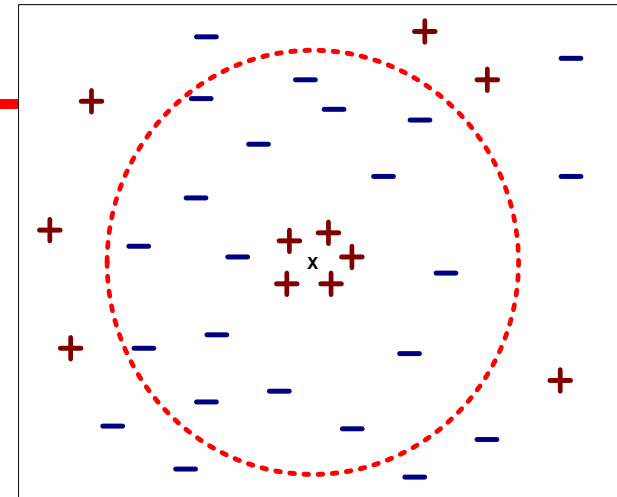


K-Nearest Neighbor : Variation

- Locally Weighted K-NN algorithm or Distance Weighted K-NN algorithm

contribution of each of the k nearest neighbors is weighted accorded to their distance to x_q

- discrete-valued target functions



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

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:

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

Kernel Functions

$$w_i = K(d(x_q, x_i))$$

$$K(d(x_q, x_i)) = 1 / d(x_q, x_i)^2$$

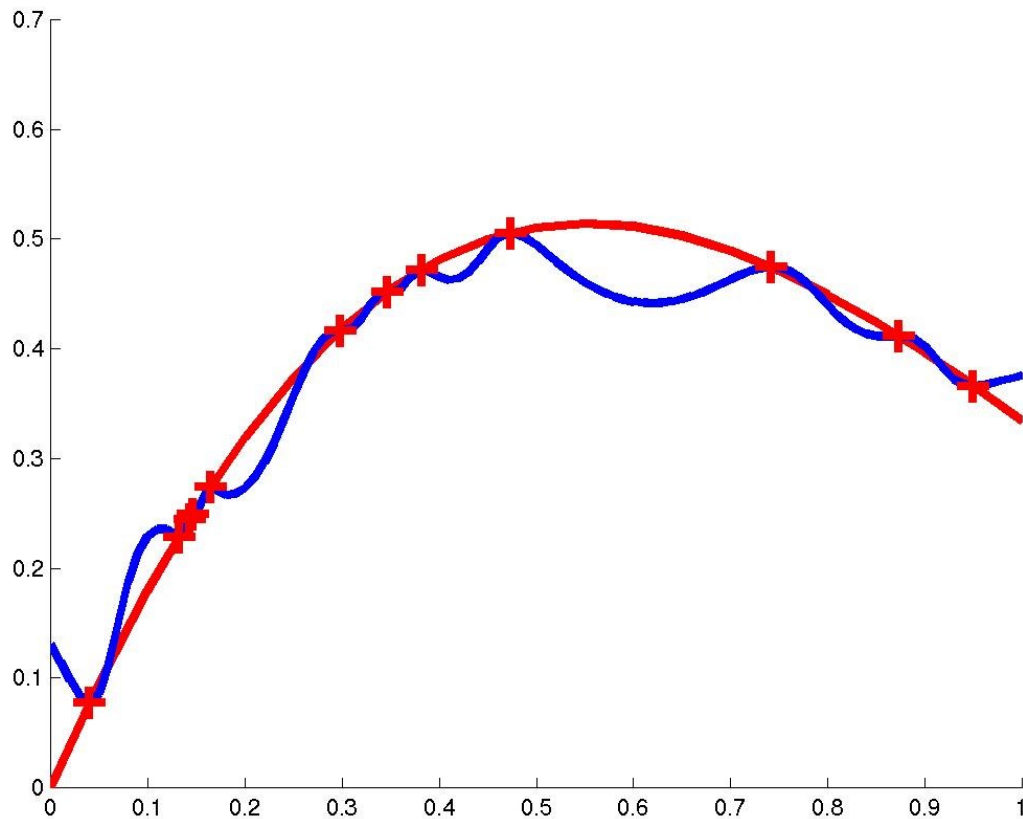
$$K(d(x_q, x_i)) = 1 / (d_0 + d(x_q, x_i))^2$$

$$K(d(x_q, x_i)) = \exp(-(d(x_q, x_i) / \sigma_0)^2)$$

$d(x_q, x_i)$ is the distance between x_q and x_i

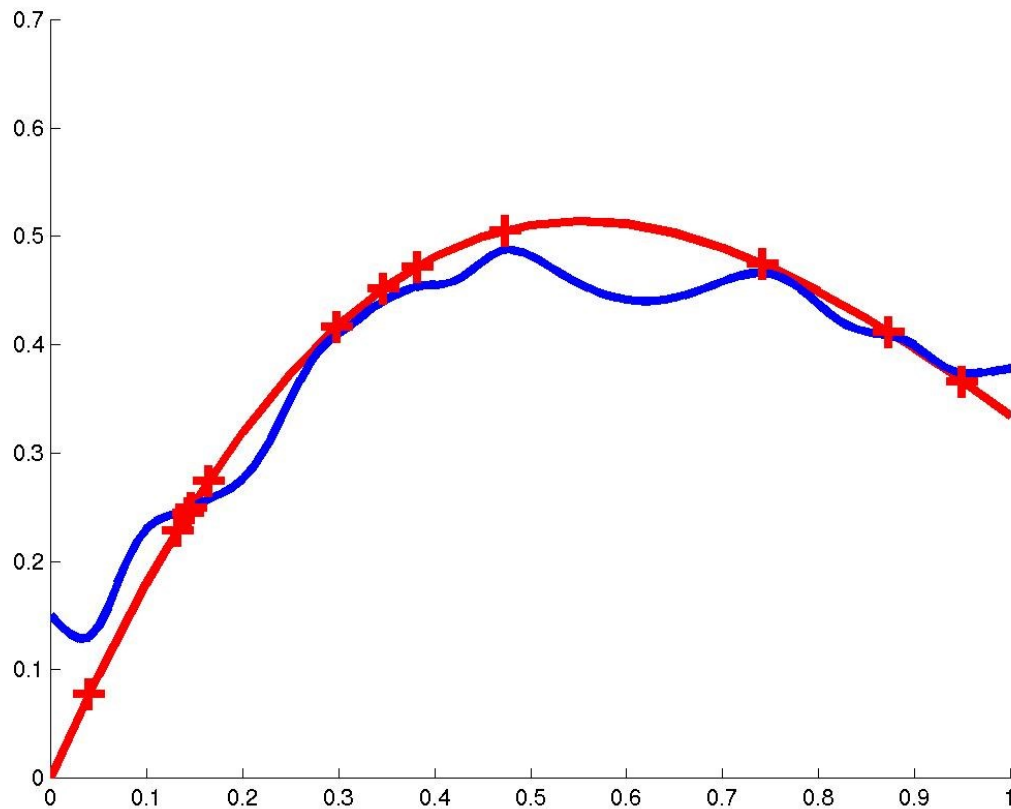
Distance Weighted NN

$$K(d(x_q, x_i)) = 1 / d(x_q, x_i)^2$$



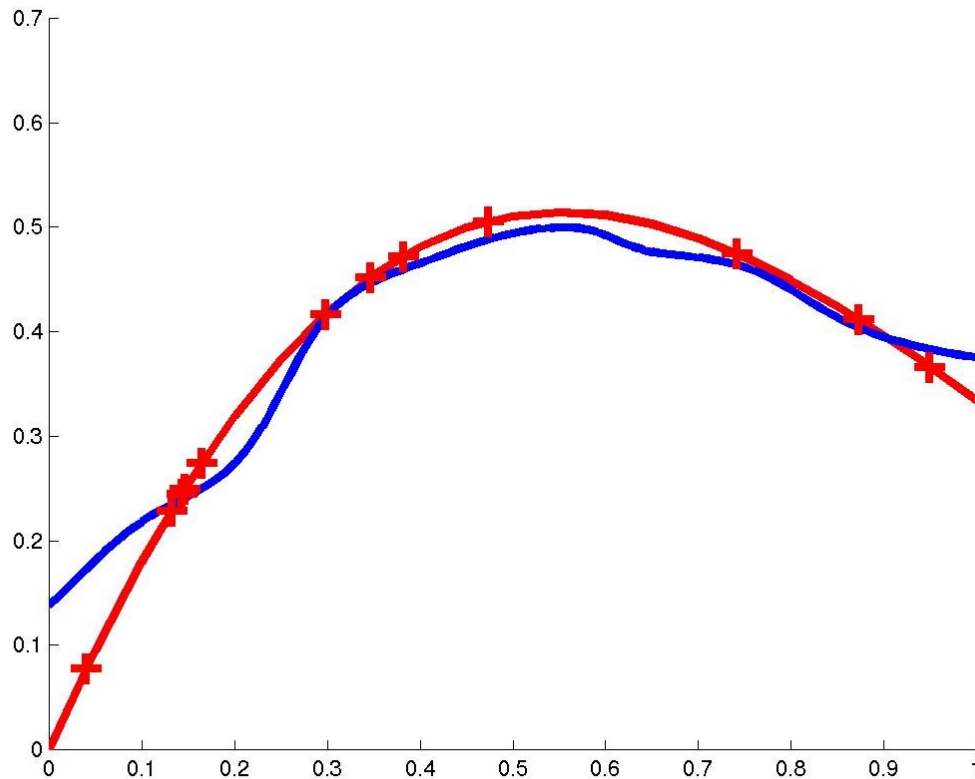
Distance Weighted NN

$$K(d(x_q, x_i)) = 1/(d_0 + d(x_q, x_i))^2$$



Distance Weighted NN

$$K(d(x_q, x_i)) = \exp(-(d(x_q, x_i)/\sigma_0)^2) - \text{Gaussian weighted}$$



Exercise - 1

Consider the following training set in 2-dimensional Euclidean space. What is the class of the point (1, 1) if 7NN classifier is considered? If the value of K is reduced whether the class will change? (Consider K=3 and K=5). What should be the final class if the above 3 values of K are considered?

Point	Coordinate	Class	Distance from (1,1)
X1	(-1, 1)	Negative	2
X2	(0, 1)	Positive	1
X3	(0, 2)	Negative	1.414
X4	(1, -1)	Negative	2
X5	(1, 0)	Positive	1
X6	(1, 2)	Positive	1
X7	(2, 2)	Negative	1.41
X8	(2, 3)	Positive	2.236

Point	Distance	Class
X2	1	Positive
X5	1	Positive
X6	1	Positive
X7	1.41	Negative
X3	1.414	Negative
X1	2	Negative
X4	2	Negative

Exercise - 2

Based on the information given in the table below, find the customer type of $x_q = C4$ using K-NN. In given example consider all the instances and use locally weighted K-NN algorithm with kernel $K(d(x_q, x_i)) = 1/d(x_q, x_i)^2$.

Apply min-max normalization on income to obtain $[0,1]$ range. Consider profession and Region as nominal. Consider Locality as ordinal variable with ranking order of [Village, Small Town, Suburban, Metropolitan]. Give equal weight to each attribute.

Customer	Income	Profession	Region	Locality	Category
C0	60000	Doctor	Hindi	Village	L1
C1	70000	Doctor	Bengali	Village	L2
C2	60000	Carpenter	Hindi	Suburban	L2
C3	80000	Doctor	Bhojpuri	Metropolitan	L2
C5	80000	Data Scientist	Hindi	Small Town	L1
C4	50000	Data Scientist	Hindi	Small Town	

Exercise - 2

Based on the information given in the table below, find the customer type of $x_q = C4$ using K-NN. In given example consider all the instances and use locally weighted K-NN algorithm with kernel $K(d(x_q, x_i)) = 1/d(x_q, x_i)^2$.

Apply min-max normalization on income to obtain $[0,1]$ range. Consider profession and Region as nominal. Consider Locality as ordinal variable with ranking order of [Village, Small Town, Suburban, Metropolitan]. Give equal weight to each attribute.

Customer	Income	Profession	Region	Scaled Locality	Category	Distance	Weight
C0	0.33	Doctor	Hindi	1 \rightarrow 0	L1	0.97	1.06
C1	0.67	Doctor	Bengali	1 \rightarrow 0	L2	1.75	0.33
C2	0.33	Carpenter	Hindi	3 \rightarrow 0.67	L2	0.97	1.06
C3	1	Doctor	Bhojpuri	4 \rightarrow 1	L2	2.2	0.21
C5	1	Data Scientist	Hindi	2 \rightarrow 0.33	L1	1	1
C4	0	Data Scientist	Hindi	2 \rightarrow 0.33		0 (NA)	

$d(C4, C2)$

= distance w.r.t to numerical attributes + distance w.r.t Nominal attributes

= Euclidean distance on (Income, Scaled Locality) + Categorical distance on attributes (Profession, Region)

= $\sqrt{(\text{Income}_{c4} - \text{Income}_{c2})^2 + (\text{Locality}_{c4} - \text{Locality}_{c2})^2} + \frac{\text{No. of Categorical features} - \text{No. of Matches between C4 \& C2}}{\text{No. of Categorical Features}}$

= $\sqrt{(0 - 0.33)^2 + (0.33 - 0.67)^2} + \frac{2-1}{2}$

= $0.47 + 0.5 = 0.97$

Exercise – 2

Based on the information given in the table below, find the customer type of $x_q = C4$ using K-NN. In given example consider all the instances and use locally weighted K-NN algorithm with kernel $K(d(x_q, x_i)) = 1 / d(x_q, x_i)^2$.

Apply min-max normalization on income to obtain [0,1] range. Consider profession and Region as nominal. Consider Locality as ordinal variable with ranking order of [Village, Small Town, Suburban, Metropolitan]. Give equal weight to each attribute.

Customer	Income	Profession	Region	Scaled Locality	Category	Distance	Weight
C0	0.33	Doctor	Hindi	1 → 0	L1	0.97	1.06
C1	0.67	Doctor	Bengali	1 → 0	L2	1.75	0.33
C2	0.33	Carpenter	Hindi	3 → 0.67	L2	0.97	1.06
C3	1	Doctor	Bhojpuri	4 → 1	L2	2.2	0.21
C5	1	Data Scientist	Hindi	2 → 0.33	L1	1	1
C4	0	Data Scientist	Hindi	2 → 0.33		0 (NA)	

Inference : Without weighted KNN, L2 is the majority voted class.

But with weighted distances, L1 class instances are more similar to C4

Point	Kernel / Weight	Class
C5	1	L1
C0	1.06	L1
C2	1.06	L2
C1	0.33	L2
C3	0.21	L2

Nearest-Neighbor Classifiers: Issues

- The value of k , the number of nearest neighbors to retrieve
- Choice of Distance Metric to compute distance between records
- Expensive, Slow at query time
 - To determine the nearest neighbour of a query point q , must compute the distance to all N training examples
 - + Pre-sort training examples into fast data structures (kd-trees)
 - + Remove redundant data (condensing)
- Storage Requirements
 - Must store all training data **P**
 - + Remove redundant data (condensing)

When to Consider Nearest Neighbors

- Suitable for Low dimensional datasets
- Local information can yield highly adaptive behavior
- Lots of training data (distance-weighted KNN)
 - Training is very fast
- Learn complex target functions
 - Do not lose information
- Noisy training data (distance-weighted KNN)
 - by taking the weighted average of the k neighbors nearest to the query point, it can smooth out the impact of isolated noisy training examples.



Lab : Practise :

Take any BITS virtual Lab dataset and experiment on below cases

- **Tune KNN.** Try different values of k to see if you can improve the performance of the algorithm.
- **Use weighted KNN** and check the performance
- **Regression.** Apply it to a regression predictive modelling problem (e.g. predict a numerical value)
- **More Distance Measures.** Implement other distance measures such as Hamming distance, Manhattan distance and Minkowski distance.
- **Data Preparation.** Distance measures are strongly affected by the scale of the input data. Experiment with and without standardization and other data preparation methods in order to improve results.

Thank you !

Required Reading for completed session :

T1 - Chapter # 8 (Tom M. Mitchell, Machine Learning)

Next Session Plan :

Remaining Topics Module 6 : Radial Basis function, locally weighted KNN on regression example

Module 7 : Support vector machines

Important Note for SVM : Below Math Pre-requisite (Yet to be discussed in MFML parallel course) :

Lagrange Multipliers

KKT – Conditions

Primal – Dual for SVM

Kernel for Non-Linear SVM