LEARN. DO. EARN

ACADGILD

MACHINE LEARNING WITH R

Website : http://www.acadgild.com
LinkedIn : https://www.linkedin.com/company/acadgild
Facebook : https://www.facebook.com/acadgild

Session 2: Nearest Neighbor Classification

Agenda

Sl. No.	Agenda Topics
1.	Instance Based Classifiers
2.	Nearest Neighbor Classifiers
3.	Definition of Nearest Neighbor
4.	1 Nearest-Neighbor
5.	Nearest Neighbor Classification
6.	Lazy vs. Eager Learning
7.	Lazy Learner: Instance-Based Methods
8.	Nearest Neighbor Search
9.	Non-Numeric Data
10.	Dealing With Non-numeric Data
11.	Preprocessing Your Dataset

Sl. No.	Agenda Topics
12.	K-NN Variations
13.	How To Determine The Good Value For K?
14.	Other Distance Measures
15.	K-NN Time Complexity
16.	Curse of Dimensionality
17.	When to Consider Nearest Neighbors
18.	Proximity Graphs
19.	Nearest Neighbour Issues

Instance Based Classifiers

Set of Stored Cases

Atr1	………	AtrN	Class
			A
			B
			B
			C
			A
			C
			B

- Store the training records
- Use training records to predict the class label of unseen cases

Unseen Case

Atr1	………	AtrN

Instance Based Classifiers (Contd.)

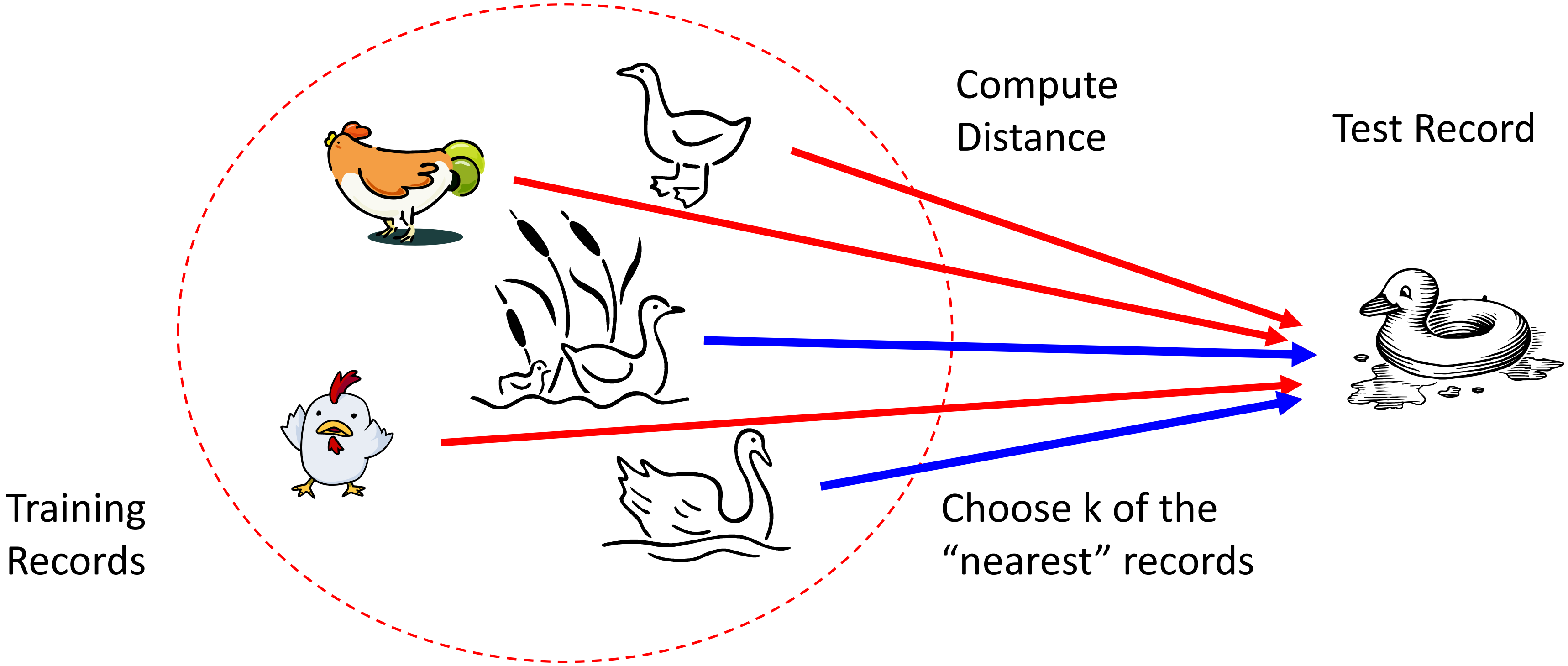
Examples:

- Rote-learner
  - Memorizes entire training data and performs classification only if attributes of record match one of the training examples exactly

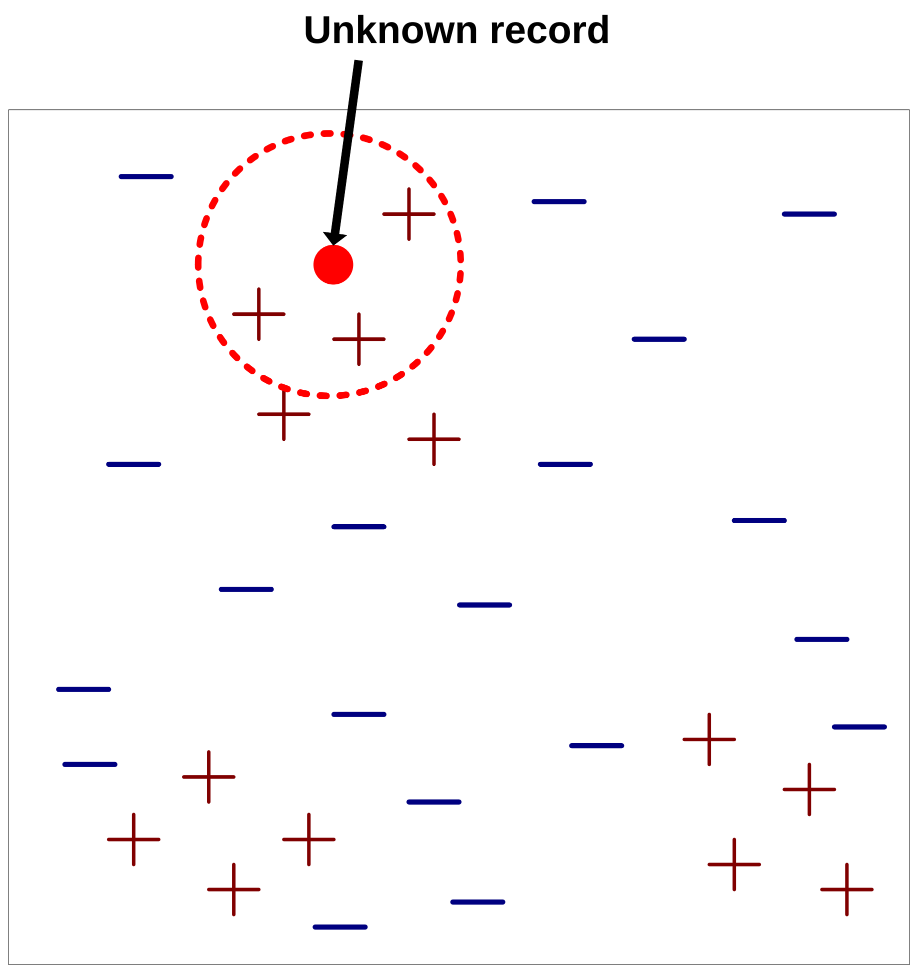
- Nearest neighbor
  - Uses k “closest” points (nearest neighbors) for performing classification

Nearest Neighbor Classifiers

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



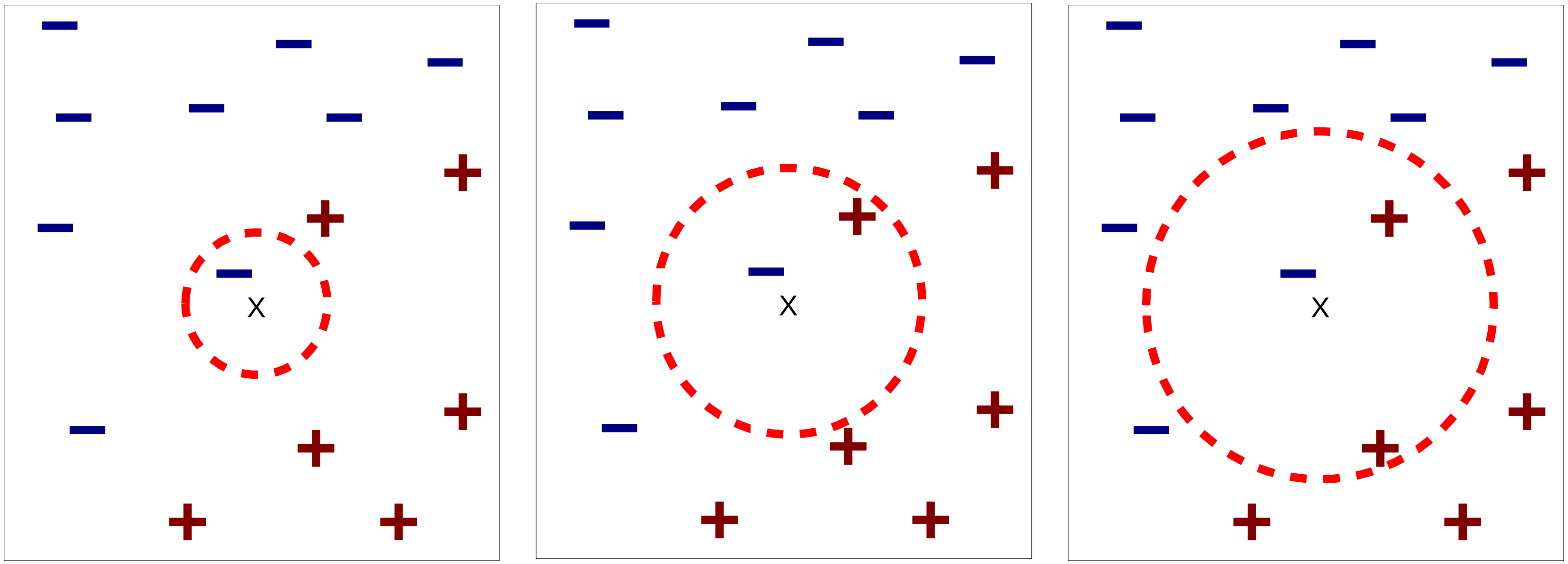
Nearest Neighbor Classifiers (Contd.)



- Requires three things:
  - The set of stored records
  - Distance Metric to compute distance between records
  - The value of k, the number of nearest neighbors to retrieve

- To classify an unknown record:
  - Compute distance to other training records
  - Identify k nearest neighbors
  - Use class labels of nearest neighbors to determine the class label of unknown record (e.g., by taking majority vote)

Definition of Nearest Neighbor

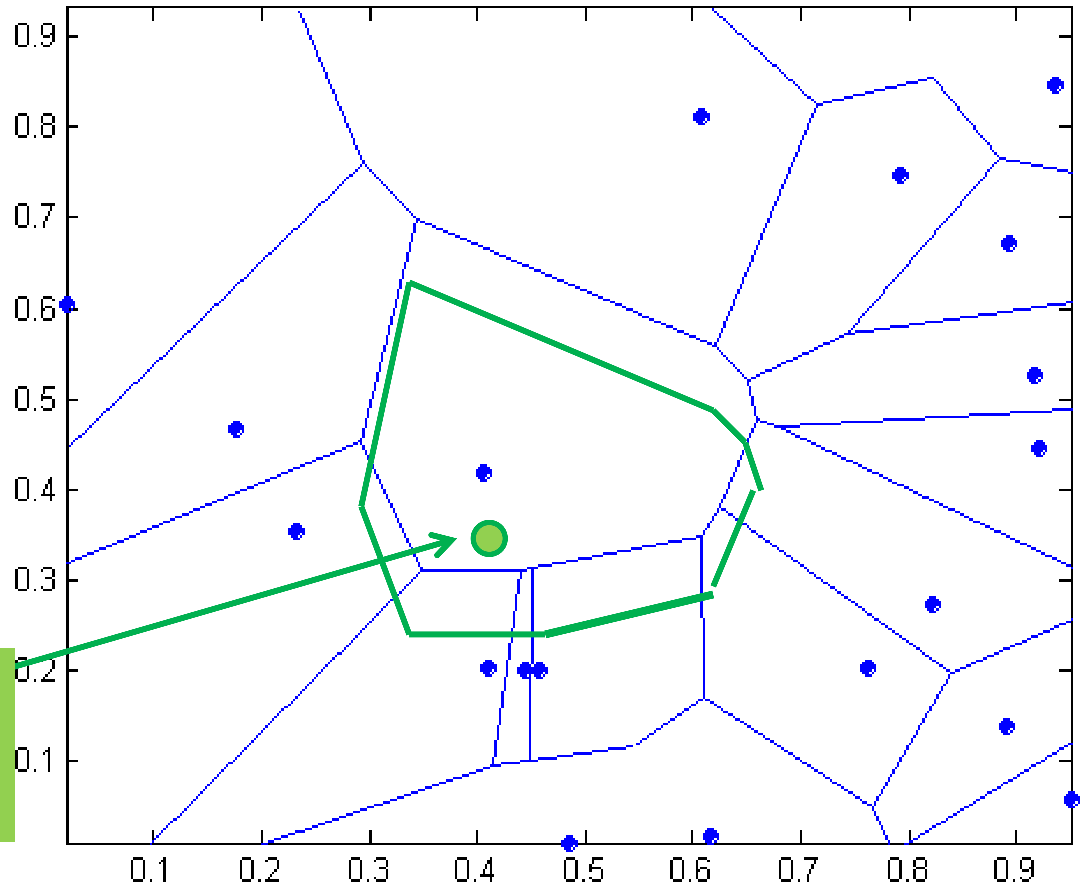


(a) 1-nearest neighbor (b) 2-nearest neighbor (c) 3-nearest neighbor

K-nearest neighbors of a record x are data points that have the k smallest distance to x

1 Nearest-Neighbor

- Voronoi Diagram defines the classification boundary



The area takes the class of the green point

Nearest Neighbor Classification

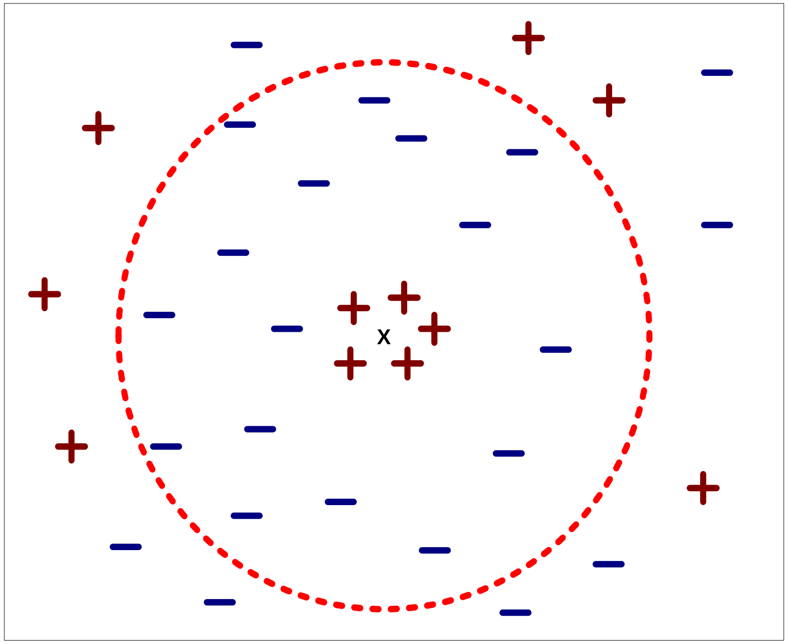
- Compute distance between two points:
  - Euclidean distance

$$d(p,q) = \sqrt{\sum_i (p_i - q_i)^2}$$

- Determine the class from nearest neighbor list
  - take the majority vote of class labels among the k-nearest neighbors
  - Weigh the vote according to distance
    - weight factor, w = 1/d2

Nearest Neighbor Classification (Contd.)

- 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



Nearest Neighbor Classification (Contd.)

- Scaling issues
  - Attributes may have to be scaled to prevent distance measures from being dominated by one of the attributes

Example:

- height of a person may vary from 1.5m to 1.8m
- weight of a person may vary from 90lb to 300lb
- income of a person may vary from $10K to $1M

Nearest Neighbor Classification (Contd.)

- Problem with Euclidean measure:
  - High dimensional data
    - curse of dimensionality
  - Can produce counter-intuitive results

1 1 1 1 1 1 1 1 1 1 1 0	vs	1 0 0 0 0 0 0 0 0 0 0 0
0 1 1 1 1 1 1 1 1 1 1 1		0 0 0 0 0 0 0 0 0 0 0 1
d = 1.4142		d = 1.4142

◆ Solution: Normalize the vectors to unit length

Nearest neighbor Classification (Contd.)

- k-NN classifiers are lazy learners
  - It does not build models explicitly
  - Unlike eager learners such as decision tree induction and rule-based systems
- Classifying unknown records are relatively expensive
  - Naïve algorithm: O(n)
  - Need for structures to retrieve nearest neighbors fast.
    - The Nearest Neighbor Search problem.

Lazy vs. Eager Learning

- Lazy vs. eager learning
  - Lazy learning (e.g., instance-based learning): Simply stores training data (or only minor processing) and waits until it is given a test tuple
  - Eager learning (the above discussed methods): Given a set of training set, constructs a classification model before receiving new (e.g., test) data to classify
- Lazy: less time in training but more time in predicting
- Accuracy
  - Lazy method effectively uses a richer hypothesis space since it uses many local linear functions to form its implicit global approximation to the target function
  - Eager: must commit to a single hypothesis that covers the entire instance space

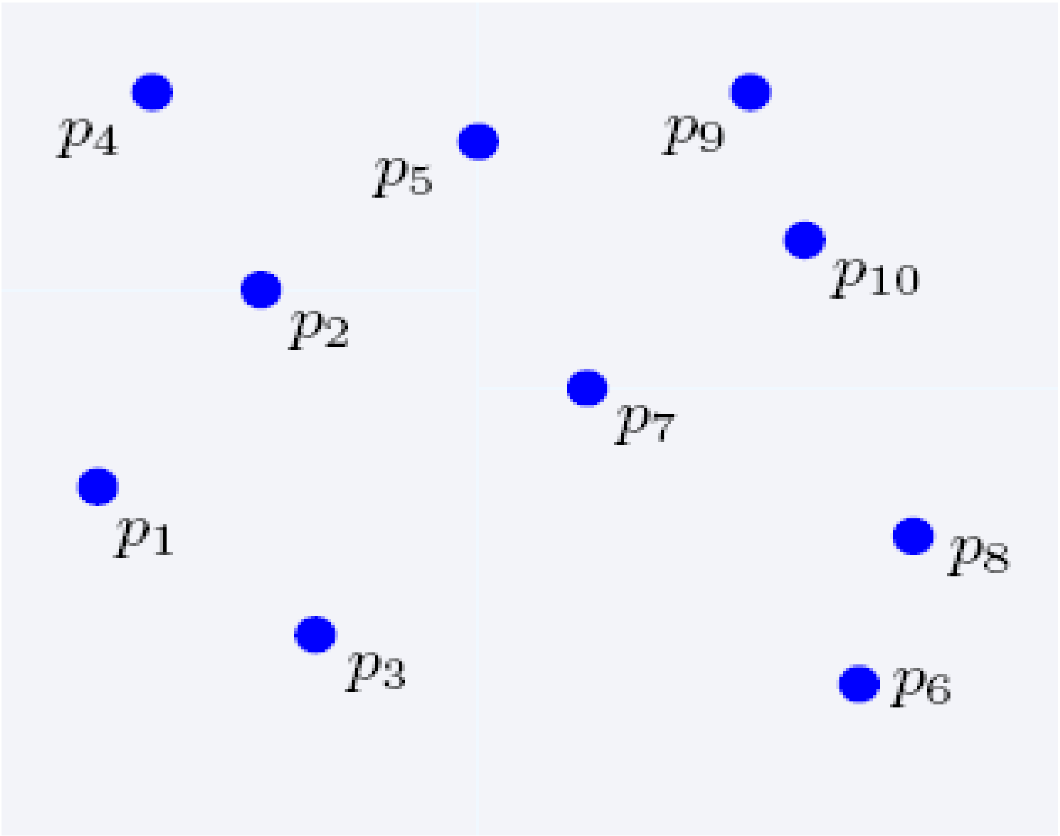
Lazy Learner: Instance-Based Methods

- Instance-based learning:
  - Store training examples and delay the processing (“lazy evaluation”) until a new instance must be classified
- Typical approaches
  - k-nearest neighbor approach
    - Instances represented as points in a Euclidean space.
  - Locally weighted regression
    - Constructs local approximation
  - Case-based reasoning
    - Uses symbolic representations and knowledge-based inference

Nearest Neighbor Search

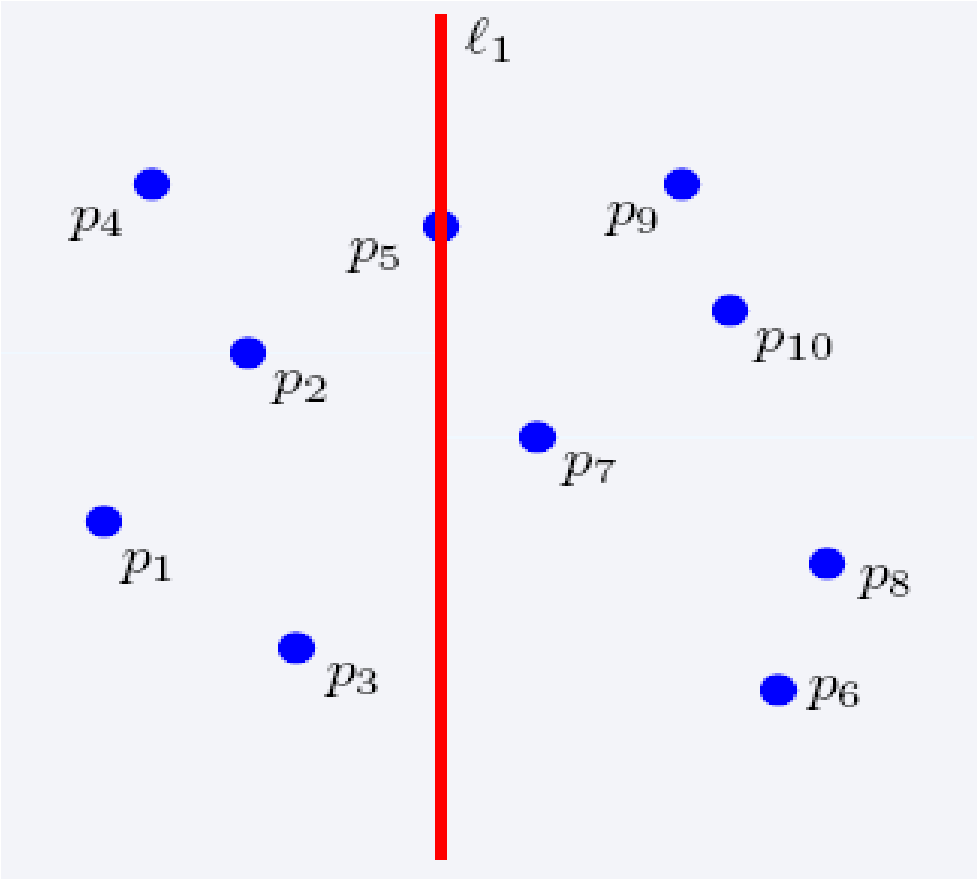
- Two-dimensional kd-trees
  - A data structure for answering nearest neighbor queries in R2
- kd-tree construction algorithm
  - Select the x or y dimension (alternating between the two)
  - Partition the space into two with a line passing from the median point
  - Repeat recursively in the two partitions as long as there are enough points

Nearest Neighbor Search (Contd.)



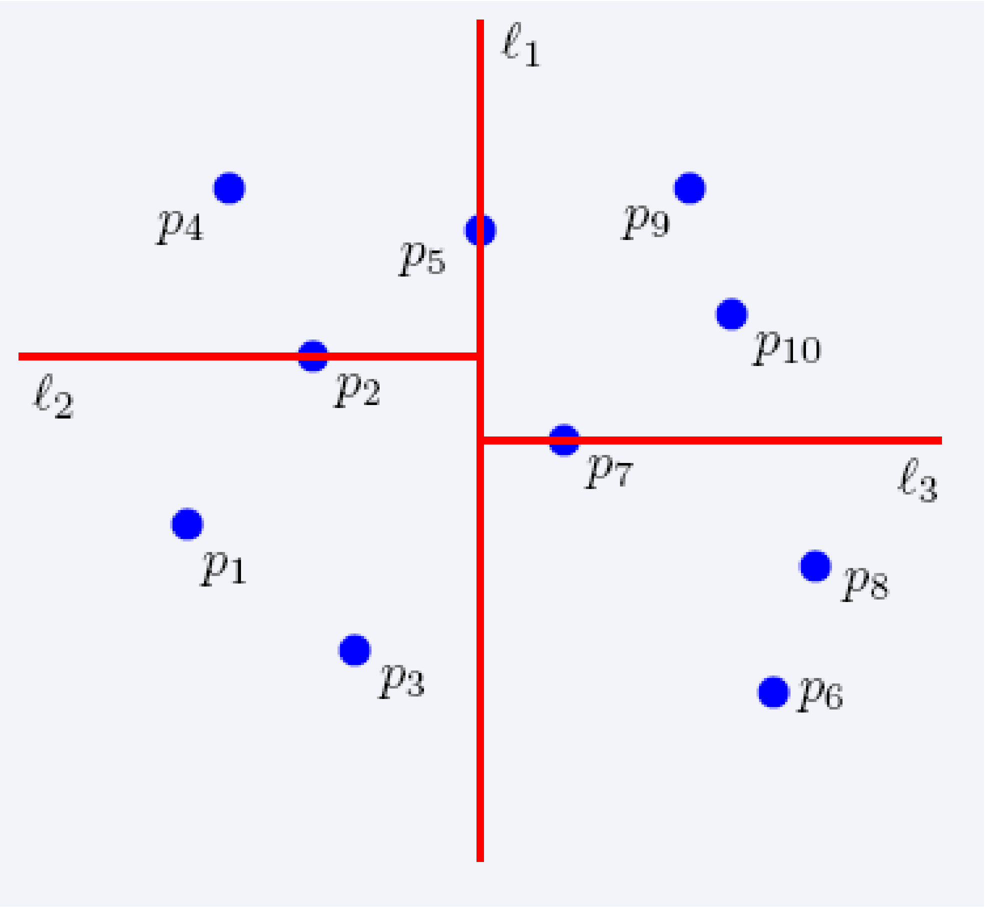
2-dimensional kd-trees

Nearest Neighbor Search (Contd.)



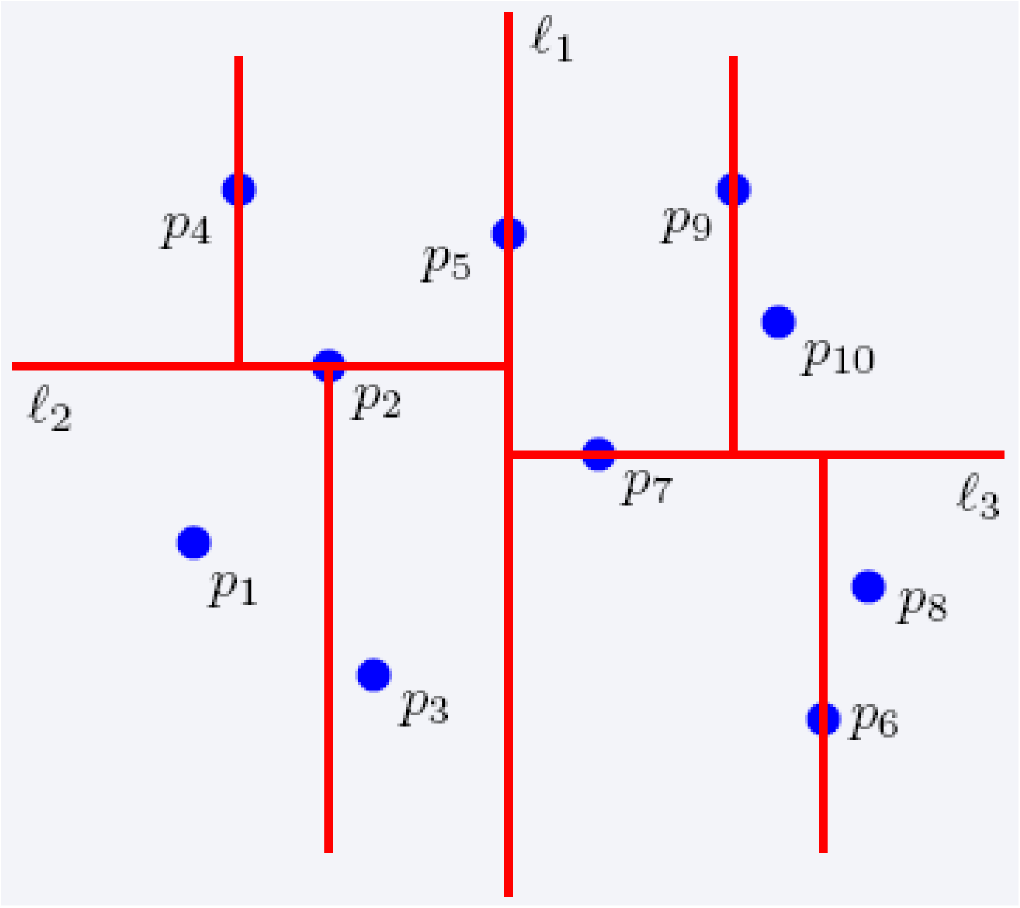
2 dimensional kd-trees

Nearest Neighbor Search (Contd.)



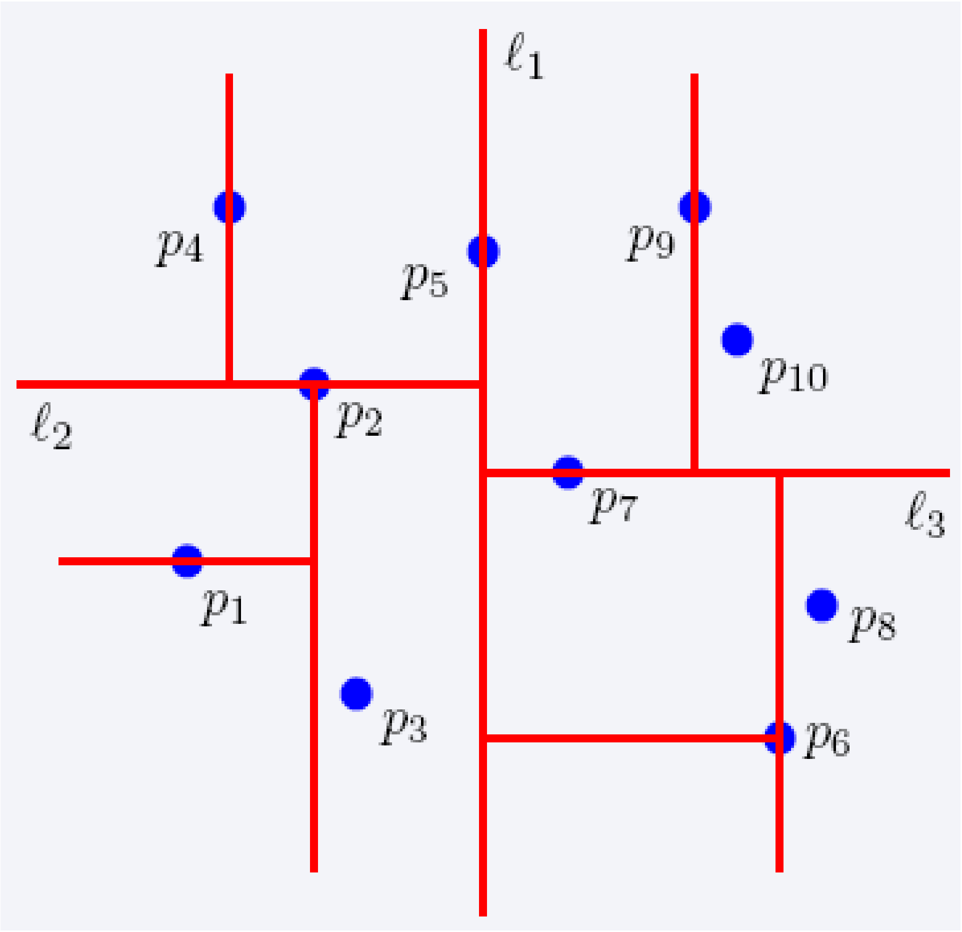
2 dimensional kd-trees

Nearest Neighbor Search (Contd.)



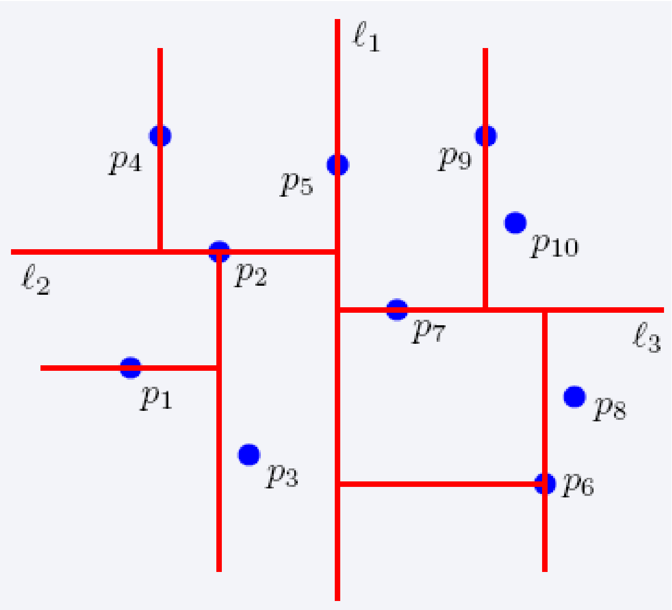
2-dimensional kd-trees

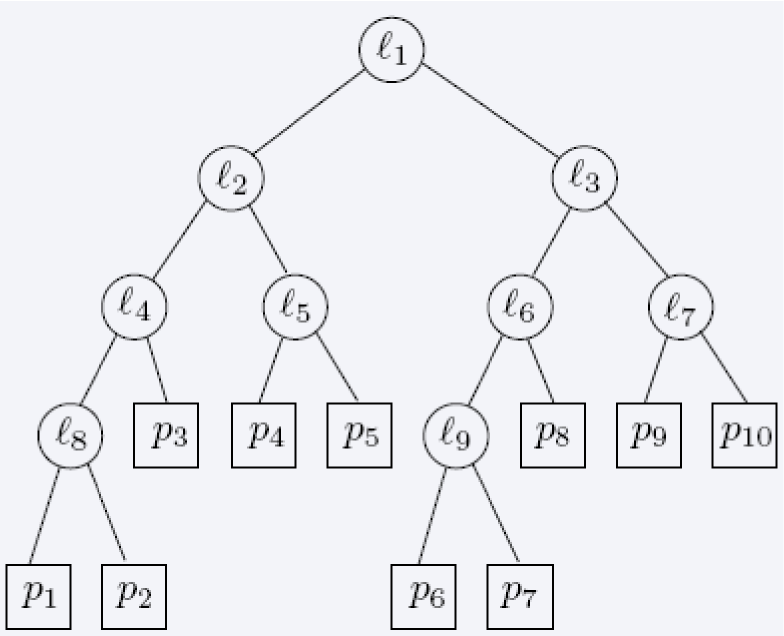
Nearest Neighbor Search (Contd.)



2-dimensional kd-trees

2-dimensional kd-trees

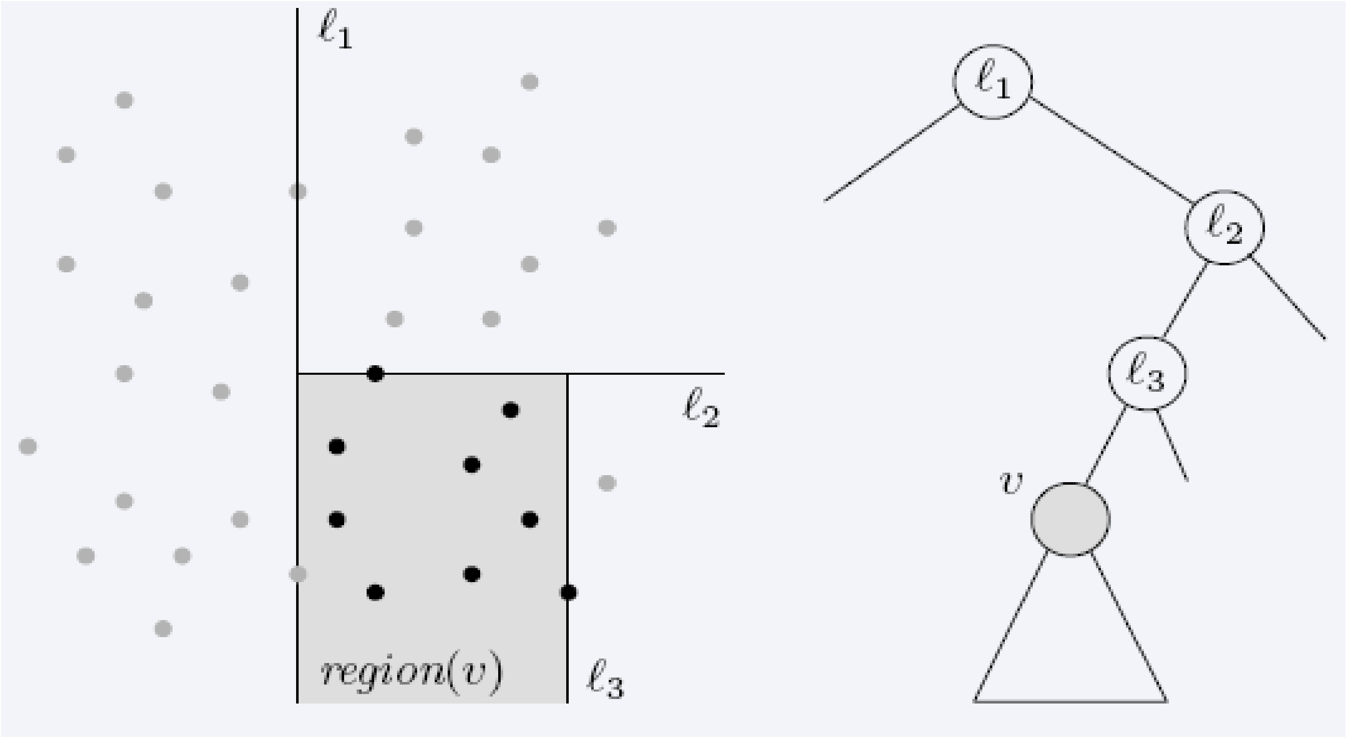




Nearest Neighbor Search (Contd.)

2-dimensional kd-trees

region(u) – all the black points in the subtree of u



2-dimensional kd-trees

- A binary tree:
  - Size O(n)
  - Depth O(logn)
  - Construction time O(nlogn)
  - Query time: worst case O(n), but for many cases O(logn)
- Generalizes to d dimensions
- Example of Binary Space Partitioning

Non-Numeric Data

- Feature values are not always numbers

Example

  - Boolean values: Yes or no, presence or absence of an attribute
  - Categories: Colors, educational attainment, gender
- How do these values factor into the computation of distance?

Dealing With Non-numeric Data

- Boolean values => convert to 0 or 1
  - Applies to yes-no/presence-absence attributes
- Non-binary characterizations
  - Use natural progression when applicable; e.g., educational attainment: GS, HS, College, MS, PHD => 1,2,3,4,5
  - Assign arbitrary numbers but be careful about distances; e.g., color: red, yellow, blue => 1,2,3
- How about unavailable data?
  (0 value not always the answer)

Preprocessing Your Dataset

- Dataset may need to be preprocessed to ensure more reliable data mining results
- Conversion of non-numeric data to numeric data
- Calibration of numeric data to reduce effects of disparate ranges
  - Particularly when using the Euclidean distance metric

K-NN Variations

- Value of k
  - Larger k increases confidence in prediction
  - Note that if k is too large, decision may be skewed
- Weighted evaluation of nearest neighbors
  - Plain majority may unfairly skew decision
  - Revise algorithm so that closer neighbors have greater “vote weight”
- Other distance measures

How To Determine The Good Value For K?

- Determined experimentally
- Start with k=1 and use a test set to validate the error rate of the classifier
- Repeat with k=k+2
- Choose the value of k for which the error rate is minimum

Note: k should be odd number to avoid ties

Other Distance Measures

- City-block distance (Manhattan dist)
  - Add absolute value of differences
- Cosine similarity
  - Measure angle formed by the two samples (with the origin)
- Jaccard distance
  - Determine percentage of exact matches between the samples (not including unavailable data)
- Others

K-NN Time Complexity

- Suppose there are m instances and n features in the dataset
- Nearest neighbor algorithm requires computing m distances
- Each distance computation involves scanning through each feature value
- Running time complexity is proportional to m X n

Curse of Dimensionality

- Imagine instances described by 20 features (attributes) but only 3 are relevant to target function
- Curse of dimensionality: nearest neighbor is easily misled when instance space is high-dimensional
- Dominated by large number of irrelevant features

Possible solutions:

- Stretch j-th axis by weight zj, where z1,…,zn chosen to minimize prediction error (weight different features differently)
- Use cross-validation to automatically choose weights z1,…,zn
- Note setting zj to zero eliminates this dimension altogether (feature subset selection)
- PCA

When to Consider Nearest Neighbors

- Instances map to points in Rd
- Less than 20 features (attributes) per instance, typically normalized
- Lot of training data

Advantages:

- Training is very fast
- Learn complex target functions
- Do not lose information

Disadvantages:

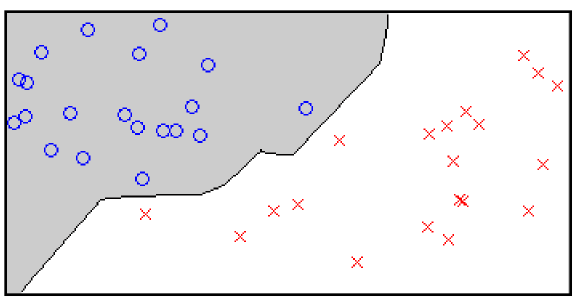
- Slow at query time
  - Presorting and indexing training samples into search trees reduces time
- Easily misled by irrelevant features (attributes)

Condensing

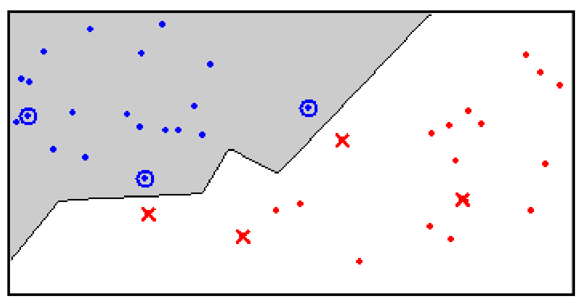
- Aim is to reduce the number of training samples
- Retain only the samples that are needed to define the decision boundary
- This is reminiscent of a Support Vector Machine

- Decision Boundary Consistent – a subset whose nearest neighbour decision boundary is identical to the boundary of the entire training set

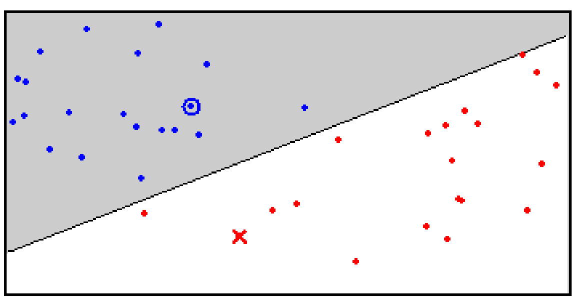
- Minimum Consistent Set – the smallest subset of the training data that correctly classifies all of the original training data



Original data



Condensed data



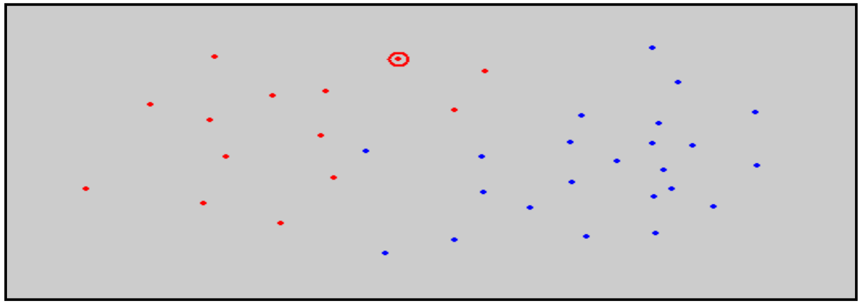
Minimum Consistent Set

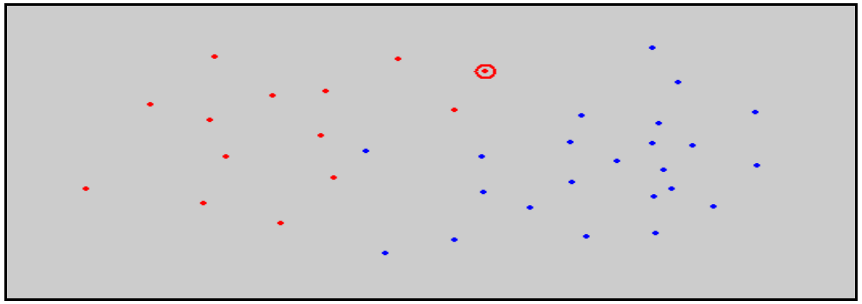
Condensing (Contd.)

- Condensed Nearest Neighbour (CNN)-Hart 1968
  - Incremental
  - Order dependent
  - Neither minimal nor decision boundary consistent
  - $O(n^3)$ for brute-force method
  - Can follow up with reduced NN [Gates72]

Remove a sample if doing so does not cause any incorrect classifications

1.Initialize subset with a single training example
2.Classify all remaining samples using the subset, and transfer any incorrectly classified samples to the subset
3.Return to 2 until no transfers occurred or the subset is full



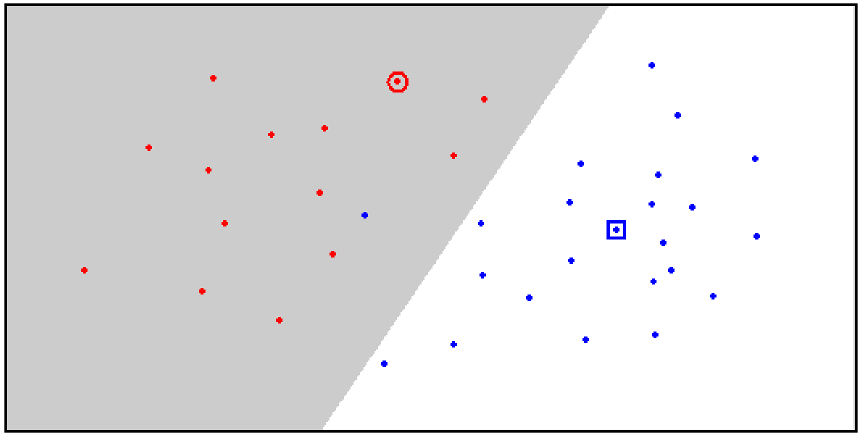


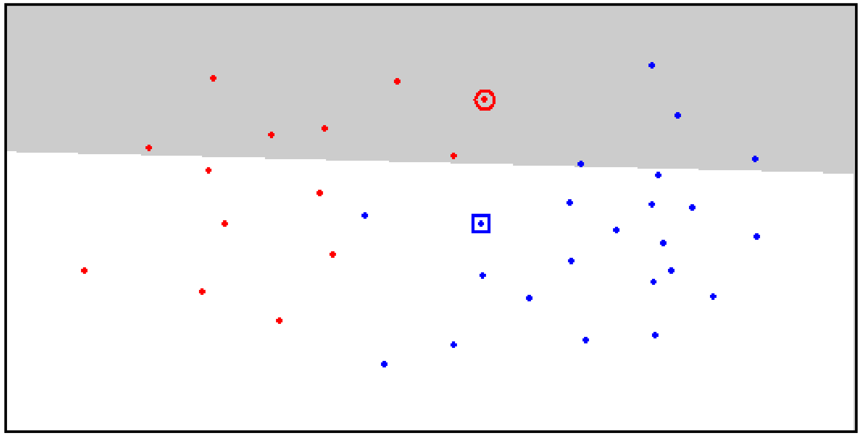
Condensing (Contd.)

- Condensed Nearest Neighbour (CNN) Hart 1968
  - Incremental
  - Order dependent
  - Neither minimal nor decision boundary consistent
  - O(n3) for brute-force method
  - Can follow up with reduced NN [Gates72]

1.Initialize subset with a single training example
2.Classify all remaining samples using the subset, and transfer any incorrectly classified samples to the subset
3.Return to 2 until no transfers occurred or the subset is full

Remove a sample if doing so does not cause any incorrect classifications



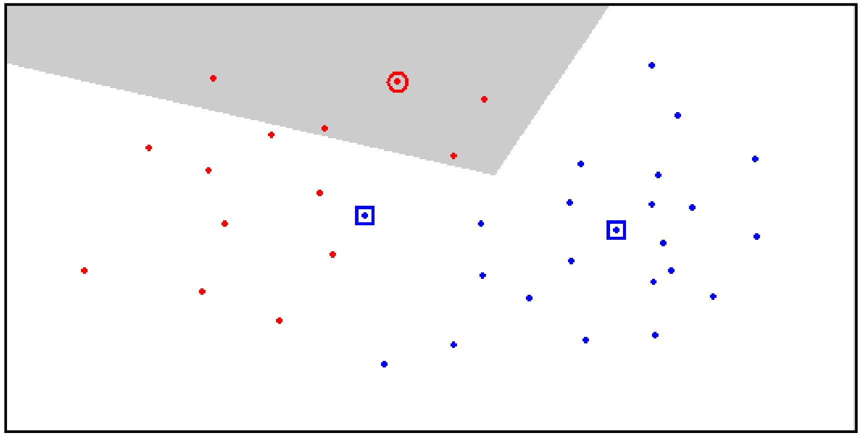


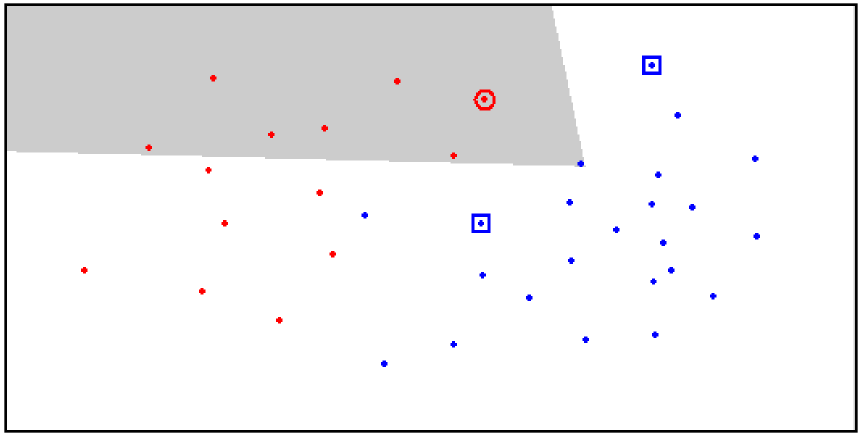
Condensing (Contd.)

- Condensed Nearest Neighbour (CNN) Hart 1968
  - Incremental
  - Order dependent
  - Neither minimal nor decision boundary consistent
  - O(n3) for brute-force method
  - Can follow up with reduced NN [Gates72]

Remove a sample if doing so does not cause any incorrect classifications

1.Initialize subset with a single training example
2.Classify all remaining samples using the subset, and transfer any incorrectly classified samples to the subset
3.Return to 2 until no transfers occurred or the subset is full



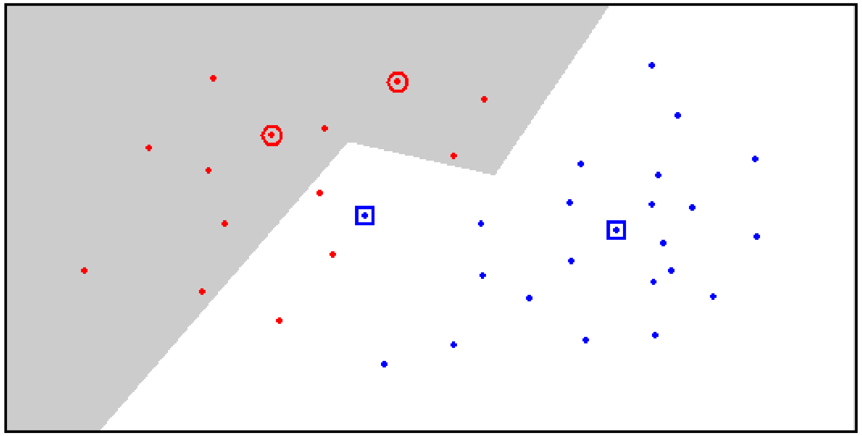


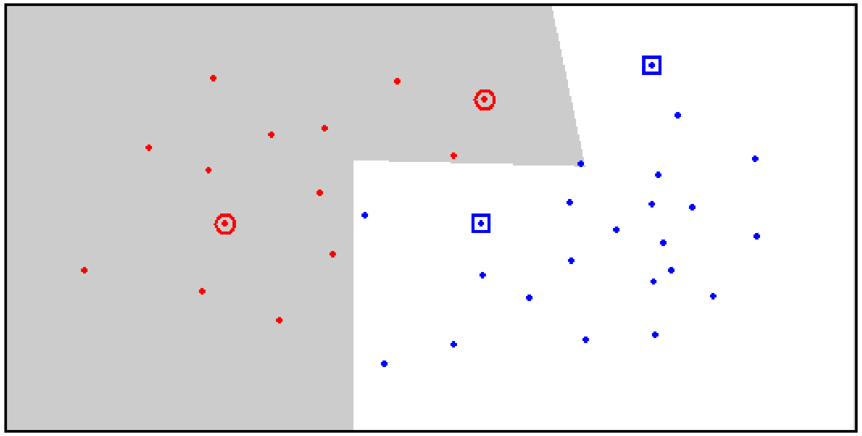
Condensing (Contd.)

- Condensed Nearest Neighbour (CNN) Hart 1968
  - Incremental
  - Order dependent
  - Neither minimal nor decision boundary consistent
  - O(n3) for brute-force method
  - Can follow up with reduced NN [Gates72]

Remove a sample if doing so does not cause any incorrect classifications

1.Initialize subset with a single training example
2.Classify all remaining samples using the subset, and transfer any incorrectly classified samples to the subset
3.Return to 2 until no transfers occurred or the subset is full

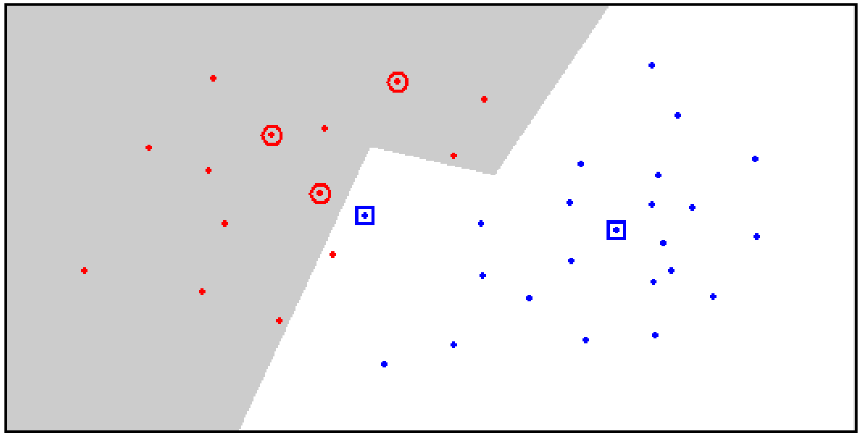


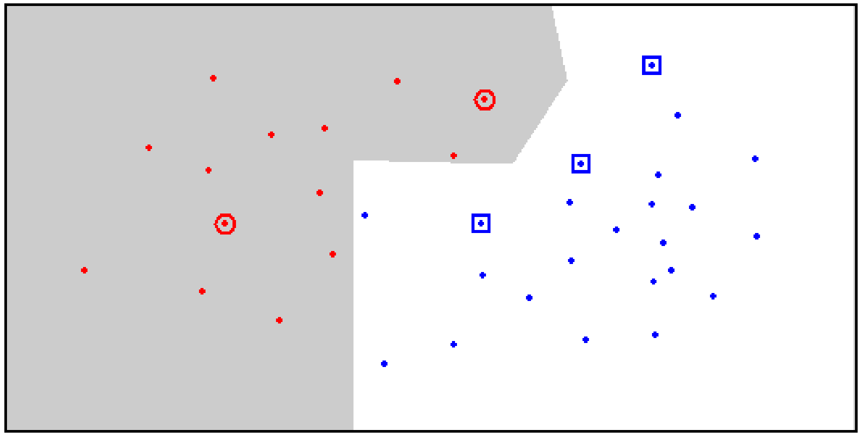


- Condensed Nearest Neighbour (CNN) Hart 1968
  - Incremental
  - Order dependent
  - Neither minimal nor decision boundary consistent
  - O(n3) for brute-force method
  - Can follow up with reduced NN [Gates72]

  Remove a sample if doing so does not cause any incorrect classifications

1.Initialize subset with a single training example
2.Classify all remaining samples using the subset, and transfer any incorrectly classified samples to the subset
3.Return to 2 until no transfers occurred or the subset is full



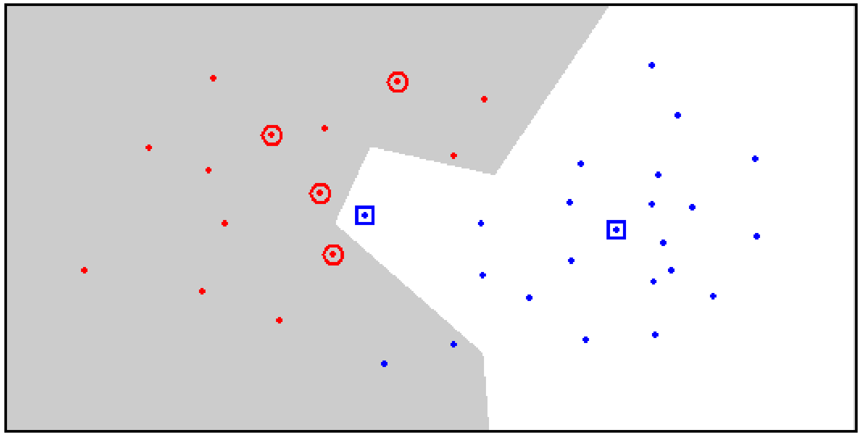


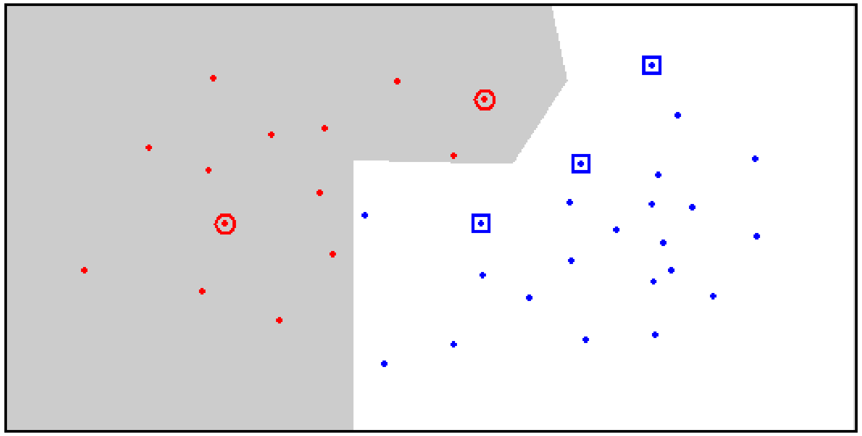
Condensing (Contd.)

- Condensed Nearest Neighbour (CNN) Hart 1968
  - Incremental
  - Order dependent
  - Neither minimal nor decision boundary consistent
  - O(n3) for brute-force method
  - Can follow up with reduced NN [Gates72]

  Remove a sample if doing so does not cause any incorrect classifications

1.Initialize subset with a single training example
2.Classify all remaining samples using the subset, and transfer any incorrectly classified samples to the subset
3.Return to 2 until no transfers occurred or the subset is full



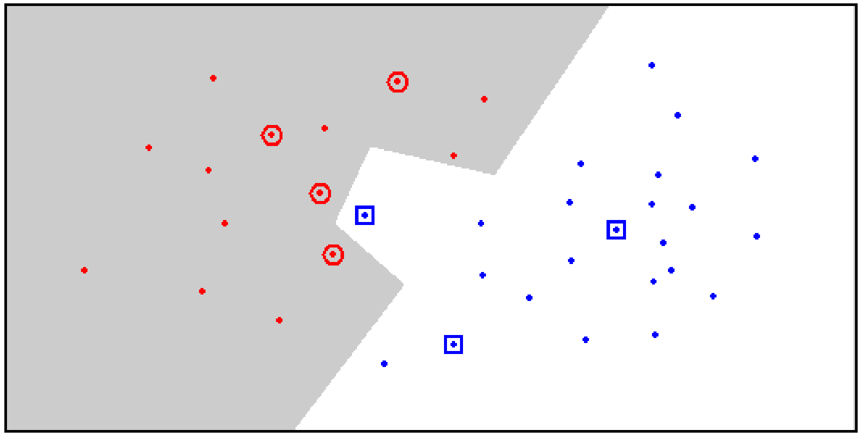


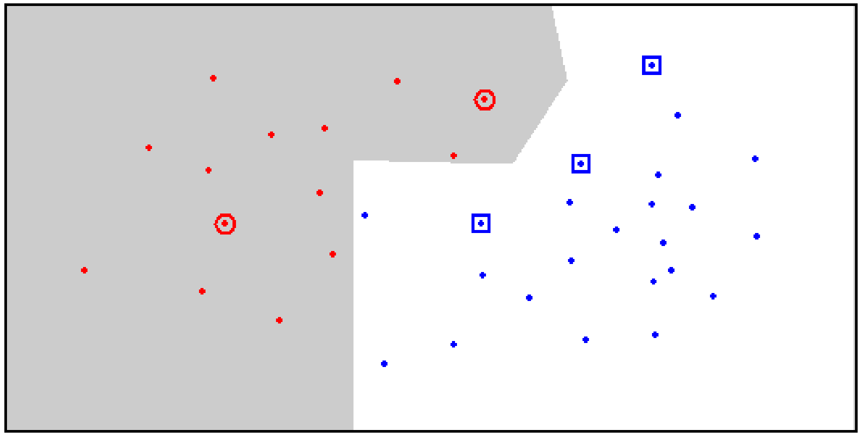
Condensing (Contd.)

- Condensed Nearest Neighbour (CNN) Hart 1968
  - Incremental
  - Order dependent
  - Neither minimal nor decision boundary consistent
  - O(n3) for brute-force method
  - Can follow up with reduced NN [Gates72]

Remove a sample if doing so does not cause any incorrect classifications

1.Initialize subset with a single training example
2.Classify all remaining samples using the subset, and transfer any incorrectly classified samples to the subset
3.Return to 2 until no transfers occurred or the subset is full





Proximity Graphs

- Condensing aims to retain points along the decision boundary
- How to identify such points?
  - Neighbouring points of different classes
- Proximity graphs provide various definitions of "neighbour"

$$NNG \subseteq MST \subseteq RNG \subseteq GG \subseteq DT$$

NNG = Nearest Neighbour Graph

MST = Minimum Spanning Tree

RNG = Relative Neighbourhood Graph

GG = Gabriel Graph

DT = Delaunay Triangulation

Nearest Neighbour Issues

- Expensive
  - 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)
    - Compute only an approximate distance (LSH)
    - Remove redundant data (condensing)
- Storage Requirements
  - Must store all training data P
    - Remove redundant data (condensing)
    - Pre-sorting often increases the storage requirements
- High Dimensional Data
  - "Curse of Dimensionality"
    - Required amount of training data increases exponentially with dimension
    - Computational cost also increases dramatically
    - Partitioning techniques degrade to linear search in high dimension

Next Class: Naïve Bayes

Sl. No.	Agenda Topics
1.	Things We'd Like To Do
2.	Classification Problem
3.	Another Application
4.	Naïve Bayes Learning
5.	A Refresher on Probability
6.	Back to the Naïve Bayes Classifier
7.	Bayesian Theorem: Basics
8.	Deriving the Naïve Bayes
9.	Estimating Parameters For the Target Function
10.	Naïve Assumptions of Independence
11.	Again About Estimation

Sl. No.	Agenda Topics
12.	The Bayes Classifier
13.	Model Parameters
14.	The Naïve Bayes Model
15.	Why Is This Useful?
16.	Naïve Bayes Training
17.	Naïve Bayes Classification
18.	Another Example of the Naïve Bayes Classifier
19.	The Naive Bayes Classifier for Data Sets with Numerical Attribute Values
20.	Numeric Weather Data with Summary Statistics
21.	Output Probabilities
22.	Performance on a Test Set
23.	Naïve Bayes Assumption
24.	Exclusive-OR Example

Next Class: Naïve Bayes (Contd.)

Sl. No.	Agenda Topics
25.	Evaluating Classification Algorithms
26.	Types of Errors
27.	Sensitivity and Specificity
28.	The ROC Space
29.	The ROC Curve
30.	ROC Analysis
31.	Holdout Estimation
32.	Repeated Holdout Method
33.	Cross-Validation
34.	Leave-One-Out Cross-Validation
35.	Leave-One-Out-CV and Stratification
36.	Points to Remember


Q & A time

Get in Touch with ACADGILD

Contact Info:

- **Website : http://www.acadgild.com**
- **LinkedIn : https://www.linkedin.com/company/acadgild**
- **Facebook : https://www.facebook.com/acadgild**
- **Support: support@acadgild.com**