DATA MINING: **INTRODUCTION**

References:

- Pang-Ning Tan, Michael Steinbach, Vipin Kumar, ***Introduction to Data Mining***, Pearson.
- Jiawei Han, Micheline Kamber, Jian Pei, ***Data Mining Concepts and Techniques***, Third Edition, Elsevier.
- Mohammed J. Zaki, Wagner Meira Jr., ***Data Mining and Analysis: Fundamental Concepts and Algorithms,*** Cambridge University press.

Email:mazlaghani@aut.ac.ir

Files address:
fileserver\common\mazlaghani\Data Mining

Grading

- Homework (20%)
- Seminar (10%)
- Midterm + Final(30%+40%)

Introduction

- vast amounts of data
- Gather whatever data you can whenever and wherever possible.
- extracting useful challenging
- well-known applications

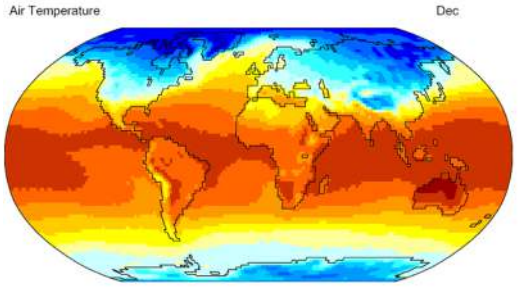
Business

- data about customer purchases
- better understand the needs of customers
- make more informed business decisions

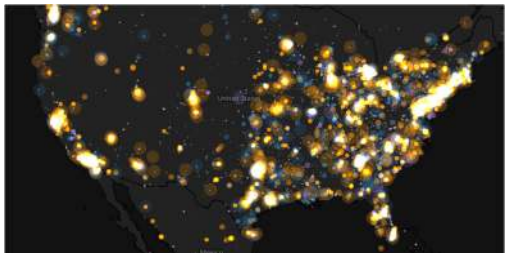
Introduction

Medicine, Science, and Engineering

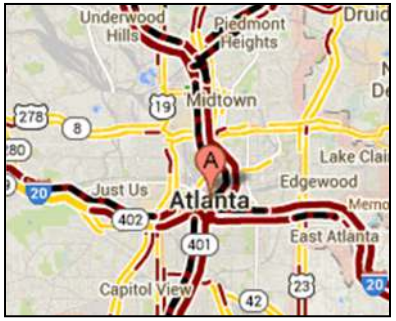
- Earth's climate system
- genomic data:microarray
- *Sensor Networks*



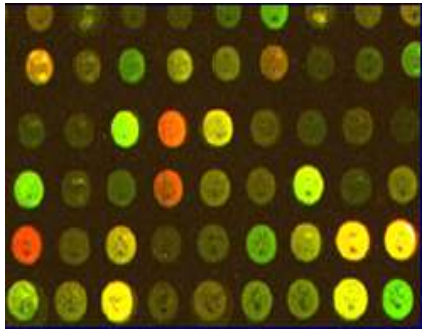
Surface Temperature of Earth



Social Networking



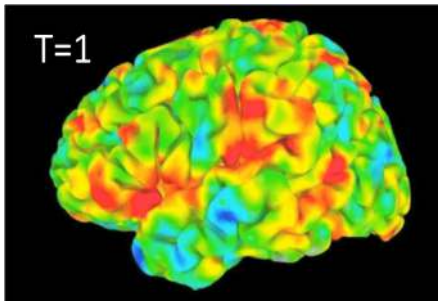
Traffic Patterns



Gene Expression Data

Introduction

- Web data
  - Yahoo has Peta Bytes of web data
  - Facebook has billions of active users
- Bank/Credit Card transactions
- FMRI



fMRI Data from Brain

What id data mining?

- Data mining is the process of automatically discovering useful information in large data repositories
- Non-trivial extraction of implicit, previously unknown and potentially useful information from data
- Predict future

Data Mining

preprocessing : transform the raw input data into an appropriate format for subsequent analysis.

fusing data
cleaning data
selecting features

Postprocessing: only valid and useful results are incorporated

Visualization
Statistical measures

Motivating Challenges

Origins of Data Mining

- Statistics
- Machine Learning
  - ❖ Optimization
  - ❖ Information theory
  - ❖ Signal Processing

Data Mining Tasks

- Prediction Methods
  Use some variables to predict unknown or future values of other variables

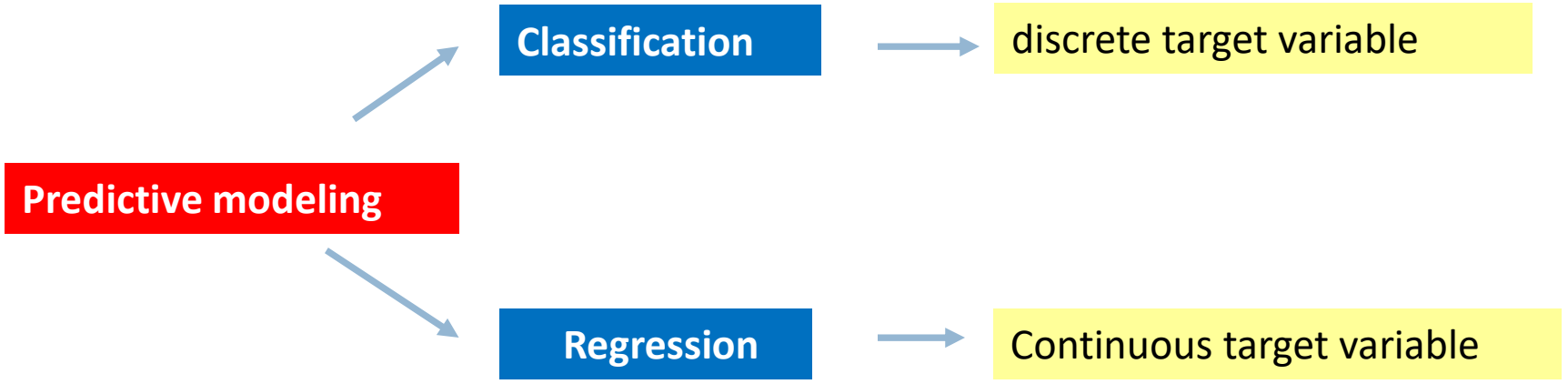
- Description Methods
  Find human-interpretable patterns that describe the data.

Core Data Mining Tasks

1. Predictive modeling

building a model for the target variable as a function of the explanatory variables

Data Mining Tasks

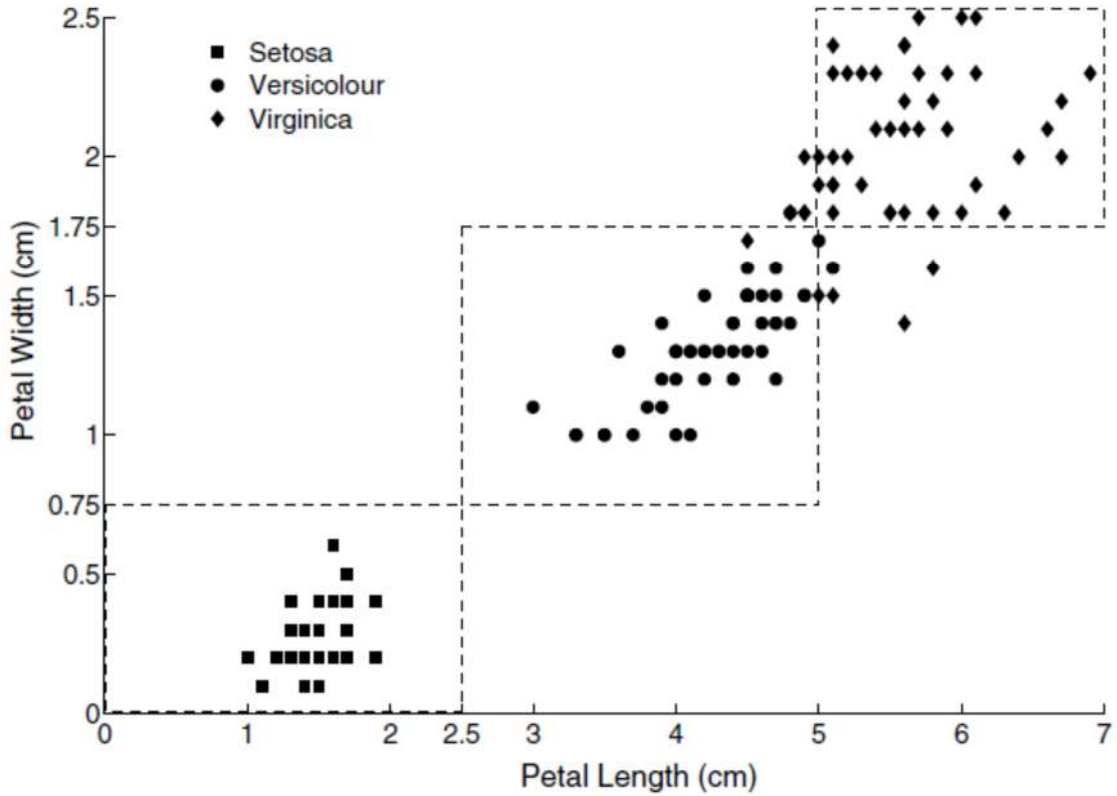


The goal of both tasks is to learn a model that minimizes the error between the predicted and true values of the target variable

Data Mining Tasks

Classification

Example :Predicting the Type of a Flower



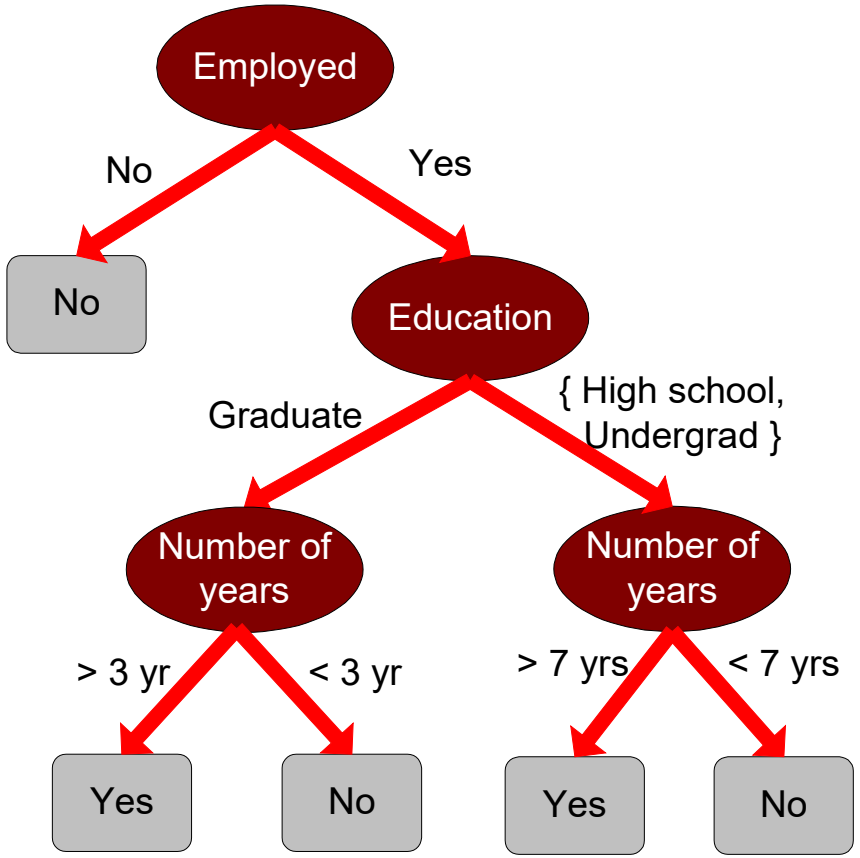
Data Mining Tasks

Example :Predicting credit worthiness

Class

Tid	Employed	Level of Education	# years at present address	Credit Worthy
1	Yes	Graduate	5	Yes
2	Yes	High School	2	No
3	No	Undergrad	1	No
4	Yes	High School	10	Yes
…	…	…	…	…

Goal: Find a model for class attribute as a function of the values of other attributes

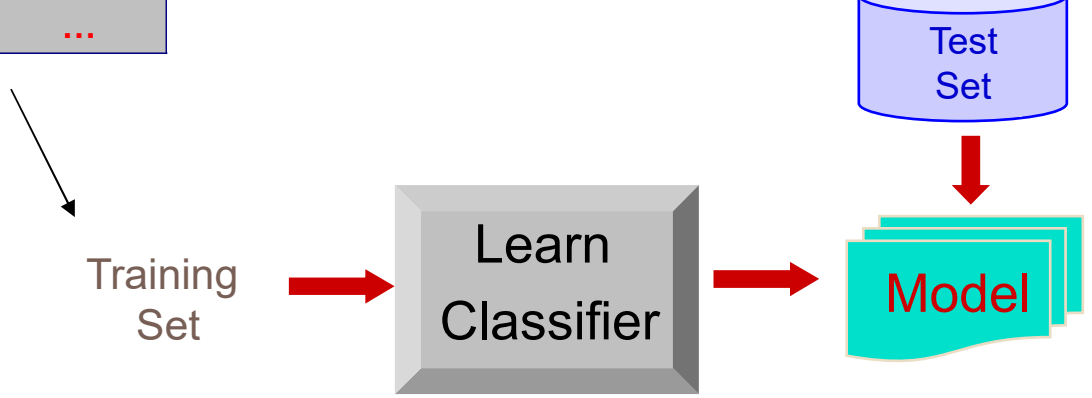


Data Mining Tasks

categorical categorical quantitative class

Tid	Employed	Level of Education	# years at present address	Credit Worthy
1	Yes	Graduate	5	Yes
2	Yes	High School	2	No
3	No	Undergrad	1	No
4	Yes	High School	10	Yes
…	…	…	…	…

Tid	Employed	Level of Education	# years at present address	Credit Worthy
1	Yes	Undergrad	7	?
2	No	Graduate	3	?
3	Yes	High School	2	?
…	…	…	…	…



Examples of Classification Task

- Classifying credit card transactions as legitimate or fraudulent
- Classifying land covers (water bodies, urban areas, forests, etc.) using satellite data
- Categorizing news stories as finance, weather, entertainment, sports, etc
- Identifying intruders in the cyberspace
- Predicting tumor cells as benign or malignant
- Predicting class of sky objects

Data Mining Tasks

Regression

Predict a value of a given continuous valued variable based on the values of other variables, assuming a linear or nonlinear model of dependency.

Examples:

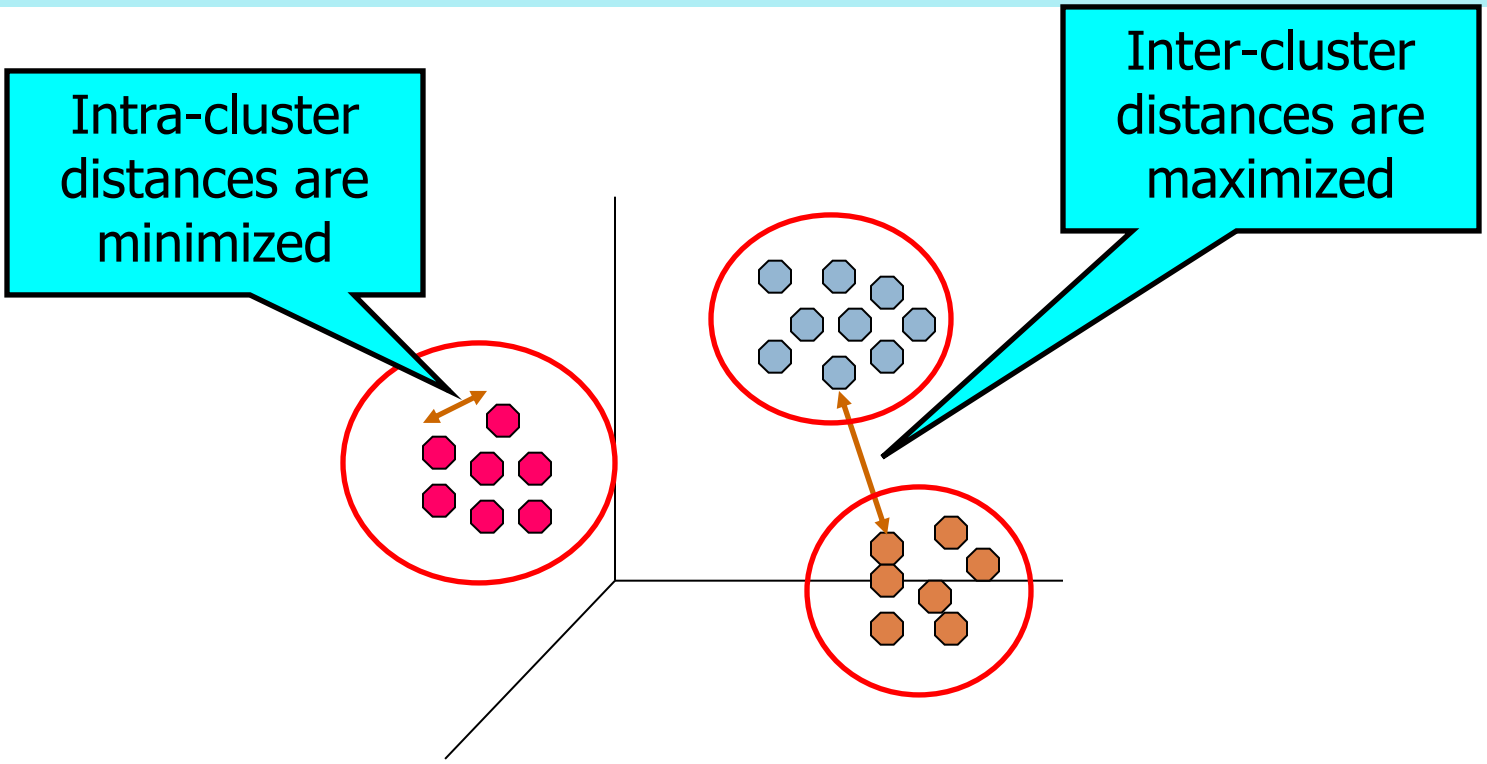
- Predicting sales amounts of new product based on advertising expenditure.
- Predicting wind velocities as a function of temperature, humidity, air pressure, etc.
- Time series prediction of stock market indices.

Clustering

Core Data Mining Tasks

2.Clustering

Goal : find groups of closely related observations so that observations that belong to the same cluster are more similar to each other than observations that belong to other clusters.



Clustering

Example (Market Segmentation)

Goal: subdivide a market into distinct subsets of customers where any subset may conceivably be selected as a market target to be reached with a distinct marketing mix.

Approach:

- Collect different attributes of customers based on their geographical and lifestyle related information.
- Find clusters of similar customers.
- Measure the clustering quality by observing buying patterns of customers in same cluster vs. those from different clusters.

Clustering

Example (Document Clustering)

Goal: To find groups of documents that are similar to each other based on the important terms appearing in them.

Approach: To identify frequently occurring terms in each document. Form a similarity measure based on the frequencies of different terms. Use it to cluster.

Clustering

Example (Document Clustering)

Article	Words
1	dollar: 1, industry: 4, country: 2, loan: 3, deal: 2, government: 2
2	machinery: 2, labor: 3, market: 4, industry: 2, work: 3, country: 1
3	job: 5, inflation: 3, rise: 2, jobless: 2, market: 3, country: 2, index: 3
4	domestic: 3, forecast: 2, gain: 1, market: 2, sale: 3, price: 2
5	patient: 4, symptom: 2, drug: 3, health: 2, clinic: 2, doctor: 2
6	pharmaceutical: 2, company: 3, drug: 2, vaccine: 1, flu: 3
7	death: 2, cancer: 4, drug: 3, public: 4, health: 3, director: 2
8	medical: 2, cost: 3, increase: 2, patient: 2, health: 3, care: 1

Association Rules

Core Data Mining Tasks

3. Association Rules

Goal: discover patterns that describe strongly associated features in the data

Examples

- ✓ Finding groups of genes that have related functionality
- ✓ Identifying Web pages that are accessed together
- ✓ Market-basket analysis

Association Rules

Example: Market Basket Analysis

Transaction ID	Items
1	{Bread, Butter, Diapers, Milk}
2	{Coffee, Sugar, Cookies, Salmon}
3	{Bread, Butter, Coffee, Diapers, Milk, Eggs}
4	{Bread, Butter, Salmon, Chicken}
5	{Eggs, Bread, Butter}
6	{Salmon, Diapers, Milk}
7	{Bread, Tea, Sugar, Eggs}
8	{Coffee, Sugar, Chicken, Eggs}
9	{Bread, Diapers, Milk, Salt}
10	{Tea, Eggs, Cookies, Diapers, Milk}

$$\{\text{Diapers}\} \longrightarrow \{\text{Milk}\}$$

Anomaly Detection

Core Data Mining Tasks

4. Anomaly detection

Goal: identifying observations whose characteristics are significantly different from the rest of the data

anomalies or outliers

Examples:

- Network Intrusion Detection
- Identify anomalous behavior from sensor networks for monitoring and surveillance.
- Detecting changes in the global forest cover.

Contents:

- Introduction
- Data (Types of Data, Data Quality, Data Preprocessing)
- Similarity and Distance
- Exploring Data (Summary Statistics, visualization)
- Regression
- Classification (decision tree, KNN, Bayesian methods, SVM, Ensemble methods, Evaluation)
- Clustering (K-means, Hierarchical Clustering, Density based Clustering, Cluster Validation, Fuzzy Clustering, Mixture Models, Spectral clustering)
- Anomaly Detection

DATA

Content

- Attributes and Objects
- Types of Attributes
- Types of Data
- Data Quality
- Similarity and Distance
- Data Preprocessing

Data

Database: collection of **data objects**

record, point, vector, instance, point, event, case, ***sample, observation****, entity*

data objects are described by a number of **attributes** capture the basic characteristics of an object

variable, characteristic, field, ***feature, dimension***

Data

Attributes

Objects

Tid	Refund	Marital Status	Taxable Income	Cheat
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

Different Types of Attributes

Nominal

The values of a nominal attribute are just different Names

Examples: ID numbers, eye color

Ordinal

The values of an ordinal attribute provide enough information to order objects.

Examples: rankings (e.g., taste of potato chips on a scale from 1-10), grades, height {tall, medium, short}

Interval

For interval attributes, the differences between values are meaningful, i.e., a unit of measurement exists.

Examples: calendar dates, temperatures in Celsius or Fahrenheit.

Ratio

Both differences and ratios are meaningful.

Examples: temperature in Kelvin, length, counts

Different Types of Attributes

The type of an attribute depends on which of the following properties/operations it possesses:

Distinctness:	$= \neq$
Order:	$< >$
Differences are meaningful :	$+ -$
Ratios are meaningful	$* /$

Nominal attribute: distinctness
Ordinal attribute: distinctness & order
Interval attribute: distinctness, order & meaningful differences
Ratio attribute: all 4 properties/operations

Different Types of Attributes

	Attribute Type	Description	Examples	Operations
Categorical Qualitative	Nominal	Nominal attribute values only distinguish. (=, ≠)	employee ID numbers, eye color, {*male, female*}	mode, entropy, contingency correlation, $\chi 2$ test
	Ordinal	Ordinal attribute values also order objects. (<, >)	hardness of minerals, {*good, better, best*}, grades, street numbers	median, percentiles, rank correlation, run tests, sign tests
Numeric Quantitative	Interval	For interval attributes, differences between values are meaningful. (+, -)	calendar dates, temperature in Celsius or Fahrenheit	mean, standard deviation, Pearson's correlation, *t* and *F* tests
	Ratio	For ratio variables, both differences and ratios are meaningful. (*, /)	temperature in Kelvin, monetary quantities, counts, age, length	geometric mean, harmonic mean, percent variation

Different Types of Attributes

Discrete Attribute

- Has only a finite or countably infinite set of values
- Examples: counts, or the set of words in a collection of documents
- Often represented as integer variables.
- Note: binary attributes are a special case of discrete attributes

Continuous Attribute

- Has real numbers as attribute values
- Examples: temperature, height, or weight.

Data

Asymmetric Attributes

only presence—a non-zero attribute value—is regarded as important

- Students and courses
- Words present in documents
- Items present in customer transactions

Asymmetric attributes typically arise from objects that are sets

General Characteristics of Data Sets

- **Dimensionality**
- **Sparsity**
- **Resolution**

Types of Data Sets

Types of Data Sets

1. Record Data

2. Graph-Based Data

3. Ordered Data

Data that consists of a collection of records, each of which consists of a fixed set of attributes

1.1 Data Matrix

1.2 Document Data

1.3 Transaction or Market Basket Data

Types of Data Sets

1.1 Data Matrix

- data objects have the same fixed set of numeric attributes
- data objects are points in a multi-dimensional space
- each dimension represents a distinct attribute
- represented by an *m* by *n* matrix

Data matrix or pattern matrix

Projection of x Load	Projection of y load	Distance	Load	Thickness
10.23	5.27	15.22	2.7	1.2
12.65	6.25	16.22	2.2	1.1

Types of Data Sets

1.2 Document Data

Each document becomes a 'term' vector

- Each term is a component (attribute) of the vector
- The value of each component is the number of times the corresponding term occurs in the document.

	team	coach	play	ball	score	game	win	lost	timeout	season
Document 1	3	0	5	0	2	6	0	2	0	2
Document 2	0	7	0	2	1	0	0	3	0	0
Document 3	0	1	0	0	1	2	2	0	3	0

Types of Data Sets

1.3 Transaction or Market Basket Data

A special type of record data, where Each record (transaction) involves a set of items.

TID	*Items*
1	Bread, Coke, Milk
2	Beer, Bread
3	Beer, Coke, Diaper, Milk
4	Beer, Bread, Diaper, Milk
5	Coke, Diaper, Milk

Types of Data Sets

2. Graph-Based Data

2.1 graph captures relationships among data objects
2.2 data objects themselves are represented as graphs.

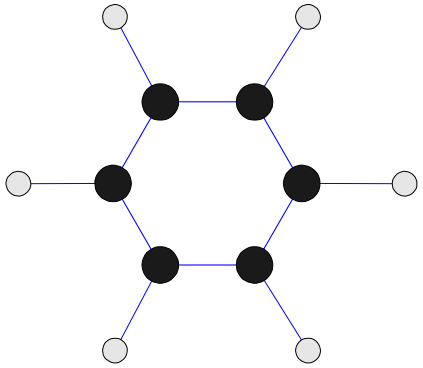
2.1 Data with Relationships among Objects

- relationships among objects frequently convey important information.
- data objects are mapped to nodes
- relationships among objects are captured by the links between objects

Types of Data Sets

2.2 Data with Objects That Are Graphs

objects contain subobjects that have relationships



Types of Data Sets

3. Ordered Data

the attributes have relationships that involve order in time or space

3.1 Sequential (Temporal) Data

each record has a time associated with it

Time	Customer	Items Purchased
t1	C1	A, B
t2	C3	A, C
t2	C1	C, D
t3	C2	A, D
t4	C2	E
t5	C1	A, E

Customer	Time and Items Purchased
C1	(t1: A,B) (t2:C,D) (t5:A,E)
C2	(t3: A, D) (t4: E)
C3	(t2: A, C)

Types of Data Sets

3.2 Sequence Data

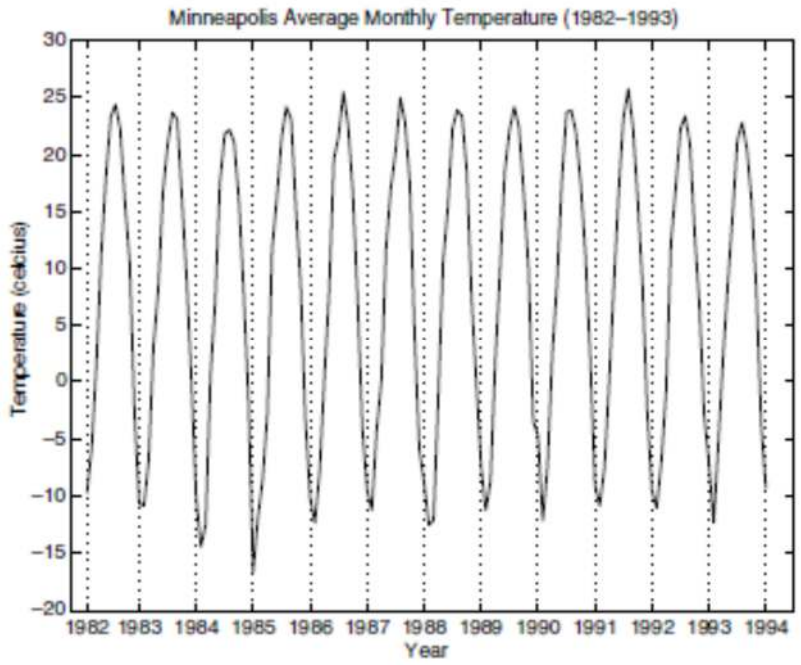
- sequence of words or letters
- no time stamps
- Positions in an ordered sequence.

```
GGTTCCGCCTTCAGCCCCGCGCC
CGCAGGGCCCGCCCCGCGCCGTC
GAGAAGGGCCCGCCTGGCGGGCG
GGGGGAGGCGGGGCCGCCCGAGC
CCAACCGAGTCCGACCAGGTGCC
CCCTCTGCTCGGCCTAGACCTGA
GCTCATTAGGCGGCAGCGGACAG
GCCAAGTAGAACACGCGAAGCGC
TGGGCTGCCTGCTGCGACCAGGG
```

Types of Data Sets

3.3 Time Series Data

- each record is a **time series**
- a series of measurements taken over time



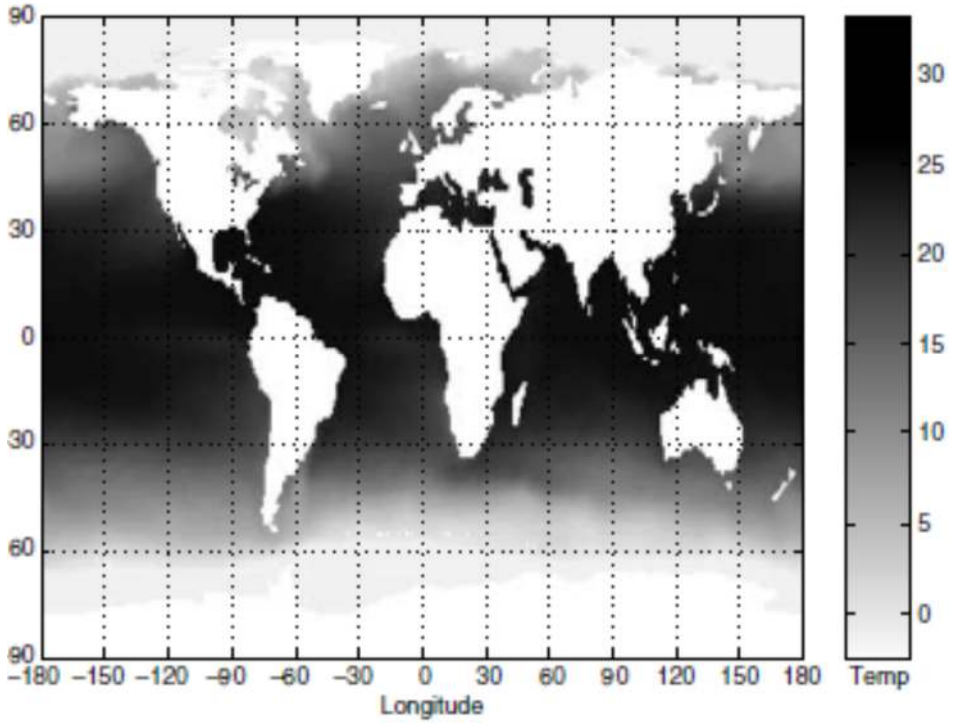
temporal autocorrelation

Types of Data Sets

3.4 Spatial Data

spatial autocorrelation

spatio-temporal data



Data Quality

Data quality

Poor data quality negatively affects many data processing efforts

(1) correction of data quality
(2) use of algorithms that can tolerate poor data quality

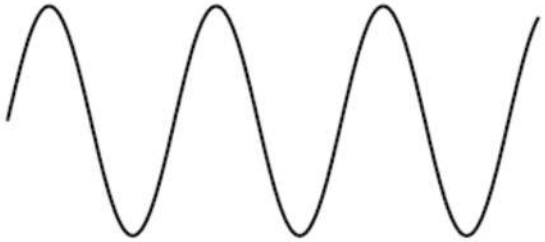
Examples of data quality problems:

- Noise and outliers
- Missing values
- Duplicate data
- Wrong data

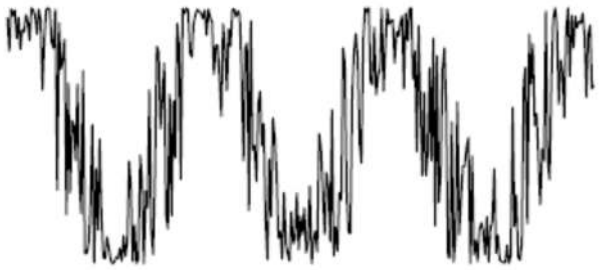
Noise

Noise is the random component of a measurement error

- For objects, noise is an extraneous object
- For attributes, noise refers to modification of original values

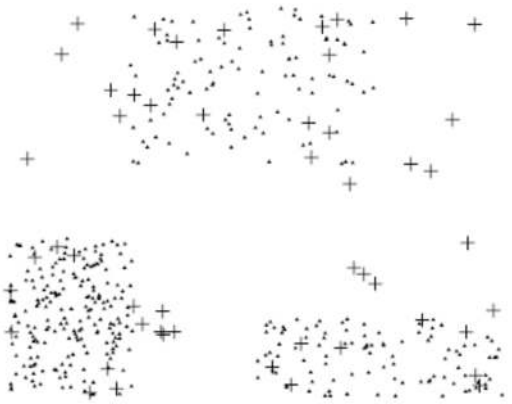


(a) Time series.



(b) Time series with noise.

(a) Three groups of points.



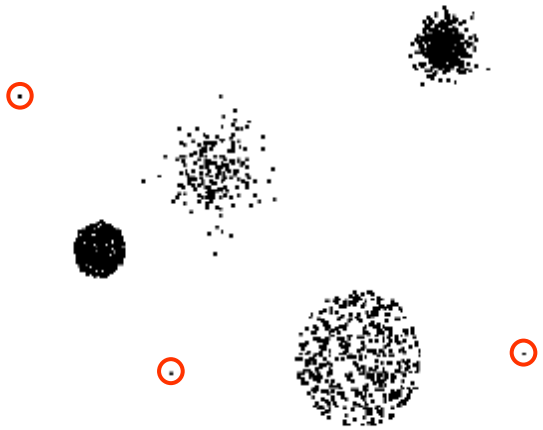
(b) With noise points (+) added.

signal processing can frequently be used to reduce noise

Outlier

(1) data objects that, in some sense, have characteristics that are different from most of the other data objects in the data set
(2) values of an attribute that are unusual with respect to the typical values for that attribute.

anomalous objects or values



Outlier

Case 1: Outliers are noise that interferes with data analysis

Case 2: Outliers are the goal of our analysis

- Credit card fraud
- Intrusion detection

Missing Values

Reasons for missing values

- Information is not collected (e.g., people decline to give their age and weight)
- Attributes may not be applicable to all cases (e.g., annual income is not applicable to children)

Handling missing values

- Eliminate data objects or variables
- Estimate missing values
  - Example: time series of temperature
  - Example: similar data points
- Ignore the missing value during analysis

Data Preprocessing

Data Preprocessing

- Aggregation
- Sampling
- Dimensionality Reduction
- Feature subset selection
- Feature creation
- Discretization and Binarization
- Attribute Transformation

Aggregation

Combining two or more attributes (or objects) into a single attribute (or object)

Data reduction

- Reduce the number of attributes or objects

Change of scale

- Cities aggregated into regions, states, countries, etc.
- Days aggregated into weeks, months, or years

More “stable” data

- Aggregated data tends to have less variability

Sampling

- Sampling is a commonly used approach for selecting a subset of the data objects to be analyzed
- Processing the entire set of data of interest is too expensive or time consuming.

Effective Sampling

- ✓ Using a sample will work almost as well as using the entire data set, if the sample is representative
- ✓ A sample is representative if it has approximately the same properties (of interest) as the original set of data

Simple Random Sampling

There is an equal probability of selecting any particular item

(a) 8000 points (b) 2000 points (c) 500 points

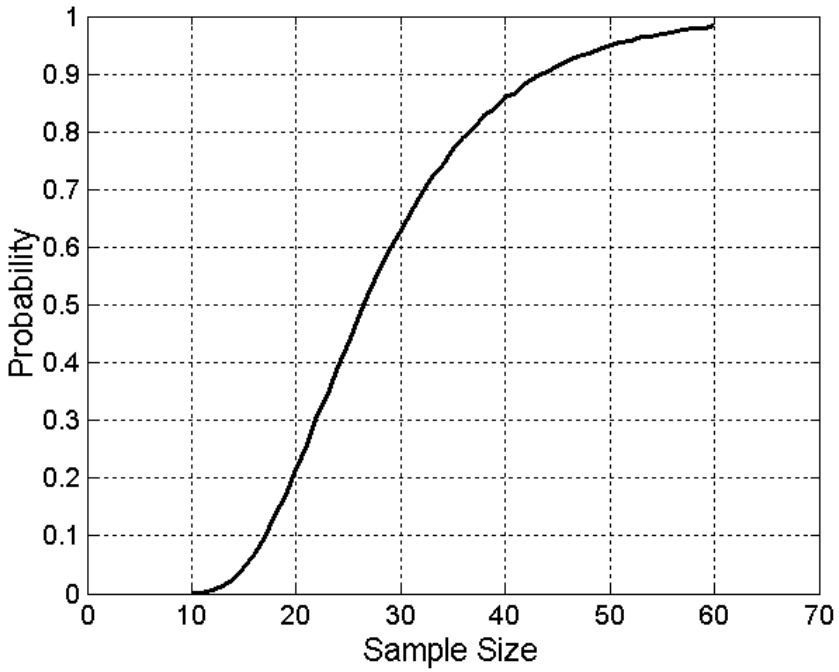
Sampling Method & size?

Sampling

- Sampling without replacement
  - ✓ As each item is selected, it is removed from the population
- Sampling with replacement
  - ✓ Objects are not removed from the population as they are selected for the sample.
  - ✓ In sampling with replacement, the same object can be picked up more than once

Sampling

- What sample size is necessary to get at least one object from each of 10 equal-sized groups.



- Stratified sampling
  - Split the data into several partitions; then draw random samples from each partition

Dimensionality Reduction

many data mining algorithms work better if the dimensionality is lower

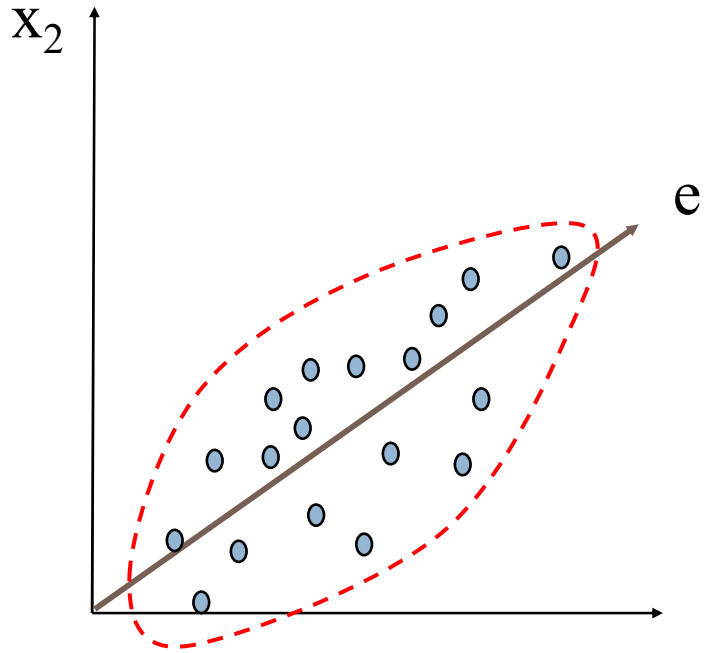
- ❖ eliminate irrelevant features and reduce noise
- ❖ curse of dimensionality
- ❖ more easily visualized
- ❖ Reduce amount of time and memory required by data mining algorithms

1) creating new features that are a combination of the old attributes
2) selecting features **feature subset selection** or **feature selection**

Curse of Dimensionality

- many types of data analysis become significantly harder as the dimensionality of the data increases
- When dimensionality increases, data becomes increasingly sparse in the space that it occupies

Dimensionality Reduction: PCA



Feature Selection

Another way to reduce dimensionality of data

Redundant features

- Duplicate much or all of the information contained in one or more other attributes
- Example: purchase price of a product and the amount of sales tax paid

Irrelevant features

- Contain no information that is useful for the data mining task at hand
- Example: students' ID is often irrelevant to the task of predicting students' GPA

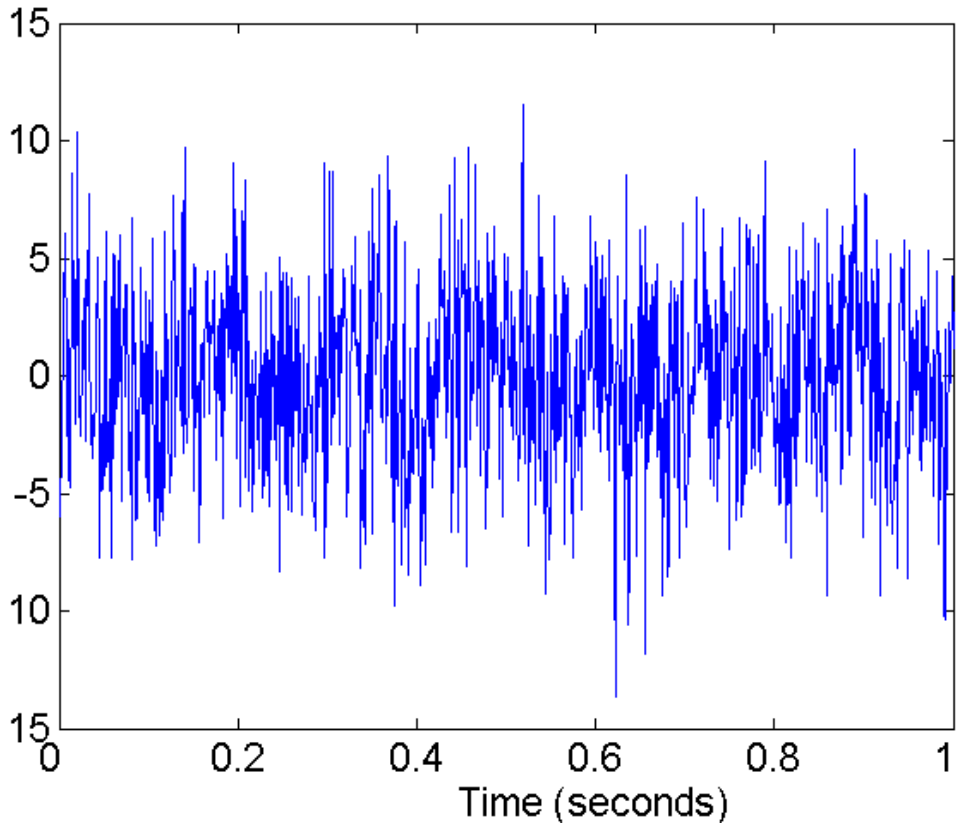
Feature Creation

Create new attributes that can capture the important information in a data set much more efficiently than the original attributes

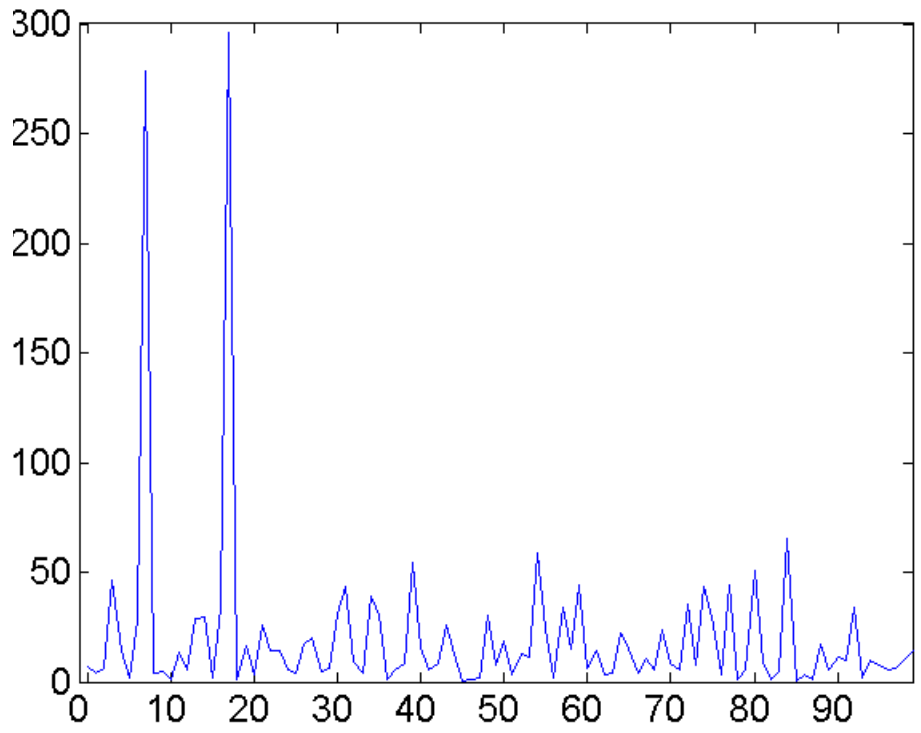
Feature extraction/construction

Mapping data to new space : different view of the data

Fourier and wavelet transform



Two Sine Waves + Noise



Frequency

Discretization

Discretization is the process of converting a continuous attribute into an ordinal attribute

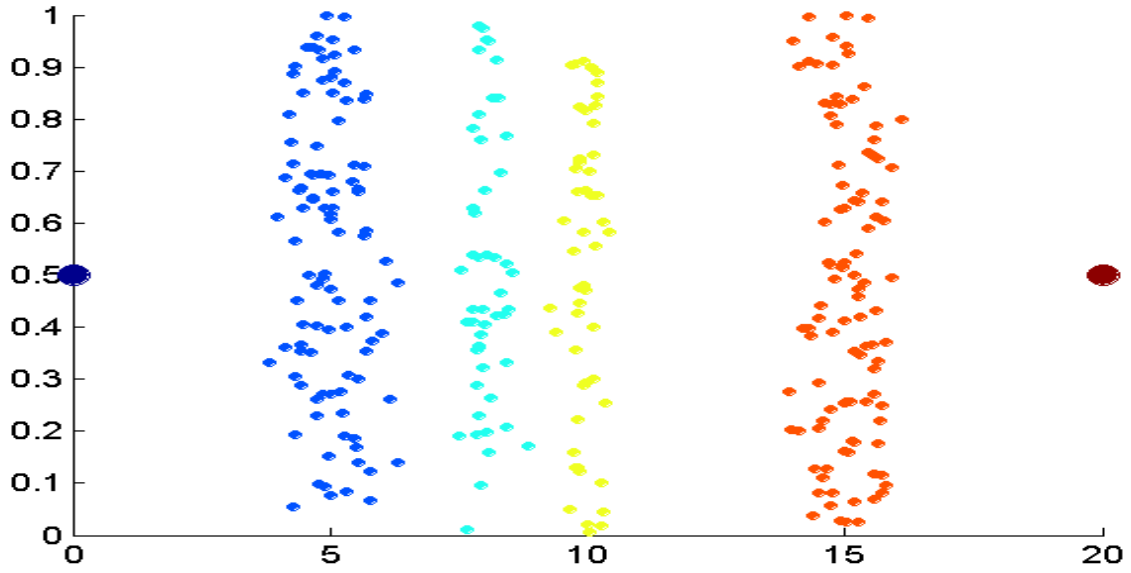
Binarization

Categorical Value	Integer Value	x_1	x_2	x_3
awful	0	0	0	0
poor	1	0	0	1
OK	2	0	1	0
good	3	0	1	1
great	4	1	0	0

Discretization

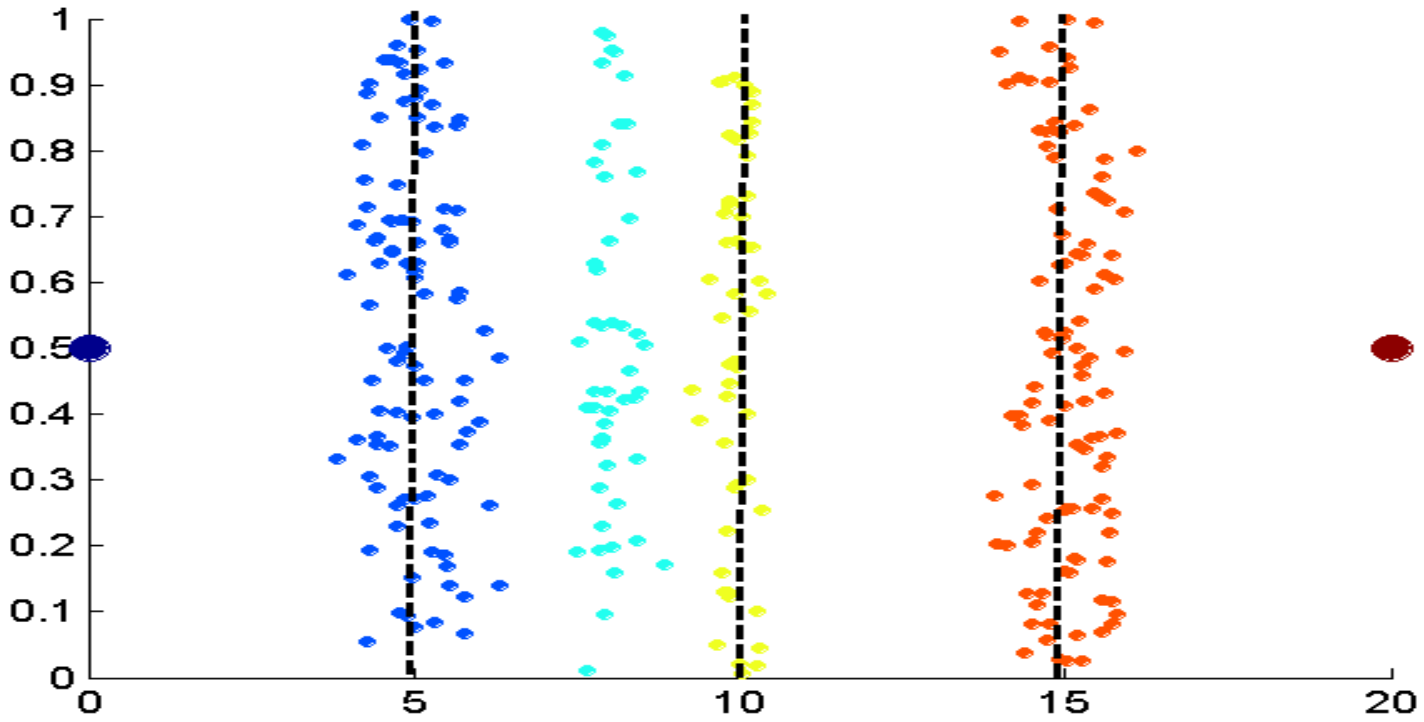
Unsupervised discretization: find breaks in the data values
Supervised discretization: Use class labels to find breaks

Example: Discretization Without Using Class Labels



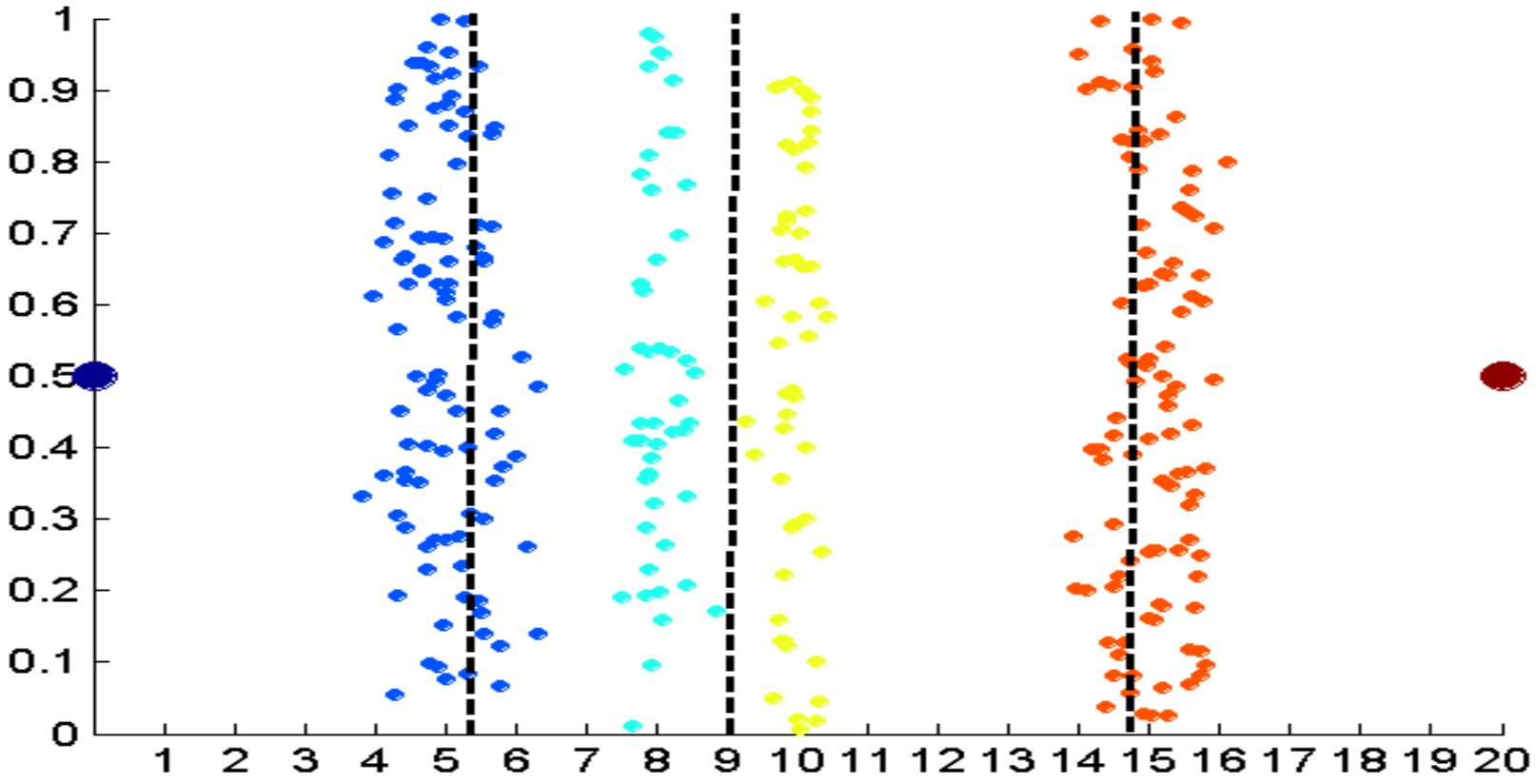
Data consists of four groups of points and two outliers. Data is one-dimensional, but a random y component is added to reduce overlap

Discretization



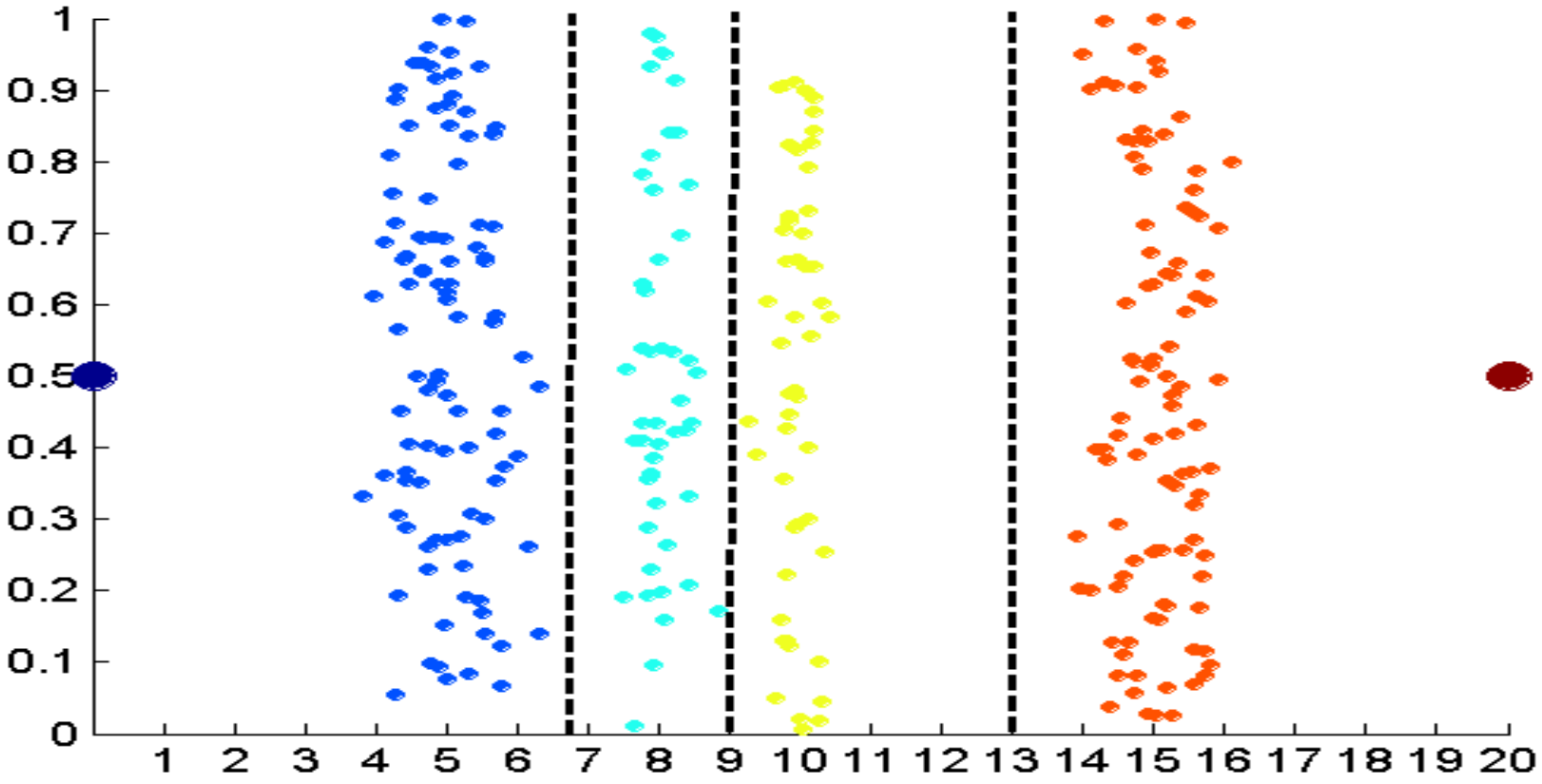
Equal interval width approach used to obtain 4 values

Discretization



Equal frequency approach used to obtain 4 values

Discretization



K-means approach to obtain 4 values.

Attribute Transformation

- An attribute transform is a function that maps the entire set of values of a given attribute to a new set of replacement values such that each old value can be identified with one of the new values
  - Simple functions: x^k, log(x), e^x, |x|
  - Normalization
    - Refers to various techniques to adjust to differences among attributes in terms of mean, variance, range
    - Take out unwanted, common signal, e.g., seasonality
  - In statistics, standardization refers to subtracting off the means and dividing by the standard deviation

Similarity and Dissimilarity Measures

Similarity and Dissimilarity Measures

Similarity measure

- Numerical measure of how alike two data objects are.
- Is higher when objects are more alike.
- Often falls in the range [0,1]

Dissimilarity measure

- Numerical measure of how different two data objects are
- Lower when objects are more alike
- Minimum dissimilarity is often 0
- Upper limit varies

Proximity refers to a similarity or dissimilarity

1. objects having only one simple attribute
2. objects with multiple attributes

Similarity/Dissimilarity for Simple Attributes

The following table shows the similarity and dissimilarity between two objects, x and y, with respect to a single attribute.

Attribute Type	**Dissimilarity**	**Similarity**
Nominal	$d = \begin{cases} 0 & \text{if } x = y \\ 1 & \text{if } x \neq y \end{cases}$	$s = \begin{cases} 1 & \text{if } x = y \\ 0 & \text{if } x \neq y \end{cases}$
Ordinal	$d = \lvert x - y \rvert / (n-1)$ (values mapped to integers 0 to $n-1$, where n is the number of values)	$s = 1 - d$
Interval or Ratio	$d = \lvert x - y \rvert$	$s = -d$, $s = \frac{1}{1+d}$, $s = e^{-d}$, $s = 1 - \frac{d - min\_d}{max\_d - min\_d}$

Dissimilarities between Data Objects

various kinds of dissimilarities

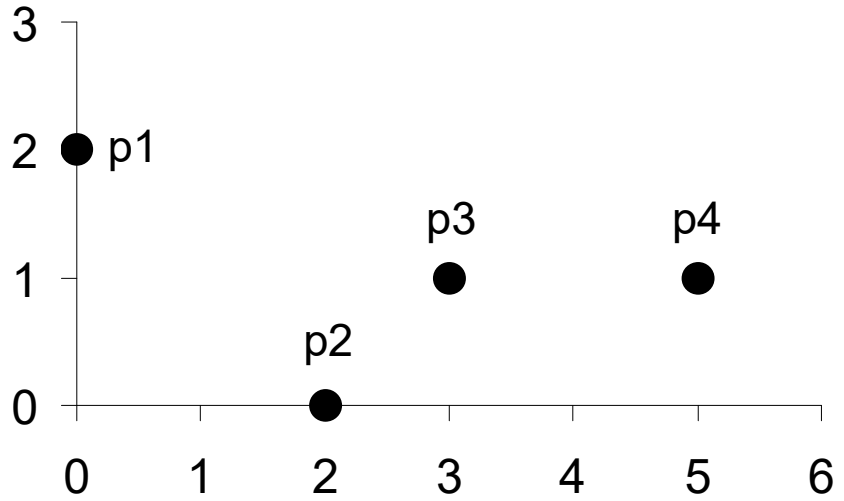
1. distances, which are dissimilarities with certain properties
2. provide examples of more general kinds of dissimilarities

Euclidean Distance

$$d(\mathbf{x}, \mathbf{y}) = \sqrt{\sum_{k=1}^{n} (x_k - y_k)^2}$$

where *n* is the number of dimensions (attributes) and x_k and y_k are, respectively, the k^{th} attributes (components) or data objects $\mathbf{x}$ and $\mathbf{y}$.

Distances:example



point	x	y
p1	0	2
p2	2	0
p3	3	1
p4	5	1

	p1	p2	p3	p4
p1	0	2.828	3.162	5.099
p2	2.828	0	1.414	3.162
p3	3.162	1.414	0	2
p4	5.099	3.162	2	0

Distance Matrix

Distances

Minkowski Distance

is a generalization of Euclidean Distance

$$d(\mathbf{x}, \mathbf{y}) = \left(\sum_{k=1}^{n} |x_k - y_k|^r \right)^{1/r}$$

Where r is a parameter, n is the number of dimensions (attributes) and x_k and y_k are, respectively, the k^{th} attributes (components) or data objects $\boldsymbol{x}$ and $\boldsymbol{y}$.

Distances

Minkowski Distance

- $r = 1$. City block (Manhattan, taxicab, L_1 norm) distance.
  - A common example of this is the Hamming distance, which is just the number of bits that are different between two binary vectors
- $r = 2$. Euclidean distance
- $r \to \infty$. "supremum" (L_{max} norm, L_∞ norm) distance.
  - This is the maximum difference between any component of the vectors

Distances:Example

point	x	y
p1	0	2
p2	2	0
p3	3	1
p4	5	1

L1	p1	p2	p3	p4
p1	0	4	4	6
p2	4	0	2	4
p3	4	2	0	2
p4	6	4	2	0

L2	p1	p2	p3	p4
p1	0	2.828	3.162	5.099
p2	2.828	0	1.414	3.162
p3	3.162	1.414	0	2
p4	5.099	3.162	2	0

L_∞	p1	p2	p3	p4
p1	0	2	3	5
p2	2	0	1	3
p3	3	1	0	2
p4	5	3	2	0

Distance Matrix

Standardization and Correlation for Distance Measures

A generalization of Euclidean distance, the **Mahalanobis distance** when attributes are correlated, have different ranges of values

$$\mathbf{mahalanobis}(\mathbf{x}, \mathbf{y}) = (\mathbf{x} - \mathbf{y})^T \Sigma^{-1} (\mathbf{x} - \mathbf{y})$$

Σ is the covariance matrix

ij_{th} entry is the covariance of the i_{th} and j_{th} attributes

Distances

Distances, such as the Euclidean distance, have some well known properties.

1. $d(x, y) \geq 0$ for all x and y
2. $d(x, y) = 0$ only if $x = y$
3. $d(x, y) = d(y, x)$ for all x and y.
4. $d(x, z) \leq d(x, y) + d(y, z)$ for all points x, y, and z.

where d(x, y) is the distance (dissimilarity) between points (data objects), x and y.

Non-metric Dissimilarities

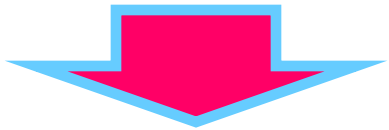
some dissimilarities do not satisfy one or more of the metric properties

Example: Non-metric Dissimilarities: Set Differences

Given two sets *A* and *B*, *A* − *B* is the set of elements of A that are not in *B*.

$$A = \{1, 2, 3, 4\} \text{ and } B = \{2, 3, 4\} \qquad A - B = \{1\} \quad B - A = \emptyset$$

$$d(A, B) = size(A - B)$$



$$d(A, B) = size(A - B) + size(B - A)$$

Similarities between Data Objects

Similarities, have some typical properties.

1. s(x, y) = 1 (or maximum similarity) only if x = y.

2. s(x, y) = s(y, x) for all x and y. (Symmetry)

where $s(\mathbf{x}, \mathbf{y})$ is the similarity between points (data objects), $\mathbf{x}$ and $\mathbf{y}$.

Similarities between Data Objects

Similarity Between Binary Vectors

p and *q*, have only binary attributes

Compute similarities using the following quantities

f_{01} = the number of attributes where *p* was 0 and *q* was 1

f_{10} = the number of attributes where *p* was 1 and *q* was 0

f_{00} = the number of attributes where *p* was 0 and *q* was 0

f_{11} = the number of attributes where *p* was 1 and *q* was 1

Simple Matching Coefficient

SMC = number of matches / number of attributes

$= (f_{11} + f_{00}) / (f_{01} + f_{10} + f_{11} + f_{00})$

Jaccard Coefficient

J = number of 11 matches / number of non-zero attributes

$= (f_{11}) / (f_{01} + f_{10} + f_{11})$

Similarities between Data Objects

Example: binary similarity

$\mathbf{x}$ = 1 0 0 0 0 0 0 0 0 0

$\mathbf{y}$ = 0 0 0 0 0 0 1 0 0 1

$f_{01} = 2$ (the number of attributes where p was 0 and q was 1)

$f_{10} = 1$ (the number of attributes where p was 1 and q was 0)

$f_{00} = 7$ (the number of attributes where p was 0 and q was 0)

$f_{11} = 0$ (the number of attributes where p was 1 and q was 1)

$$\text{SMC} = (f_{11} + f_{00}) / (f_{01} + f_{10} + f_{11} + f_{00})$$
$$= (0+7) / (2+1+0+7) = 0.7$$

$$J = (f_{11}) / (f_{01} + f_{10} + f_{11}) = 0 / (2 + 1 + 0) = 0$$

Similarities between Data Objects

Cosine Similarity

If $\mathbf{d}_1$ and $\mathbf{d}_2$ are two document vectors, then

$$\cos(\mathbf{d}_1, \mathbf{d}_2) = \langle \mathbf{d}_1, \mathbf{d}_2 \rangle / \|\mathbf{d}_1\| \, \|\mathbf{d}_2\| ,$$

Example:

$$\mathbf{d}_1 = 3\ 2\ 0\ 5\ 0\ 0\ 0\ 2\ 0\ 0$$

$$\mathbf{d}_2 = 1\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 2$$

$$\langle \mathbf{d}_1, \mathbf{d2} \rangle = 3*1 + 2*0 + 0*0 + 5*0 + 0*0 + 0*0 + 0*0 + 2*1 + 0*0 + 0*2 = 5$$

$$\| \mathbf{d}_1 \| = (3*3+2*2+0*0+5*5+0*0+0*0+0*0+2*2+0*0+0*0)^{0.5} = (42)^{0.5} = 6.481$$

$$\| \mathbf{d}_2 \| = (1*1+0*0+0*0+0*0+0*0+0*0+0*0+1*1+0*0+2*2)^{0.5} = (6)^{0.5} = 2.449$$

$$\cos(\mathbf{d}_1, \mathbf{d}_2) = 0.3150$$

Similarities between Data Objects

Correlation

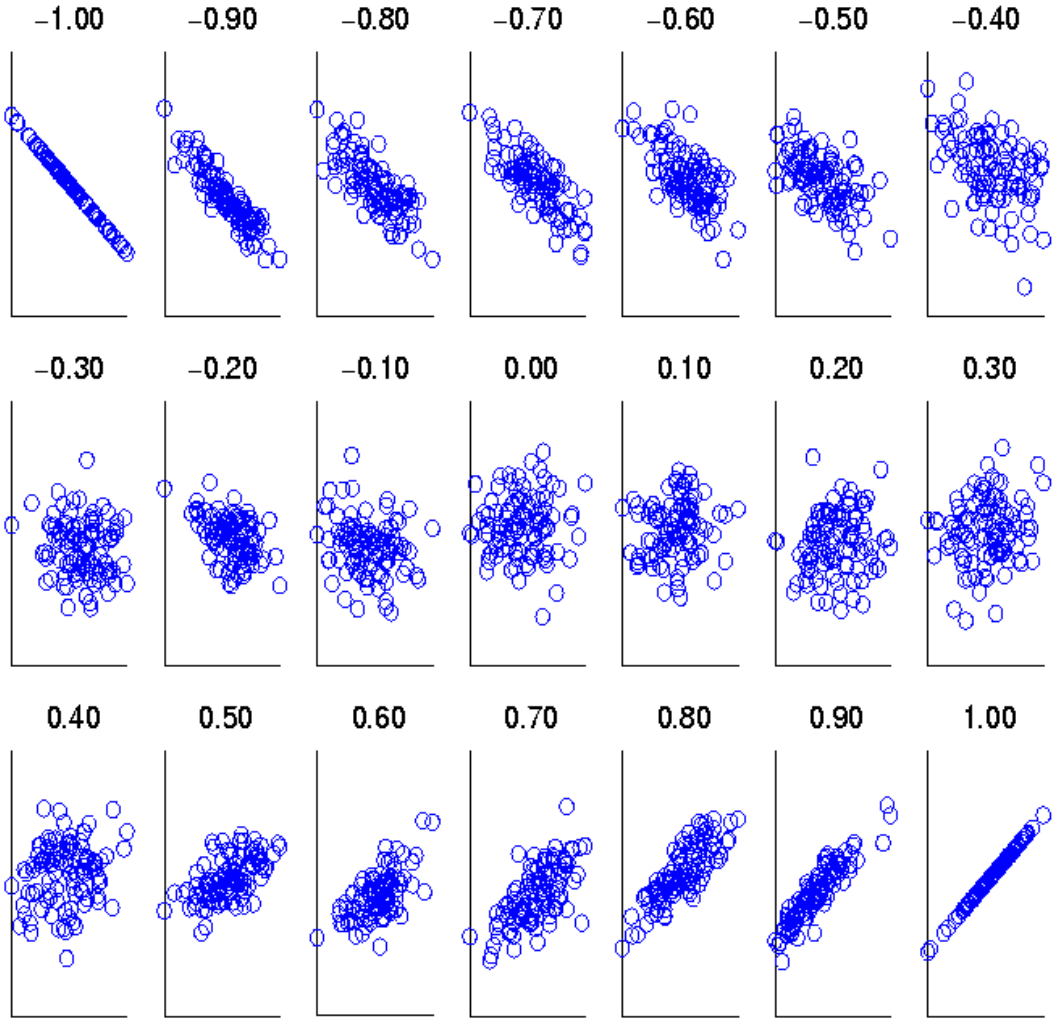
$$\text{corr}(\text{x}, \text{y}) = \frac{\text{covariance}(\text{x}, \text{y})}{\text{standard\_deviation}(\text{x}) * \text{standard\_deviation}(\text{y})} = \frac{s_{xy}}{s_x \; s_y}$$

$$\text{covariance}(\text{x}, \text{y}) = s_{xy} = \frac{1}{n-1} \sum_{k=1}^{n} (x_k - \overline{x})(y_k - \overline{y})$$

$$\overline{x} = \frac{1}{n} \sum_{k=1}^{n} x_k \text{ is the mean of x} \qquad s_x = \sqrt{\frac{1}{n-1} \sum_{k=1}^{n} (x_k - \overline{x})^2}$$

$$\overline{y} = \frac{1}{n} \sum_{k=1}^{n} y_k \text{ is the mean of y} \qquad s_y = \sqrt{\frac{1}{n-1} \sum_{k=1}^{n} (y_k - \overline{y})^2}$$

Similarities between Data Objects



Scatter plots showing the similarity from –1 to 1.

Similarities between Data Objects

Drawback of Correlation : **Non-linear Relationships**

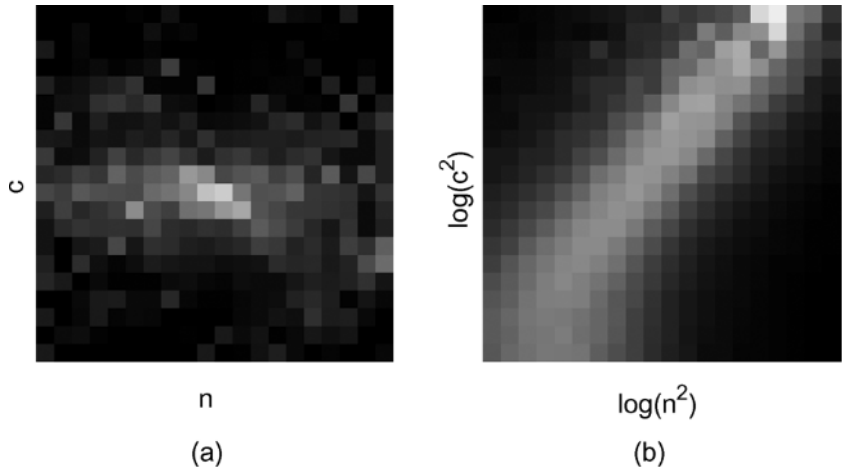
- correlation is 0, no linear relationship between the attributes of the two data objects
- non-linear relationships may still exist.

Example:

$\mathbf{x} = (-3, -2, -1, 0, 1, 2, 3)$
$\mathbf{y} = (9, 4, 1, 0, 1, 4, 9)$
$y_i = x_i^2$
$\text{mean}(\mathbf{x}) = 0, \text{mean}(\mathbf{y}) = 4$
$\text{std}(\mathbf{x}) = 2.16, \text{std}(\mathbf{y}) = 3.74$

$$\text{corr} = (-3)(5)+(-2)(0)+(-1)(-3)+(0)(-4)+(1)(-3)+(2)(0)+3(5) / (6 * 2.16 * 3.74) = 0$$

Similarities between Data Objects



Information theory

- Information theory
- similarity measures
- handle non-linear relationships
- complicated and time intensive to compute
- Information relates to possible outcomes of an event
- information is related the probability of an outcome
- The smaller the probability of an outcome, the more information it provides
- Entropy is the commonly used measure

Entropy

- ✓ a variable (event), X,
- ✓ with n possible values (outcomes), $x_1, x_2 \ldots, x_n$
- ✓ each outcome having probability, $p_1, p_2 \ldots, p_n$
- ✓ the entropy of X , $H(X)$, is given by

$$H(X) = -\sum_{i=1}^{n} p_i \log_2 p_i$$

Entropy is between 0 and $\log_2 n$ and is measured in bits

entropy is a measure of how many bits it takes to represent an observation of X on average

Example:

For a coin with probability p of heads and probability $q = 1 - p$ of tails

$$H = -p \log_2 p - q \log_2 q$$

For $p = 0.5$, $q = 0.5$ (fair coin) $H = 1$

For $p = 1$ or $q = 1$, $H = 0$

Entropy

a number of observations (m) of some attribute, X, e.g., the hair color of students in the class, where there are n different possible values
the number of observation in the i^{th} category is m_i

$$H(X) = -\sum_{i=1}^{n} \frac{m_i}{m} \log_2 \frac{m_i}{m}$$

Hair Color	Count
Black	75
Brown	15
Blond	5
Red	0
Other	5
Total	100

Maximum entropy is $\log_2 5 = 2.3219$

Mutual Information

Information one variable provides about another

Formally, $I(X,Y) = H(X) + H(Y) - H(X,Y)$, where

$H(X,Y)$ is the joint entropy of X and Y,

$$H(X,Y) = -\sum_i \sum_j p_{ij} \log_2 p_{ij}$$

Where p_{ij} is the probability that the i^{th} value of X and the j^{th} value of Y occur together

✓how similar the joint distribution *p(X, Y) is to the factored distribution p(X)p(Y).*

MI is zero iff the variables are independent

MI between *X and Y as the reduction in uncertainty about X after* observing *Y*

Mutual Information example

Student Status	Count
Undergrad	45
Grad	55
Total	100

Grade	Count
A	35
B	50
C	15
Total	100

Student Status	Grade	Count
Undergrad	A	5
Undergrad	B	30
Undergrad	C	10
Grad	A	30
Grad	B	20
Grad	C	5
Total		100

Mutual information of Student Status and Grade = 0.9928 + 1.4406 - 2.2710 = 0.1624

EXPLORING DATA

Data Exploration

- A preliminary exploration of the data to better understand its characteristics.
- Helping to select the right tool for preprocessing or analysis
- In our discussion of data exploration, we focus on
  - Summary statistics
  - Visualization

Iris Sample Data Set

Many of the exploratory data techniques are illustrated with the Iris Plant data set.

Can be obtained from the UCI Machine Learning Repository
http://www.ics.uci.edu/~mlearn/MLRepository.html

- Three flower types (classes):
  - Setosa
  - Virginica
  - Versicolour
- Four (non-class) attributes
  - Sepal width and length
  - Petal width and length

Summary Statistics

Summary statistics are quantities, such as the mean and standard deviation, that capture various characteristics of a potentially large set of values with a single number or a small set of numbers.

Summarized properties include frequency, location and spread

Examples: location - mean

spread - standard deviation

Frequencies and the Mode

The frequency of an attribute value is the percentage of time the value occurs in the data set

- For example, given the attribute 'gender' and a representative population of people, the gender 'female' occurs about 50% of the time.

The mode of a an attribute is the most frequent attribute value

✓ The notions of frequency and mode are typically used with nominal data

Percentiles

For continuous data, the notion of a percentile is more useful.

Given an ordinal or continuous attribute *x* and a number *p* between 0 and 100, the *p*th percentile is a value x_p of x such that *p*% of the observed values of x are less than x_p .

For instance, the 50th percentile is the value $x_{50\%}$ such that 50% of all values of x are less than $x_{50\%}$

$$\min(x) = x_{0\%} \text{ and } \max(x) = x_{100\%}$$

Example

Percentile	Sepal Length	Sepal Width	Petal Length	Petal Width
0	4.3	2.0	1.0	0.1
10	4.8	2.5	1.4	0.2
20	5.0	2.7	1.5	0.2
30	5.2	2.8	1.7	0.4
40	5.6	3.0	3.9	1.2
50	5.8	3.0	4.4	1.3
60	6.1	3.1	4.6	1.5
70	6.3	3.2	5.0	1.8
80	6.6	3.4	5.4	1.9
90	6.9	3.6	5.8	2.2
100	7.9	4.4	6.9	2.5

Measures of Location: Mean and Median

Mean

mean is the most common measure of the location of a set of points.

$\{x_1, \ldots, x_m\}$

attribute values of *x* for *m* objects.

$$\text{mean}(x) = \overline{x} = \frac{1}{m}\sum_{i=1}^{m} x_i$$

mean is very sensitive to outliers

Median

$\{x_{(1)}, \ldots, x_{(m)}\}$

$$\text{median}(x) = \begin{cases} x_{(r+1)} & \text{if } m \text{ is odd, i.e., } m = 2r+1 \\ \frac{1}{2}(x_{(r)} + x_{(r+1)}) & \text{if } m \text{ is even, i.e., } m = 2r \end{cases}$$

trimmed mean

trimmed mean

- ✓ A percentage *p* between 0 and 100 is specified
- ✓ top and bottom (*p*/2)% of the data is thrown out
- ✓ mean is then calculated in the normal way.

Measure	Sepal Length	Sepal Width	Petal Length	Petal Width
mean	5.84	3.05	3.76	1.20
median	5.80	3.00	4.35	1.30
trimmed mean (20%)	5.79	3.02	3.72	1.12

Measures of Spread: Range and Variance

- ✓ Range is the difference between the max and min
- ✓ Variance or standard deviation is the most common measure of the spread of a set of points.

$$\text{variance}(x) = s_x^2 = \frac{1}{m-1}\sum_{i=1}^{m}(x_i - \overline{x})^2$$

sensitive to outliers

absolute average deviation (AAD)

$$\text{AAD}(x) = \frac{1}{m}\sum_{i=1}^{m}|x_i - \overline{x}|$$

median absolute deviation (MAD)

$$\text{MAD}(x) = median\left(\{|x_1 - \overline{x}|, \ldots, |x_m - \overline{x}|\}\right)$$

$$\text{interquartile range}(x) = x_{75\%} - x_{25\%}$$

Multivariate Summary Statistics

if x_i and x_j are the i_{th} and j_{th} attributes, then

$$s_{ij} = \text{covariance}(x_i, x_j)$$

$$\text{covariance}(x_i, x_j) = \frac{1}{m-1} \sum_{k=1}^{m} (x_{ki} - \overline{x_i})(x_{kj} - \overline{x_j})$$

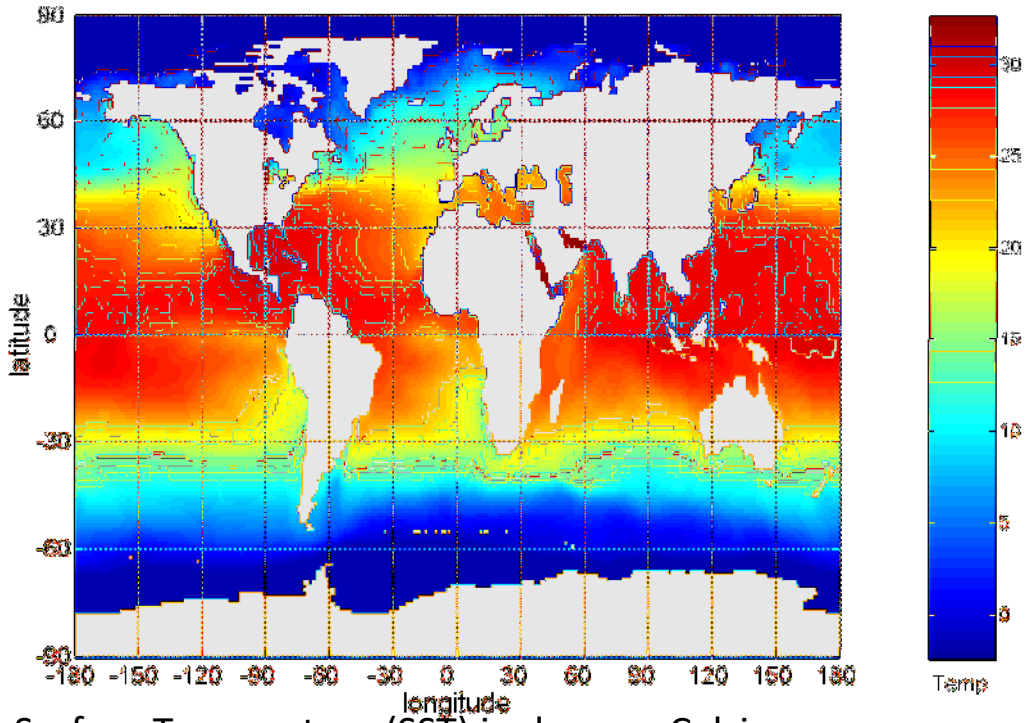
$$r_{ij} = \text{correlation}(x_i, x_j) = \frac{\text{covariance}(x_i, x_j)}{s_i s_j}$$

correlation matrix R

Visualization

Visualization

- ✓ Visualization is the conversion of data into a visual format so that the characteristics of the data and the relationships among data items or attributes can be analyzed or reported
- ✓ people can quickly absorb large amounts of visual information

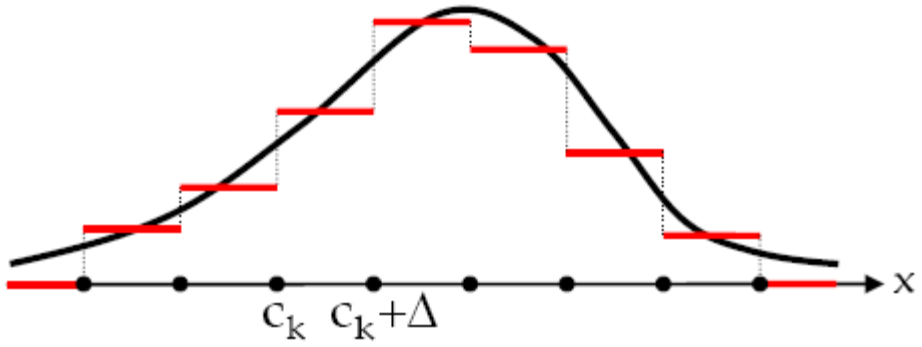


Sea Surface Temperature (SST) in degrees Celsius

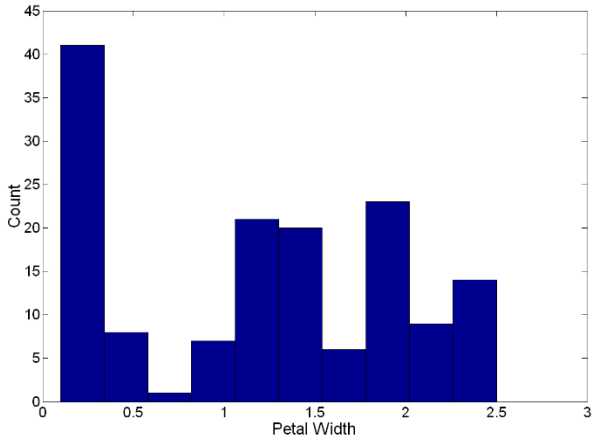
Visualization Techniques: Histograms

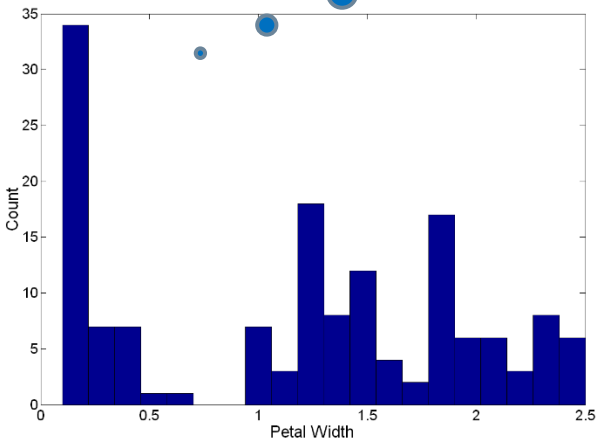
Histogram

- Usually shows the distribution of values of a single variable
- Divide the values into bins and show a bar plot of the number of objects in each bin.
- The height of each bar indicates the number of objects
- Shape of histogram depends on the number of bins

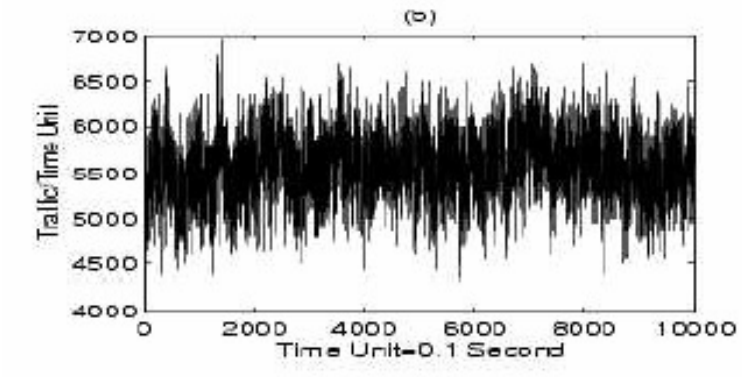


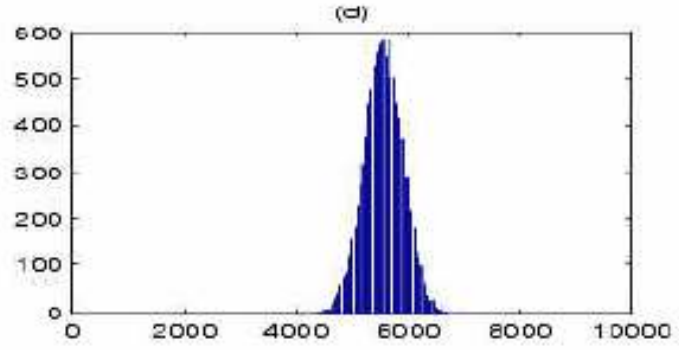
Example: Petal Width (10 and 20 bins, respectively)



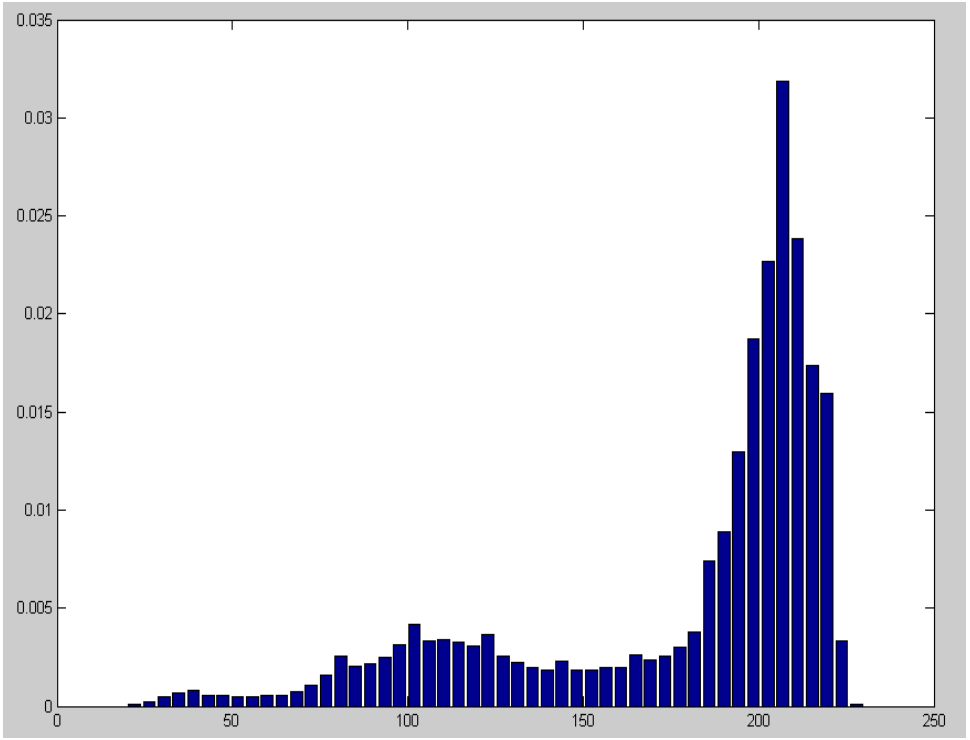


relative frequency histogram


(b)
Traffic/Time Unit
7000
6500
6000
5500
5000
4500
4000
0
2000
4000
6000
8000
10000
Time Unit=0.1 Second


(d)
600
500
400
300
200
100
0
0
2000
4000
6000
8000
10000

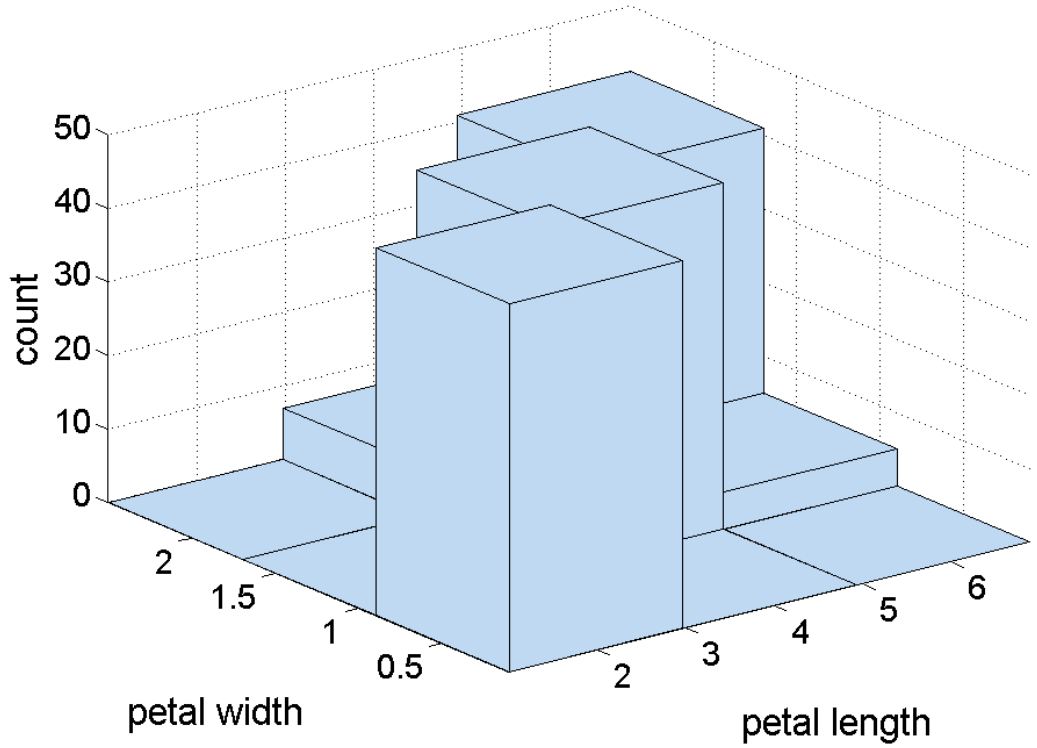

F-16
01568
U.S.AIR FORCE


0.035
0.03
0.025
0.02
0.015
0.01
0.005
0
0
50
100
150
200
250

Two-Dimensional Histograms

Show the joint distribution of the values of two attributes

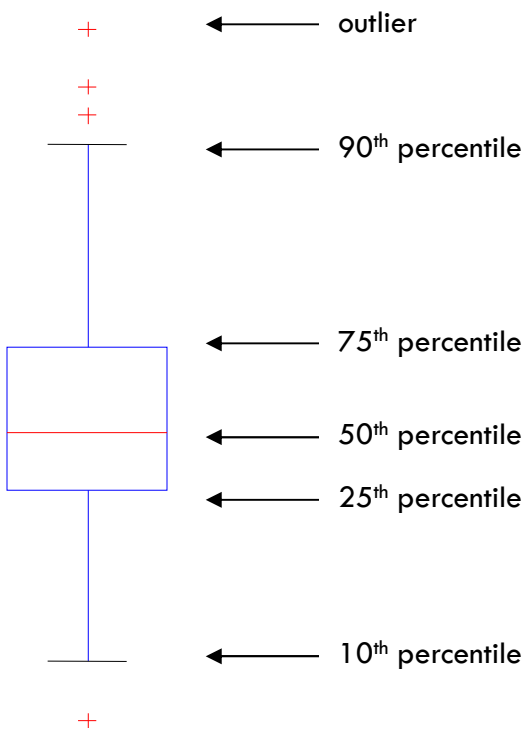
Example: petal width and petal length



Visualization Techniques: Box Plots

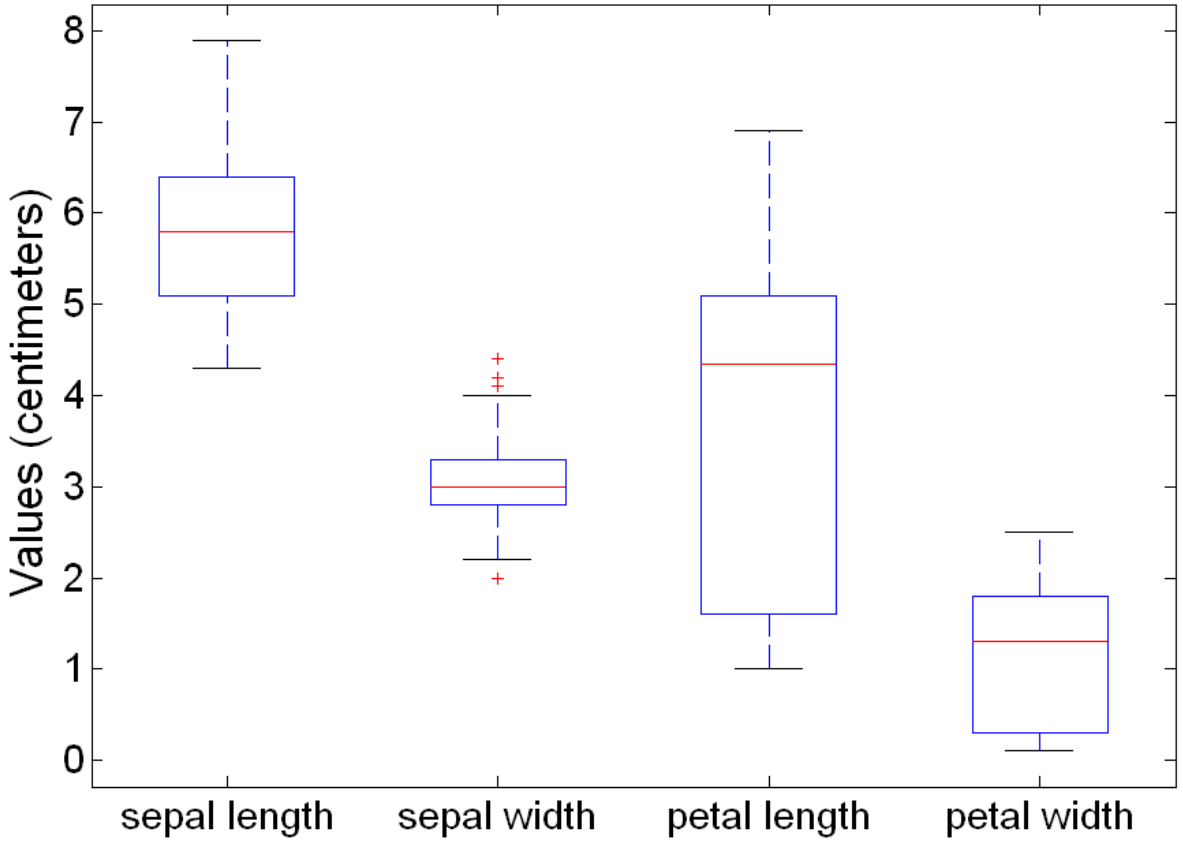
Box Plots
Another way of displaying the distribution of data

Following figure shows the basic part of a box plot



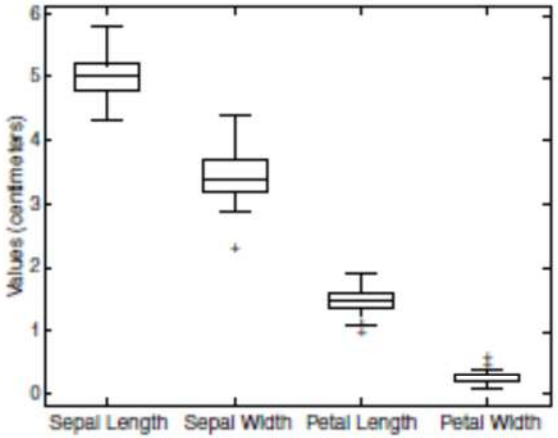
Visualization Techniques: Box Plots

Box plots can be used to compare attributes

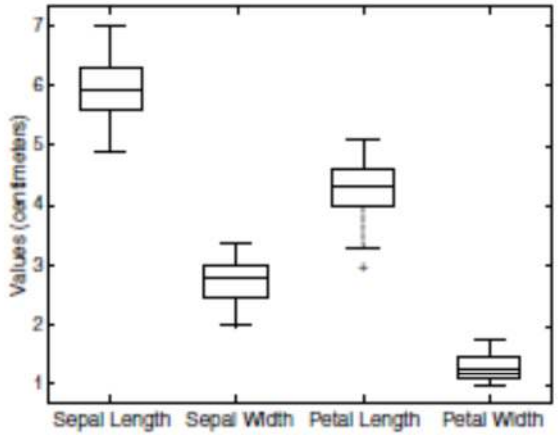


Visualization Techniques: Box Plots

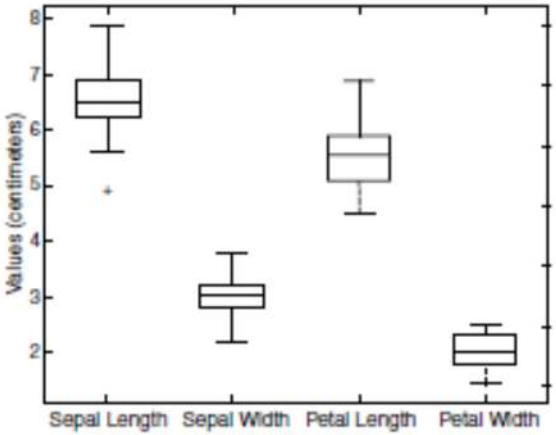
Box plots can also be used to compare how attributes vary between different classes of objects



(a) Setosa.



(b) Versicolour.



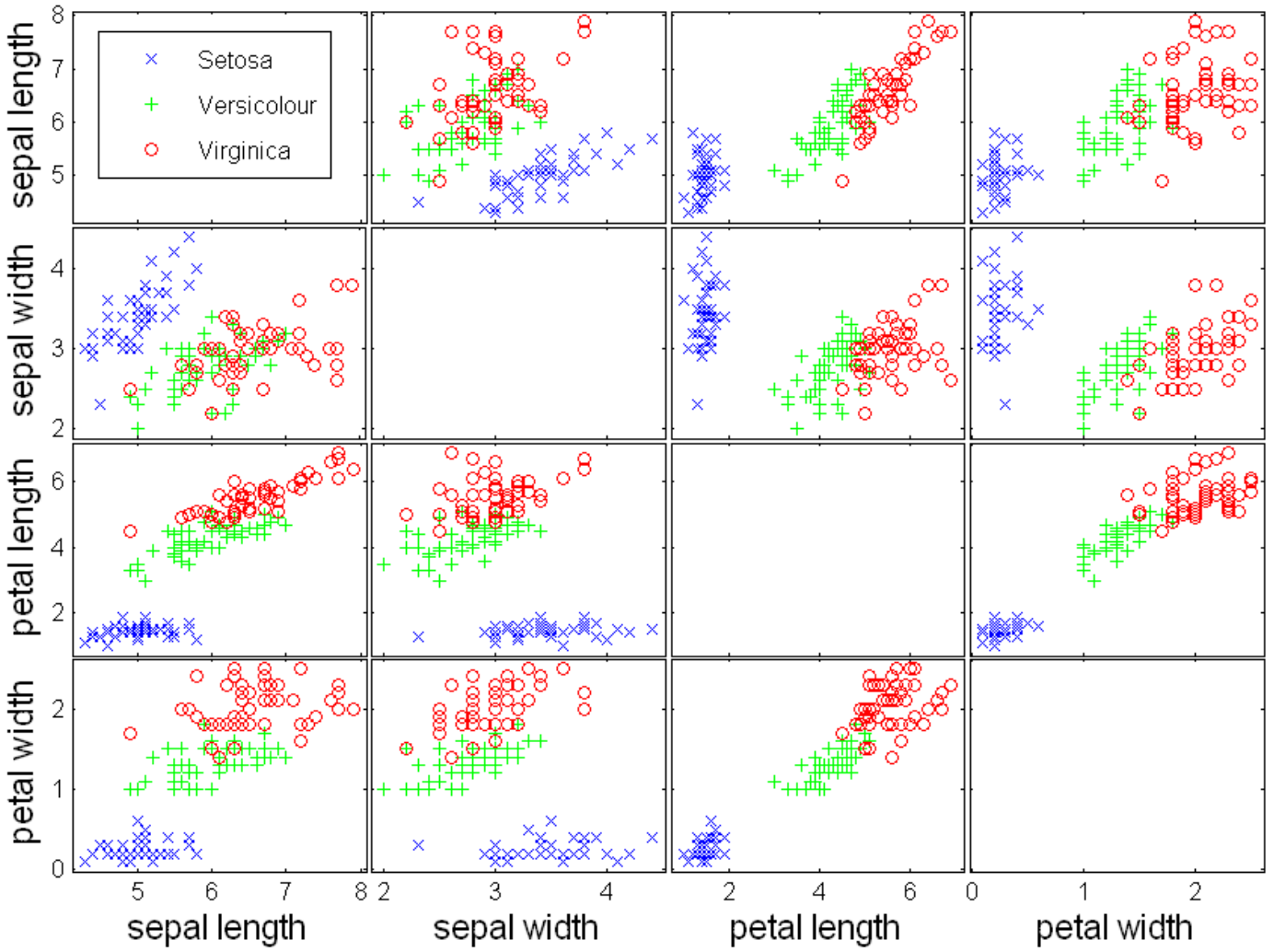
(c) Virginica.

Visualization Techniques: Scatter Plots

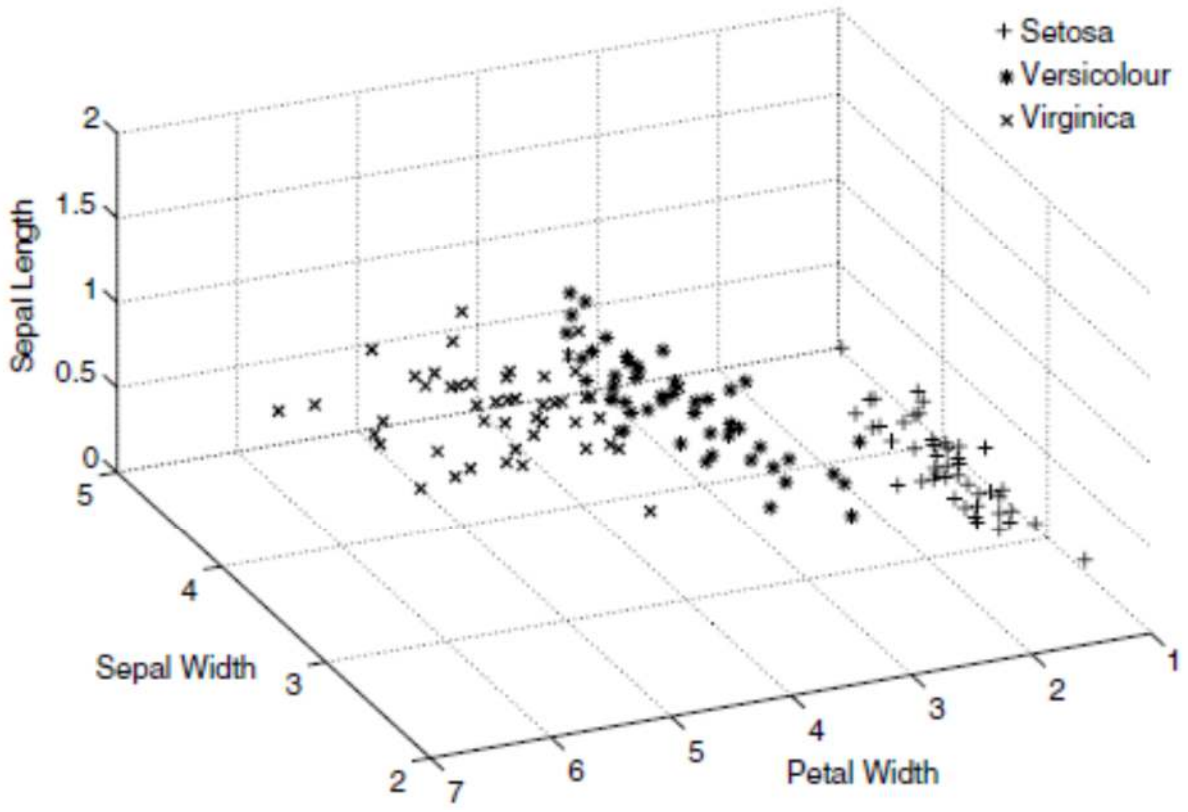
Scatter plots

- ✓ Attributes values determine the position
- ✓ Two-dimensional scatter plots most common, but can have three-dimensional scatter plots
- ✓ Often additional attributes can be displayed by using the size, shape, and color of the markers that represent the objects
- ✓ It is useful to have arrays of scatter plots can compactly summarize the relationships of several pairs of attributes

Visualization Techniques: Scatter Plots



Visualization Techniques: Scatter Plots

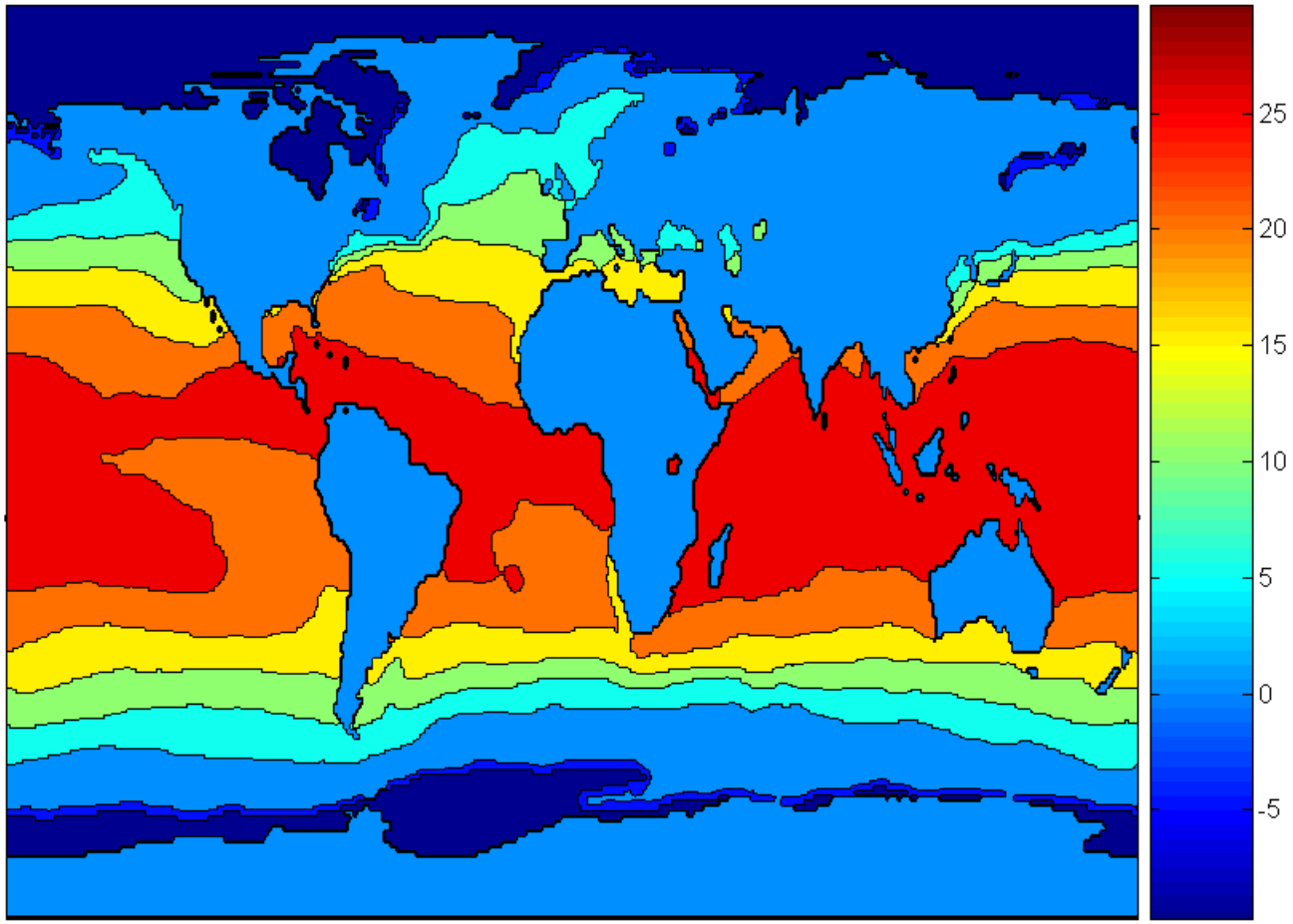


Visualization Techniques: Contour Plots

Contour plots

- ✓ Useful when a continuous attribute is measured on a spatial grid
- ✓ They partition the plane into regions of similar values
- ✓ The contour lines that form the boundaries of these regions connect points with equal values
- ✓ The most common example is contour maps of elevation
- ✓ Can also display temperature, rainfall, air pressure, etc.

Visualization Techniques: Contour Plots

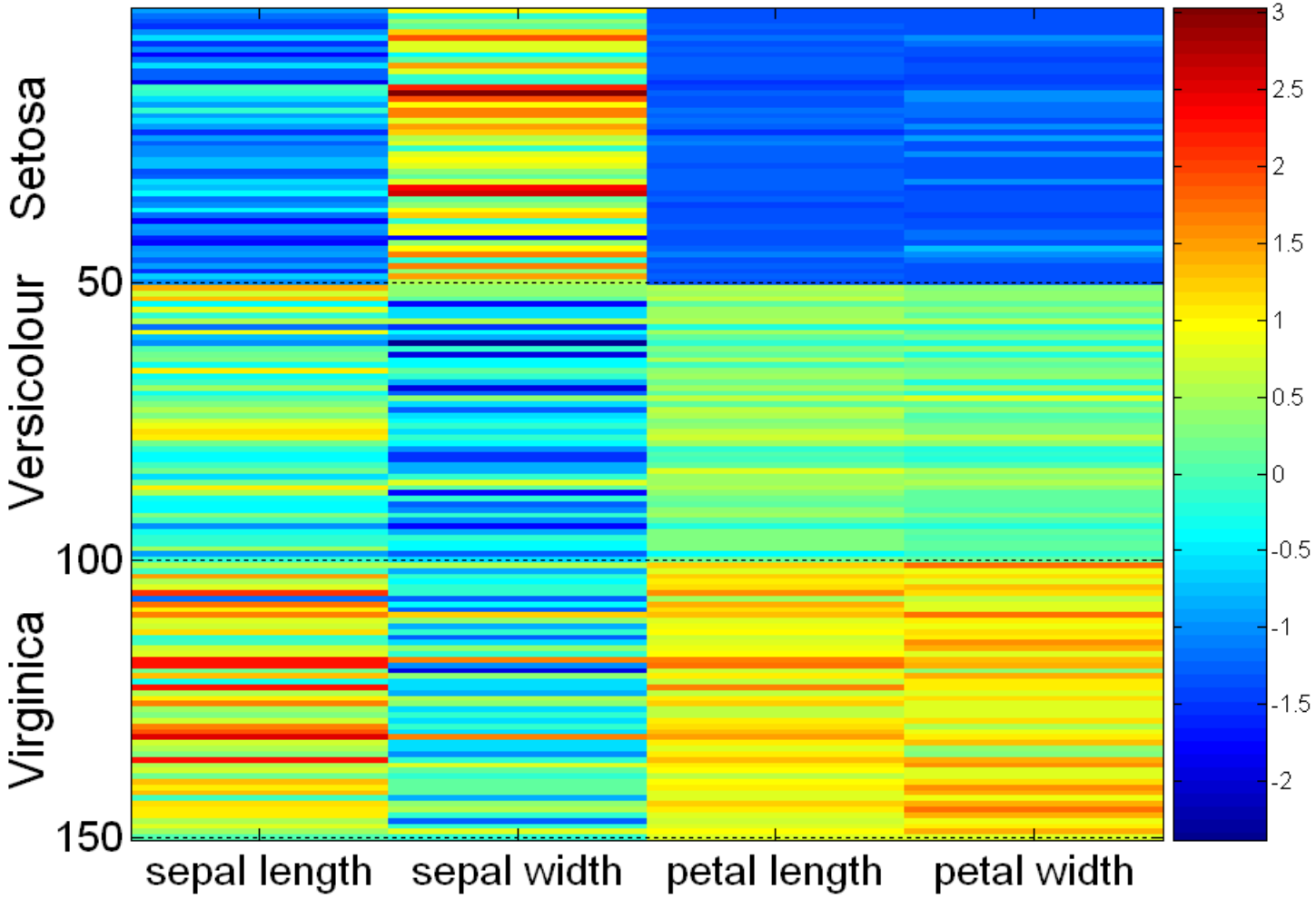


Visualization Techniques: Matrix Plots

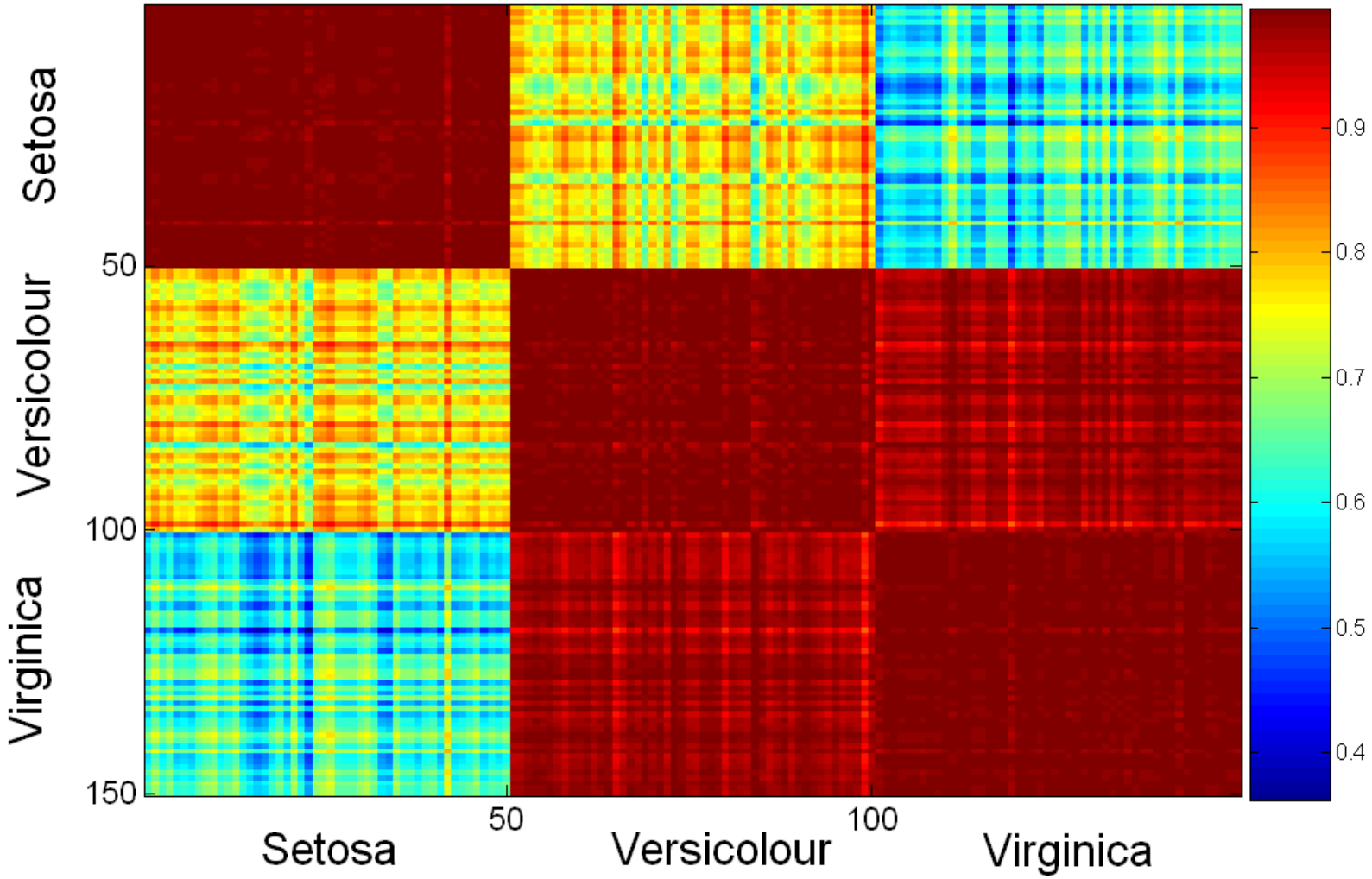
Matrix plots

- ✓ Can plot the data matrix
- ✓ This can be useful when objects are sorted according to class
- ✓ Typically, the attributes are normalized to prevent one attribute from dominating the plot
- ✓ Plots of similarity or distance matrices can also be useful for visualizing the relationships between objects

Visualization Techniques: Matrix Plots



Visualization Techniques: Matrix Plots



REGRESSION

Regression

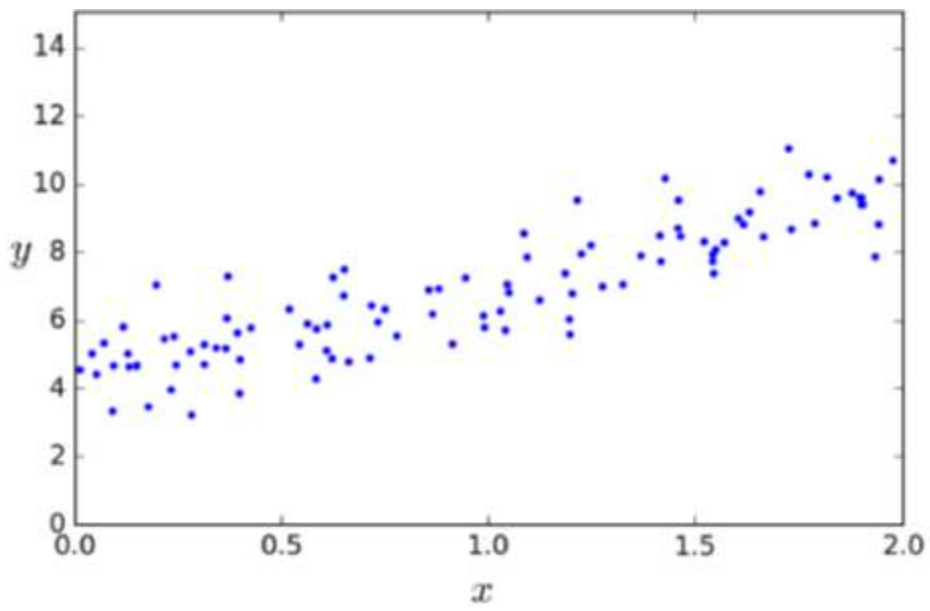
Relationships among several quantities

build a model that predicts the value of one variable as a function of other variables

simplest relation between two variables x and y is the linear equation

Regression coefficients

$$y = \beta_0 + \beta_1 x \qquad (x_1, y_1), \ldots, (x_n, y_n)$$



Regression of y on x

Regression

$$y = \beta_0 + \beta_1 x$$

that's the Linear Regression model—but how do we train it?

training a model means setting its parameters so that the model best fits the training set

Regression

If the data points were on the line, the parameters would satisfy the equations

Predicted y-value		Observed y-value
$\beta_0 + \beta_1 x_1$	=	y_1
$\beta_0 + \beta_1 x_2$	=	y_2
$\vdots$		$\vdots$
$\beta_0 + \beta_1 x_n$	=	y_n

if the data points don't lie on a line

$$X = \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \vdots \\ 1 & x_n \end{bmatrix}, \quad \boldsymbol{\beta} = \begin{bmatrix} \beta_0 \\ \beta_1 \end{bmatrix}, \quad \mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$$

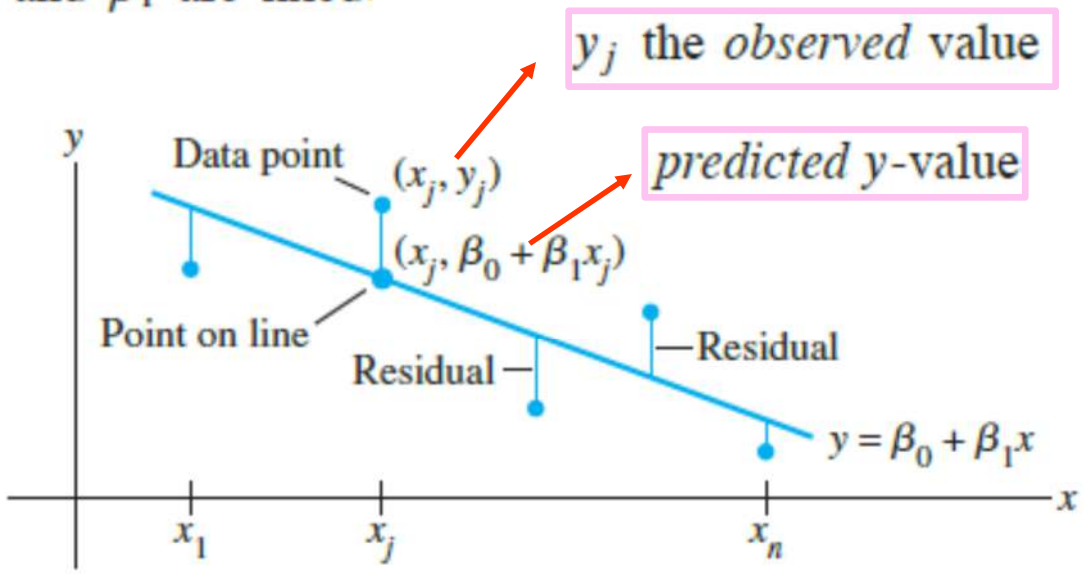
$$X\boldsymbol{\beta} = \mathbf{y}$$

training a model means setting its parameters so that the model best fits the training set

we first need a measure of how well (or poorly) the model fits the training data.

Regression

Suppose β_0 and β_1 are fixed



Regression

There are several ways to measure how "close" the line is to the data

The usual choice is to add the squares of the residuals

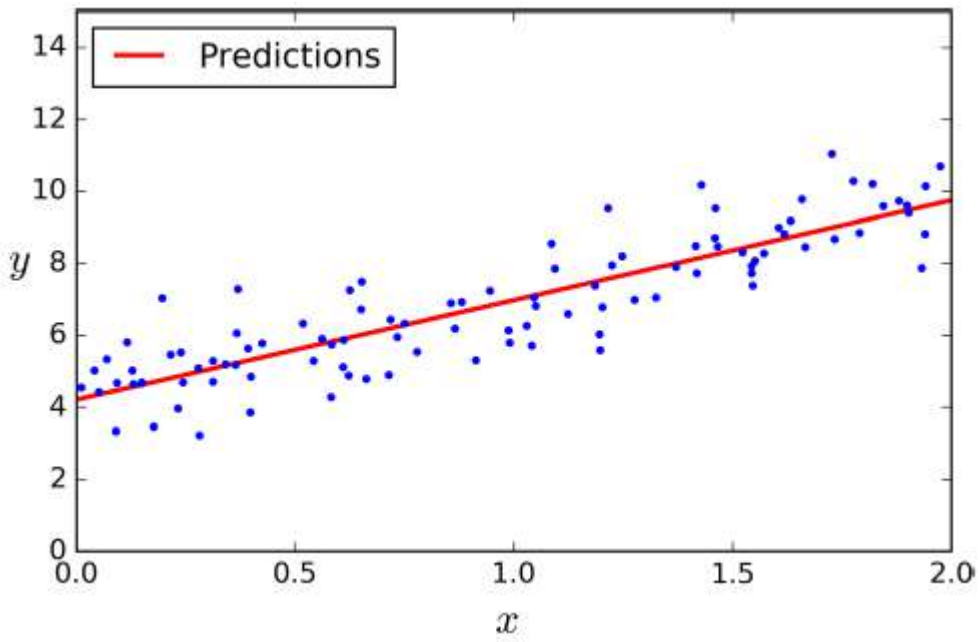
least-squares line is the that minimizes the sum of the squares of the residuals

$$residual = \epsilon = y - X\beta$$

$$\min \|X\beta - y\|_2^2$$

Cost function

MSE
RMSE

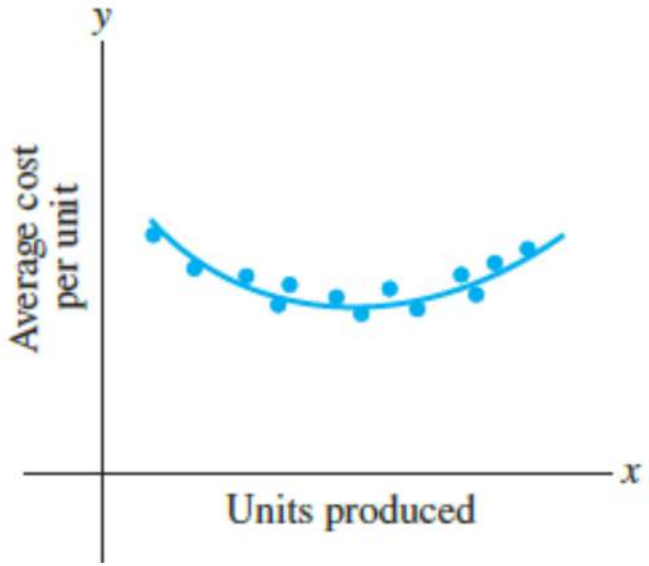


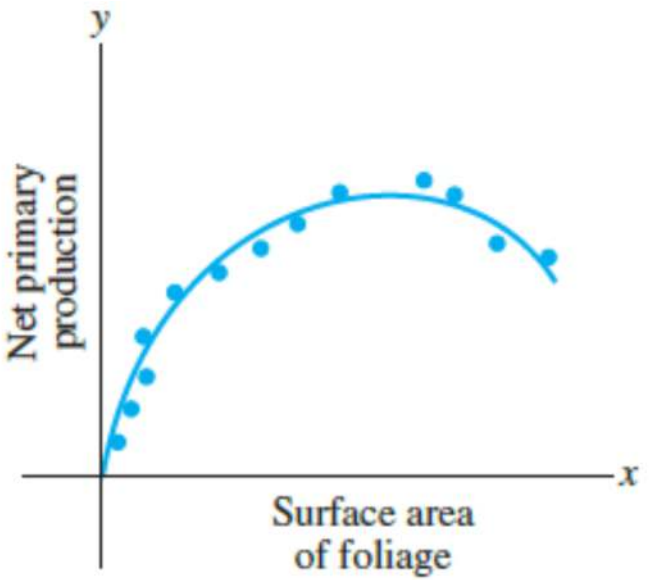
Regression(Curve fitting)

data points $(x_1, y_1), \ldots, (x_n, y_n)$ on a scatter plot do not lie close to any line.

some other functional relationship between x and y

$$y = \beta_0 f_0(x) + \beta_1 f_1(x) + \cdots + \beta_k f_k(x)$$





Regression(Curve fitting)

Example

$$y = \beta_0 + \beta_1 x + \beta_2 x^2 \qquad (x_1, y_1), \ldots, (x_n, y_n)$$

$$\begin{aligned} y_1 &= \beta_0 + \beta_1 x_1 + \beta_2 x_1^2 + \epsilon_1 \\ y_2 &= \beta_0 + \beta_1 x_2 + \beta_2 x_2^2 + \epsilon_2 \\ &\vdots \qquad\qquad\qquad \vdots \\ y_n &= \beta_0 + \beta_1 x_n + \beta_2 x_n^2 + \epsilon_n \end{aligned}$$

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} = \begin{bmatrix} 1 & x_1 & x_1^2 \\ 1 & x_2 & x_2^2 \\ \vdots & \vdots & \vdots \\ 1 & x_n & x_n^2 \end{bmatrix} \begin{bmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \end{bmatrix} + \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ \vdots \\ \epsilon_n \end{bmatrix}$$

$$\mathbf{y} \quad = \quad X \quad \beta \quad + \quad \epsilon$$

$$residual = \epsilon = y - X\beta$$

$$\min \|X\beta - y\|_2^2$$

Multiple Regression

Multiple Regression

We have n features and we want to predict y based on them

$$\boldsymbol{x_1, x_2, \dots, x_m} \quad \Rightarrow \quad \boldsymbol{y}$$

$$\boldsymbol{y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_m x_m}$$

$$\boldsymbol{y = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_m x_m}$$

$$\boldsymbol{y = \omega_0 + \omega_1 x_1 + \omega_2 x_2 + \cdots + \omega_m x_m}$$

$$y = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} \qquad X = \begin{bmatrix} 1 & x_1^{(1)} & \dots & x_m^{(1)} \\ \vdots & \vdots & \vdots & \vdots \\ 1 & x_1^{(n)} & \dots & x_m^{(n)} \end{bmatrix} \qquad \beta = \begin{bmatrix} \beta_0 \\ \vdots \\ \beta_m \end{bmatrix}$$

$$\boldsymbol{residual = \epsilon = y - X\beta}$$

$$\boldsymbol{\min \|X\beta - y\|_2^2}$$

Multiple Regression

$$\boldsymbol{y = \beta_0 + \beta_1 f_1(x_1) + \beta_2 f_2(x_2) + \cdots + \beta_m f_m(x_m)}$$

$$X = \begin{bmatrix} 1 & f_1(x_1^{(1)}) & \dots & f_m(x_m^{(1)}) \\ \vdots & \vdots & \vdots & \vdots \\ 1 & f_1(x_1^{(n)}) & \dots & f_m(x_m^{(n)}) \end{bmatrix} \quad y = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} \quad \beta = \begin{bmatrix} \beta_0 \\ \vdots \\ \beta_m \end{bmatrix}$$

Least Square Problem

$$\boldsymbol{residual = \epsilon = y - X\beta}$$

$$\boldsymbol{\min \|X\beta - y\|_2^2}$$

Solving Least Square Problem

least-squares solution is a solution of the normal equations

$$\min \|X\beta - y\|_2^2$$

$$X^T X \beta = X^T \mathbf{y}$$

High computational complexity

Optimization Methods: Gradient Descent

Optimization algorithms

x_0

generate a sequence of iterates $\{x_k\}_{k=0}^{\infty}$

terminate : no more progress or a solution point with sufficient accuracy

$$x_{k+1} = x_k + \alpha p_k$$

direction

Step length or Learning Rate

α?

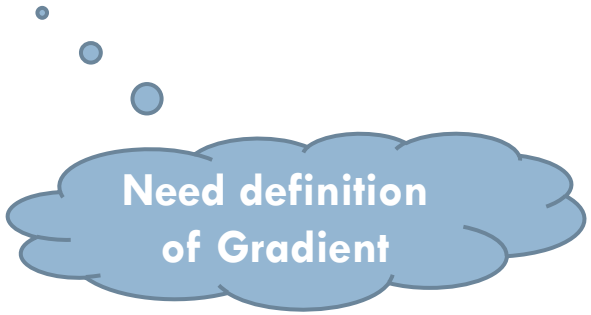
p_k?

Optimization algorithms

direction

Descent methods

✓any descent direction is guaranteed to produce a decrease in f , provided that the step length is sufficiently small



Gradient

$$f : \mathbf{R}^n \to \mathbf{R}$$

f is real-valued

$$\nabla f(x) \quad \Rightarrow \quad \nabla f(x)_i = \frac{\partial f(x)}{\partial x_i}, \quad i = 1, \ldots, n.$$

Gradient: example

Example:

quadratic function $f : \mathbf{R}^n \to \mathbf{R}$

$$f(x) = (1/2)x^T Px + q^T x + r$$

$P \in \mathbf{S}^n$, $q \in \mathbf{R}^n$, and $r \in \mathbf{R}$

$$\nabla f(x) = Px + q$$

Directional Derivative

$$\nabla_p f(x) = < \nabla f(x), p >$$

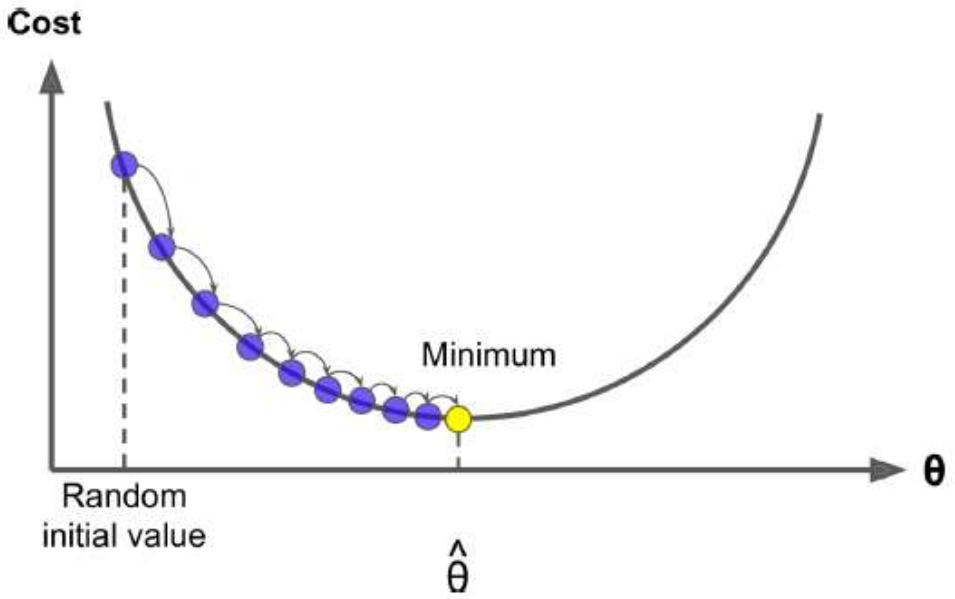
$\nabla_p f(x) < 0$  P is a descent direction

Gradient Descent

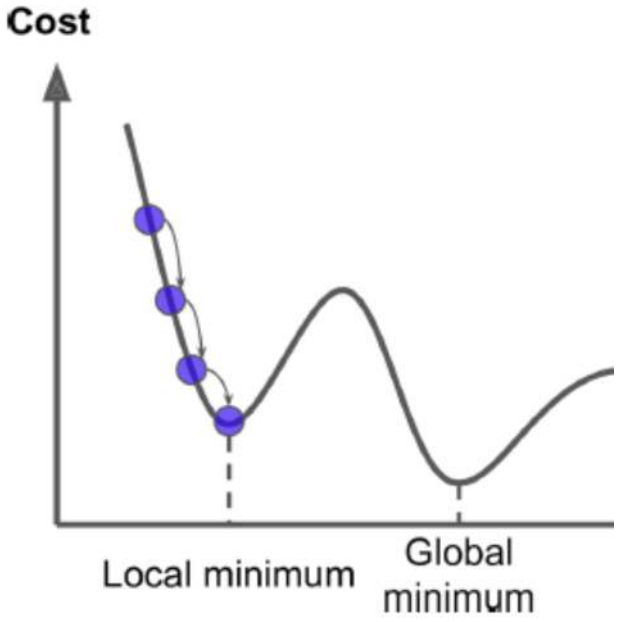
steepest descent direction

✓steepest descent direction $-\nabla f_k$ is the most obvious choice for search direction for a line search method.

✓choose the step length α in a variety of ways $x_{k+1} = x_k + \alpha_k(-\nabla f(x_k))$



Gradient Descent



$$f(\boldsymbol{\beta}) = \|\boldsymbol{X\beta} - \boldsymbol{y}\|_2^2$$

Convex function

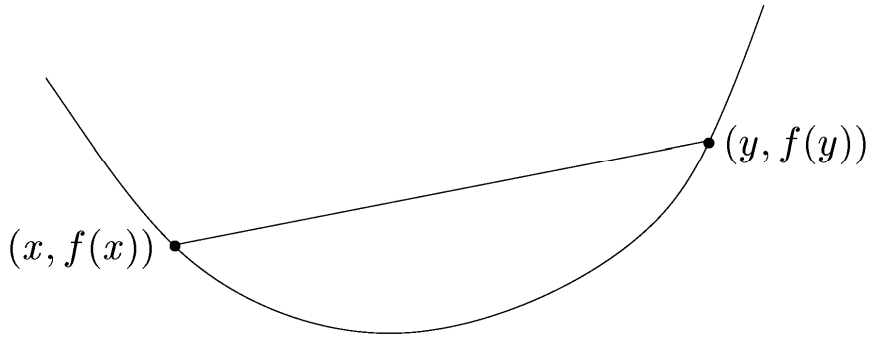
$$\nabla f(\boldsymbol{\beta}) = \boldsymbol{X}^T\boldsymbol{X\beta} - \boldsymbol{X}^T\boldsymbol{y}$$

Definition

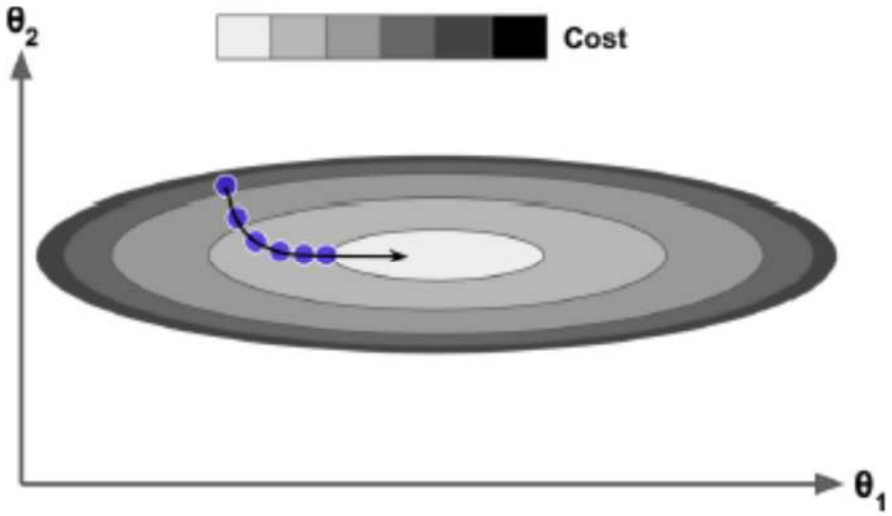
Convex function

A function $f : \mathbf{R}^n \to \mathbf{R}$ is *convex* if $\mathbf{dom}\, f$ is a convex set and if for all x, $y \in \mathbf{dom}\, f$, and θ with $0 \leq \theta \leq 1$, we have

$$f(\theta x + (1-\theta)y) \leq \theta f(x) + (1-\theta) f(y)$$



$$f(\boldsymbol{\beta}) = \|\boldsymbol{X\beta} - \boldsymbol{y}\|_2^2$$



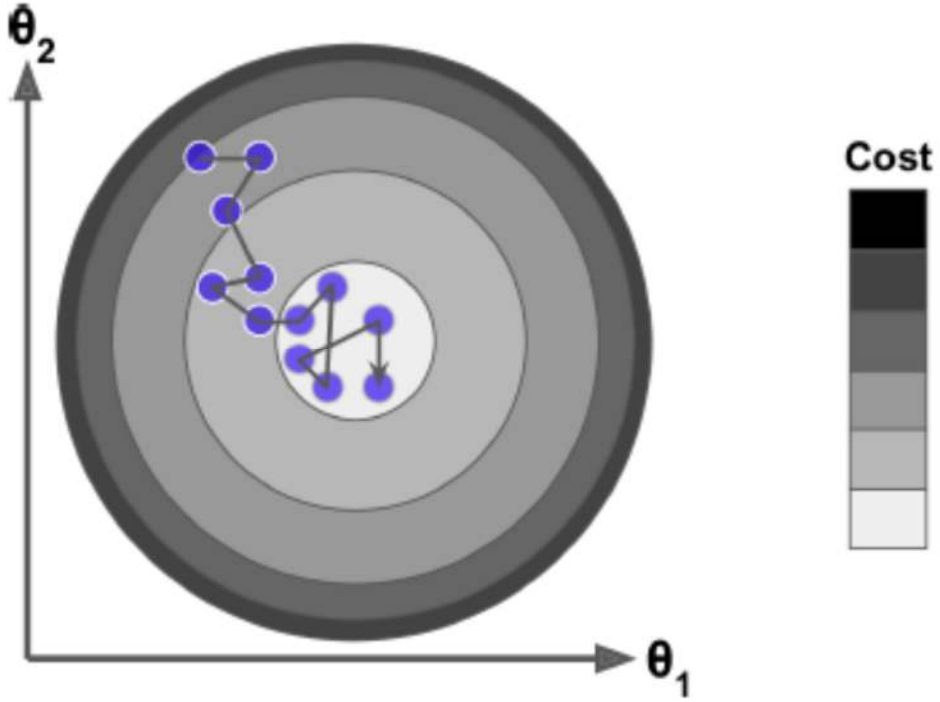
- ✓ training a model means searching for a combination of model parameters minimizes a cost function (over the training set)
- ✓ a search in the model's *parameter space*

Gradient Descent

- ✓ more parameters a model has, more dimensions this space has, and the harder search
- ✓ Batch Gradient Descent uses the whole training set to compute the gradients at every step, which makes it very slow when the training set is large.

- ✓ Stochastic Gradient Descent : picks a random instance in the training set at every step and computes the gradients based only on that single instance.
- ✓ is much less regular than Batch Gradient Descent
- ✓ instead of gently decreasing until it reaches the minimum, the cost function will bounce up and down, decreasing only on average.
- ✓ Mini- Batch Gradient Descent
- ✓ Mini-batch GD computes the gradients on small random sets of instances called mini-batches

Stochastic Gradient Descent

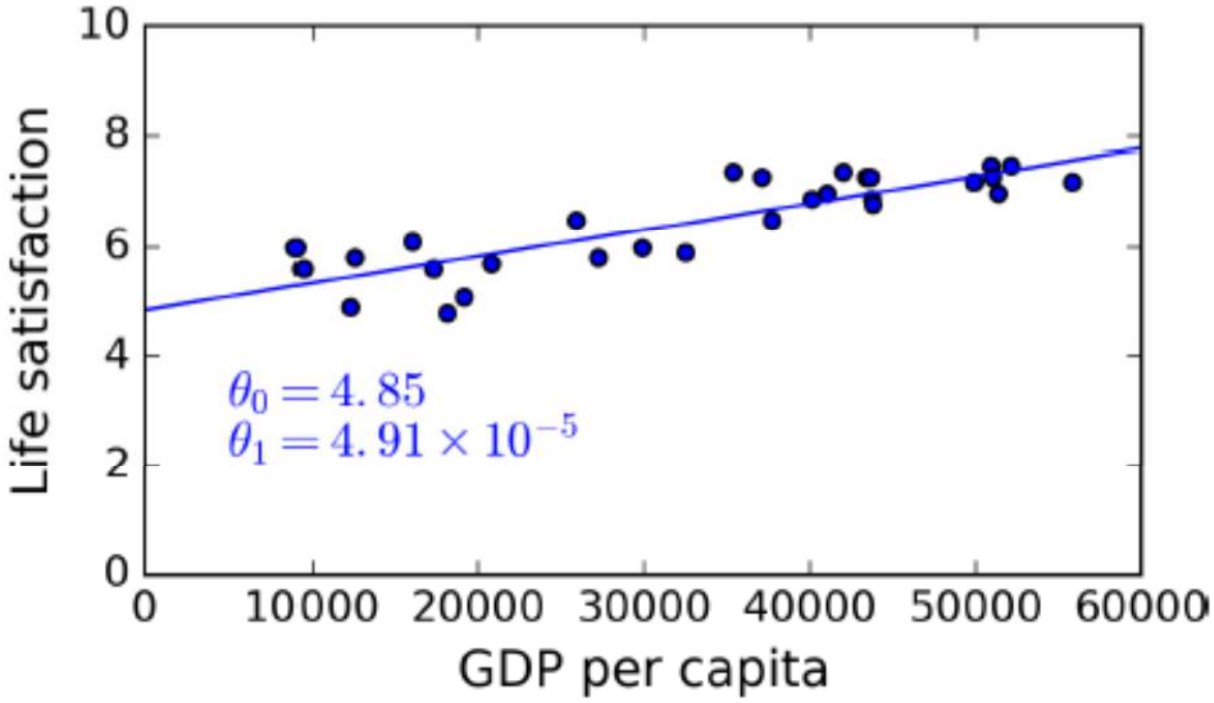


final parameter values are good, but not optimal

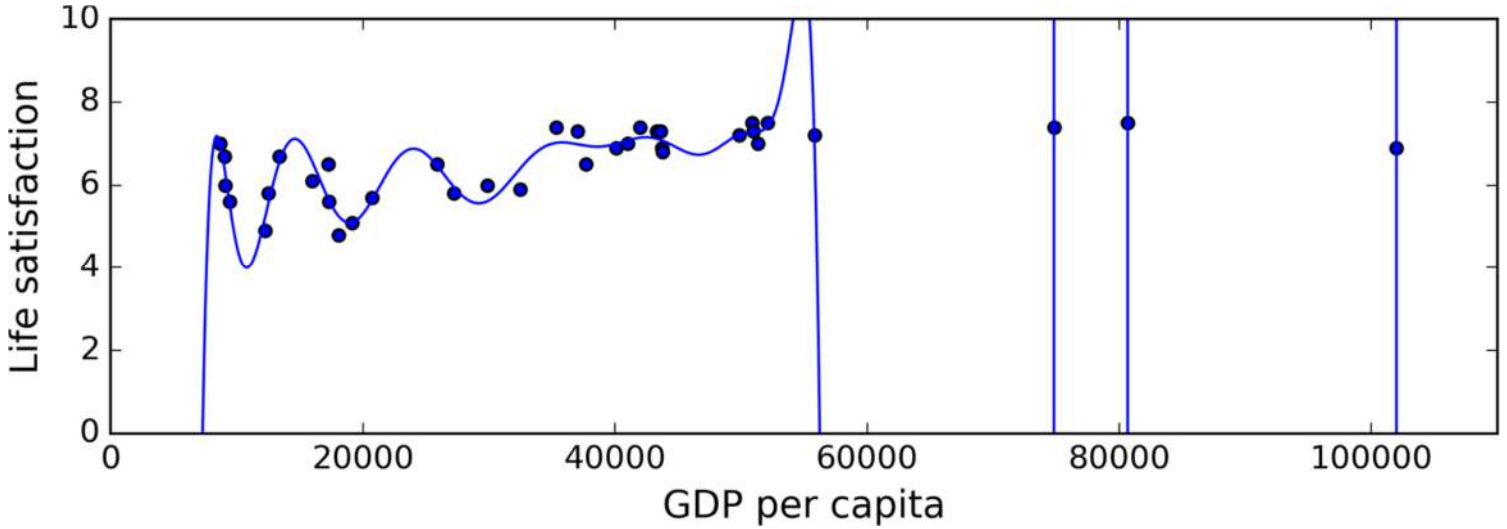
Overfitting and Underfitting

Overfitting

- Overgeneralizing is something that we humans do all too often
- machines can fall into the same trap
- In Machine Learning this is called overfitting
- model performs well on the training data, but it does not generalize well



Overfitting



Even though it performs much better on the training data than the simple linear model, would you really trust its predictions?

Overfitting

Overfitting happens when the model is too complex relative to the amount and noisiness of the training data

- simplify the model
- gather more training data
- reduce the noise in the training data

Underfitting

- underfitting is the opposite of overfitting
- it occurs when your model is too simple to learn the underlying structure of the data
- more powerful model

Bias/Variance Tradeoff

Bias

- due to wrong assumptions
- Assuming that the data is linear when it is actually quadratic.
- A high-bias model is most likely to underfit the training data.

Variance

- model's excessive sensitivity to small variations in the training data.
- A model with many degrees of freedom is likely to have high variance
- overfit the training data.

Regularization

- Constraining a model to make it simpler and reduce the risk of overfitting is called regularization
- degrees of freedom
- find the right balance between fitting the data perfectly and keeping the model simple enough to ensure that it will generalize well
- A simple way to regularize a polynomial model is to reduce the number of polynomial degrees.

- ✓ the fewer degrees of freedom model has, the harder it will be for it to overfit the data
- ✓ For a linear model, regularization is typically achieved by constraining the weights of the model.

Regularized Regression

Ridge Regression : *Tikhonov regularization*

regularization term $\|\beta\|^2$

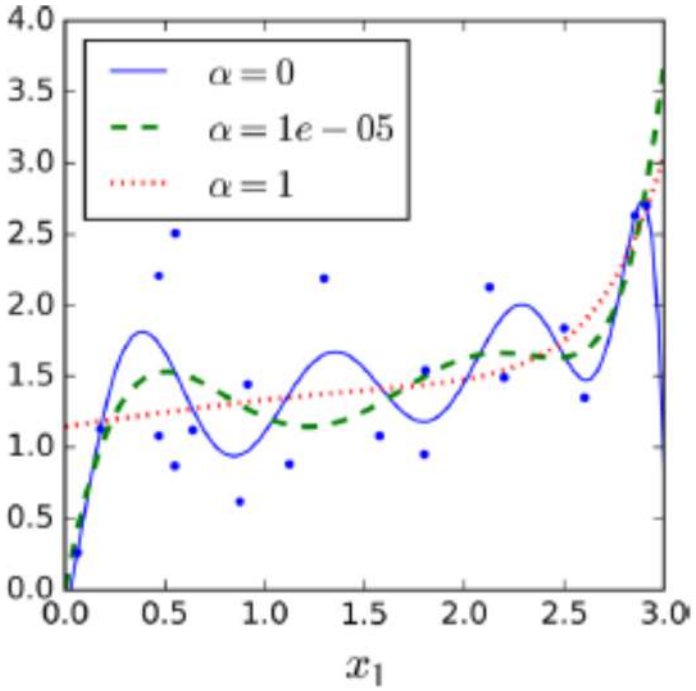
Hyperparameter

Cost Function $f(\boldsymbol{\beta}) = \|\boldsymbol{X\beta} - \boldsymbol{y}\|_2^2 + \boldsymbol{\alpha}\|\boldsymbol{\beta}\|_2^2$

Once the model is trained, you want to use the unregularized performance measure to evaluate the model's performance.

It is important to scale the data before performing Ridge Regression, as it is sensitive to the scale of the input features. This is true of most regularized models.

Tikhonov regularization

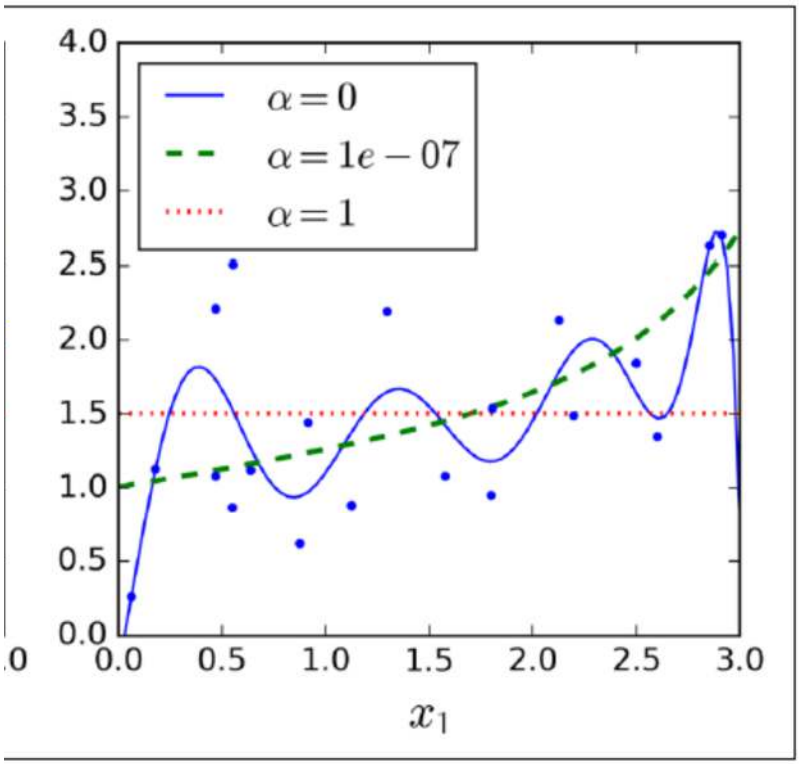


As with Linear Regression, we can perform Ridge Regression either by computing a closed-form equation or by performing Gradient Descent

Lasso Regression

Least Absolute Shrinkage
Selection Operator Regression

$$\boldsymbol{f(\beta) = \|X\beta - y\|_2^2 + \alpha\|\beta\|_1}$$



Lasso Regression

Lasso Regression automatically performs feature selection and outputs a *sparse model*

Lasso cost function is not differentiable
Gradient Descent
Subgradient

$$\boldsymbol{\alpha}\|\boldsymbol{\beta}\|_{\mathbf{1}} \Rightarrow \alpha \begin{bmatrix} sign(\beta_1) \\ sign(\beta_2) \\ \vdots \\ sign(\beta_m) \end{bmatrix}$$

Elastic Net
a middle ground between Ridge Regression and Lasso Regression

تمرین (سری اول)

- الف) در منظم سازی tikhonov معادله نرمال مربوطه را به دست آورید. ب) رابطه هر گام روش GD را برای این روش بنویسید ج) با استفاده از تعریف تابع محدب نشان دهید ترم منظم ساز اضافه شده در این روش یک ترم محدب است

$$f(\boldsymbol{\beta}) = \|\boldsymbol{X\beta} - \boldsymbol{y}\|_2^2 + \boldsymbol{\alpha}\|\boldsymbol{\beta}\|_2^2$$

CLASSIFICATION
DECISION TREES

Classification

- Given a collection of records (*training set*)
  - ✓ Each record contains a set of *attributes*, one of the attributes is the *class*.
- Find a *model* for class attribute as a function of the values of other attributes.
- Goal: previously unseen records should be assigned a class as accurately as possible.
  - ✓ A *test set* is used to determine the accuracy of the model. Usually, the given data set is divided into training and test sets, with training set used to build the model and test set used to validate it.

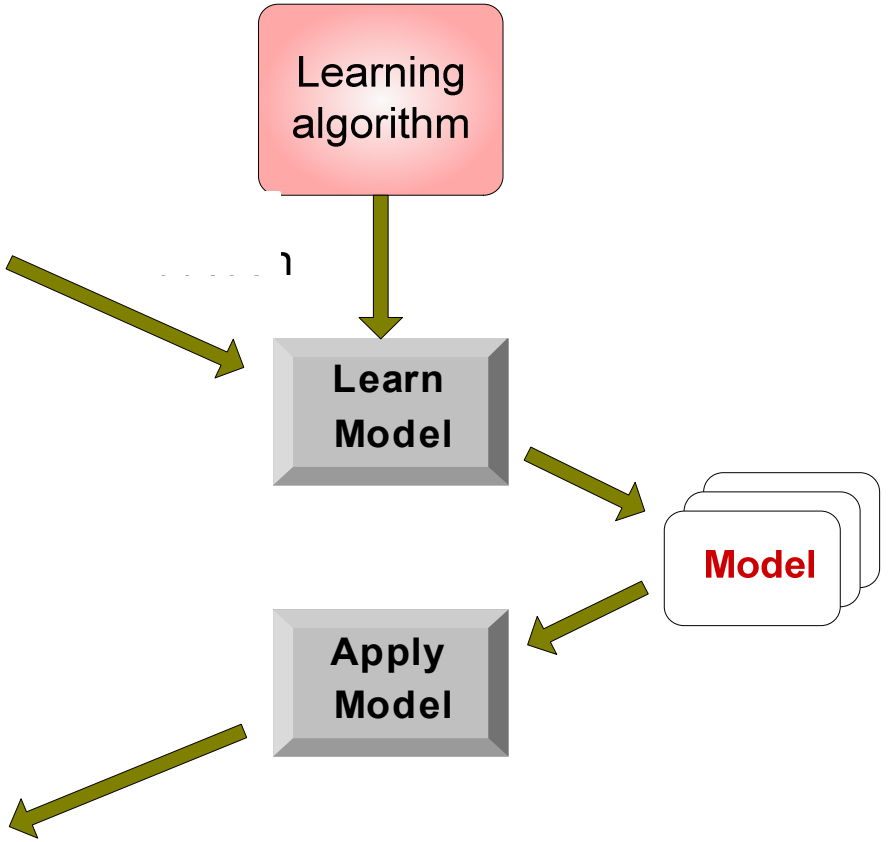
Classification

Tid	Attrib1	Attrib2	Attrib3	Class
1	Yes	Large	125K	No
2	No	Medium	100K	No
3	No	Small	70K	No
4	Yes	Medium	120K	No
5	No	Large	95K	Yes
6	No	Medium	60K	No
7	Yes	Large	220K	No
8	No	Small	85K	Yes
9	No	Medium	75K	No
10	No	Small	90K	Yes

Training Set

Tid	Attrib1	Attrib2	Attrib3	Class
11	No	Small	55K	?
12	Yes	Medium	80K	?
13	Yes	Large	110K	?
14	No	Small	95K	?
15	No	Large	67K	?

Test Set



Classification

		Predicted Class	
		$Class = 1$	$Class = 0$
Actual Class	$Class = 1$	f_{11}	f_{10}
	$Class = 0$	f_{01}	f_{00}

Confusion Matrix

$$\text{Accuracy} = \frac{\text{Number of correct predictions}}{\text{Total number of predictions}} = \frac{f_{11} + f_{00}}{f_{11} + f_{10} + f_{01} + f_{00}}$$

$$\text{Error rate} = \frac{\text{Number of wrong predictions}}{\text{Total number of predictions}} = \frac{f_{10} + f_{01}}{f_{11} + f_{10} + f_{01} + f_{00}}.$$

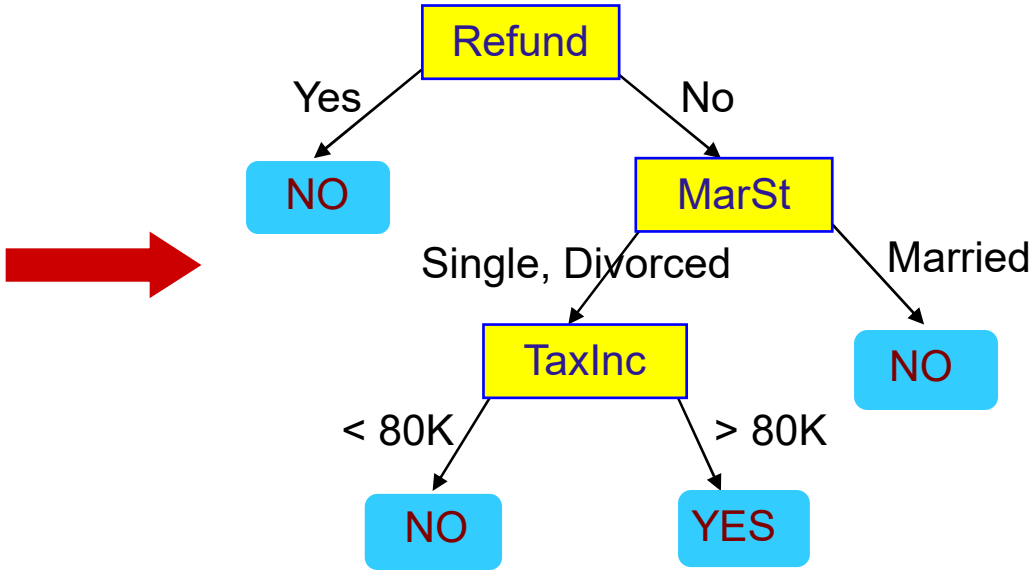
Decision Trees

- ✓ solve a classification problem by asking a series of carefully crafted questions about the attributes of the test record
- ✓ series of questions and their possible answers can be organized in the form of a decision Tree
- ✓ nodes and directed edges
  - ❖ **root node**
  - ❖ **Internal nodes**
  - ❖ **Leaf or terminal nodes**

Decision Trees

Tid	Refund	Marital Status	Taxable Income	Cheat
	categorical	categorical	continuous	class
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

Training Data



Model: Decision Tree

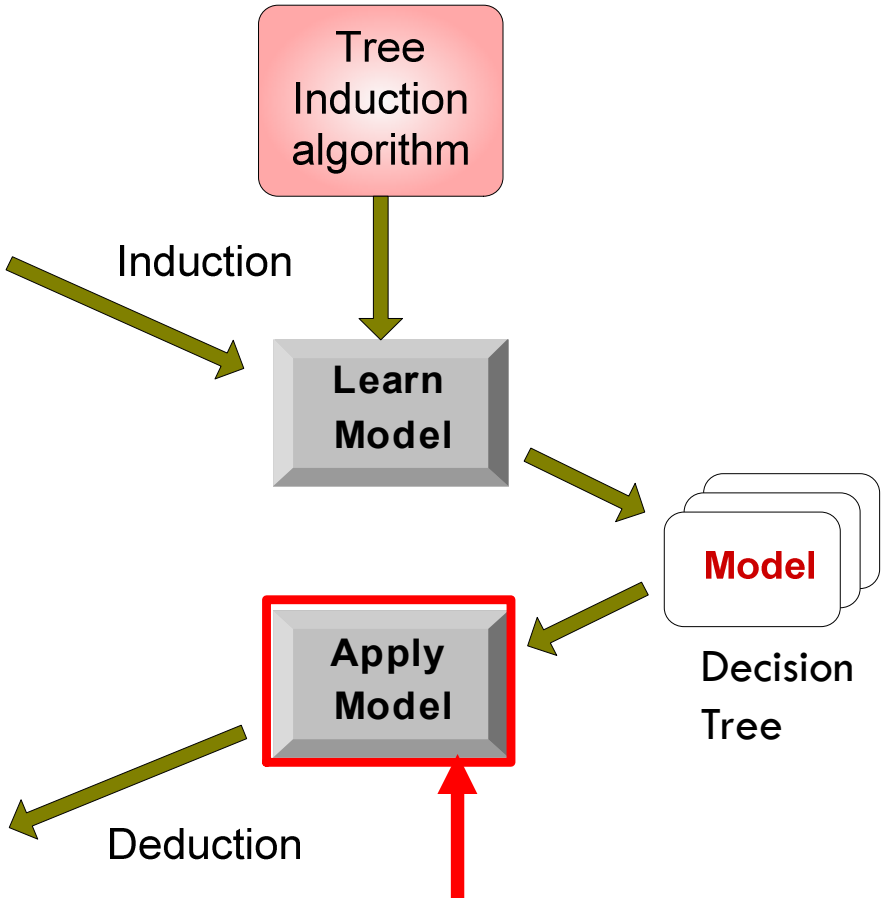
Decision Tree Classification Task

Tid	Attrib1	Attrib2	Attrib3	Class
1	Yes	Large	125K	No
2	No	Medium	100K	No
3	No	Small	70K	No
4	Yes	Medium	120K	No
5	No	Large	95K	Yes
6	No	Medium	60K	No
7	Yes	Large	220K	No
8	No	Small	85K	Yes
9	No	Medium	75K	No
10	No	Small	90K	Yes

Training Set

Tid	Attrib1	Attrib2	Attrib3	Class
11	No	Small	55K	?
12	Yes	Medium	80K	?
13	Yes	Large	110K	?
14	No	Small	95K	?
15	No	Large	67K	?

Test Set

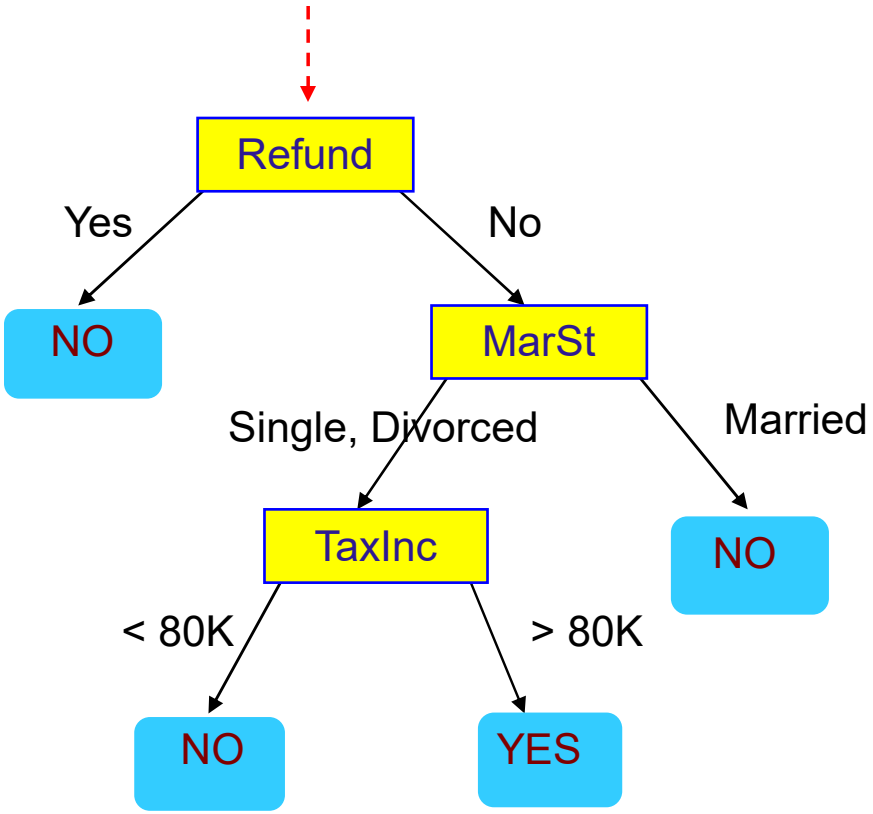


Apply Model to Test Data

Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

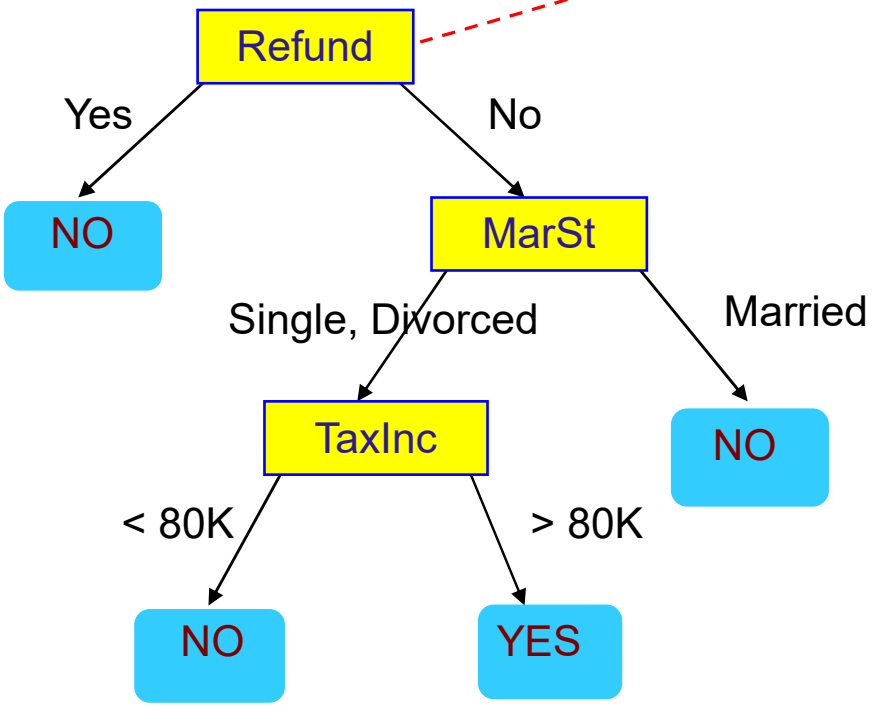
Start from the root of tree.



Apply Model to Test Data

Test Data

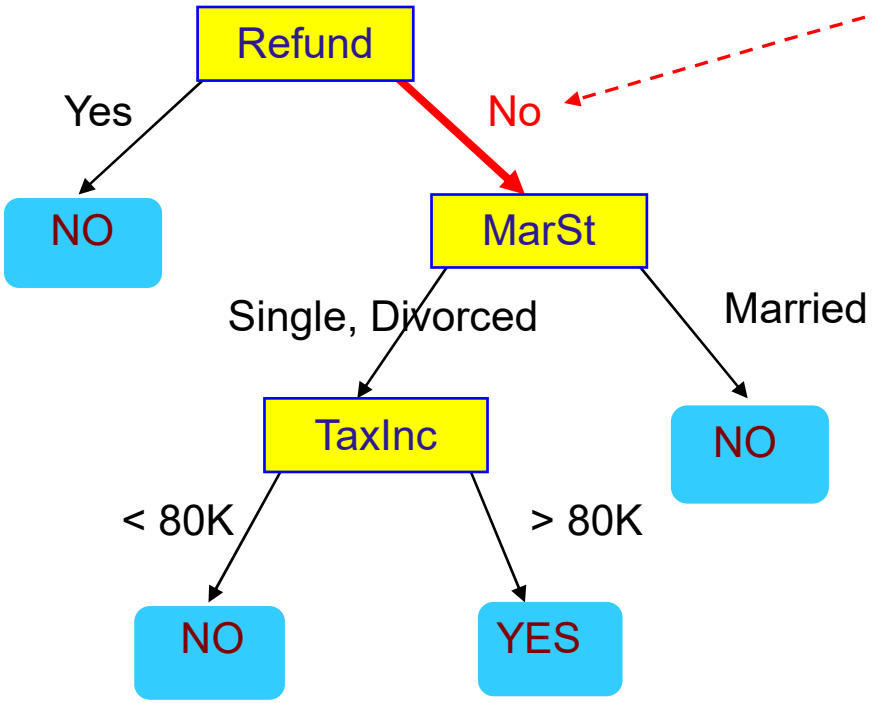
Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?



Apply Model to Test Data

Test Data

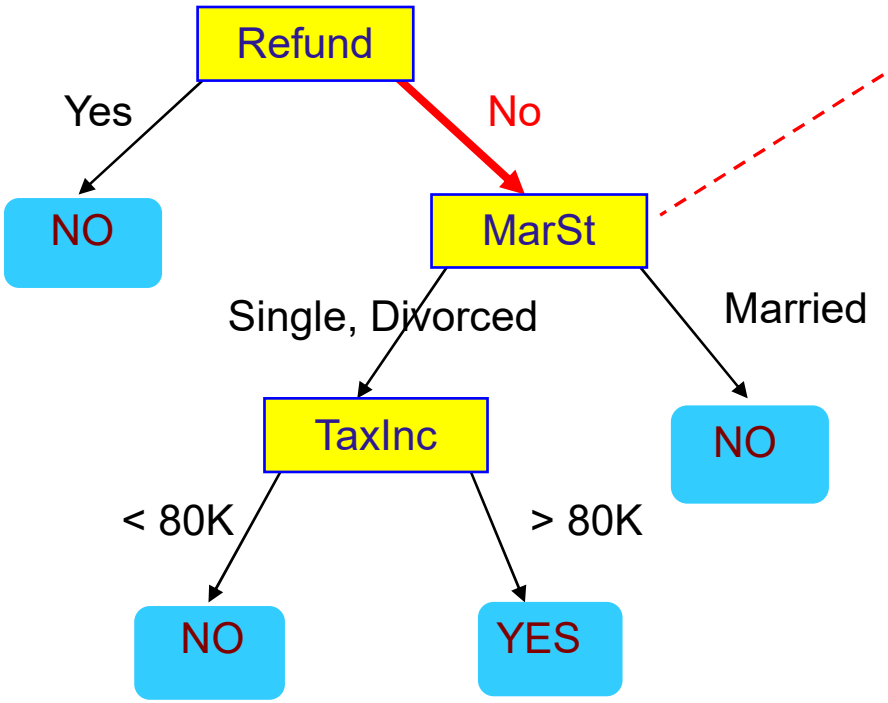
Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?



Apply Model to Test Data

Test Data

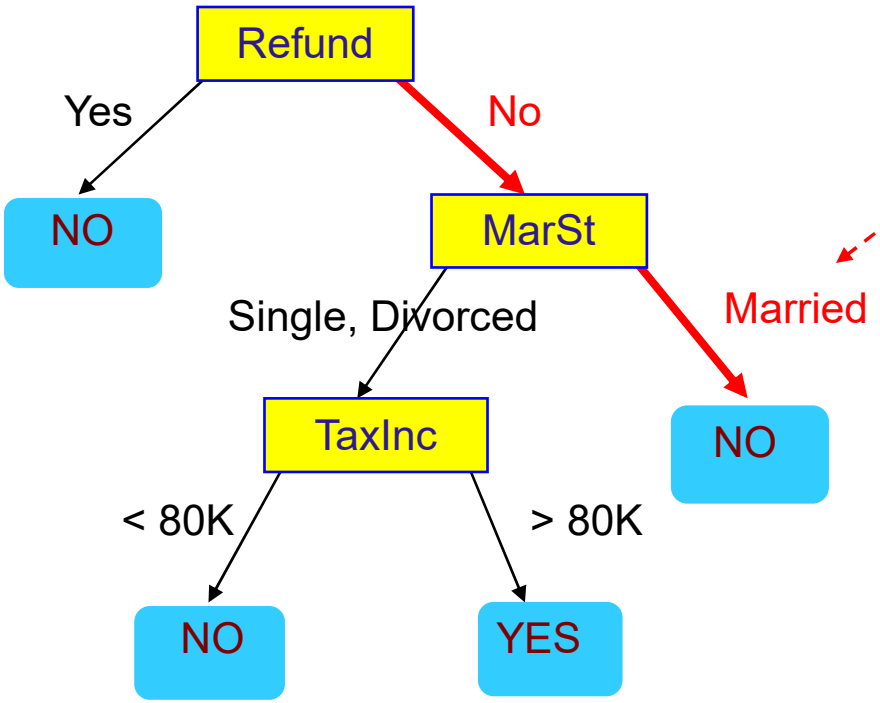
Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?



Apply Model to Test Data

Test Data

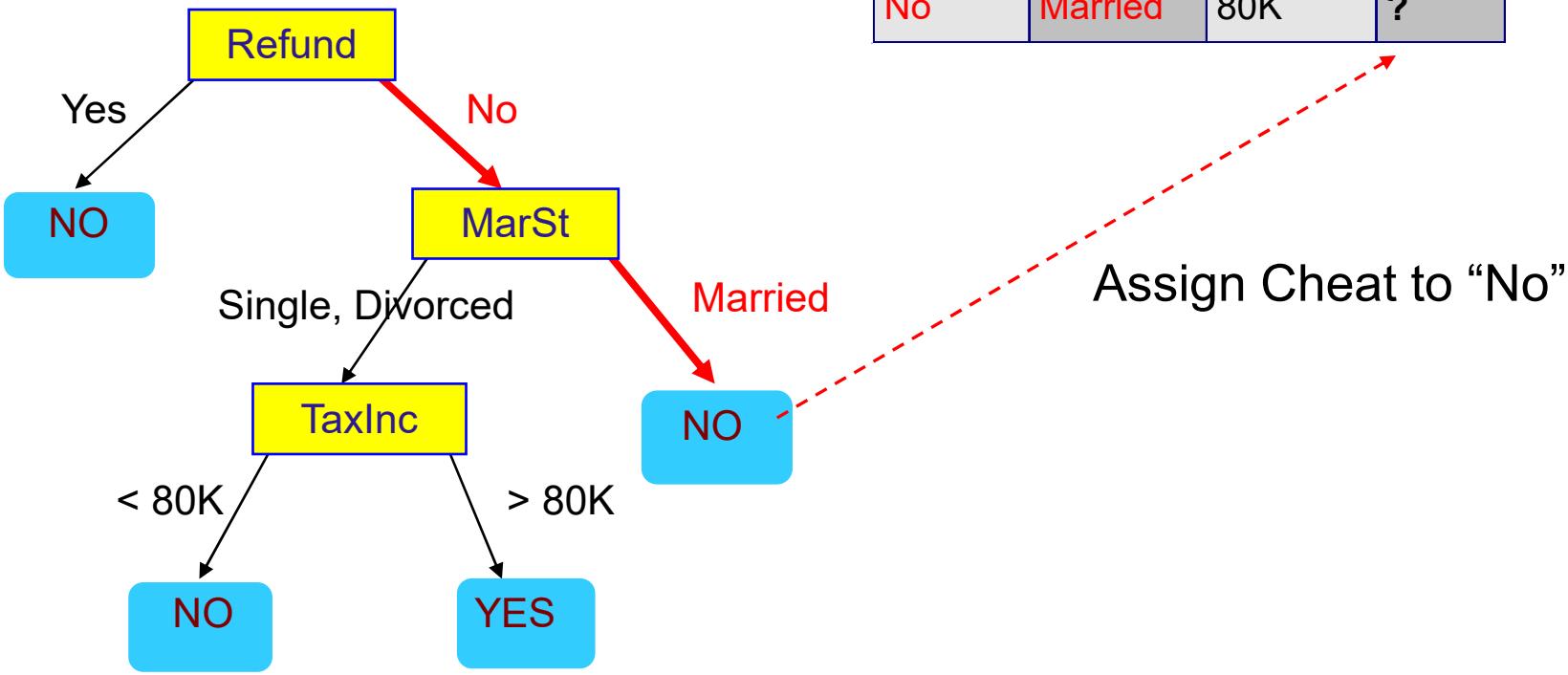
Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?



Apply Model to Test Data

Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?



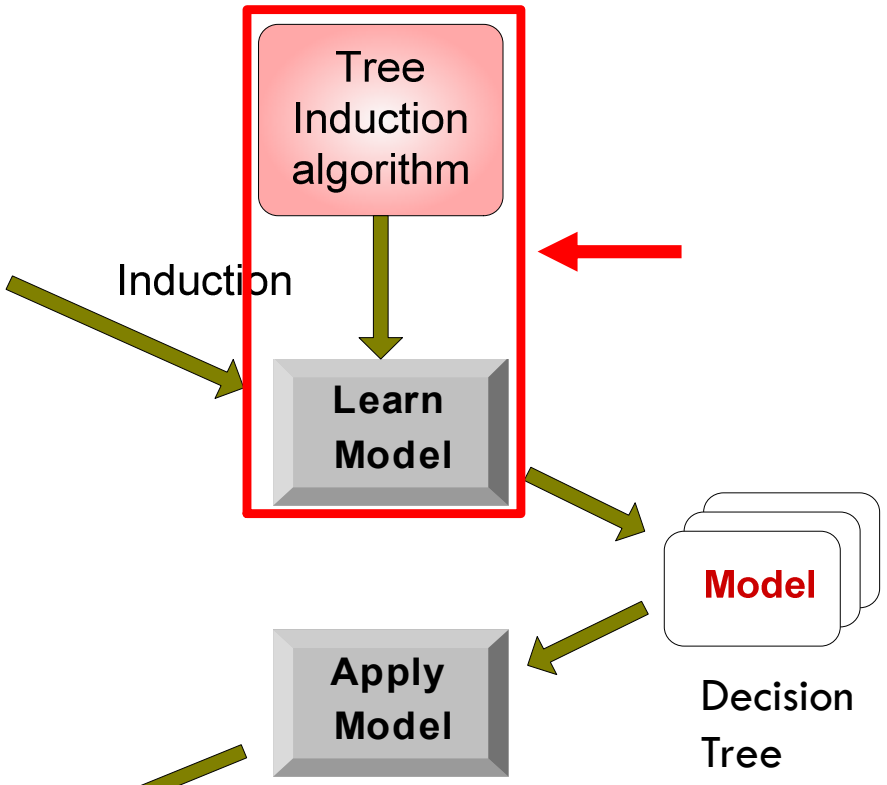
Decision Tree Classification Task

Tid	Attrib1	Attrib2	Attrib3	Class
1	Yes	Large	125K	No
2	No	Medium	100K	No
3	No	Small	70K	No
4	Yes	Medium	120K	No
5	No	Large	95K	Yes
6	No	Medium	60K	No
7	Yes	Large	220K	No
8	No	Small	85K	Yes
9	No	Medium	75K	No
10	No	Small	90K	Yes

Training Set

Tid	Attrib1	Attrib2	Attrib3	Class
11	No	Small	55K	?
12	Yes	Medium	80K	?
13	Yes	Large	110K	?
14	No	Small	95K	?
15	No	Large	67K	?

Test Set



How to Build a Decision Tree

Decision Trees

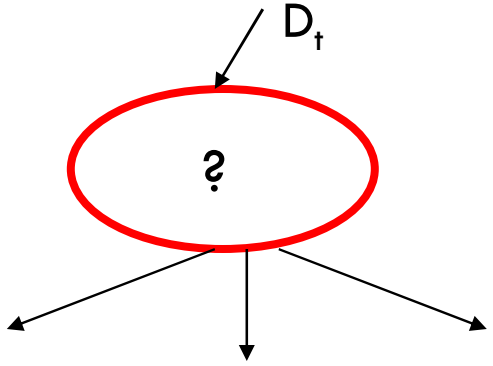
- many decision trees
- finding the optimal tree is computationally infeasible
- efficient algorithms to induce a reasonably accurate, albeit suboptimal, decision tree in a reasonable amount of time
- Employ a **greedy strategy**
- grows a decision tree by making a series of locally optimum decisions about which attribute to use for partitioning the data

Hunt's Algorithm

General Structure of Hunt's Algorithm

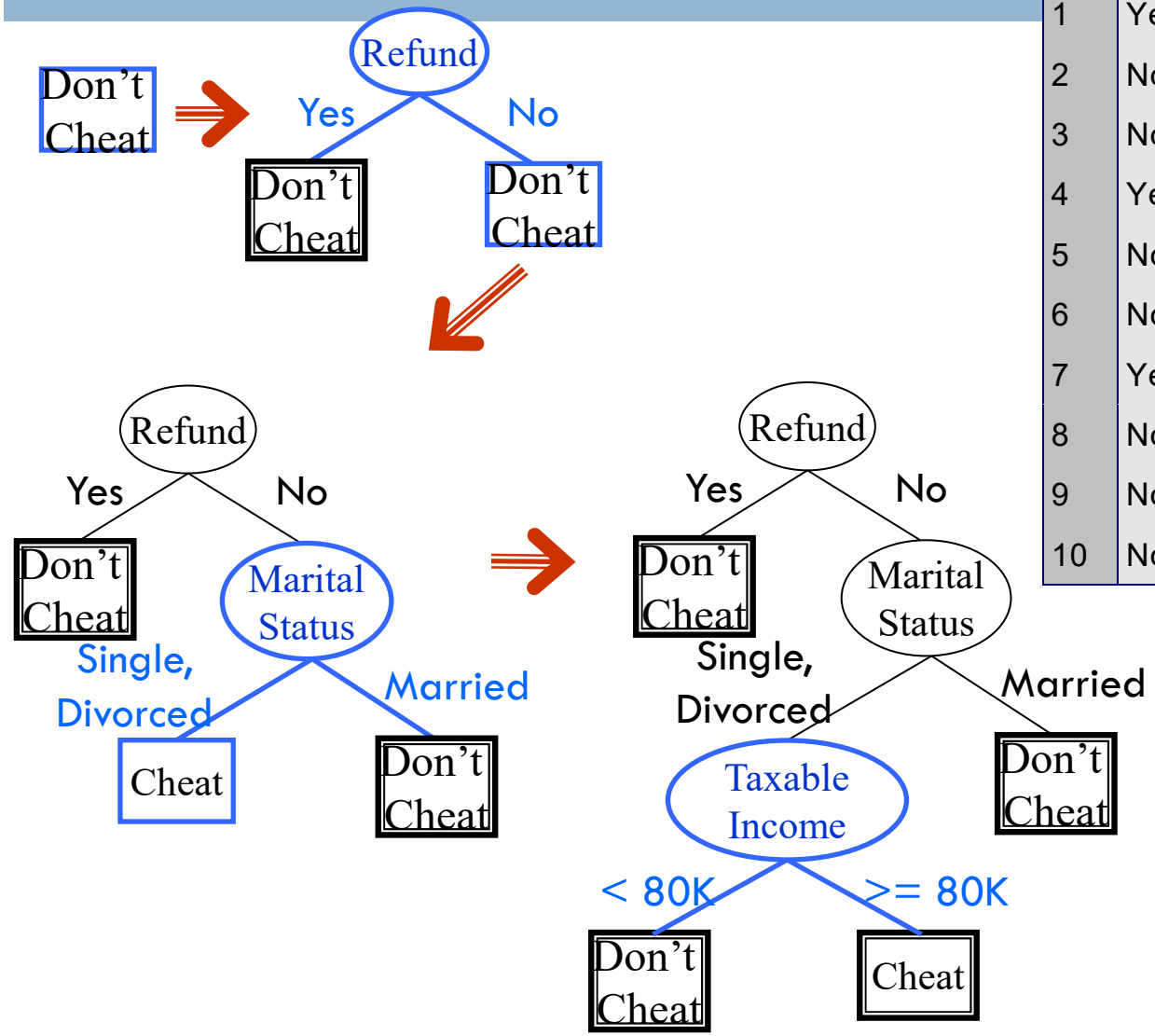
- Let D_t be the set of training records that reach a node t
- General Procedure:
  - If D_t contains records that belong the same class y_t, then t is a leaf node labeled as y_t
  - If D_t is an empty set, then t is a leaf node labeled by the default class, y_d
  - If D_t contains records that belong to more than one class, use an attribute test to split the data into smaller subsets. Recursively apply the procedure to each subset.

Tid	Refund	Marital Status	Taxable Income	Cheat
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes



Hunt's Algorithm

Tid	Refund	Marital Status	Taxable Income	Cheat
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes



Tree Induction

- Greedy strategy.
  - Split the records based on an attribute test that optimizes certain criterion.

- Issues
  - Determine how to split the records
    - How to specify the attribute test condition?
    - How to determine the best split?
  - Determine when to stop splitting

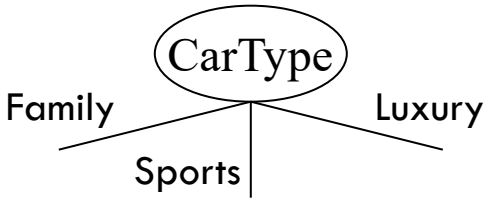
How to Specify Test Condition?

- Depends on attribute types
  - Nominal
  - Ordinal
  - Continuous

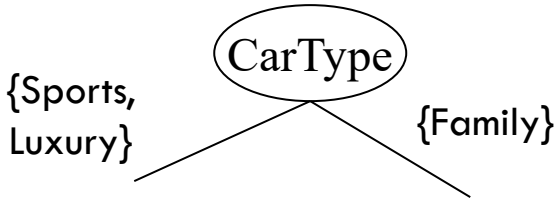
- Depends on number of ways to split
  - 2-way split
  - Multi-way split

Splitting Based on Nominal Attributes

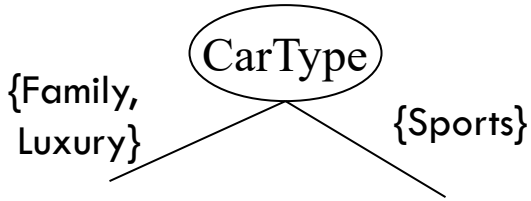
- Multi-way split: Use as many partitions as distinct values.



- Binary split: Divides values into two subsets.
  Need to find optimal partitioning.

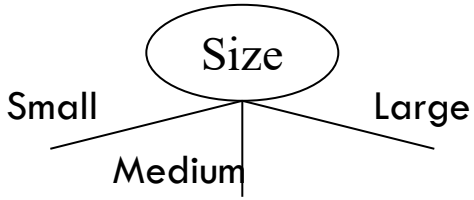


OR

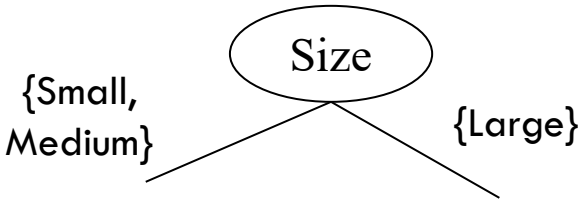


Splitting Based on Ordinal Attributes

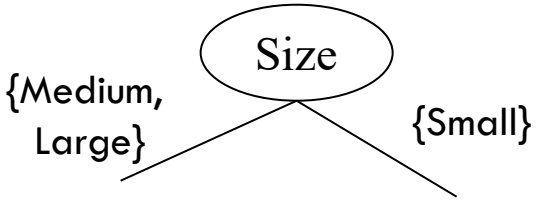
- **Multi-way split:** Use as many partitions as distinct values.



- **Binary split:** Divides values into two subsets.
  Need to find optimal partitioning.



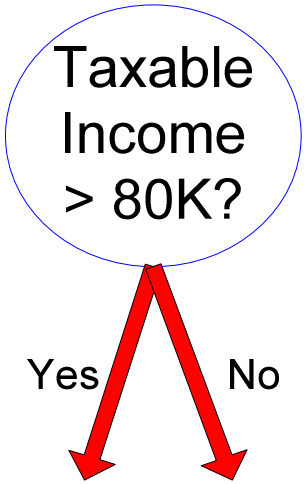
OR



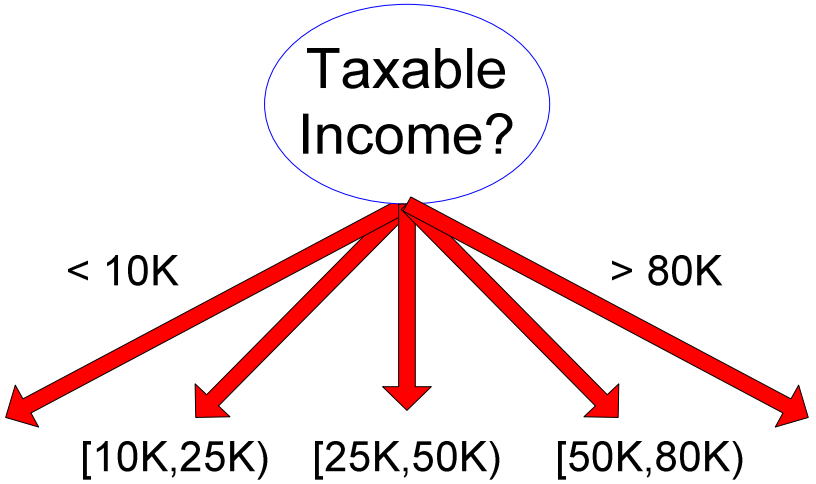
Splitting Based on Continuous Attributes

- Different ways of handling
  - Discretization to form an ordinal categorical attribute
  - Binary Decision: $(A < v)$ or $(A \geq v)$
    - consider all possible splits and finds the best cut

Splitting Based on Continuous Attributes



(i) Binary split



(ii) Multi-way split

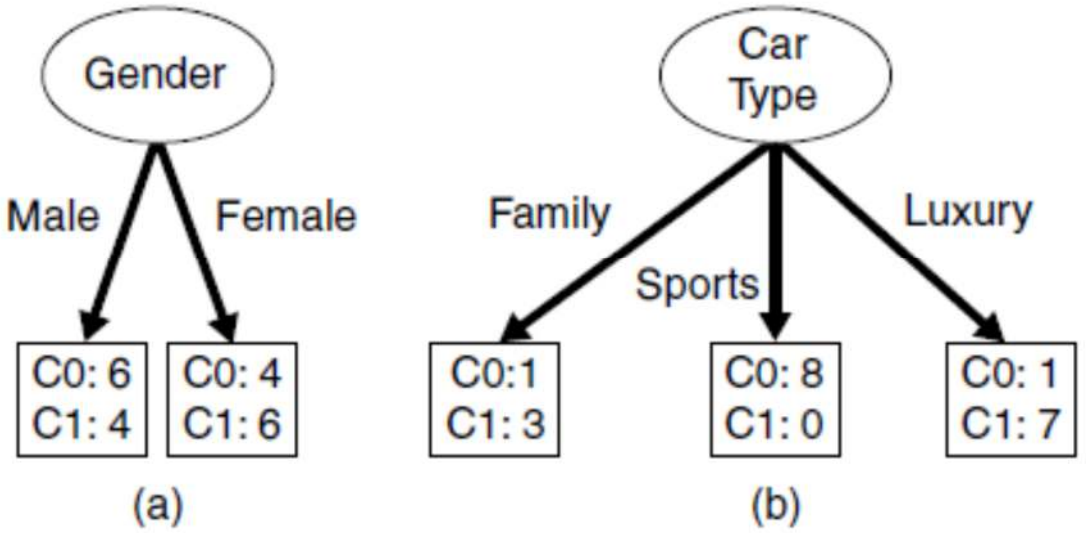
Tree Induction

- Greedy strategy.
  - Split the records based on an attribute test that optimizes certain criterion.

- Issues
  - Determine how to split the records
    - How to specify the attribute test condition?
    - How to determine the best split?
  - Determine when to stop splitting

How to determine the Best Split

Before Splitting: 10 records of class 0,
10 records of class 1



Which test condition is the best?

How to determine the Best Split

- Greedy approach:
  - Nodes with homogeneous class distribution are preferred
- Need a measure of node impurity:

C0: 5 C1: 5	C0: 9 C1: 1
Non-homogeneous, High degree of impurity	Homogeneous, Low degree of impurity

Measure of Impurity: GINI

- Gini Index for a given node t :

$$GINI(t) = 1 - \sum_j [p(j \mid t)]^2$$

(NOTE: $p(j \mid t)$ is the relative frequency of class j at node t).

- Maximum $(1 - 1/n_c)$ when records are equally distributed among all classes, implying least interesting information
- Minimum (0) when all records belong to one class, implying most interesting information

Examples for computing GINI

$$GINI(t) = 1 - \sum_j [p(j \mid t)]^2$$

C1	**0**
C2	**6**

P(C1) = 0/6 = 0 P(C2) = 6/6 = 1

Gini = 1 – P(C1)^2 – P(C2)^2 = 1 – 0 – 1 = 0

C1	**1**
C2	**5**

P(C1) = 1/6 P(C2) = 5/6

Gini = 1 – (1/6)^2 – (5/6)^2 = 0.278

C1	**2**
C2	**4**

P(C1) = 2/6 P(C2) = 4/6

Gini = 1 – (2/6)^2 – (4/6)^2 = 0.444

Splitting Based on GINI

- Used in CART, SLIQ, SPRINT.
- When a node p is split into k partitions (children), the quality of split is computed as,

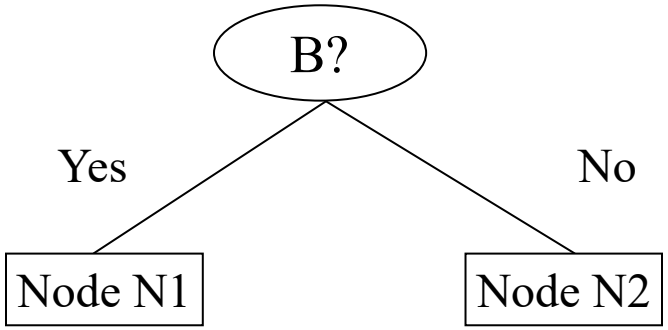
$$GINI_{split} = \sum_{i=1}^{k} \frac{n_i}{n} GINI(i)$$

where, n_i = number of records at child i,

n = number of records at node p.

Binary Attributes: Computing GINI Index

- Splits into two partitions

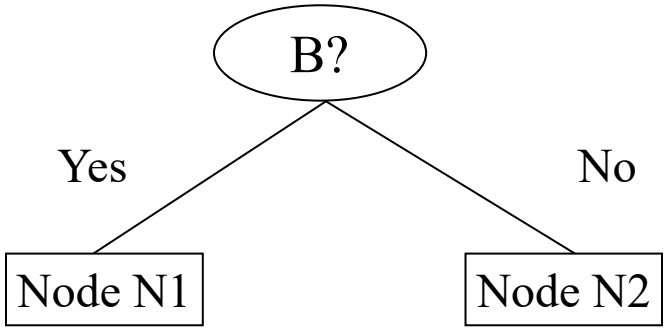


	Parent
C1	**6**
C2	**6**
Gini = 0.500	

	N1	N2
C1	**5**	**1**
C2	**2**	**4**
Gini=0.333		

Binary Attributes: Computing GINI Index

- Splits into two partitions



	Parent
C1	**6**
C2	**6**
Gini = 0.500	

Gini(N1)
= 1 – $(5/7)^2$ – $(2/7)^2$
= 0.194

Gini(N2)
= 1 – $(1/5)^2$ – $(4/5)^2$
= 0.528

	N1	N2
C1	**5**	**1**
C2	**2**	**4**
Gini=0.333		

Gini(Children)
= 7/12 * 0.194 +
5/12 * 0.528
= 0.333

Categorical Attributes: Computing Gini Index

- For each distinct value, gather counts for each class in the dataset
- Use the count matrix to make decisions

Multi-way split

	CarType		
	Family	Sports	Luxury
C1	1	2	1
C2	4	1	1
Gini	0.393		

Two-way split
(find best partition of values)

	CarType	
	{Sports, Luxury}	{Family}
C1	3	1
C2	2	4
Gini	0.400	

	CarType	
	{Sports}	{Family, Luxury}
C1	2	2
C2	1	5
Gini	0.419	

Continuous Attributes: Computing Gini Index...

- For efficient computation: for each attribute,
  - Sort the attribute on values
  - scan these values, each time updating the count matrix and computing gini index
  - Choose the split position that has the least gini index

Cheat	No		No		No		Yes		Yes		Yes		No		No		No		No	
Taxable Income																				
Sorted Values →	60		70		75		85		90		95		100		120		125		220	
Split Positions →	55		65		72		80		87		92		97		110		122		172		230	
	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>
Yes	0	3	0	3	0	3	0	3	1	2	2	1	3	0	3	0	3	0	3	0	3	0
No	0	7	1	6	2	5	3	4	3	4	3	4	3	4	4	3	5	2	6	1	7	0
Gini	0.420		0.400		0.375		0.343		0.417		0.400		***0.300***		0.343		0.375		0.400		0.420	

Alternative Splitting Criteria based on INFO

- Entropy at a given node t:

$$Entropy(t) = -\sum_j p(j \mid t) \log p(j \mid t)$$

(NOTE: $p(j \mid t)$ is the relative frequency of class j at node t).

- Measures homogeneity of a node.
  - Maximum (log n_c) when records are equally distributed among all classes implying least information
  - Minimum (0.0) when all records belong to one class, implying most information
- Entropy based computations are similar to the GINI index computations

Examples for computing Entropy

$$Entropy(t) = -\sum_j p(j \mid t) \log_2 p(j \mid t)$$

C1	**0**
C2	**6**

C1	**1**
C2	**5**

C1	**2**
C2	**4**

Examples for computing Entropy

$$Entropy(t) = -\sum_j p(j \mid t) \log_2 p(j \mid t)$$

C1	**0**
C2	**6**

P(C1) = 0/6 = 0 P(C2) = 6/6 = 1

Entropy = – 0 log 0 – 1 log 1 = – 0 – 0 = 0

C1	**1**
C2	**5**

P(C1) = 1/6 P(C2) = 5/6

Entropy = – (1/6) $\log_2$ (1/6) – (5/6) $\log_2$ (1/6) = 0.65

C1	**2**
C2	**4**

P(C1) = 2/6 P(C2) = 4/6

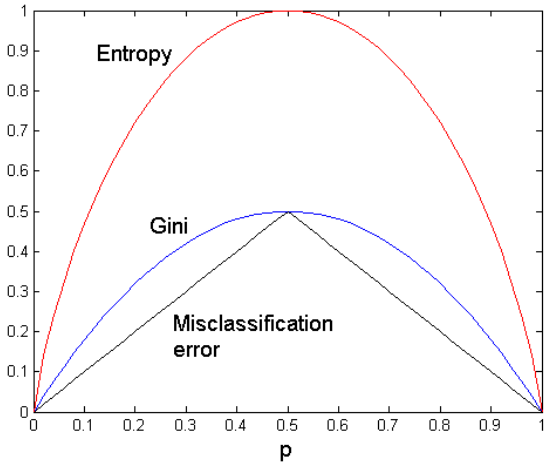
Entropy = – (2/6) $\log_2$ (2/6) – (4/6) $\log_2$ (4/6) = 0.92

Splitting Criteria based on Classification Error

- Classification error at a node t :

$$Error(t) = 1 - \max_i P(i \mid t)$$

- Measures misclassification error made by a node.
  - Maximum (1 - $1/n_c$) when records are equally distributed among all classes, implying least interesting information
  - Minimum (0.0) when all records belong to one class, implying most interesting information



Gain

gain, Δ, is a criterion that can be used to determine the goodness of a split

Entropy: Information Gain

$$\Delta = I(\text{parent}) - \sum_{j=1}^{k} \frac{N(v_j)}{N} I(v_j)$$

$I(\cdot)$ is the impurity measure
N is the total number of records at the parent node
k is the number of attribute values
$N(v_j)$ is the number of records associated with the child node, v_j.

Decision tree induction algorithms often choose a test condition that maximizes the gain

Gain ratio

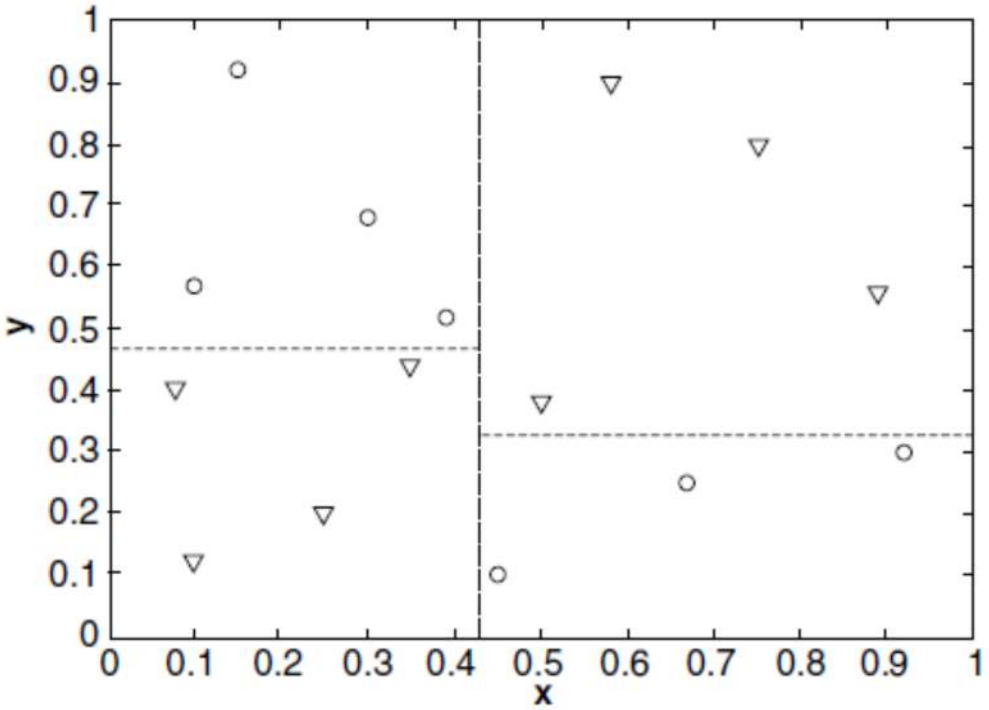
$$GainRATIO_{split} = \frac{GAIN_{Split}}{SplitINFO} \qquad SplitINFO = -\sum_{i=1}^{k} \frac{n_i}{n} \log \frac{n_i}{n}$$

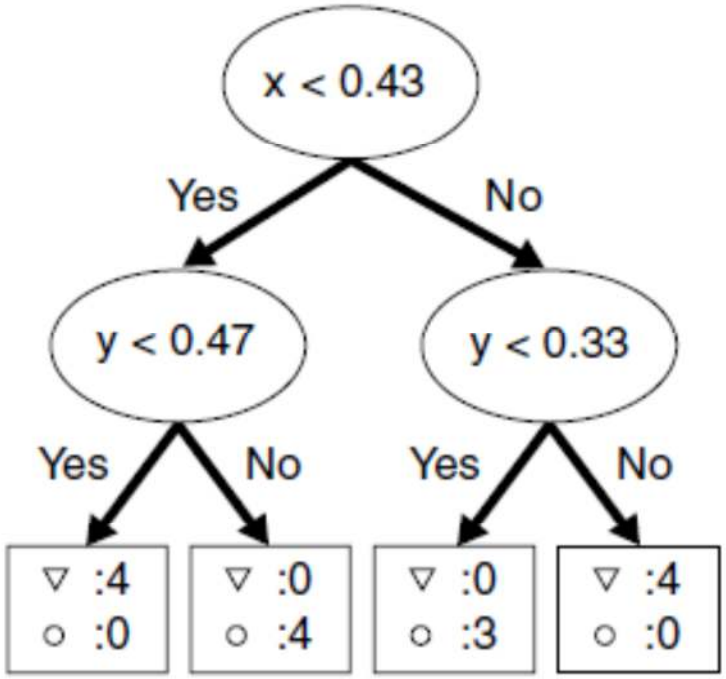
Parent Node, p is split into k partitions
n_i is the number of records in partition I
entropy of the partitioning

Characteristics

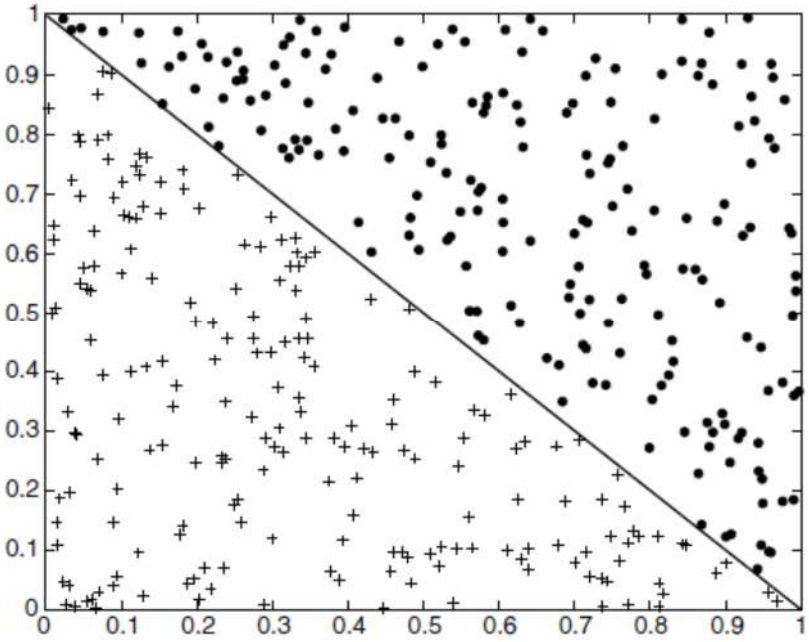
- ✓ Decision tree induction is a nonparametric approach for building classification Models
- ✓ Finding an optimal decision tree is a hard problem
- ✓ greedy approach
- ✓ relatively easy to interpret
- ✓ choice of impurity measure has little effect on the performance of decision tree induction algorithms
- ✓ using only a single attribute at a time
- ✓ partitioning the attribute space into disjoint regions until each region contains records of the same class
- ✓ parallel to the “coordinate axes.”

Characteristics





Characteristics



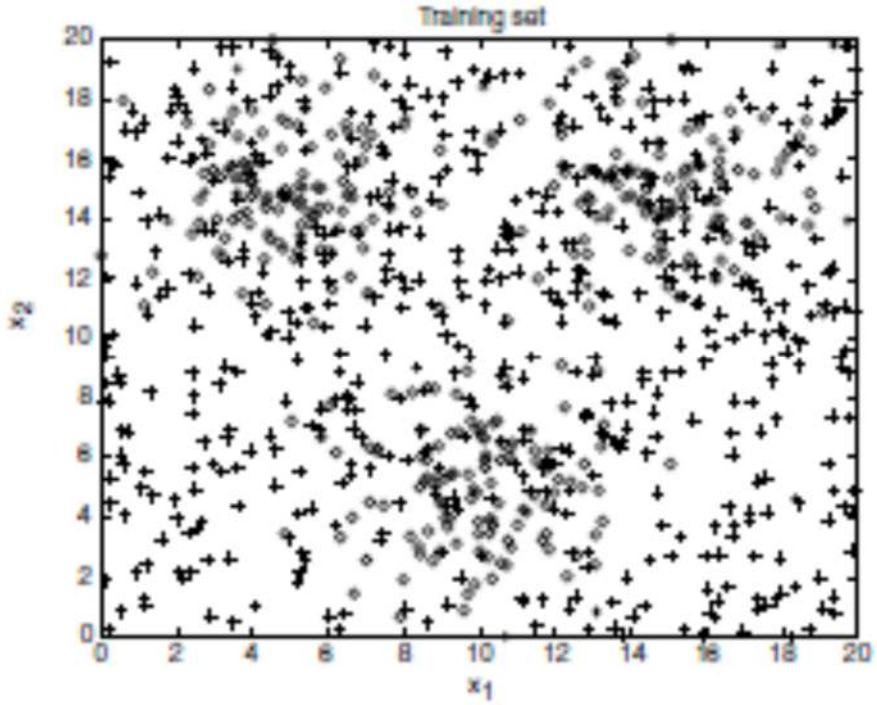
oblique decision tree

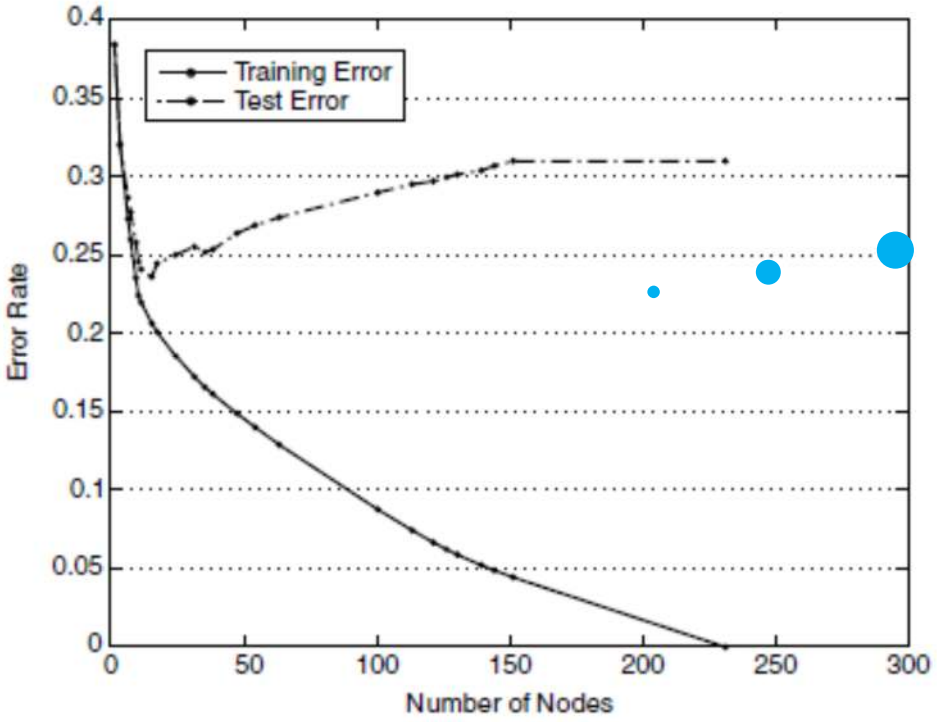
$$x + y < 1$$

Model Overfitting

Training errors
Test errors (generalization)

good classification model must not only fit training data well, it must also accurately classify records it has never

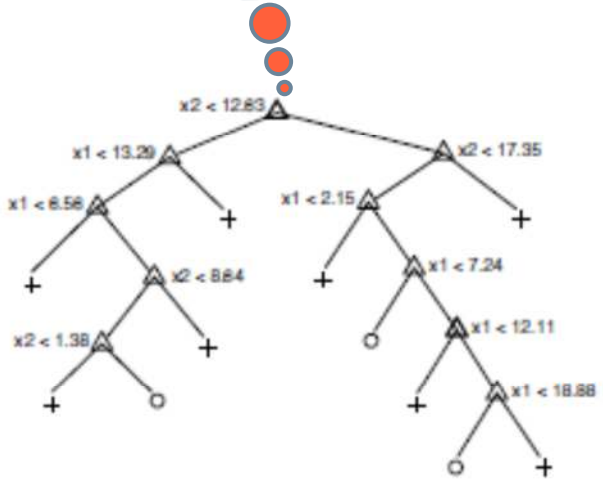




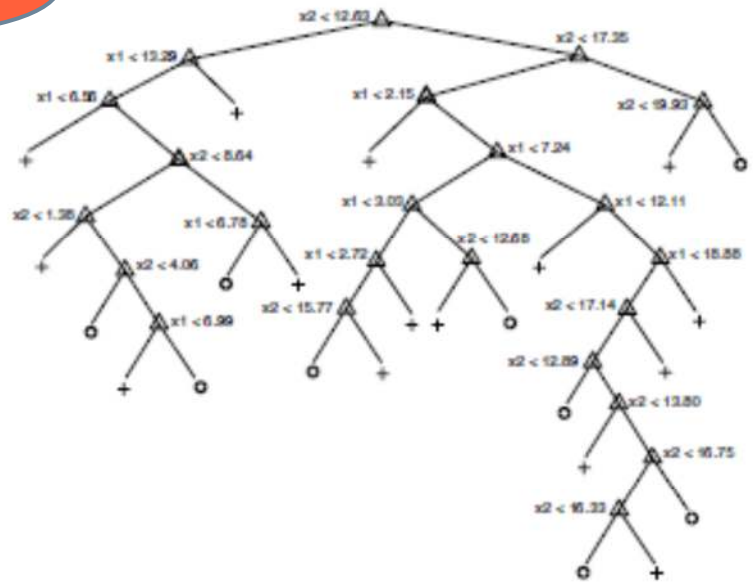
Underfitting and Overfitting

Model Overfitting

a higher training error rate, but a lower test error rate



(a) Decision tree with 11 leaf nodes.



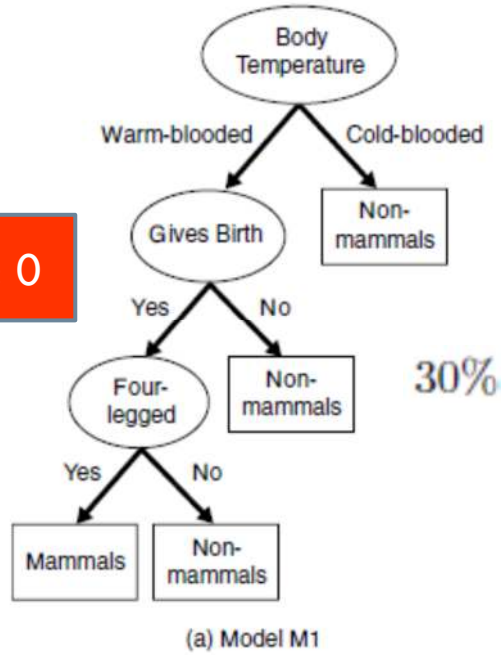
(b) Decision tree with 24 leaf nodes.

Overfitting Due to Presence of Noise

Train

Name	Body Temperature	Gives Birth	Four-legged	Hibernates	Class Label
porcupine	warm-blooded	yes	yes	yes	yes
cat	warm-blooded	yes	yes	no	yes
bat	warm-blooded	yes	no	yes	no*
whale	warm-blooded	yes	no	no	no*
salamander	cold-blooded	no	yes	yes	no
komodo dragon	cold-blooded	no	yes	no	no
python	cold-blooded	no	no	yes	no
salmon	cold-blooded	no	no	no	no
eagle	warm-blooded	no	no	no	no
guppy	cold-blooded	yes	no	no	no

Train error: 0

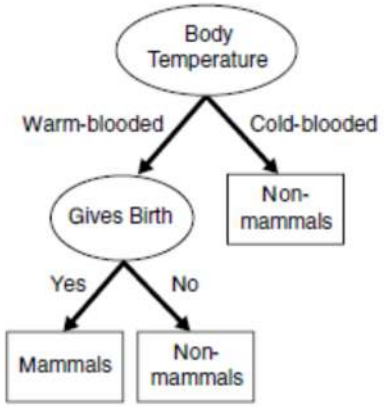


30%

(a) Model M1

Test

Name	Body Temperature	Gives Birth	Four-legged	Hibernates	Class Label
human	warm-blooded	yes	no	no	yes
pigeon	warm-blooded	no	no	no	no
elephant	warm-blooded	yes	yes	no	yes
leopard shark	cold-blooded	yes	no	no	no
turtle	cold-blooded	no	yes	no	no
penguin	cold-blooded	no	no	no	no
eel	cold-blooded	no	no	no	no
dolphin	warm-blooded	yes	no	no	yes
spiny anteater	warm-blooded	no	yes	yes	yes
gila monster	cold-blooded	no	yes	yes	no

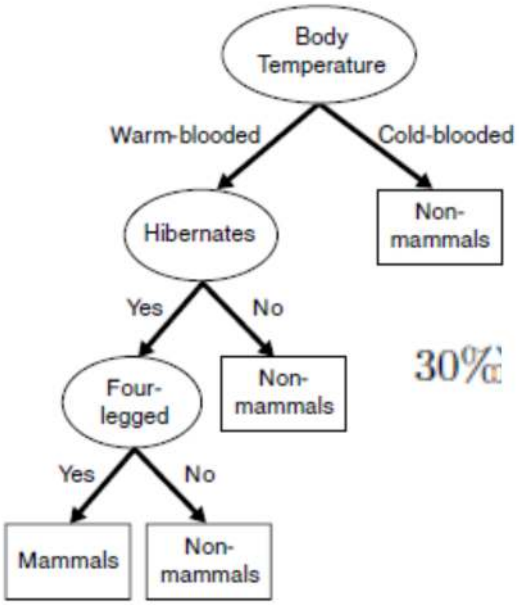


Train error: 20%

error rate (10%)

small number of training records

Name	Body Temperature	Gives Birth	Four-legged	Hibernates	Class Label
salamander	cold-blooded	no	yes	yes	no
guppy	cold-blooded	yes	no	no	no
eagle	warm-blooded	no	no	no	no
poorwill	warm-blooded	no	no	yes	no
platypus	warm-blooded	no	yes	yes	yes



30%

Decision Trees

Prepruning :Early Stopping Rule

a more restrictive stopping condition
stop expanding a leaf node when the observed gain in impurity measure is low

Post-pruning

decision tree is initially grown to its maximum size
tree-pruning step
replacing a subtree with a new leaf node

Evaluating the Performance of a Classifier

accuracy or error rate computed from the test set can used to compare different classifiers

class labels of test records must be known

Holdout Method

1. labeled examples partitioned into two disjoint sets: training and the test sets
2. classification model is then induced from the training set
3. its performance is evaluated on the test set

- ✓ smaller training set size, larger variance of the model
- ✓ training set is too large, then the estimated accuracy computed from the smaller test set is less reliable

Evaluating the Performance of a Classifier

Random Subsampling

Repeated holdout

Bootstrap

Sampling with replacement

Cross-Validation

each record is used the same number of times for training and exactly once for testing
K-fold Cross-Validation

CLASSIFICATION METHODS

Nearest-Neighbor classifiers

instance-based learning

eager learners

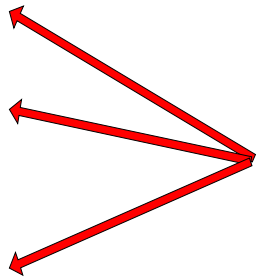
(1) constructing a classification model from data
(2) applying the model to test examples

uses specific training instances to make predictions without having to maintain an abstraction (or model) derived from data

Lazy learners

Set of Stored Cases

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



Unseen Case

Atr1	…….	AtrN

Nearest-Neighbor classifiers

Rote-learner

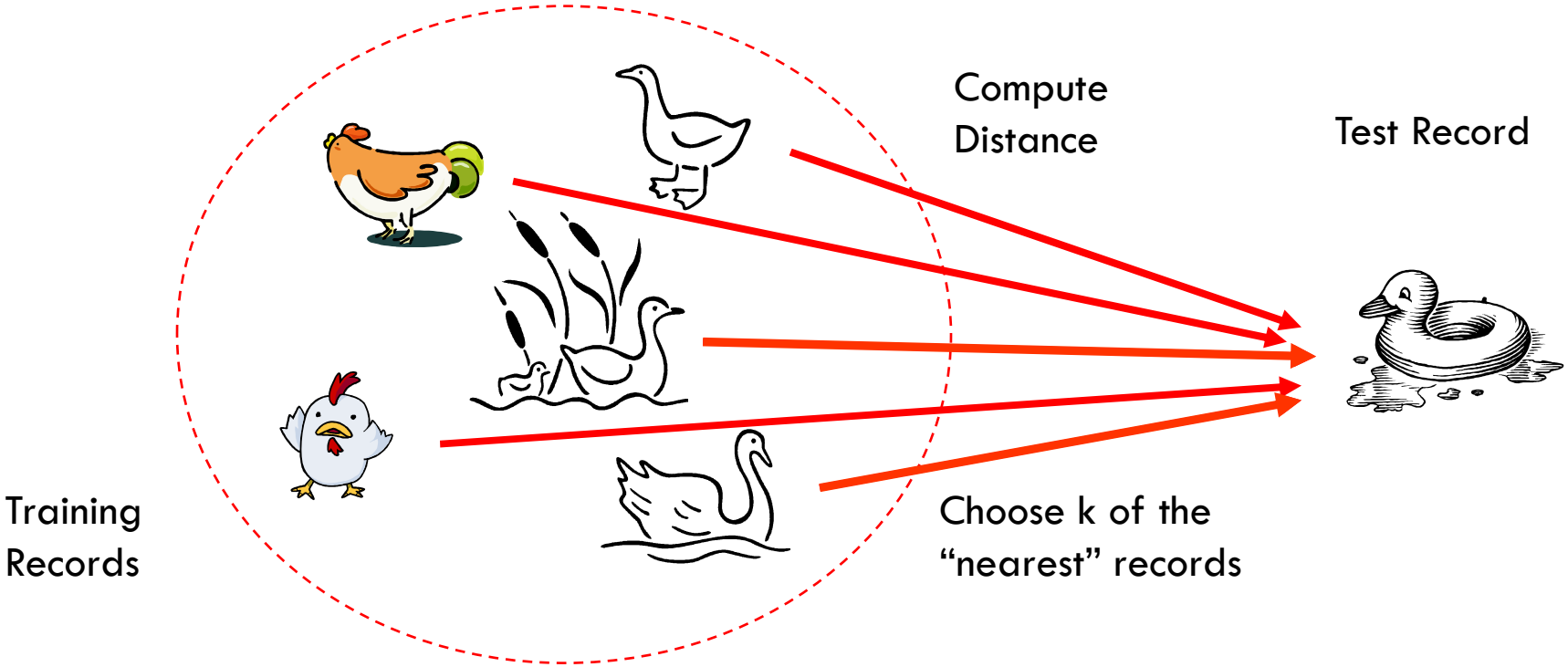
- Memorizes entire training data and performs classification only if attributes of record match one of the training examples exactly
- problem

Nearest neighbor

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

Nearest Neighbor Classifiers

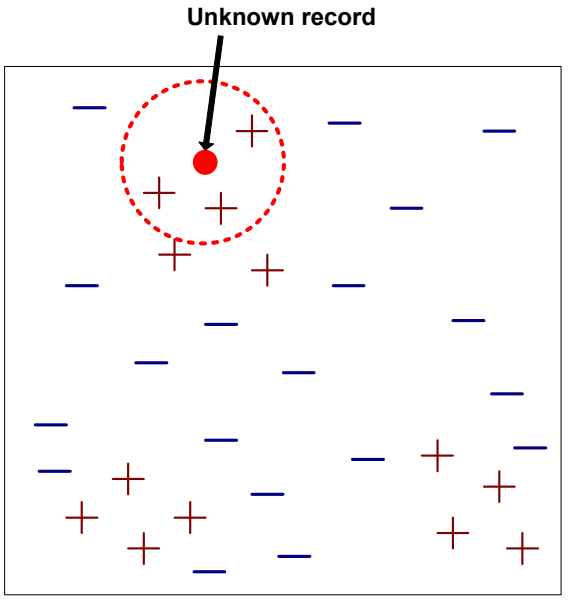
- Basic idea:
  - If it walks like a duck, quacks like a duck, then it's probably a duck



Nearest-Neighbor Classifiers

- 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)

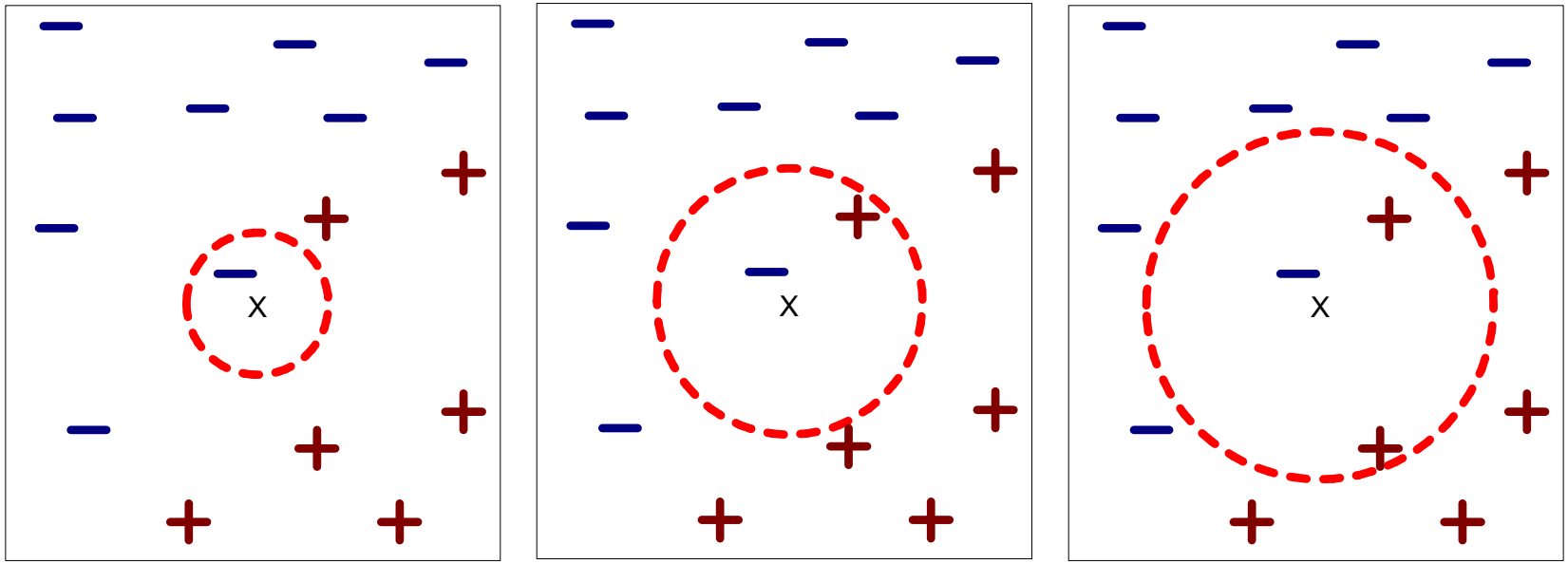


Algorithm 5.2 The k-nearest neighbor classification algorithm.

1: Let k be the number of nearest neighbors and D be the set of training examples.
2: **for** each test example $z = (\mathbf{x}', y')$ **do**
3: Compute $d(\mathbf{x}', \mathbf{x})$, the distance between z and every example, $(\mathbf{x}, y) \in D$.
4: Select $D_z \subseteq D$, the set of k closest training examples to z.
5: $y' = \underset{v}{\operatorname{argmax}} \sum_{(\mathbf{x}_i, y_i) \in D_z} I(v = y_i)$
6: **end for**

Majority Voting: $y' = \underset{v}{\operatorname{argmax}} \sum_{(\mathbf{x}_i, y_i) \in D_z} I(v = y_i)$

Definition of Nearest Neighbor



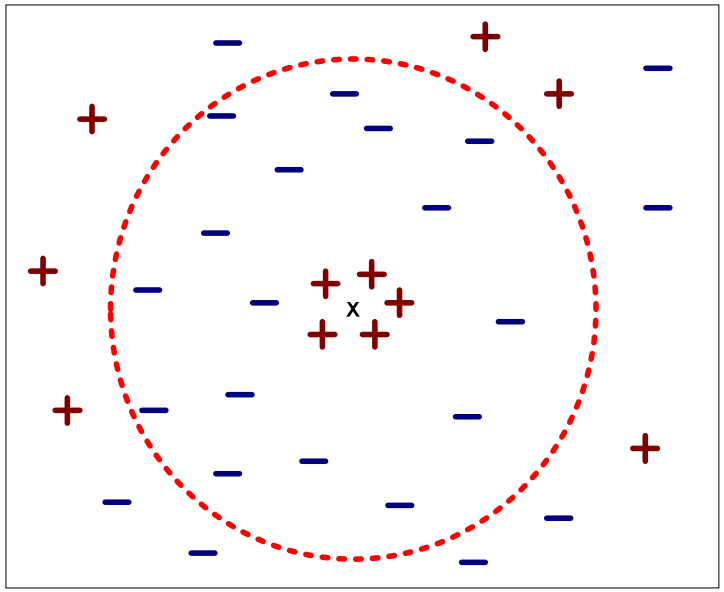
(a) 1-nearest neighbor (b) 2-nearest neighbor (c) 3-nearest neighbor

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

Nearest-Neighbor Classifiers

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 Classifiers

Distance-Weighted Voting: $y' = \underset{v}{\operatorname{argmax}} \sum_{(\mathbf{x}_i, y_i) \in D_z} w_i \times I(v = y_i)$

$$w_i = 1/d(\mathbf{x}', \mathbf{x}_i)^2$$

- ✓ Lazy learners
- ✓ Nearest-neighbor classifiers can produce arbitrarily shaped decision boundaries
- ✓ appropriate proximity measure
- ✓ data preprocessing steps

Bayesian Classifiers

Bayes Classifier

- A probabilistic framework for solving classification problems
- Conditional Probability:

$$P(C \mid A) = \frac{P(A, C)}{P(A)}$$

$$P(A \mid C) = \frac{P(A, C)}{P(C)}$$

- Bayes theorem:

$$P(C \mid A) = \frac{P(A \mid C)P(C)}{P(A)}$$

Example of Bayes Theorem

- Given:
  - A doctor knows that meningitis causes stiff neck 50% of the time
  - Prior probability of any patient having meningitis is 1/50,000
  - Prior probability of any patient having stiff neck is 1/20

- If a patient has stiff neck, what's the probability he/she has meningitis?

$$P(M \mid S) = \frac{P(S \mid M)P(M)}{P(S)} = \frac{0.5 \times 1/50000}{1/20} = 0.0002$$

Bayesian Classifiers

- Consider each attribute and class label as random variables

- Given a record with attributes (x_1, x_2,…,x_n)
  - Goal is to predict class Y
  - Specifically, we want to find the value of Y that maximizes $P(Y \mid x_1, x_2,\ldots,x_n)$

- Can we estimate $P(Y \mid x_1, x_2,\ldots,x_n)$ directly from data?

Bayesian Classifiers

- Approach:
  - compute the posterior probability $P(Y \mid x_1, x_2, \ldots, x_n)$ for all values of Y using the Bayes theorem

$$P(Y \mid x_1 x_2 \ldots x_n) = \frac{P(x_1 x_2 \ldots x_n \mid Y) P(Y)}{P(x_1 x_2 \ldots x_n)}$$

  - Choose value of Y that maximizes

    $P(Y \mid x_1, x_2, \ldots, x_n)$

  - Equivalent to choosing value of Y that maximizes

    $P(x_1, x_2, \ldots, x_n \mid Y)\ P(Y)$

- How to estimate $P(x_1, x_2, \ldots, x_n \mid Y)$?

Naïve Bayes Classifier

Naïve Bayes Classifier

- Assume independence among attributes A_i when class is given:
  - $P(x_1, x_2, \ldots, x_n \mid Y) = P(x_1 \mid Y_i)\, P(x_2 \mid Y_i) \ldots P(x_n \mid Y_i)$
  - Can estimate $P(x_i \mid Y_i)$ for all x_i and Y_i.
  - New point is classified to Y_i if $P(Y_i) \prod P(x_i \mid Y_i)$ is maximal.

How to Estimate Probabilities from Data?

Tid	Refund	Marital Status	Taxable Income	Evade
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

- Class: $P(Y_k) = N_k/N$
  - e.g., P(No) = 7/10,
    P(Yes) = 3/10
- For discrete attributes:

$$P(x_i \mid Y_k) = |x_{ik}| / N_k$$

  - where $|x_{ik}|$ is number of instances having attribute x_i and belongs to class Y_k
  - Examples:

P(Status=Married|No) = 4/7
P(Refund=Yes|Yes)=0

How to Estimate Probabilities from Data?

- For continuous attributes:
  - Discretize the range into bins
  - Probability density estimation:
    - Assume attribute follows a normal distribution
    - Use data to estimate parameters of distribution (e.g., mean and standard deviation)
    - Once probability distribution is known, can use it to estimate the conditional probability $P(x_i \mid Y)$

How to Estimate Probabilities from Data?

Tid	Refund	Marital Status	Taxable Income	Evade
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

- Normal distribution:

$$P(x_i \mid Y_j) = \frac{1}{\sqrt{2\pi\sigma_{ij}^2}} e^{-\frac{(x_i-\mu_{ij})^2}{2\sigma_{ij}^2}}$$

  - One for each (x_i, Y_i) pair

- For (Income, Class=No):
  - If Class=No
    - sample mean = 110
    - sample variance = 2975

$$P(Income = 120 \mid No) = \frac{1}{\sqrt{2\pi}(54.54)} e^{-\frac{(120-110)^2}{2(2975)}} = 0.0072$$

Bayes Error Rate

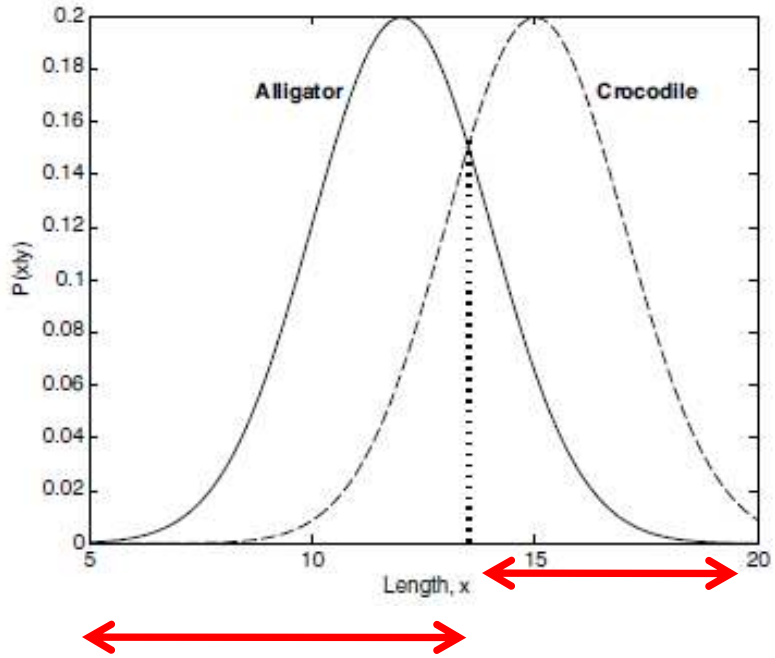
Example:

identifying alligators and crocodiles based on their lengths
their prior probabilities are the same

$$P(X|\texttt{Crocodile}) = \frac{1}{\sqrt{2\pi}\cdot 2}\exp\left[-\frac{1}{2}\left(\frac{X-15}{2}\right)^2\right]$$

$$P(X|\texttt{Alligator}) = \frac{1}{\sqrt{2\pi}\cdot 2}\exp\left[-\frac{1}{2}\left(\frac{X-12}{2}\right)^2\right]$$

Bayes Error Rate



Directed graphical models (Bayesian Belief network)

Introduction

- multiple correlated variables, such as words in a document, pixels in an image, or genes in a microarray
- compactly *represent* the **joint distribution** $p(\mathbf{x})$ (probabilistic modeling)
- *learn* the parameters of this distribution with a reasonable amount of data (learning)
- *infer* one set of variables given another (inference)

Introduction

- **Chain rule**

$$p(x_{1:V}) = p(x_1)p(x_2|x_1)p(x_3|x_2,x_1)p(x_4|x_1,x_2,x_3)\ldots p(x_V|x_{1:V-1})$$

- becomes more and more complicated to represent the conditional distributions
- suppose all the variables have *K* states
- $p(x2/x1)$, $p(x3/x1, x2)$

Conditional independence(CI)

- The key to efficiently representing large joint distributions is to make some assumptions about conditional independence

$$X \perp Y|Z \iff p(X,Y|Z) = p(X|Z)p(Y|Z)$$

$$x_{t+1} \perp \mathbf{x}_{1:t-1}|x_t$$

Markov Assumption

$$p(x_{1:V}) = p(x_1)p(x_2|x_1)p(x_3|x_2,x_1)p(x_4|x_1,x_2,x_3)\ldots p(x_V|x_{1:V-1})$$

$$p(\mathbf{x}_{1:V}) = p(x_1)\prod_{t=1}^{V} p(x_t|x_{t-1})$$

Graphical Models

- first-order Markov assumption is useful for defining distributions on 1d sequences
- A **graphical model** (**GM**) is a way to represent a joint distribution by making CI assumptions
- nodes in the graph represent random variables
- lack of edges represents CI assumptions

$$\textbf{graph } G = (\mathcal{V}, \mathcal{E})$$

$$\mathcal{V} = \{1, \ldots, V\}$$

$$\mathcal{E} = \{(s,t) : s,t \in \mathcal{V}\}$$

adjacency matrix

$G(s,t) = 1$ to denote $(s,t) \in \mathcal{E}$

Graph terminology

- **Parent** For a directed graph, the **parents** of a node is the set of all nodes that feed into it: $\mathrm{pa}(s) \triangleq \{t : G(t,s) = 1\}$.

- **Child** For a directed graph, the **children** of a node is the set of all nodes that feed out of it: $\mathrm{ch}(s) \triangleq \{t : G(s,t) = 1\}$.

- **Root** For a directed graph, a **root** is a node with no parents.

- **Path or trail** A **path** or **trail** $s \rightsquigarrow t$ is a series of directed edges leading from s to t.

Graph terminology

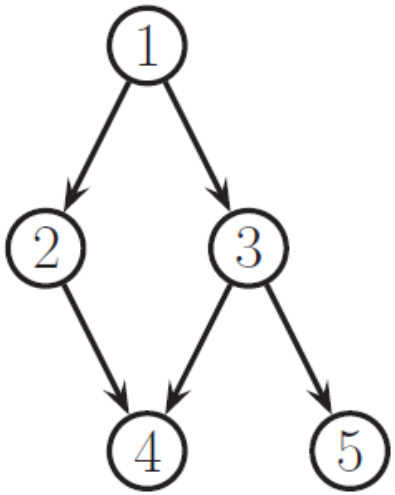
- **Ancestors** For a directed graph, the **ancestors** are the parents, grand-parents, etc of a node. That is, the ancestors of t is the set of nodes that connect to t via a trail: $\text{anc}(t) \triangleq \{s : s \rightsquigarrow t\}$.

- **Descendants** For a directed graph, the **descendants** are the children, grand-children, etc of a node. That is, the descendants of s is the set of nodes that can be reached via trails from s: $\text{desc}(s) \triangleq \{t : s \rightsquigarrow t\}$.

- **Cycle or loop** For any graph, we define a **cycle** or **loop** to be a series of nodes such that we can get back to where we started by following edges

Graph terminology

- **DAG** A **directed acyclic graph** or **DAG** is a directed graph with no directed cycles.



- **Topological ordering** For a DAG, a **topological ordering** or **total ordering** is a numbering of the nodes such that parents have lower numbers than their children.

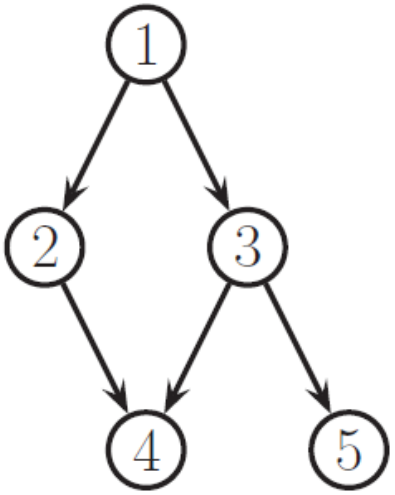
Directed graphical models

- A **directed graphical model** or **DGM** is a GM whose graph is a DAG
- Bayesian Network, Causal Network
- topological ordering
- **ordered Markov property**

$$x_s \perp \mathbf{x}_{\mathrm{pred}(s) \setminus \mathrm{pa}(s)} | \mathbf{x}_{\mathrm{pa}(s)}$$

natural generalization of the first-order Markov property

Directed graphical models

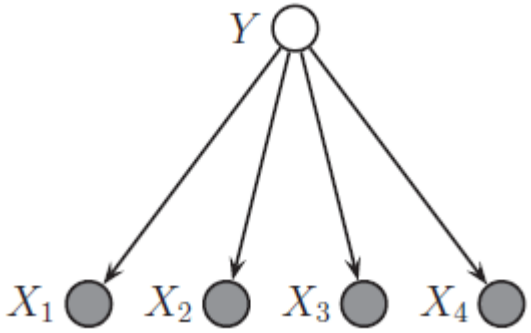


$$p(\mathbf{x}_{1:5}) = p(x_1)p(x_2|x_1)p(x_3|x_1,\cancel{x_2})p(x_4|\cancel{x_1},x_2,x_3)p(x_5|\cancel{x_1},\cancel{x_2},x_3,\cancel{x_4})$$
$$= p(x_1)p(x_2|x_1)p(x_3|x_1)p(x_4|x_2,x_3)p(x_5|x_3)$$

$$p(\mathbf{x}_{1:V}|G) = \prod_{t=1}^{V} p(x_t|\mathbf{x}_{\mathrm{pa}(t)})$$

Example:Naive Bayes classifiers

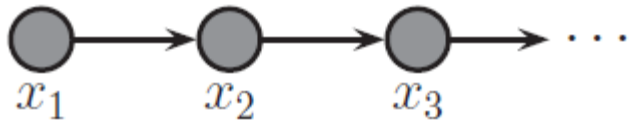
features are conditionally independent given the class label



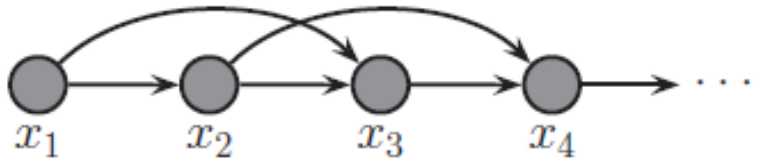
$$p(y, \mathbf{x}) = p(y) \prod_{j=1}^{D} p(x_j|y)$$

Example: Markov Chain

first-order Markov chain

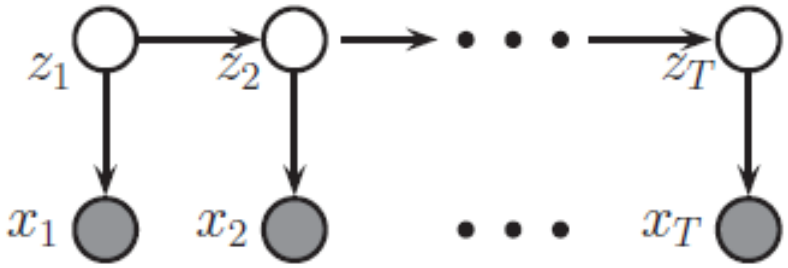


second order Markov chain



$$p(\mathbf{x}_{1:T}) = p(x_1, x_2)p(x_3|x_1, x_2)p(x_4|x_2, x_3)\ldots = p(x_1, x_2)\prod_{t=3}^{T} p(x_t|x_{t-1}, x_{t-2})$$

Example: HMM



- hidden variables often represent quantities of interest
- estimate the hidden state given the data
- another form of probabilistic inference

Inference

- ✓ graphical models : compact way to define joint probability distributions
- ✓ The main use for such a joint distribution is to perform **probabilistic inference**

task of estimating unknown quantities from known quantities

set of correlated random variables with joint distribution

$$p(\mathbf{x}_{1:V}|\boldsymbol{\theta})$$

visible variables and hidden variables

Inference

$$p(\mathbf{x}_h|\mathbf{x}_v,\theta) = \frac{p(\mathbf{x}_h,\mathbf{x}_v|\theta)}{p(\mathbf{x}_v|\theta)} = \frac{p(\mathbf{x}_h,\mathbf{x}_v|\theta)}{\sum_{\mathbf{x}'_h} p(\mathbf{x}'_h,\mathbf{x}_v|\theta)}$$

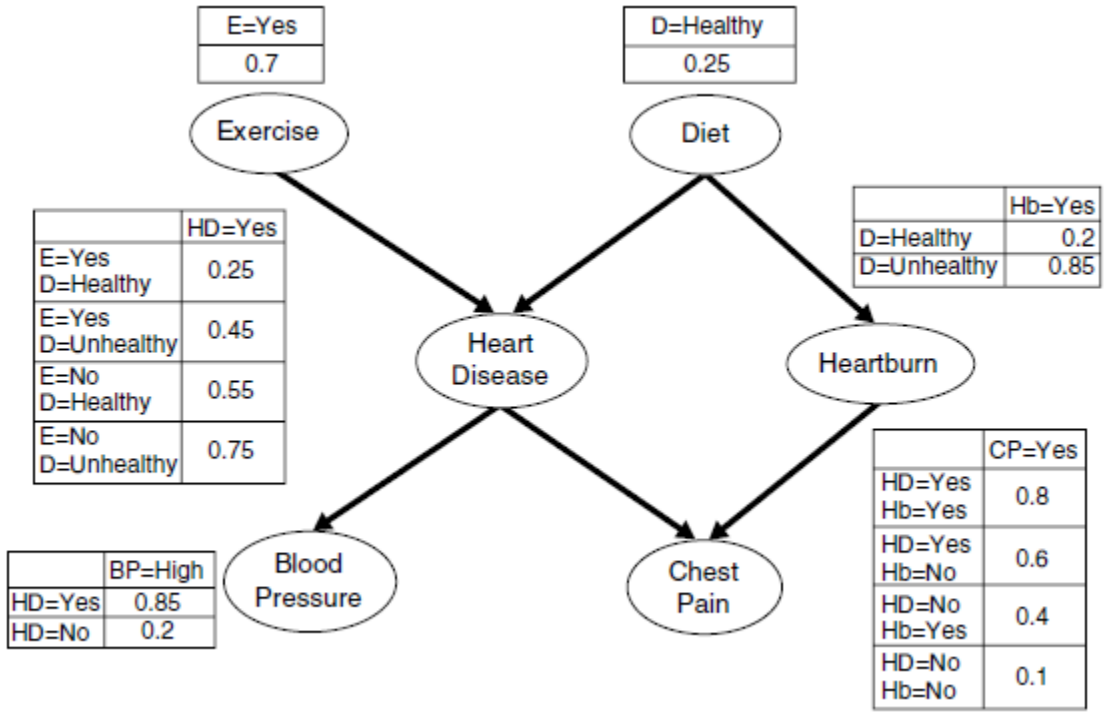
In Classification : class label :Y

Model Paramters

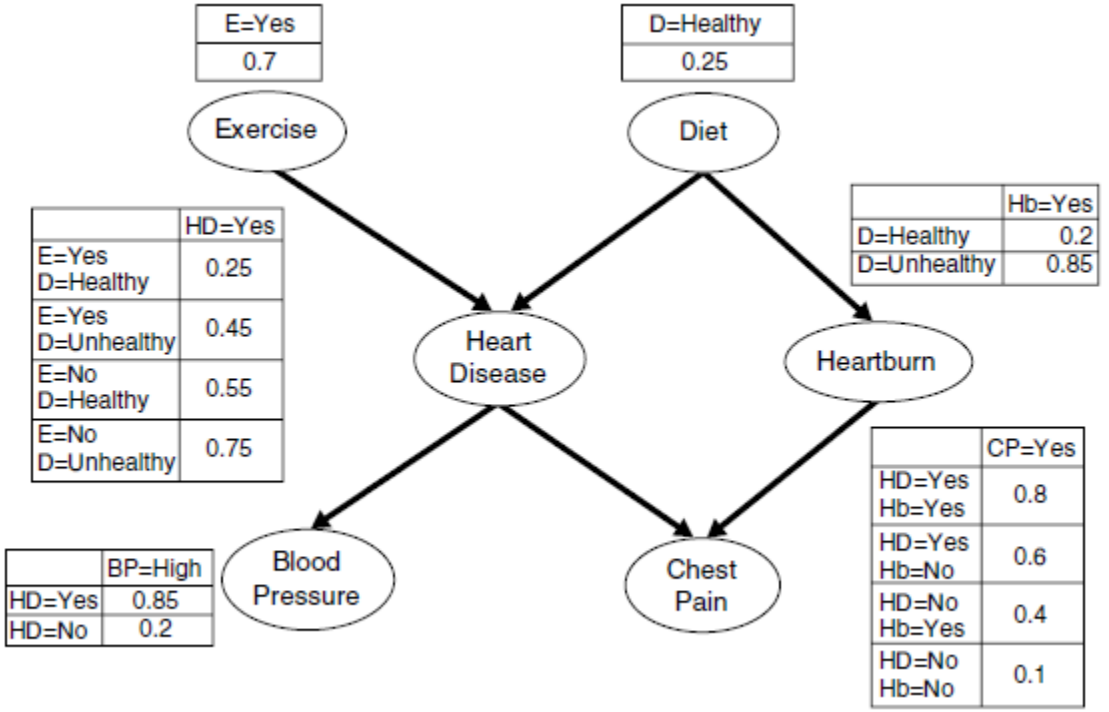
Example of Inferencing Using

Case 1: No Prior Information

Without any prior information, we can determine whether the person is likely to have heart disease



Example of Inferencing Using

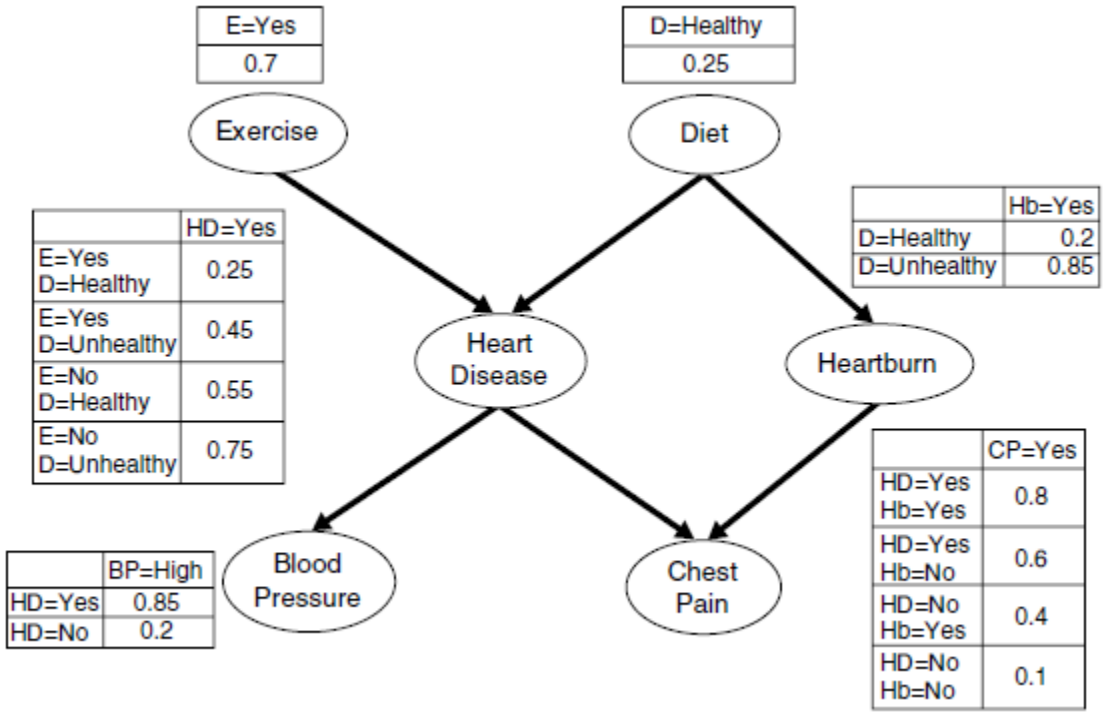


$$
\begin{aligned}
P(\texttt{HD} = \texttt{Yes}) &= \sum_{\alpha}\sum_{\beta} P(\texttt{HD} = \texttt{Yes}|E = \alpha, D = \beta)P(E = \alpha, D = \beta) \\
&= \sum_{\alpha}\sum_{\beta} P(\texttt{HD} = \texttt{Yes}|E = \alpha, D = \beta)P(E = \alpha)P(D = \beta) \\
&= 0.25 \times 0.7 \times 0.25 + 0.45 \times 0.7 \times 0.75 + 0.55 \times 0.3 \times 0.25 \\
&\quad + 0.75 \times 0.3 \times 0.75 \\
&= 0.49.
\end{aligned}
$$

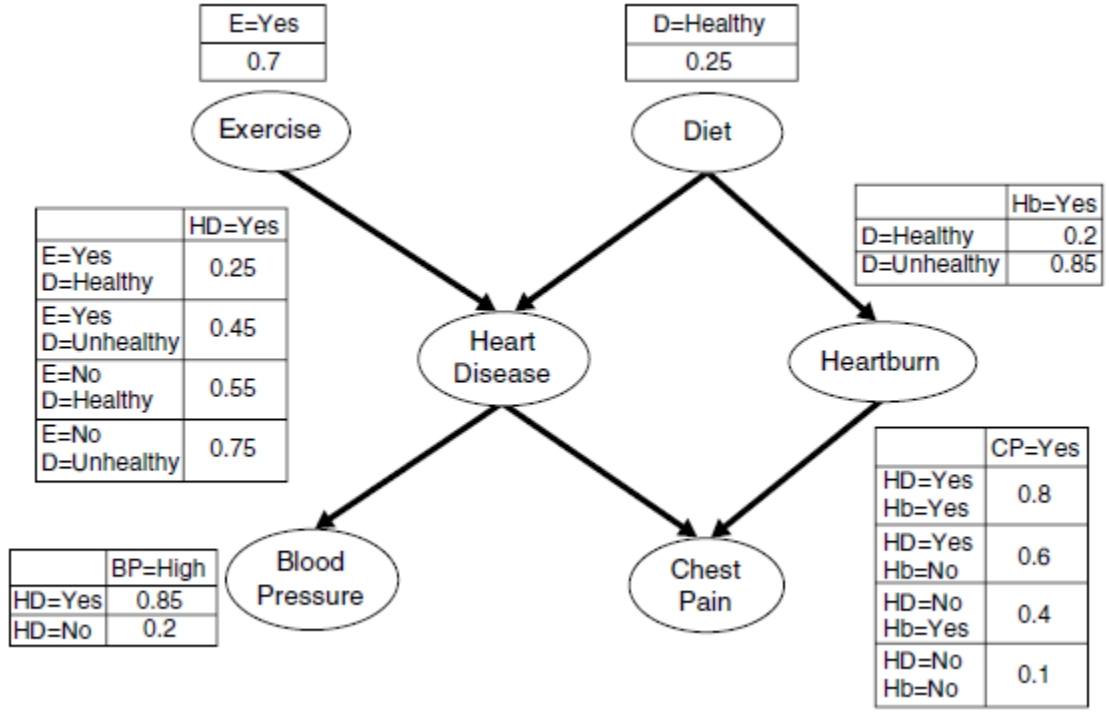
Example

Case 2: High Blood Pressure

If the person has high blood pressure, we can make a diagnosis about heart disease



Example



$$P(\texttt{HD} = \texttt{Yes}|\texttt{BP} = \texttt{High})$$

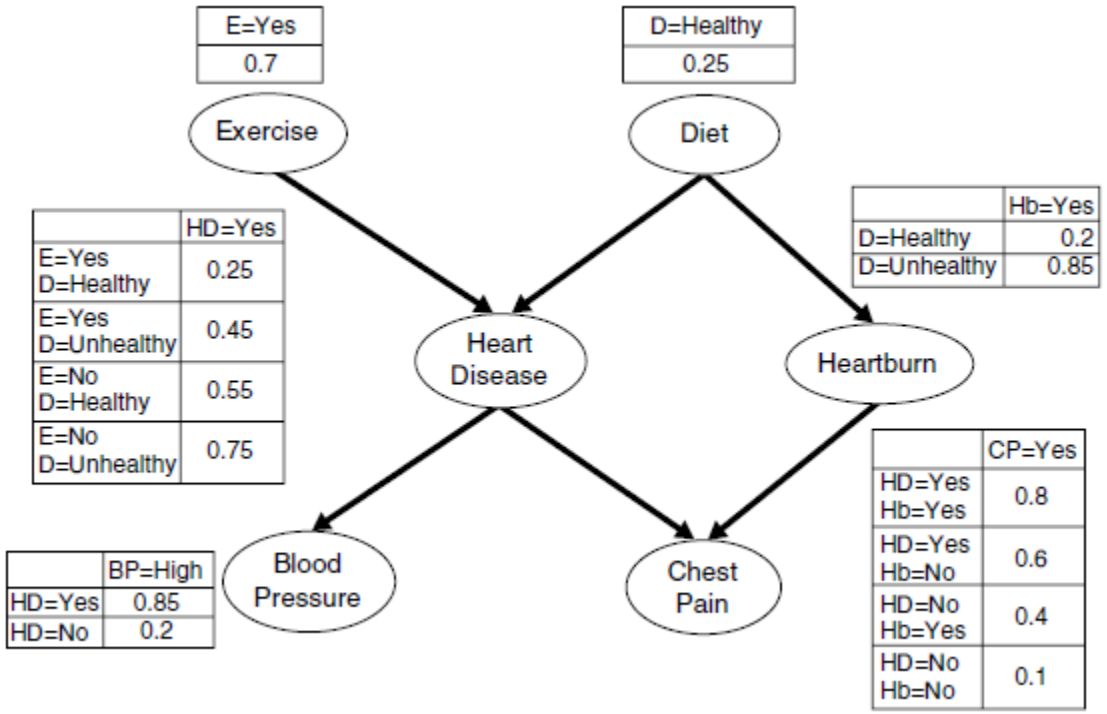
$$\begin{aligned} P(\texttt{BP} = \texttt{High}) &= \sum_{\gamma} P(\texttt{BP} = \texttt{High}|\texttt{HD} = \gamma) P(\texttt{HD} = \gamma) \\ &= 0.85 \times 0.49 + 0.2 \times 0.51 = 0.5185. \end{aligned}$$

$$\begin{aligned} &= \frac{P(\texttt{BP} = \texttt{High}|\texttt{HD} = \texttt{Yes}) P(\texttt{HD} = \texttt{Yes})}{P(\texttt{BP} = \texttt{High})} \\ &= \frac{0.85 \times 0.49}{0.5185} = 0.8033. \end{aligned}$$

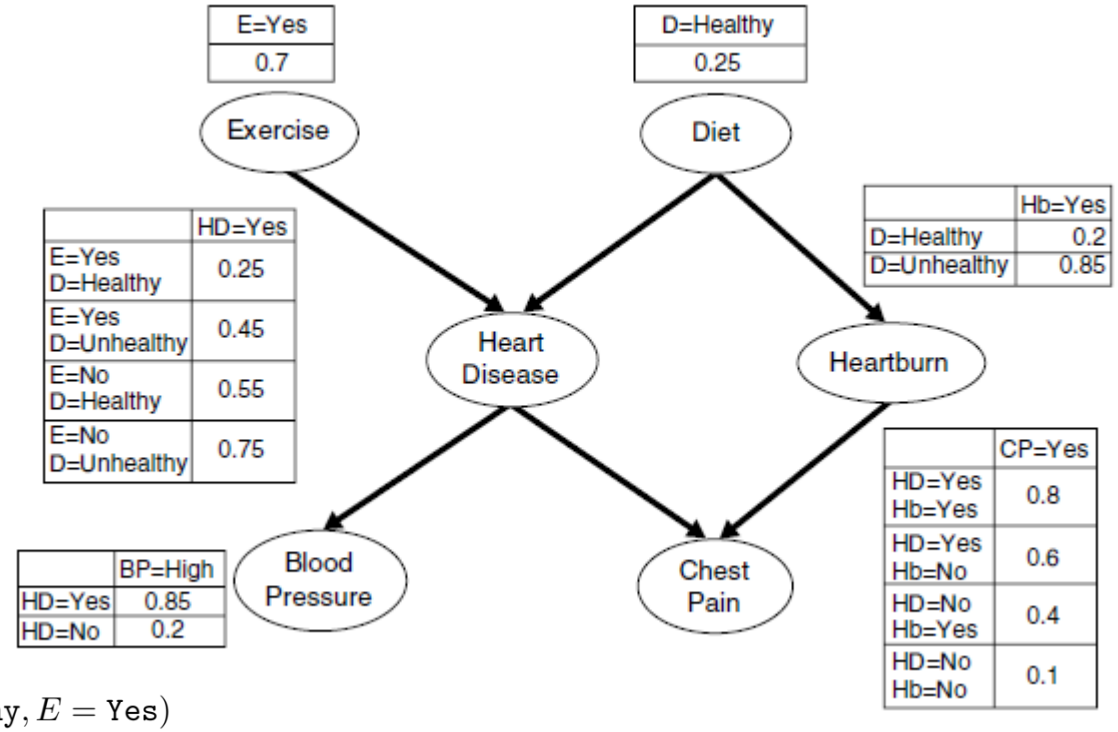
Example

Case 3: High Blood Pressure, Healthy Diet, and Regular Exercise

Suppose we are told that the person exercises regularly and eats a healthy diet. How does the new information affect our diagnosis?



Example



$$
\begin{aligned}
& P(\texttt{HD} = \texttt{Yes} | \texttt{BP} = \texttt{High}, D = \texttt{Healthy}, E = \texttt{Yes}) \\
= & \left[\frac{P(\texttt{BP} = \texttt{High} | \texttt{HD} = \texttt{Yes}, D = \texttt{Healthy}, E = \texttt{Yes})}{P(\texttt{BP} = \texttt{High} | D = \texttt{Healthy}, E = \texttt{Yes})} \right] \\
& \times P(\texttt{HD} = \texttt{Yes} | D = \texttt{Healthy}, E = \texttt{Yes}) \\
= & \frac{P(\texttt{BP} = \texttt{High} | \texttt{HD} = \texttt{Yes}) P(\texttt{HD} = \texttt{Yes} | D = \texttt{Healthy}, E = \texttt{Yes})}{\sum_\gamma P(\texttt{BP} = \texttt{High} | \texttt{HD} = \gamma) P(\texttt{HD} = \gamma | D = \texttt{Healthy}, E = \texttt{Yes})} \\
= & \frac{0.85 \times 0.25}{0.85 \times 0.25 + 0.2 \times 0.75} \\
= & \ 0.5862,
\end{aligned}
$$

Support Vector Machine (SVM)

Hyperplanes

A *hyperplane* is a set of the form

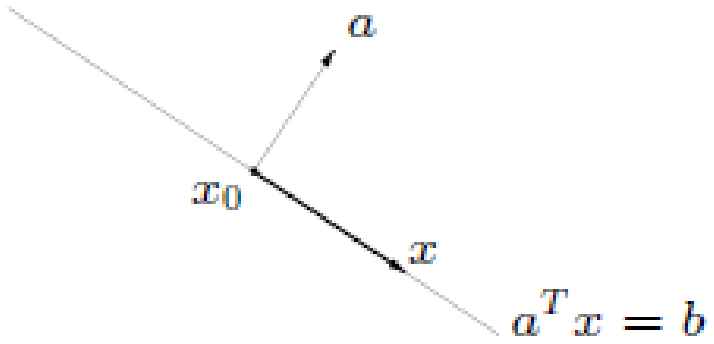
$$\{x \mid a^T x = b\},$$

$a \in \mathbf{R}^n$, $a \neq 0$, and $b \in \mathbf{R}$.

a is the normal vector

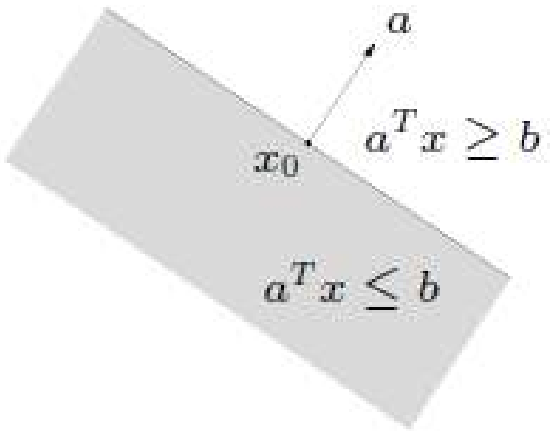
Geometrical interpretation

x_0 is any point in the hyperplane $a^T x_0 = b$ $\{x \mid a^T(x - x_0) = 0\},$

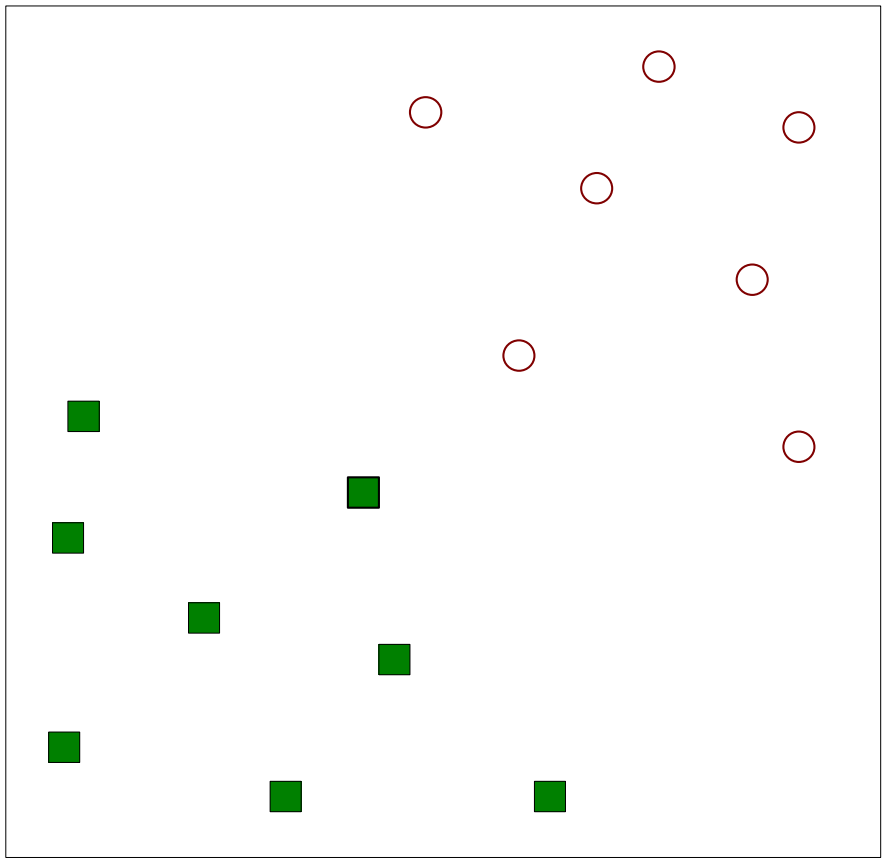


halfspaces

halfspace: set of the form $\{x \mid a^T x \leq b\}$ $(a \neq 0)$

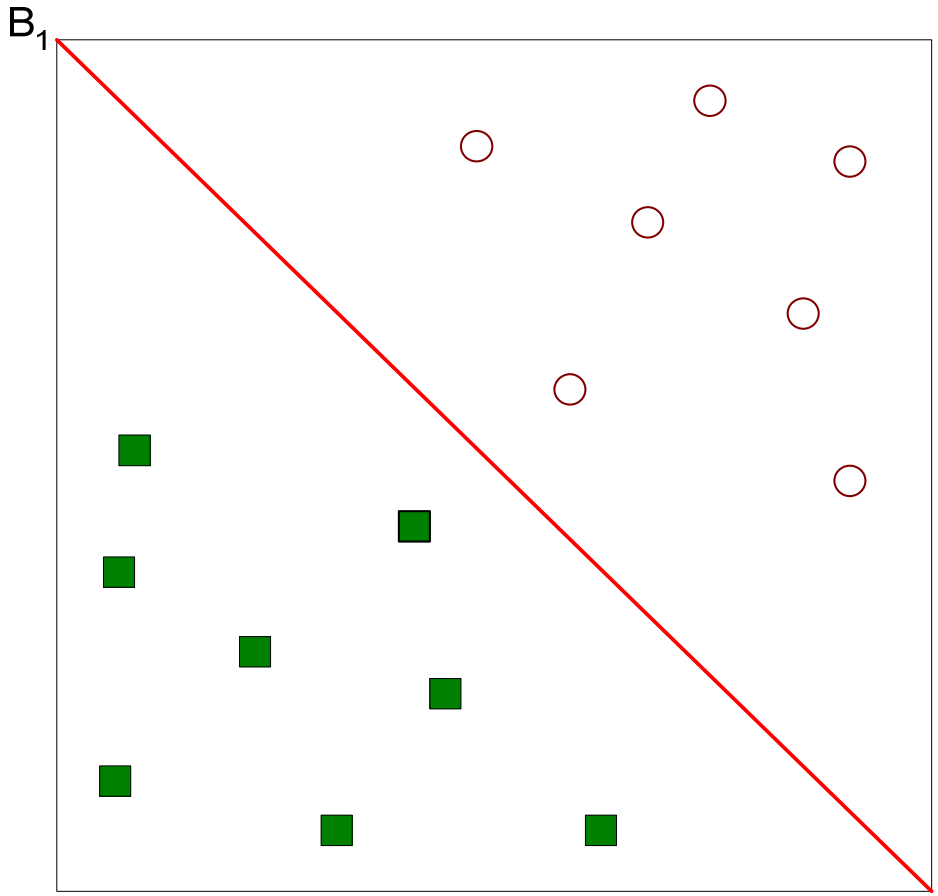


Maximum Margin Hyperplanes



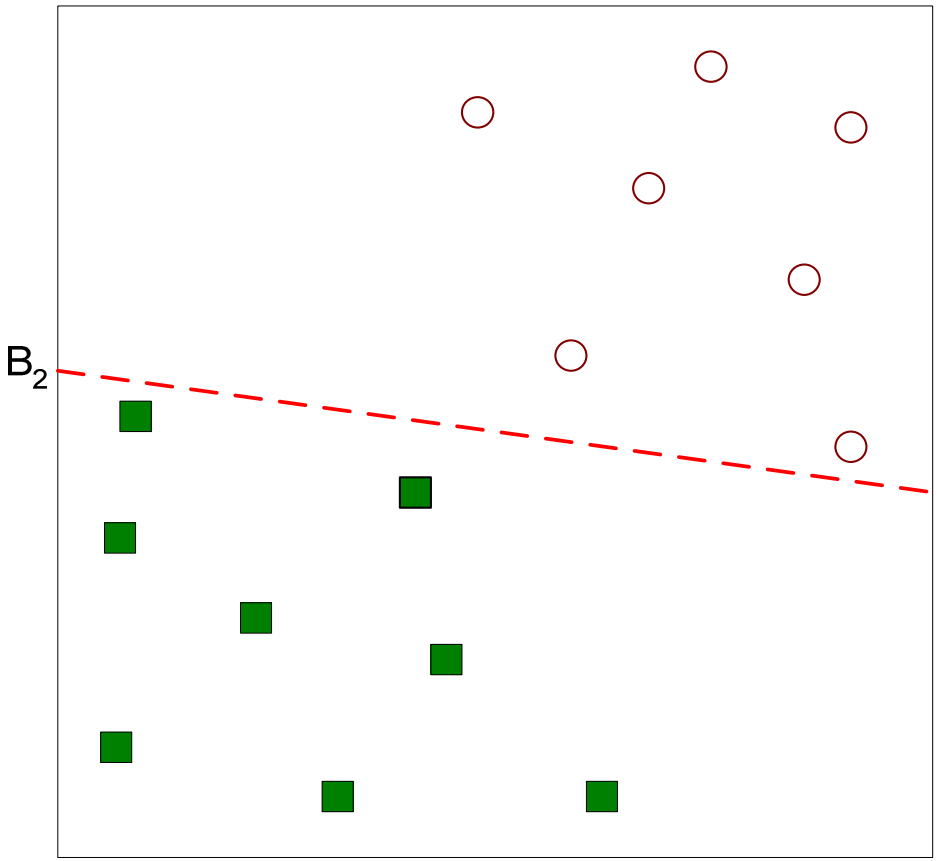
- Find a linear hyperplane (decision boundary) that will separate the data

Maximum Margin Hyperplanes



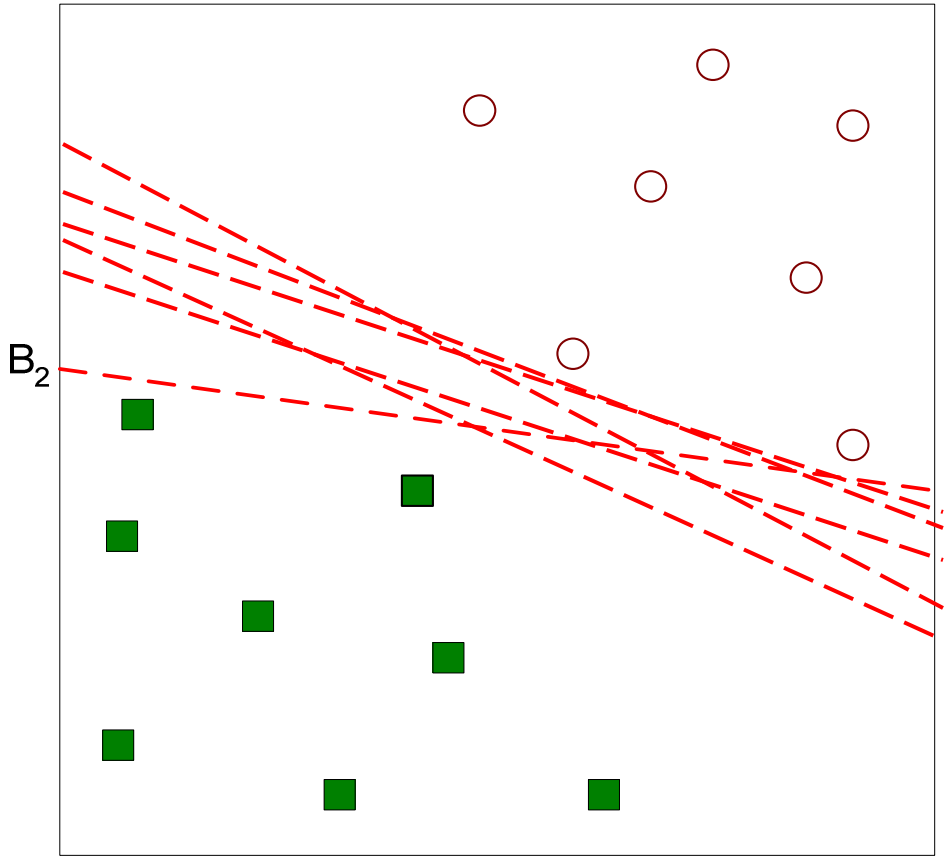
- One Possible Solution

Maximum Margin Hyperplanes



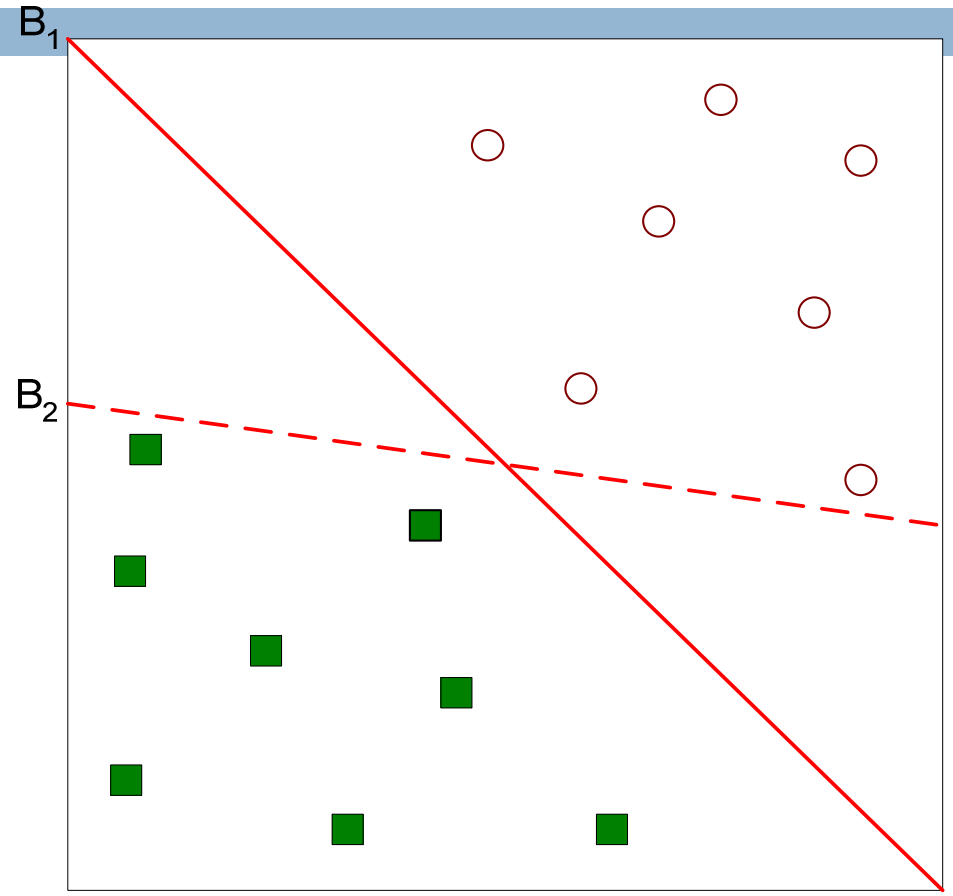
- Another possible solution

Maximum Margin Hyperplanes



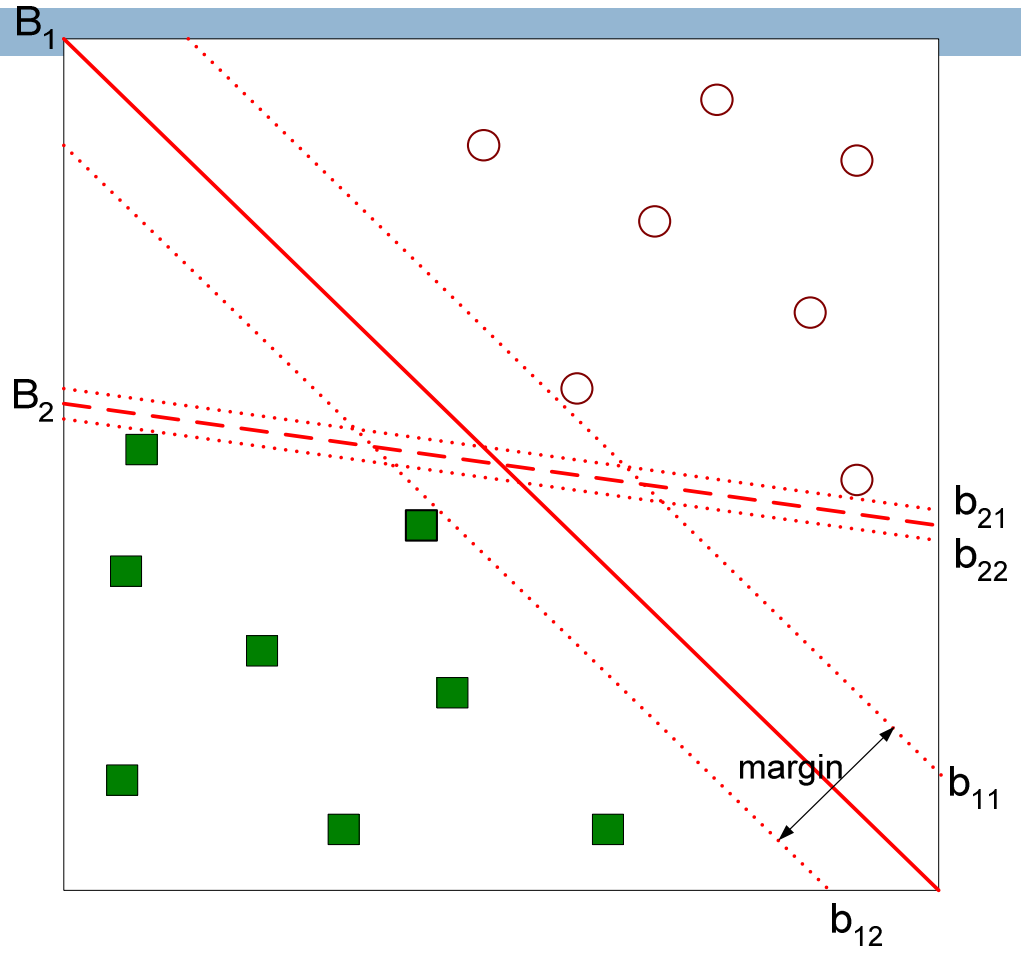
- Other possible solutions

Maximum Margin Hyperplanes



- Which one is better? B1 or B2?
- How do you define better?

Maximum Margin Hyperplanes



- Find hyperplane maximizes the margin => B1 is better than B2
- Generalization error

Linear SVM: Separable Case

A linear SVM is a classifier that searches for a hyperplane with the largest margin

$$(\mathbf{x_i}, y_i) \quad (i = 1, 2, \ldots, N)$$

$$(x_{i1}, x_{i2}, \ldots, x_{id})^T \qquad y_i \in \{-1, 1\}$$

decision boundary of a linear classifier

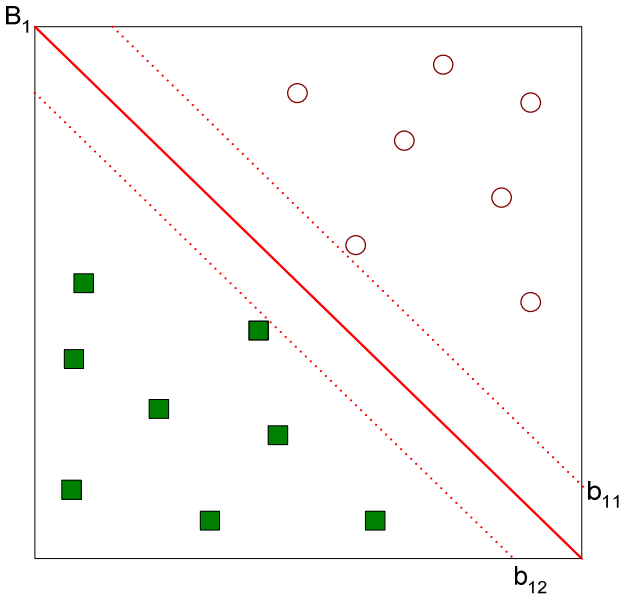
$$\mathbf{w} \cdot \mathbf{x} + b = 0,$$

w and *b* are parameters of the model

$$\mathbf{w} \cdot \mathbf{x}_s + b = k, \qquad k > 0.$$
$$\mathbf{w} \cdot \mathbf{x}_c + b = k', \qquad k' < 0.$$

$$y = \begin{cases} 1, & \text{if } \mathbf{w} \cdot \mathbf{z} + b > 0; \\ -1, & \text{if } \mathbf{w} \cdot \mathbf{z} + b < 0. \end{cases}$$

Linear SVM: Separable Case



$$b_{i1}: \ \mathbf{w} \cdot \mathbf{x} + b = 1,$$
$$b_{i2}: \ \mathbf{w} \cdot \mathbf{x} + b = -1.$$

margin of the decision boundary is given by the distance between these two hyperplanes

$$d = \frac{2}{\|\mathbf{w}\|}$$

SVM

$$\mathbf{w} \cdot \mathbf{x_i} + b \geq 1 \text{ if } y_i = 1,$$
$$\mathbf{w} \cdot \mathbf{x_i} + b \leq -1 \text{ if } y_i = -1.$$

$$y_i(\mathbf{w} \cdot \mathbf{x_i} + b) \geq 1, \quad i = 1, 2, \ldots, N.$$

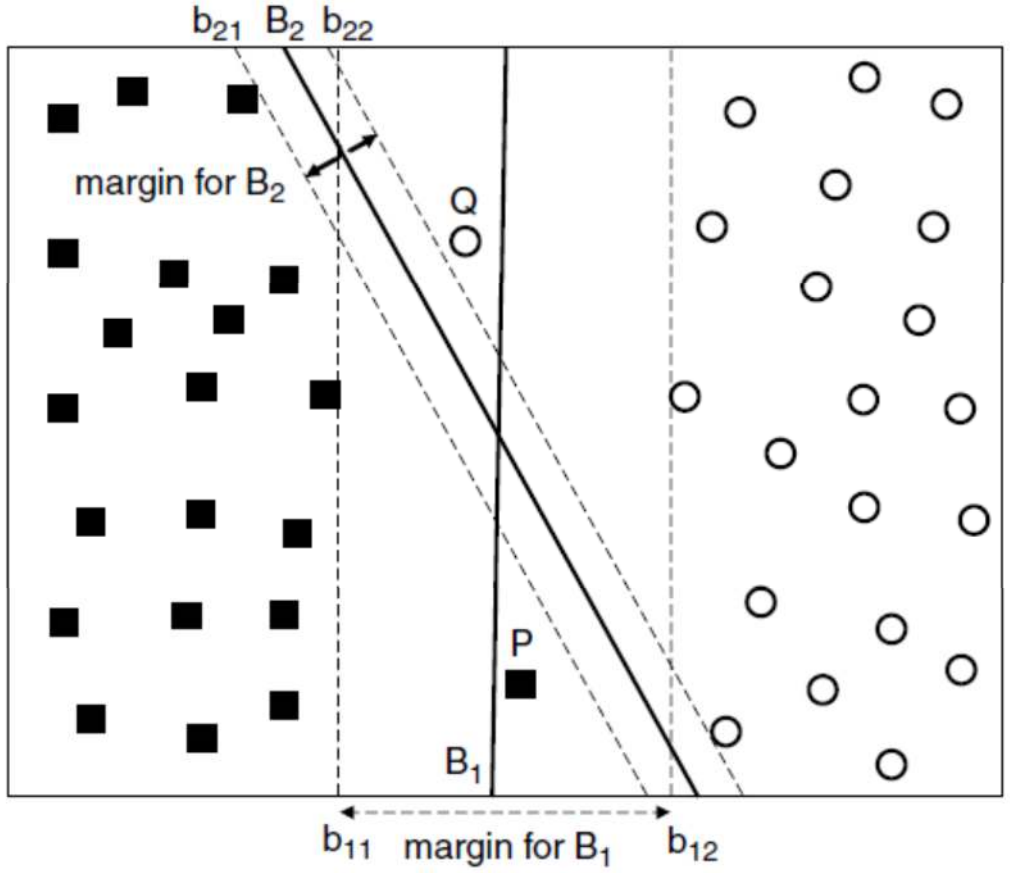
Definition 5.1 (Linear SVM: Separable Case). The learning task in SVM can be formalized as the following constrained optimization problem:

$$\min_{\mathbf{w}} \frac{\|\mathbf{w}\|^2}{2}$$
$$\text{subject to} \quad y_i(\mathbf{w} \cdot \mathbf{x_i} + b) \geq 1, \quad i = 1, 2, \ldots, N.$$

This is a constrained optimization problem

Numerical approaches to solve it (e.g., quadratic programming)

Linear SVM: Nonseparable Case

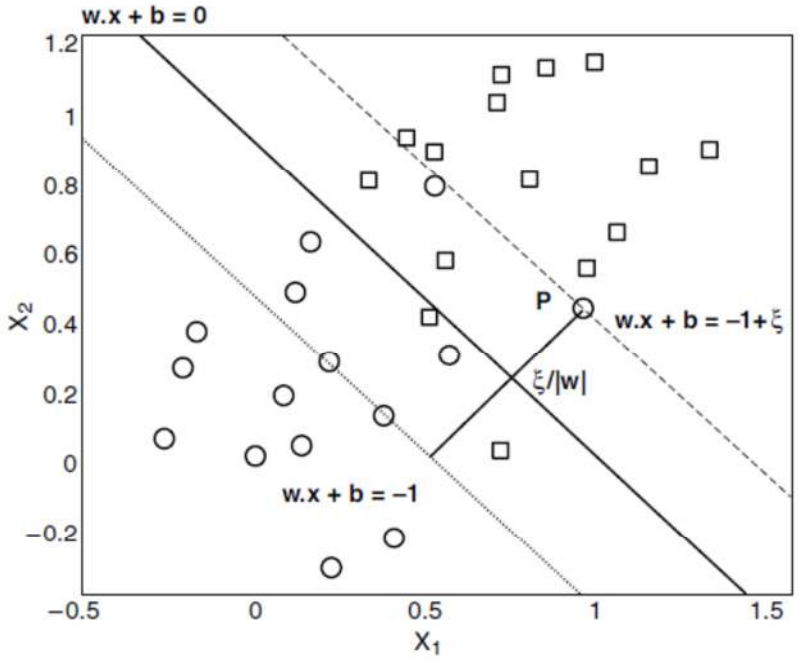


while B_2 classifies them correctly, this does not mean that B_2 is a better decision boundary than B_1

Linear SVM: Nonseparable Case

- ✓ learn a decision boundary that is tolerable to small training errors
- ✓ trade-off between the width of the margin and the number of training errors
- ✓ construct a linear decision boundary even in situations where the classes are not linearly separable
- ✓ inequality constraints must therefore be relaxed to accommodate the nonlinearly separable data

$$\mathbf{w} \cdot \mathbf{x_i} + b \geq 1 - \xi_i \quad \text{if } y_i = 1,$$
$$\mathbf{w} \cdot \mathbf{x_i} + b \leq -1 + \xi_i \quad \text{if } y_i = -1,$$



ξ provides an estimate of the error of the decision boundary

Linear SVM: Nonseparable Case

$$\min_{\mathbf{w}} \frac{\|\mathbf{w}\|^2}{2}$$

$$\mathbf{w} \cdot \mathbf{x_i} + b \geq 1 - \xi_i \quad \text{if } y_i = 1,$$
$$\mathbf{w} \cdot \mathbf{x_i} + b \leq -1 + \xi_i \quad \text{if } y_i = -1,$$

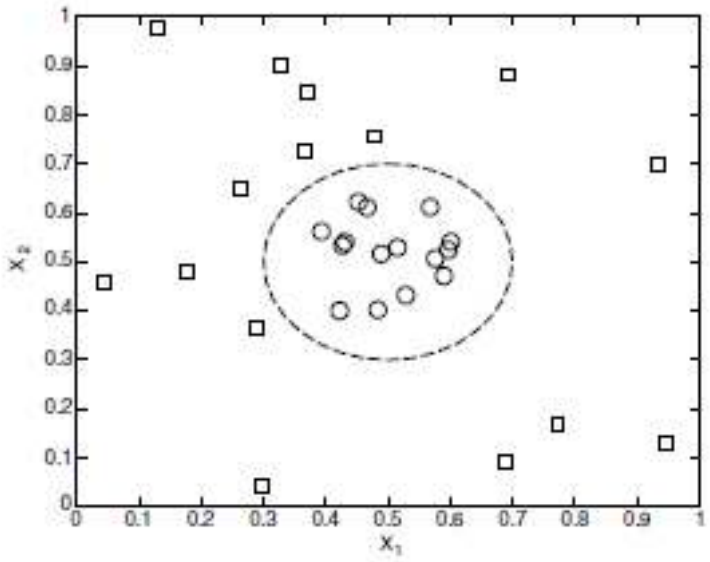
problem

$$f(\mathbf{w}) = \frac{\|\mathbf{w}\|^2}{2} + C(\sum_{i=1}^{N} \xi_i)^k,$$

C and *k* are user-specified parameters

Nonlinear SVM

✓applying SVM to data sets that have nonlinear decision boundaries
✓transform the data **x into a new space Φ(x) so that a linear decision** boundary can be used to separate the instances in the transformed space



(a) Decision boundary in the original two-dimensional space.

$$y(x_1, x_2) = \begin{cases} 1 & \text{if } \sqrt{(x_1 - 0.5)^2 + (x_2 - 0.5)^2} > 0.2, \\ -1 & \text{otherwise.} \end{cases}$$

$$x_1^2 - x_1 + x_2^2 - x_2 = -0.46$$

$$\Phi : (x_1, x_2) \longrightarrow (x_1^2, x_2^2, \sqrt{2}x_1, \sqrt{2}x_2, 1).$$

$$\mathbf{w} = (w_0, w_1, \ldots, w_4)$$

$$w_4 x_1^2 + w_3 x_2^2 + w_2 \sqrt{2} x_1 + w_1 \sqrt{2} x_2 + w_0 = 0.$$

Nonlinear SVM

Definition 5.2 (Nonlinear SVM). The learning task for a nonlinear SVM can be formalized as the following optimization problem:

$$\min_{\mathbf{w}} \frac{\|\mathbf{w}\|^2}{2}$$

$$\text{subject to} \quad y_i(\mathbf{w} \cdot \Phi(\mathbf{x}_i) + b) \geq 1, \quad i = 1, 2, \ldots, N.$$

$$\begin{aligned}
\Phi(\mathbf{u}) \cdot \Phi(\mathbf{v}) &= (u_1^2, u_2^2, \sqrt{2}u_1, \sqrt{2}u_2, 1) \cdot (v_1^2, v_2^2, \sqrt{2}v_1, \sqrt{2}v_2, 1) \\
&= u_1^2 v_1^2 + u_2^2 v_2^2 + 2u_1 v_1 + 2u_2 v_2 + 1 \\
&= (\mathbf{u} \cdot \mathbf{v} + 1)^2.
\end{aligned}$$

$$K(\mathbf{u}, \mathbf{v}) = \Phi(\mathbf{u}) \cdot \Phi(\mathbf{v}) = (\mathbf{u} \cdot \mathbf{v} + 1)^2$$

Nonlinear SVM

The main requirement for the kernel function used in nonlinear SVM is that there must exist a corresponding transformation such that the kernel function computed for a pair of vectors is equivalent to the dot product between the vectors in the transformed space.

$$K(\mathbf{x}, \mathbf{y}) = (\mathbf{x} \cdot \mathbf{y} + 1)^p$$
$$K(\mathbf{x}, \mathbf{y}) = e^{-\|\mathbf{x}-\mathbf{y}\|^2/(2\sigma^2)}$$
$$K(\mathbf{x}, \mathbf{y}) = \tanh(k\mathbf{x} \cdot \mathbf{y} - \delta)$$

Artificial Neural Network (ANN)

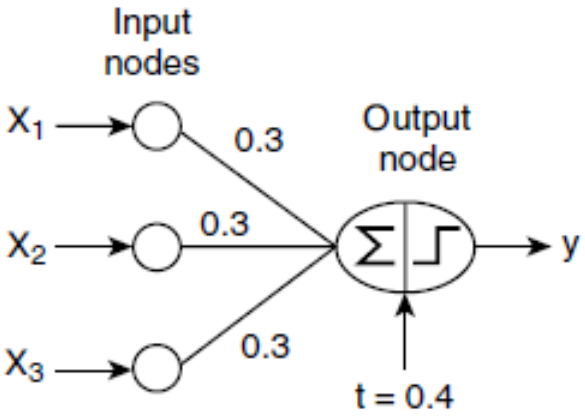
Perceptron

inspired by attempts to simulate biological neural systems

X_1	X_2	X_3	y
1	0	0	−1
1	0	1	1
1	1	0	1
1	1	1	1
0	0	1	−1
0	1	0	−1
0	1	1	1
0	0	0	−1

(a) Data set.

$$\hat{y} = \begin{cases} 1, & \text{if } 0.3x_1 + 0.3x_2 + 0.3x_3 - 0.4 > 0; \\ -1, & \text{if } 0.3x_1 + 0.3x_2 + 0.3x_3 - 0.4 < 0. \end{cases}$$



(b) Perceptron.

$x_1 = 1, x_2 = 1, x_3 = 0$, then $\hat{y} = +1$

Perceptron

$$\hat{y} = sign(w_d x_d + w_{d-1} x_{d-1} + \ldots + w_2 x_2 + w_1 x_1 - t)$$

$$\hat{y} = sign[w_d x_d + w_{d-1} x_{d-1} + \ldots + w_1 x_1 + w_0 x_0] = sign(\mathbf{w} \cdot \mathbf{x})$$

Activation Function

Learning Perceptron Model

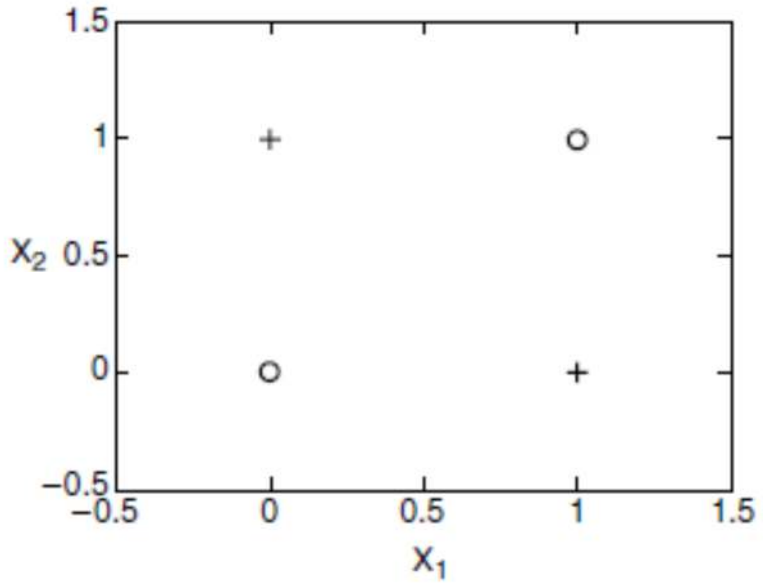
Algorithm 5.4 Perceptron learning algorithm.

1: Let $D = \{(\mathbf{x}_i, y_i) \mid i = 1, 2, \ldots, N\}$ be the set of training examples.
2: Initialize the weight vector with random values, $\mathbf{w}^{(0)}$
3: **repeat**
4: **for** each training example $(\mathbf{x}_i, y_i) \in D$ **do**
5: Compute the predicted output $\hat{y}_i^{(k)}$
6: **for** each weight w_j **do**
7: Update the weight, $w_j^{(k+1)} = w_j^{(k)} + \lambda(y_i - \hat{y}_i^{(k)})x_{ij}$.
8: **end for**
9: **end for**
10: **until** stopping condition is met

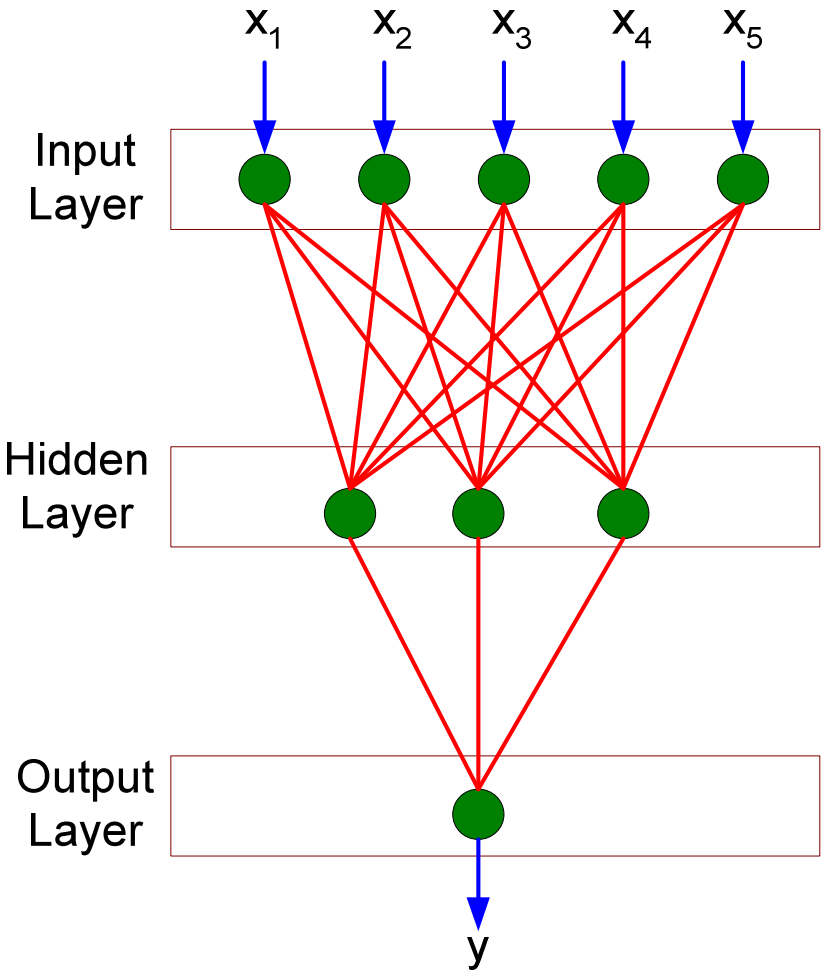
Perceptron

✓linear in its parameters **w and attributes x.**
✓decision boundary of a perceptron, which is obtained by setting ˆ*y = 0, is a linear hyperplane that separates the* data into two classes, *−1 and +1.*
✓The perceptron learning algorithm is guaranteed to converge to an Optimal for linearly separable classification problems.

X_1	X_2	y
0	0	−1
1	0	1
0	1	1
1	1	−1

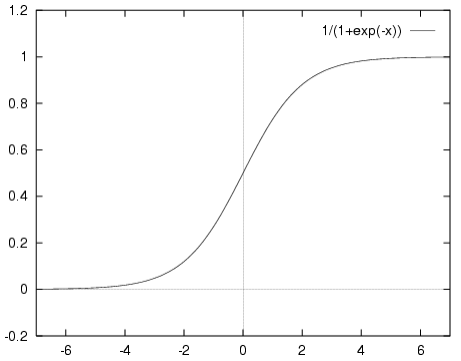


Multilayer Artificial Neural Network

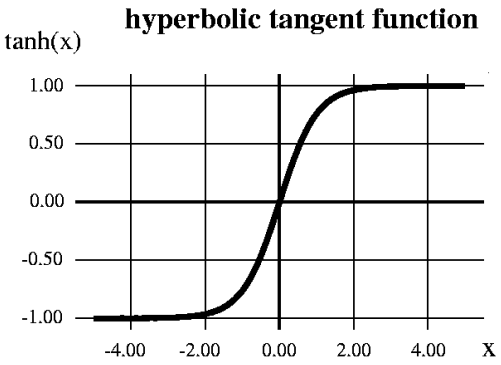


Different Activation Functions

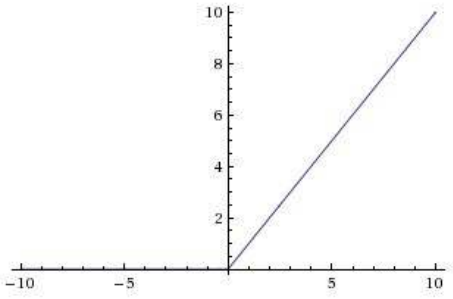
Sigmoid or Logistic



Hyperbolic Tangent function



ReLu- Rectified Linear units



Multilayer Artificial Neural Network

additional complexities allow multilayer neural networks to model more complex relationships between the input and output variables

Learning the ANN Model

goal of the ANN learning algorithm is to determine a set of weights **w** that minimize the total sum of squared errors:

$$E(\mathbf{w}) = \frac{1}{2}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2.$$

$$E(\mathbf{w}) = \sum_{n=1}^{N} E_n(\mathbf{w})$$

$$w_j \longleftarrow w_j - \lambda \frac{\partial E(\mathbf{w})}{\partial w_j}$$

Backpropagation

$$E(\mathbf{w}) = \sum_{n=1}^{N} E_n(\mathbf{w})$$

$$\frac{\partial E_n}{\partial w_{ji}}$$

Simple Case

$$y_k = \sum_i w_{ki} x_i$$

$$E = \frac{1}{2} \sum_k (y_k - t_k)^2$$

Desired output

$$\frac{\partial E}{\partial w_{ji}} = (y_j - t_j) x_i$$

Backpropagation

In a general feed-forward network, each unit computes a weighted sum of its inputs of the form

$$a_j = \sum_i w_{ji} z_i$$

$$z_j = h(a_j)$$

forward propagation

How to compute?

$$\frac{\partial E}{\partial w_{ji}} = \frac{\partial E}{\partial a_j} \frac{\partial a_j}{\partial w_{ji}}$$

$$\delta_j \equiv \frac{\partial E}{\partial a_j} \qquad \frac{\partial a_j}{\partial w_{ji}} = z_i$$

$$\frac{\partial E}{\partial w_{ji}} = \delta_j z_i$$

Backpropagation

for the output units, we have

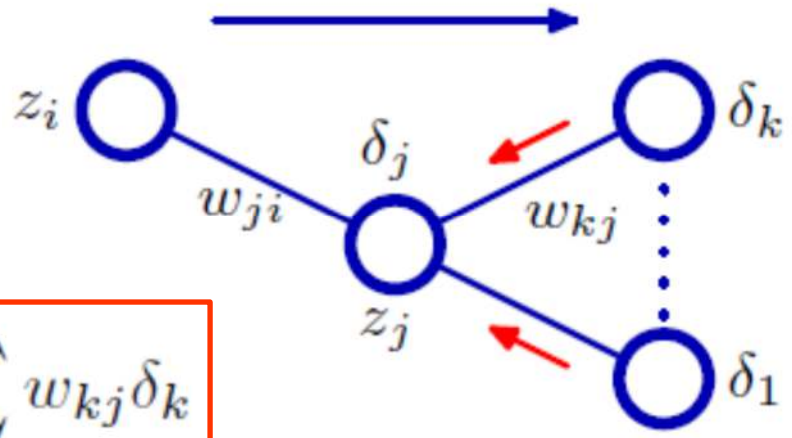
$$E = \frac{1}{2}\sum_k (y_k - t_k)^2$$

$$\delta_k = y_k - t_k$$

$$\delta_j \equiv \frac{\partial E}{\partial a_j} = \sum_k \frac{\partial E}{\partial a_k}\frac{\partial a_k}{\partial a_j}$$

$$z_j = h(a_j)$$

$$\delta_j = h'(a_j)\sum_k w_{kj}\delta_k$$



Backpropagation

$$E(\mathbf{w}) = \sum_{n=1}^{N} E_n(\mathbf{w})$$

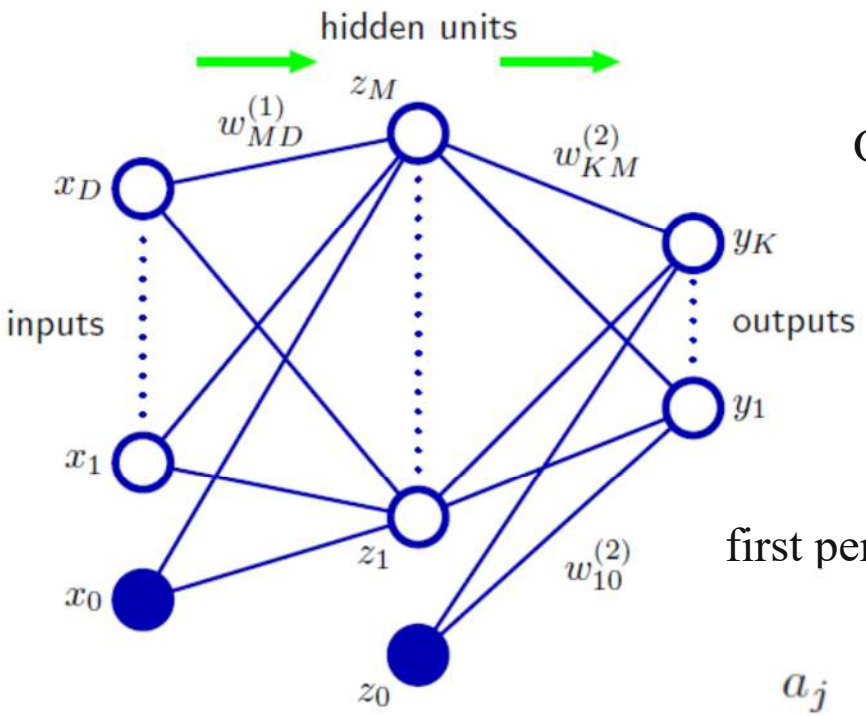
$$\frac{\partial E_n}{\partial w_{ji}}$$

$$\frac{\partial E}{\partial w_{ji}} = \delta_j z_i$$

$$\delta_j = h'(a_j) \sum_k w_{kj} \delta_k$$

1. Apply an input vector $\mathbf{x}_n$ to the network and forward propagate through the network to find the activations of all the hidden and output units.
2. Evaluate the δ_k for all the output units
3. Backpropagate the δ's to obtain δ_j for each hidden unit in the network.
4. evaluate the required derivatives.

example



$$h(a) \equiv \tanh(a)$$

Output units have linear activation functions

$$y_k = a_k \qquad h'(a) = 1 - h(a)^2$$

$$E_n = \frac{1}{2}\sum_{k=1}^{K}(y_k - t_k)^2$$

first perform a forward propagation using

$$a_j = \sum_{i=0}^{D} w^{(1)}_{ji} x_i$$

$$z_j = \tanh(a_j)$$

$$y_k = \sum_{j=0}^{M} w^{(2)}_{kj} z_j$$

example

Next we compute the δ's for each output unit using

$$\delta_k = y_k - t_k$$

Then we backpropagate these to obtain δs for the hidden units using

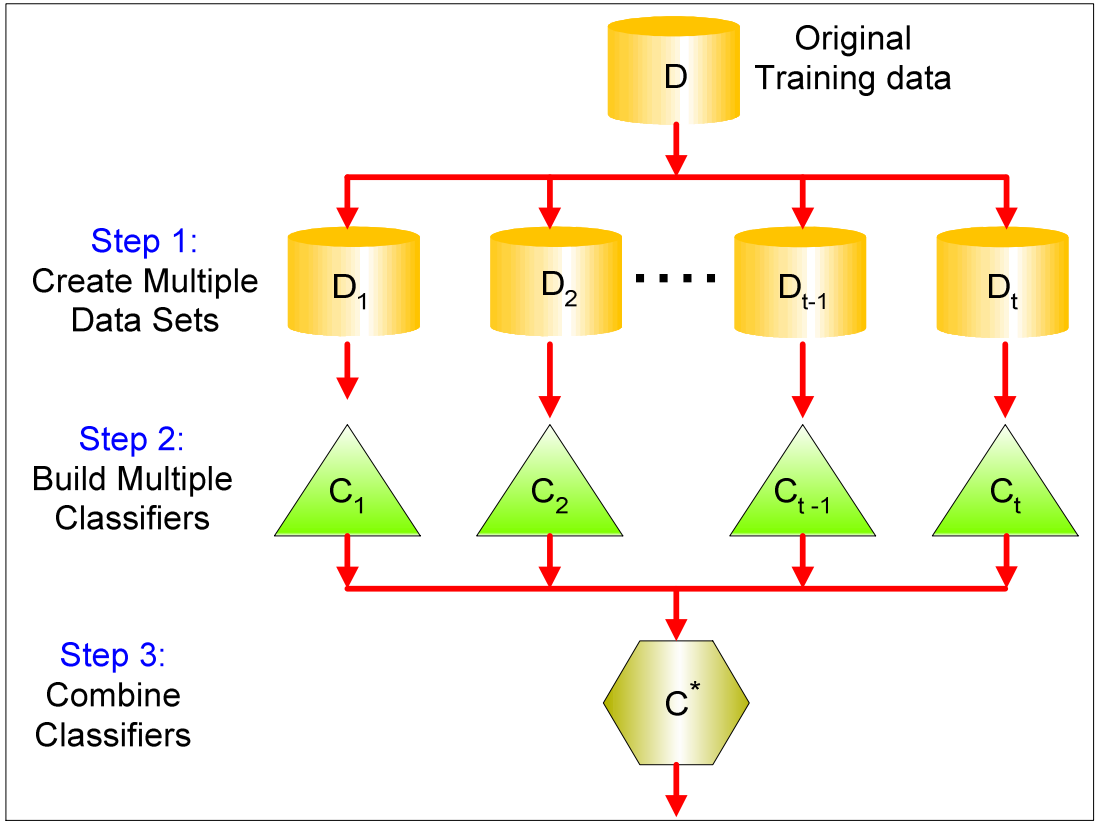
$$\delta_j = (1 - z_j^2) \sum_{k=1}^{K} w_{kj} \delta_k$$

$$\frac{\partial E_n}{\partial w_{ji}^{(1)}} = \delta_j x_i, \qquad \frac{\partial E_n}{\partial w_{kj}^{(2)}} = \delta_k z_j$$

Ensemble Methods

Ensemble Methods

- ✓ improving classification accuracy by aggregating the predictions of multiple classifiers
- ✓ Construct a set of base classifiers from the training data
- ✓ Predict class label of previously unseen records by aggregating predictions made by multiple classifiers



Ensemble Methods

- Suppose there are 25 base classifiers
  - Each classifier has error rate, $\varepsilon = 0.35$
  - Assume classifiers are independent
  - Probability that the ensemble classifier makes a wrong prediction:

$$\sum_{i=13}^{25} \binom{25}{i} \varepsilon^i (1-\varepsilon)^{25-i} = 0.06$$

How to generate an ensemble of classifiers?

- Bagging: bootstrap aggregating
- Boosting

Bagging

- Sampling with replacement
- Build classifier on each bootstrap sample

Original Data	1	2	3	4	5	6	7	8	9	10
Bagging (Round 1)	7	8	10	8	2	5	10	10	5	9
Bagging (Round 2)	1	4	9	1	2	3	2	7	3	2
Bagging (Round 3)	1	8	5	10	5	5	9	6	3	7

Algorithm 5.6 Bagging algorithm.

1: Let k be the number of bootstrap samples.
2: **for** $i = 1$ to k **do**
3: Create a bootstrap sample of size N, D_i.
4: Train a base classifier C_i on the bootstrap sample D_i.
5: **end for**
6: $C^*(x) = \underset{y}{\operatorname{argmax}} \sum_i \delta(C_i(x) = y)$.

$\{\delta(\cdot) = 1$ if its argument is true and 0 otherwise$\}$.

Boosting

- An iterative procedure to adaptively change distribution of training data by focusing more on previously misclassified records
  - Initially, all N records are assigned equal weights
  - Unlike bagging, weights may change at the end of boosting round

1. how the weights of the training examples are updated at the end of each boosting round
2. how the predictions made by each classifier are combined.

Boosting: AdaBoost

Let $\{(\mathbf{x}_j, y_j) \mid j = 1, 2, \ldots, N\}$ denote a set of N training examples.
Unlike bagging, importance of a base classifier Ci depends on its error rate

$$\epsilon_i = \frac{1}{N}\left[\sum_{j=1}^{N} w_j \, I\Big(C_i(\mathbf{x}_j) \neq y_j\Big)\right], \qquad \alpha_i = \frac{1}{2}\ln\left(\frac{1-\epsilon_i}{\epsilon_i}\right)$$

weight assigned to example (**x***i*, *yi*) during the *jth* boosting round

$$w_i^{(j+1)} = \frac{w_i^{(j)}}{Z_j} \times \begin{cases} \exp^{-\alpha_j} & \text{if } C_j(\mathbf{x_i}) = y_i \\ \exp^{\alpha_j} & \text{if } C_j(\mathbf{x_i}) \neq y_i \end{cases}$$

Normalization factor

intermediate rounds produce an error rate higher than 50%, the weights wi = 1/N

AdaBoost

Algorithm 5.7 AdaBoost algorithm.

1: $\mathbf{w} = \{w_j = 1/N \mid j = 1, 2, \ldots, N\}$. {Initialize the weights for all N examples.}
2: Let k be the number of boosting rounds.
3: **for** $i = 1$ to k **do**
4: Create training set D_i by sampling (with replacement) from D according to $\mathbf{w}$.
5: Train a base classifier C_i on D_i.
6: Apply C_i to all examples in the original training set, D.
7: $\epsilon_i = \frac{1}{N}\left[\sum_j w_j \, \delta(C_i(x_j) \neq y_j)\right]$ {Calculate the weighted error.}
8: **if** $\epsilon_i > 0.5$ **then**
9: $\mathbf{w} = \{w_j = 1/N \mid j = 1, 2, \ldots, N\}$. {Reset the weights for all N examples.}
10: Go back to Step 4.
11: **end if**
12: $\alpha_i = \frac{1}{2} \ln \frac{1-\epsilon_i}{\epsilon_i}$.
13: Update the weight of each example according to Equation 5.69.
14: **end for**
15: $C^*(\mathbf{x}) = \underset{y}{\operatorname{argmax}} \sum_{j=1}^{T} \alpha_j \delta(C_j(\mathbf{x}) = y))$.

Example

x	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
y	1	1	1	−1	−1	−1	−1	1	1	1

Boosting Round 1:

x	0.1	0.4	0.5	0.6	0.6	0.7	0.7	0.7	0.8	1
y	1	-1	-1	-1	-1	-1	-1	-1	1	1

Boosting Round 2:

x	0.1	0.1	0.2	0.2	0.2	0.2	0.3	0.3	0.3	0.3
y	1	1	1	1	1	1	1	1	1	1

Boosting Round 3:

x	0.2	0.2	0.4	0.4	0.4	0.4	0.5	0.6	0.6	0.7
y	1	1	-1	-1	-1	-1	-1	-1	-1	-1

Round	x=0.1	x=0.2	x=0.3	x=0.4	x=0.5	x=0.6	x=0.7	x=0.8	x=0.9	x=1.0
1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1
2	0.311	0.311	0.311	0.01	0.01	0.01	0.01	0.01	0.01	0.01
3	0.029	0.029	0.029	0.228	0.228	0.228	0.228	0.009	0.009	0.009

Random Forest

- ✓ ensemble methods specifically designed for decision tree classifiers
- ✓ multiple decision trees where each tree is generated based on the values of an independent set of random vectors

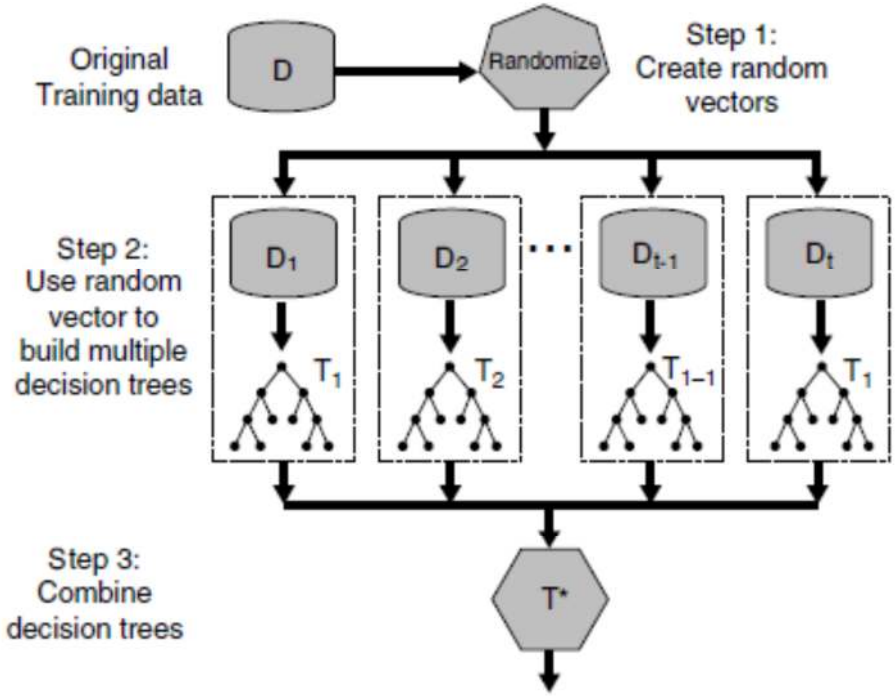


Figure 5.40. Random forests.

Random Forest

- ✓ Bagging using decision trees is a special case of random forests
- ✓ randomly select *F* input features to split at each node of the decision tree
- ✓ majority voting scheme
- ✓ To increase randomness, bagging can also be used to generate bootstrap samples for Forest-RI

Metrics for class imbalance problem

Imbalance

- ✓ Data sets with imbalanced class distributions
- ✓ in credit card fraud detection, fraudulent transactions are outnumbered by legitimate transactions
- ✓ accuracy measure, used extensively for classifiers, may not be well suited for evaluating models derived from imbalanced data sets

example : 1% of the credit card transactions fraudulent,
a model that predicts every transaction as legitimate
accuracy 99%
it fails to detect any of the fraudulent activities.

binary classification, the rare class is often denoted as the positive class against negative class

		Predicted Class	
		+	−
Actual Class	+	f_{++} (TP)	f_{+-} (FN)
	−	f_{-+} (FP)	f_{--} (TN)

confusion matrix

Imbalance

Precision : fraction of records that actually turns out to be positive in the group the classifier has declared as a positive class

$$\text{Precision, } p = \frac{TP}{TP + FP}$$

Recall measures the fraction of positive examples correctly predicted by the classifier

$$\text{Recall, } r = \frac{TP}{TP + FN}$$

maximizes both precision and recall

Imbalance

Precision and recall can be summarized into another metric known as the F1 measure

$$F_1 = \frac{2}{\frac{1}{r} + \frac{1}{p}}.$$

tends to be closer to the smaller of the two numbers
a high value of *F*1-measure ensures that both precision and recall are reasonably high

$$\text{Weighted accuracy} = \frac{w_1 TP + w_4 TN}{w_1 TP + w_2 FP + w_3 FN + w_4 TN}.$$