Data Mining Workshop

Aachen, 23-25 November 2016

Gerasimos (Jerry) Spanakis, PhD
http://dke.maastrichtuniversity.nl/jerry.spanakis
@gerasimoss

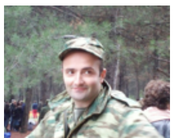
Maastricht University

Who am I?

2001-2006 **Diploma (MSc)** in Electrical & Computer Engineering
(Neural Networks and Reinforcement Learning)

2007-2012 **Diploma (MSc)** in Civil (Transport) Engineering
(Machine learning in transport optimization problems)

2007-2012 **PhD** in Computational Intelligence
(Intelligent techniques in text analysis)



2012-2013 **Army service:** maintenance, update, development for the army personnel software system

2013-2014 **Lecturer** at Technical University of Crete
Scientific Associate at Technological Institute of Crete

2016 **Visiting researcher** at University of Alberta
(Recommender Systems for Higher Education)

2014-2016 **Postdoctoral Researcher** at Maastricht University
Department of Data Science & Knowledge Engineering

2016- **Assistant Professor** at Maastricht University
Department of Data Science & Knowledge Engineering

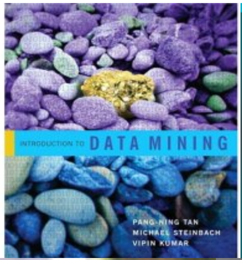
Who am I (really)?

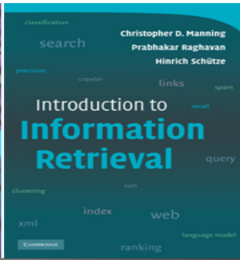
- Machine learner & data miner
- Coffee addict
- TV series binge-watcher
- Aviation enthusiast
- Gin&Tonic lover
- Proud geek

Outline

- Data mining: Why world has gone crazy?
  - Data preparation
- Different forms of data & learning
  - Classification
  - Clustering, Dimensionality reduction
  - Recommender Systems
  - Topic modeling for texts
- Deep learning ”hype”
  - How it works?

Some slide credits to:









Scheduling

- Mornings: “Lectures”, Afternoons: Practicals
  - Can be adjusted...
- Day 1: Introduction, Data Pre-processing, Supervised learning: Classification
- Day 2: Unsupervised learning: Clustering, Dimensionality Reduction, Recommender Systems, Topic modeling for texts
- Day 3: Deep learning for text & images

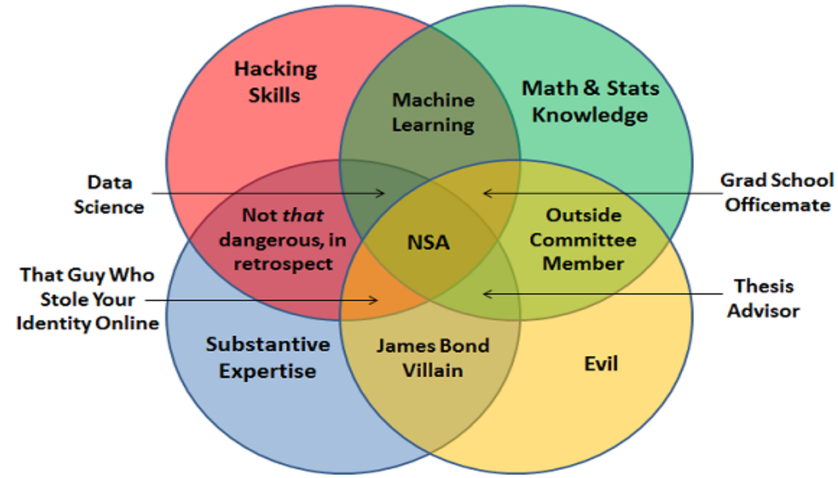
https://dke.maastrichtuniversity.nl/jerry.spanakis/aachen16/

Let's start...

- How many of you have already applied data mining?
- How many of you have heard words like classification, clustering, matrix factorization, SVM, accuracy, LDA,... ?
- How many of you have worked with R?

What is data science or data mining?

- Learning = Representation + Evaluation + Optimization



[Joel Grus, 2016]

Representation: Choice of model & hypothesis space

Evaluation: Choice of objective function

Optimization: The algorithm to compute the best model

We mine data every day

- Yes, it works ☺
- Spam detection
- Credit card fraud detection
- Digit recognition
- Voice recognition
- Face detection
- Stock trading
- Games

Why it works?

- Supervised learning techniques

Cars

Motorcycles

What is this?

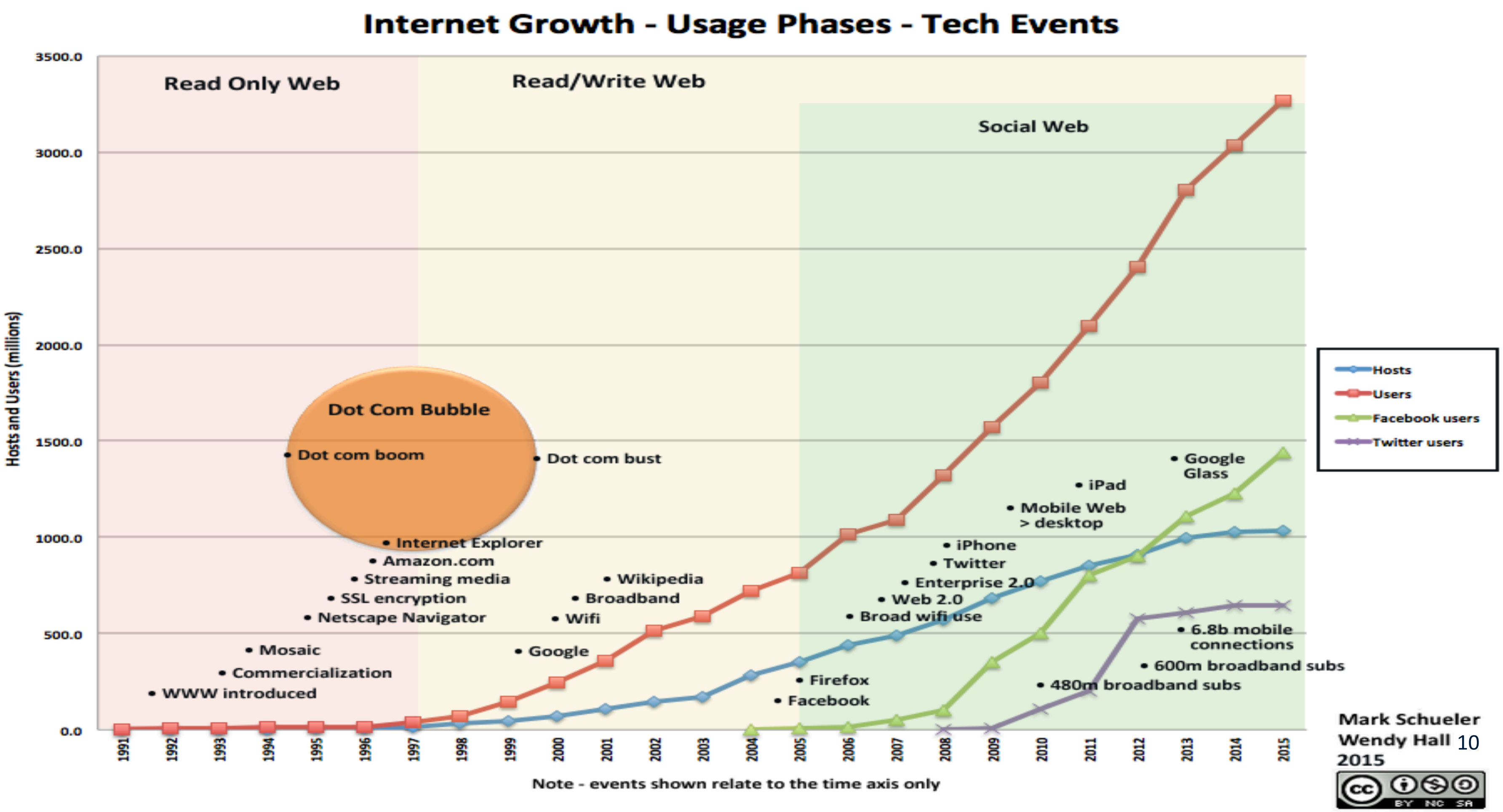
Why it works?

- Lots of (labeled) data IMAGENET

14,197,122 images, 21841 synsets indexed

Explore Download Challenges Publications CoolStuff About

Not logged in. Login | Signup

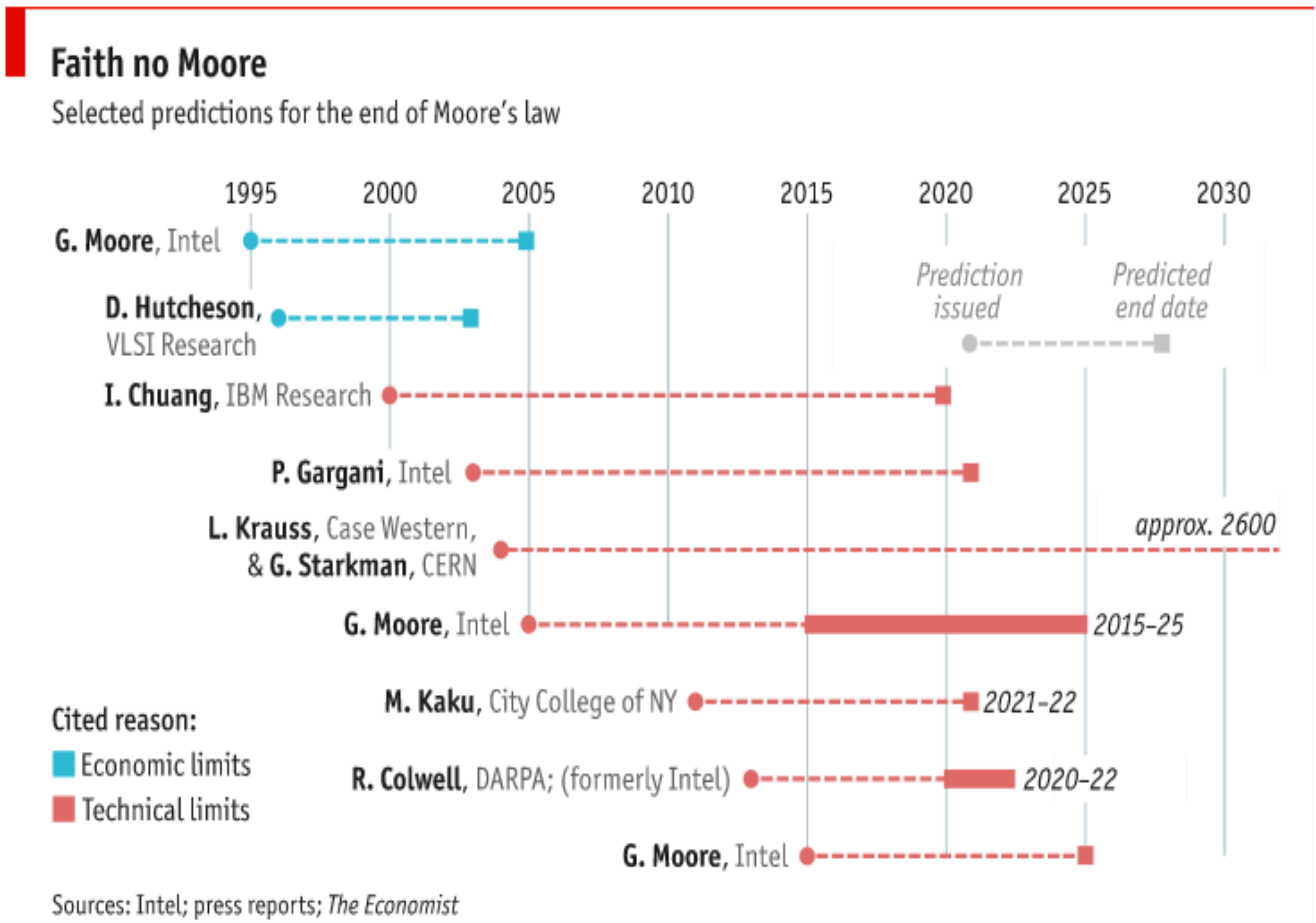


Why it works?

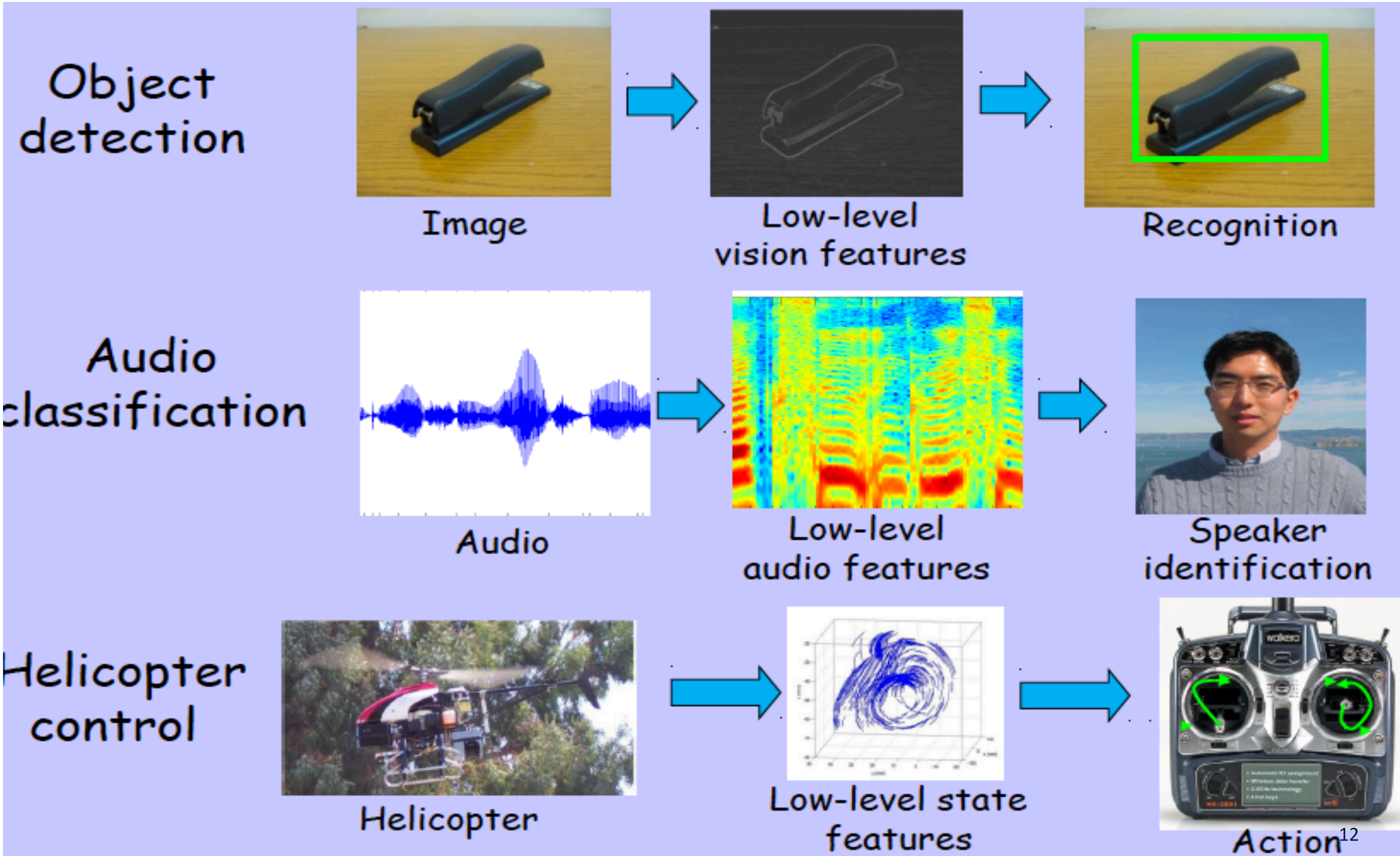
- Computational power

“The number of people predicting the death of Moore’s law doubles every two years”

-- Peter Lee,
VP MS Research



How is computer perception done?



Case Study: Bank

- Business goal: Sell more home equity loans
- Current models:
  - Customers with college-age children use home equity loans to pay for tuition
  - Customers with variable income use home equity loans to even out stream of income
- Data:
  - Large data warehouse
  - Consolidates data from 42 operational data sources

Case Study: Bank (Contd.)

1. Select subset of customer records who have received home equity loan offer
   - Customers who declined
   - Customers who signed up

Income	Number of Children	Average Checking Account Balance	…	Reponse
$40,000	2	$1500		Yes
$75,000	0	$5000		No
$50,000	1	$3000		No
…	…	…	…	…

Case Study: Bank (Contd.)

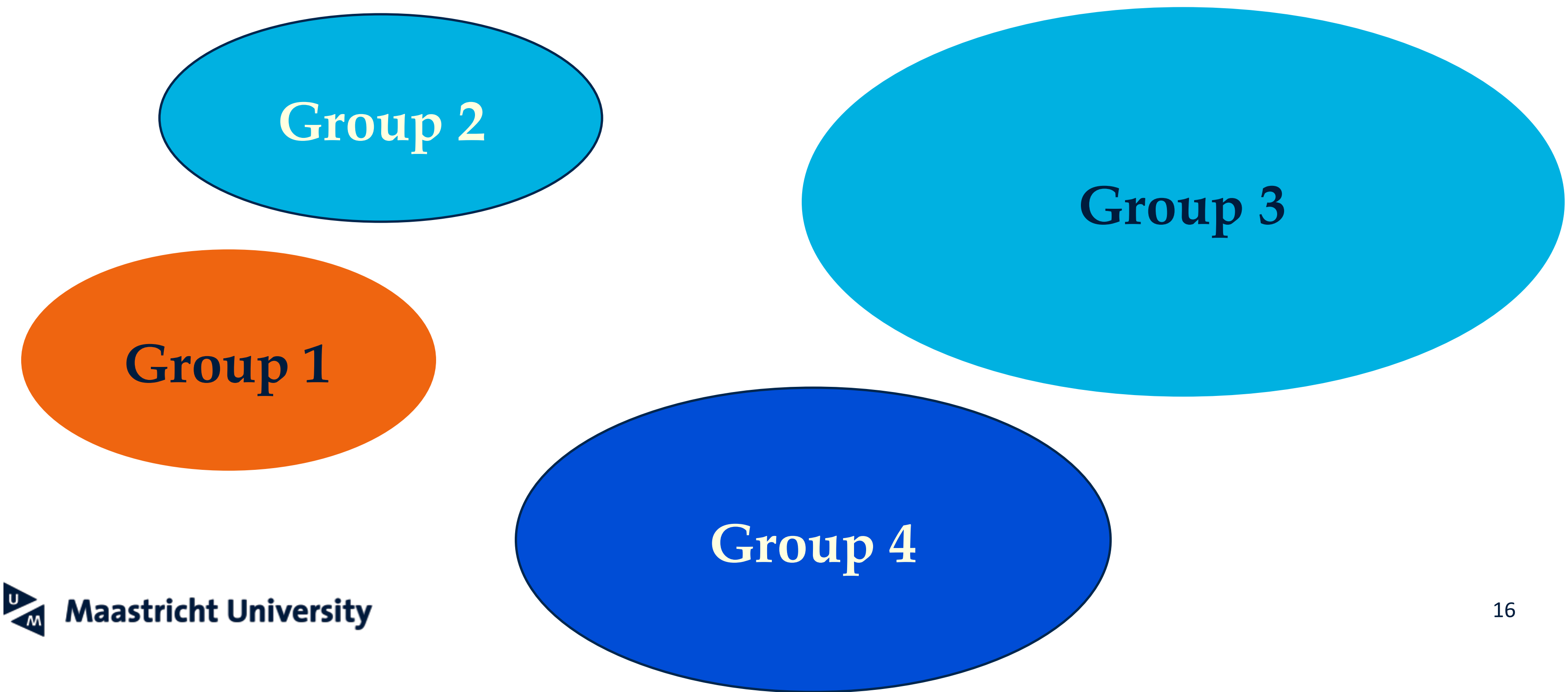
2. Find rules to predict whether a customer would respond to home equity loan offer

IF (Salary < 40k) and
(numChildren > 0) and
(ageChild1 > 18 and ageChild1 < 22)
THEN YES

…

Case Study: Bank (Contd.)

3. Group customers into clusters and investigate clusters



Case Study: Bank (Contd.)

4. Evaluate results:
   – Many “uninteresting” clusters
   – One interesting cluster! Customers with both business and personal accounts; unusually high percentage of likely respondents

What is a Data Mining Model?

A data mining model is a description of a certain aspect of a dataset. It produces output values for an assigned set of inputs.

Examples:

- Clustering
- Linear regression model
- Classification model
- Frequent itemsets and association rules
- Support Vector Machines
- …

Learning from data is difficult

Supervised learning:

- It has nothing to do like human learning
- Transition to open but big data

Unsupervised learning:

- Exploratory data analysis: Goal is not clear
- Difficult to assess performance
- High-dimensional data

Our goal: Vector representations

Name	Gender	Fever	Cough	Test-1	Test-2	Test-3	Test-4
Jack	M	1	0	1	0	0	0
Mary	F	1	0	1	0	1	0
Jim	M	1	1	0	0	0	0

- Discrete vs. Continuous data
- Numeric, Binary, Ordinal features
- Different distance measures

$$\mathbf{x} = (x_1\, x_2 \cdots x_n) \text{ and } \mathbf{y} = (y_1\, y_2 \cdots y_n)$$

$$d(\mathbf{x}, \mathbf{y}) = \left(|x_1 - y_1|^p + |x_2 - y_2|^p \cdots + |x_n - y_n|^p\right)^{\frac{1}{p}}, \quad p > 0$$

Why Data Preprocessing?

- Data in the real world is dirty
  - incomplete: lacking attribute values, lacking certain attributes of interest, or containing only aggregate data
    - e.g., occupation=" "
  - noisy: containing errors or outliers
    - e.g., Salary="-10"
  - inconsistent: containing discrepancies in codes or names
    - e.g., Age="42" Birthday="03/07/1997"
    - e.g., Was rating "1,2,3", now rating "A, B, C"
    - e.g., discrepancy between duplicate records

Why Is Data Dirty?

- Incomplete data may come from
  - "Not applicable" data value when collected
  - Different considerations between the time when the data was collected and when it is analyzed.
  - Human/hardware/software problems
- Noisy data (incorrect values) may come from
  - Faulty data collection instruments
  - Human or computer error at data entry
  - Errors in data transmission
- Inconsistent data may come from
  - Different data sources
  - Functional dependency violation (e.g., modify some linked data)
- Duplicate records also need data cleaning

Why Is Data Preprocessing Important?

- No quality data, no quality mining results!
- ”Garbage in – Garbage out”
  - Quality decisions must be based on quality data
    - e.g., duplicate or missing data may cause incorrect or even misleading statistics.
  - Data warehouse needs consistent integration of quality data
- Data extraction, cleaning, and transformation comprises the majority of the work of building a data warehouse
- You may assume that in a data mining problem 80% of the work is the preparation of the data

Major Tasks in Data Preprocessing

- Data cleaning
  - Fill in missing values, smooth noisy data, identify or remove outliers, and resolve inconsistencies
- Data integration
  - Integration of multiple databases, data cubes, or files
- Data transformation
  - Normalization and aggregation
- Data reduction
  - Obtains reduced representation in volume but produces the same or similar analytical results
- Data discretization
  - Part of data reduction but with particular importance, especially for numerical data

Mining Data Descriptive Characteristics

- Motivation
  - To better understand the data: central tendency, variation and spread
- Data dispersion characteristics
  - median, max, min, quantiles, outliers, variance, etc.
- Numerical dimensions correspond to sorted intervals
  - Data dispersion: analyzed with multiple granularities of precision
  - Boxplot or quantile analysis on sorted intervals
- Dispersion analysis on computed measures
  - Folding measures into numerical dimensions
  - Boxplot or quantile analysis on the transformed cube

Measuring the Central Tendency

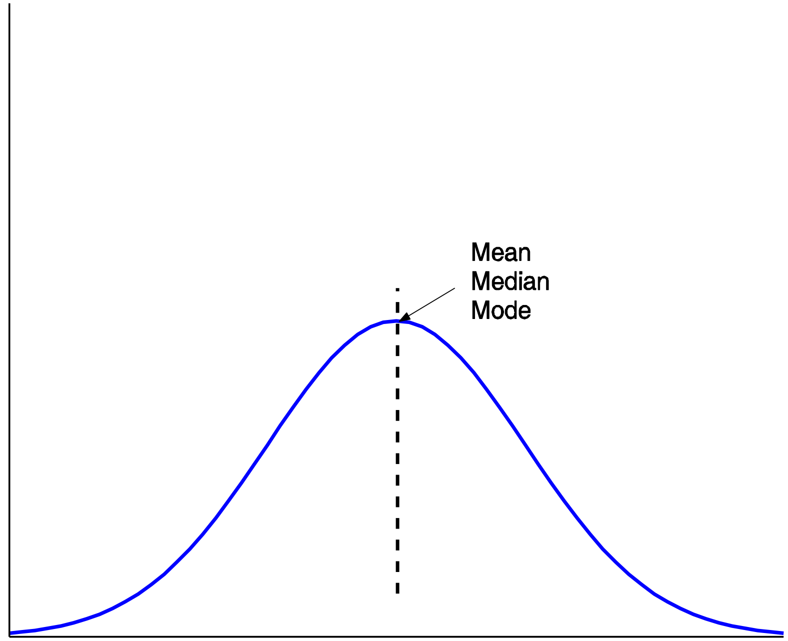
- Mean (algebraic measure) (sample vs. population): $\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i \qquad \mu = \frac{\sum x}{N}$
  - Weighted arithmetic mean: $\bar{x} = \frac{\sum_{i=1}^{n} w_i x_i}{\sum_{i=1}^{n} w_i}$
  - Trimmed mean: chopping extreme values
- Median: A holistic measure
  - Middle value if odd number of values, or average of the middle two values otherwise
  - Estimated by interpolation (for *grouped data*):

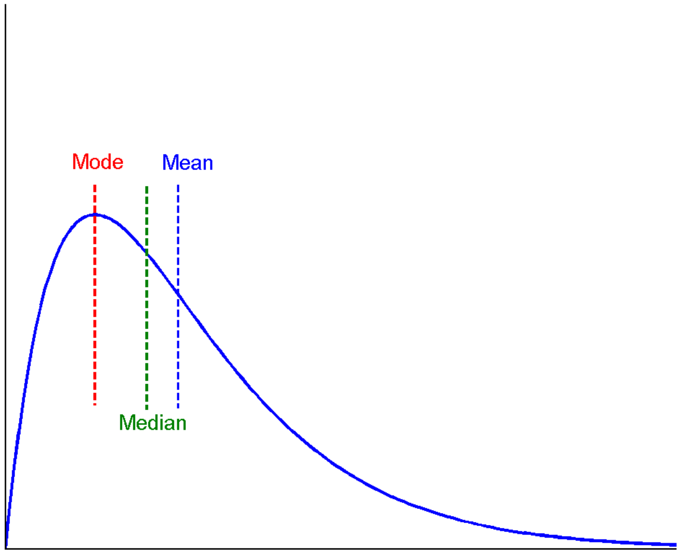
$$median = L_1 + \left(\frac{n/2 - (\sum f)l}{f_{median}}\right)c$$

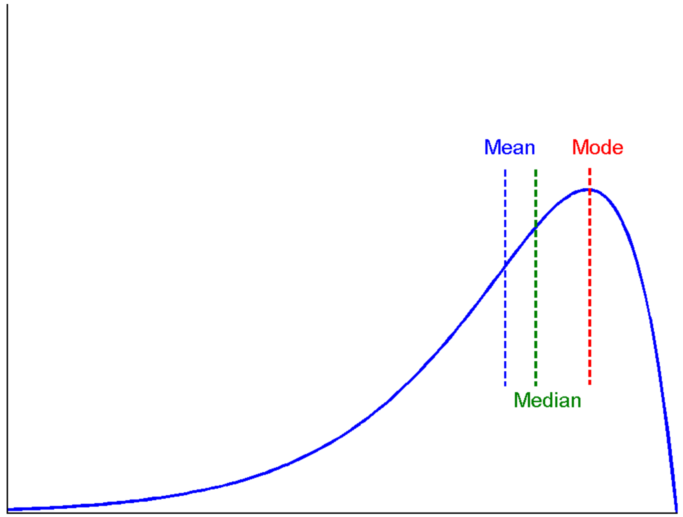
- Mode
  - Value that occurs most frequently in the data
  - Unimodal, bimodal, trimodal
  - Empirical formula: $mean - mode = 3 \times (mean - median)$

Symmetric vs. Skewed Data

- Median, mean and mode of symmetric, positively and negatively skewed data







Measuring the Dispersion of Data

- Quartiles, outliers and boxplots
  - Quartiles: Q_1 (25th percentile), Q_3 (75th percentile)
  - Inter-quartile range: IQR = $Q_3 - Q_1$
  - Five number summary: min, Q_1, M, Q_3, max
  - Boxplot: ends of the box are the quartiles, median is marked, whiskers, and plot outlier individually
  - Outlier: usually, a value higher/lower than 1.5 x IQR
- Variance and standard deviation (*sample: s, population: σ)*
  - Variance: (algebraic, scalable computation)

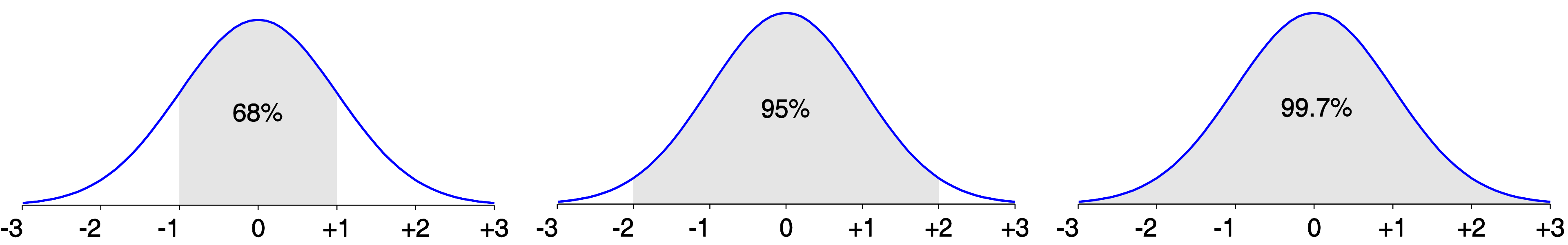
$$\sigma^2 = \frac{1}{N}\sum_{i=1}^{n}(x_i - \mu)^2 = \frac{1}{N}\sum_{i=1}^{n} x_i^{\,2} - \mu^2$$

  - Standard deviation *s (or σ)* is the square root of variance s^2 *(or σ^2)*

$$s^2 = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2 = \frac{1}{n-1}\left[\sum_{i=1}^{n} x_i^{\,2} - \frac{1}{n}\left(\sum_{i=1}^{n} x_i\right)^2\right]$$

Properties of Normal Distribution Curve

- The normal (distribution) curve
  - From μ–σ to μ+σ: contains about 68% of the measurements (μ: mean, σ: standard deviation)
  - From μ–2σ to μ+2σ: contains about 95% of it
  - From μ–3σ to μ+3σ: contains about 99.7% of it

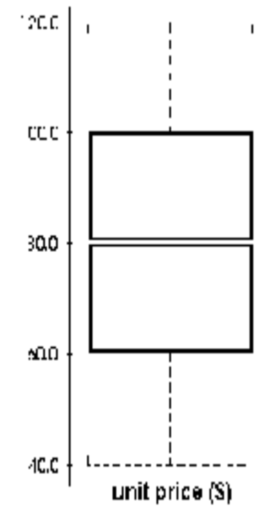


Boxplot Analysis

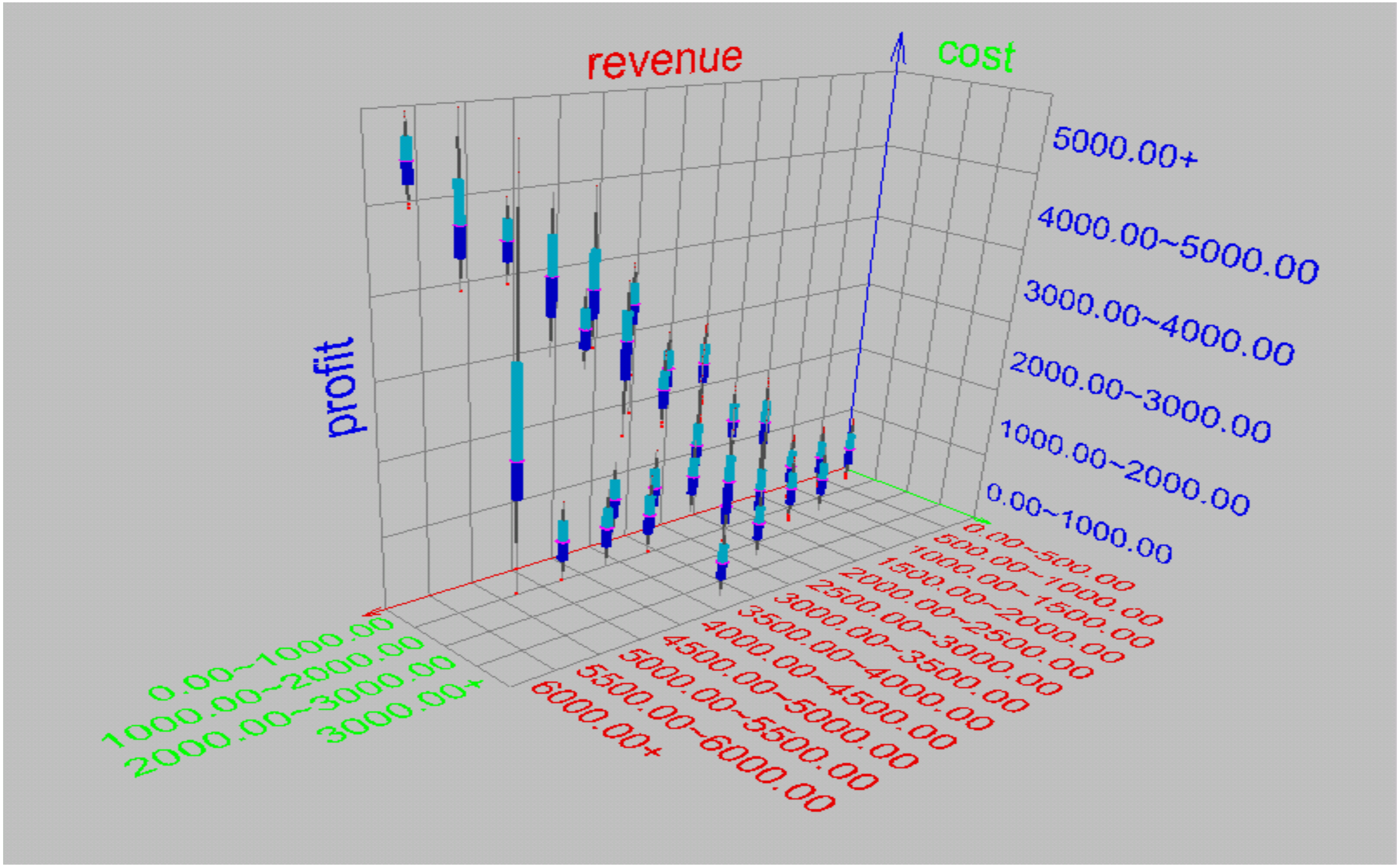
- Five-number summary of a distribution:

  Minimum, Q1, M, Q3, Maximum

- Boxplot
  - Data is represented with a box
  - The ends of the box are at the first and third quartiles, i.e., the height of the box is IRQ
  - The median is marked by a line within the box
  - Whiskers: two lines outside the box extend to Minimum and Maximum



Visualization of Data Dispersion: Boxplot Analysis



Data Cleaning

- Importance
  - "Data cleaning is one of the three biggest problems in data warehousing"—Ralph Kimball
  - "Data cleaning is the number one problem in data related tasks" – Unknown programmer
- Data cleaning tasks
  - Fill in missing values
  - Identify outliers and smooth out noisy data
  - Correct inconsistent data
  - Resolve redundancy caused by data integration

Missing Data

- Data is not always available
  - E.g., many tuples have no recorded value for several attributes, such as customer income in sales data
- Missing data may be due to
  - equipment malfunction
  - inconsistent with other recorded data and thus deleted
  - data not entered due to misunderstanding
  - certain data may not be considered important at the time of entry
  - not register history or changes of the data
- Missing data may need to be inferred.

How to Handle Missing Data?

- Ignore the tuple: usually done when class label is missing (assuming the tasks in classification—not effective when the percentage of missing values per attribute varies considerably.
- Fill in the missing value manually: tedious + infeasible?
- Fill in it automatically with
  - a global constant : e.g., “unknown”, a new class?!
  - the attribute mean
  - the attribute mean for all samples belonging to the same class: smarter
  - the most probable value: inference-based such as Bayesian formula or decision tree

Noisy Data

- Noise: random error or variance in a measured variable
- Incorrect attribute values may due to
  - faulty data collection instruments
  - data entry problems
  - data transmission problems
  - technology limitation
  - inconsistency in naming convention
- Other data problems which requires data cleaning
  - duplicate records
  - incomplete data
  - inconsistent data

How to Handle Noisy Data?

- Binning
  - first sort data and partition into (equal-frequency) bins
  - then one can smooth by bin means, smooth by bin median, smooth by bin boundaries, etc.
- Regression
  - smooth by fitting the data into regression functions
- Clustering
  - detect and remove outliers
- Combined computer and human inspection
  - detect suspicious values and check by human (e.g., deal with possible outliers)

Data Transformation

- Smoothing: remove noise from data
- Aggregation: summarization, data cube construction
- Generalization: concept hierarchy climbing
- Normalization: scaled to fall within a small, specified range
  - min-max normalization
  - z-score normalization
  - normalization by decimal scaling
- Attribute/feature construction
  - New attributes constructed from the given ones

Data Transformation: Normalization

- Min-max normalization: to [new_min$_A$, new_max$_A$]

$$v' = \frac{v - min_A}{max_A - min_A}(new\_max_A - new\_min_A) + new\_min_A$$

  - Ex. Let income range \$12,000 to \$98,000 normalized to [0.0, 1.0]. Then \$73,000 is mapped to $\frac{73,600 - 12,000}{98,000 - 12,000}(1.0 - 0) + 0 = 0.716$

- Z-score normalization (μ: mean, σ: standard deviation):

$$v' = \frac{v - \mu_A}{\sigma_A}$$

  - Ex. Let μ = 54,000, σ = 16,000. Then $\frac{73,600 - 54,000}{16,000} = 1.225$

- Normalization by decimal scaling

$$v' = \frac{v}{10^j}$$

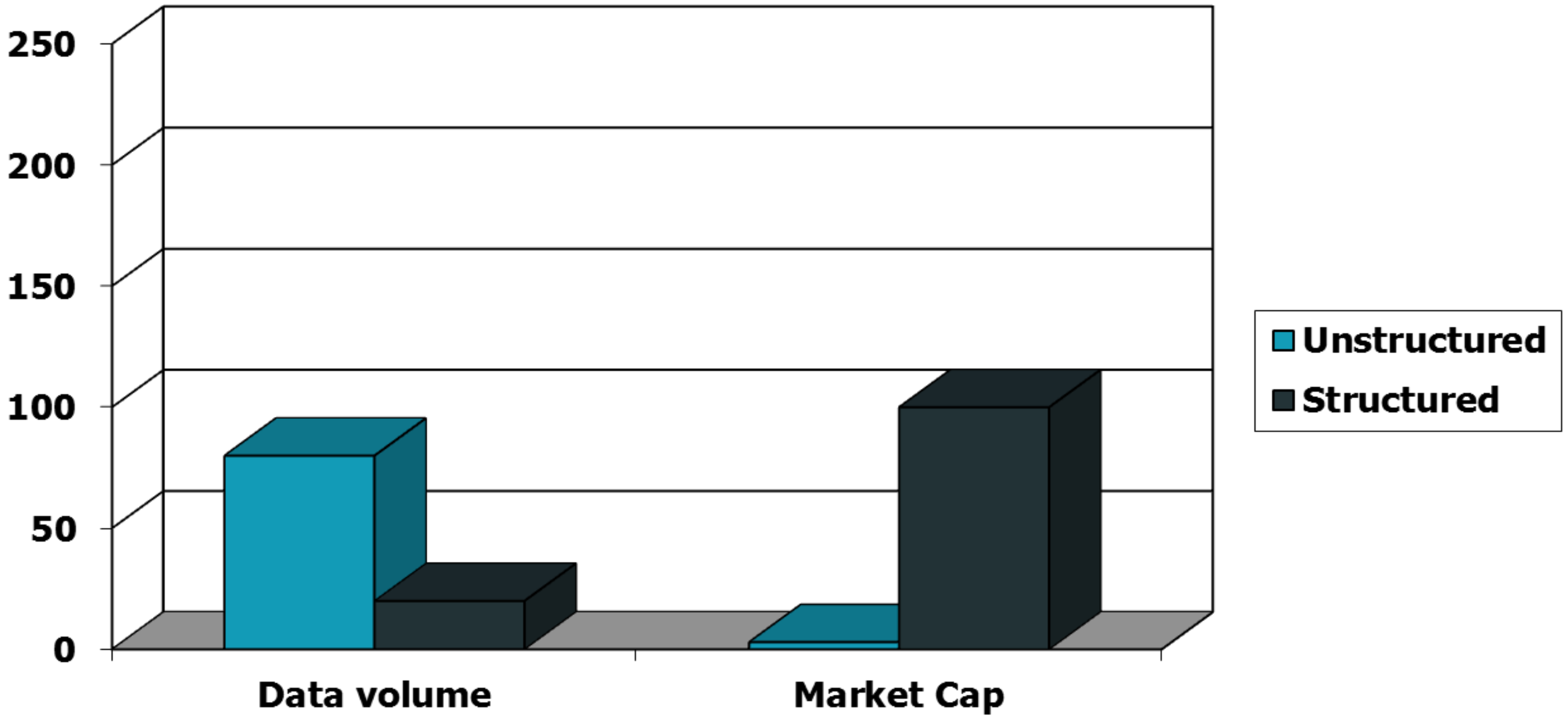
Where *j* is the smallest integer such that Max(|v'|) < 1

How is this applied to different data?

- Data can be :
  - Numeric
  - Categorical
  - Text
  - Images
- The time dimension creates more complex structures
  - Timeseries (sequences of numerical data)
  - Videos (sequences of images)

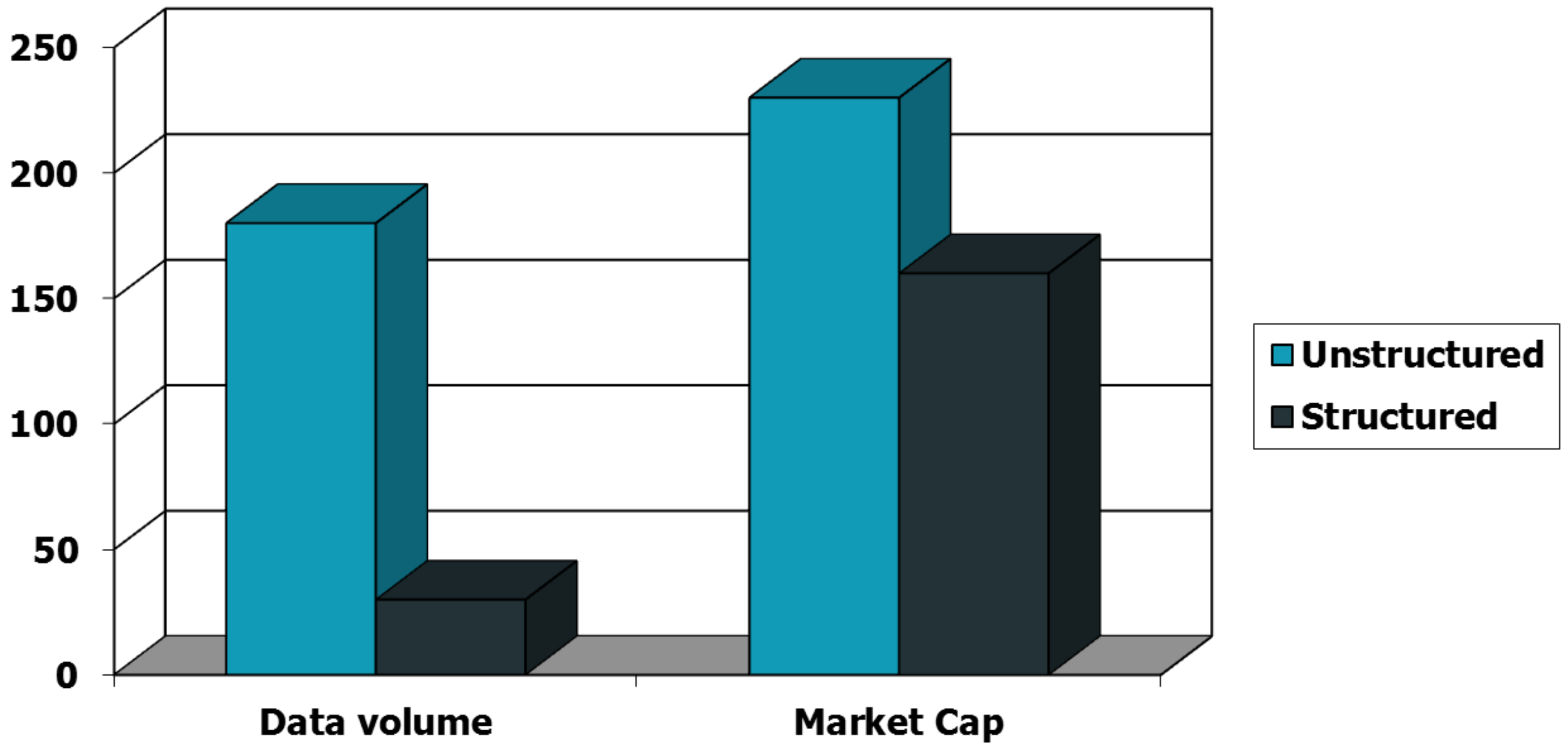
What's the problem with textual data

Unstructured (text) vs. structured (database) data in the mid-nineties



What's the problem with textual data

Unstructured (text) vs. structured (database) data today



All started from information retrieval (or web search)

- Collection: A set of documents
  - Assume it is a static collection for the moment

- Goal: Retrieve documents with information that is relevant to the user's information need and helps the user complete a task

Unstructured data in 1620

- Which plays of Shakespeare contain the words ***Brutus*** *AND* ***Caesar*** but *NOT* ***Calpurnia***?
- One could `grep` all of Shakespeare's plays for ***Brutus*** and ***Caesar,*** then strip out lines containing ***Calpurnia***?
- Why is that not a feasible answer?
  - Slow (for large corpora)
  - *NOT* ***Calpurnia*** is non-trivial
  - Other operations (e.g., find the word ***Romans*** near ***countrymen***) not feasible
  - Ranked retrieval (best documents to return)
    - And it starts getting very complex...

Term-document incidence matrices

	Antony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth
Antony	1	1	0	0	0	1
Brutus	1	1	0	1	0	0
Caesar	1	1	0	1	1	1
Calpurnia	0	1	0	0	0	0
Cleopatra	1	0	0	0	0	0
mercy	1	0	1	1	1	1
worser	1	0	1	1	1	0

Brutus *AND* ***Caesar*** *BUT NOT* ***Calpurnia***

1 if play contains word, 0 otherwise

Incidence vectors

- So we have a 0/1 vector for each term.
- To answer query: take the vectors for ***Brutus, Caesar*** and ***Calpurnia*** (complemented) ➔ bitwise *AND*.

	Antony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth
Antony	1	1	0	0	0	1
Brutus	1	1	0	1	0	0
Caesar	1	1	0	1	1	1
Calpurnia	0	1	0	0	0	0
Cleopatra	1	0	0	0	0	0
mercy	1	0	1	1	1	1
worser	1	0	1	1	1	0

Answers to query

- Antony and Cleopatra, Act III, Scene ii

Agrippa [Aside to DOMITIUS ENOBARBUS]: Why, Enobarbus,
When Antony found Julius ***Caesar*** dead,
He cried almost to roaring; and he wept
When at Philippi he found ***Brutus*** slain.

- Hamlet, Act III, Scene ii

Lord Polonius: I did enact Julius ***Caesar*** I was killed i' the Capitol; ***Brutus*** killed me.

Bigger collections

- Consider *N* = 1 million documents, each with about 1000 words.
- Avg 6 bytes/word including spaces/punctuation
  - 6GB of data in the documents.
- Say there are *M* = 500K *distinct* terms among these.

- 500K x 1M matrix has half-a-trillion 0's and 1's.
- But it has no more than one billion 1's.
  - matrix is extremely sparse.
- Programmer's note: What's a better representation?
  - We only record the 1 positions.

Why?

Problem with Boolean search: feast or famine

- Boolean queries often result in either too few (=0) or too many (1000s) results.
- Query 1: "*standard user dlink 650*" → 200,000 hits
- Query 2: "*standard user dlink 650 no card found*": 0 hits
- It takes a lot of skill to come up with a query that produces a manageable number of hits.
  - AND gives too few; OR gives too many

Term-document count matrices

- Consider the number of occurrences of a term in a document:
  - Each document is a count vector in $\mathbb{N}^{v}$: a column below

	Antony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth
Antony	157	73	0	0	0	0
Brutus	4	157	0	1	0	0
Caesar	232	227	0	2	1	1
Calpurnia	0	10	0	0	0	0
Cleopatra	57	0	0	0	0	0
mercy	2	0	3	5	5	1
worser	2	0	1	1	1	0

Bag of words model

- Vector representation doesn't consider the ordering of words in a document
- *John is quicker than Mary* and *Mary is quicker than John* have the same vectors
- This is called the bag of words model.
- Other well-known problems:
  - Breaks multi-words (e.g. data mining)
  - Does not consider synonymy, hyponymy, etc.
  - Main problem is that it's sparse!

Term frequency tf

- The term frequency $\mathrm{tf}_{t,d}$ of term *t* in document *d* is defined as the number of times that *t* occurs in *d*.
- We want to use tf when computing query-document match scores. But how?
- Raw term frequency is not what we want:
  - A document with 10 occurrences of the term is more relevant than a document with 1 occurrence of the term.
  - But not 10 times more relevant.
- Relevance does not increase proportionally with term frequency.

NB: frequency = count in IR

Log-frequency weighting

- The log frequency weight of term t in d is

$$w_{t,d} = \begin{cases} 1 + \log_{10} \mathrm{tf}_{t,d}, & \text{if } \mathrm{tf}_{t,d} > 0 \\ 0, & \text{otherwise} \end{cases}$$

- 0 → 0, 1 → 1, 2 → 1.3, 10 → 2, 1000 → 4, etc.
- Score for a document-query pair: sum over terms *t* in both *q* and *d*:
- score $= \sum_{t \in q \cap d} (1 + \log \mathrm{tf}_{t,d})$

- The score is 0 if none of the query terms is present in the document.

Document frequency

- Rare terms are more informative than frequent terms
  - Recall stop words
- Consider a term in the query that is rare in the collection (e.g., *arachnophobia*)
- A document containing this term is very likely to be relevant to the query *arachnophobia*
- → We want a high weight for rare terms like *arachnophobia*.

Document frequency, continued

- Frequent terms are less informative than rare terms
- Consider a query term that is frequent in the collection (e.g., *high, increase, line*)
- A document containing such a term is more likely to be relevant than a document that doesn’t
- But it’s not a sure indicator of relevance.
- → For frequent terms, we want high positive weights for words like *high, increase, and line*
- But lower weights than for rare terms.
- We will use document frequency (df) to capture this.

idf weight

- df_t is the document frequency of t: the number of documents that contain t
  - df_t is an inverse measure of the informativeness of t
  - $\mathrm{df}_t \leq N$
- We define the idf (inverse document frequency) of t by $\mathrm{idf}_t = \log_{10}(N/\mathrm{df}_t)$
  - We use log (N/df_t) instead of N/df_t to "dampen" the effect of idf.

Will turn out the base of the log is immaterial.

idf example, suppose *N* = 1 million

term	df_t	idf_t
calpurnia	1	
animal	100	
sunday	1,000	
fly	10,000	
under	100,000	
the	1,000,000	

$$\mathrm{idf}_t = \log_{10}\left(N/\mathrm{df}_t\right)$$

There is one idf value for each term *t* in a collection.

Effect of idf on ranking

- Does idf have an effect on ranking for one-term queries, like
  - iPhone
- idf has no effect on ranking one term queries
  - idf affects the ranking of documents for queries with at least two terms
  - For the query capricious person, idf weighting makes occurrences of capricious count for much more in the final document ranking than occurrences of person.

Collection vs. Document frequency

- The collection frequency of *t* is the number of occurrences of *t* in the collection, counting multiple occurrences.
- Example:

Word	Collection frequency	Document frequency
insurance	10440	3997
try	10422	8760

- Which word is a better search term (and should get a higher weight)?

tf-idf weighting

- The tf-idf weight of a term is the product of its tf weight and its idf weight.

$$\mathrm{w}_{t,d} = \log(1 + \mathrm{tf}_{t,d}) \times \log_{10}(N / \mathrm{df}_t)$$

- Best known weighting scheme in information retrieval
- Increases with the number of occurrences within a document
- Increases with the rarity of the term in the collection

Score for a document given a query

$$\mathrm{Score}(q,d) = \sum_{t \in q \cap d} \mathrm{tf.idf}_{t,d}$$

- There are many variants
  - How “tf” is computed (with/without logs)
  - Whether the terms in the query are also weighted
  - …

Binary → count → weight matrix

	Antony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth
Antony	5.25	3.18	0	0	0	0.35
Brutus	1.21	6.1	0	1	0	0
Caesar	8.59	2.54	0	1.51	0.25	0
Calpurnia	0	1.54	0	0	0	0
Cleopatra	2.85	0	0	0	0	0
mercy	1.51	0	1.9	0.12	5.25	0.88
worser	1.37	0	0.11	4.15	0.25	1.95

Each document is now represented by a real-valued vector of tf-idf weights $\in \mathbb{R}^{|V|}$

Documents as vectors

- So we have a |V|-dimensional vector space
- Terms are axes of the space
- Documents are points or vectors in this space
- Very high-dimensional: tens of millions of dimensions when you apply this to a web search engine
- These are very sparse vectors - most entries are zero.

Queries as vectors

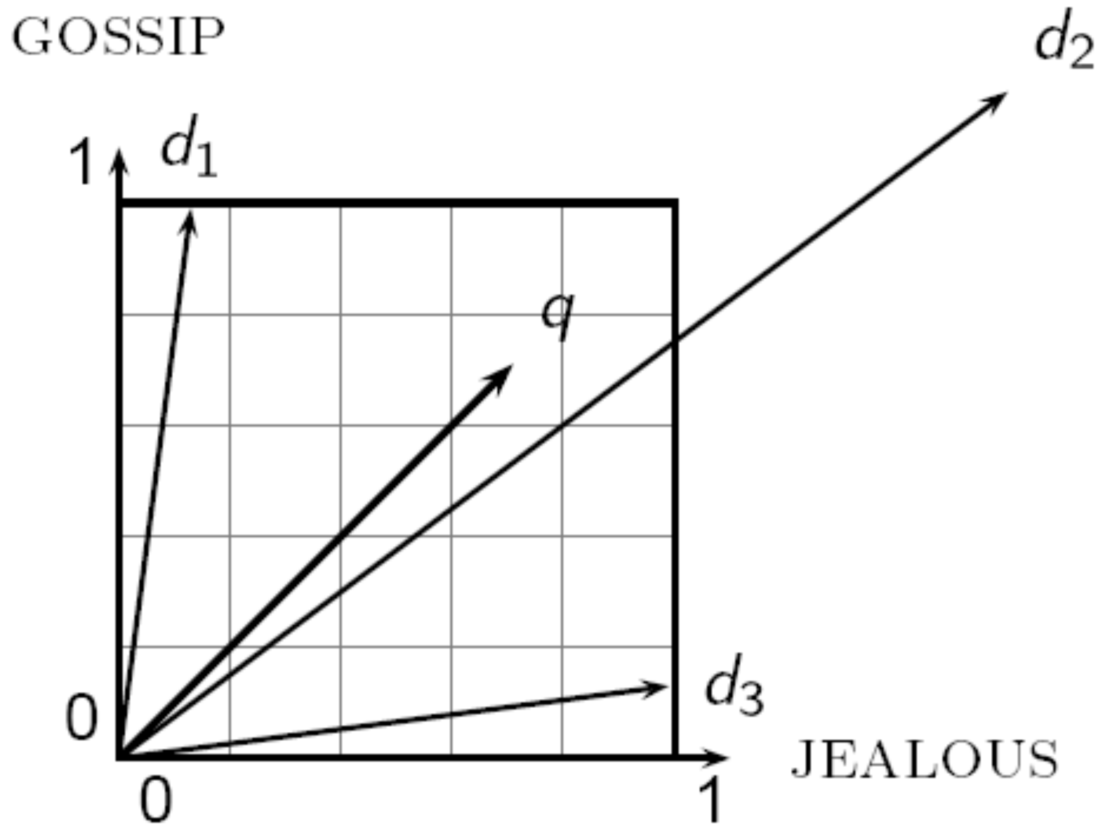
- Key idea 1: Do the same for queries: represent them as vectors in the space
- Key idea 2: Rank documents according to their proximity to the query in this space
- proximity = similarity of vectors
- proximity ≈ inverse of distance
- Recall: We do this because we want to get away from the you're-either-in-or-out Boolean model.
- Instead: rank more relevant documents higher than less relevant documents

Formalizing vector space proximity

- First cut: distance between two points
  - (= distance between the end points of the two vectors)
- Euclidean distance?
- Euclidean distance is a bad idea . . .
- . . . because Euclidean distance is large for vectors of different lengths.

Why distance is a bad idea

The Euclidean distance between $\vec{q}$ and $\vec{d_2}$ is large even though the distribution of terms in the query $\vec{q}$ and the distribution of terms in the document $\vec{d_2}$ are very similar.



Use angle instead of distance

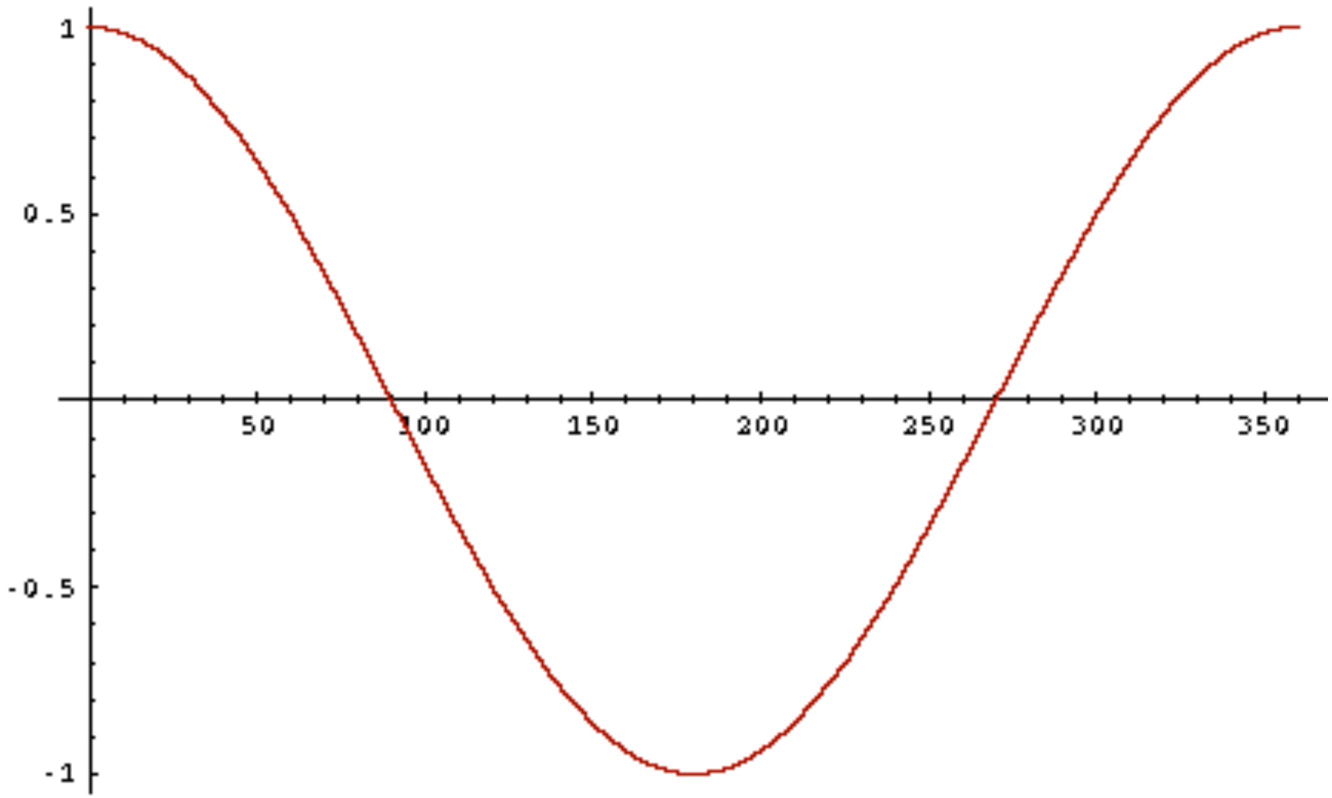
- Thought experiment: take a document *d* and append it to itself. Call this document *d*′.
- “Semantically” d and d′ have the same content
- The Euclidean distance between the two documents can be quite large
- The angle between the two documents is 0, corresponding to maximal similarity.

- Key idea: Rank documents according to angle with query.

From angles to cosines

- The following two notions are equivalent.
  - Rank documents in <u>decreasing</u> order of the angle between query and document
  - Rank documents in <u>increasing</u> order of cosine(query,document)
- Cosine is a monotonically decreasing function for the interval $[0^o, 180^o]$

From angles to cosines



- But how – *and why* – should we be computing cosines?

Length normalization

- A vector can be (length-) normalized by dividing each of its components by its length – for this we use the L_2 norm: $\left\|\vec{x}\right\|_2 = \sqrt{\sum_i x_i^2}$

- Dividing a vector by its L_2 norm makes it a unit (length) vector (on surface of unit hypersphere)
- Effect on the two documents d and d′ (d appended to itself) from earlier slide: they have identical vectors after length-normalization.
  - Long and short documents now have comparable weights

cosine(query,document)

Dot product

Unit vectors

$$\cos(\vec{q},\vec{d}) = \frac{\vec{q} \bullet \vec{d}}{|\vec{q}||\vec{d}|} = \frac{\vec{q}}{|\vec{q}|} \bullet \frac{\vec{d}}{|\vec{d}|} = \frac{\sum_{i=1}^{|V|} q_i d_i}{\sqrt{\sum_{i=1}^{|V|} q_i^2}\sqrt{\sum_{i=1}^{|V|} d_i^2}}$$

q_i is the tf-idf weight of term i in the query
d_i is the tf-idf weight of term i in the document

$\cos(\vec{q},\vec{d})$ is the cosine similarity of $\vec{q}$ and $\vec{d}$ … or, equivalently, the cosine of the angle between $\vec{q}$ and $\vec{d}$.

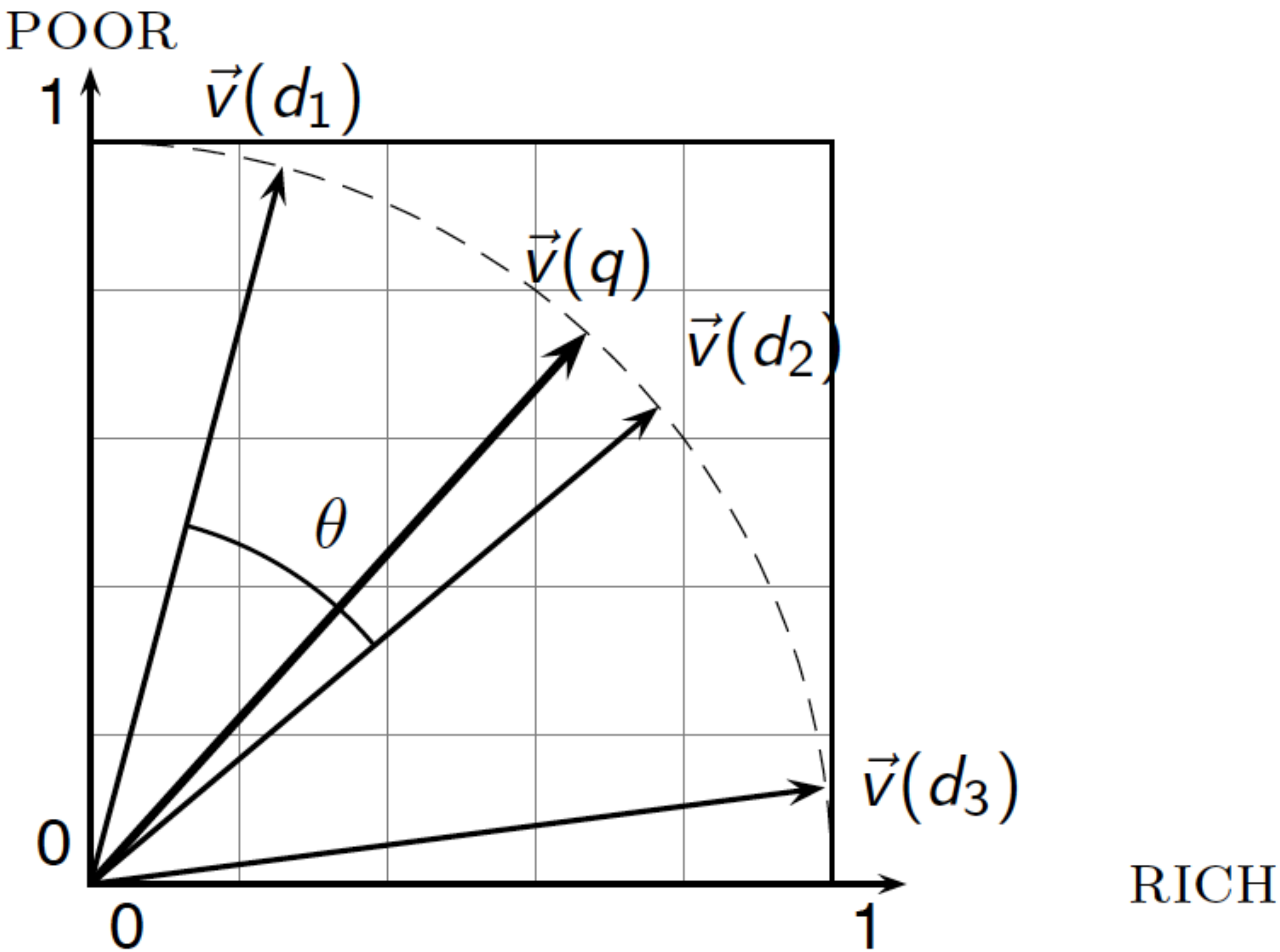
Cosine for length-normalized vectors

- For length-normalized vectors, cosine similarity is simply the dot product (or scalar product):

$$\cos(\vec{q},\vec{d}) = \vec{q} \bullet \vec{d} = \sum_{i=1}^{|V|} q_i d_i$$

for q, d length-normalized.

Cosine similarity illustrated



Cosine similarity amongst 3 documents

How similar are the novels

SaS: *Sense and Sensibility*

PaP: *Pride and Prejudice*, and

WH: *Wuthering Heights*?

term	SaS	PaP	WH
affection	115	58	20
jealous	10	7	11
gossip	2	0	6
wuthering	0	0	38

Term frequencies (counts)

Note: To simplify this example, we don't do idf weighting.

3 documents example contd.

Log frequency weighting

term	SaS	PaP	WH
affection	3.06	2.76	2.30
jealous	2.00	1.85	2.04
gossip	1.30	0	1.78
wuthering	0	0	2.58

After length normalization

term	SaS	PaP	WH
affection	0.789	0.832	0.524
jealous	0.515	0.555	0.465
gossip	0.335	0	0.405
wuthering	0	0	0.588

cos(SaS,PaP) ≈
$0.789 \times 0.832 + 0.515 \times 0.555 + 0.335 \times 0.0 + 0.0 \times 0.0$
≈ 0.94

cos(SaS,WH) ≈ 0.79

cos(PaP,WH) ≈ 0.69

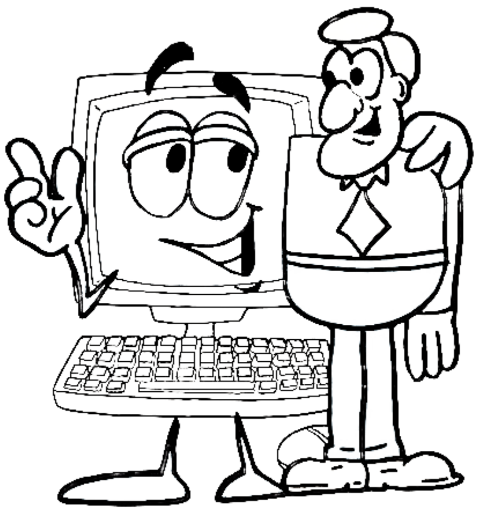
Why do we have cos(SaS,PaP) > cos(SaS,WH)?

The Dream for Natural Language Processing

- It'd be great if machines could
  - Process our email (usefully)
  - Translate languages accurately
  - Help us manage, summarize, and aggregate information
  - Use speech as a UI (when needed)
  - Talk to us / listen to us
- But they can't:
  - Language is complex, ambiguous, flexible, and subtle
  - Good solutions need linguistics and machine learning knowledge
- So:

The mystery

- What's now impossible for computers (and any other species) to do is effortless for humans



What is NLP?

- Fundamental goal: *deep* understand of *broad* language
  - Not just string processing or keyword matching!

What is NLP?

- Computers use (analyze, understand, generate) natural language
- Text Processing
  - Lexical: tokenization, part of speech, head, lemmas
  - Parsing and chunking
  - Semantic tagging: semantic role, word sense
  - Certain expressions: named entities
  - Discourse: coreference, discourse segments
- Speech Processing
  - Phonetic transcription
  - Segmentation (punctuations)
  - Prosody

Why should you care?

- Tremendous progress in NLP
  - An enormous amount of knowledge is now available in machine readable form as natural language text
  - Conversational agents are becoming an important form of human-computer communication
  - Much of human-human communication is now mediate by computers

Commercial world is blooming
Sponsors @ EMNLP 2016

What's the problem with images?

- Images can be represented as vectors of pixels (RGB values) and then can be (simply?) fed to a classification algorithm
- But for a simple image (256x256) there are $2^{524,888}$ possible images
  - There are about 10^{24} starts in the universe
- Think also of variations per class
  - Fruit?
  - Cats in different positions?

What is computer vision?

Terminator 2

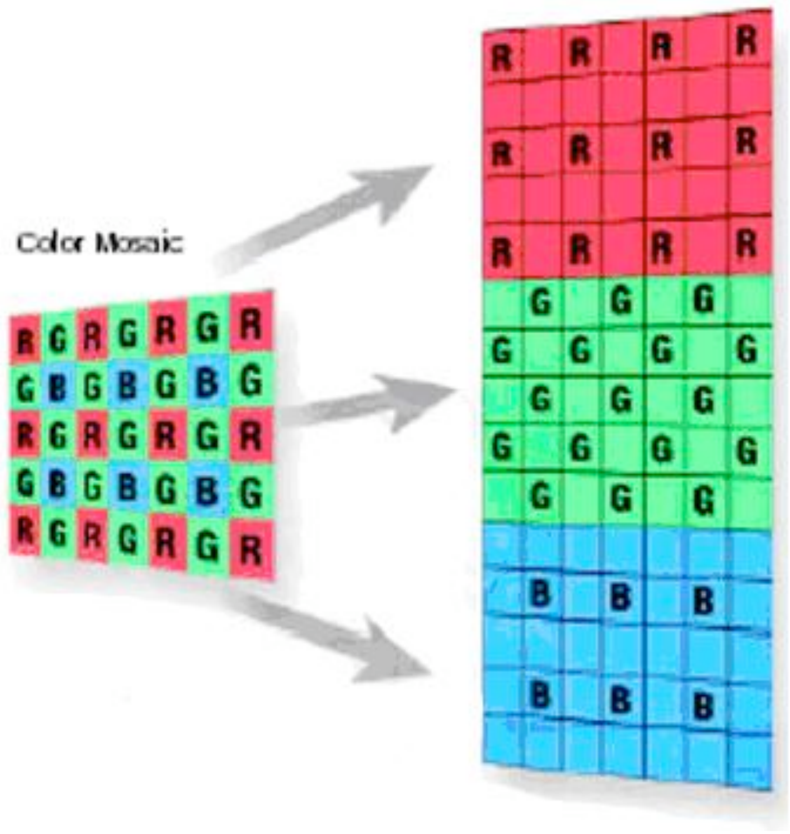
Every picture tells a story

- Goal of computer vision is to write computer programs that can interpret images

Can computers match (or beat) human vision?

- Yes and no (but mostly no!)
  - humans are much better at “hard” things
  - computers can be better at “easy” things

How does it work?



Supervised learning

Classification

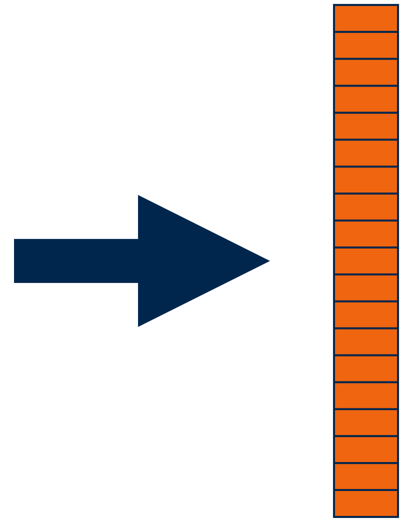
- In classification problems, each entity in some domain can be placed in one of a discrete set of categories: yes/no, friend/foe, good/bad/indifferent, blue/red/green, etc.
- Given a training set of labeled entities, develop a rule for assigning labels to entities in a test set
- Many variations on this theme:
  - binary classification
  - multi-category classification
  - non-exclusive categories
  - ranking
- Many criteria to assess rules and their predictions
  - overall errors
  - costs associated with different kinds of errors
  - operating points

Representation of Objects

- Each object to be classified is represented as a pair (x, y):
  - where x is a description of the object (see examples of data types in the following slides)
  - where y is a label (assumed binary for now)
- Success or failure of a machine learning classifier often depends on choosing good descriptions of objects
  - the choice of description can also be viewed as a learning problem, and indeed we'll discuss automated procedures for choosing descriptions in a later lecture
  - but good human intuitions are often needed here

Data Types

- Vectorial data:
  - physical attributes
  - behavioral attributes
  - context
  - history
  - etc



- We'll assume for now that such vectors are explicitly represented in a table, but practically can take different forms (e.g. kernels)

Data Types

– text and hypertext

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<html>
<head>
	<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
	<title>Welcome to FairmontNET</title>
</head>
<STYLE type="text/css">
.stdtext {font-family: Verdana, Arial, Helvetica, sans-serif; font-size: 11px; color: #1F3D4E;}
.stdtext_wh {font-family: Verdana, Arial, Helvetica, sans-serif; font-size: 11px; color: WHITE;}
</STYLE>

<body leftmargin="0" topmargin="0" marginwidth="0" marginheight="0" bgcolor="BLACK">
<TABLE cellpadding="0" cellspacing="0" width="100%" border="0">
  <TR>
    <TD width=50% background="/TFN/en/CDA/Images/common/labels/decorative_2px_blk.gif"> </TD>
    <TD><img src="/TFN/en/CDA/Images/common/labels/decorative.gif"></td>
    <TD width=50% background="/TFN/en/CDA/Images/common/labels/decorative_2px_blk.gif"> </TD>
  </TR>
</TABLE>
<tr>
<td align="right" valign="middle"><IMG src="/TFN/en/CDA/Images/common/labels/centrino_logo_blk.gif"></td>
</tr>
</body>
</html>
```

Data Types

– email

Return-path <bmiller@eecs.berkeley.edu>Received from relay2.EECS.Berkeley.EDU (relay2.EECS.Berkeley.EDU [169.229.60.28]) by imap4.CS.Berkeley.EDU (iPlanet Messaging Server 5.2 HotFix 1.16 (built May 14 2003)) with ESMTP id <0HZ000F506JV5S@imap4.CS.Berkeley.EDU>; Tue, 08 Jun 2004 11:40:43 -0700 (PDT)Received from relay3.EECS.Berkeley.EDU (localhost [127.0.0.1]) by relay2.EECS.Berkeley.EDU (8.12.10/8.9.3) with ESMTP id i58Ieg3N000927; Tue, 08 Jun 2004 11:40:43 -0700 (PDT)Received from redbirds (dhcp-168-35.EECS.Berkeley.EDU [128.32.168.35]) by relay3.EECS.Berkeley.EDU (8.12.10/8.9.3) with ESMTP id i58IegFp007613; Tue, 08 Jun 2004 11:40:42 -0700 (PDT)Date Tue, 08 Jun 2004 11:40:42 -0700From Robert Miller <bmiller@eecs.berkeley.edu>Subject RE: SLT headcount = 25In-reply-to <6.1.1.1.0.20040607101523.02623298@imap.eecs.Berkeley.edu>To 'Randy Katz' <randy@eecs.berkeley.edu>Cc "'Glenda J. Smith'" <glendajs@eecs.berkeley.edu>, 'Gert Lanckriet' <gert@eecs.berkeley.edu>Message-id <200406081840.i58IegFp007613@relay3.EECS.Berkeley.EDU>MIME-version 1.0X-MIMEOLE Produced By Microsoft MimeOLE V6.00.2800.1409X-Mailer Microsoft Office Outlook, Build 11.0.5510Content-type multipart/alternative; boundary="----=_NextPart_000_0033_01C44D4D.6DD93AF0"Thread-index AcRMtQRp+R26lVFaRiuz4BfImikTRAA0wf3Qthe headcount is now 32. -- Robert Miller, Administrative Specialist University of California, Berkeley Electronics Research Lab 634 Soda Hall #1776 Berkeley, CA 94720-1776 Phone: 510-642-6037 fax: 510-643-1289

Data Types

- protein sequences

Data Types

– sequences of Unix system calls

Process Management

pid = fork()	Create a child process
s=waitpid(pid, &status, opts)	Wait for a child to terminate
s=execve(name, argv, envp)	Replace a process' core image
exit(status)	Terminate execution
s=sigaction(sig, &act, &oact)	Specify action to take for a signal
s=kill(pid, sig)	Send a signal to process
residual=alarm(seconds)	Schedule a SIGALRM signal later
pause()	Suspend the caller unit until next signal

Memory Management

size=brk(addr)	Set the size of data segment

Input/Output Management

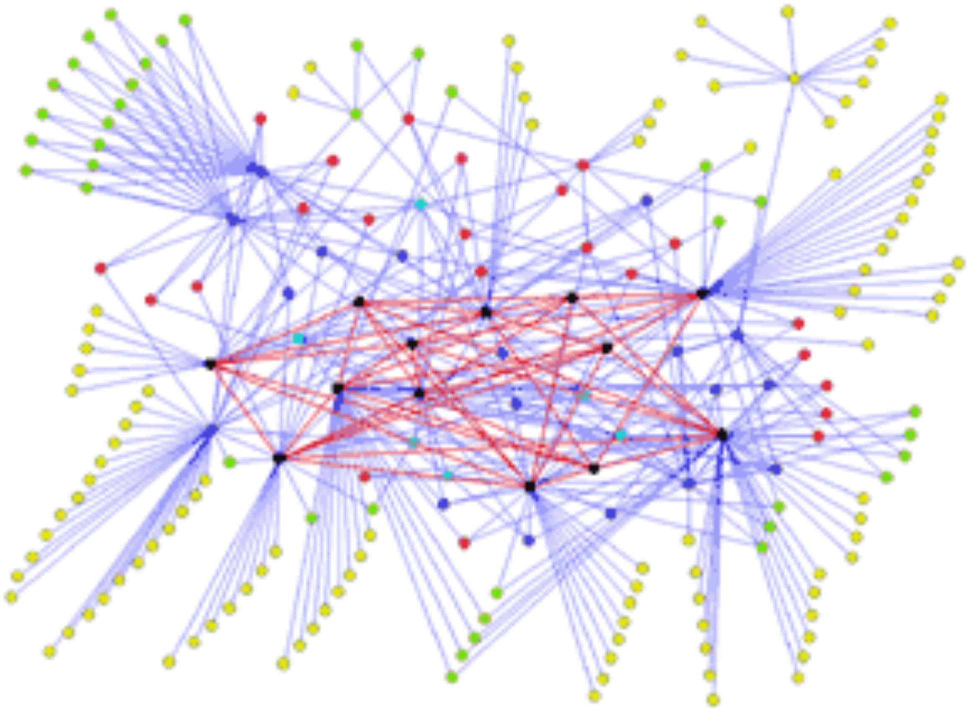
s=cfsetospeed(&termios, speed)	Set the output speed
s=cfsetispeed(&termios, speed)	Set the input speed
s=cfgetospeed(&termios, speed)	Get the output speed
s=cfgetispeed(&termios, speed)	Get the input speed
s=tcsetattr(fd, opt, &termios)	Set terminal attributes
s=tcgetattr(fd, &termios)	Get terminal attributes

Files and Directories Management

fd=create(name, mode)	Create a new file
fd=open(name, how)	Open a file for reading or writing
s=close(fd)	Close an open file
n=read(fd, buffer, nbytes)	Read data from file into a buffer
n=write(fd, buffer, nbytes)	Write data from buffer to file
pos=lseek(fd, offset, whence)	Move the file pointer somewhere
s=start(name, &buf)	Read and return info. about file
s=mkdir(name, mode)	Create a new directory
s=rmdir(name)	Delete an empty directory
s=link(name1, name2)	Create a new directory entry for an old file
s=unlink(name)	Remove a directory entry
s=chdir(dirname)	Change the working directory
s=chmod(name, mode)	Change a file's protection bits

Data Types

– network layout: graph



Data Types

– images

Example: Spam Filter

- Input: email
- Output: spam/ham
- Setup:
  - Get a large collection of example emails, each labeled “spam” or “ham”
  - Note: someone has to hand label all this data
  - Want to learn to predict labels of new, future emails

- Features: The attributes used to make the ham / spam decision
  - Words: FREE!
  - Text Patterns: $dd, CAPS
  - Non-text: SenderInContacts
  - …

Dear Sir.

First, I must solicit your confidence in this transaction, this is by virture of its nature as being utterly confidencial and top secret. …

TO BE REMOVED FROM FUTURE MAILINGS, SIMPLY REPLY TO THIS MESSAGE AND PUT "REMOVE" IN THE SUBJECT.

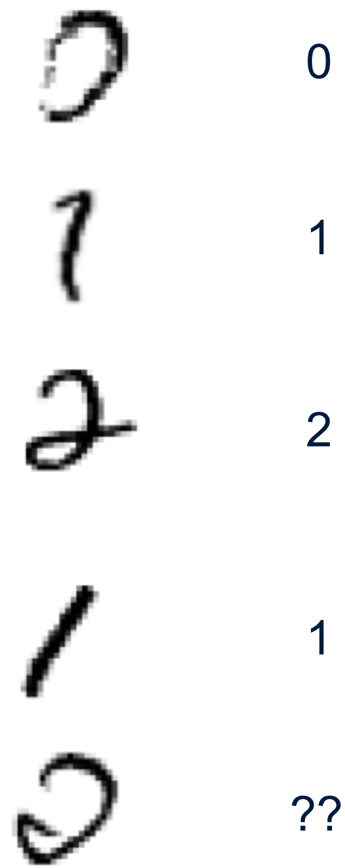
99 MILLION EMAIL ADDRESSES FOR ONLY $99

Ok, Iknow this is blatantly OT but I'm beginning to go insane. Had an old Dell Dimension XPS sitting in the corner and decided to put it to use, I know it was working pre being stuck in the corner, but when I plugged it in, hit the power nothing happened.

Example: Digit Recognition

- Input: images / pixel grids
- Output: a digit 0-9
- Setup:
  - Get a large collection of example images, each labeled with a digit
  - Note: someone has to hand label all this data
  - Want to learn to predict labels of new, future digit images

- Features: The attributes used to make the digit decision
  - Pixels: (6,8)=ON
  - Shape Patterns: NumComponents, AspectRatio, NumLoops
  - …
- Current state-of-the-art: Human-level performance



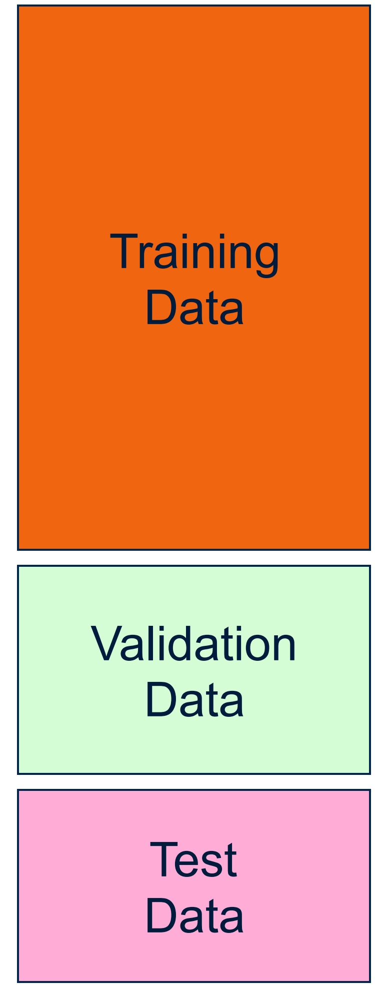
Other Examples of Real-World Classification Tasks

- Fraud detection (input: account activity, classes: fraud / no fraud)
- Web page spam detection (input: HTML/rendered page, classes: spam / ham)
- Speech recognition and speaker recognition (input: waveform, classes: phonemes or words)
- Medical diagnosis (input: symptoms, classes: diseases)
- Automatic essay grader (input: document, classes: grades)
- Customer service email routing and foldering
- Link prediction in social networks
- Catalytic activity in drug design
- … many many more

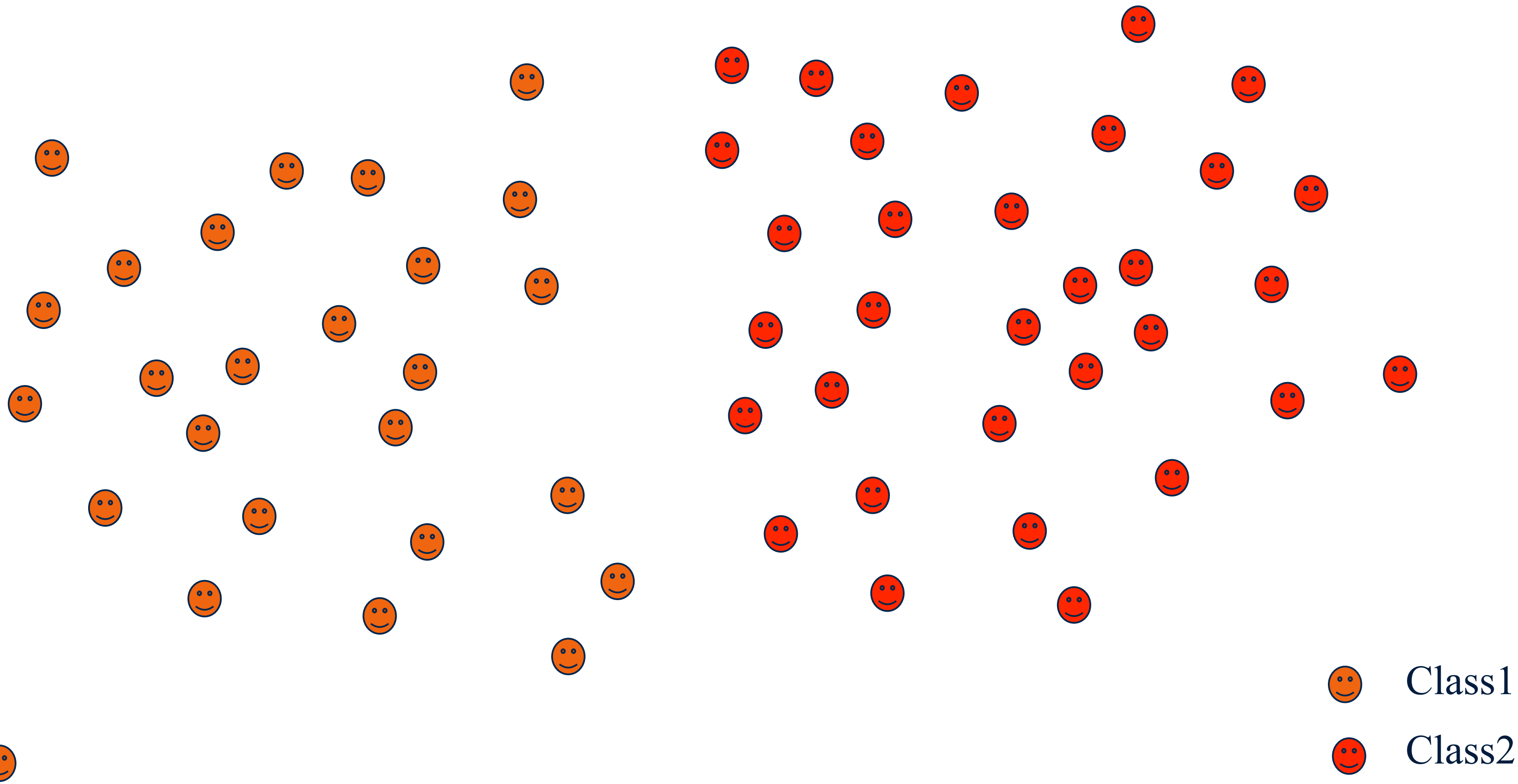
- Classification is an important commercial technology

Training and Validation

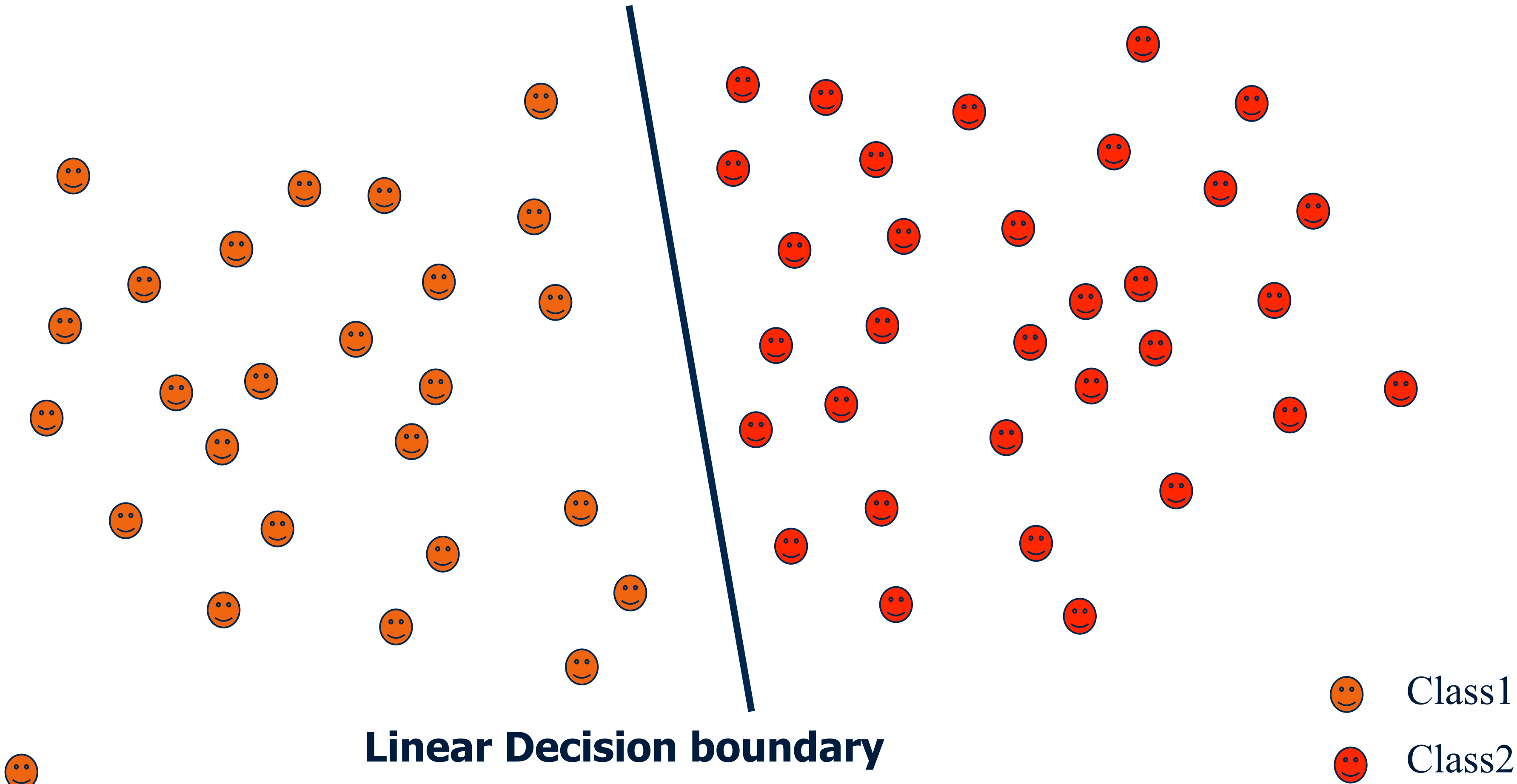
- Data: labeled instances, e.g. emails marked spam/ham
  - Training set
  - Validation set
  - Test set
- Training
  - Estimate parameters on training set
  - Tune hyperparameters on validation set
  - Report results on test set
  - Anything short of this yields over-optimistic claims
- Evaluation
  - Many different metrics
  - Ideally, the criteria used to train the classifier should be closely related to those used to evaluate the classifier
- Statistical issues
  - Want a classifier which does well on *test* data
  - Overfitting: fitting the training data very closely, but not generalizing well
  - Error bars: want realistic (conservative) estimates of accuracy



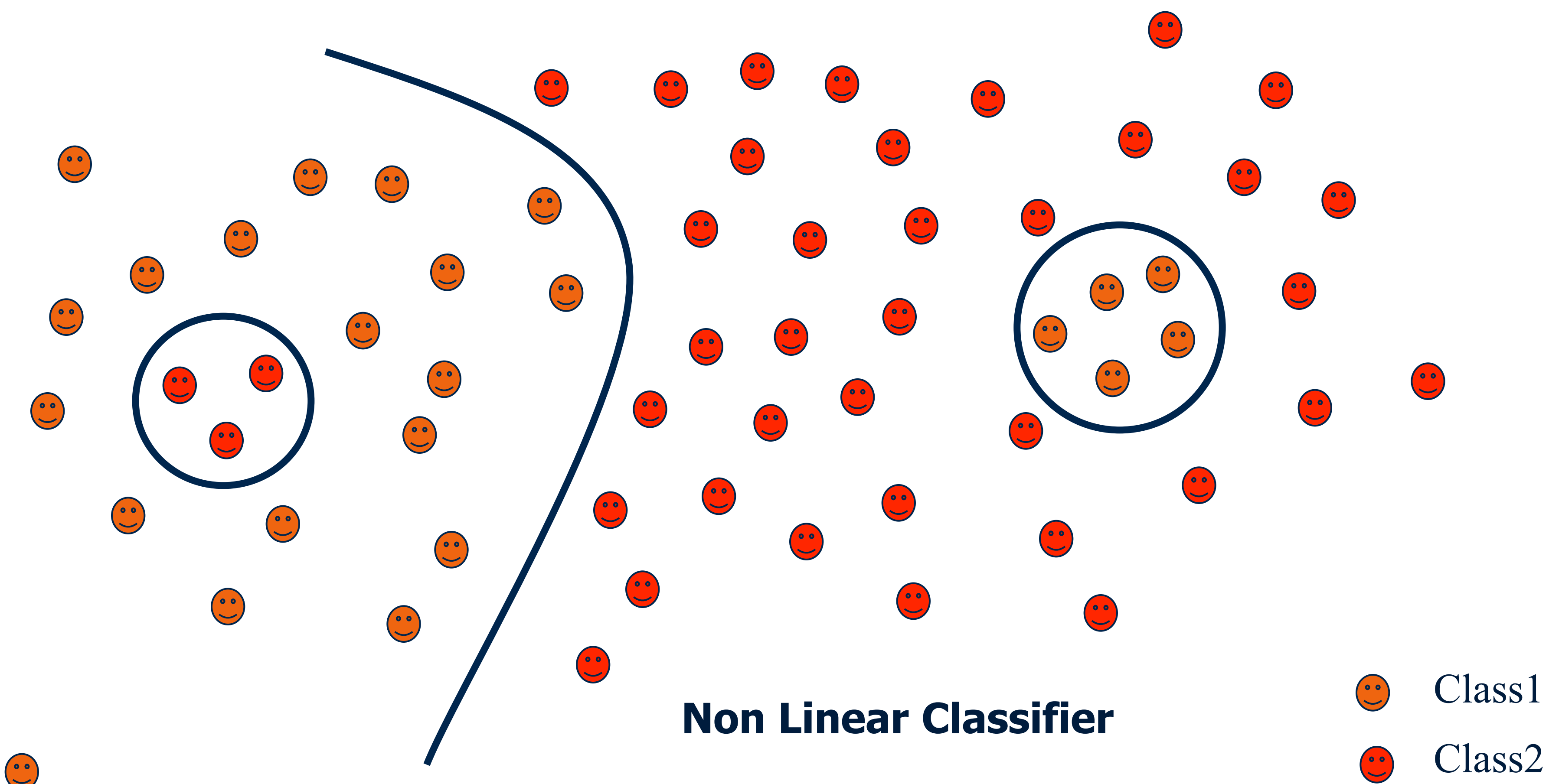
Intuitive Picture of the Problem



Linearly Separable Data



Nonlinearly Separable Data



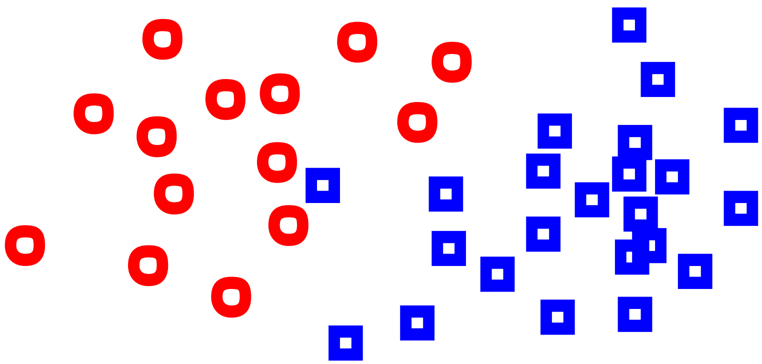
Some Issues

- There may be a simple separator (e.g., a straight line in 2D or a hyperplane in general) or there may not
- There may be “noise” of various kinds
- There may be “overlap”
- One should not be deceived by one’s low-dimensional geometrical intuition
- Some classifiers explicitly represent separators (e.g., straight lines), while for other classifiers the separation is done implicitly
- Some classifiers just make a decision as to which class an object is in; others estimate class probabilities

Methods

I) Instance-based methods:
   1) Nearest neighbor
II) Probabilistic models:
   1) Naïve Bayes
   2) Logistic Regression
III) Linear Models:
   1) Perceptron
   2) Support Vector Machine
IV) Decision Models:
   1) Decision Trees
   2) Boosted Decision Trees
   3) Random Forest

Nearest Neighbour Rule

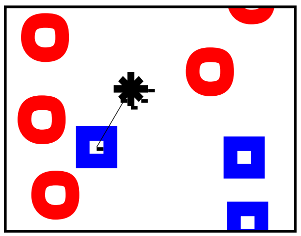


Non-parametric pattern classification.

Consider a two class problem where each sample consists of two measurements (x,y).

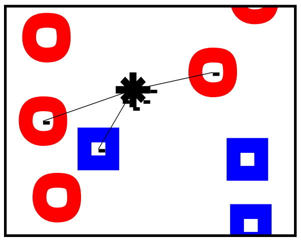
For a given query point q, assign the class of the nearest neighbour.

$k = 1$



Compute the k nearest neighbours and assign the class by majority vote.

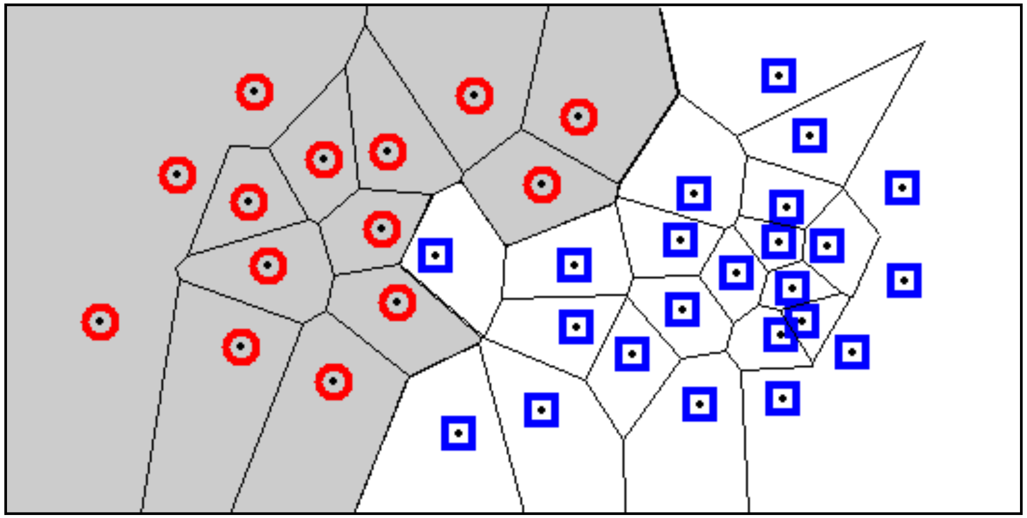
$k = 3$



Questions

- What distance measure to use?
  - Often Euclidean distance is used
  - Locally adaptive metrics
  - More complicated with non-numeric data, or when different dimensions have different scales
- Choice of *k*?
  - Cross-validation
  - 1-NN often performs well in practice
  - k-NN needed for overlapping classes
  - Re-label all data according to k-NN, then classify with 1-NN
  - Reduce *k*-NN problem to 1-NN through dataset editing

Decision Regions



Each cell contains one sample, and every location within the cell is closer to that sample than to any other sample.

A Voronoi diagram divides the space into such cells.

Every query point will be assigned the classification of the sample within that cell. The *decision boundary* separates the class regions based on the 1-NN decision rule.

Knowledge of this boundary is sufficient to classify new points.

The boundary itself is rarely computed; many algorithms seek to retain only those points necessary to generate an identical boundary.

Nearest Neighbor Classification…

- Scaling issues
  - Attributes may have to be scaled to prevent distance measures from being dominated by one of the attributes
  - Example:
    - height of a person may vary from 1.5m to 1.8m
    - weight of a person may vary from 90lb to 300lb
    - income of a person may vary from $10K to $1M

Nearest Neighbor Classification…

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

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

◆ Solution: Normalize the vectors to unit length

k-NN summary

- Pros
  - Can express complex boundary (non-parametric)
  - Very fast training
  - Simple, but still good in practice (e.g. applications in computer vision)
  - Somewhat interpretable by looking at closest points
- Cons
  - Large memory requirements for prediction
  - Not best accuracy among different classifiers

Methods

I) Instance-based methods:
   1) Nearest neighbor
II) Probabilistic models:
   1) Naïve Bayes
   2) Logistic Regression
III) Linear Models:
   1) Perceptron
   2) Support Vector Machine
IV) Decision Models:
   1) Decision Trees
   2) Boosted Decision Trees
   3) Random Forest

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 (A_1, A_2,…,A_n)
  - Goal is to predict class C
  - Specifically, we want to find the value of C that maximizes $P(C| A_1, A_2,…,A_n)$
- Can we estimate $P(C| A_1, A_2,…,A_n)$ directly from data?

Bayesian Classifiers

- Approach:
  - compute the posterior probability $P(C \mid A_1, A_2, \ldots, A_n)$ for all values of C using the Bayes theorem

$$\boxed{P(C \mid A_1 A_2 \ldots A_n) = \frac{P(A_1 A_2 \ldots A_n \mid C) P(C)}{P(A_1 A_2 \ldots A_n)}}$$

  - Choose value of C that maximizes
    $P(C \mid A_1, A_2, \ldots, A_n)$
  - Equivalent to choosing value of C that maximizes
    $P(A_1, A_2, \ldots, A_n \mid C)\, P(C)$
- How to estimate $P(A_1, A_2, \ldots, A_n \mid C)$?

Naïve Bayes Classifier

- Assume independence among attributes A_i when class is given:
  - $P(A_1, A_2, \ldots, A_n | C) = P(A_1| C_j) P(A_2| C_j) \ldots P(A_n| C_j)$
  - Can estimate $P(A_i| C_j)$ for all A_i and C_j.
  - New point is classified to C_j if $P(C_j) \Pi P(A_i| C_j)$ 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: $P(C) = N_c/N$
  - e.g., P(No) = 7/10,
    P(Yes) = 3/10
- For discrete attributes:

$$P(A_i \mid C_k) = |A_{ik}| / N_{c_k}$$

  - where $|A_{ik}|$ is number of instances having attribute A_i and belongs to class C_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
    - one ordinal attribute per bin
    - violates independence assumption
  - Two-way split: (A < v) or (A > v) k
    - choose only one of the two splits as new attribute
  - 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 $P(A_i|c)$

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:

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

  - One for each (A_i,c_i) pair
- For (Income, Class=No):
  - If Class=No
    - sample mean = 110
    - sample variance = 2975

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

Example of Naïve Bayes Classifier

Given a Test Record:

$$X = (\text{Refund} = \text{No}, \text{Married}, \text{Income} = 120\text{K})$$

naive Bayes Classifier:

P(Refund=Yes|No) = 3/7
P(Refund=No|No) = 4/7
P(Refund=Yes|Yes) = 0
P(Refund=No|Yes) = 1
P(Marital Status=Single|No) = 2/7
P(Marital Status=Divorced|No)=1/7
P(Marital Status=Married|No) = 4/7
P(Marital Status=Single|Yes) = 2/7
P(Marital Status=Divorced|Yes)=1/7
P(Marital Status=Married|Yes) = 0

For taxable income:
If class=No: sample mean=110
sample variance=2975
If class=Yes: sample mean=90
sample variance=25

- P(X|Class=No) = P(Refund=No|Class=No)
  $\times$ P(Married| Class=No)
  $\times$ P(Income=120K| Class=No)
  = 4/7 $\times$ 4/7 $\times$ 0.0072 = 0.0024

- P(X|Class=Yes) = P(Refund=No| Class=Yes)
  $\times$ P(Married| Class=Yes)
  $\times$ P(Income=120K| Class=Yes)
  = $1 \times 0 \times 1.2 \times 10^{-9} = 0$

Since P(X|No)P(No) > P(X|Yes)P(Yes)

Therefore P(No|X) > P(Yes|X)
=> Class = No

Naïve Bayes Classifier

- If one of the conditional probability is zero, then the entire expression becomes zero
- Probability estimation:

$$\text{Original}: P(A_i \mid C) = \frac{N_{ic}}{N_c}$$

$$\text{Laplace}: P(A_i \mid C) = \frac{N_{ic} + 1}{N_c + c}$$

$$\text{m - estimate}: P(A_i \mid C) = \frac{N_{ic} + mp}{N_c + m}$$

c: number of classes

p: prior probability

m: parameter

Example of Naïve Bayes Classifier

Name	Give Birth	Can Fly	Live in Water	Have Legs	Class
human	yes	no	no	yes	mammals
python	no	no	no	no	non-mammals
salmon	no	no	yes	no	non-mammals
whale	yes	no	yes	no	mammals
frog	no	no	sometimes	yes	non-mammals
komodo	no	no	no	yes	non-mammals
bat	yes	yes	no	yes	mammals
pigeon	no	yes	no	yes	non-mammals
cat	yes	no	no	yes	mammals
leopard shark	yes	no	yes	no	non-mammals
turtle	no	no	sometimes	yes	non-mammals
penguin	no	no	sometimes	yes	non-mammals
porcupine	yes	no	no	yes	mammals
eel	no	no	yes	no	non-mammals
salamander	no	no	sometimes	yes	non-mammals
gila monster	no	no	no	yes	non-mammals
platypus	no	no	no	yes	mammals
owl	no	yes	no	yes	non-mammals
dolphin	yes	no	yes	no	mammals
eagle	no	yes	no	yes	non-mammals

A: attributes

M: mammals

N: non-mammals

$$P(A|M) = \frac{6}{7} \times \frac{6}{7} \times \frac{2}{7} \times \frac{2}{7} = 0.06$$

$$P(A|N) = \frac{1}{13} \times \frac{10}{13} \times \frac{3}{13} \times \frac{4}{13} = 0.0042$$

$$P(A|M)P(M) = 0.06 \times \frac{7}{20} = 0.021$$

$$P(A|N)P(N) = 0.004 \times \frac{13}{20} = 0.0027$$

Give Birth	Can Fly	Live in Water	Have Legs	Class
yes	no	yes	no	?

P(A|M)P(M) > P(A|N)P(N)

=> Mammals

Naïve Bayes Summary

- Robust to isolated noise points
- Handle missing values by ignoring the instance during probability estimate calculations
- Robust to irrelevant attributes
- Independence assumption may not hold for some attributes
  - Use other techniques such as Bayesian Belief Networks (BBN)
- Naïve Bayes can produce a probability estimate, but it is usually a very biased one
  - Logistic Regression is better for obtaining probabilities.

Methods

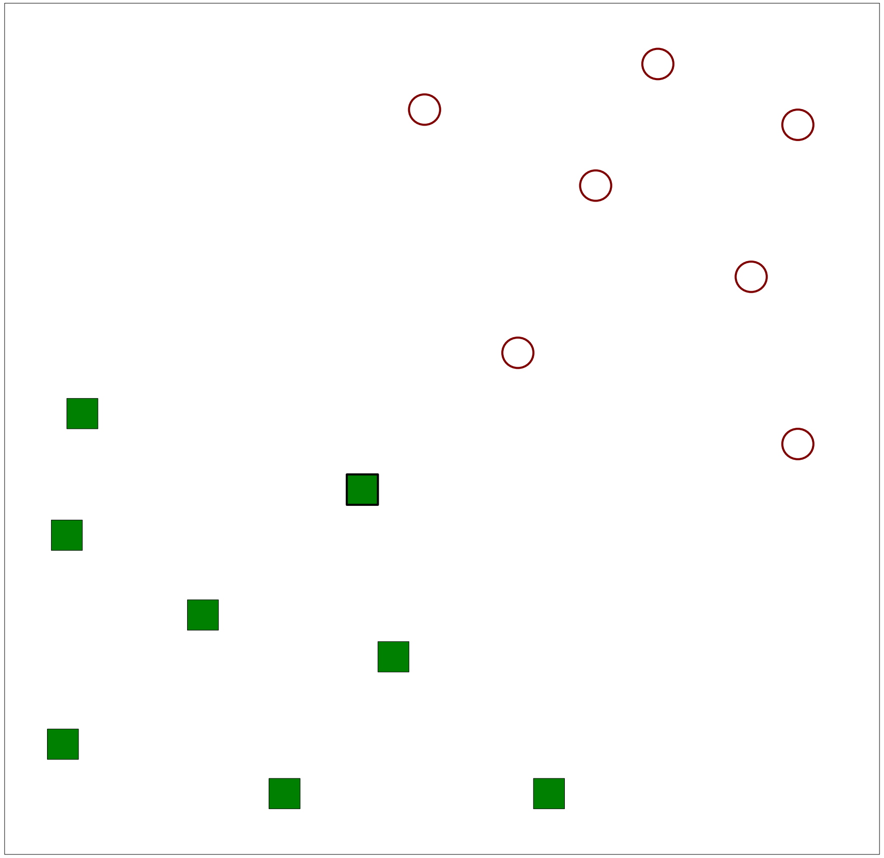
I) Instance-based methods:
   1) Nearest neighbor

II) Probabilistic models:
   1) Naïve Bayes
   2) Logistic Regression

III) Linear Models:
   1) Perceptron
   2) Support Vector Machine

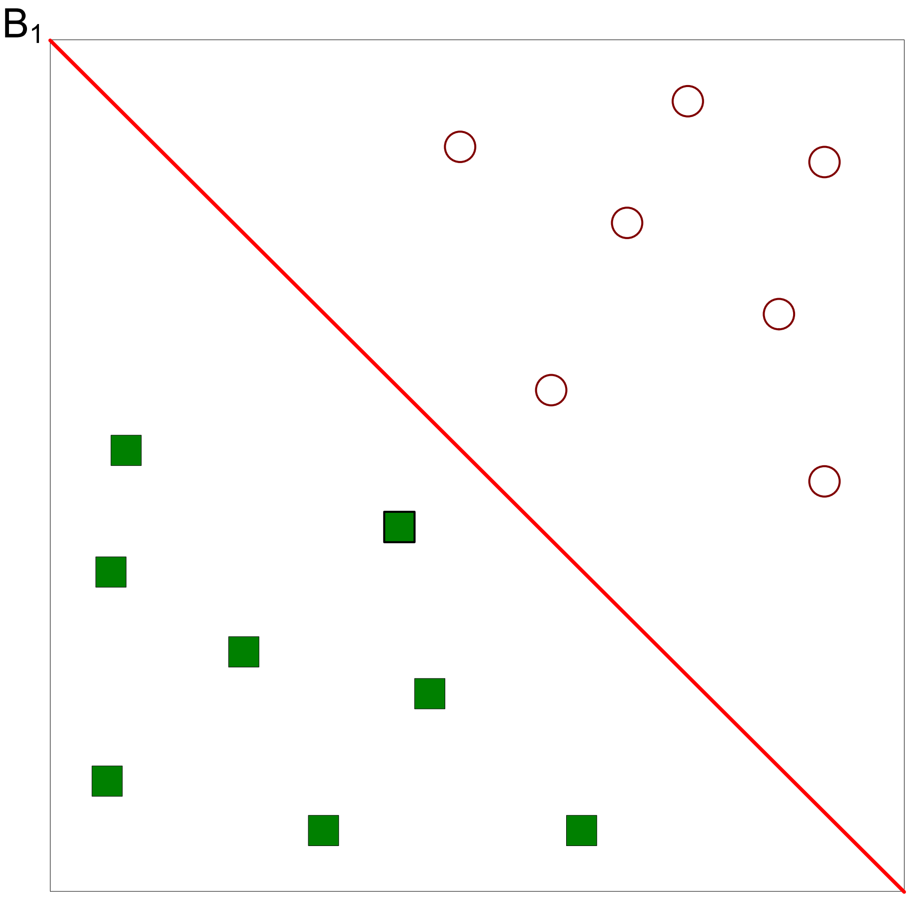
IV) Decision Models:
   1) Decision Trees
   2) Boosted Decision Trees
   3) Random Forest

Support Vector Machines



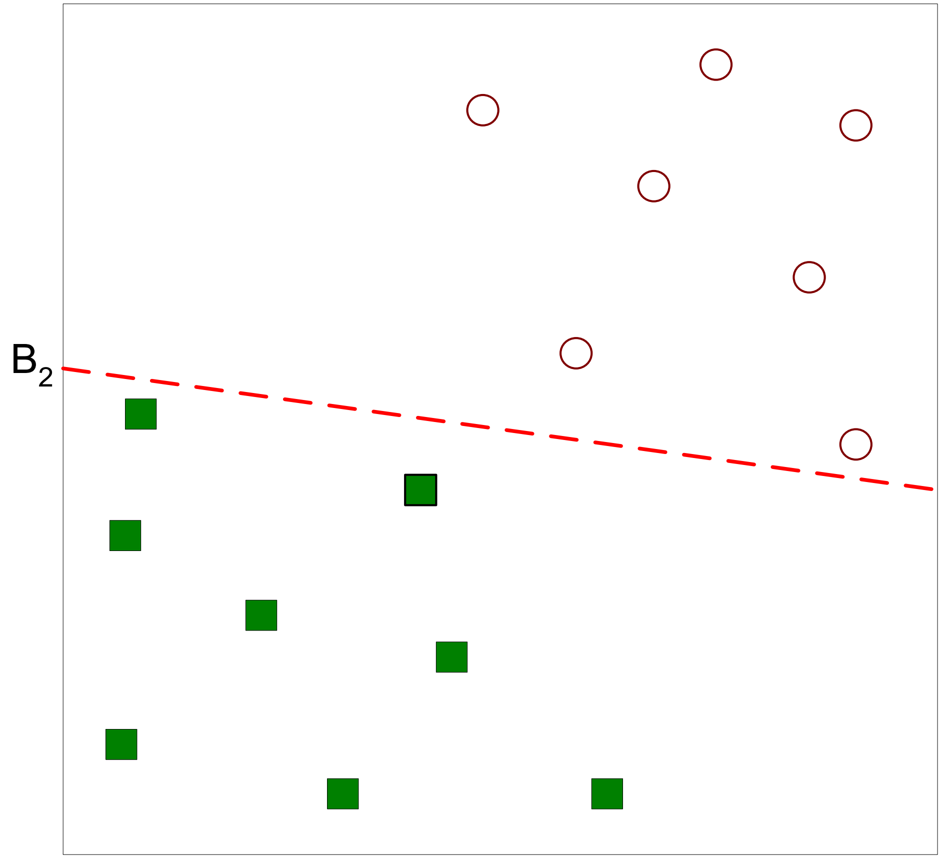
- Find a linear hyperplane (decision boundary) that will separate the data

Support Vector Machines



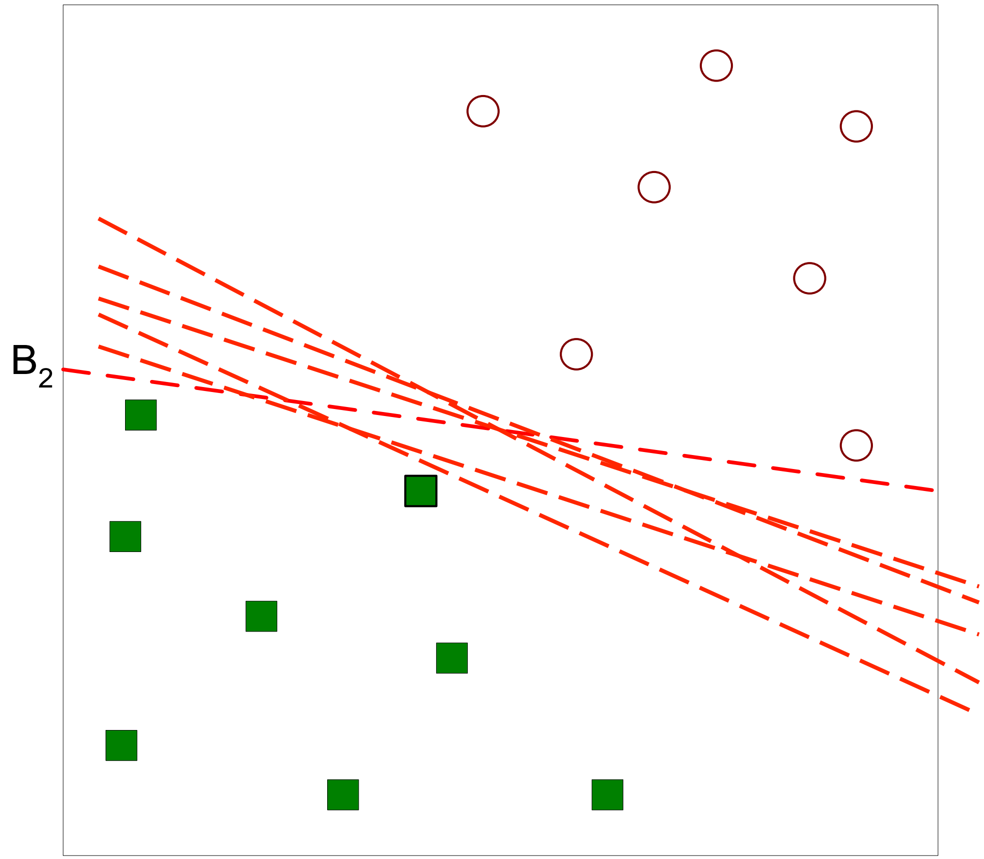
- One Possible Solution

Support Vector Machines



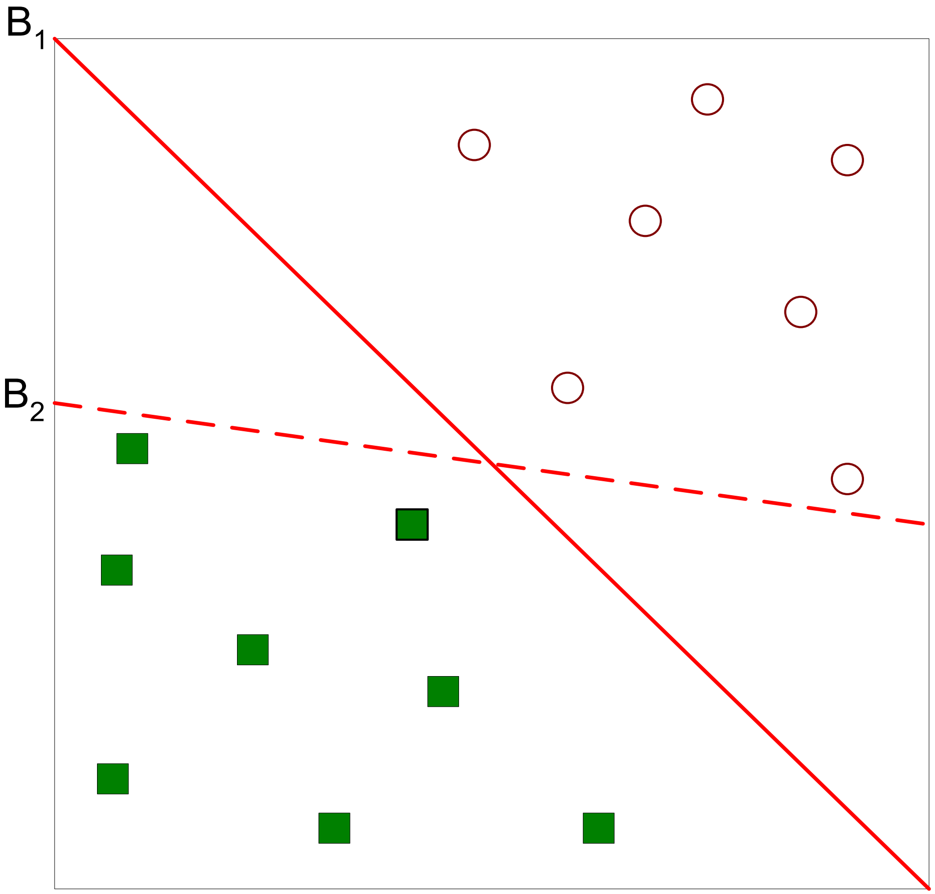
- Another possible solution

Support Vector Machines



- Other possible solutions

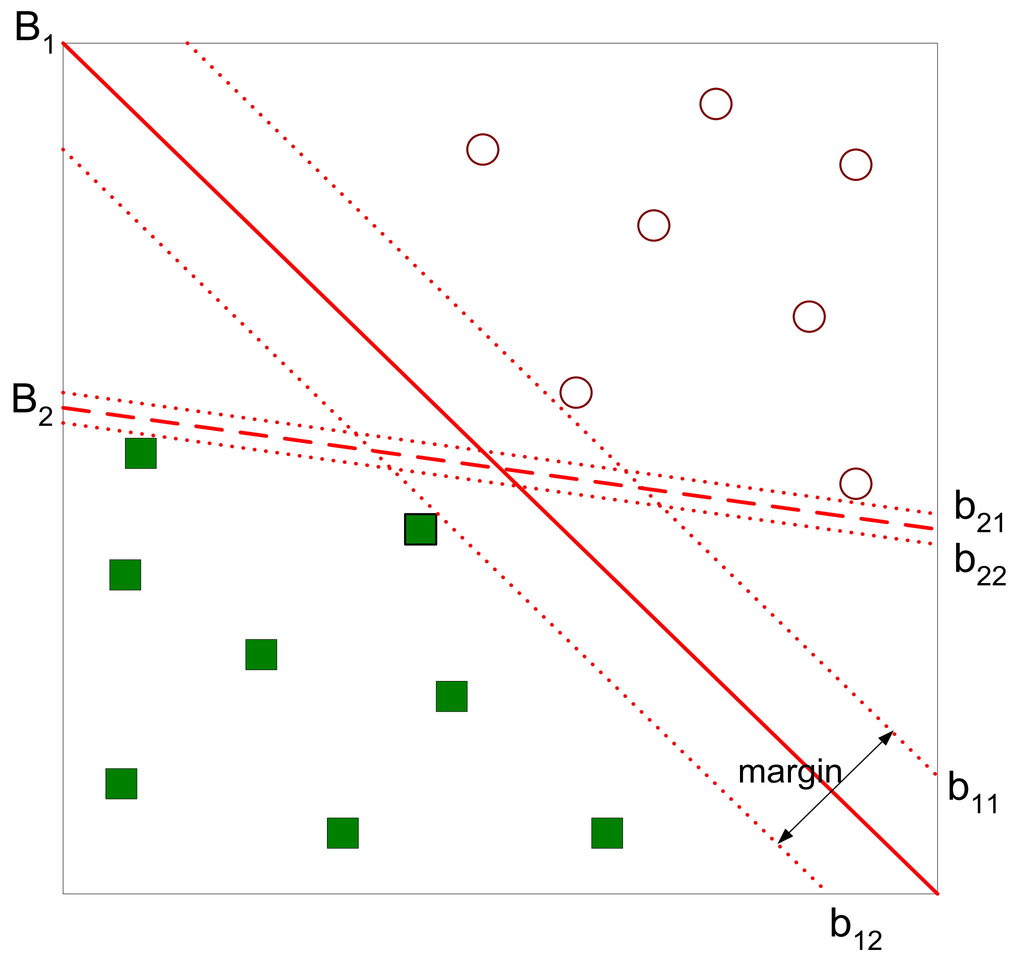
Support Vector Machines



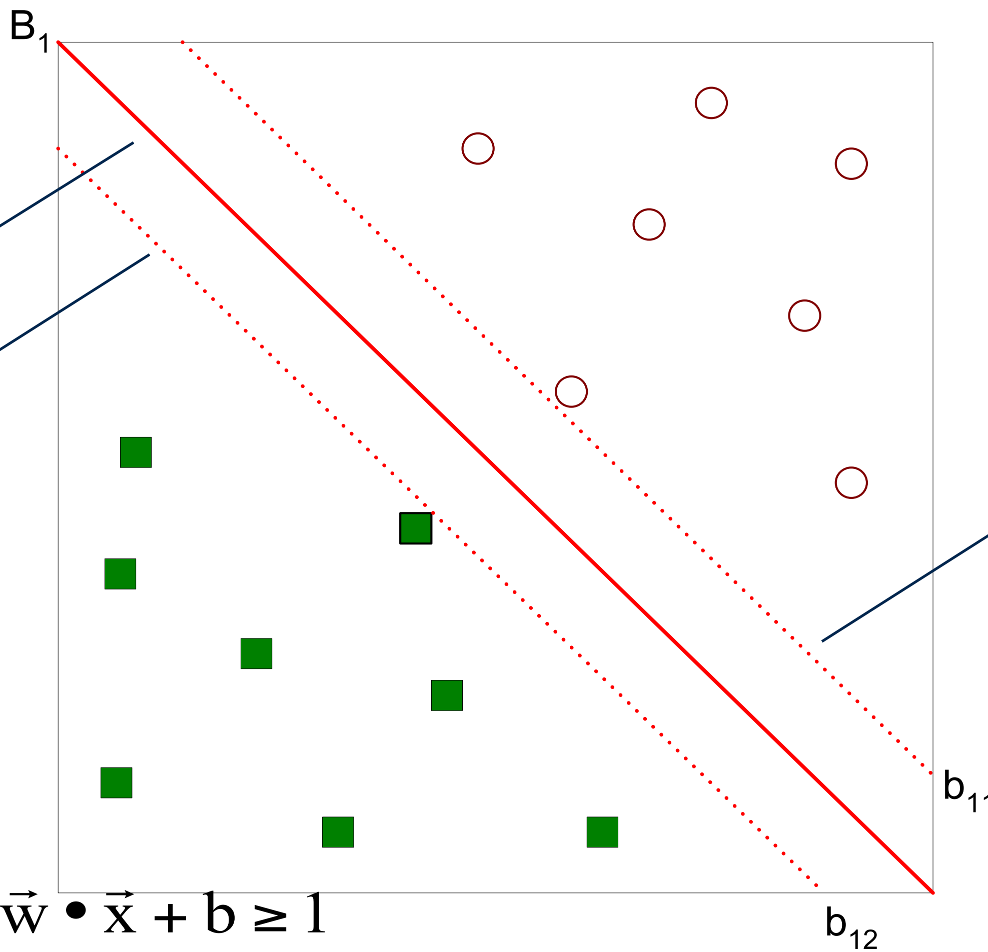
- Which one is better? B1 or B2?
- How do you define better?

Support Vector Machines

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



Support Vector Machines



$$\vec{w} \bullet \vec{x} + b = 0$$

$$\vec{w} \bullet \vec{x} + b = -1$$

$$\vec{w} \bullet \vec{x} + b = +1$$

$$f(\vec{x}) = \begin{cases} 1 & \text{if } \vec{w} \bullet \vec{x} + b \geq 1 \\ -1 & \text{if } \vec{w} \bullet \vec{x} + b \leq -1 \end{cases}$$

$$\text{Margin} = \frac{2}{\|\vec{w}\|^2}$$

Support Vector Machines

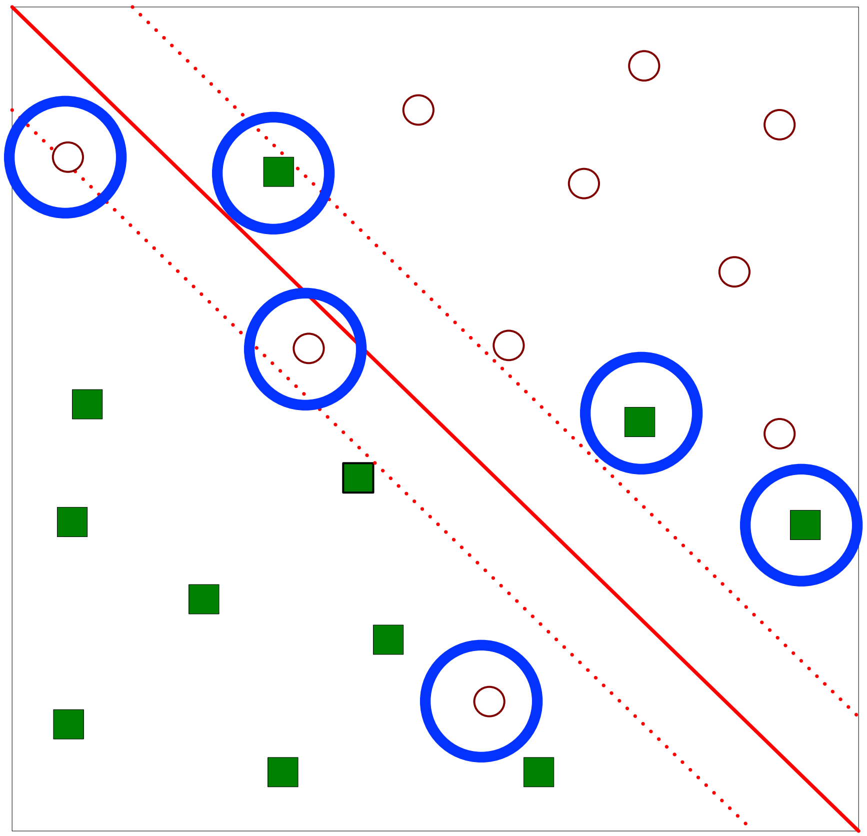
- We want to maximize: $\text{Margin} = \frac{2}{\|\vec{w}\|^2}$
  - Which is equivalent to minimizing: $L(w) = \frac{\|\vec{w}\|^2}{2}$
  - But subjected to the following constraints:

$$f(\vec{x}_i) = \begin{cases} 1 & \text{if } \vec{\mathrm{w}} \bullet \vec{\mathrm{x}}_\mathrm{i} + \mathrm{b} \geq 1 \\ -1 & \text{if } \vec{\mathrm{w}} \bullet \vec{\mathrm{x}}_\mathrm{i} + \mathrm{b} \leq -1 \end{cases}$$

  - This is a constrained optimization problem
    - Numerical approaches to solve it (e.g., quadratic programming)

Support Vector Machines

- What if the problem is not linearly separable?

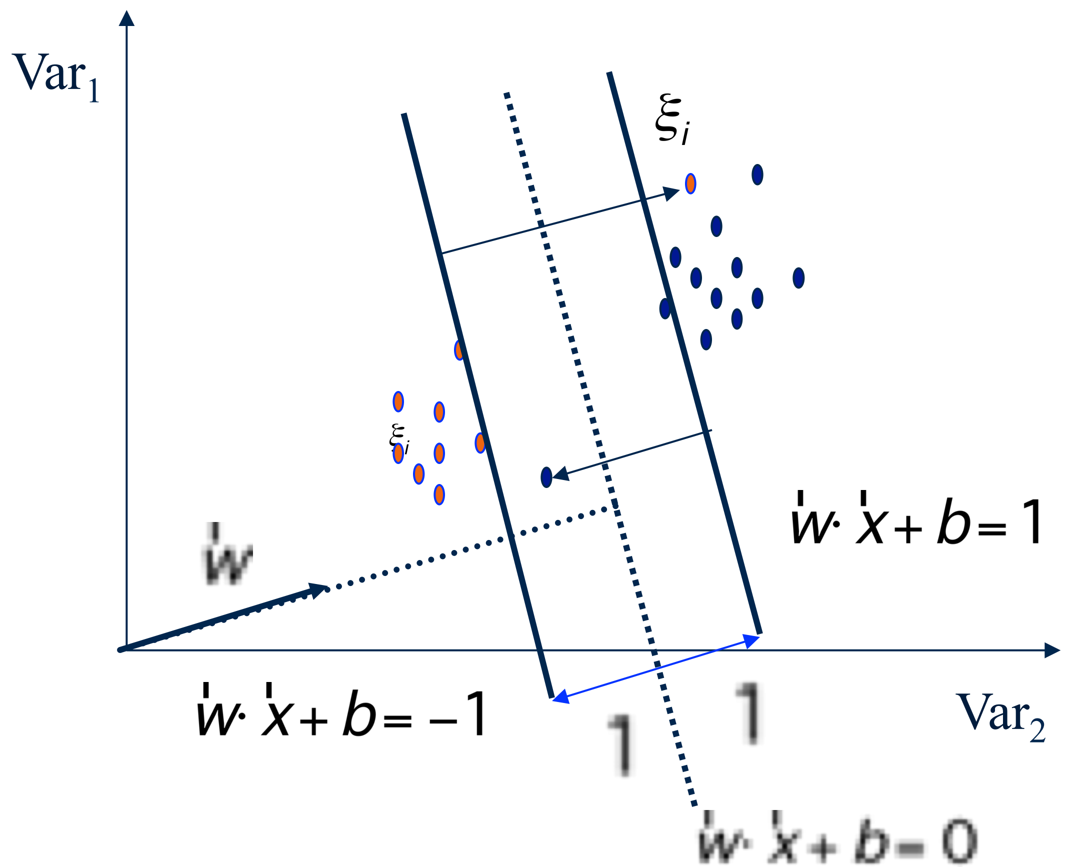


Support Vector Machines

- What if the problem is not linearly separable?
  - Introduce slack variables $L(w) = \frac{\| \vec{w} \|^2}{2} + C\left(\sum_{i=1}^{N} \xi_i^k \right)$
    - Need to minimize:
    - Subject to:

$$f(\vec{x}_i) = \begin{cases} 1 & \text{if } \vec{\mathrm{w}} \bullet \vec{\mathrm{x}}_\mathrm{i} + \mathrm{b} \geq 1 - \xi_\mathrm{i} \\ -1 & \text{if } \vec{\mathrm{w}} \bullet \vec{\mathrm{x}}_\mathrm{i} + \mathrm{b} \leq -1 + \xi_\mathrm{i} \end{cases}$$

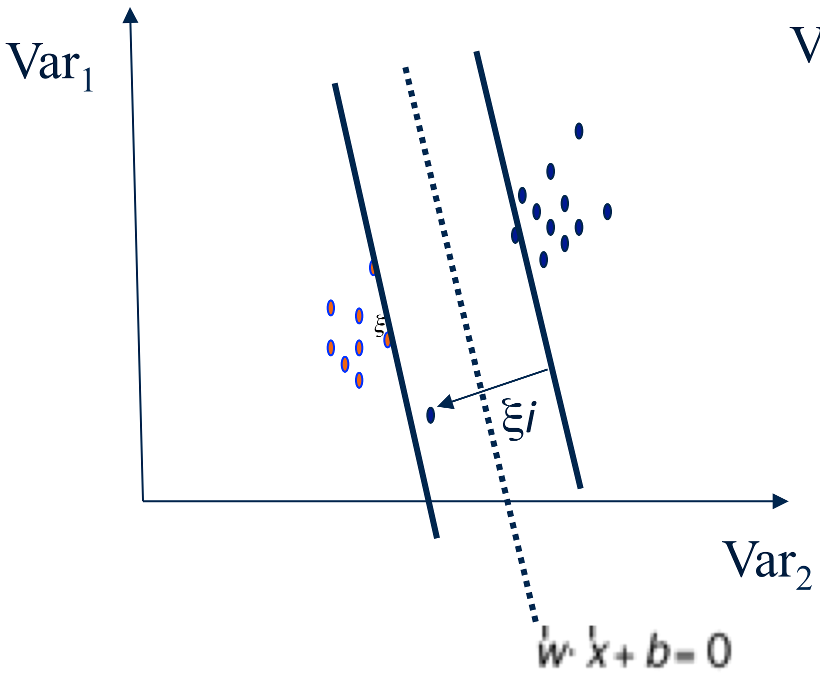
Nonlinearly Separable Data



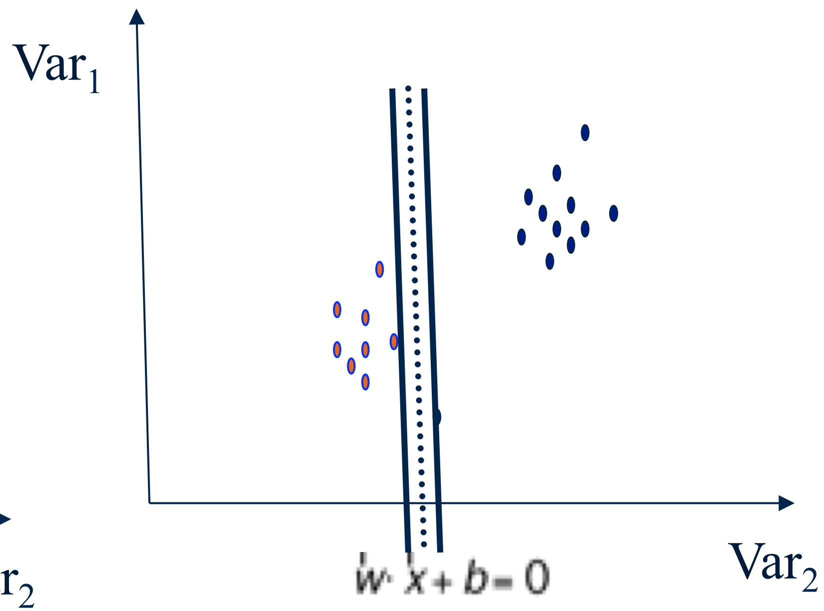
Introduce slack variables ξ_i

Allow some instances to fall within the margin, but penalize them

Robustness of Soft vs Hard Margin SVMs



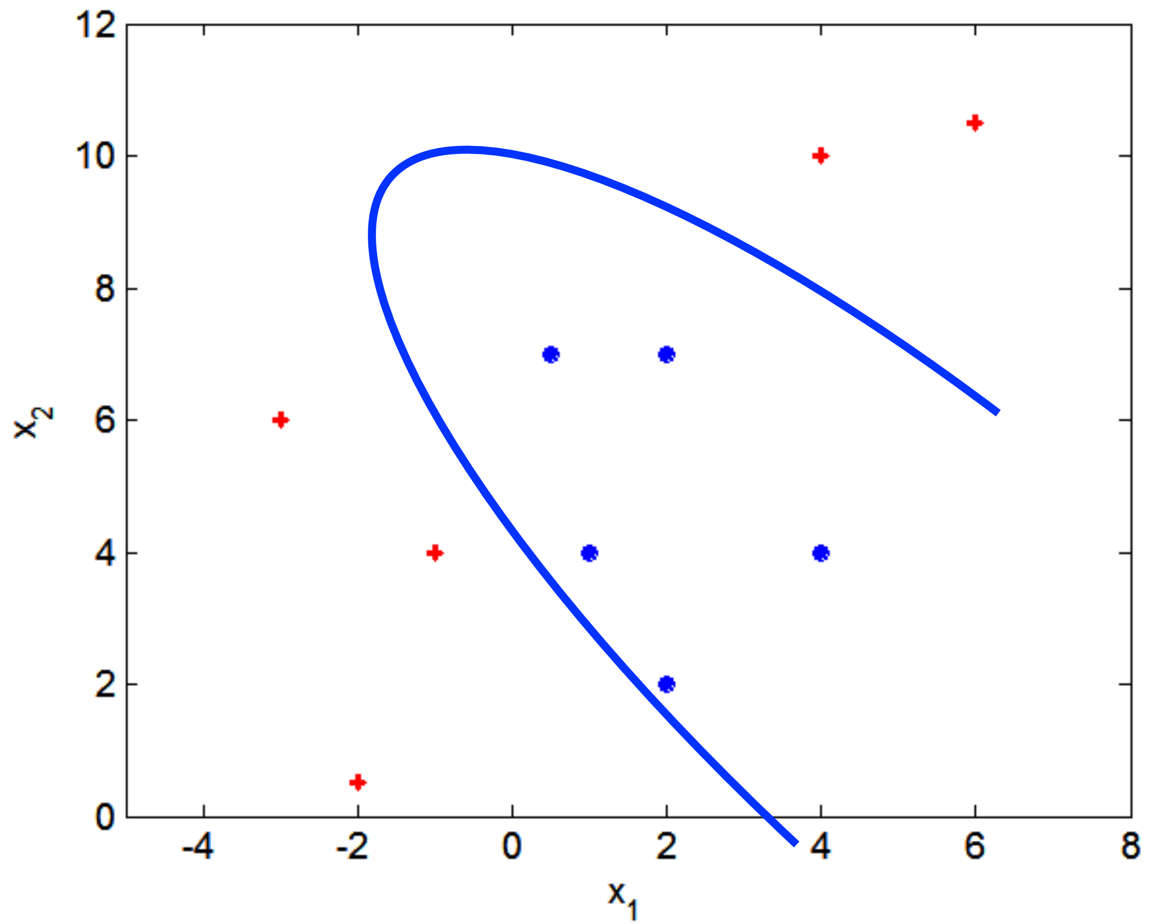
Soft Margin SVN



Hard Margin SVN

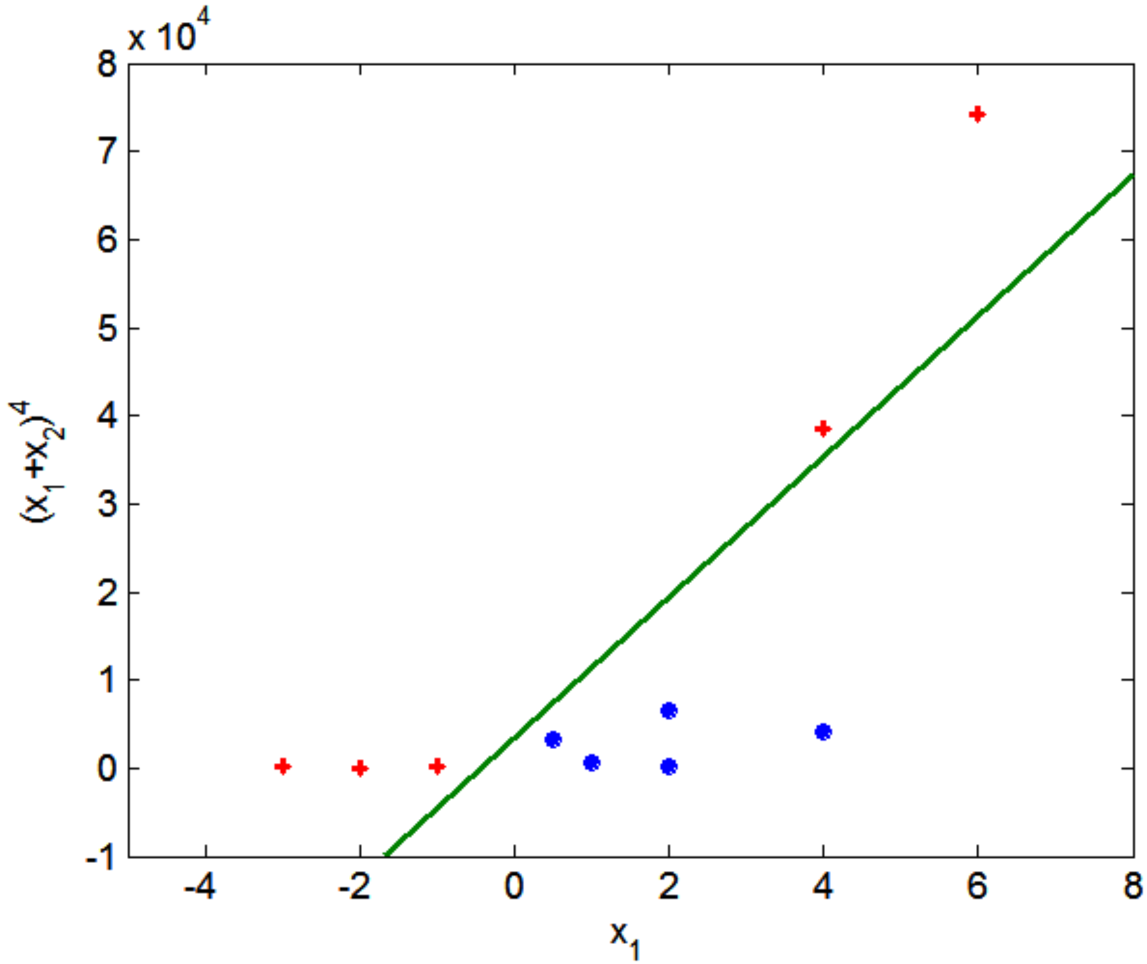
Nonlinear Support Vector Machines

- What if things are not good?

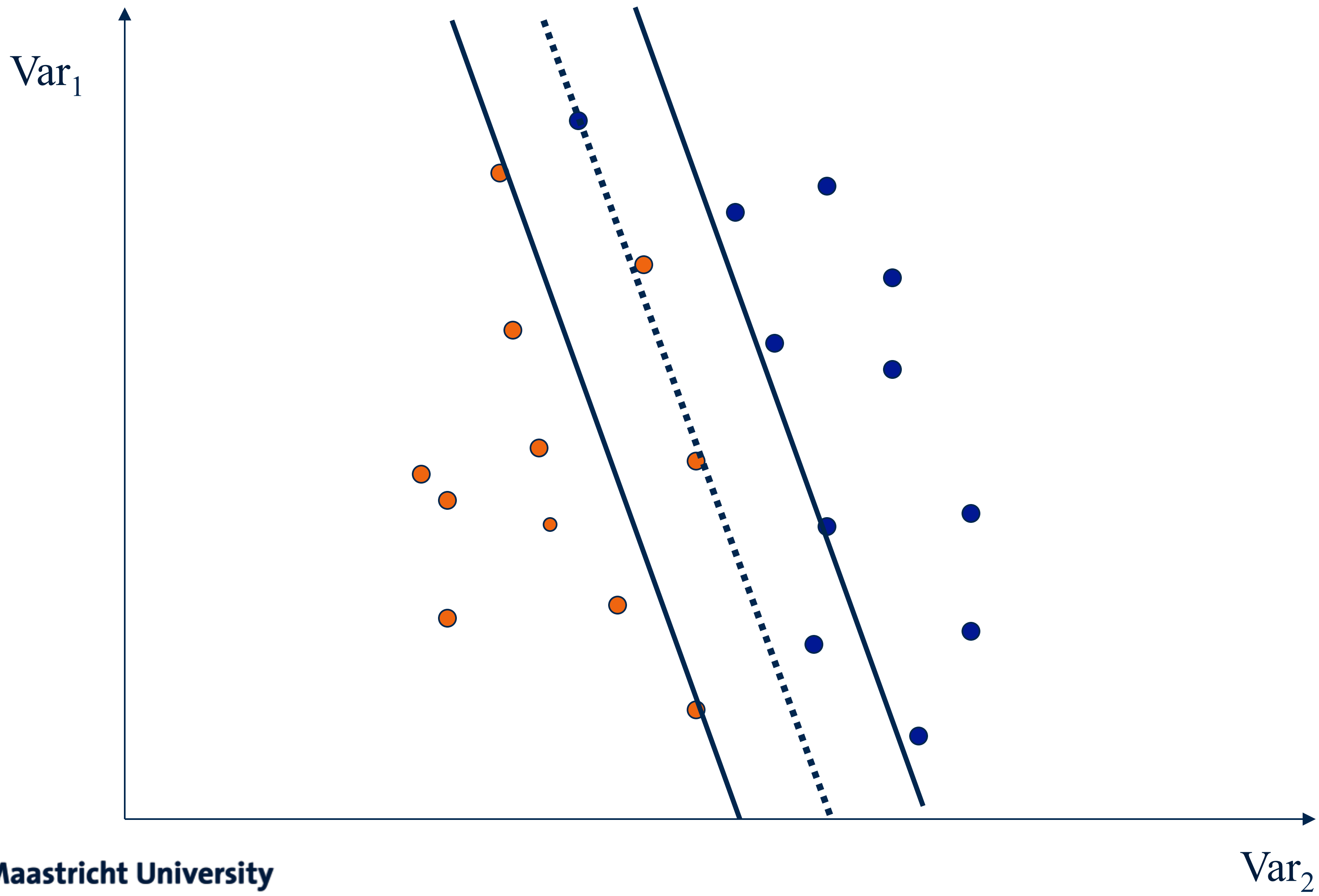


Nonlinear Support Vector Machines

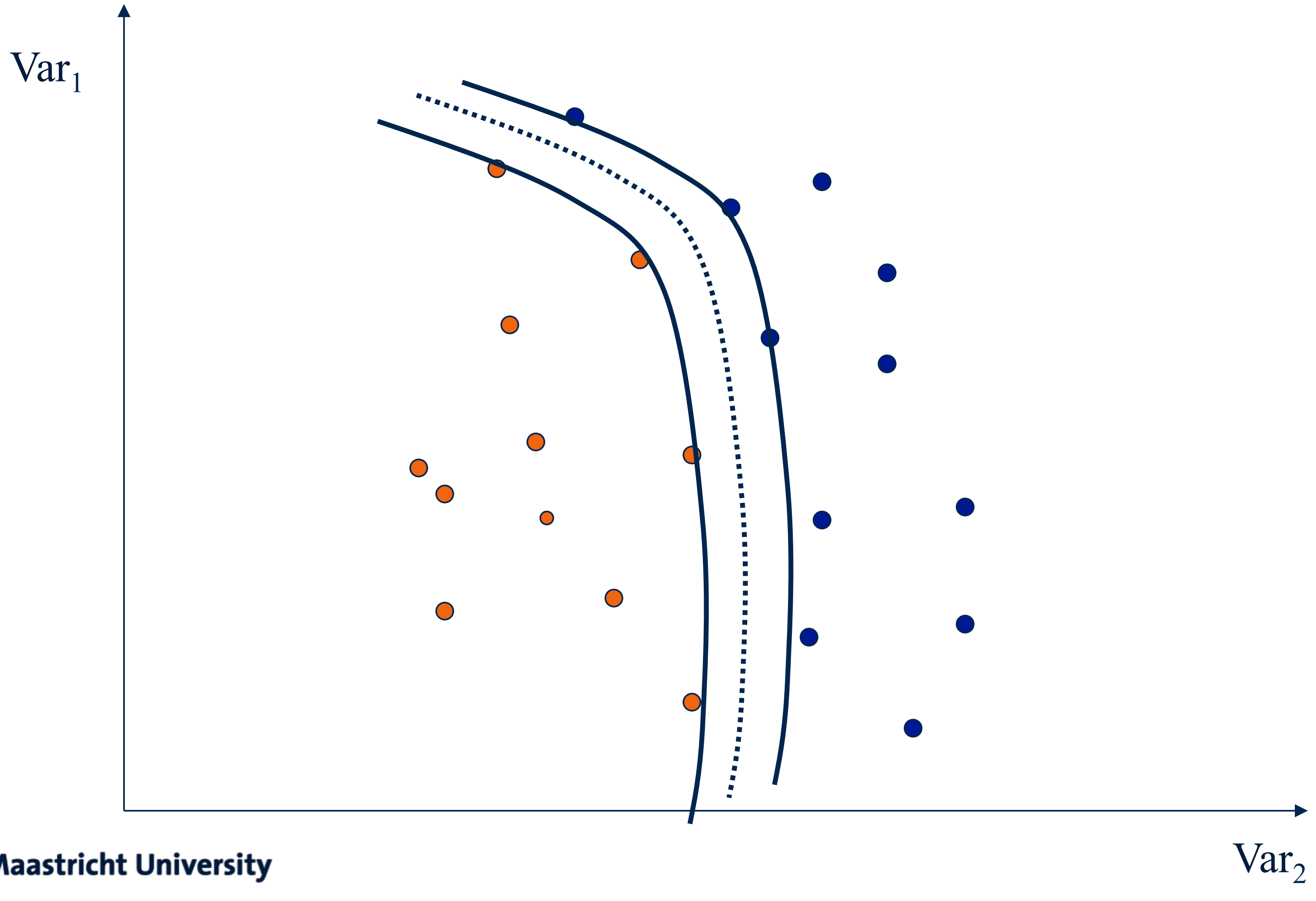
- Transform data into higher dimensional space



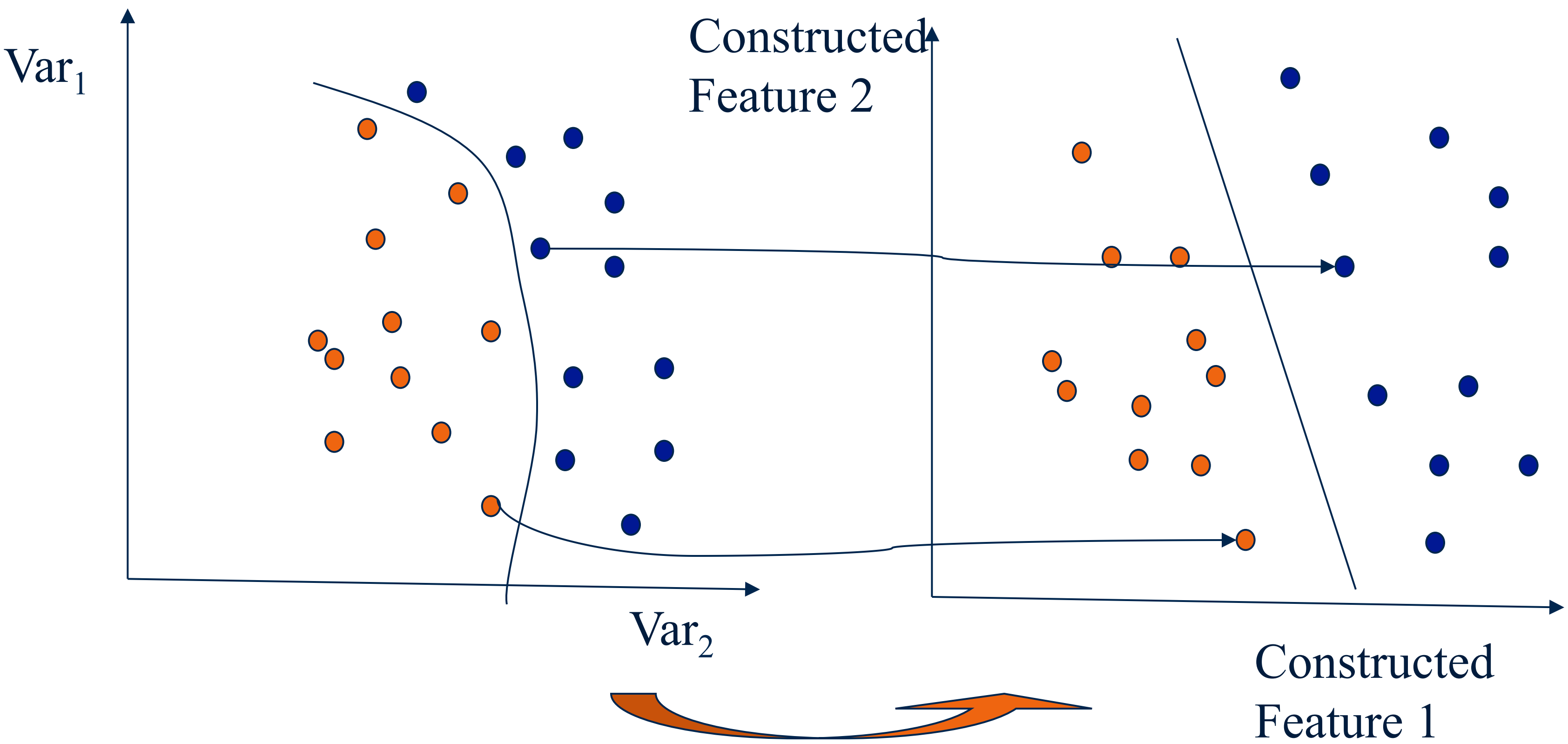
Disadvantages of Linear Decision Surfaces



Advantages of Nonlinear Surfaces



Linear Classifiers in High-Dimensional Spaces



Find function $\Phi(x)$ to map to a different space

Mapping Data to a High-Dimensional Space

- Find function $\Phi(x)$ to map to a different space, then SVM formulation becomes:

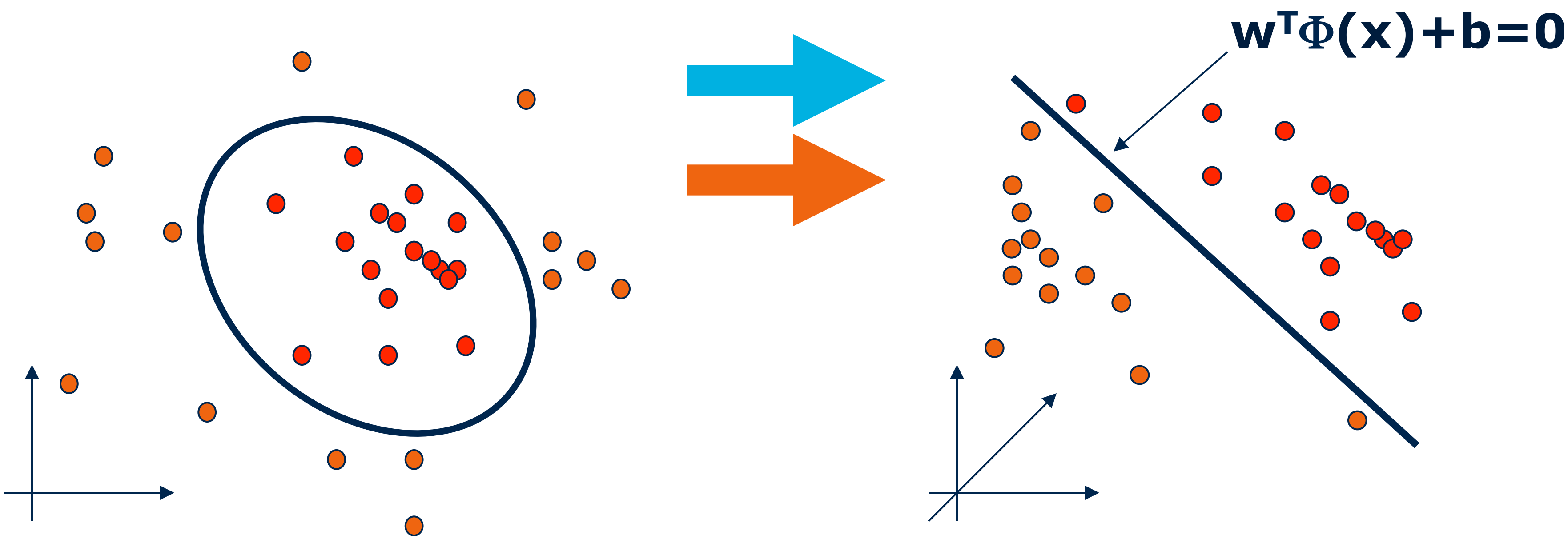
$$\min \frac{1}{2}\|w\|^2 + C\sum_i \xi_i \qquad s.t.\; y_i(w \cdot \Phi(x) + b) \geq 1 - \xi_i, \forall x_i$$
$$\xi_i \geq 0$$

- Data appear as $\Phi(x)$, weights *w* are now weights in the new space
- Explicit mapping expensive if $\Phi(x)$ is very high dimensional
- Solving the problem without explicitly mapping the data is desirable

The Kernel Trick

- $\Phi(x_i) \cdot \Phi(x_j)$: means, map data into new space, then take the inner product of the new vectors
- We can find a function such that: $K(x_i \cdot x_j) = \Phi(x_i) \cdot \Phi(x_j)$, i.e., the image of the inner product of the data is the inner product of the images of the data
- Then, we do not need to explicitly map the data into the high-dimensional space to solve the optimization problem

Example



$X=[x\ z]$ $\qquad$ $\Phi(X)=[x^2\ z^2\ xz]$

$f(x) = \mathrm{sign}(w_1x^2+w_2z^2+w_3xz + b)$

Example

$X_1=[x_1\ z_1]$
$X_2=[x_2\ z_2]$

$\Rightarrow \Phi(X_1)=[x_1^2\ z_1^2\ 2^{1/2}x_1z_1]$

$\Rightarrow \Phi(X_2)=[x_2^2\ z_2^2\ 2^{1/2}x_2z_2]$

$$\Phi(X_1)^T\Phi(X_2)= [x_1^2\ z_1^2\ 2^{1/2}x_1z_1]\ [x_2^2\ z_2^2\ 2^{1/2}x_2z_2]^T$$

Expensive!
O(d²)

$$= x_1^2z_1^2 + x_2^2\ z_2^2 + 2\ x_1\ z_1\ x_2\ z_2$$

$$= (x_1z_1 + x_2z_2)^2$$

Efficient!
O(d)

$$= (X_1^T\ X_2)^2$$

Maastricht University

Kernel Trick

- **Kernel function**: a symmetric function

  k : R^d x R^d -> R
- **Inner product kernels**: additionally,

  $\mathbf{k(x,z) = \Phi(x)^T \Phi(z)}$
- Example:

O(d²)

O(d)

$$\boxed{\Phi(x)^T \Phi(z) = \sum_{i,j=(1,1)}^{d,d} (x_i x_j)(z_i z_j)} = \left(\sum_{i=1}^{d} x_i z_i \right)^2 = \boxed{(x^T z)^2 = K(x,z)}$$

Kernel Trick

- Implement an infinite-dimensional mapping implicitly
- Only inner products explicitly needed for training and evaluation
- Inner products computed efficiently, in finite dimensions
- The underlying mathematical theory is that of reproducing kernel Hilbert space from functional analysis

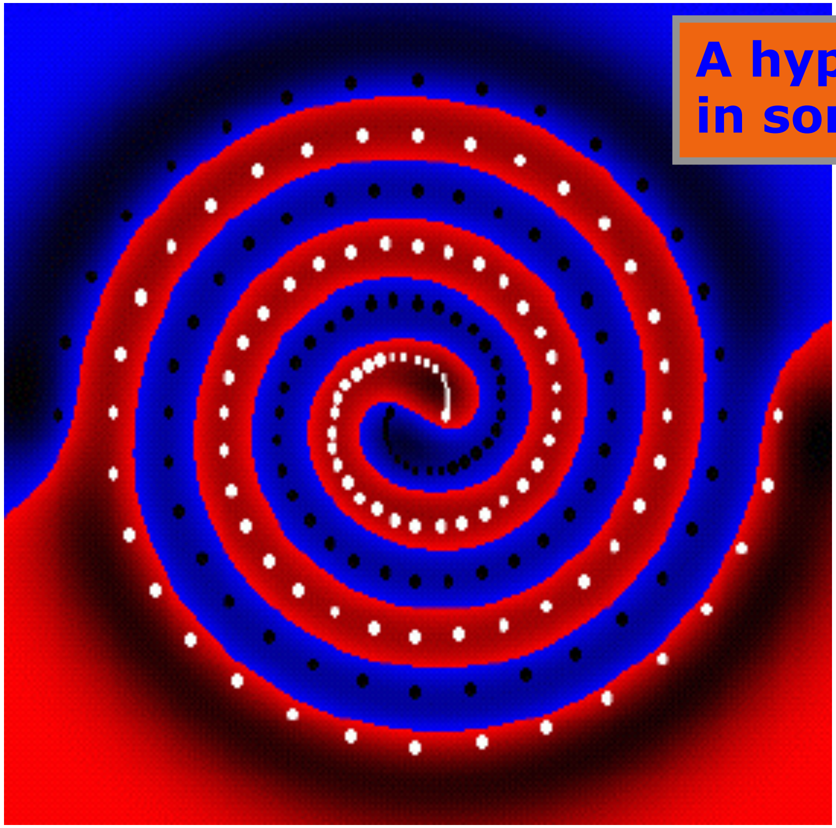
Kernel Methods

- If a linear algorithm can be expressed only in terms of inner products
  - it can be “kernelized”
  - find linear pattern in high-dimensional space
  - nonlinear relation in original space
- Specific kernel function determines nonlinearity

Kernels

- Some simple kernels
  - Linear kernel: **$k(x,z) = x^T z$**
    → equivalent to linear algorithm
  - Polynomial kernel: **$k(x,z) = (1+x^T z)^d$**
    → polynomial decision rules
  - RBF kernel: **$k(x,z) = \exp(-||x-z||^2/2\sigma)$**
    → highly nonlinear decisions

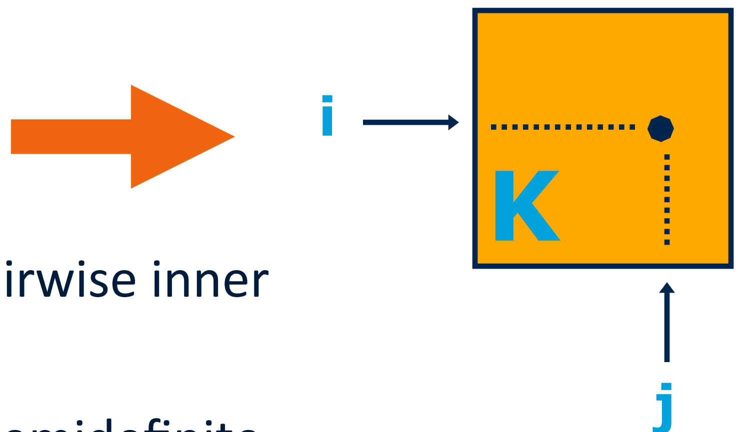
Gaussian Kernel: Example



A hyperplane
in some space

Kernel Matrix

$$k(x,y)$$



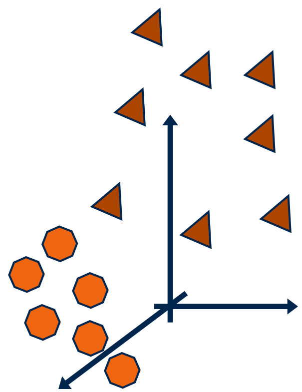
- Kernel matrix **K** defines all pairwise inner products
- Mercer theorem: **K** positive semidefinite
- Any symmetric positive semidefinite matrix can be regarded as an inner product matrix in some space

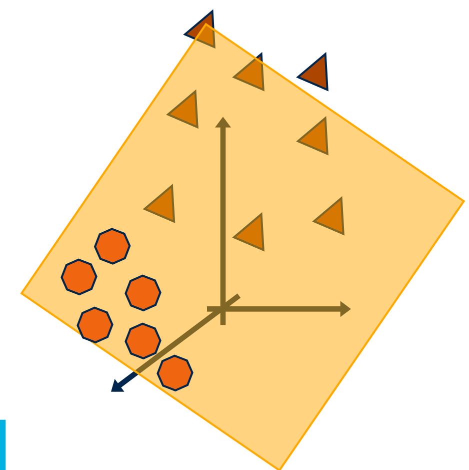
$$K_{ij}=k(x_i,x_j)$$

Kernel-Based Learning

Data | **Embedding** | **Linear algorithm**

$\{(x_i, y_i)\}$



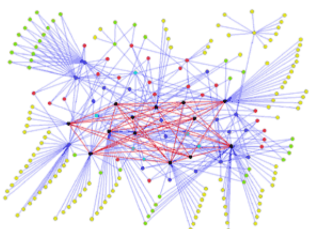


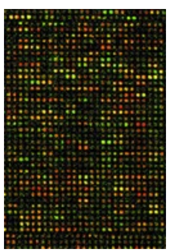
- **k(x,y) or**
- **y**

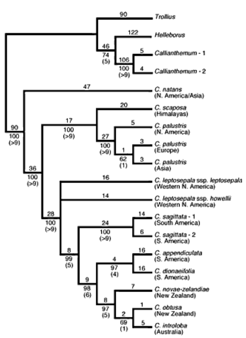
K

Kernel-Based Learning

Data





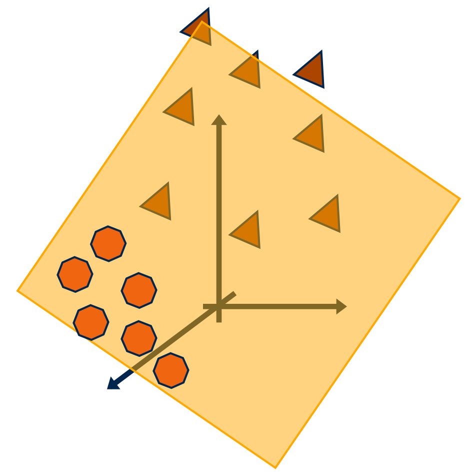


Embedding

K

Kernel design

Linear algorithm



Kernel algorithm

Methods

I) Instance-based methods:
   1) Nearest neighbor

II) Probabilistic models:
   1) Naïve Bayes
   2) Logistic Regression

III) Linear Models:
   1) Perceptron
   2) Support Vector Machine

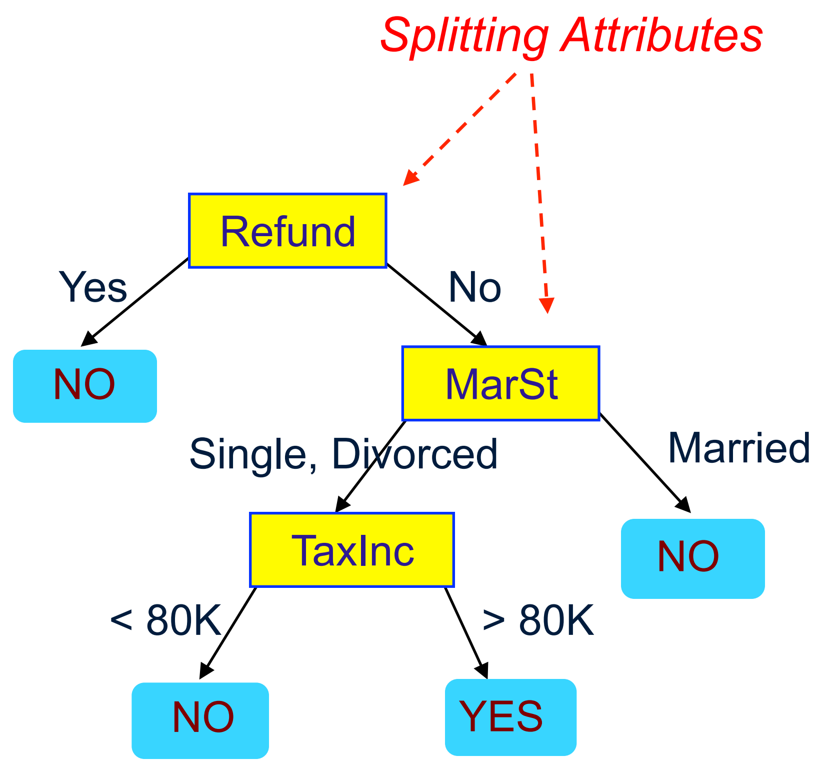
IV) $\boxed{\text{Decision Models:}}$
   1) Decision Trees
   2) Boosted Decision Trees
   3) Random Forest

Example of a Decision Tree

categorical categorical continuous class

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

Training Data



Model: Decision Tree

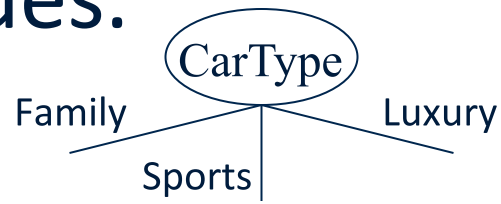
How to Specify Test Condition?

- Depends on attribute types
  - Nominal
  - Ordinal
  - Continuous

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

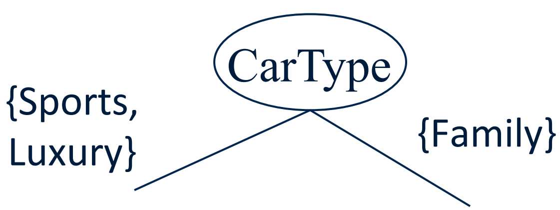
Splitting Based on Nominal Attributes

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

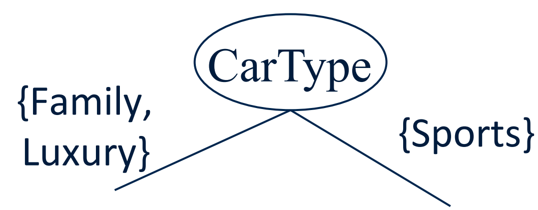


- Binary split: Divides values into two subsets.

Need to find optimal partitioning.

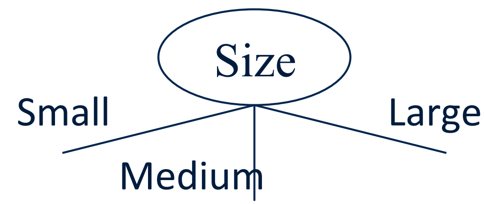


OR

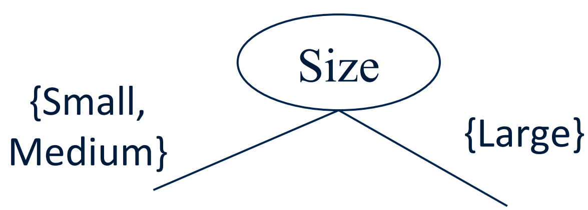


Splitting Based on Ordinal Attributes

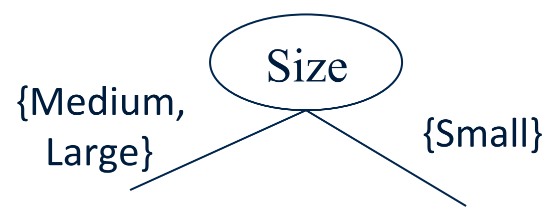
- Multi-way split: Use as many partitions as distinct values.



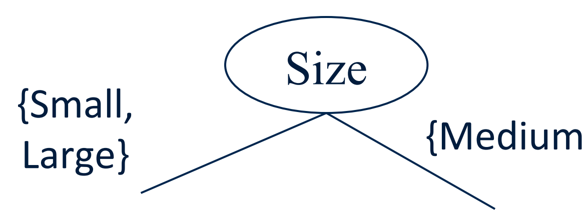
- Binary split: Divides values into two subsets.
  Need to find optimal partitioning.



OR



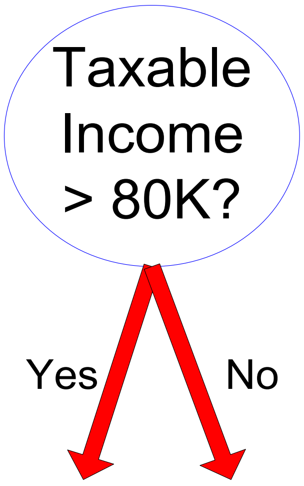
- What about this split?



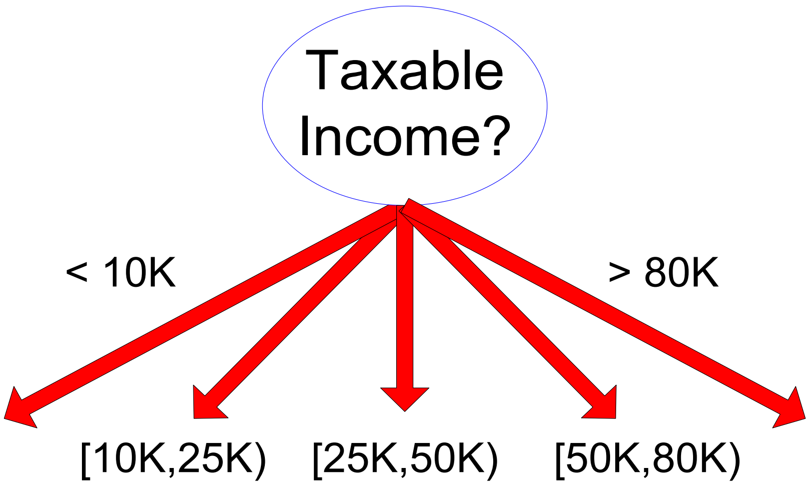
Splitting Based on Continuous Attributes

- Different ways of handling
  - Discretization to form an ordinal categorical attribute
    - Static – discretize once at the beginning
    - Dynamic – ranges can be found by equal interval bucketing, equal frequency bucketing (percentiles), or clustering.
  - Binary Decision: $(A < v)$ or $(A \geq v)$
    - consider all possible splits and finds the best cut
    - can be more compute intensive

Splitting Based on Continuous Attributes



(i) Binary split



(ii) Multi-way split

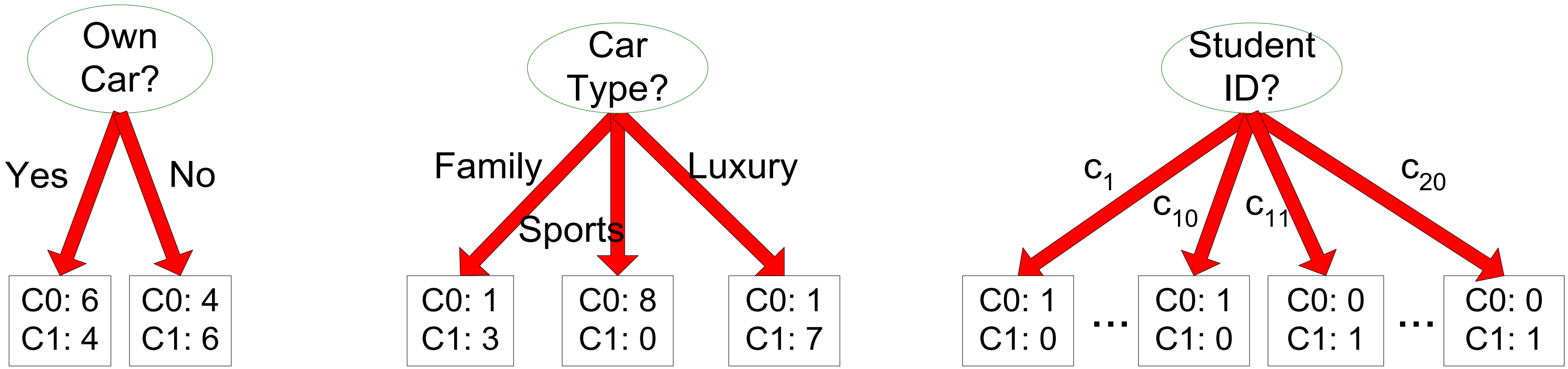
Tree Induction

- 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



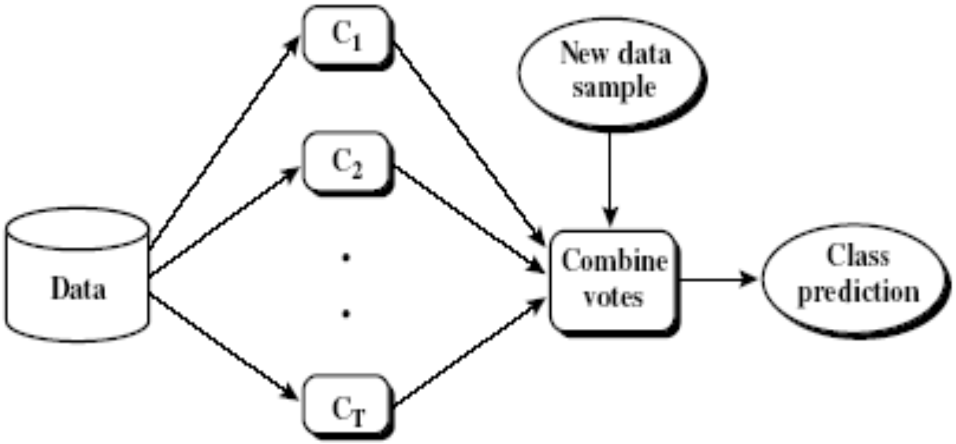
Which test condition is the best?

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

Ensemble Methods: Increasing the Accuracy



- Ensemble methods
  - Use a combination of models to increase accuracy
  - Combine a series of k learned models, M_1, M_2, …, M_k, with the aim of creating an improved model M*
- Popular ensemble methods
  - Bagging: averaging the prediction over a collection of classifiers
  - Boosting: weighted vote with a collection of classifiers
  - Ensemble: combining a set of heterogeneous classifiers

Bagging and Randomised Trees

other classifier combinations:

Bagging:

combine trees grown from “bootstrap” samples

(i.e re-sample training data with replacement)

Randomised Trees: **(Random Forest: trademark L.Breiman, A.Cutler)**

combine trees grown with:

random bootstrap (or subsets) of the training data only

consider at each node only a random subsets of variables for the split

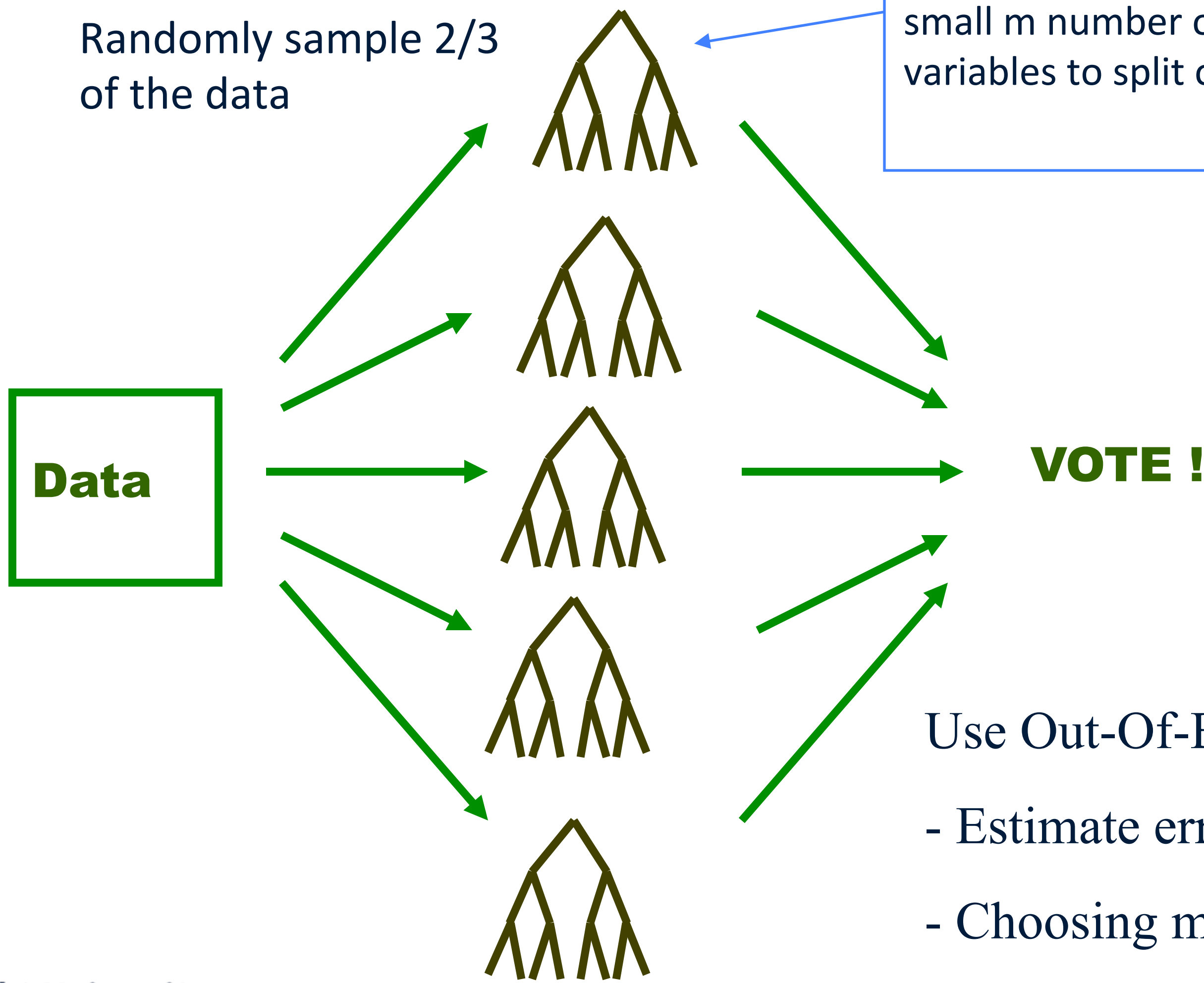
NO Pruning!

These combined classifiers work surprisingly well, are very stable and almost perfect “out of the box” classifiers

Random Forest

Randomly sample 2/3 of the data

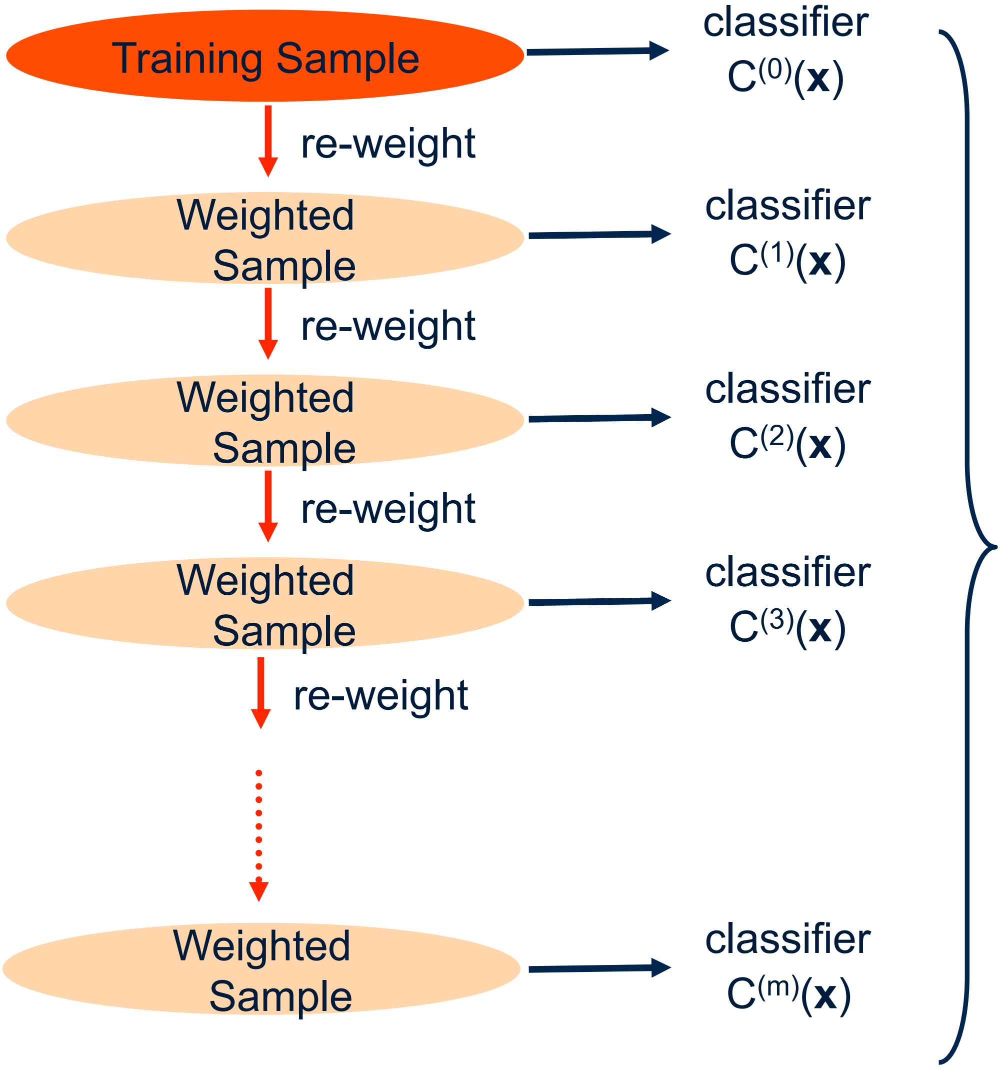
Each node: pick randomly a small m number of input variables to split on



Use Out-Of-Bag samples to:

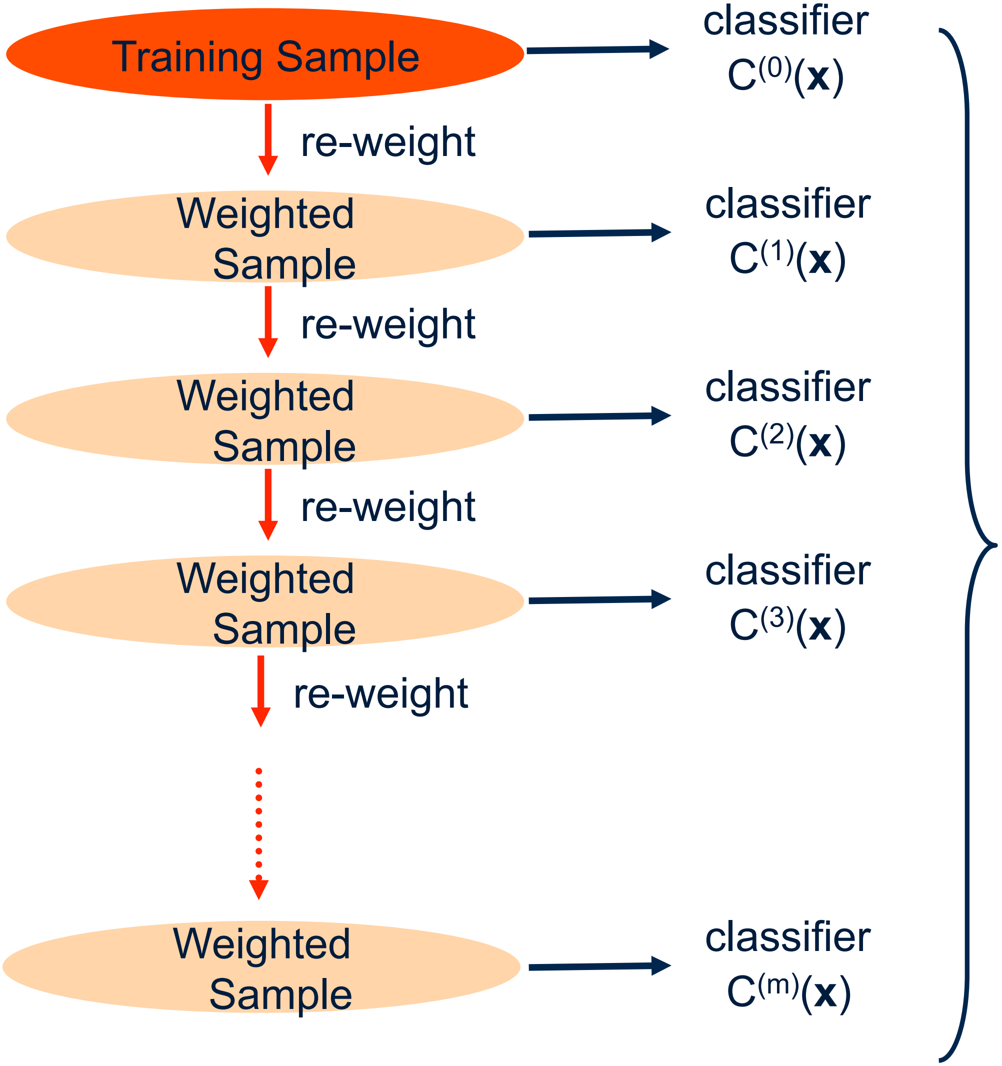
- Estimate error
- Choosing m
- Estimate variable importance

Boosting



$$y(x) = \sum_{i}^{N_{Classifier}} w_i C^{(i)}(x)$$

Adaptive Boosting (AdaBoost)



AdaBoost re-weights events misclassified by previous classifier by:

$$\frac{1-f_{err}}{f_{err}} \quad \text{with :}$$

$$f_{err} = \frac{\text{misclassified events}}{\text{all events}}$$

AdaBoost weights the classifiers also using the error rate of the individual classifier according to:

$$y(x) = \sum_{i}^{N_{Classifier}} \log\left(\frac{1-f_{err}^{(i)}}{f_{err}^{(i)}}\right) C^{(i)}(x)$$

Tricks and Evaluation

Let's tie classification to text

- Representations of text are usually very high dimensional
  - "The curse of dimensionality"
- High-bias algorithms should generally work best in high-dimensional space
  - They prevent overfitting
  - They generalize more
- For most text categorization tasks, there are many relevant features and many irrelevant ones

Which classifier do I use for a given (text) classification problem?

- Is there a learning method that is optimal for all (text) classification problems?
  - No, because there is a tradeoff between bias and variance.
- Factors to take into account:
  - How much training data is available?
  - How simple/complex is the problem? (linear vs. nonlinear decision boundary)
  - How noisy is the data?
  - How stable is the problem over time?
    - For an unstable problem, it's better to use a simple and robust classifier.

Manually written rules

- No training data, adequate editorial staff?
- Never forget the hand-written rules solution!
  - If (wheat or grain) and not (whole or bread) then
    - Categorize as grain
- In practice, rules get a lot bigger than this
  - Can also be phrased using tf or tf.idf weights
- With careful crafting (human tuning on development data) performance is high:
  - Construe: 94% recall, 84% precision over 675 categories (Hayes and Weinstein 1990)
- Amount of work required is huge
  - Estimate 2 days per class … plus maintenance

Very little data?

- If you're just doing supervised classification, you should stick to something high bias
  - There are theoretical results that Naïve Bayes should do well in such circumstances (Ng and Jordan 2002 NIPS)
- The interesting theoretical answer is to explore semi-supervised training methods:
  - Bootstrapping, EM over unlabeled documents, …
- The practical answer is to get more labeled data as soon as you can
  - How can you insert yourself into a process where humans will be willing to label data for you??

A reasonable amount of data?

- We can use all our clever classifiers
- "Roll out the SVM!"

- But if you are using an SVM/NB etc., you should probably be prepared with the "hybrid" solution where there is a Boolean overlay
  - Or else to use user-interpretable Boolean-like models like decision trees
  - Users like to hack, and management likes to be able to implement quick fixes immediately

A huge amount of data?

- This is great in theory for doing accurate classification…
- But it could easily mean that expensive methods like SVMs (train time) or kNN (test time) are quite impractical
- Naïve Bayes can come back into its own again!
  - Or other advanced methods with linear training/test complexity like regularized logistic regression (though much more expensive to train)

How many categories?

- A few (well separated ones)?
  - Easy!
- A zillion closely related ones?
  - Think: Yahoo! Directory, Library of Congress classification, legal applications
  - Quickly gets difficult!
    - Classifier combination is always a useful technique
      - Voting, bagging, or boosting multiple classifiers
    - Much literature on hierarchical classification
      - Mileage fairly unclear, but helps a bit (Tie-Yan Liu et al. 2005)
      - Definitely helps for scalability, even if not in accuracy
    - May need a hybrid automatic/manual solution

Model Evaluation and Selection

- Evaluation metrics: How can we measure accuracy? Other metrics to consider?
- Use **validation test set** of class-labeled tuples instead of training set when assessing accuracy
- Methods for estimating a classifier's accuracy:
  - Holdout method, random subsampling
  - Cross-validation
  - Bootstrap
- Comparing classifiers:
  - Confidence intervals
  - Cost-benefit analysis and ROC Curves

Classifier Evaluation Metrics: Confusion Matrix for 2 classes

Confusion Matrix:

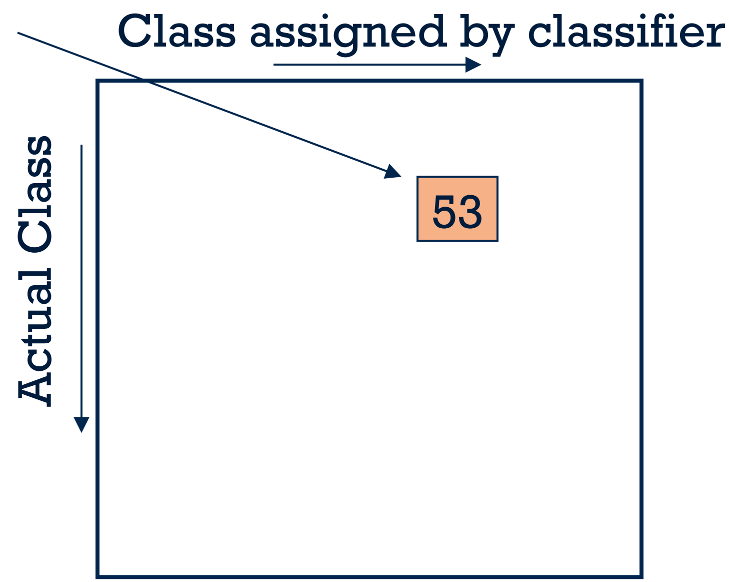
Actual class\Predicted class	C_1	$\neg C_1$
C_1	**True Positives (TP)**	**False Negatives (FN)**
$\neg C_1$	**False Positives (FP)**	**True Negatives (TN)**

Example of Confusion Matrix:

Actual class\Predicted class	buy_computer = yes	buy_computer = no	Total
buy_computer = yes	**6954**	**46**	7000
buy_computer = no	**412**	**2588**	3000
Total	7366	2634	10000

Classifier Evaluation Metrics: Confusion Matrix for more classes

This (*i*, *j*) entry means 53 of the docs actually in class *i* were put in class *j* by the classifier.



- In a perfect classification, only the diagonal has non-zero entries
- Look at common confusions and how they might be addressed

Classifier Evaluation Metrics: Accuracy, Error Rate, Sensitivity and Specificity

A\P	C	¬C	
C	**TP**	**FN**	**P**
¬C	**FP**	**TN**	**N**
	P’	**N’**	**All**

- **Classifier Accuracy,** or recognition rate: percentage of test set tuples that are correctly classified
  Accuracy = (TP + TN)/All
- **Error rate:** *1 – accuracy*, or
  Error rate = (FP + FN)/All

- **Class Imbalance Problem**:
  - One class may be *rare*, e.g. fraud, or HIV-positive
  - Significant *majority of the negative class* and minority of the positive class
  - **Sensitivity**: True Positive recognition rate
    - **Sensitivity = TP/P**
  - **Specificity**: True Negative recognition rate
    - **Specificity = TN/N**

Classifier Evaluation Metrics: Precision and Recall, and F-measures

- **Precision**: exactness – what % of tuples that the classifier labeled as positive are actually positive

$$precision = \frac{TP}{TP + FP}$$

- **Recall:** completeness – what % of positive tuples did the classifier label as positive?

$$recall = \frac{TP}{TP + FN}$$

- Perfect score is 1.0
- Inverse relationship between precision & recall
- ***F* measure (*F*$_1$ or *F*-score)**: harmonic mean of precision and recall,
- ***F*$_\beta$**: weighted measure of precision and recall
  - assigns ß times as much weight to recall as to precision

$$F = \frac{2 \times precision \times recall}{precision + recall}$$

$$F_\beta = \frac{(1 + \beta^2) \times precision \times recall}{\beta^2 \times precision + recall}$$

Classifier Evaluation Metrics: Example

Actual Class\Predicted class	cancer = yes	cancer = no	Total	Recognition(%)
cancer = yes	**90**	**210**	300	30.00 (*sensitivity*
cancer = no	**140**	**9560**	9700	98.56 (*specificity)*
Total	230	9770	10000	96.40 (*accuracy*)

- *Precision* = 90/230 = 39.13% *Recall* = 90/300 = 30.00%

Micro- vs. Macro-Averaging

- If we have more than one class, how do we combine multiple performance measures into one quantity?
- Macroaveraging: Compute performance for each class, then average.
- Microaveraging: Collect decisions for all classes, compute contingency table, evaluate.

Micro- vs. Macro-Averaging: Example

Class 1

	Truth: yes	Truth: no
Classifier: yes	10	10
Classifier: no	10	970

Class 2

	Truth: yes	Truth: no
Classifier: yes	90	10
Classifier: no	10	890

Micro Ave. Table

	Truth: yes	Truth: no
Classifier: yes	100	20
Classifier: no	20	1860

- Macroaveraged precision: $(0.5 + 0.9)/2 = 0.7$
- Microaveraged precision: $100/120 = .83$

- Microaveraged score is dominated by score on common classes

Evaluating Classifier Accuracy: Holdout & Cross-Validation Methods

- **Holdout method**
  - Given data is randomly partitioned into two independent sets
    - Training set (e.g., 2/3) for model construction
    - Test set (e.g., 1/3) for accuracy estimation
  - Random sampling: a variation of holdout
    - Repeat holdout k times, accuracy = avg. of the accuracies obtained
- **Cross-validation** (*k*-fold, where k = 10 is most popular)
  - Randomly partition the data into *k mutually exclusive* subsets, each approximately equal size
  - At *i*-th iteration, use D_i as test set and others as training set
  - Leave-one-out: *k* folds where *k* = # of tuples, for small sized data
  - ***Stratified cross-validation***: folds are stratified so that class dist. in each fold is approx. the same as that in the initial data

Estimating Confidence Intervals: Classifier Models M_1 vs. M_2

- Suppose we have 2 classifiers, M_1 and M_2, which one is better?
- Use 10-fold cross-validation to obtain $\overline{err}(M_1)$ and $\overline{err}(M_2)$
- These mean error rates are just *estimates* of error on the true population of *future* data cases
- What if the difference between the 2 error rates is just attributed to *chance*?
  - Use a **test of statistical significance**
  - Obtain **confidence limits** for our error estimates

Estimating Confidence Intervals: Null Hypothesis

- Perform 10-fold cross-validation
- Assume samples follow a **t distribution** with $k\!-\!1$ **degrees of freedom** (here, $k=10$)
- Use **t-test** (or **Student's t-test**)
- **Null Hypothesis**: M_1 & M_2 are the same
- If we can **reject** null hypothesis, then
  - we conclude that the difference between M_1 & M_2 is **statistically significant**
  - Chose model with lower error rate

Estimating Confidence Intervals: t-test

- If only 1 test set available: **pairwise comparison**
  - For i[th] round of 10-fold cross-validation, the same cross partitioning is used to obtain *err(M₁)ᵢ* and *err(M₂)ᵢ*
  - Average over 10 rounds to get $\overline{err}(M_1)\overline{err}(M_2)$
  - **t-test** computes **t-statistic** with *k-1* **degrees of freedom:**

$$t = \frac{\overline{err}(M_1) - \overline{err}(M_2)}{\sqrt{var(M_1 - M_2)/k}}$$

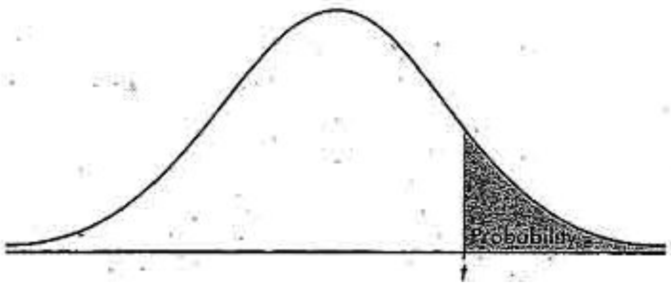
$$var(M_1 - M_2) = \sqrt{\frac{var(M_1)}{k_1} + \frac{var(M_2)}{k_2}},$$

- If two test sets available: use **non-paired t-test**

$$var(M_1 - M_2) = \frac{1}{k}\sum_{i=1}^{k}\left[err(M_1)_i - err(M_2)_i - (\overline{err}(M_1) - \overline{err}(M_2))\right]^2$$

where k_1 & k_2 are # of cross-validation samples used for M_1 & M_2, resp.

Estimating Confidence Intervals: Table for t-distribution



- Symmetric
- **Significance level**, e.g., *sig = 0.05* or *5%* means M_1 & M_2 are *significantly different* for 95% of population
- **Confidence limit**, *z = sig/2*

TABLE B: t-DISTRIBUTION CRITICAL VALUES

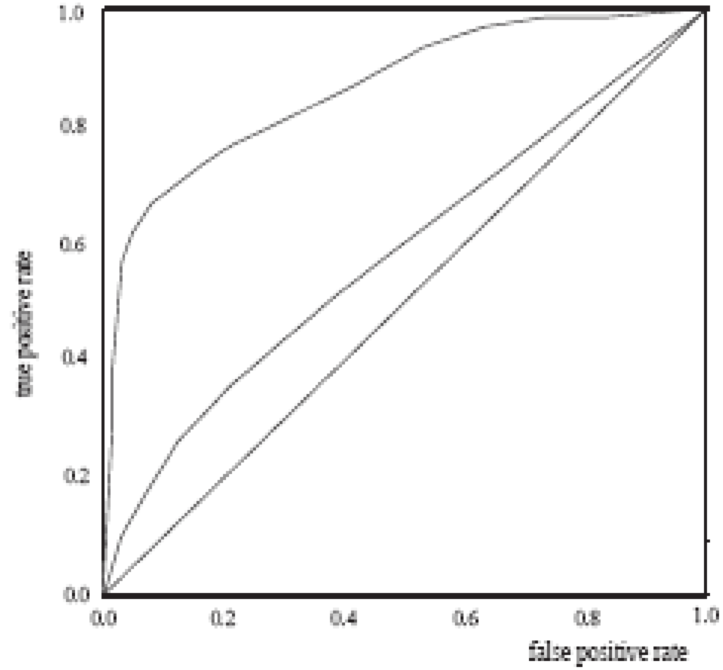
	Tail probability p											
df	.25	.20	.15	.10	.05	.025	.02	.01	.005	.0025	.001	.0005
1	1.000	1.376	1.963	3.078	6.314	12.71	15.89	31.82	63.66	127.3	318.3	636.6
2	.816	1.061	1.386	1.886	2.920	4.303	4.849	6.965	9.925	14.09	22.33	31.60
3	.765	.978	1.250	1.638	2.353	3.182	3.482	4.541	5.841	7.453	10.21	12.92
4	.741	.941	1.190	1.533	2.132	2.776	2.999	3.747	4.604	5.598	7.173	8.610
5	.727	.920	1.156	1.476	2.015	2.571	2.757	3.365	4.032	4.773	5.893	6.869
6	.718	.906	1.134	1.440	1.943	2.447	2.612	3.143	3.707	4.317	5.208	5.959
7	.711	.896	1.119	1.415	1.895	2.365	2.517	2.998	3.499	4.029	4.785	5.408
8	.706	.889	1.108	1.397	1.860	2.306	2.449	2.896	3.355	3.833	4.501	5.041
9	.703	.883	1.100	1.383	1.833	2.262	2.398	2.821	3.250	3.690	4.297	4.781
10	.700	.879	1.093	1.372	1.812	2.228	2.359	2.764	3.169	3.581	4.144	4.587
11	.697	.876	1.088	1.363	1.796	2.201	2.328	2.718	3.106	3.497	4.025	4.437
12	.695	.873	1.083	1.356	1.782	2.179	2.303	2.681	3.055	3.428	3.930	4.318
13	.694	.870	1.079	1.350	1.771	2.160	2.282	2.650	3.012	3.372	3.852	4.221
14	.692	.868	1.076	1.345	1.761	2.145	2.264	2.624	2.977	3.326	3.787	4.140
15	.691	.866	1.074	1.341	1.753	2.131	2.249	2.602	2.947	3.286	3.733	4.073
16	.690	.865	1.071	1.337	1.746	2.120	2.235	2.583	2.921	3.252	3.686	4.015
17	.689	.863	1.069	1.333	1.740	2.110	2.224	2.567	2.898	3.222	3.646	3.965
18	.688	.862	1.067	1.330	1.734	2.101	2.214	2.552	2.878	3.197	3.611	3.922
19	.688	.861	1.066	1.328	1.729	2.093	2.205	2.539	2.861	3.174	3.579	3.883
20	.687	.860	1.064	1.325	1.725	2.086	2.197	2.528	2.845	3.153	3.552	3.850
21	.686	.859	1.063	1.323	1.721	2.080	2.189	2.518	2.831	3.135	3.527	3.819
22	.686	.858	1.061	1.321	1.717	2.074	2.183	2.508	2.819	3.119	3.505	3.792
23	.685	.858	1.060	1.319	1.714	2.069	2.177	2.500	2.807	3.104	3.485	3.768
24	.685	.857	1.059	1.318	1.711	2.064	2.172	2.492	2.797	3.091	3.467	3.745
25	.684	.856	1.058	1.316	1.708	2.060	2.167	2.485	2.787	3.078	3.450	3.725
26	.684	.856	1.058	1.315	1.706	2.056	2.162	2.479	2.779	3.067	3.435	3.707
27	.684	.855	1.057	1.314	1.703	2.052	2.158	2.473	2.771	3.057	3.421	3.690
28	.683	.855	1.056	1.313	1.701	2.048	2.154	2.467	2.763	3.047	3.408	3.674
29	.683	.854	1.055	1.311	1.699	2.045	2.150	2.462	2.756	3.038	3.396	3.659
30	.683	.854	1.055	1.310	1.697	2.042	2.147	2.457	2.750	3.030	3.385	3.646
40	.681	.851	1.050	1.303	1.684	2.021	2.123	2.423	2.704	2.971	3.307	3.551
50	.679	.849	1.047	1.299	1.676	2.009	2.109	2.403	2.678	2.937	3.261	3.496
60	.679	.848	1.045	1.296	1.671	2.000	2.099	2.390	2.660	2.915	3.232	3.460
80	.678	.846	1.043	1.292	1.664	1.990	2.088	2.374	2.639	2.887	3.195	3.416
100	.677	.845	1.042	1.290	1.660	1.984	2.081	2.364	2.626	2.871	3.174	3.390
1000	.675	.842	1.037	1.282	1.646	1.962	2.056	2.330	2.581	2.813	3.098	3.300
∞	.674	.841	1.036	1.282	1.645	1.960	2.054	2.326	2.576	2.807	3.091	3.291
	50%	60%	70%	80%	90%	95%	96%	98%	99%	99.5%	99.8%	99.9%
	Confidence level C											

Estimating Confidence Intervals: Statistical Significance

- Are M_1 & M_2 **significantly different**?
  - Compute *t*. Select *significance level* (e.g. *sig = 5%)*
  - Consult table for t-distribution: Find *t value* corresponding to *k-1 degrees of freedom* (here, 9)
  - t-distribution is symmetric: typically upper % points of distribution shown → look up value for **confidence limit** *z=sig/2* (here, 0.025)
  - **If t > z or t < -z**, then t value lies in rejection region:
    - **Reject null hypothesis** that mean error rates of M_1 & M_2 are same
    - Conclude: statistically significant difference between M_1 & M_2
  - **Otherwise**, conclude that any difference is **chance**

Model Selection: ROC Curves

- **ROC** (Receiver Operating Characteristics) curves: for visual comparison of classification models
- Originated from signal detection theory
- Shows the trade-off between the true positive rate and the false positive rate
- The area under the ROC curve is a measure of the accuracy of the model
- Rank the test tuples in decreasing order: the one that is most likely to belong to the positive class appears at the top of the list
- The closer to the diagonal line (i.e., the closer the area is to 0.5), the less accurate is the model



- Vertical axis represents the true positive rate
- Horizontal axis rep. the false positive rate
- The plot also shows a diagonal line
- A model with perfect accuracy will have an area of 1.0


KEEP
CALM
YOU
SURVIVED
Day 1