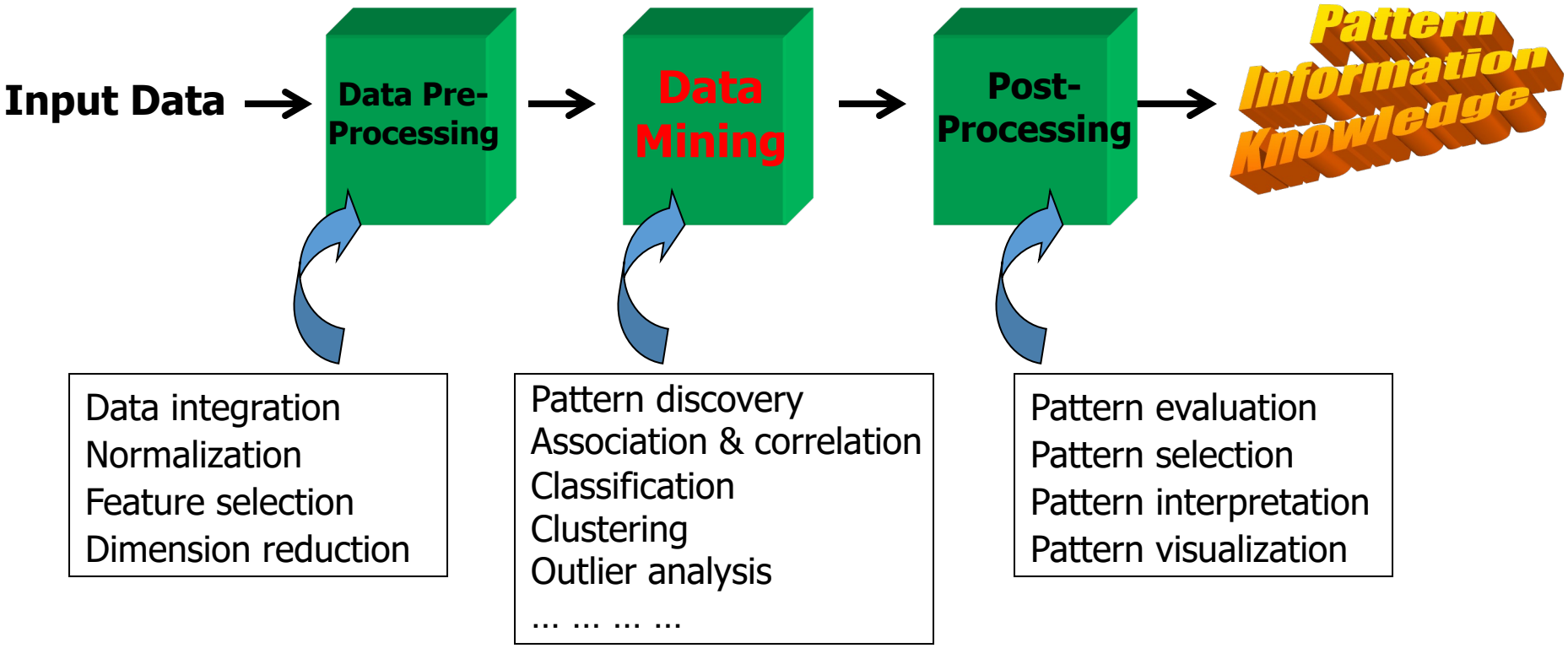
COSC 3337
Data Science I
Section 14623

Data Preprocessing

Instructor: Jingchao Ni
Fall 2024

KDD Process: A Typical View from ML and Statistics

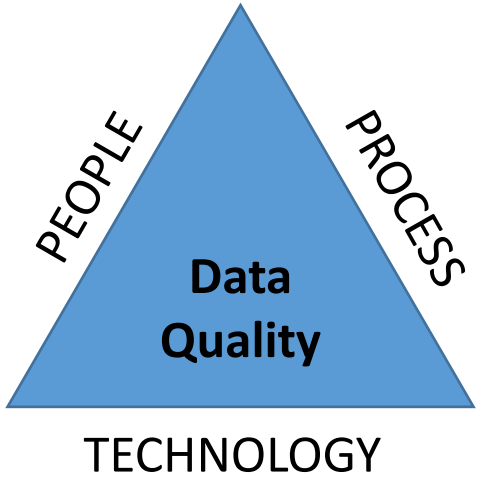


- This is a view from typical machine learning and statistics communities

Outlines

- Data quality issues
- Data cleaning
- Data integration
- Data transformation
- Data reduction

Real world data is dirty



- Measurement errors
- Process failures
- Interface errors
- Attacks
- …

- Incomplete
  - Missing attribute values
  - Missing interesting attributes
  - Information loss after aggregation
- Noisy
  - Errors or outliers
- Inconsistent
  - Discrepancies in codes or names

Incomplete Data

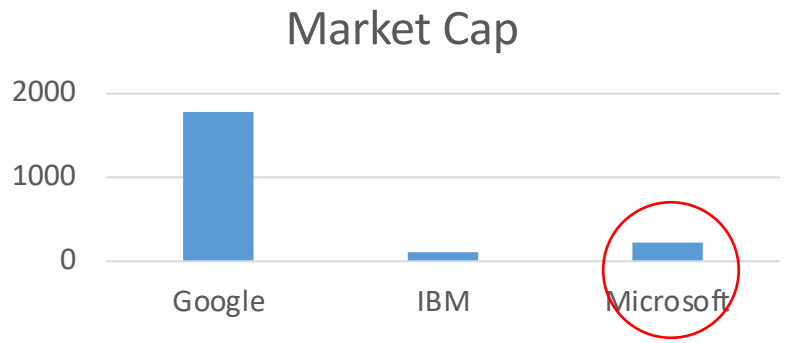
- Missing attribute values
  - Market Cap of Intel
- Missing interesting attributes
  - How many employees does Microsoft have?
- Information loss after aggregation
  - What is the price per stock of Google?
  - Market Cap: multiplying the price of a stock by its total number of outstanding shares

Company_Name	Address	Market Cap
Google	Googleplex, Mtn. View, CA	$1.78T
Intl. Business Machines	Armonk, NY	$111.68B
Microsoft	Redmond, WA	$2.23B
Intel	Mtn. View, CA	NULL

Noisy Data

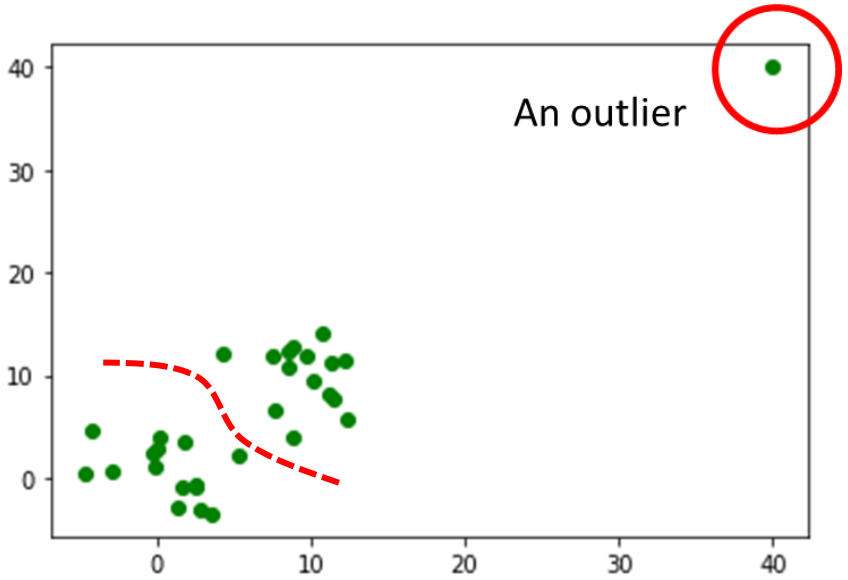
- Errors or outliers

Company_Name	Address	Market Cap
Google	Googleplex, Mtn. View, CA	$1.78T
Intl. Business Machines	Armonk, NY	$111.68B
Microsoft	Redmond, WA	$2.23B -> $3.16T
Intel	Mtn. View, CA	NULL



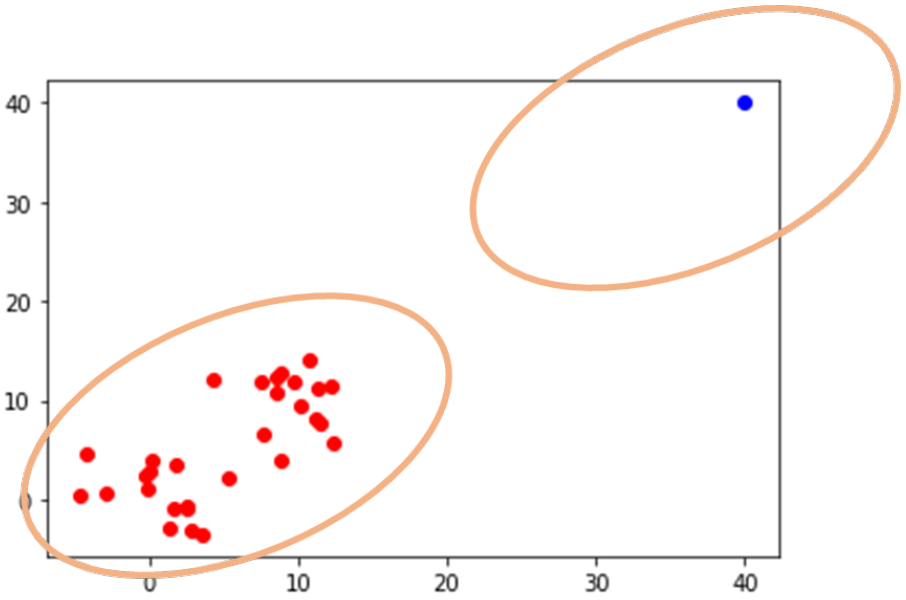
Noisy Data

- Errors or outliers



Dataset with an outlier

K-Means



Results of Kmeans

Inconsistent Queries

Company_Name	Address	Market Cap
Google	Googleplex, Mtn. View, CA	$1.78T
Intl. Business Machines	Armonk, NY	$111.68B
Microsoft	Redmond, WA	$2.23B
Intel	Mtn. View, CA	NULL

```
SELECT Market_Cap
From Companies
Where Company_Name = “IBM”
```

```
Number of Rows: 0
```

Problem:
Entity Resolution
“IBM” and “Intl. Business Machines”

Motivation

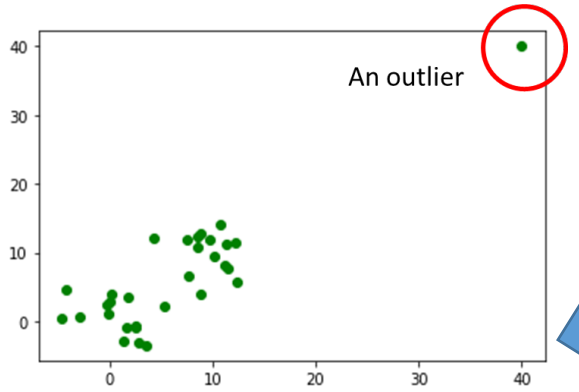
- Data in the real world is dirty
  - **incomplete**: missing attribute values, missing interesting attributes
  - **noisy**: errors or outliers
  - **inconsistent**: discrepancies or contradictions

- Quality of data mining results largely depends on the quality of the input data
  - Quality decisions are based on quality data
  - Data warehouse needs consistent integration of quality data

Garbage in, garbage out!

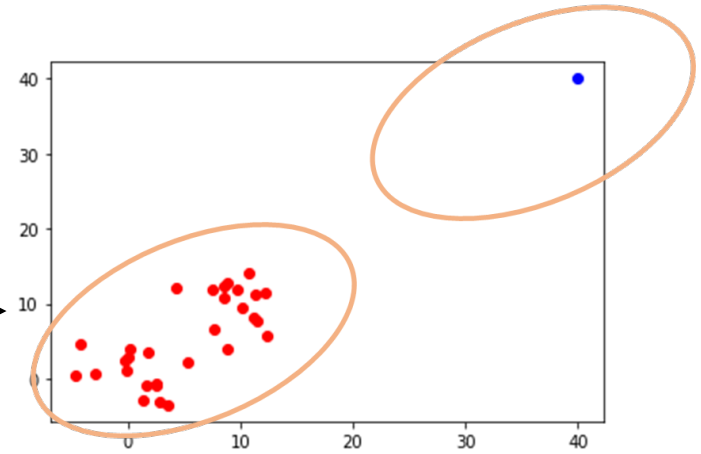
Data Mining Perspective

- Clustering after removing outliers



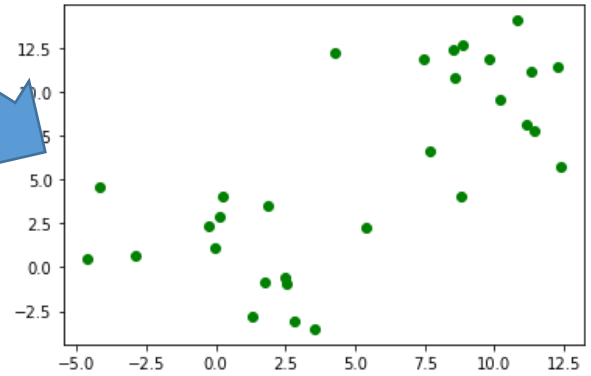
Dataset with an outlier

K-Means

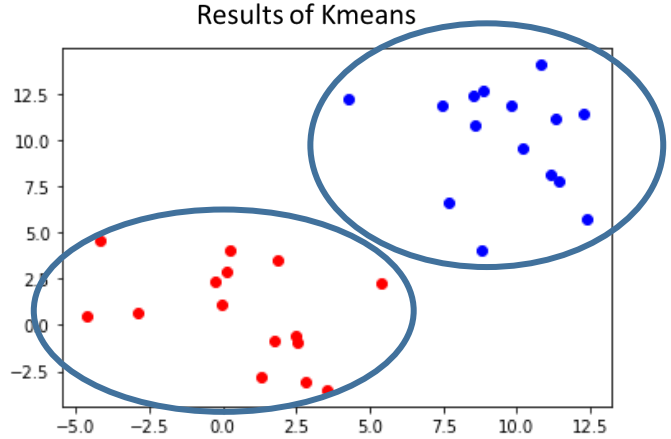


Results of Kmeans

Data Cleaning



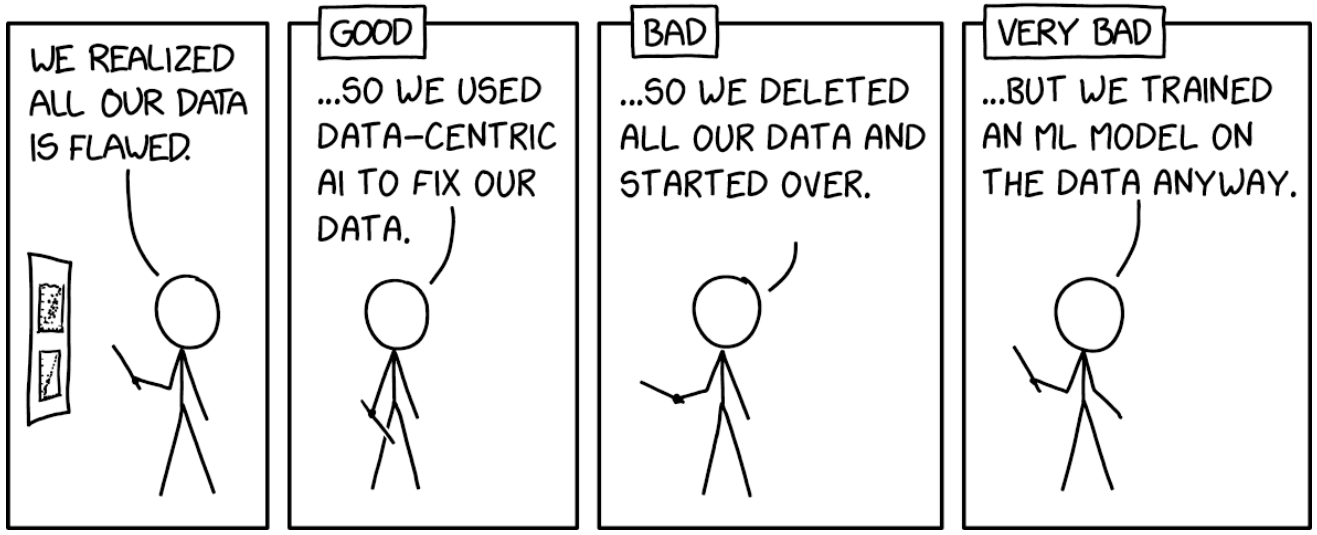
Remove the outlier

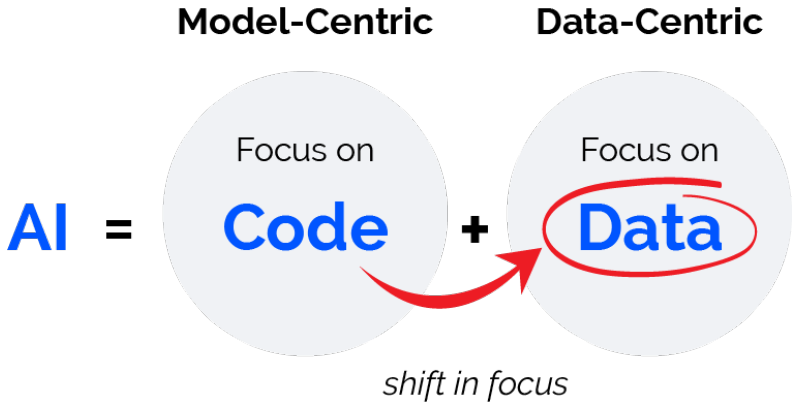


K-means

The Emergent Data Centric AI

- Data-Centric AI (DCAI)
  - An emerging science that studies techniques to improve datasets, which is often the best way to improve performance in practical ML applications
  - **Model-centric AI** is to produce the best model for a given dataset
  - **Data-centric AI** is to systematically & algorithmically produce the best dataset to feed a given ML model.





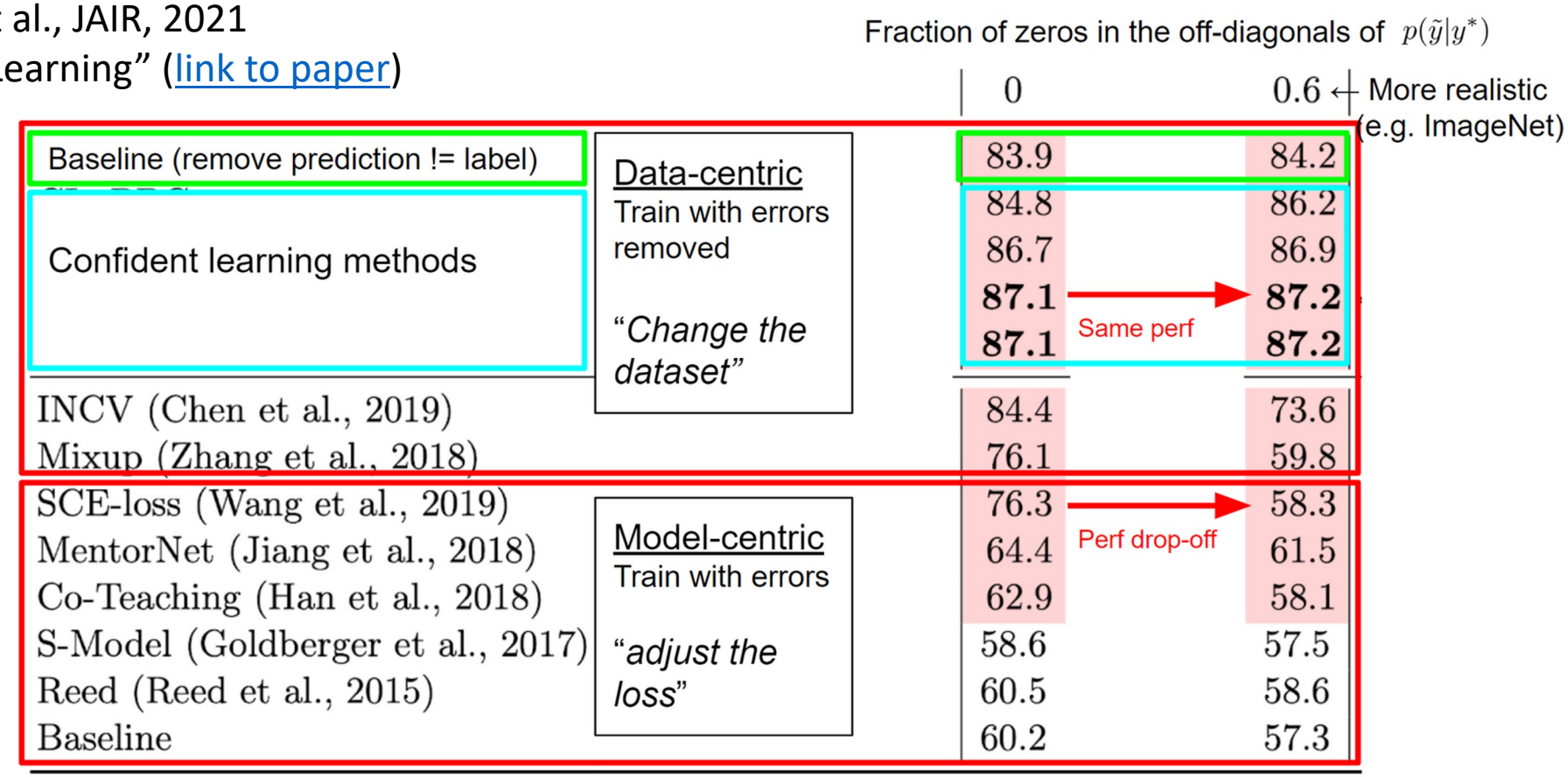
Source: https://landing.ai/data-centric-ai

Source: https://dcai.csail.mit.edu/

The Emergent Data Centric AI

Compare Accuracy: Learning with 40% label noise in CIFAR-10

Northcutt et al., JAIR, 2021
"Confident Learning" (link to paper)



Source: https://dcai.csail.mit.edu/

Types for Data Preprocessing

- Data cleaning
  - Fill in missing values, smooth noisy data, identify or remove outliers, resolve inconsistencies
- Data integration
  - Integration of multiple databases, data cubes, or files
- Data transformation
  - Normalization, aggregation
- Data reduction
  - Reduce number of records, attributes or attribute values

Outlines

- Data quality issues
- Data cleaning
- Data integration
- Data transformation
- Data reduction

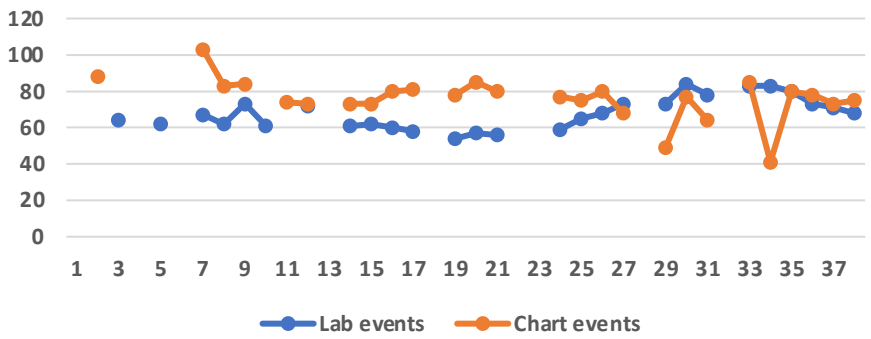
Data Cleaning – Missing Data

- Data is not always complete/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
  - data were not considered important at the time of collection
  - data format/contents of database changes over time

Data Cleaning – Missing Data

- Data is not always complete/available
  - E.g., many tuples have no recorded value for several attributes, such as customer income in sales data

```
Hours,Capillary refill rate,Diastolic blood pressure,Fraction inspired oxygen,Glascow coma scale eye opening,Glascow coma scale motor response,Glascow coma scale total,Glascow coma scale verbal response,G
lucose,Heart Rate,Height,Mean blood pressure,Oxygen saturation,Respiratory rate,Systolic blood pressure,Temperature,Weight,pH
-4.416666666666667,,,,,,,208.0,,,,,,,,
-3.4166666666666665,,,,,,,,,,,,,,,,5.0
-1.25,,,,,,,,,,,,,,,,7.34
0.36666666666666664,,59.0,,4 Spontaneously,6 Obeys Commands,15,5 Oriented,,117,,83.333297729492188,90.0,36,132.0,37.444445292154946,97.0999984741211,
1.6166666666666667,,47.0,,4 Spontaneously,6 Obeys Commands,15,5 Oriented,,108,,69.666702270507812,93.0,36,115.0,37.555599212646484,,
3.6166666666666667,,57.0,,,,,,,106,,78.666702270507812,92.0,32,122.0,,,
5.616666666666666,,55.0,,4 Spontaneously,6 Obeys Commands,15,5 Oriented,,108,,76.333297729492188,92.0,32,119.0,37.38890075683594,,
7.616666666666666,,54.0,,,,,,,105,,73.333297729492188,92.0,28,112.0,,,
8.783333333333333,,,,,,,,198.0,,,,,,,,
9.616666666666667,,55.0,,4 Spontaneously,6 Obeys Commands,15,5 Oriented,,106,,72.666702270507812,92.0,29,108.0,36.38890075683594,,
9.733333333333333,,,,,,,,,,,,,,,,7.3
10.616666666666667,,59.0,,,,,,,109,,78.666702270507812,92.0,27,118.0,,,
11.616666666666667,,58.0,,,,,,,109,,75,93.0,36,109.0,,,
```



Electronic health records of ICU patients (MIMIC-III)

Data Cleaning – Missing Data

- Data is not always complete/available
  - E.g., many tuples have no recorded value for several attributes, such as customer income in sales data
- Missing data may be caused by
  - Equipment malfunction
  - Inconsistent with other recorded data and thus deleted
  - Data not entered due to misunderstanding
  - Data were not considered important at the time of collection
  - Data format/contents of database changes over time

Data Cleaning – Handling Missing Data

- Ignore the data objects

- Fill in the missing values
  - Manually: tedious + infeasible?
  - Use a global constant to fill in the missing value: e.g., 0
  - Use the attribute mean to fill in the missing value
  - Use the attribute mean for all samples of the same class to fill in the missing value: smarter
  - Use the most probable value to fill in the missing value: inference-based methods such as regression and the decision tree

Data Cleaning – Handling Missing Data

- House information data
  - # of bedrooms
  - Size of the house
  - Price
  - Type

	#bedrooms	size	price	type
0	1	499	275	?
1	1	?	265	1
2	1	645	77	1
3	1	996	264	2
4	2	1068	273	1
5	2	1441	392	2
6	2	?	253	1
7	2	1749	216	1
8	2	307	191	1
9	2	1035	?	?
10	2	1858	338	1

Data Cleaning – Handling Missing Data

- Ignore the data objects
  - For example, in a regression or classification problem, if the label is missing, we just remove the data object

	#bedrooms	size	price	type
0	1	499	275	?
1	1	?	265	1
2	1	645	77	1
3	1	996	264	2
4	2	1068	273	1
5	2	1441	392	2
6	2	?	253	1
7	2	1749	216	1
8	2	307	191	1
9	2	1035	?	?
10	2	1858	338	1

Ignore null-type data

	#bedrooms	size	price	type
1	1	?	265	1
2	1	645	77	1
3	1	996	264	2
4	2	1068	273	1
5	2	1441	392	2
6	2	?	253	1
7	2	1749	216	1
8	2	307	191	1
10	2	1858	338	1

Data Cleaning – Handling Missing Data

- Use a global constant to fill in the missing value: e.g., 0

	#bedrooms	size	price	type
1	1	?	265	1
2	1	645	77	1
3	1	996	264	2
4	2	1068	273	1
5	2	1441	392	2
6	2	?	253	1
7	2	1749	216	1
8	2	307	191	1
10	2	1858	338	1

Fill in with a constant

	#bedrooms	size	price	type
1	1	0	265	1
2	1	645	77	1
3	1	996	264	2
4	2	1068	273	1
5	2	1441	392	2
6	2	0	253	1
7	2	1749	216	1
8	2	307	191	1
10	2	1858	338	1

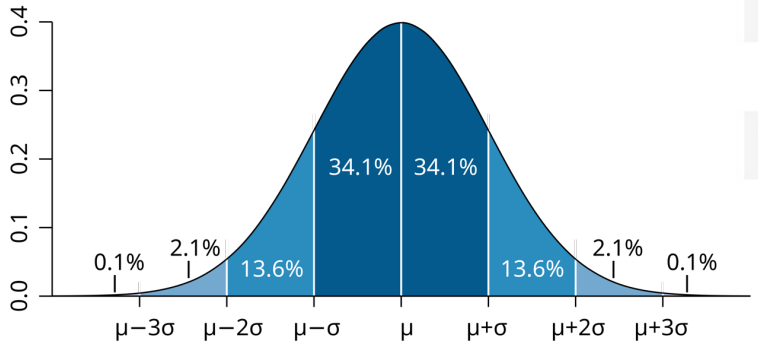
Data Cleaning – Handling Missing Data

- Use the attribute mean to fill in the missing entries

	#bedrooms	size	price	type
1	1	?	265	1
2	1	645	77	1
3	1	996	264	2
4	2	1068	273	1
5	2	1441	392	2
6	2	?	253	1
7	2	1749	216	1
8	2	307	191	1
10	2	1858	338	1

The average size is 1327

Fill in with mean



	#bedrooms	size	price	type
1	1	1327	265	1
2	1	645	77	1
3	1	996	264	2
4	2	1068	273	1
5	2	1441	392	2
6	2	1327	253	1
7	2	1749	216	1
8	2	307	191	1
10	2	1858	338	1

Data Cleaning – Handling Missing Data

- Use the attribute mean for all samples of the same class to fill in the missing value: smarter
  - The average price of apartments is different from the one of studios

	#bedrooms	size	price	type
1	1	?	265	1
2	1	645	77	1
3	1	996	264	2
4	2	1068	273	1
5	2	1441	392	2
6	2	?	253	1
7	2	1749	216	1
8	2	307	191	1
10	2	1858	338	1

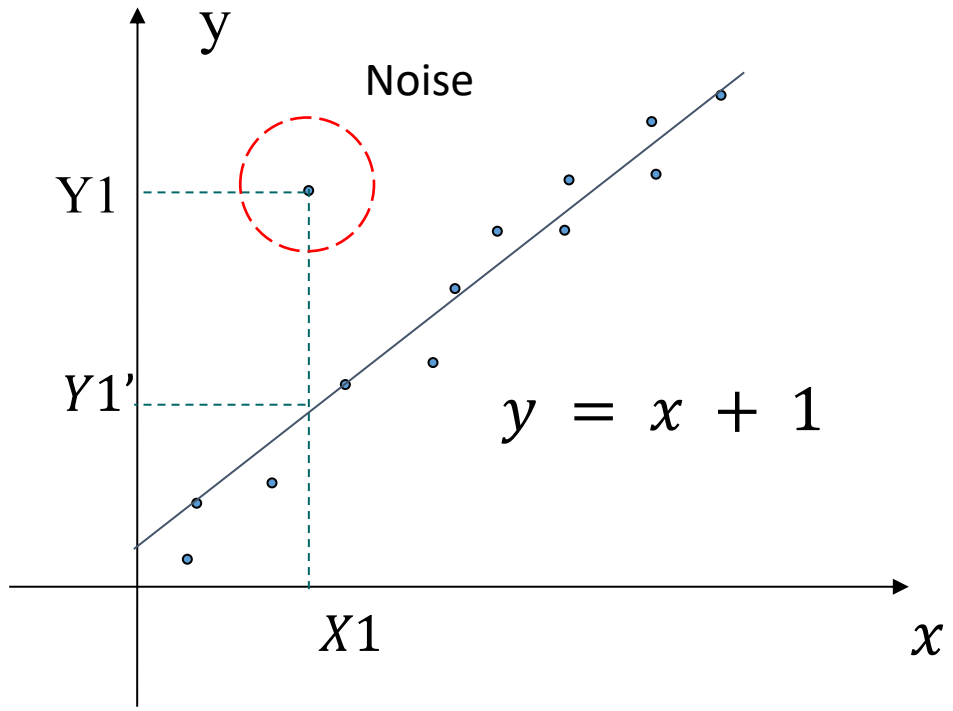
The average of type 1 size is 1329

Fill in with mean

	#bedrooms	size	price	type
1	1	1329	265	1
2	1	645	77	1
3	1	996	264	2
4	2	1068	273	1
5	2	1441	392	2
6	2	1329	253	1
7	2	1749	216	1
8	2	307	191	1
10	2	1858	338	1

Data Cleaning – Handling Missing Data

- Use the most probable value to fill in the missing value: inference-based such as regression
  - Replace noisy or missing values by predicted values
  - Requires model of attribute dependencies (maybe wrong!)
  - Can be used for data smoothing or for handling missing data



Data Cleaning - Regression

- The price of a house is highly related to the number of its bedrooms and its size
  1. Build a predictive model $y = f_\omega(x_1, x_2)$
  2. Linear regression $y = \omega_1 x_1 + \omega_2 x_2 + \omega_0$
     - y is the price
     - x_1 is the # of bedrooms
     - x_2 is the size
     - $\omega_1, \omega_2, \omega_0$ are parameters (coefficients and bias)
  3. Train with complete records in the dataset
  4. Predict the missing values

Data Cleaning – Noisy Data

- Noise: random error or variance in a measured attribute

- Noisy attribute values may be caused by:
  - Faulty data collection instruments
  - Data entry problems
  - Data transmission problems
  - Technology limitation
  - Inconsistency in naming convention
  - …

Data Cleaning – Handling Noisy Data

- Binning
  - Sort data and partition into (equal-depth) bins
  - Smooth by bin means, bin median, bin boundaries, etc.
- Regression
  - Smooth by fitting a regression function
- Clustering
  - Detect and remove outliers
- Combined computer and human inspection
  - Detect suspicious values automatically and check by human

Binning Methods

- Step 1: Sort data objects

- Step 2: Partition them into equal frequency bins
- Divides the range into N intervals, each containing the approximately same number of samples (equal-depth partitioning)

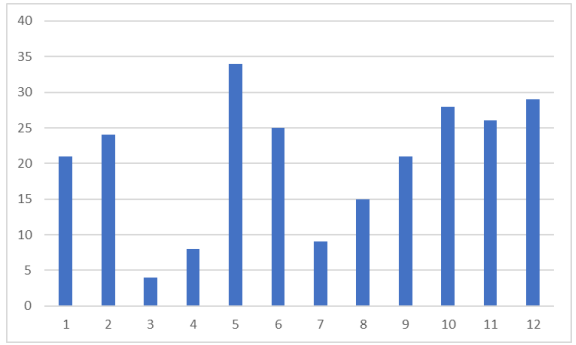
- Step 3: Smooth
  - bin means
  - bin medians
  - bin boundaries

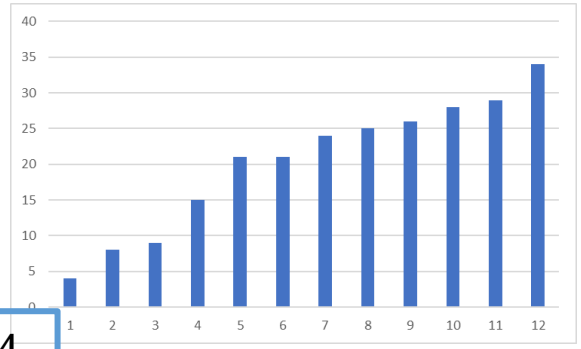
Binning Methods

Original values: 21, 24, 4, 8,34, 25, 9, 15, 21, 28,26,29

sorting

After sorting: 4, 8, 9, 15, 21, 21, 24, 25, 26, 28, 29, 34



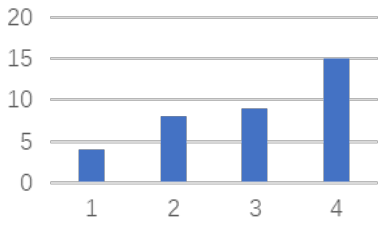


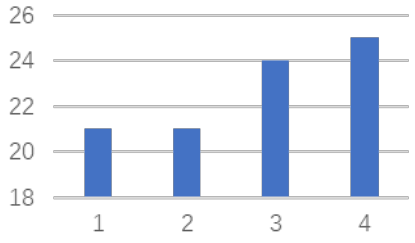
Partition into equal frequency bins.

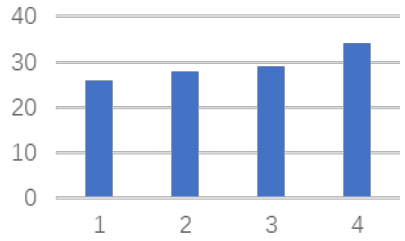
Bin 1: 4, 8, 9, 15

Bin 2: 21, 21, 24, 25

Bin 3: 26, 28, 29, 34

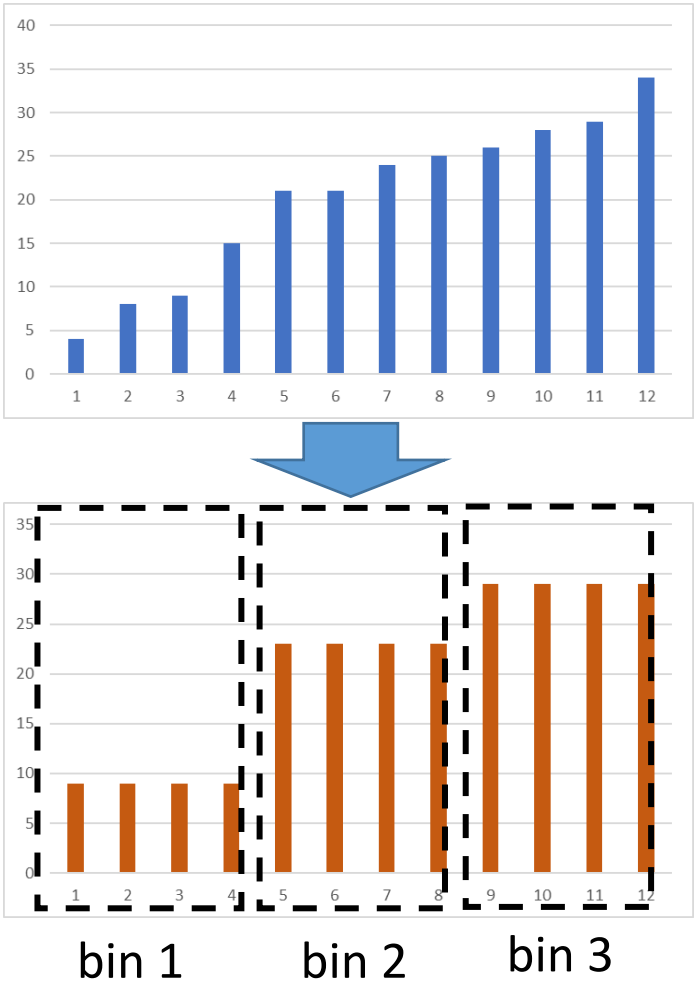






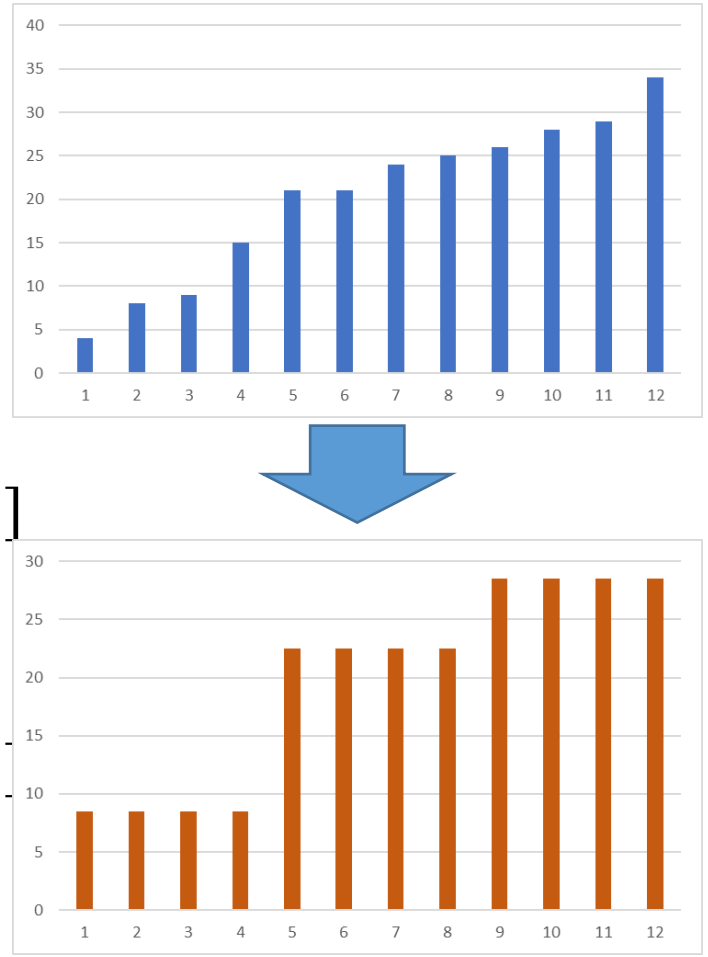
Binning Methods

- For bin 1: 4, 8, 9, 15
- Mean of bin 1 is $\frac{4+8+9+15}{4} = 9$,
  - 4 is number of values in bin 1.
  - After smoothing we have Bin 1 = [9,9,9,9]
- For bin 2: 21, 21, 24, 25
  - Mean is $\frac{21+21+24+25}{4} = 23$
  - After smoothing we have Bin 2 = [23,23,23,23]
- For bin 3: 26, 28, 29, 34
  - Mean is $\frac{26+28+29+34}{4} = 29$
  - After smoothing we have Bin 3 = [29,29,29,29]



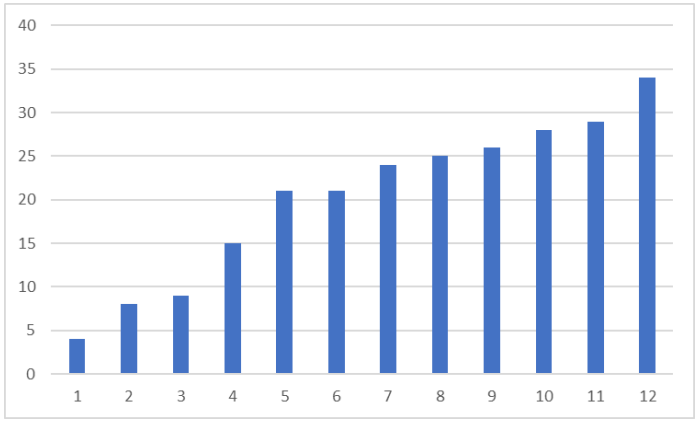
Binning Methods: Smooth by Median

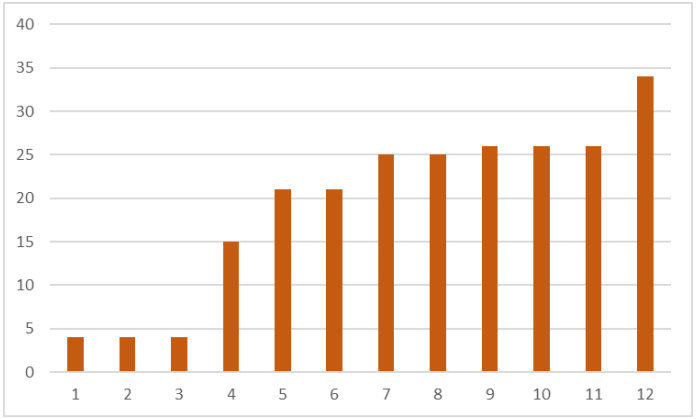
- For bin 1: 4, 8, 9, 15
- Median of bin 1 is $\frac{8+9}{2} = 8.5$,
  - 4 is number of values in bin 1.
  - After smoothing we have Bin 1 = [8.5,8.5,8.5,8.5]
- For bin 2: 21, 21, 24, 25
  - Median is $\frac{21+24}{2} = 22.5$
  - After smoothing we have Bin 2 = [22.5,22.5,22.5,22.5]
- For bin 3: 26, 28, 29, 34
  - Median is $\frac{28+29}{2} = 28.5$
  - After smoothing we have Bin 3 = [28.5,28.5,28.5,28.5]



Binning Methods: Smooth by Boundaries

- Keep the minimum and the maximum values.
- Middle values in the bin are assigned to the closest boundary of the bin
- For bin 1: 4, 8, 9, 15
  - Boundaries are 4 and 15
  - 8 is closer to 4 than 15, (8-4)<(15-8), so we smooth it with 4
  - 9 is closer to 4 than 15, (9-4)<(15-9),so we smooth it with 4
  - After smoothing, we have Bin 1: 4, 4, 4, 15
- For bin 2: 21, 21, 24, 25
  - Boundaries are 21 and 25
  - 24 is closer to 25 than 21, (25-21)<(24-21), so we smooth it with 25
  - After smoothing, we have Bin 2: 21,21,25,25
- For bin 3: 26, 28, 29, 34
  - Boundaries are 26 and 34
  - 28 is closer to 26 than 34, (28-26)<(34-28), so we smooth it with 26
  - 29 is closer to 26 than 34, (29 -26)<(34- 29), so we smooth it with 26
  - After smoothing, we have Bin 3: 26,26,26,34





Outlines

- Data quality issues
- Data cleaning
- Data integration
- Data transformation
- Data reduction

Data Integration

- Purpose
  - A preprocessing step when data are distributed in multiple sources and need to be combined for a comprehensive analysis
- Schema integration
  - Integrate metadata from different sources
  - Attribute identification problem: “same” attributes from multiple data sources may have different names
- Instance integration
  - Integrate instances from different sources
  - For the same instance, the same attributes from different sources may have different values
  - Possible causes
    - different representations, different styles, different scales, errors

Data Integration - Approach

- Identification
  - Detect corresponding tables from different sources manually
  - Detect corresponding attributes from different sources may use correlation analysis
    - e.g., A.cust-id ≡ B.cust-#
  - Detect duplicate records from different sources involves approximate matching of attribute values
    - e.g. 3.14283 ≡ 3.1, Schwartz ≡ Schwarz
- Treatment
  - Merge corresponding tables
  - Remove duplicate records

Outlines

- Data quality issues
- Data cleaning
- Data integration
- Data transformation
- Data reduction

Data Transformation

- Normalization
  - To make different records (from different sources/tests) comparable
- Discretization
  - To allow the application of data mining methods for discrete attribute values
- Attribute/feature construction
  - New attributes constructed from the given ones (derived attributes)
    - pattern may only exist for derived attributes
    - e.g., change of revenue for consecutive years
- Mapping into vector space
  - To allow the application of standard data mining methods

Data Transformation - Normalization

- Min-max normalization

$$v' = \frac{v - min_A}{max_A - min_A}$$

A: attribute
v: original value
v': normalized value

- Z-score normalization

$$v' = \frac{v - mean_A}{std_A}$$

Normalization Example

- Normalize these [2,3,4,5] with Min-max normalization and Z-score normalization

Min-max normalization:

- Max = 5
- Min = 2
- Range = 5-2 =3
- $[\frac{2-2}{3}, \frac{3-2}{3}, \frac{4-2}{3}, \frac{5-2}{3}]= [0, \frac{1}{3}, \frac{2}{3}, 1]$

Z-score normalization:

- Std = 1.29
- Mean = 3.5
- $[\frac{2-3.5}{1.29}, \frac{3-3.5}{1.29}, \frac{4-3.5}{1.29}, \frac{5-3.5}{1.29}]$
- =[-1.16, -0.387, 0.387, 1.16]

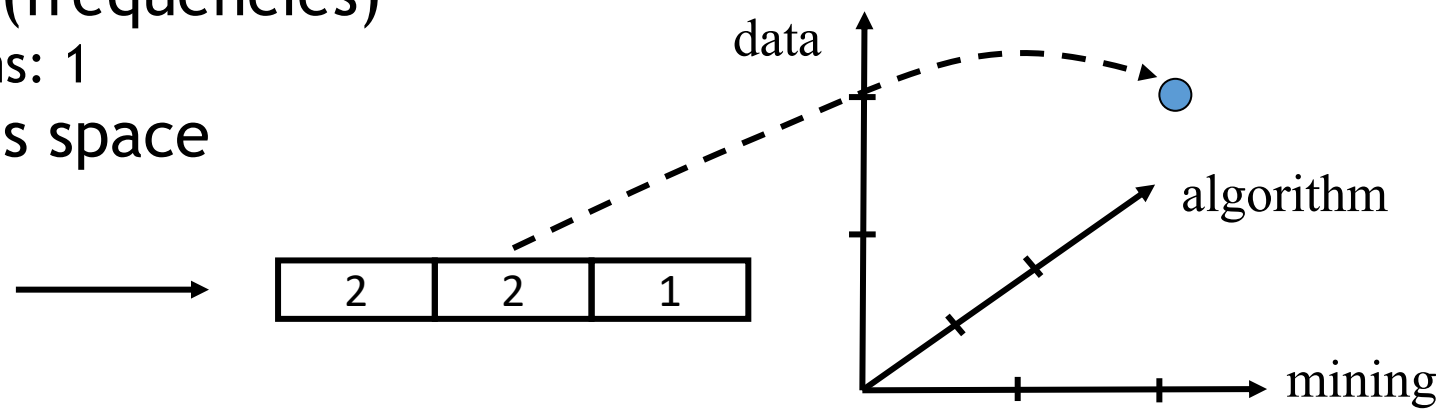
Data Transformation - Discretization

- Three types of attributes
  - Categorical — values from an unordered set
    - colors: blue, red, green
  - Ordinal — values from an ordered set
    - economic status (“low income”, ”middle income”, ”high income”),
    - Grade: ABCDEF
  - Continuous (numerical) — real numbers
- Discretization is the process to transform **continuous** variables, models or functions into a **discrete** form
- Motivation for discretization
  - Some data mining algorithms only accept categorical attributes
  - May improve understandability of patterns

Mapping into Vector Space

- Many algorithms assume that the input is a vector:
  - Linear regression, SVM, neural networks, etc.
- Text documents not represented as records / tuples
  - "Clustering is one of the generic data mining tasks. One of the most important data mining algorithms"
  - Step 1:Choose attributes (relevant terms / dimensions of vector space)
    - "Data", "mining", and "algorithms"
  - Calculate attribute values (frequencies)
    - Data: 2, mining:2 algorithms: 1
  - Map object to vector in this space

Clustering is one of the generic data mining tasks. One of the most important data mining algorithms . . .



Outlines

- Data quality issues
- Data cleaning
- Data integration
- Data transformation
- Data reduction

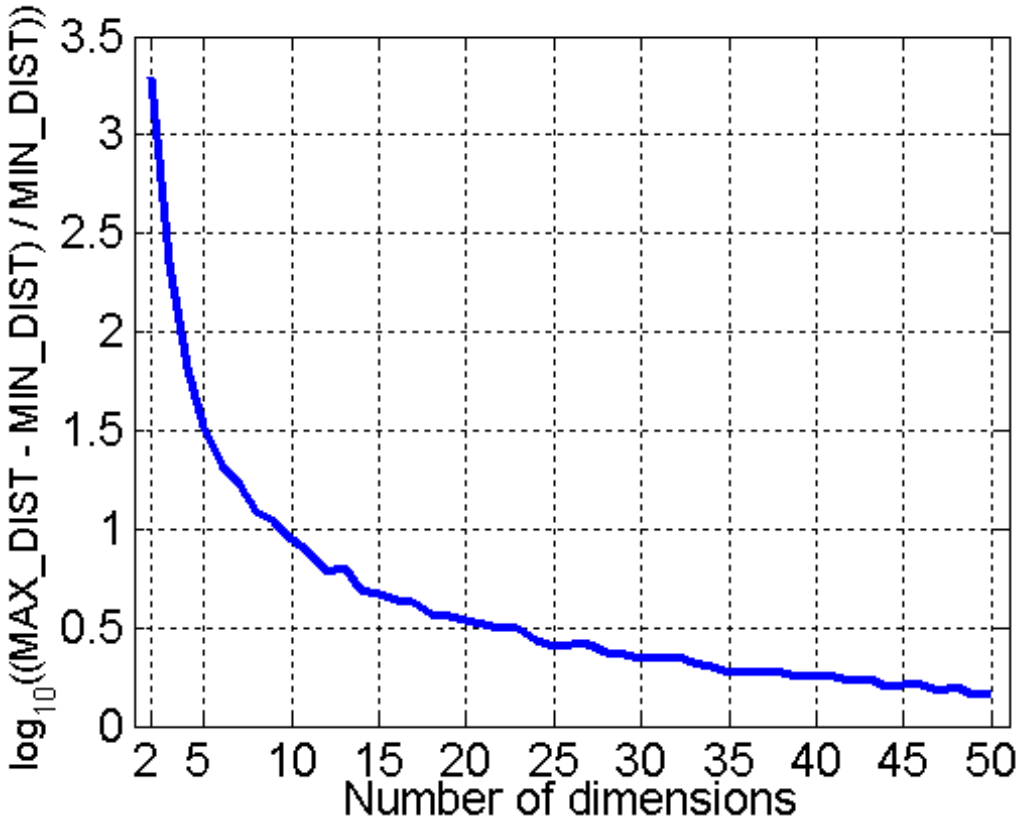
Data Reduction

- Data is too big to work with
- Data reduction
  - Goal: Obtain a reduced representation of the data set that is much smaller in volume but produce the same (or almost the same) analytical results
- Data reduction strategies
  - Dimensionality reduction—remove unimportant attributes
  - Aggregation and clustering
  - Sampling
- Improved Efficiency
  - Runtime of data mining algorithms is (super-)linear w.r.t. number of records and number of attributes
- Improved Quality
  - Removal of noisy attributes improves the quality of the discovered patterns

Data Reduction – Dimensionality Reduction

- When dimensionality increases, data becomes increasingly sparse in the space that it occupies

- Definitions of density and distance between points, which is critical for clustering and outlier detection, become less meaningful



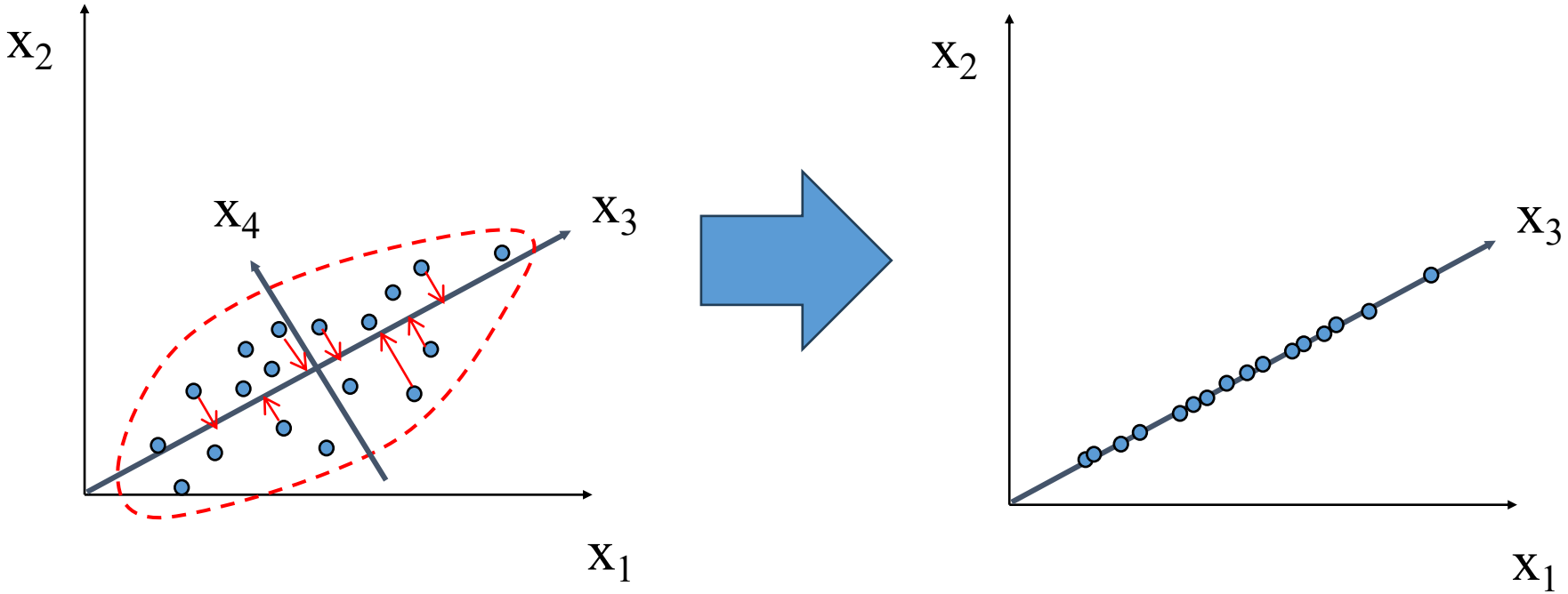
- Randomly generate 500 points
- Compute difference between max and min distance between any pair of points

Data Reduction – Dimensionality Reduction

- Purpose
  - Making the notion of distance meaningful in the feature space
  - Reduce amount of time and memory required by data mining algorithms
  - Allow data to be more easily visualized
  - May help to eliminate irrelevant features or reduce noise
- Techniques
  - Principle Component Analysis (PCA)
  - Singular Value Decomposition (SVD)
  - Multi-dimensional Scaling
  - …

Data Reduction – Dimensionality Reduction

- Principle Component Analysis (PCA)
  - Goal is to find a projection that captures the largest amount of variation in data



Data Reduction – Dimensionality Reduction

- PCA Example
  - Suppose we have 10 2-D data points
  - We want to find a basis vector to project the 2-D data points to 1-D data points while maintaining their largest variation

$$[p_1 \quad p_2] \times \begin{bmatrix} x_{1,1} & x_{1,2} & x_{1,3} & x_{1,4} & x_{1,5} & x_{1,6} & x_{1,7} & x_{1,8} & x_{1,9} & x_{1,10} \\ x_{2,1} & x_{2,2} & x_{2,3} & x_{2,4} & x_{2,5} & x_{2,6} & x_{2,7} & x_{2,8} & x_{2,9} & x_{2,10} \end{bmatrix}$$

1-D basis vector P (Principal components)

Raw data X

$$= \quad [y_1 \quad y_2 \quad y_3 \quad y_4 \quad y_5 \quad y_6 \quad y_7 \quad y_8 \quad y_9 \quad y_{10}]$$

New data Y

Data Reduction – Dimensionality Reduction

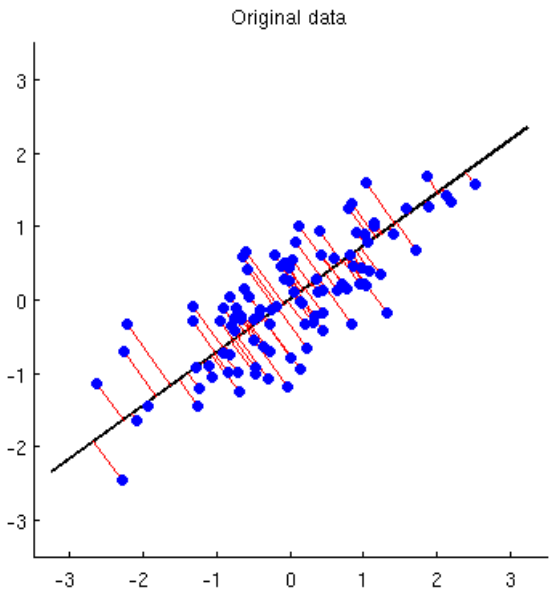
- PCA Example
  - Suppose we have 10 2-D data points
  - We want to find a basis vector to project the 2-D data points to 1-D data points while maintaining their largest variation

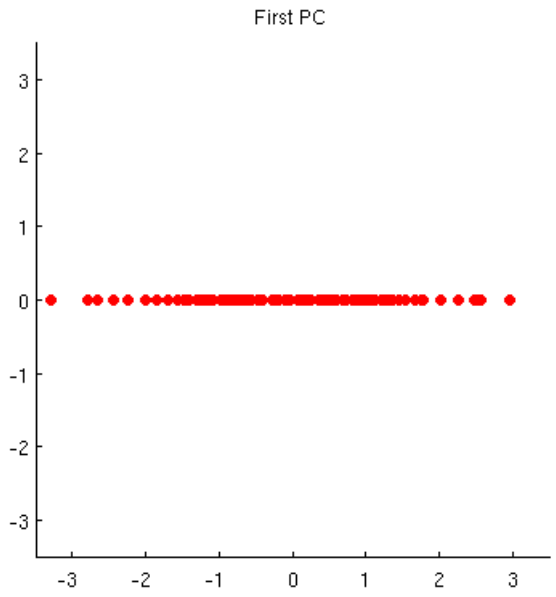
$$P \times X = Y$$

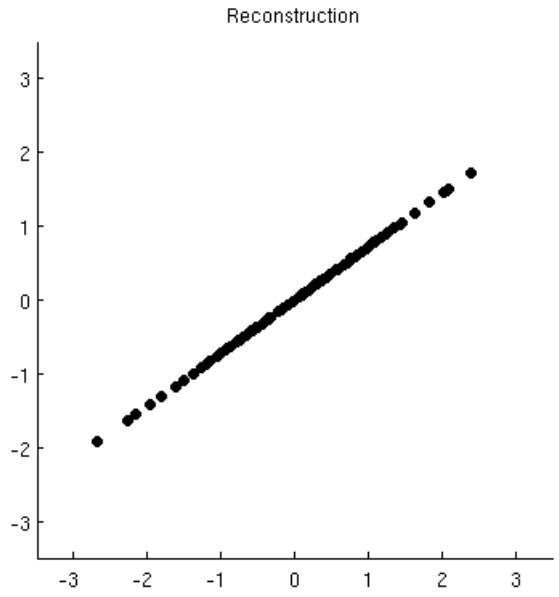
K by M matrix M by N matrix K by N matrix

Data Reduction – Dimensionality Reduction

- PCA Example
  - Reconstruct the original data X (10 by 2) from P (1 by 2) and Y (10 by 1)
    - $X = P^T Y$
  - If each number takes 4 bytes
  - Data compression: from 20 x 4 = 160 bytes to (2 + 10) x 4 = 48 bytes



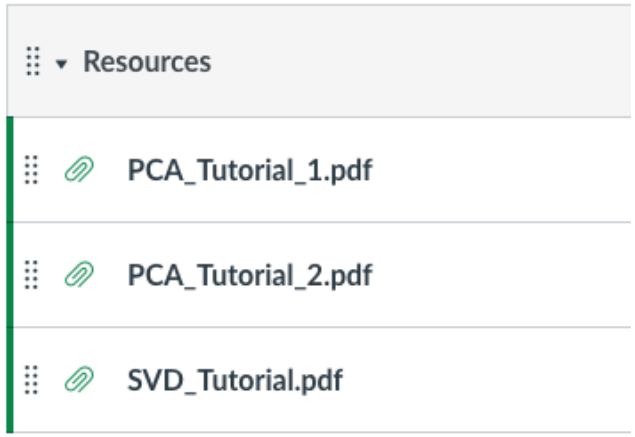




Data Reduction – Dimensionality Reduction

- PCA Computation
  1. Subtract mean from each attribute
  2. Calculate the covariance matrix $S = XX^T$
  3. Calculate the eigenvectors and eigenvalues of S
  4. Choose top K eigenvalues and their corresponding eigenvectors $P \in \mathbb{R}^{k \times m}$
  5. Derive the new data $Y = PX$

Detailed theoretical development of PCA can be found in the articles in Canvas



Data Reduction – Dimensionality Reduction

- PCA Computation

```
>>> from sklearn.decomposition import PCA
>>> pca = PCA(n_components=2)
>>> Y = pca.fit_transform(X)
```

https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.PCA.html

***class* sklearn.decomposition.PCA(*n_components*=None, **, copy*=True, *whiten*=False, *svd_solver*='auto', *tol*=0.0, *iterated_power*='auto', *n_oversamples*=10, *power_iteration_normalizer*='auto', *random_state*=None)**

components_: ndarray of shape (n_components, n_features)
- Principal axes in feature space

explained_variance_: ndarray of shape (n_components,)
- The amount of variance explained by each of the selected components.

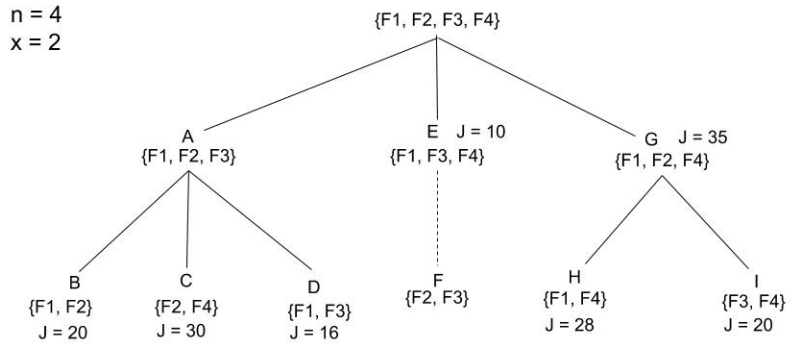
singular_values_: ndarray of shape (n_components,)
- The singular values corresponding to each of the selected components.

Data Reduction – Feature Selection

- Goal: select a relevant subset of all attributes (features)
- For classification
  - Select a minimum set of features s.t. the probability distribution of different classes given the values for those features is as close as possible to the class distribution given the values of all features
  - All features: A, subset of features S, label Y
    - $\underset{S}{\operatorname{argmin}}\,\text{Distance}(P(Y|S), P(Y|A))$
- Problems
  - 2^d possible subsets of set of d features
  - Need heuristic feature selection methods

Data Reduction – Feature Selection

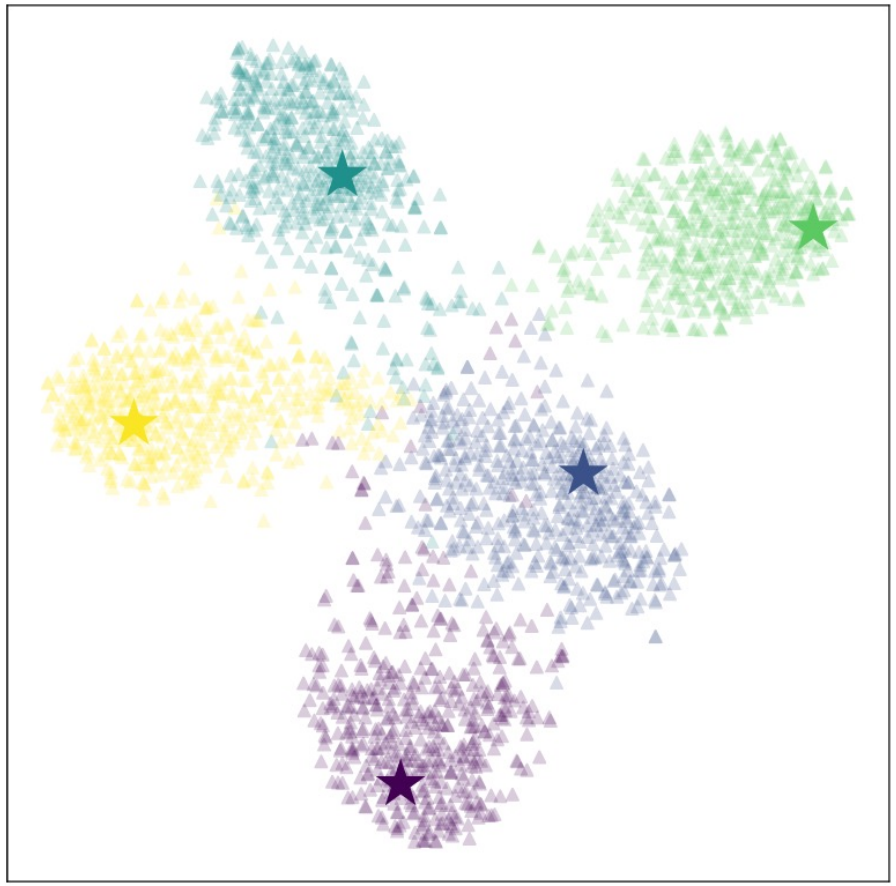
- Feature selection methods
  - Feature independence assumption
    - choose features independently by their significance
  - Greedy bottom-up feature selection:
    - The best single-feature is picked first
    - Then next best feature condition to the first, ...
  - Greedy top-down feature elimination:
    - Repeatedly eliminate the worst feature
  - Branch and bound
    - Returns optimal set of features
    - Requires monotone structure of the feature space



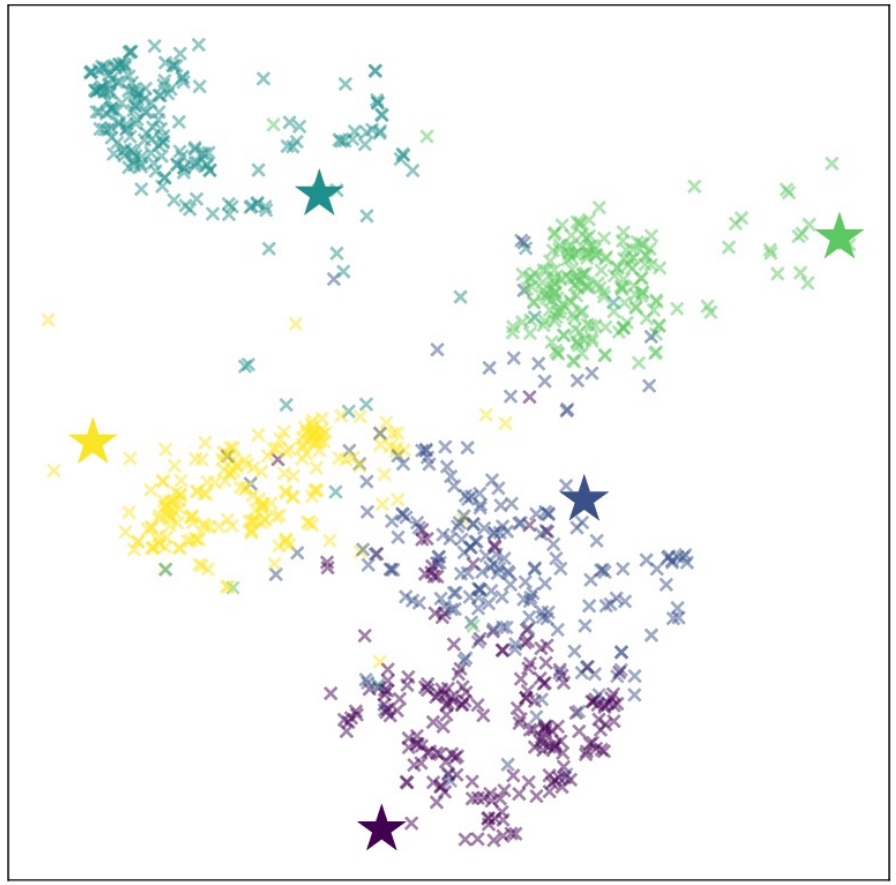
Data Reduction – Clustering

- Partition data set into clusters, and one can store cluster representation only
- Can be very effective if data is clustered but not if data is “smeared”
- There are many choices of clustering definitions and clustering algorithms. We will discuss them later

Data Reduction – Clustering



Data with Good Clusters



Data with Bad Clusters

Data Reduction – Sampling

- Sampling is the main technique employed for data selection
  - It is often used for both the preliminary investigation of the data and the final data analysis

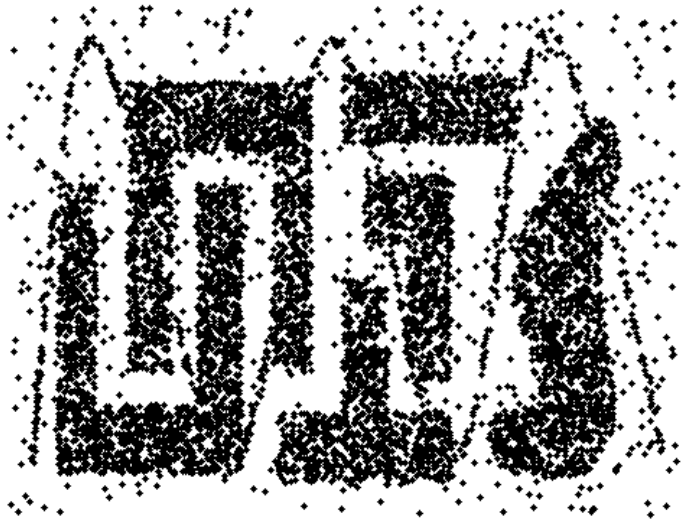
- Statisticians sample because obtaining the entire set of data of interest is too expensive or time consuming

- Sampling is used in data mining/data science because processing the entire set of data of interest is too expensive or time consuming.

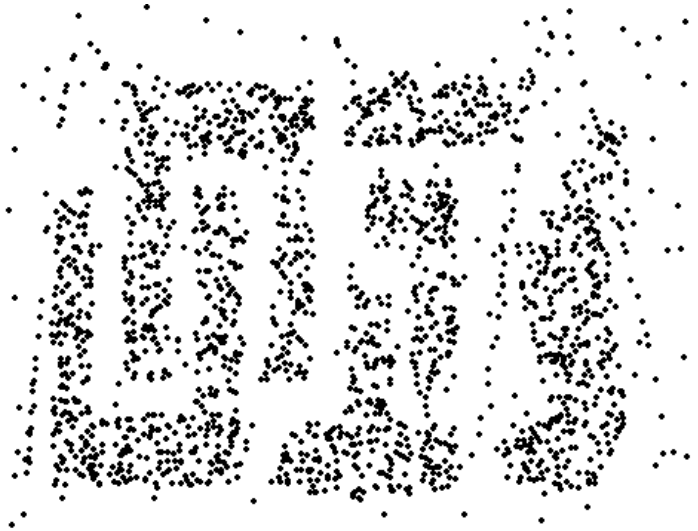
Data Reduction – Sampling

- Task
  - Choose a representative subset of the data records
  - Representative
    - using a sample will work almost as well as using the entire data sets
- Problem
  - Random sampling may overlook small (but important) groups
- Advanced sampling methods
  - Stratified sampling (use percentage instead of absolute numbers)
    - Draw random samples independently from each partition/stratum
    - (e.g. age group)
  - Cluster sampling (use percentage instead of absolute numbers)
    - Draw random samples independently from each given cluster
    - (e.g. customer segment)

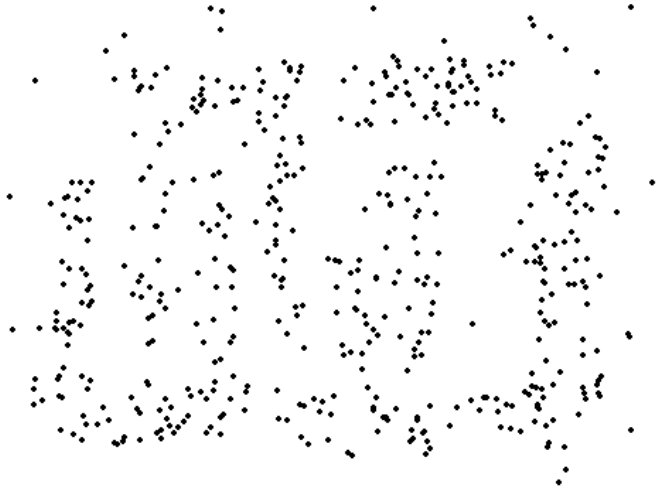
Data Reduction – Sampling



8000 points



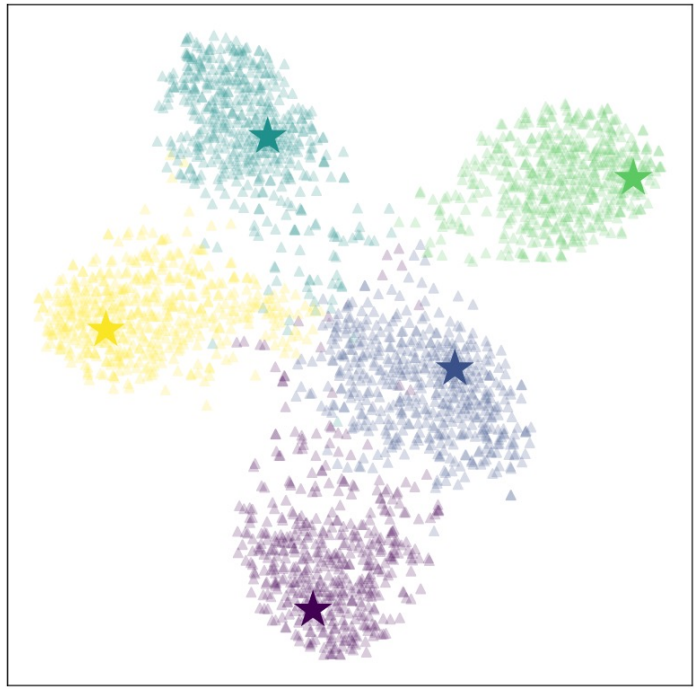
2000 Points



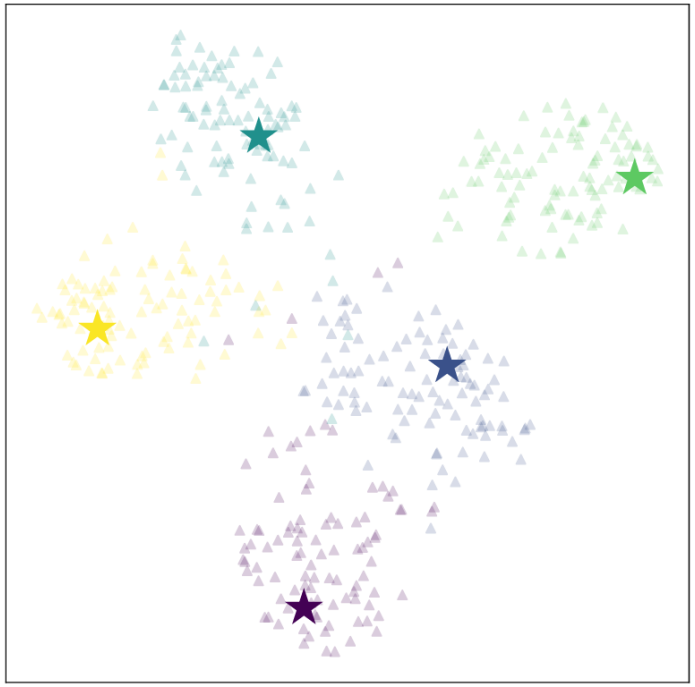
500 Points

Data Reduction – Sampling

- Draw random samples independently from each given cluster



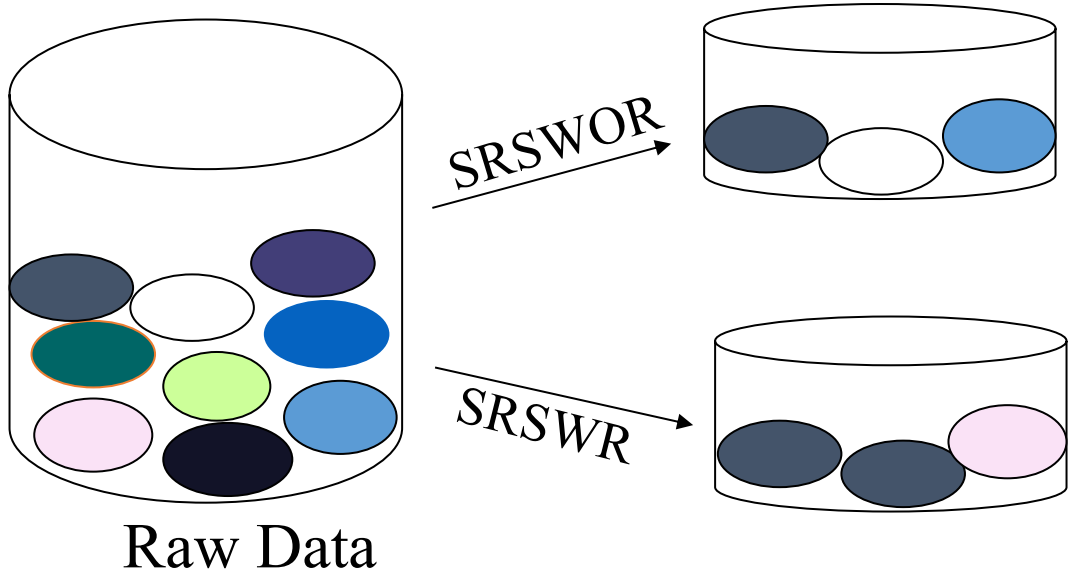
Original Data



Cluster/Stratified Sample
(10% samples per cluster)

Data Reduction – Sampling

- Simple random sample without replacement
  - As each item is selected, it is removed from the population
  - Down-sampling (# samples <= # instances)
- Simple random sample with replacement
  - Objects are not removed from the population as they are sampled
  - Both down-sampling and up-sampling



The same object can be picked up more than once

COSC 3337
Data Science I
Section 14623

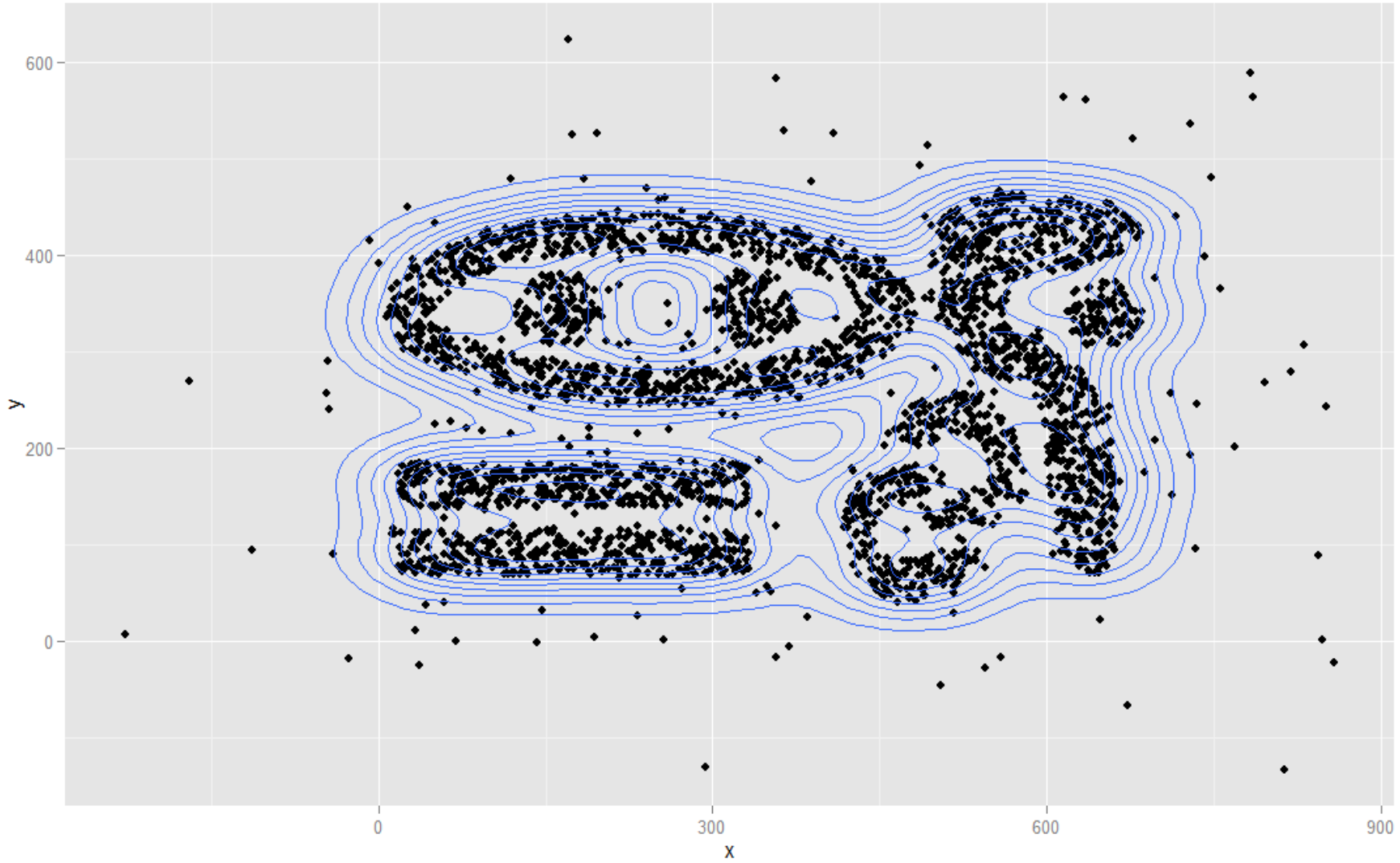
Exploratory Data Analysis

Instructor: Jingchao Ni
Fall 2024

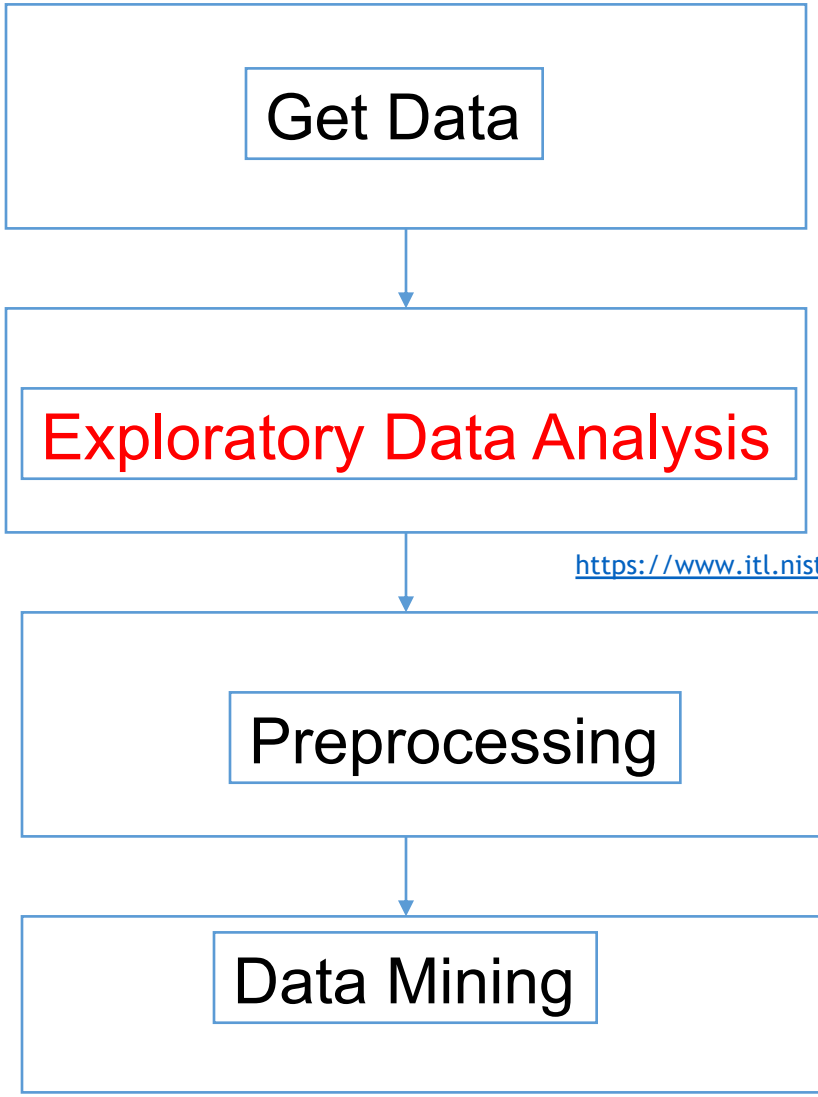
Why Data Exploration?

- A preliminary exploration of the data to better understand its characteristics
  - Key motivations of data exploration include
    - Getting a better understanding of the data
    - Create background knowledge about the data to be analyzed
    - Helping to select the right tool for preprocessing, data analysis and data mining
    - Making use of humans' abilities to recognize patterns
      - People can recognize patterns not captured by data analysis tools
  - Related to the area of Exploratory Data Analysis (EDA)
    - Created by statistician John Tukey
    - Seminal book is Exploratory Data Analysis by Tukey (1977)
    - A nice online introduction can be found in Chapter 1 of the NIST Engineering Statistics Handbook
    - http://www.itl.nist.gov/div898/handbook/index.htm

What Clusters Do You See in the Display?



Exploratory Data Analysis



https://www.itl.nist.gov/div898/handbook/eda/section1/eda14.htm

Techniques Used in Data Exploration

- History: In EDA, as originally defined by Tukey
  - The focus was on visualization
  - Clustering and anomaly detection were viewed as exploratory techniques
  - In data mining, clustering and anomaly detection are major areas of interest, and not thought of as just exploratory

- 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 dataset
  - Can be obtained from the UCI Machine Learning Repository https://archive.ics.uci.edu/
  - From the statistician Douglas Fisher
  - Three flower types (classes):
    - Setosa
    - Virginica
    - Versicolour
  - Four (non-class) attributes
    - Sepal width and length
    - Petal width and length

Virginica. Robert H. Mohlenbrock. USDA NRCS. 1995. Northeast wetland flora: Field office guide to plant species. Northeast National Technical Center, Chester, PA. Courtesy of USDA NRCS Wetland Science Institute.

Summary Statistics

- Summary statistics are numbers that summarize properties of the data
  - Summarized properties include frequency, location and spread
    - Examples:
      - location – mean
      - spread - standard deviation
  - Most summary statistics can be calculated in a single pass through the data

Frequency and Mode

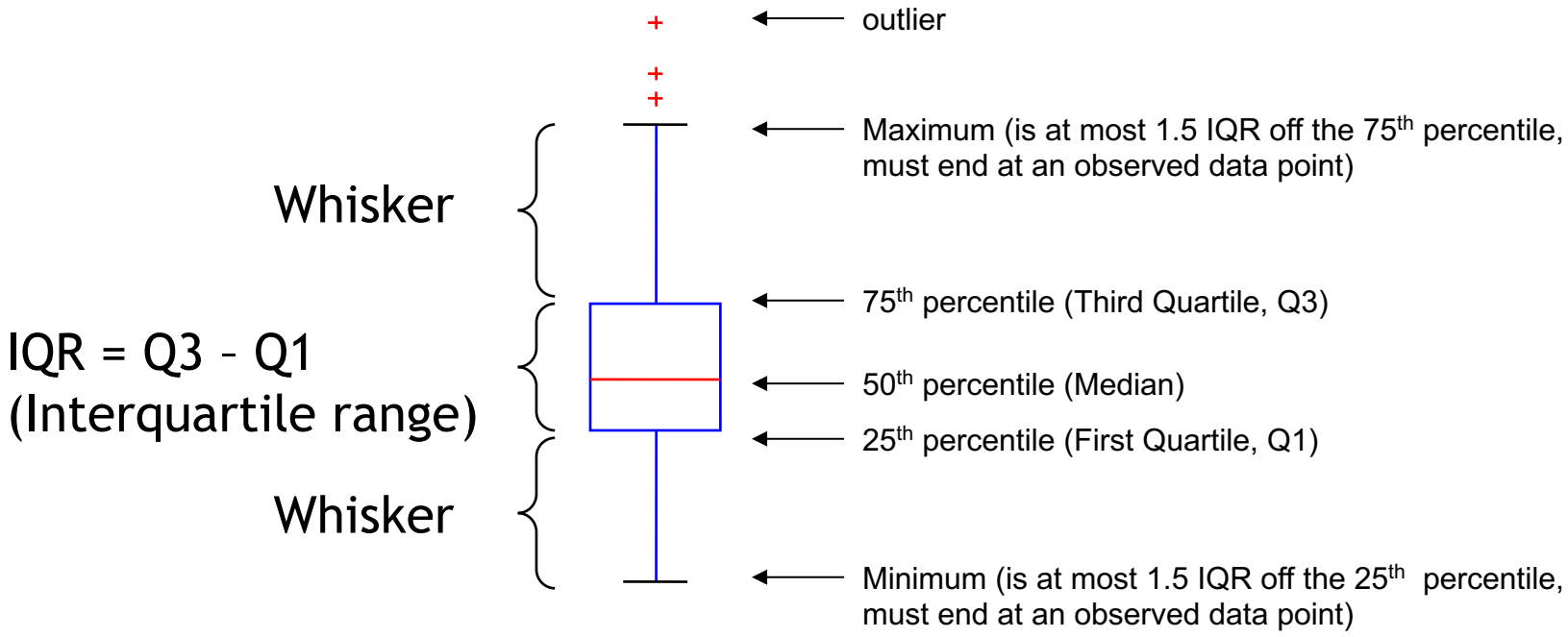
- The frequency of an attribute value is the percentage of time the value occurs in the data set
  - For example, given the attribute ‘gender’ and a representative population of people, the gender ‘female’ occurs about 50% of the time
- The mode of an attribute is the most frequent attribute value
- The notions of frequency and mode are typically used with categorical 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 pth 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 $x_{50\%}$ value such that 50% of all values of x are less than $x_{50\%}$
- Key Idea: associate attribute value with a rank

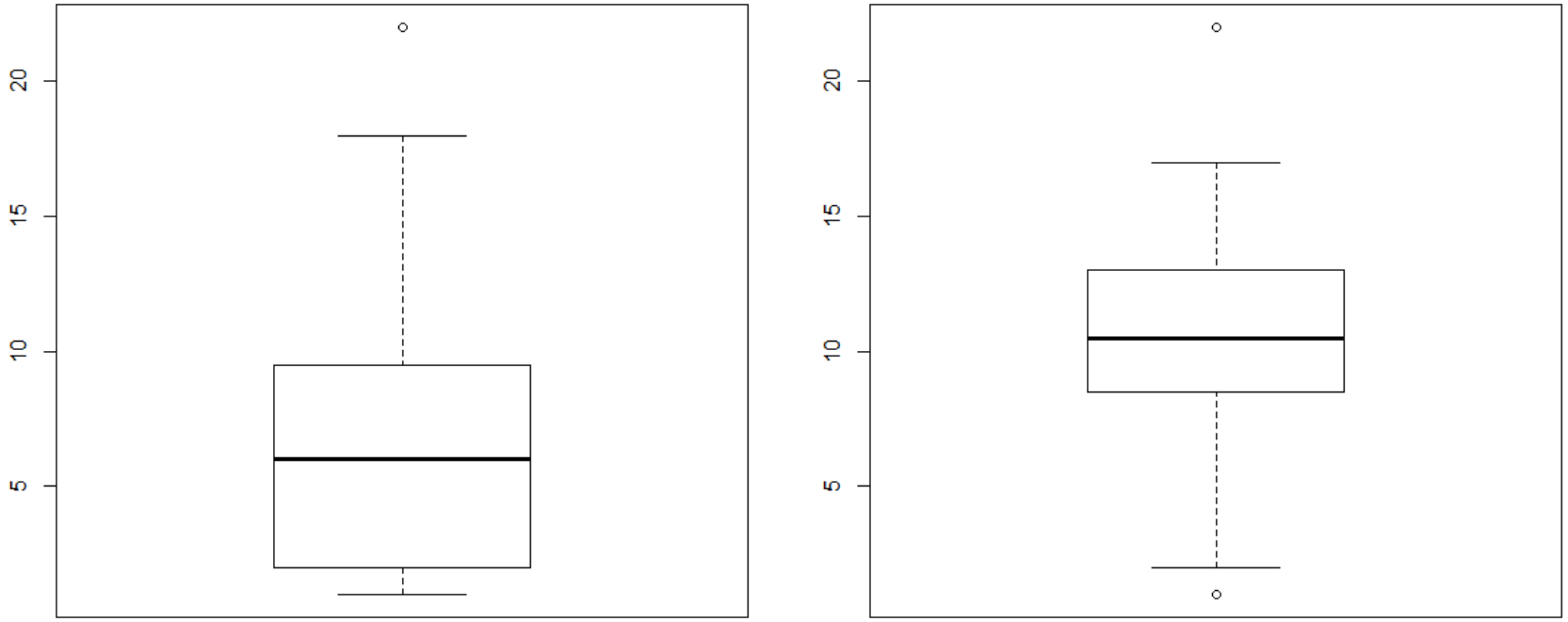
Box Plots

- Box Plots
  - Invented by J. Tukey
  - Displaying the distribution of data based on percentiles
  - Following figure shows the basic parts of a box plots



Compare Two Boxplots

- x<-c(1,2,2,2,4,4,8,9,9,10,18,22)
- y<-c(1,2,8,9,10,10,11,11,12,14,17,22)



Compare Two Boxplots

- Questions to ask:
  1. Compare the two medians: are the two medians: more or less the same: at most 1/10 box_size apart, somewhat similar: 0.1 to 0.25 box_size apart, different: 0.25 to 1 box_size apart, very different: more than 1 box_size apart
  2. Compare the IQRs of the two box plots by interpreting them as intervals, computing their overlap (overlap:=length of the intersection of the intervals they cover/length of the union of the intervals they cover): 90-100%: IQRs almost the same; 75-90%: quite similar, 25-75: somewhat overlapping, 25-10%: dissimilar, 10% or less: very dissimilar.
  3. Compare the intervals defined by the two whiskers (WI) of the two boxplots by computing their WI overlap, and draw conclusions as follows: Range of attribute is: 90-100%: almost the same; 75-90%: quite similar, 25-75%: has medium overlap, 25-10%: dissimilar, 10% or less: very dissimilar.
  4. Assess and compare skewness by summarizing the position of the median in the box
  5. Mention and assess agreement/disagreement with respect to outliers that occur in both boxplots
  6. Combine the agreement/disagreement in the 2 boxplots you observed by transforming the answers to questions 1-5 into a final summary.

  Heuristic: We use the IQR as a proxy for standard deviation when comparing the respective distributions and set box_size to (IQR1+IQR2)/2!

Measures of Location: Mean and Median

- The mean is the most common measure of the location of a set of points
- However, the mean is very sensitive to outliers
- Thus, the median or a trimmed mean is also commonly used

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

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

Measures of Spread: Range and Variance

- Range is the difference between the max and min
- The variance or standard deviation

0, 2, 3, 7, 8 → $\bar{x} = 4$

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

9.2

standard_deviation(x)= s_x

3.3

- However, this is also sensitive to outliers, so that other measures are often used

2.8

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

(Mean Absolute Deviation) [Han]
(Absolute Average Deviation) [Tan]

3

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

(Median Absolute Deviation)

5

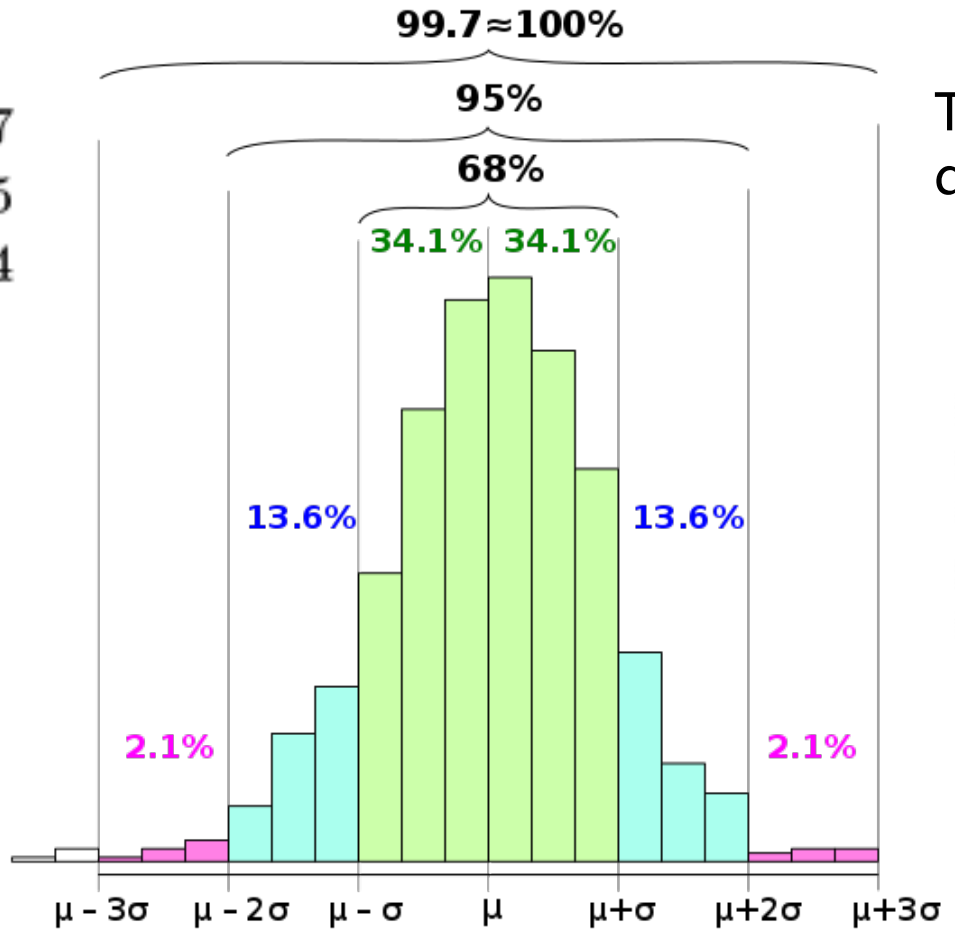
$$\text{interquartile range}(x) = x_{75\%} - x_{25\%}$$

68-95-99.7 Rule (Empirical Rule)

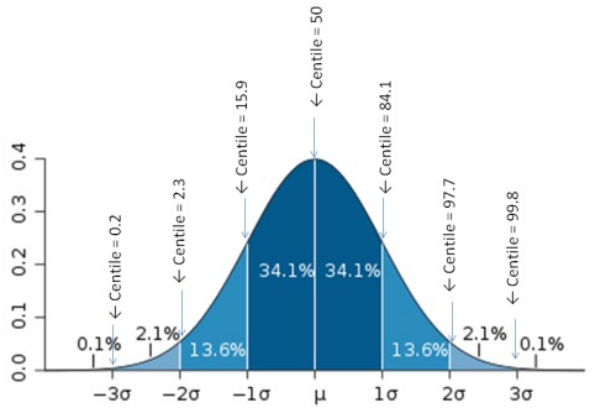
For more details see: https://en.wikipedia.org/wiki/68%E2%80%9395%E2%80%9399.7_rule

$$\Pr(\mu - \sigma \leq X \leq \mu + \sigma) \approx 0.6827$$
$$\Pr(\mu - 2\sigma \leq X \leq \mu + 2\sigma) \approx 0.9545$$
$$\Pr(\mu - 3\sigma \leq X \leq \mu + 3\sigma) \approx 0.9974$$

Remark:
μ is mean value and
σ is standard deviation



This rule assumes a Gaussian distribution!

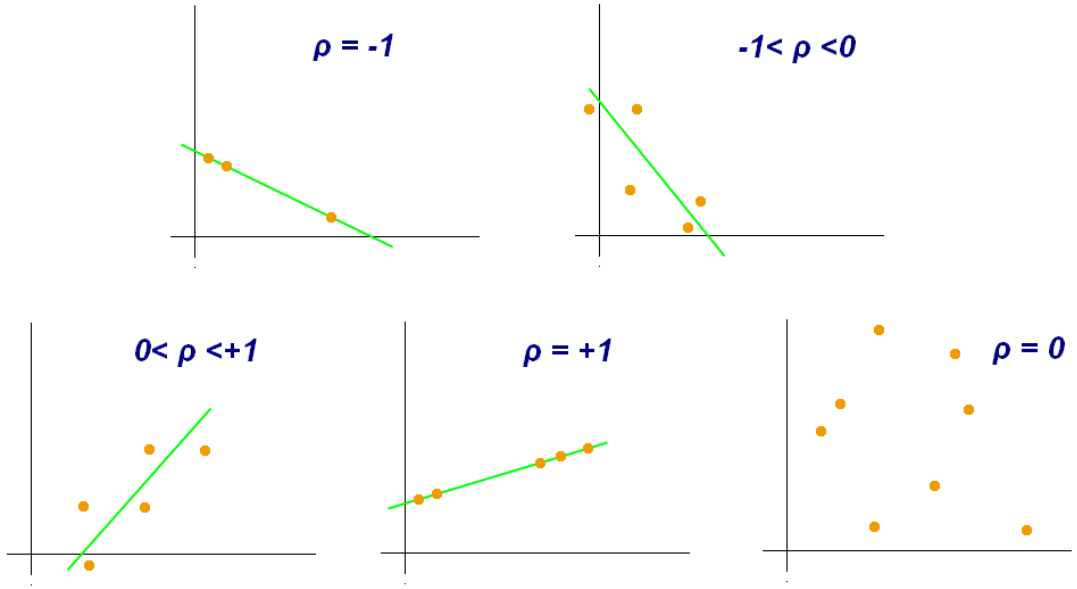


68-95-99.7 Rule (Empirical Rule)

Range	Expected fraction of population inside range	Approximate expected frequency outside range	Approximate frequency outside range for daily event
$\mu \pm 0.5\sigma$	0.382924922548026	3 in 5	Four times a week
$\mu \pm \sigma$	0.682689492137086	1 in 3	Twice a week
$\mu \pm 1.5\sigma$	0.866385597462284	2 in 15	Weekly
$\mu \pm 2\sigma$	0.954499736103642	1 in 22	Every three weeks
$\mu \pm 2.5\sigma$	0.987580669348448	1 in 81	Quarterly
$\mu \pm 3\sigma$	0.997300203936740	1 in 370	Yearly
$\mu \pm 3.5\sigma$	0.999534741841929	1 in 2149	Every six years
$\mu \pm 4\sigma$	0.999936657516334	1 in 15787	Every 43 years (twice in a lifetime)

Correlation

- Pearson Correlation Coefficient
  - PCC measures linear correlation between two sets of data
  - It is the ratio between the covariance of two variables and the product of their standard deviations
  - A normalized measurement of the covariance, such that the result always has a value between −1 and 1
  - The measure can only reflect a linear correlation of variables, and ignores many other types of relationships or correlations



$$\rho_{X,Y} = \frac{cov(X,Y)}{\sigma_X \sigma_Y}$$

$$cov(X,Y) = \mathbb{E}[(X - \mu_X)(Y - \mu_Y)]$$

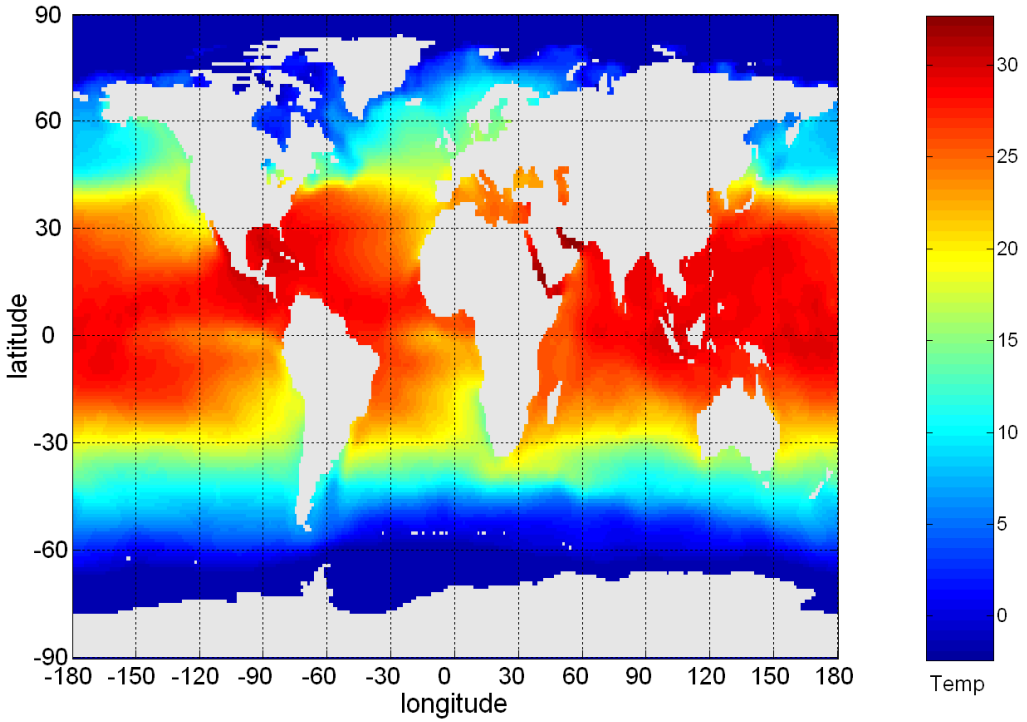
```
>>> import numpy as np
>>> from scipy import stats
>>> x = np.random.rand(10)
>>> y = np.random.rand(10)
>>> res = stats.pearsonr(x, y)
```

Visualization

- Visualization is the conversion of data into a visual or tabular format so that the characteristics of the data and the relationships among data items or attributes can be analyzed or reported
- Visualization of data is one of the most powerful and appealing techniques for data exploration
  - Humans have a well-developed ability to analyze large amounts of information that is presented visually
  - Can detect general patterns and trends
  - Can detect outliers and unusual patterns

Example: See Surface Temperature

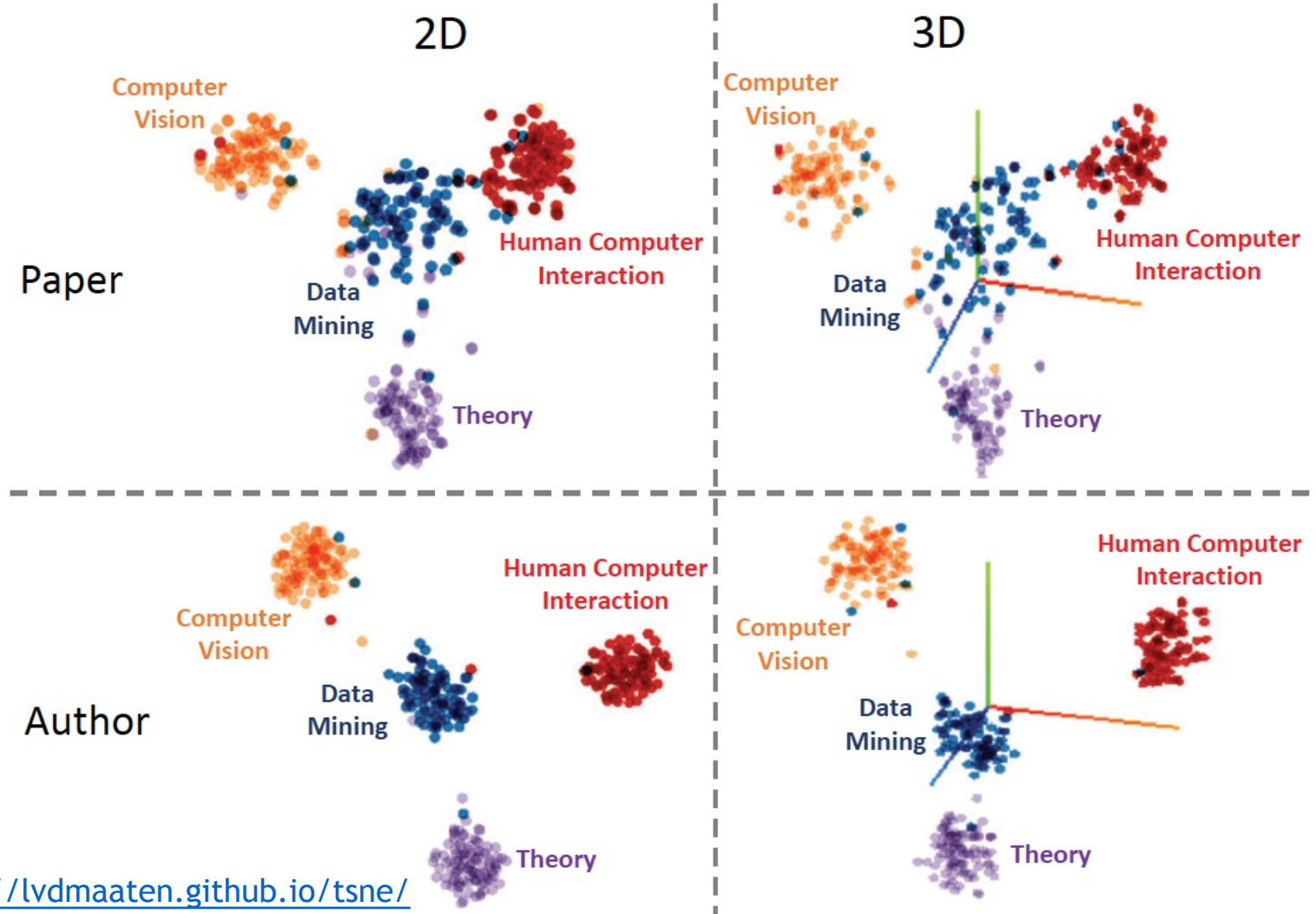
- The following shows the Sea Surface Temperature (SST) for July 1982
  - Tens of thousands of data points are summarized in a single figure

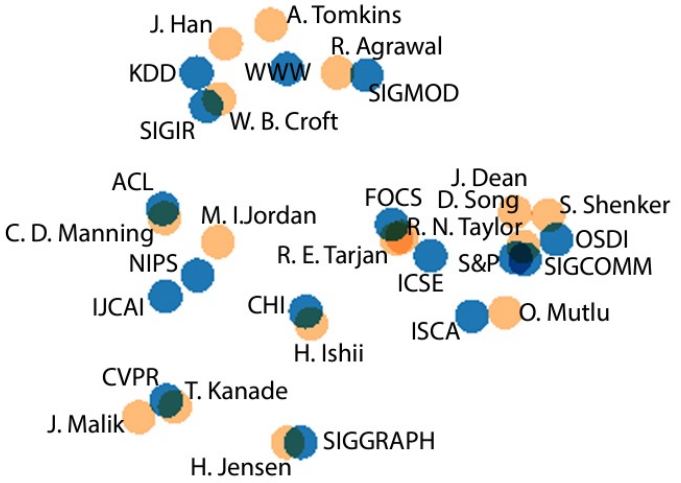


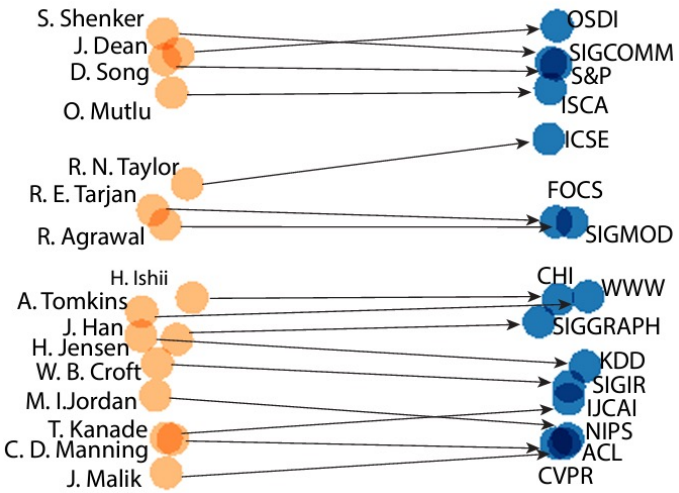
Representation

- Is the mapping of information to a visual format
- Data objects, their attributes, and the relationships among data objects are translated into graphical elements such as points, lines, shapes, and colors
- Example:
  - Objects are often represented as points
  - Their attribute values can be represented as the position of the points or the characteristics of the points, e.g., color, size, and shape
  - If position is used, then the relationships of points, i.e., whether they form groups or a point is an outlier, is easily perceived

Example: Visualizing Bibliography Data



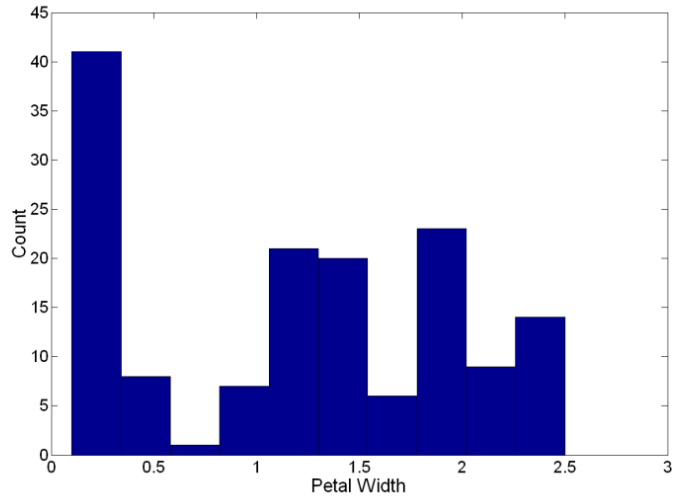


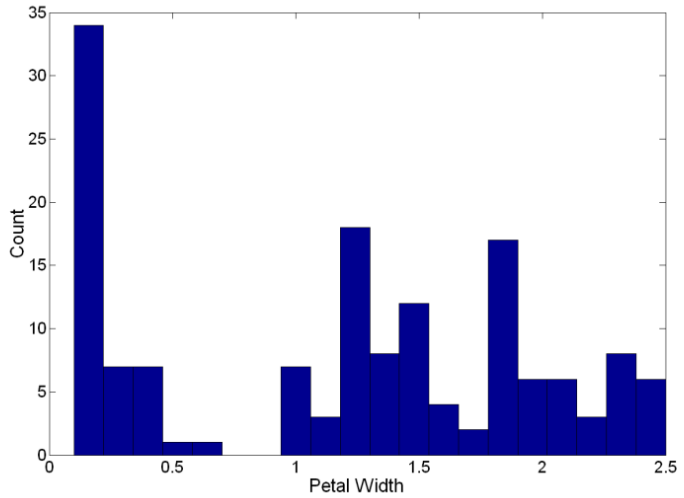


https://lvdmaaten.github.io/tsne/
https://scikit-learn.org/stable/modules/generated/sklearn.manifold.TSNE.html

Visualization Techniques: Histogram

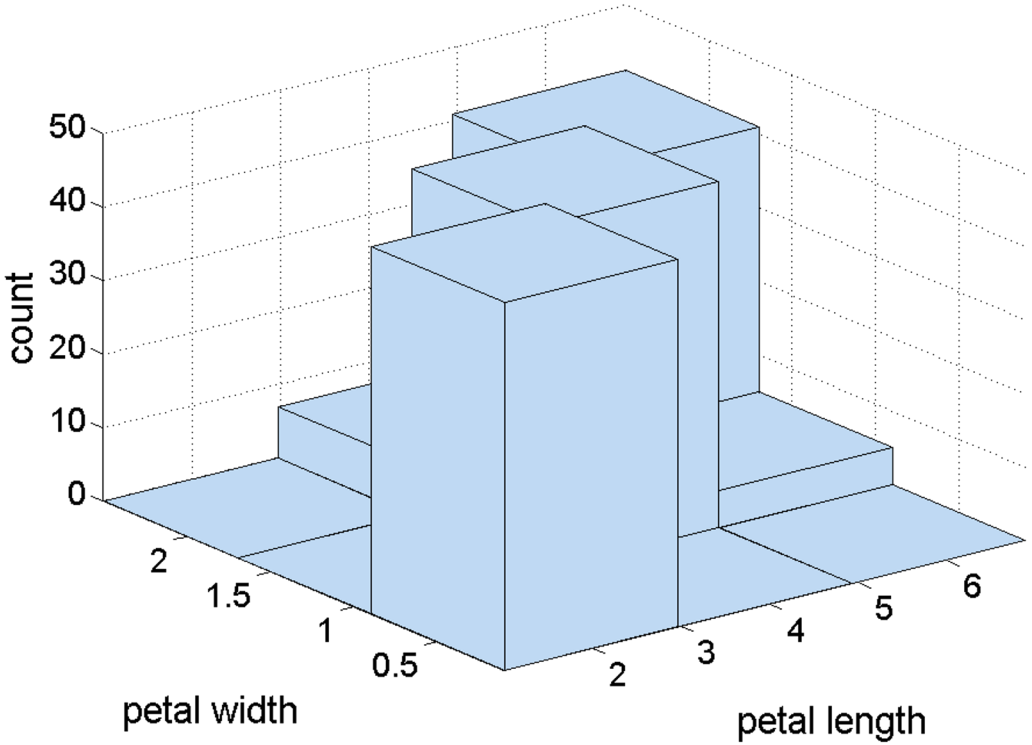
- Histogram
  - Usually shows the distribution of values of a single variable
  - Divide the values into bins and show a bar plot of the number of objects in each bin.
  - The height of each bar indicates the number of objects
  - Shape of histogram depends on the number of bins and origin
- Example: Petal Width (10 and 20 bins, respectively)





Two-Dimensional Histograms

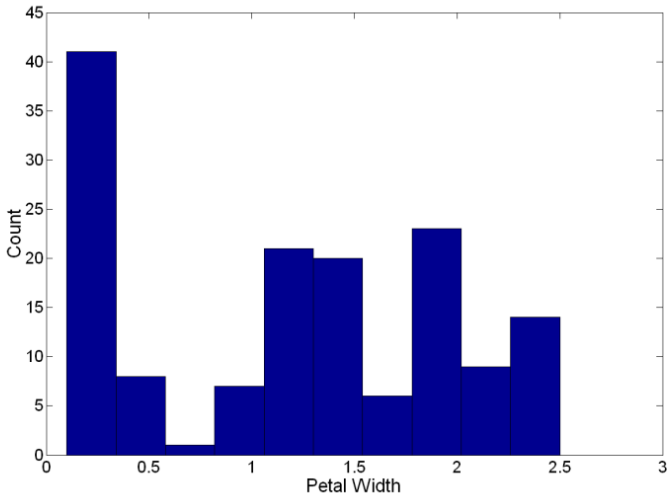
- Show the joint distribution of the values of two attributes
- Example: petal width and petal length
  - What does this tell us?

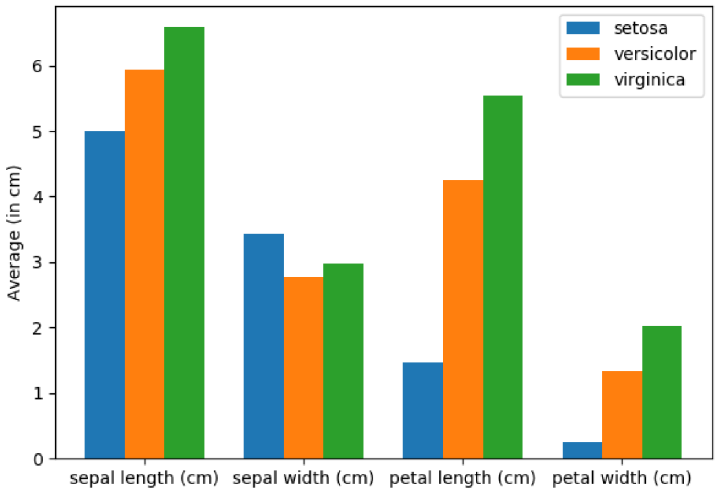


Histograms vs Bar Chart

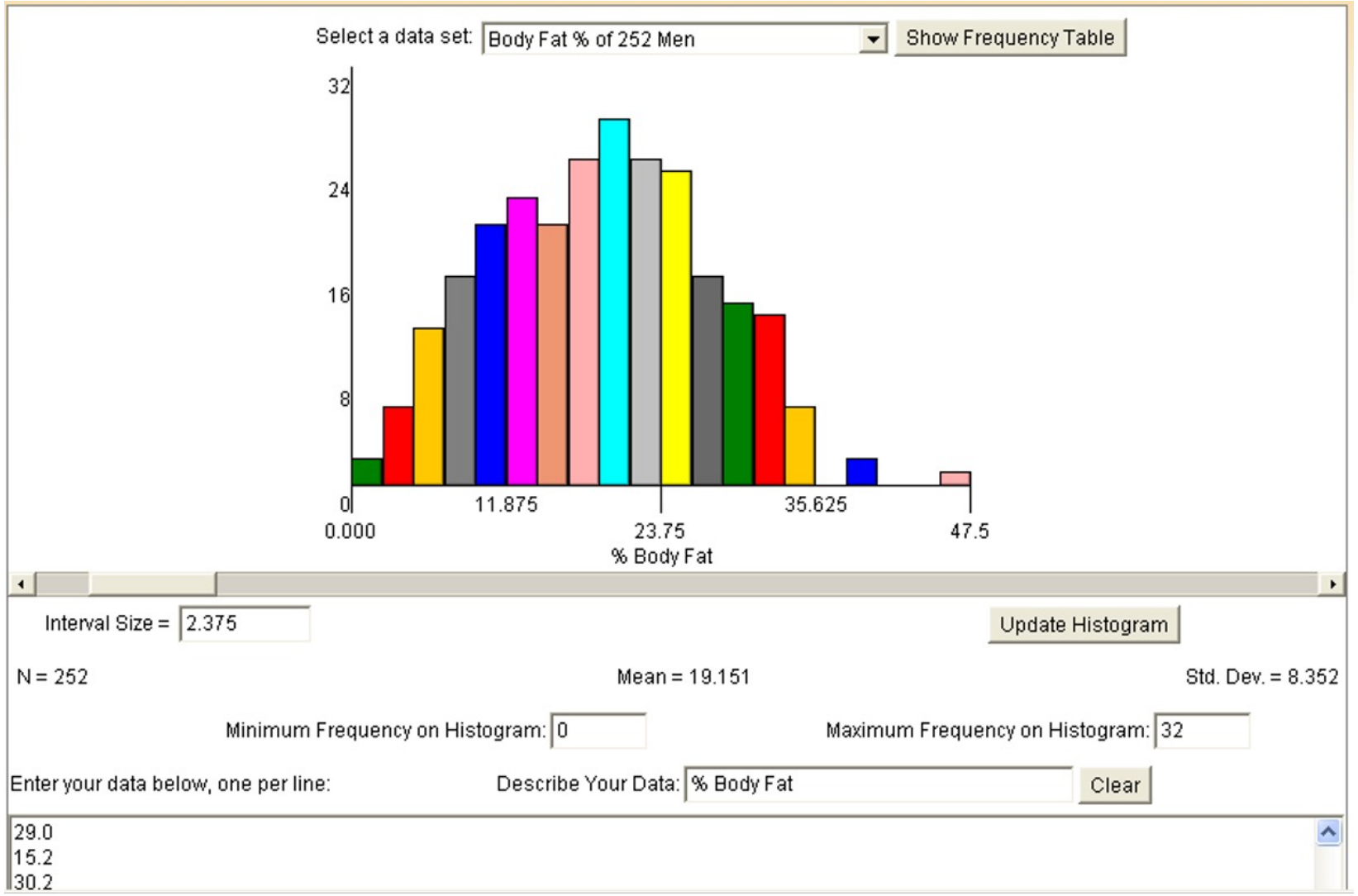
In a histogram, each bin is for a different range of values, so altogether the histogram illustrates the distribution of values.

In a bar chart, each bar is for a different category of observations, so altogether the bar chart can be used to compare different categories. Some authors recommend that bar charts always have gaps between the bars to clarify that they are not histograms.





Body Fat Histogram B

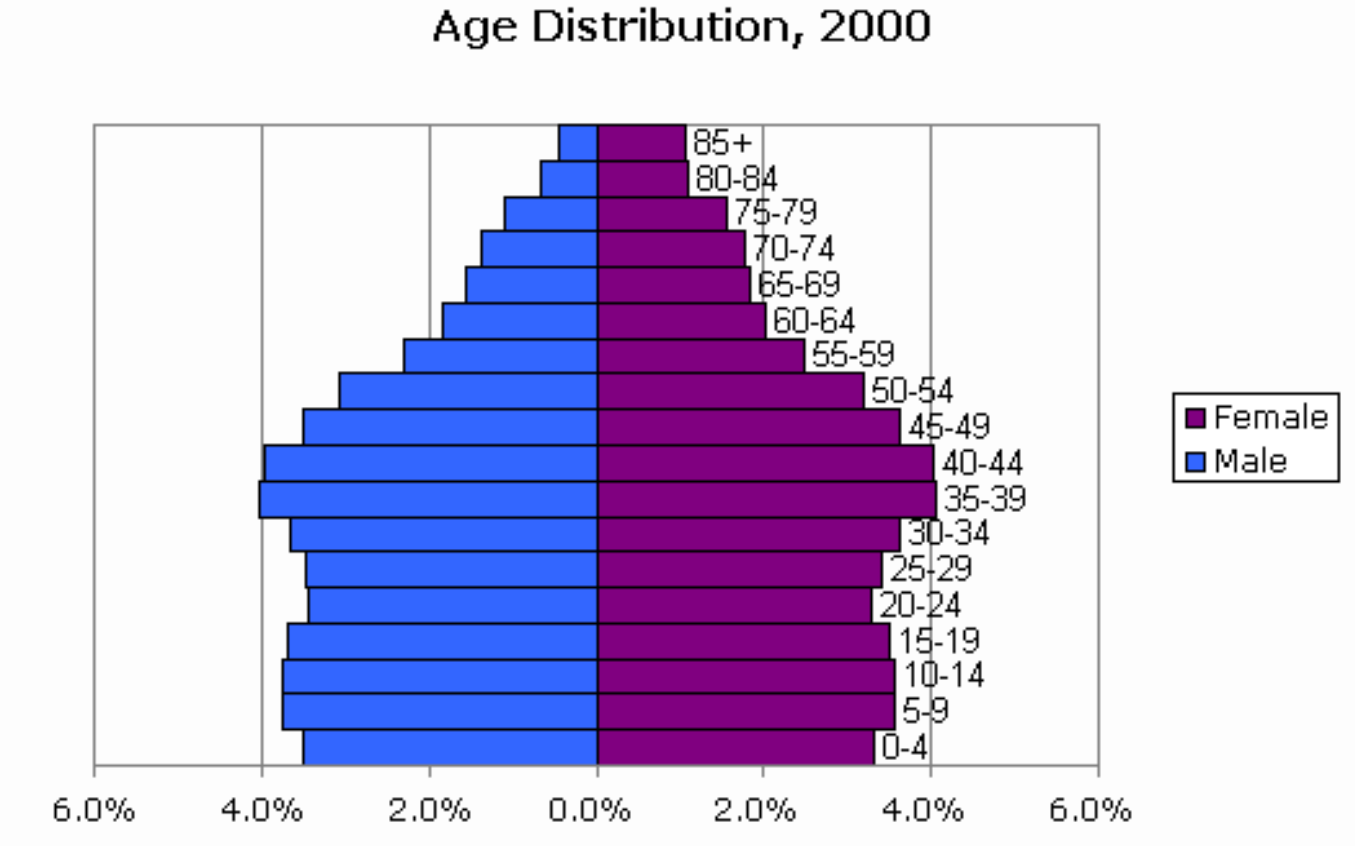


Interpretation of Histogram

- What is the type of the attribute? Positive real numbers
- What is the mean value?
- Is there a lot of spread or not (compute the standard deviation)? Not much
- Is the distribution unimodal (one hill or no hill)) or multi-modal (multiple hills)? One hill or two hills, depending on how you interpret the data. The second hill is not very well separated; therefore I would say unimodal.
- Is the distribution skewed (e.g. compare mean with median)? Only very mildly skewed
- Are there any outliers? Yes values above 45 …
- Are there any duplicate values?
- Are there any gaps in the attribute value distribution? Yes two gaps: 1)… 2)…
- Characterize the shape of the density function! Bell Curve

Compare Two Histograms

- Interpret the following 2 histograms and their relationships which describe the male and female age distribution in the US, based on Census Data



Compare Two Histograms

- Both histograms:
  - curves are continuous with no gaps or outliers, and somewhat smooth,
  - bimodal with 2 modalities not well separated maxima at 5-19 and 35-44,
  - values significantly drop beyond age 55 → skewed distribution

- Comparison:
  - curves are somewhat similar until age 55 (although there are more males initially); e.g. shape of the density function and the 2 local maxima match
  - however, the decline in the male curve is significantly steeper: women live longer

Attribute Standardization/Normalization

- The domains of numerical attributes vary a lot in their scale: e.g., height varies between 0 and 8 feet and age varies between 0 and 110 years
- For many data analysis and data mining tasks we want to make attributes “equally important” by using the same range of their attribute values
- The two most popular attribute normalization/standardization include:
  - Z-scores
  - Min-Max Normalization

Attribute Standardization/Normalization

- Standardize data using z-scores
  - Calculate the mean, the standard deviation sf :
  - Calculate the standardized measurement (z-score)

$$z_{if} = \frac{x_{if} - m_f}{s_f}$$

  - Result of Z-score standardization is a dataset in which each attribute has a mean of 0 and a standard deviation of 1

- Min-Max normalization

$$z_{if} = (x_{if} - \min_f) / ((\max_f - \min_f)$$

  - where $\min_f$ denotes the minimum value and $\max_f$ denotes the maximum value of the f-th attribute in the data set
  - How it works: The minimum value is mapped to 0 and the maximum value is mapped to 1, and for the in-between values linear interpolation is used
  - Remark: frequently used after applying some form of outlier removal

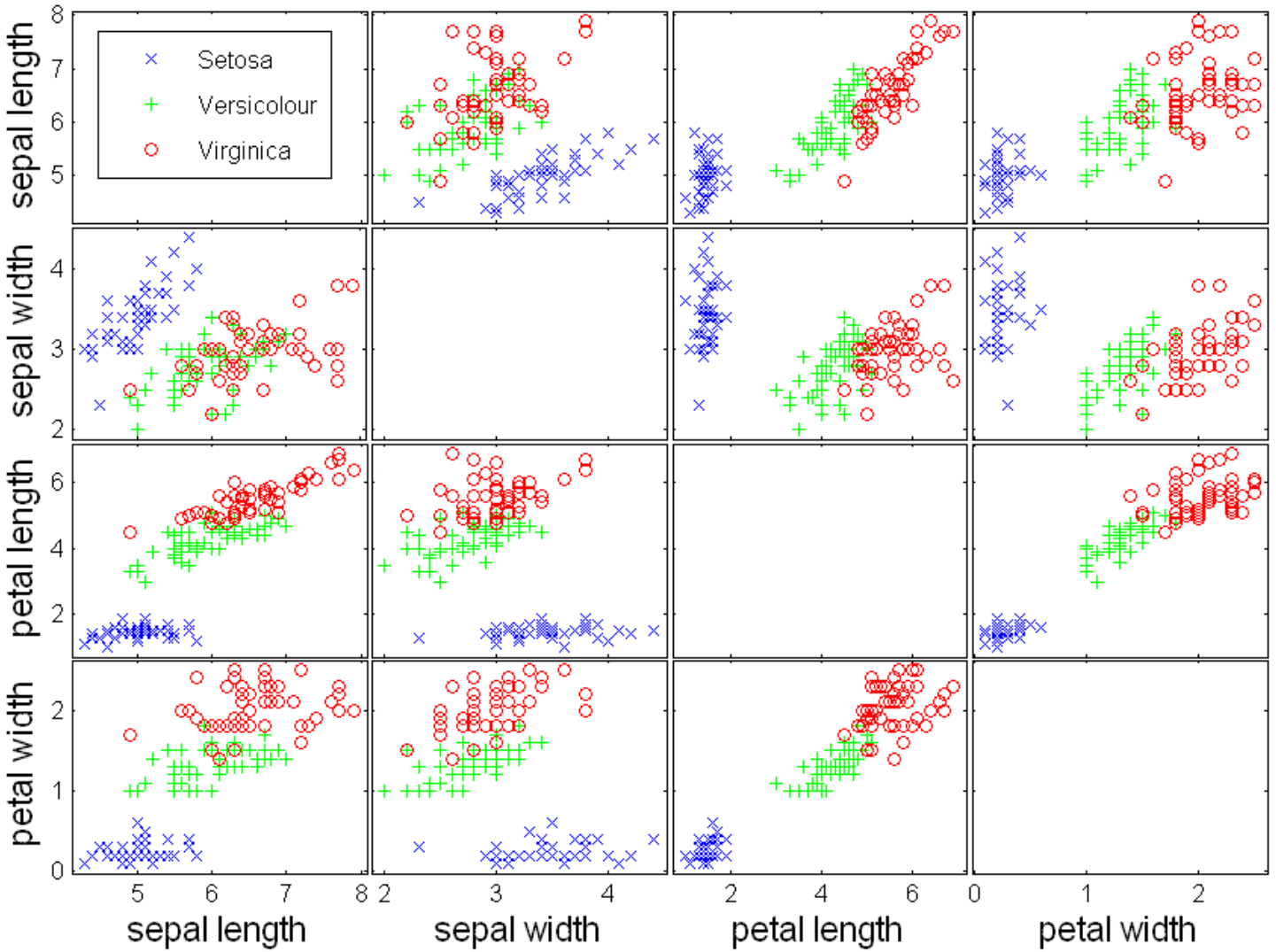
Attribute Standardization/Normalization

- Having the value of 1 in Min-Max normalization indicates the maximum value, and a value of 0 indicates the minimum value and a value of 0.5 indicates the mean of max and min values
- A z-score of -2 indicates a value 2 standard deviations below the mean value, a value of 0 indicates the mean value of the attribute and a z-score of +1 indicates a value one standard deviation above the mean value.
- As z-scores allow for statistical interpretation they are usually the preferred choice, but some scientists prefer [0,1] normalization, as they do not like to work with negative values in the dataset.
- Datasets are frequently standardized before applying a data mining technique to the dataset

Visualization Techniques: Scatter Plots

- Scatter plots
  - Attribute 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 to compactly summarize the relationships of several pairs of attributes
    - For more about scatter plots see:
      - http://en.wikipedia.org/wiki/Scatter_plot
      - See example for classification, also called supervised scatter plots

Scatter Plot Array of Iris Attributes

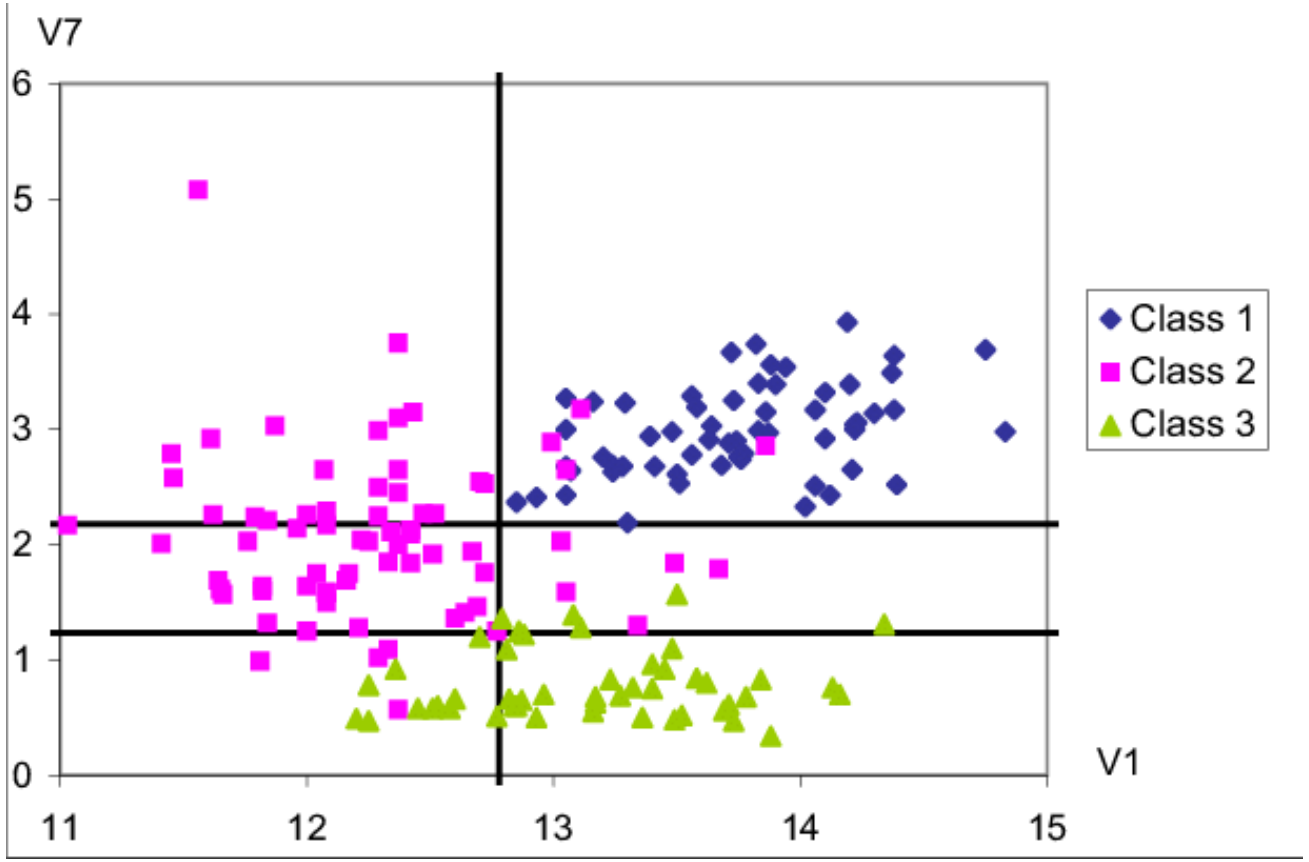


Interpreting Scatter Plots

- Questions to Ask
  - Characterize the distribution of each class in the attribute space; is it unimodal or multi-modal? Unimodal each
  - Characterize the overall distribution (including all examples); do you observe any correlation or other characteristics? quite strong positive correlation between the two attributes
  - Analyze the separation of a single class from all the other classes. Analyze the separation between pairs of classes. Blue is clearly separated from the two other; red and green only slightly overlap
  - If classes overlap characterize the extend to which they overlap.
  - If decision boundaries between classes can be inferred characterize those decision boundaries. Test using just petal length will mostly do a good job
  - Assess the difficulty of the classification based on your findings of looking at a set of scatter plots. Easy

Interpreting More Supervised Scatter Plots

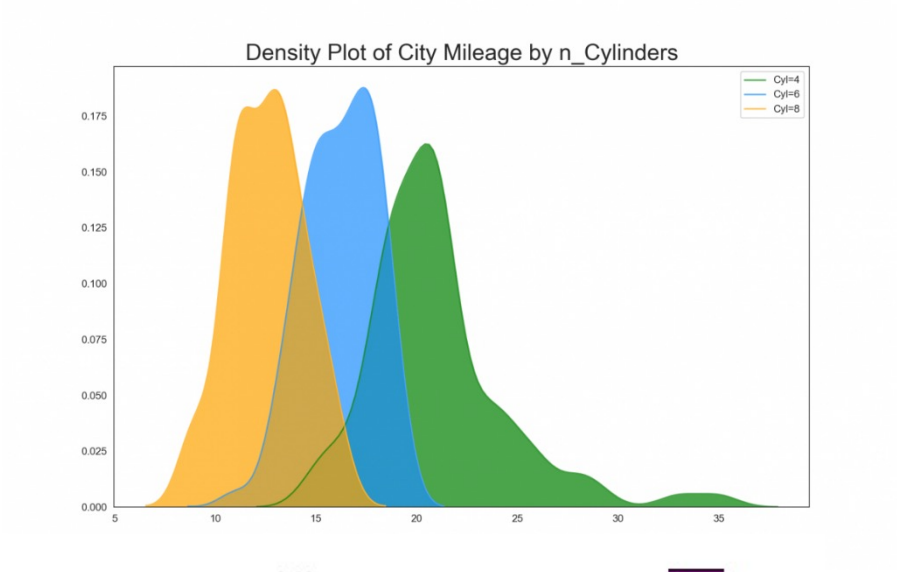
- Interpret the supervised scatter plot depicted below that consists of instances of 3 classes; moreover, assess the difficulty of separating instances of the 3 classes using attributes V1 and V7 based on the scatter plot!

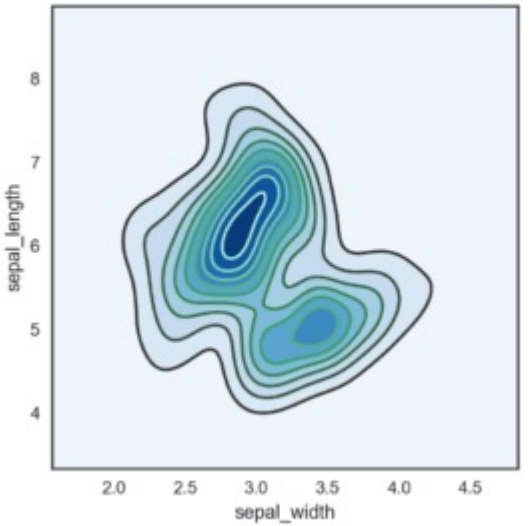


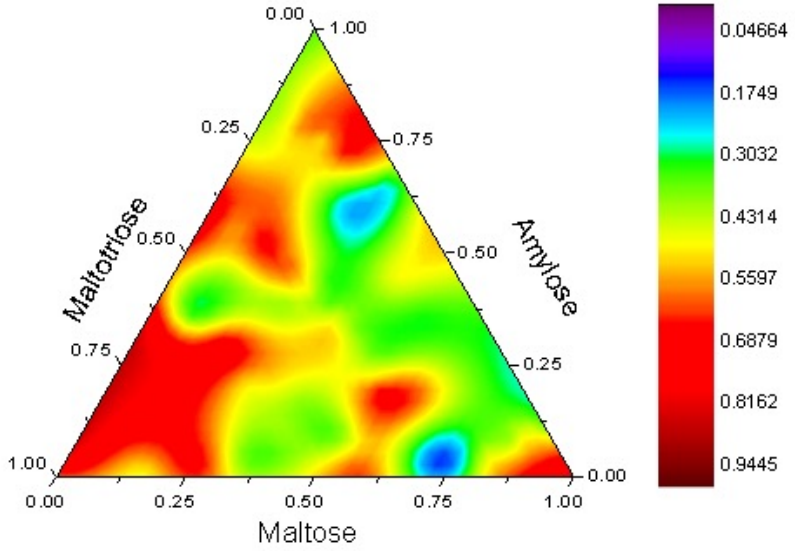
Interpreting More Supervised Scatter Plots

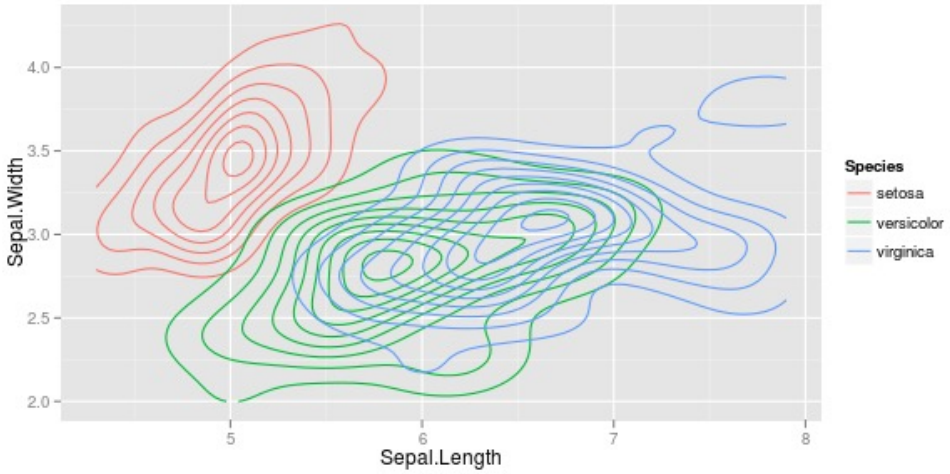
- Characterization of the location of the instances of each class in the attribute space: the pink class is mostly located in [11,13.3]x[1,3.8], the green class is mostly located in [12.3,14.2]x[0.3,1.1], the blue class is mostly located in [12.8, 14.8]x[2.1,3.8] in the V1XV7 Attribute space.
- The distribution of all three classes appears unimodal [2] with no major gaps in the data density [0.5].
- Attribute V7 is useful to separate the green class from the other 2 classes[1]; if V7 is less than 1.2 objects mostly belong to the green class[0.5].
- Attribute V1 is useful to separate the purple class from the blue class[1]; all the example whose V1 value is above 12.8 are blue[0.5].
- Additionally using attribute V7 requiring that instances whose attribute values for V1 that lie in [1.2,2.2] leads to an even clearer distinction between the two classes [0.5].
- Moreover, using the fact that the V1 value is a higher than 12.2 for instances of the green class, leads to a clearer separation of the green and purple class [0.5].
- By combining those rules [0.5], the classification task should not be too difficult as the examples are well separated although there are a few exceptions.[1]
- It is hard to see any patterns with respect to the relationship of V1 and V7; there definitely seems to be no linear relationship.

Visualization Techniques: Density Plots







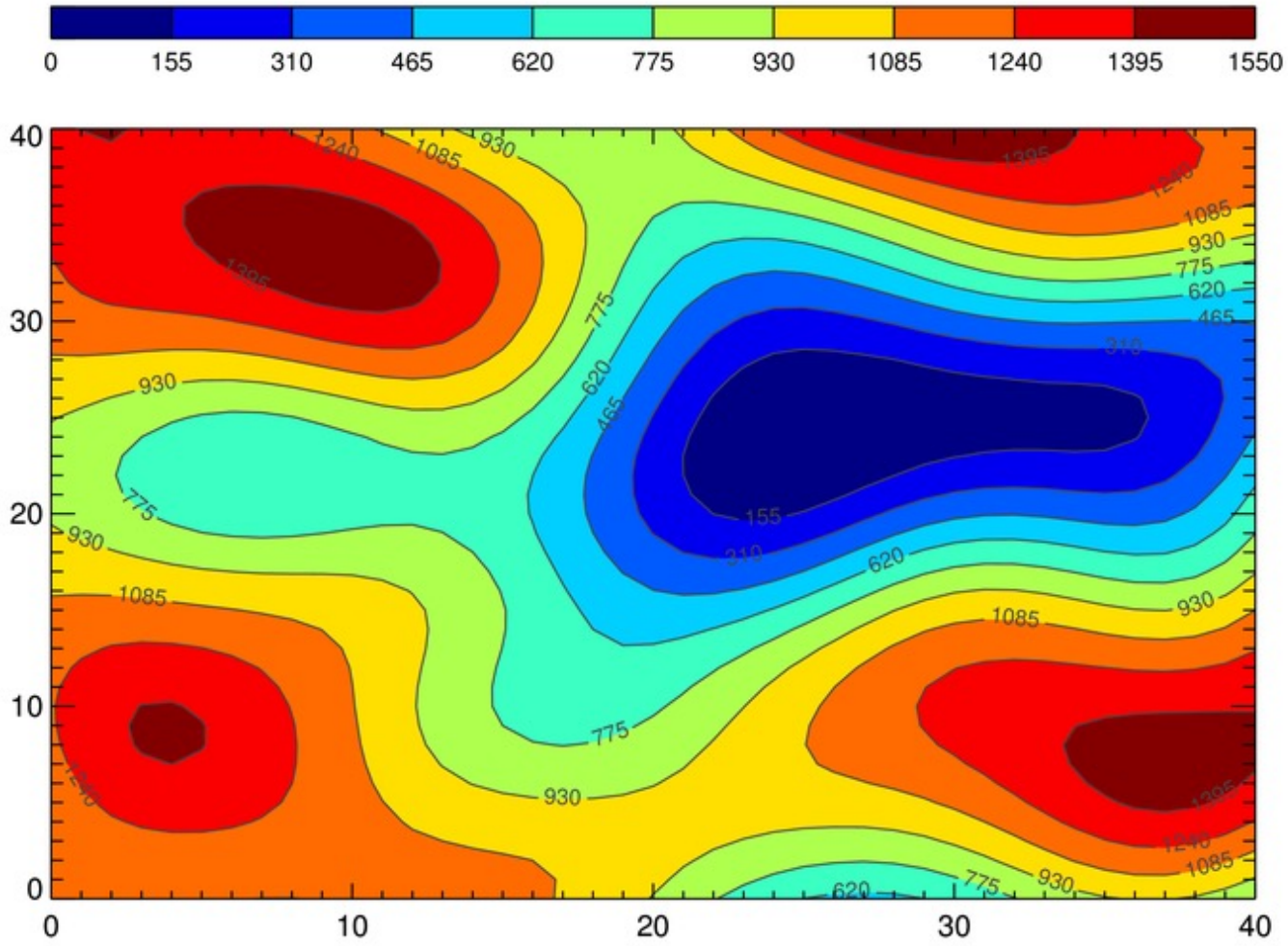


Visualization Techniques: Contour Plots

- 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.

https://plot.ly/r/contour-plots/

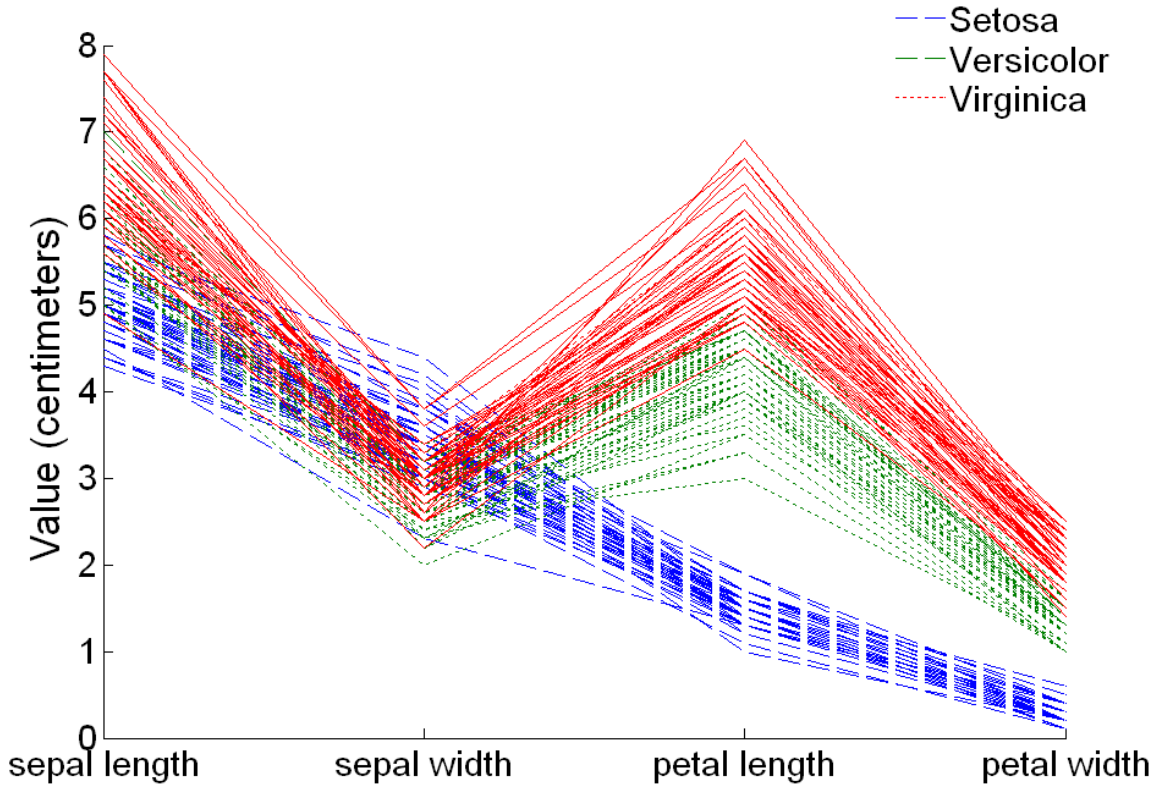
Contour Plots of Sea Surface Temperature

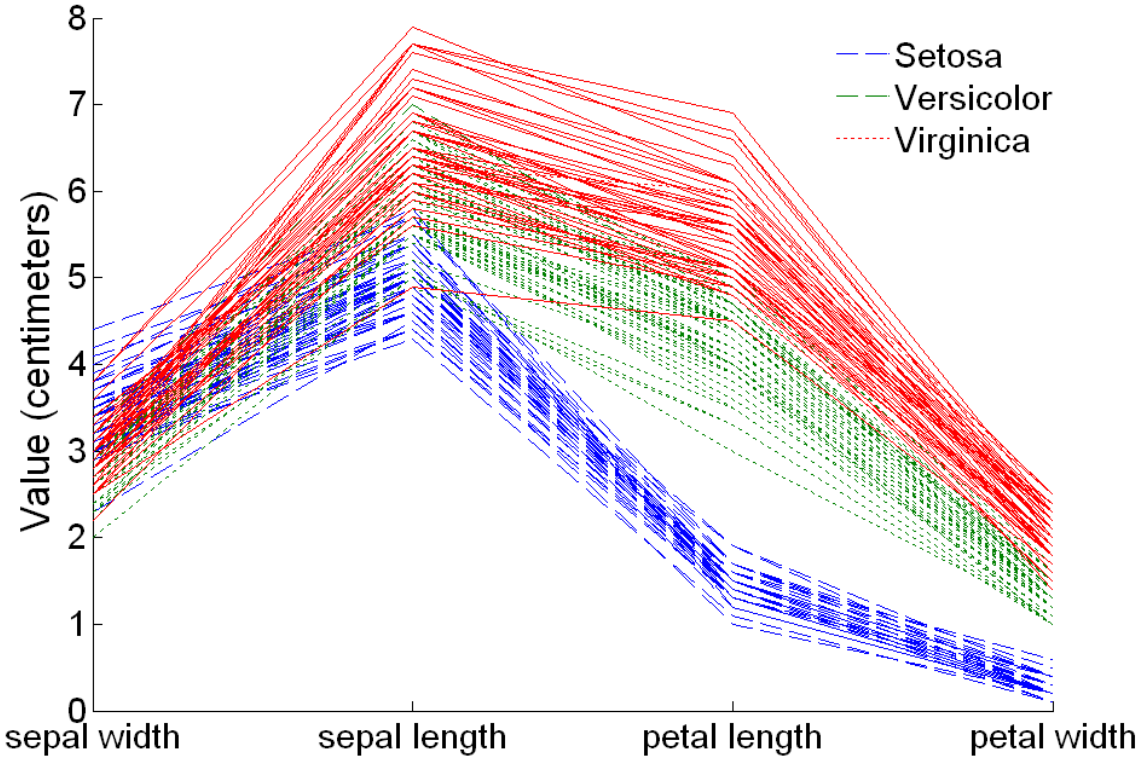


Visualization Techniques: Parallel Coordinates

- Parallel Coordinates
  - Used to plot the attribute values of high-dimensional data
  - Instead of using perpendicular axes, use a set of parallel axes
  - The attribute values of each object are plotted as a point on each corresponding coordinate axis and the points are connected by a line
  - Thus, each object is represented as a line
  - Often, the lines representing a distinct class of objects group together, at least for some attributes
  - Ordering of attributes is important in seeing such groupings

Visualization Techniques: Parallel Coordinates

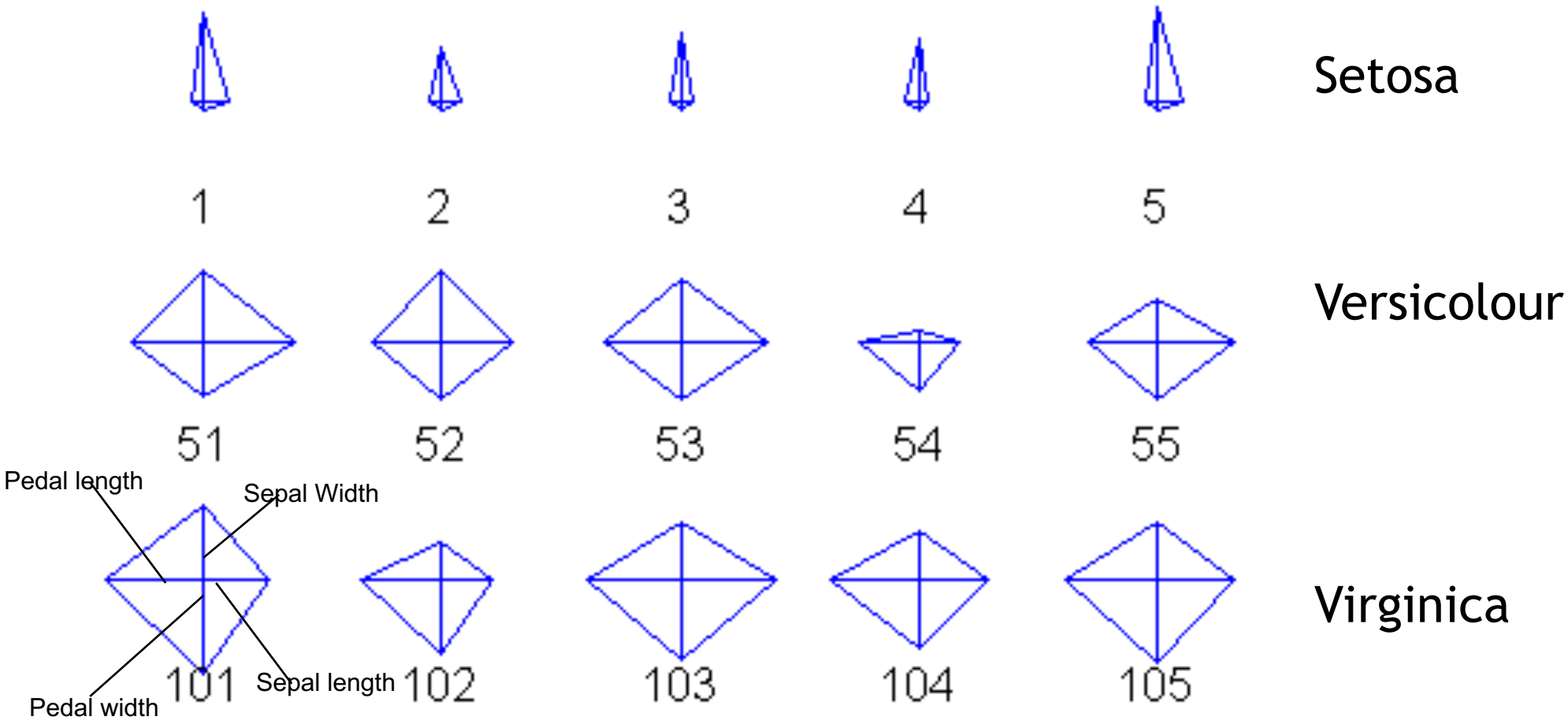




Other Visualization Techniques

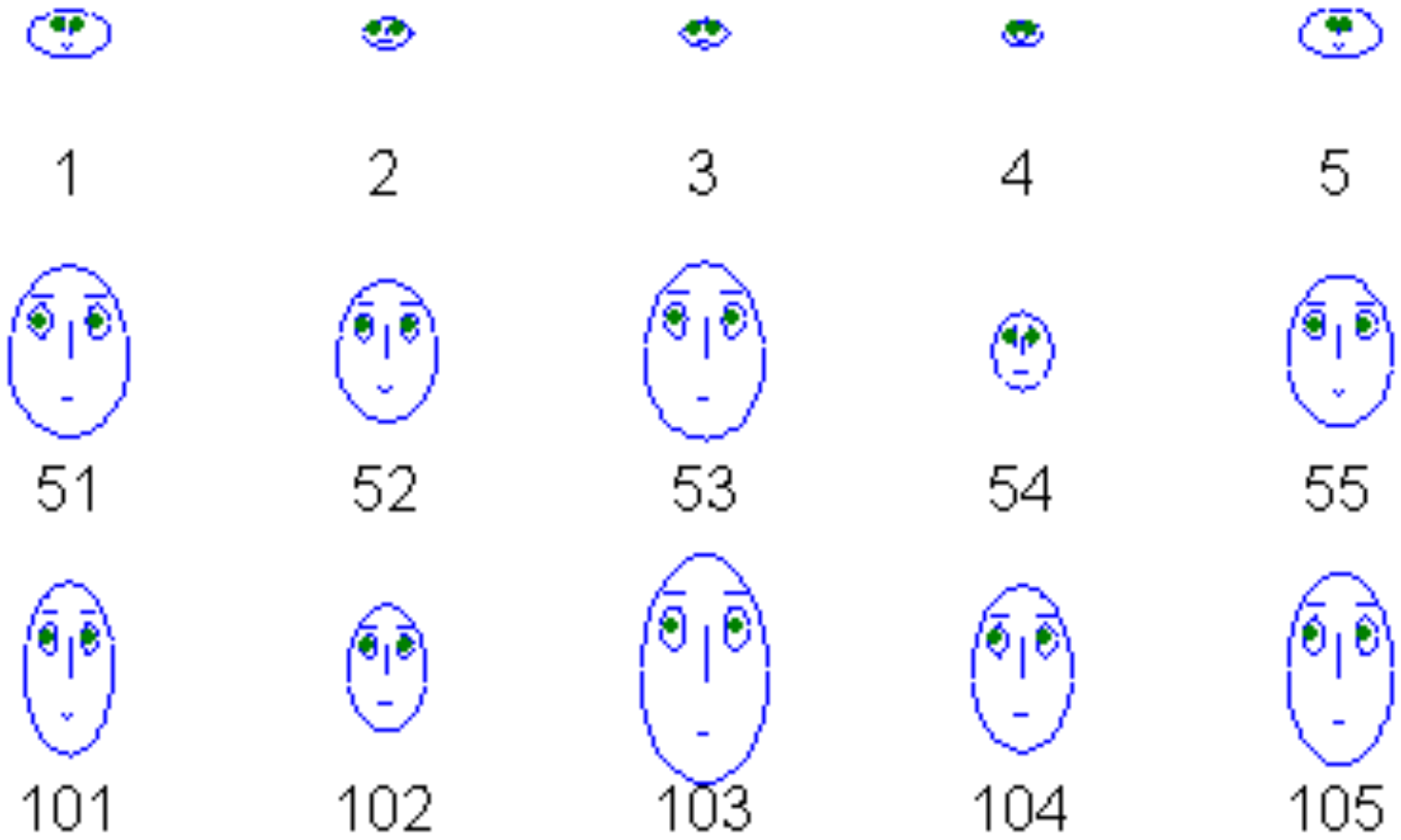
- Star Coordinate Plots
  - Similar to parallel coordinates, but axes radiate from a central point
  - The line connecting the values of an object is a polygon
- Chernoff Faces
  - Approach created by Herman Chernoff
  - This approach associates each attribute with a characteristic of a face
  - The values of each attribute determine the appearance of the corresponding facial characteristic
  - Each object becomes a separate face
  - Relies on human's ability to distinguish faces
  - http://people.cs.uchicago.edu/~wiseman/chernoff/
  - http://kspark.kaist.ac.kr/Human%20Engineering.files/Chernoff/Chernoff%20Faces.htm#

Star Plots for Iris Data



Chernoff Faces for Iris Data

Translation: sepal length→size of face; sepal width →forhead/jaw relative to arc-length;
Petal length→shape of forhead; pedal width→ shape of jaw; width of mouth→…; width between eyes→…

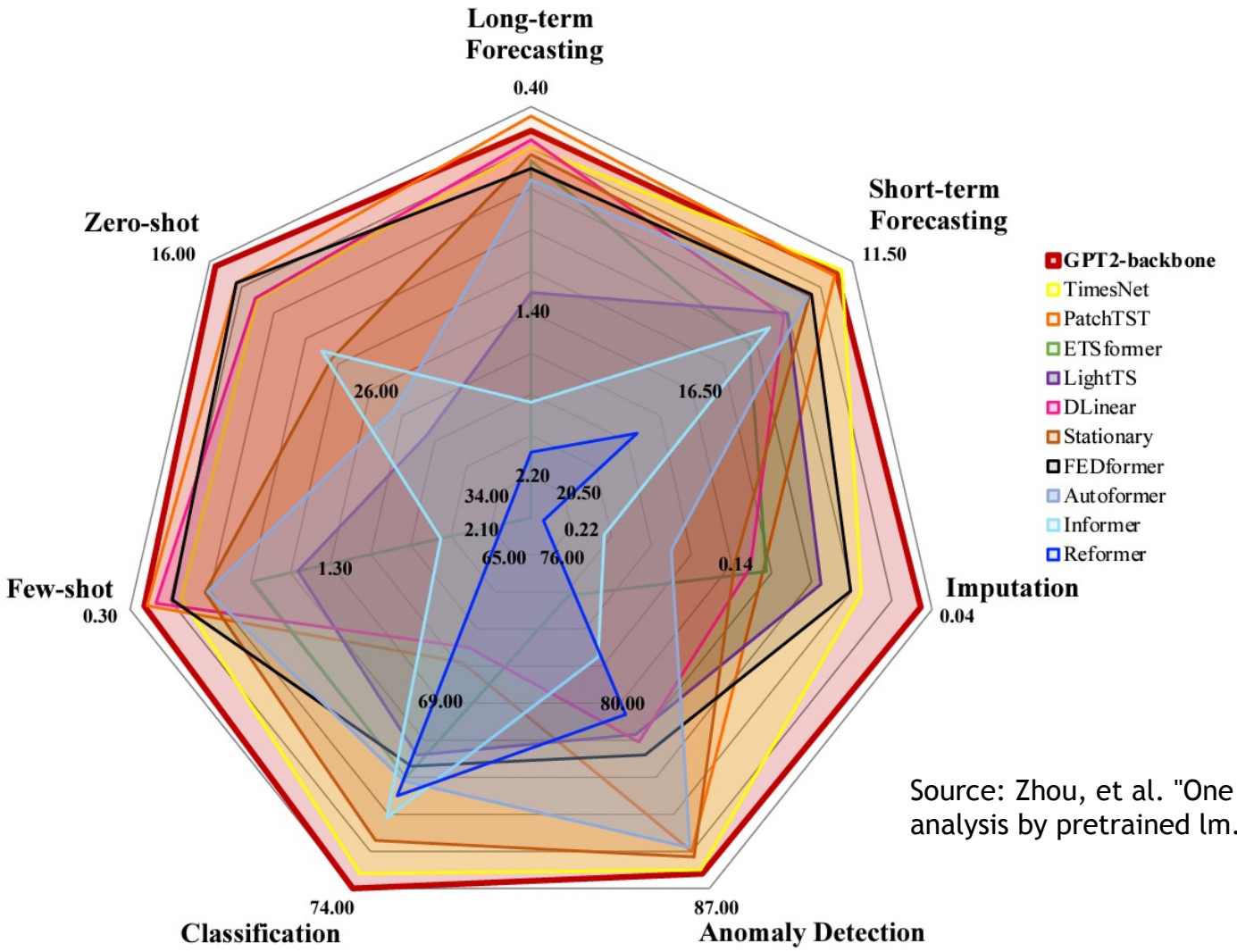


Setosa

Versicolour

Virginica

Radar Chart/Spiderweb Chart/Star Plot



Source: Zhou, et al. "One fits all: Power general time series analysis by pretrained lm." In NeurIPS, 2023.

Useful Background “Engineering St. Handbook”

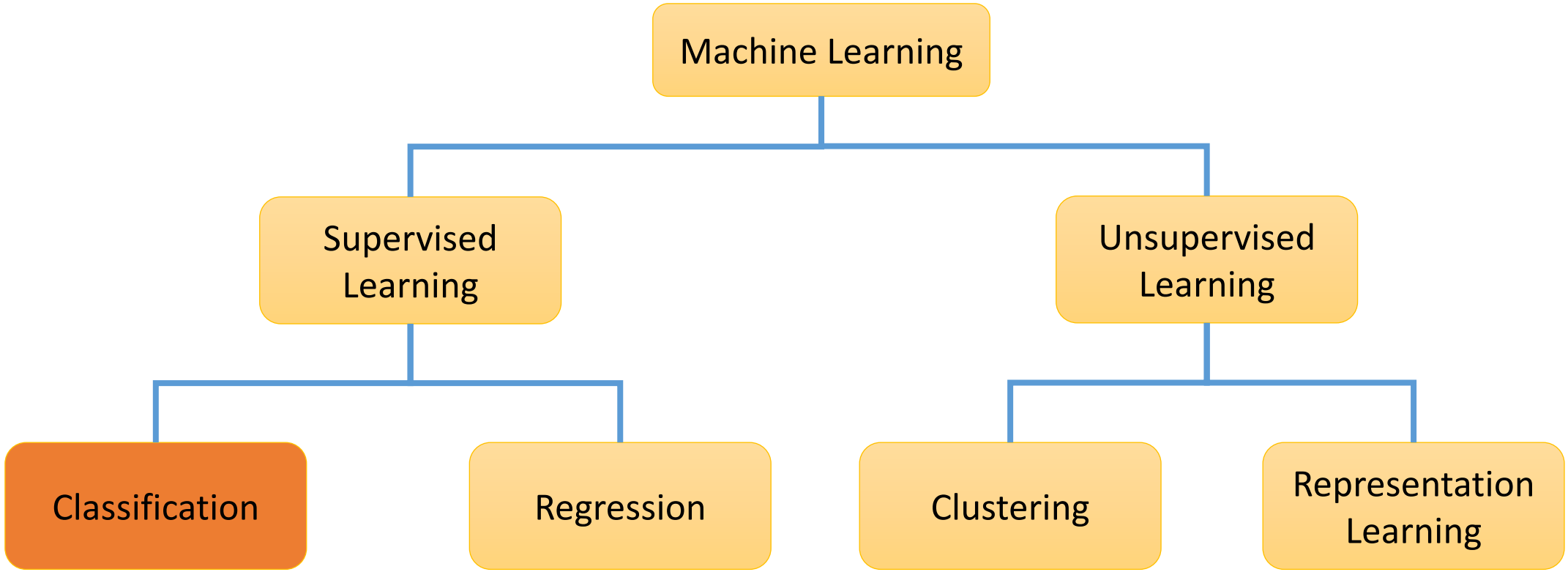
- http://www.itl.nist.gov/div898/handbook/eda/section1/eda15.htm (graphical techniques)
- http://www.itl.nist.gov/div898/handbook/eda/section3/eda35.htm (quantitative analysis)
- http://www.itl.nist.gov/div898/handbook/eda/section2/eda23.htm (testing assumptions)
- http://www.itl.nist.gov/div898/handbook/eda/section3/eda34.htm (survey graphical techniques)
- Remark: The material is very good if your focus is on prediction, hypothesis testing, clustering; however, providing good visualizations/statistics for classification problems is not discussed much

COSC 3337
Data Science I
Section 14623

Classification

Instructor: Jingchao Ni
Fall 2024

Machine Learning: Topics Taxonomy



Machine Learning: Topics Taxonomy

- Supervised Learning
  - A paradigm where input objects and a desired output target (e.g., labels) train a model. The model is a function that maps new data to the expected output target
- Unsupervised Learning
  - A framework where, in contrast to supervised learning, algorithms learn patterns (e.g., clusters) exclusively from unlabeled data
- Semi-Supervised Learning
  - A paradigm characterized by using a combination of a small amount of labeled data, followed by a large amount of unlabeled data. In other words, the desired output targets are provided only for a subset of the training data. The remaining data is unlabeled or imprecisely labeled

Classification: Definition

- 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
  - $f(X) = Y$
- 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

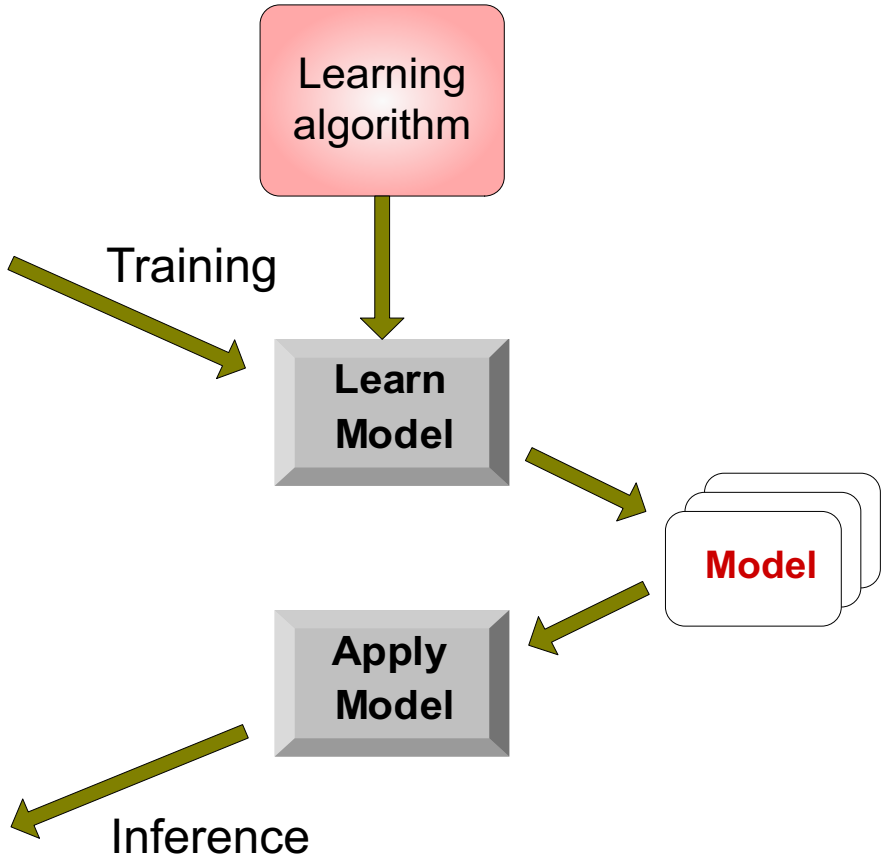
Classification: Illustration

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: Examples

- Disease Diagnosis
  - Predicting heart diseases using Electrocardiography (ECG) signals
- Fraud Detection
  - Classifying credit card transactions as legitimate or fraudulent using transaction features
- Sentiment Analysis
  - Classifying user reviews as positive or negative using review contents
- Face Recognition
  - Predicting identity of people using face images of system users

Evaluation of Classifiers

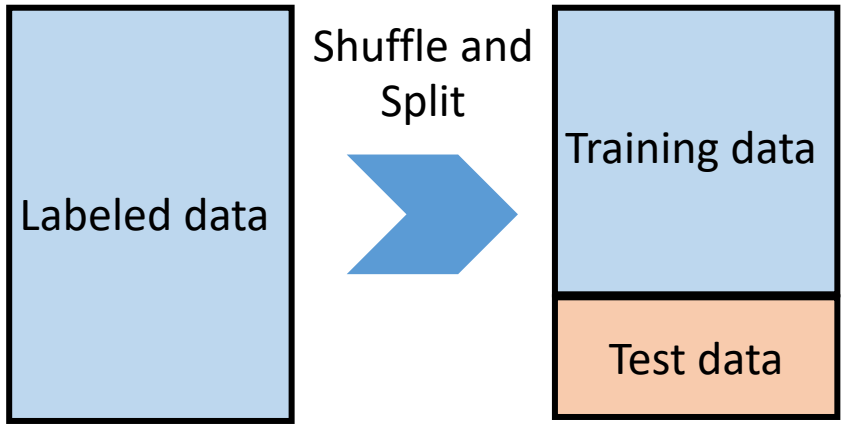
- What to evaluate regarding performance?
  - **Accuracy**: predicting class labels
  - Speed
    - time to construct the model (training time)
    - time to use the model (inference time)
  - Robustness: handling noise and missing values
  - Scalability: efficiency in disk-resident databases
  - Interpretability: understanding and insight provided by the model
  - Other: e.g., such as decision tree size, compactness of classification rules, generalization

Evaluation of Classifiers

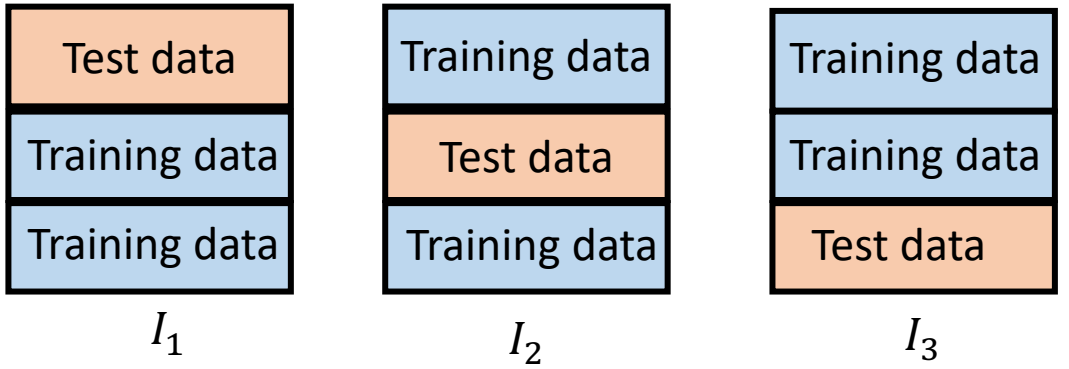
- After training, evaluate the accuracy of the classifier on unseen data
- Need separate, disjoint training and test data (all labeled)
  - Training data: for training the classifier (model construction)
  - Test data: for evaluating the trained classifier

Evaluation of Classifiers: Approaches

- Train-and-Test split
  - Shuffle and partition set X into two (disjoint) subsets: Training data and Test data
  - not recommended for small X

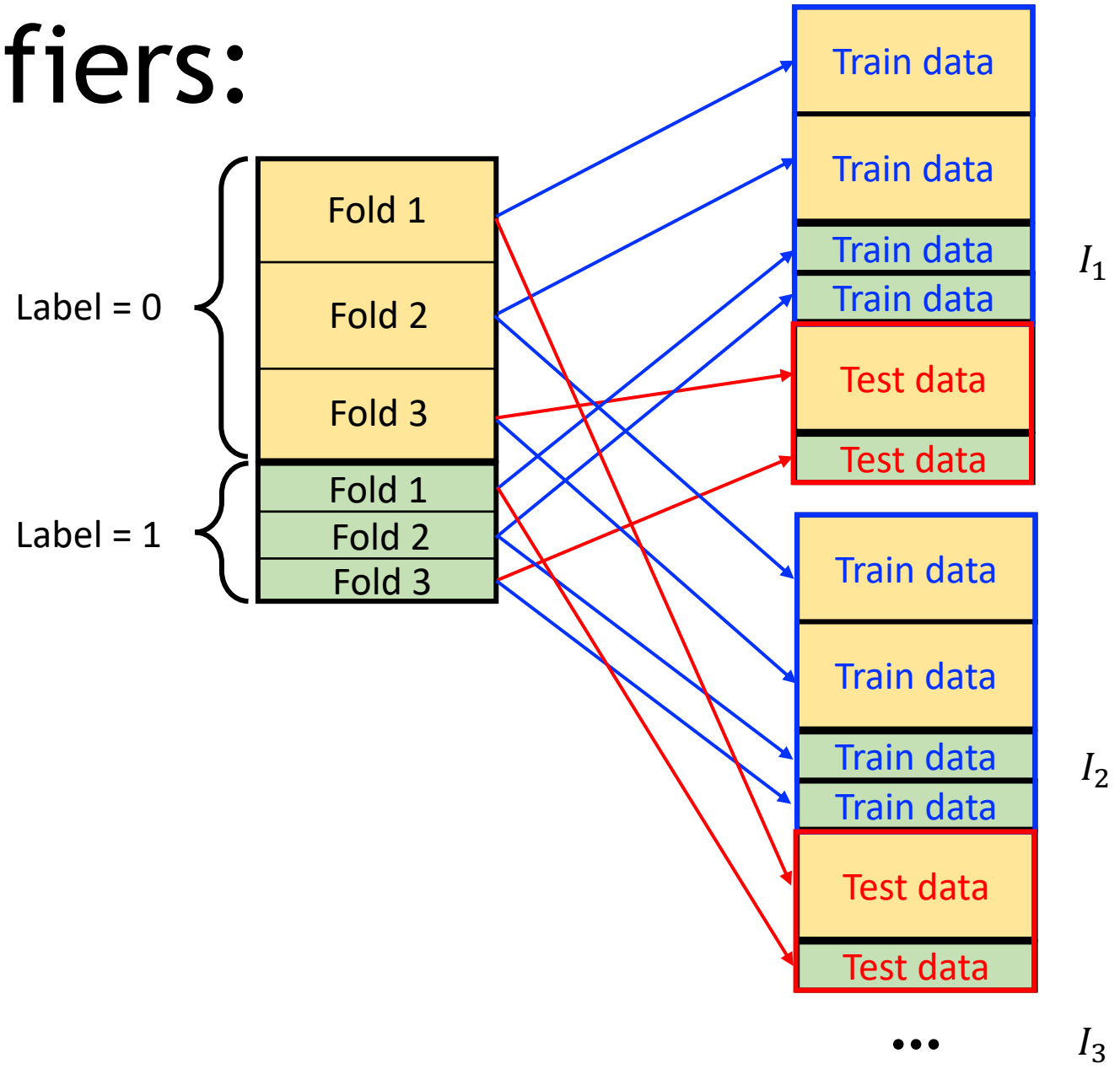


- K-fold Cross Validation
  - partition set D into k same size subsets
  - train K different classifiers
    - K-1 subsets for training
    - 1 subset for test
  - average the evaluation results of the K classifiers
  - appropriate also for small D



Evaluation of Classifiers: Approaches

- Stratified Cross Validation
  - Instead of randomly shuffling and splitting the entire dataset into K folds
  - Perform random shuffle and split in each class and combine the splits for each fold



Evaluation of Classifiers: Approaches

- Split Data with Sklearn

```
>>> import numpy as np
>>> from sklearn.model_selection import StratifiedShuffleSplit
>>> X = np.array([[1, 2], [3, 4], [1, 2], [3, 4], [1, 2], [3, 4]])
>>> y = np.array([0, 0, 0, 1, 1, 1])
>>> sss = StratifiedShuffleSplit(n_splits=5, test_size=0.5, random_state=0)
>>> sss.get_n_splits(X, y)
5
>>> print(sss)
StratifiedShuffleSplit(n_splits=5, random_state=0, ...)
>>> for i, (train_index, test_index) in enumerate(sss.split(X, y)):
... print(f"Fold {i}:")
... print(f" Train: index={train_index}")
... print(f" Test: index={test_index}")
Fold 0: Train: index=[5 2 3] Test: index=[4 1 0]
Fold 1: Train: index=[5 1 4] Test: index=[0 2 3]
Fold 2: Train: index=[5 0 2] Test: index=[4 3 1]
Fold 3: Train: index=[4 1 0] Test: index=[2 3 5]
Fold 4: Train: index=[0 5 1] Test: index=[3 4 2]
```

Classification Accuracy

- Let f be a classifier, $X_{tr} \subseteq X$ the training data, $X_{te} \subseteq X$ the test data.
- $L(x) = y$: actual class of object x
- $f(x) = \hat{y}$: predicted class of object x

- Classification accuracy of f on X_{te} :

$$ACC(f) = \frac{|\{x \in X_{te} | f(x) = L(x)\}|}{|X_{te}|}$$

- Classification error

$$ERR(f) = \frac{|\{x \in X_{te} | f(x) \neq C(x)\}|}{|X_{te}|}$$

- not appropriate if minority class is most important

Confusion Matrix

- For Binary Classification

	PREDICTED CLASS		
ACTUAL CLASS		Class=Yes	Class=No
	Class=Yes	a (TP)	b (FN)
	Class=No	c (FP)	d (TN)

a: TP (true positive)

b: FN (false negative)

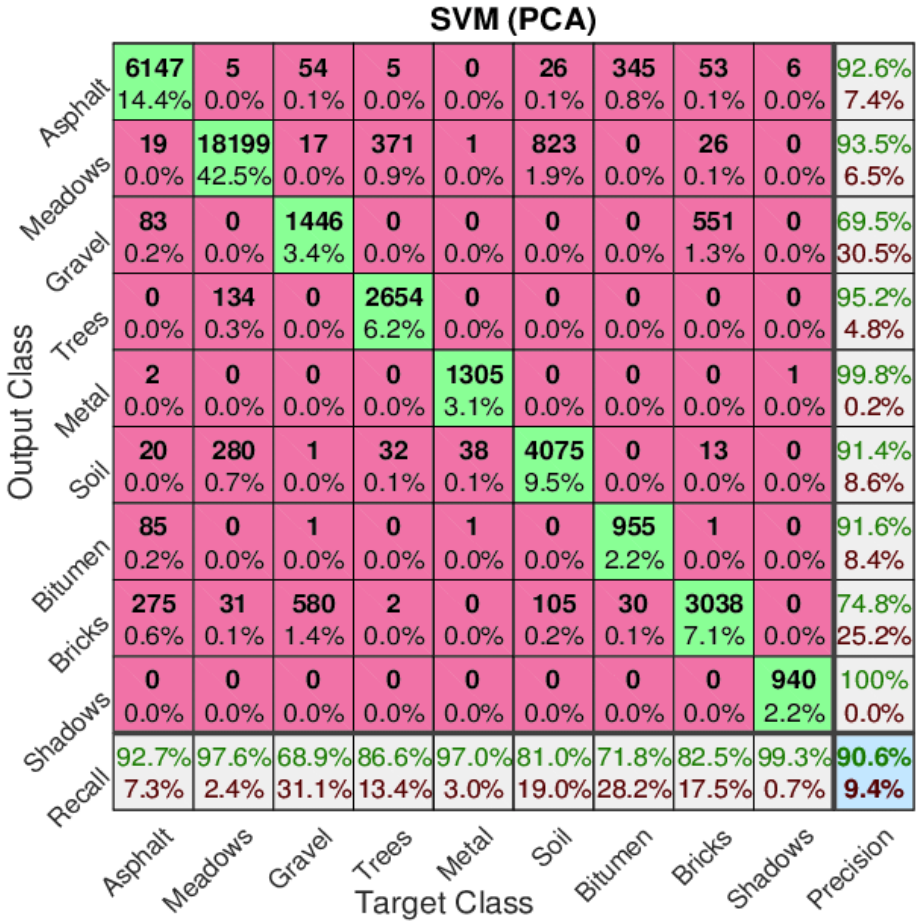
c: FP (false positive)

d: TN (true negative)

$$\text{Accuracy} = \frac{a+d}{a+b+c+d} = \frac{TP+TN}{TP+TN+FP+FN}$$

Confusion Matrix

- For Multi-Class Classification



COSC 3337
Data Science I
Section 14623

Classification

Instructor: Jingchao Ni
Fall 2024

Precision, Recall, F1 Score

- We define the following two measures w.r.t. the given target class

		PREDICTED CLASS	
		Class=Yes	Class=No
ACTUAL CLASS	Class=Yes	a (TP)	b (FN)
	Class=No	c (FP)	d (TN)

$$Precision = \frac{|TP|}{\#\text{ Positive Predictions}} = \frac{|TP|}{|TP| + |FP|}$$

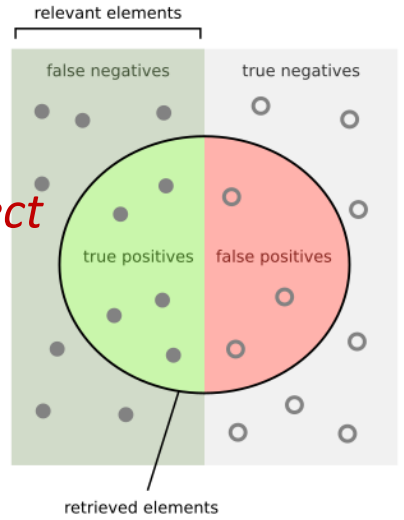
$$Recall = \frac{|TP|}{\#\text{Actual positives}} = \frac{|TP|}{|TP| + |FN|}$$

Predict everything as positive

- *Recall=1, Precision is low*

Predict only 1 positive and it's correct

- *Precision=1, Recall is low*

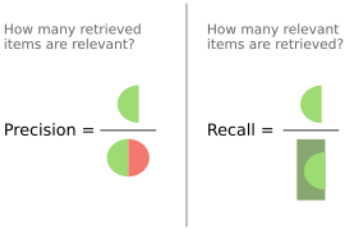


- There is a trade-off between precision and recall

$$F1\ Score = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

$$\mathrm{F}_\beta\ Score = \frac{(1 + \beta^2) \cdot \text{Precision} \cdot \text{Recall}}{\beta^2 \text{Precision} + \text{Recall}}$$

Harmonic Mean



Precision, Recall, F1 Score

- Precision: The classifier predict class C; how often is this decision correct?
- Recall: The correct class is C; how often does the classifier predict class C correctly?
- F-measure somewhat combines recall and precision using harmonic mean

Exercise

- Given a classifier f and a test set
  - $X_{test} = \{(x_1, 0), (x_2, 0), (x_3, 1), (x_4, 1)\}$
  - The prediction of f is

	x_1	x_2	x_3	x_4
Actual label	0	0	1	1
prediction	0	1	1	1

- Calculate the classification accuracy, classification error, Precision, Recall, and F1 scores

Exercise

- Given a classifier f and a test set
  - $X_{test} = \{(x_1, 0), (x_2, 0), (x_3, 1), (x_4, 1)\}$
  - The prediction of f is

	x_1	x_2	x_3	x_4
Actual label	0	0	1	1
prediction	0	1	1	1

- Classification accuracy: 0.75
  - x_1, x_3, x_4 were correctly classified
- Classification error: 1-0.75=0.25
- Precision: 2/3=67%
- Recall: 100%
- F1: 0.8

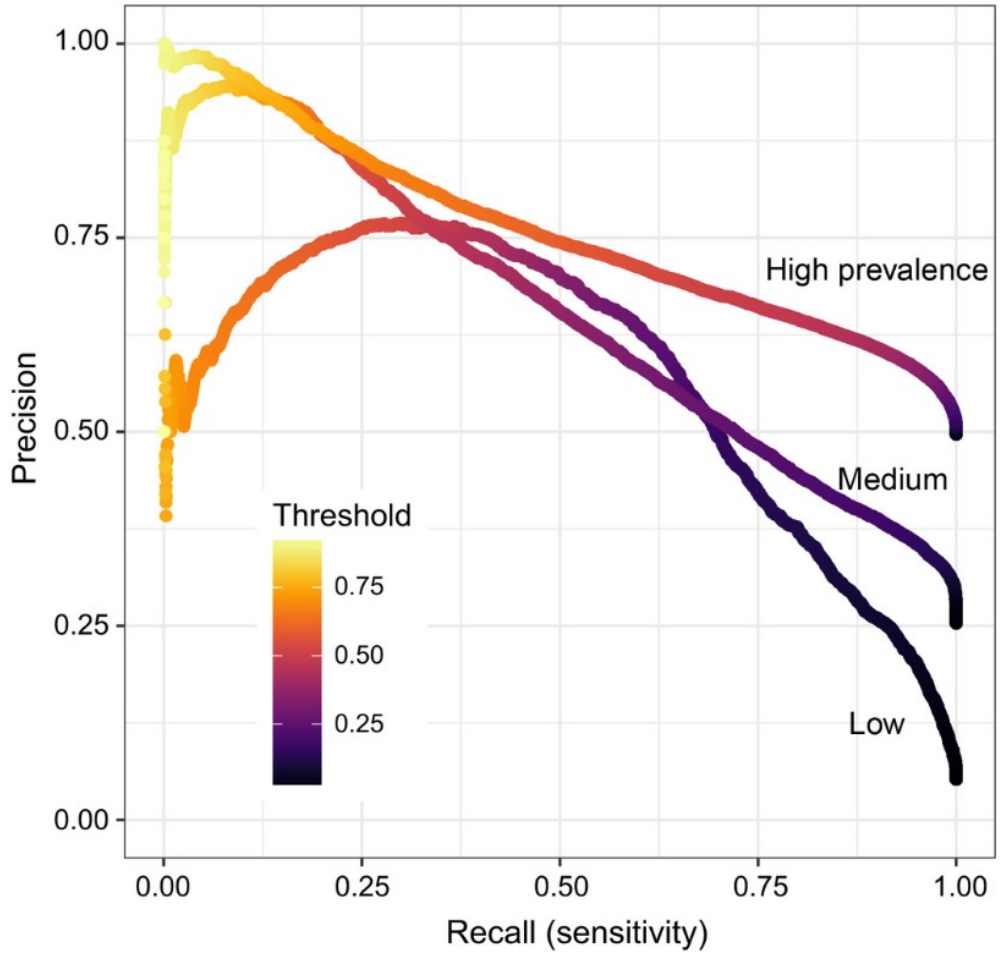
	Predicted as positive	Predicted as negative
Actual positive	x_3, x_4	NULL
Actual negative	x_2	x_1

Precision, Recall, F1 Score

- Precision-Recall (PR) Curve and Area under the PR Curve (AUPRC)

Target Class (Sorted)

		Class 1	Class 2
TP, FP, FN, Precision, Recall	Threshold=0.91 →	0.91	0.09
		0.85	0.15
		0.83	0.17
TP, FP, FN, Precision, Recall	Threshold=0.8 →	0.82	0.08
		0.62	0.38
TP, FP, FN, Precision, Recall	Threshold=0.5 →	0.53	0.47
		0.35	0.65
		…	…



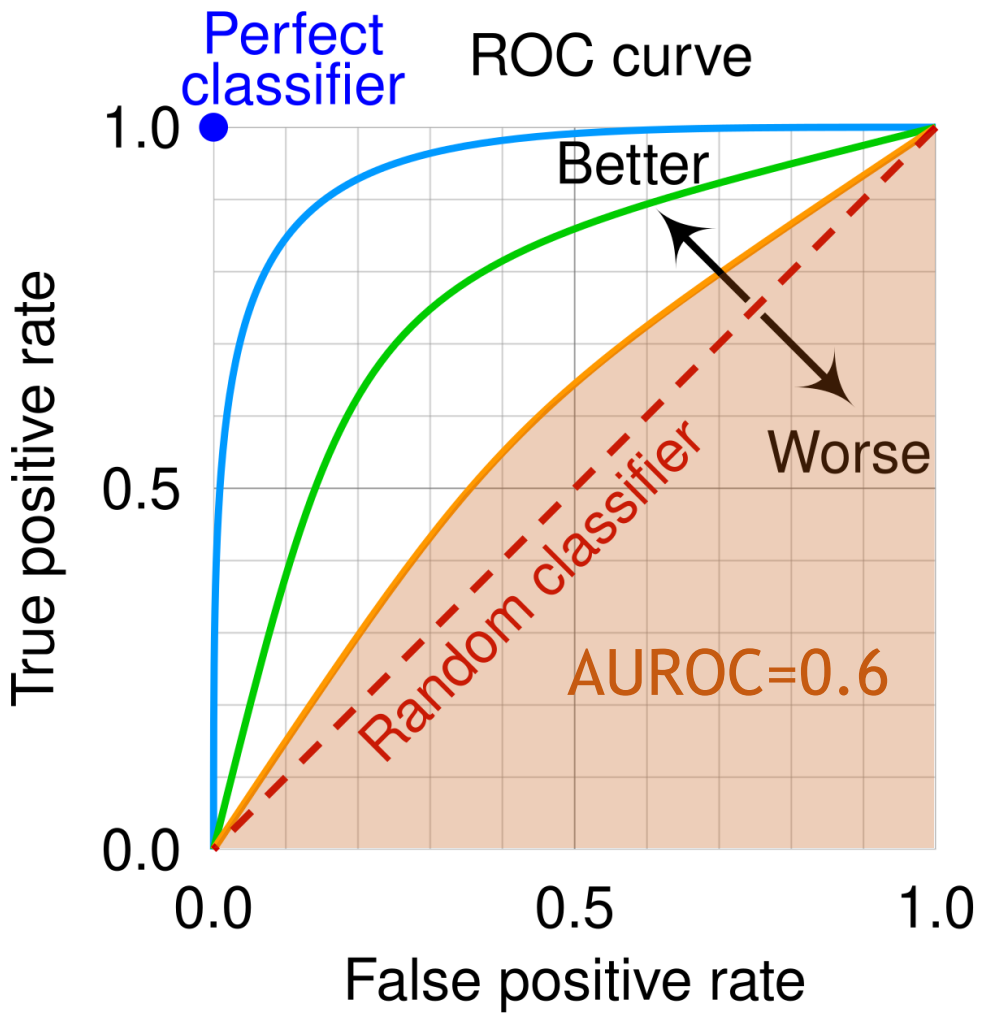
Receiver Operating Characteristic Curve

- ROC and AUROC
  - TPR: True Positive Rate
  - FPR: False Positive Rate
  - ROC: (TPR, FPR) at different thresholds
  - AUROC: between 0.5 and 1

$$TPR = Recall = \frac{|TP|}{|TP| + |FN|}$$

$$FPR = \frac{|FP|}{|FP| + |TN|}$$

AUROC is not recommended for imbalanced classes, it is always high



Confusion Matrix and Measurements

		Predicted condition		Sources: [8][9] [10][11][12][13][14][15] view · talk · edit	
	Total population $= P + N$	**Predicted Positive (PP)**	**Predicted Negative (PN)**	Informedness, bookmaker informedness (BM) $= TPR + TNR - 1$	Prevalence threshold (PT) $= \frac{\sqrt{TPR \times FPR} - FPR}{TPR - FPR}$
Actual condition	**Positive (P)** [a]	**True positive** (TP), hit[b]	**False negative** (FN), miss, underestimation	True positive rate (TPR), recall, sensitivity (SEN), probability of detection, hit rate, power $= \frac{TP}{P} = 1 - FNR$	False negative rate (FNR), miss rate type II error [c] $= \frac{FN}{P} = 1 - TPR$
	Negative (N)[d]	**False positive** (FP), false alarm, overestimation	**True negative** (TN), correct rejection[e]	False positive rate (FPR), probability of false alarm, fall-out type I error [f] $= \frac{FP}{N} = 1 - TNR$	True negative rate (TNR), specificity (SPC), selectivity $= \frac{TN}{N} = 1 - FPR$
	Prevalence $= \frac{P}{P + N}$	Positive predictive value (PPV), precision $= \frac{TP}{PP} = 1 - FDR$	False omission rate (FOR) $= \frac{FN}{PN} = 1 - NPV$	Positive likelihood ratio (LR+) $= \frac{TPR}{FPR}$	Negative likelihood ratio (LR−) $= \frac{FNR}{TNR}$
	Accuracy (ACC) $= \frac{TP + TN}{P + N}$	False discovery rate (FDR) $= \frac{FP}{PP} = 1 - PPV$	Negative predictive value (NPV) $= \frac{TN}{PN} = 1 - FOR$	Markedness (MK), deltaP (Δp) $= PPV + NPV - 1$	Diagnostic odds ratio (DOR) $= \frac{LR+}{LR-}$
	Balanced accuracy (BA) $= \frac{TPR + TNR}{2}$	F_1 score $= \frac{2\,PPV \times TPR}{PPV + TPR} = \frac{2\,TP}{2\,TP + FP + FN}$	Fowlkes–Mallows index (FM) $= \sqrt{PPV \times TPR}$	Matthews correlation coefficient (MCC) $= \sqrt{TPR \times TNR \times PPV \times NPV} - \sqrt{FNR \times FPR \times FOR \times FDR}$	Threat score (TS), critical success index (CSI), Jaccard index $= \frac{TP}{TP + FN + FP}$

a. ^ the number of real positive cases in the data
b. ^ A test result that correctly indicates the presence of a condition or characteristic
c. ^ Type II error: A test result which wrongly indicates that a particular condition or attribute is absent
d. ^ the number of real negative cases in the data
e. ^ A test result that correctly indicates the absence of a condition or characteristic
f. ^ Type I error: A test result which wrongly indicates that a particular condition or attribute is present

https://en.wikipedia.org/wiki/Receiver_operating_characteristic

Evaluation of Multi-Class Classification

- Macro-F1 Score and Micro-F1 Score
  - Calculate Precision for N Classes: $P_1, P_2, \ldots, P_N$
  - Calculate Recall for N Classes: $R_1, R_2, \ldots, R_N$
  - Calculate F1 for N Classes: $F_1, F_2, \ldots, F_N$

$$Macro\_F1 = \frac{1}{N}\sum_{i=1}^{N} F_i \qquad Weighted\_F1 = \frac{\sum_{i=1}^{N}(F_i \times w_i)}{\sum_{i=1}^{N} w_i}$$

  - Count $TP_1, TP_2, \ldots, TP_N,\ FP_1, FP_2, \ldots, FP_N,\ FN_1, FN_2, \ldots, FN_N$

$$\bar{P} = \frac{\sum_{i=1}^{N} TP_i}{\sum_{i=1}^{N} TP_i + \sum_{i=1}^{N} FP_i} \qquad \bar{R} = \frac{\sum_{i=1}^{N} TP_i}{\sum_{i=1}^{N} TP_i + \sum_{i=1}^{N} FN_i}$$

$$Micro_{F1} = \frac{2\bar{P}\bar{R}}{\bar{P} + \bar{R}} = \frac{2\sum_{i=1}^{N} TP_i}{2\sum_{i=1}^{N} TP_i + \sum_{i=1}^{N} FP_i + \sum_{i=1}^{N} FN_i}$$

Evaluation of Classification

```
import numpy as np

def pre_rec_f1(y_pred, y):
    eps = np.finfo(float).eps
    num_cls = len(np.unique(y))
    tp_sum, fp_sum, fn_sum, f1_sum = 0, 0, 0, 0
    tp, fp, fn, pre, rec, f1 = [], [], [], [], [], []
    for i in range(num_cls):
        y_pred_i = y_pred == i
        y_i = y == i
        tp = np.logical_and(y_pred_i, y_i).sum()
        fp = np.logical_and(y_pred_i, (~y_i)).sum()
        fn = np.logical_and((~y_pred_i), y_i).sum()
        tp_sum += tp
        fp_sum += fp
        fn_sum += fn
        f1 = (2 * tp) / (2 * tp + fp + fn + eps)
        f1_sum += f1
        pre.append(tp / (tp + fp + eps))
        rec.append(tp / (tp + fn + eps))
        tp.append(tp)
        fp.append(fp)
        fn.append(fn)
        f1.append(f1)

    macrof1 = f1_sum / (num_cls + eps)
    microf1 = (2 * tp_sum) / (2 * tp_sum + fp_sum + fn_sum + eps)
    return macrof1, microf1, tp, fp, fn, pre, rec, f1
```

TPR, FPR, AUC

```
>>> import numpy as np
>>> from sklearn import metrics
>>> y = np.array([1, 1, 2, 2])
>>> y_pred = np.array([0.1, 0.4, 0.35, 0.8])
>>> fpr, tpr, thresholds = metrics.roc_curve(y,
y_pred, pos_label=1)
>>> fpr
array([0. , 0. , 0.5, 0.5, 1. ])
>>> tpr
array([0. , 0.5, 0.5, 1. , 1. ])
>>> thresholds
array([ inf, 0.8 , 0.4 , 0.35, 0.1 ])
>>> auc = metrics.auc(fpr, tpr)
```

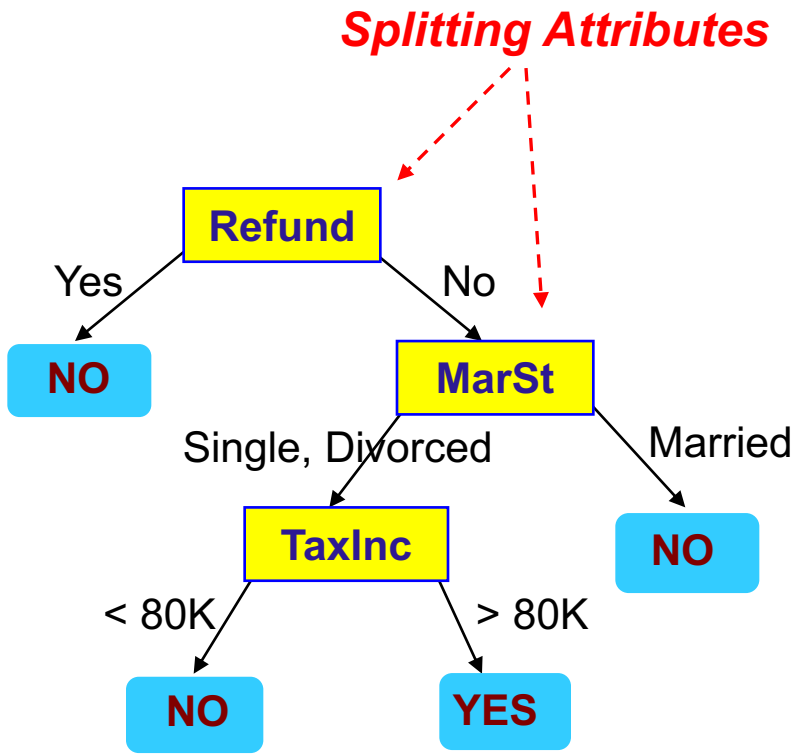
Classification Methods

- Decision Tree based Methods
- Rule-based Methods
- Memory based reasoning, instance-based learning, kNN
- Neural Networks
- Naïve Bayes and Bayesian Belief Networks
- Support Vector Machines
- Ensemble Methods

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

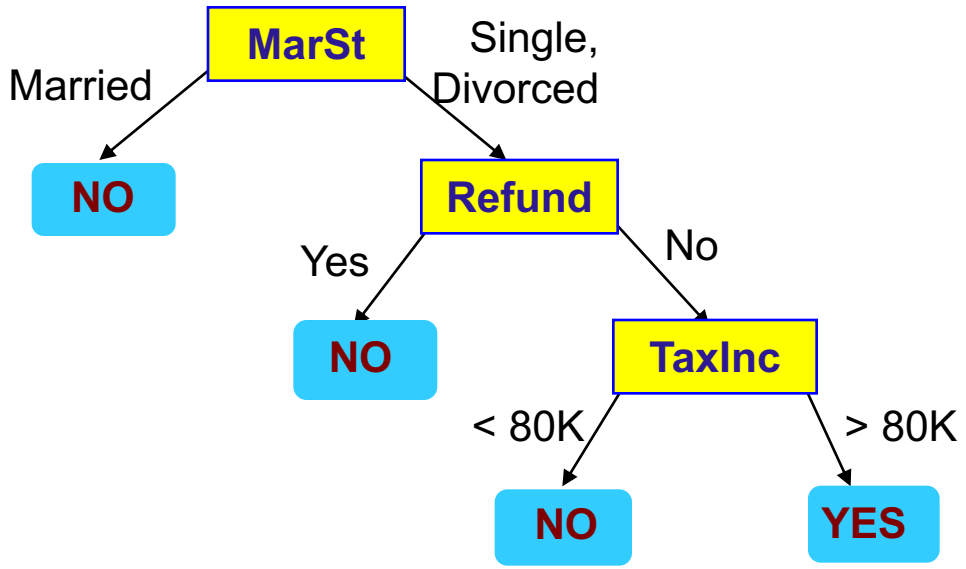


Model: Decision Tree

Rokach, Lior; Maimon, O. (2014). Data mining with decision trees: theory and applications, 2nd Edition. World Scientific Pub Co Inc.

Another Example of 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



There could be more than one tree that fits the same data!

Decision Trees

- A decision tree is a tree with the following properties
  - An inner node represents an attribute
  - An edge represents a test on the attribute of the parent node
    - Labeled with the possible values of the attribute
  - A leaf represents a class
- Construction of a decision tree (training)
  - Based on the training data
  - Top-Down strategy
- Inference
  - Traversal of the decision tree from the root to one of the leaves
  - Unique path
  - Assignment of the object to class of the resulting leaf

Decision Trees

- General Idea of Tree Construction
  - Initially, all training data records belong to the root
  - Next attribute is selected and split
  - Training data records are partitioned according to the chosen split
  - Method is applied recursively to each partition
- Termination conditions
  - No more split attributes
  - All (most) training data records of the node belong to the same class
    - No longer adds value to the predictions
  - No sample left
- Many Algorithms
  - Hunt's Algorithm (one of the earliest)
  - CART (Classification And Regression Tree)
  - ID3, C4.5
  - SLIQ, SPRINT

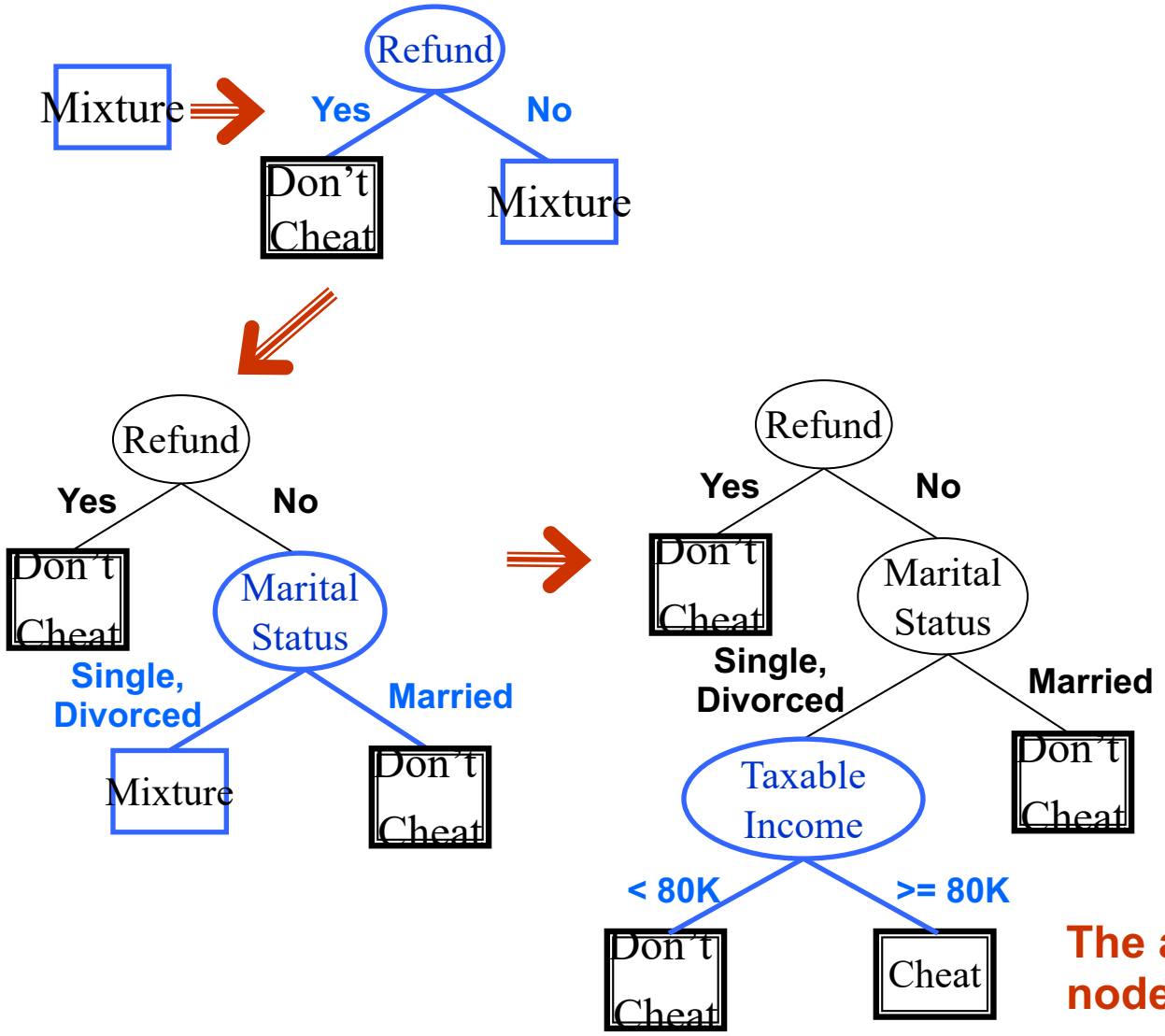
General Structure of Hunt's Algorithm

1. Create a node t, suppose the training dataset when reach node t is D_t, Initially, D_t is the entire training set D
2. If D_t contains records that belong to the same class y_t, then t is a leaf node labeled as y_t
3. If D_t is not an empty set but Att_{list} is empty (there is no more test attributes), then t is a leaf node labeled by the label of the majority records in this node
4. If D_t contains records that belong to more than one class and Att_{list} is not empty, use *Best_Split* method to choose next best attribute from Att_{list} and remove it from Att_{list}, use the attribute and its condition as next test condition
5. Repeat steps 2, 3, 4 until all the records in the subset belong to the same class

Remark

- Greedy Algorithms
  - Grow a decision tree by making a series of locally optimum decisions on which attributes to use for partitioning the data
  - Top-down approach is by far the most common strategy
- Finding The Optimal Decision Tree is NP-Hard
  - Exponentially many decision trees can be constructed from a given set of attributes
- Fundamental Problems
  - Determine how to split the records
    - How to specify the attribute test condition?
    - How to determine the best split?
  - Determine when to stop splitting

Illustration of 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

The algorithm stopped because (1) all leaf nodes are pure, (2) no attributes left

Side Discussion “Greedy Algorithms”

- Makes locally optimal choices at each stage
- Fast and therefore attractive to solve NP-hard and other problems with high complexity. Later decisions are made in the context of decision selected early dramatically reducing the size of the search space
- They do not backtrack: if they make a bad decision (based on local criteria), they never revise the decision
- They are not guaranteed to find the optimal solution(s), and sometimes can get deceived and find really bad solutions
- In spite of what is said above, a lot successful and popular algorithms in Computer Science are greedy algorithms
- Greedy algorithms are particularly popular in AI and Operations Research
- See also: http://en.wikipedia.org/wiki/Greedy_algorithm
- Popular Greedy Algorithms: Decision Tree Induction

COSC 3337
Data Science I
Section 14623

Classification

Instructor: Jingchao Ni
Fall 2024

Finding the Best Split

- Suppose there are M features
- For each feature f_i (i = 1, …, M):
  - Determine attribute test conditions
  - For each possible attribute test condition of f_i:
    - Determine split values
    - For each split value:
      - Calculate a “goodness” score of the split
- Select the split with the best “goodness” score

Remark

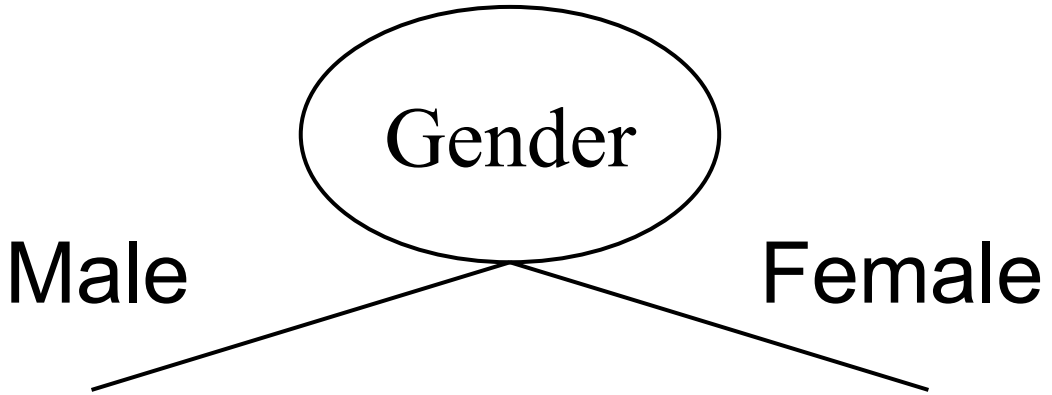
- Greedy Algorithms
  - Grow a decision tree by making a series of locally optimum decisions on which attributes to use for partitioning the data
  - Top-down approach is by far the most common strategy
- Finding The Optimal Decision Tree is NP-Hard
  - Exponentially many decision trees can be constructed from a given set of attributes
- Fundamental Problems
  - 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 A Specify Test Condition?

- Depends on attribute types
  - Binary
  - Non-Binary
    - Nominal
      - categorical attributes that do not have an inherent order or ranking
    - Ordinal
      - categorical attributes that have a natural order or rank to their categories
    - Continuous
- Depends on number of ways to split
  - 2-way split
  - Multi-way split

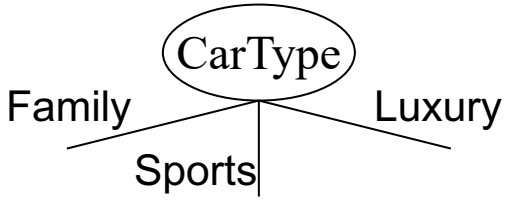
Splitting Based on Binary Attributes

- The test condition for a binary attribute is simple because it only generates two potential outcomes

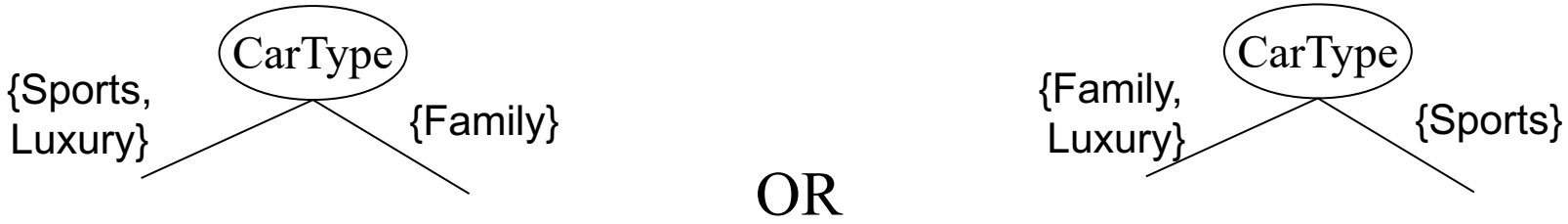


Splitting Based on Nominal Attributes

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



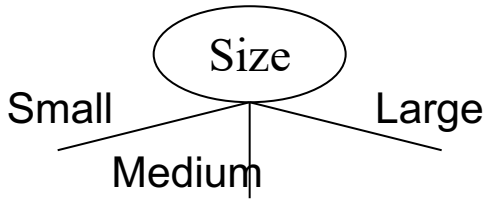
- Binary split: Grouping values into two subsets



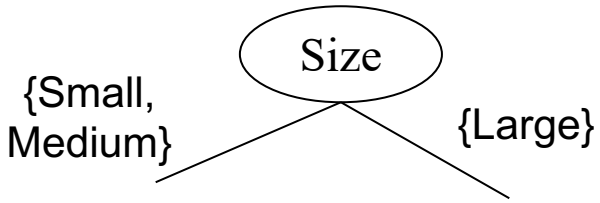
- Need to find optimal partitioning

Splitting Based on Ordinal Attributes

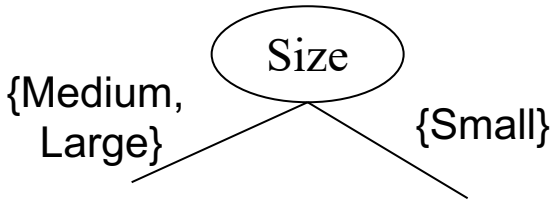
- Multi-way split: Use as many partitions as distinct values



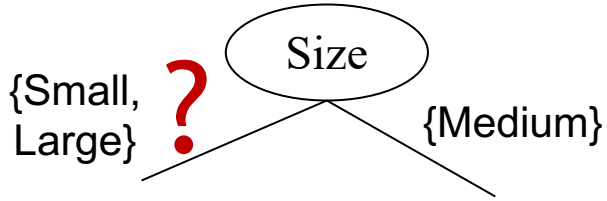
- Binary split: Grouping values into two subsets



OR



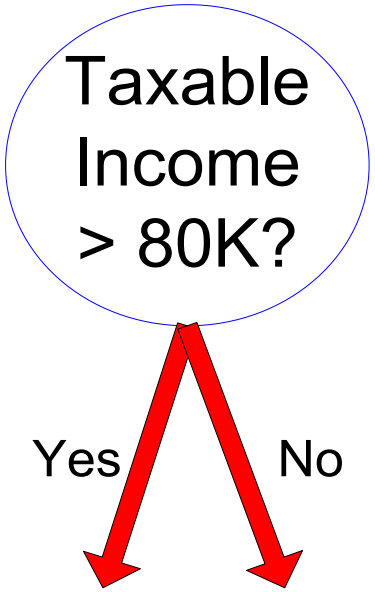
- Need to find optimal partitioning
- Grouping: maintain the ordinal property



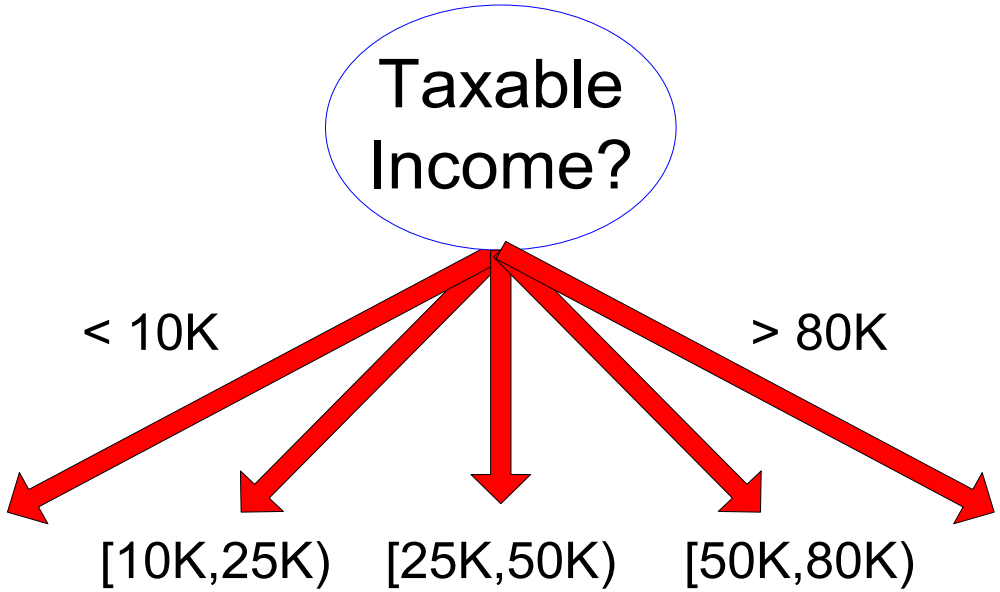
Splitting Based on Continuous Attributes

- 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), clustering
  - Binary Decision: $(A < v)$ or $(A \geq v)$
    - consider all possible splits and finds the best cut v
    - can be compute intensive

Splitting Based on Continuous Attributes



(i) Binary split



(ii) Multi-way split

Remark

- Greedy Algorithms
  - Grow a decision tree by making a series of locally optimum decisions on which attributes to use for partitioning the data
  - Top-down approach is by far the most common strategy
- Finding The Optimal Decision Tree is NP-Hard
  - Exponentially many decision trees can be constructed from a given set of attributes
- Fundamental Problems
  - Determine how to split the records
    - How to specify the attribute test condition?
    - How to determine the best split?
  - Determine when to stop splitting

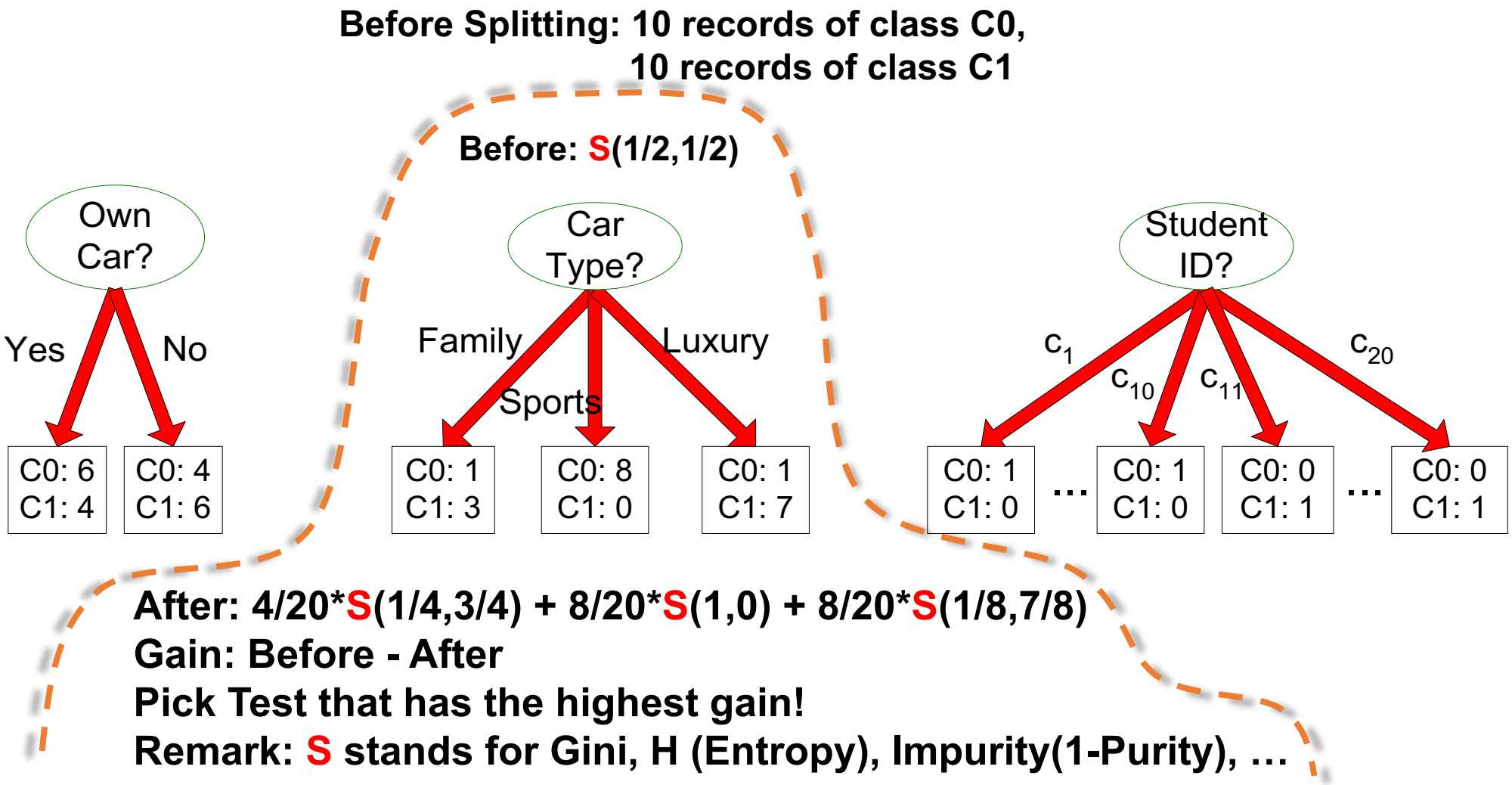
How to Determine the Best Split?

- Greedy approach:
  - Nodes with homogeneous class distribution (pure nodes) are preferred

- Need a measure of node impurity, named $s(\cdot)$:

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

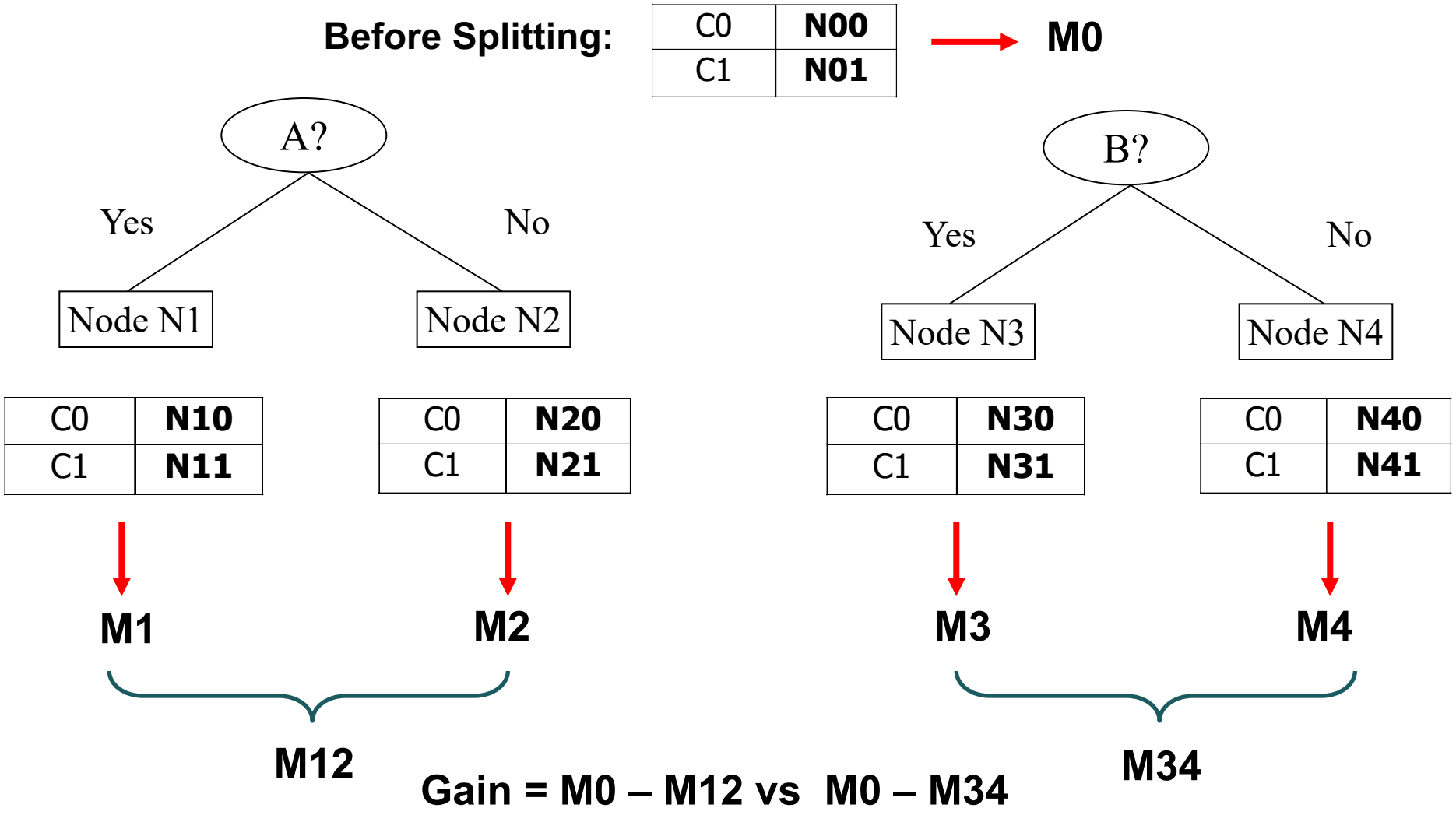
How to Determine the Best Split?



How to Find the Best Split

- Gini Index
- Entropy
- Misclassification error
- Remark: All three approaches center on selecting tests that maximize purity. They just disagree in how exactly purity is measured

Measures of Node Impurity



Measure of Impurity: GINI

- GINI Index for a given node t:

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

Smaller is Better

  - NOTE: $p(j|t)$ 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 (0.0) when all records belong to one class, implying most interesting information

C1	**0**
C2	**6**
Gini=0.000	

C1	**1**
C2	**5**
Gini=0.278	

C1	**2**
C2	**4**
Gini=0.444	

C1	**3**
C2	**3**
Gini=0.500	

$1 - (1/6)^2 - (5/6)^2 = 0.278$

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

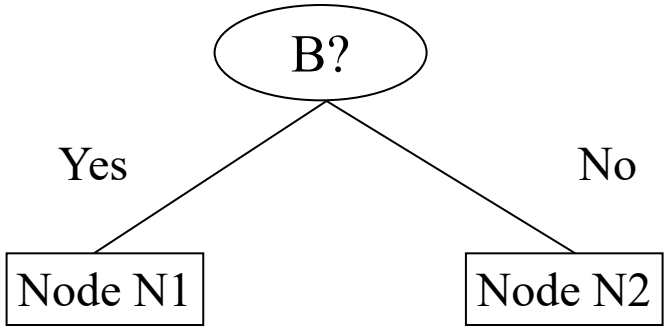
- Used in CART, SLIQ, SPRINT
- When a node t is split into k partitions (children), the quality of the split is computed as

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

- where, n_i = number of records at child node i
  n = number of records at parent node t

Binary Attributes: Computing GINI Index

- Splitting the samples at node t in two partitions
- Effect of weighted partitions
  - Search for large and pure partitions



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

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

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

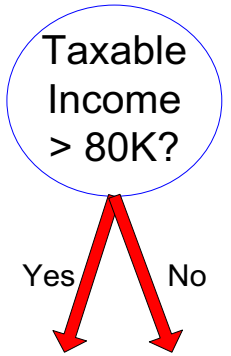
	N1	N2
C1	**5**	**1**
C2	**2**	**4**
Gini=y		

Gini(Children)
= 7/12 * Gini(N1) +
5/12 * Gini(N2)=y

Continuous Attributes: Computing GINI

- Use Binary Decisions based on one value
- Several Choices for the splitting value
  - Number of possible splitting values = Number of distinct values
- Each splitting value has a count matrix associated with it
  - Class counts in each of the partitions, $A < v$ and $A \geq v$
- Simple method to choose best v
  - For each v, scan the database to gather count matrix and compute its Gini index
  - Computationally Inefficient! Repetition of work

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



Continuous Attributes: Computing GINI

- For efficient computation:
- For each attribute,
  - Sort the attribute on values
  - Linearly 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	

Another Measure of Impurity: Entropy

- Entropy at a given node t:

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

(NOTE: $p(j|t)$ 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**

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) $\log_2$ (5/6) = 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

Range of Impurity Measures

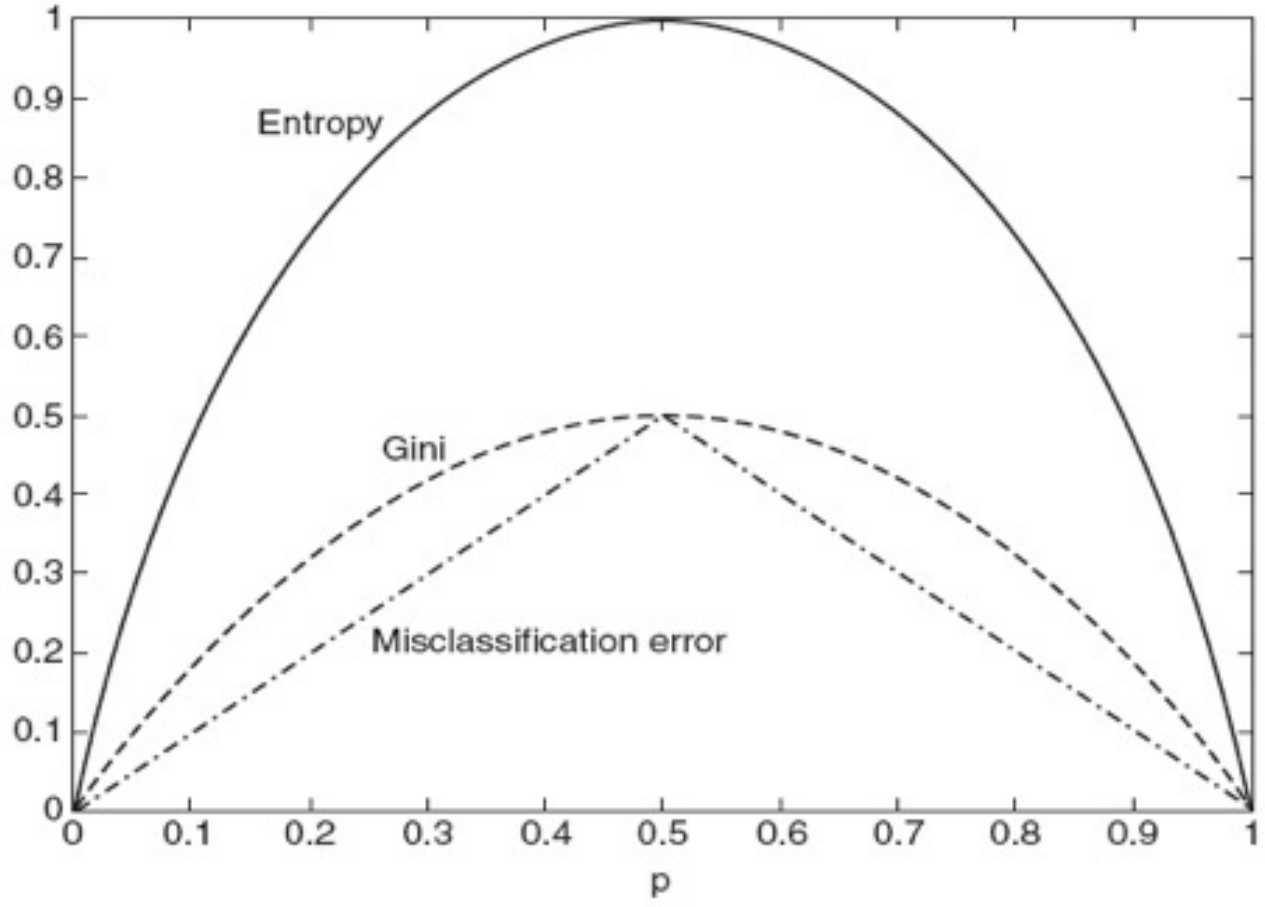


Figure 4.13. Comparison among the impurity measures for binary classification problems.

Splitting Based on Entropy

- Information Gain:

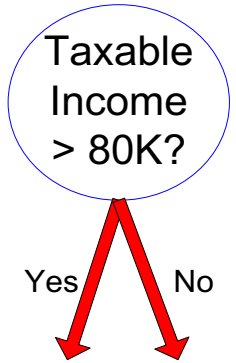
$$GAIN_{split} = Entropy(p) - \left(\sum_{i=1}^{k} \frac{n_i}{n} Entropy(i) \right)$$

Parent Node, p is split into k partitions
n_i is number of records in partition i

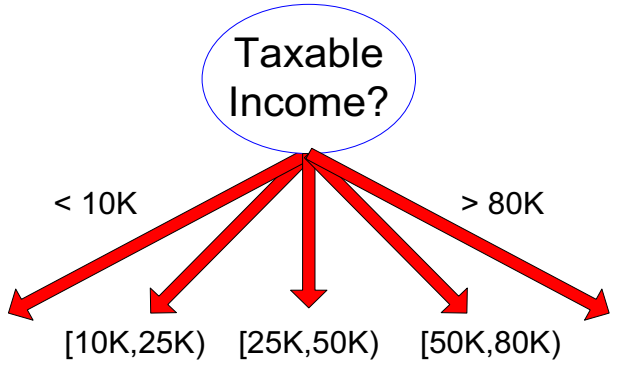
  - Measures Reduction in Entropy achieved because of the split. Choose the split that achieves most reduction (maximizes GAIN)
  - Used in ID3 and C4.5

Is Minimizing Impurity Enough?

- Disadvantage: impurity measures favor splits that result in large number of partitions, each being small but pure
- A test condition with large number of outcomes may not be desirable
  - The number of records in each partition is too small to make predictions



(i) Binary split



(ii) Multi-way split

Splitting Based on Entropy

- 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
ni is the number of records in partition i

- Adjusts Information Gain by the entropy of the partitioning (SplitINFO). Tests which generate more nodes are penalized
- Used in C4.5
- Designed to overcome the disadvantage of information gain not considering the number of nodes generated by a split
- If each split has the same number of records, SplitINFO = logk
- Large number of splits -> large splitINFO -> small gain ratio

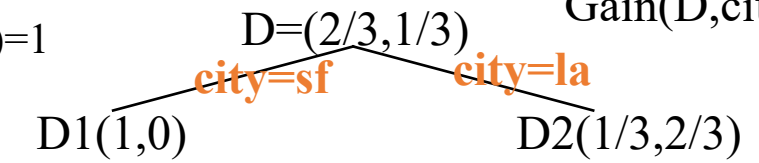
Information Gain vs Gain Ratio

Result:
I_Gain_Ratio:
city>age>car

sale	custId	car	age	city	newCar
	c1	taurus	27	sf	yes
	c2	van	35	la	yes
	c3	van	40	sf	yes
	c4	taurus	22	sf	yes
	c5	merc	50	la	no
	c6	taurus	25	la	no

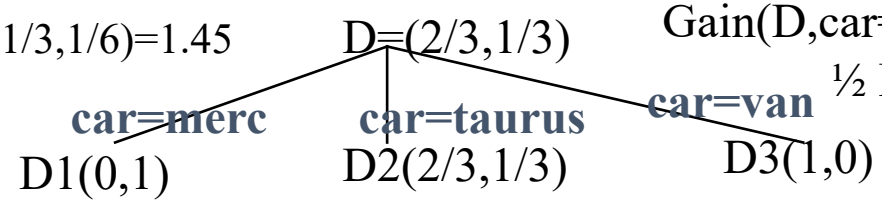
Result:
I_Gain:
age > car=city

SplitINFO(city=)=E(1/2,1/2)=1



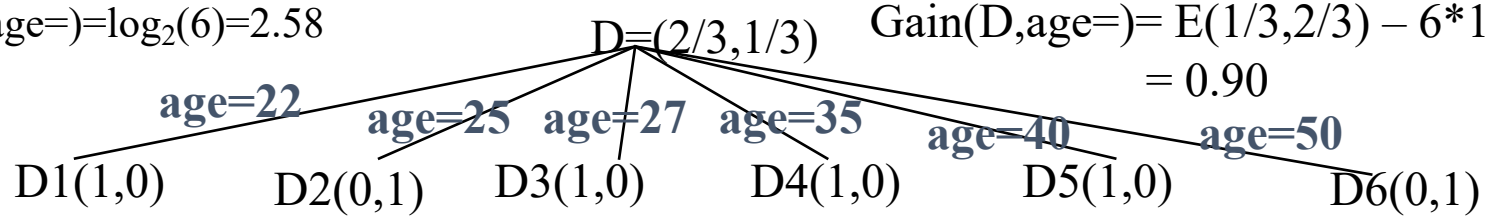
Gain(D,city=)= E(1/3,2/3) – ½ E(1,0) – ½ E(1/3,2/3)=0.45

SplitINFO(car=)= E(1/2,1/3,1/6)=1.45



Gain(D,car=)= E(1/3,2/3) – 1/6 E(0,1) – ½ E(2/3,1/3) – 1/3 E(1,0)=0.45

SplitINFO(age=)=$\log_2(6)$=2.58



Gain(D,age=)= E(1/3,2/3) – 6*1/6 H(0,1) = 0.90

Splitting 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

Comment: same as impurity which is (1-Purity)!

Example:
Error((0.4,0.2,0,0.1,0.3))= 1-Purity((0.4,0.2,0,0.1,0.3))=1-0.4=0.6

Examples for Computing Error

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

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

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

Error = 1 – max (0, 1) = 1 – 1 = 0

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

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

Error = 1 – max (1/6, 5/6) = 1 – 5/6 = 1/6

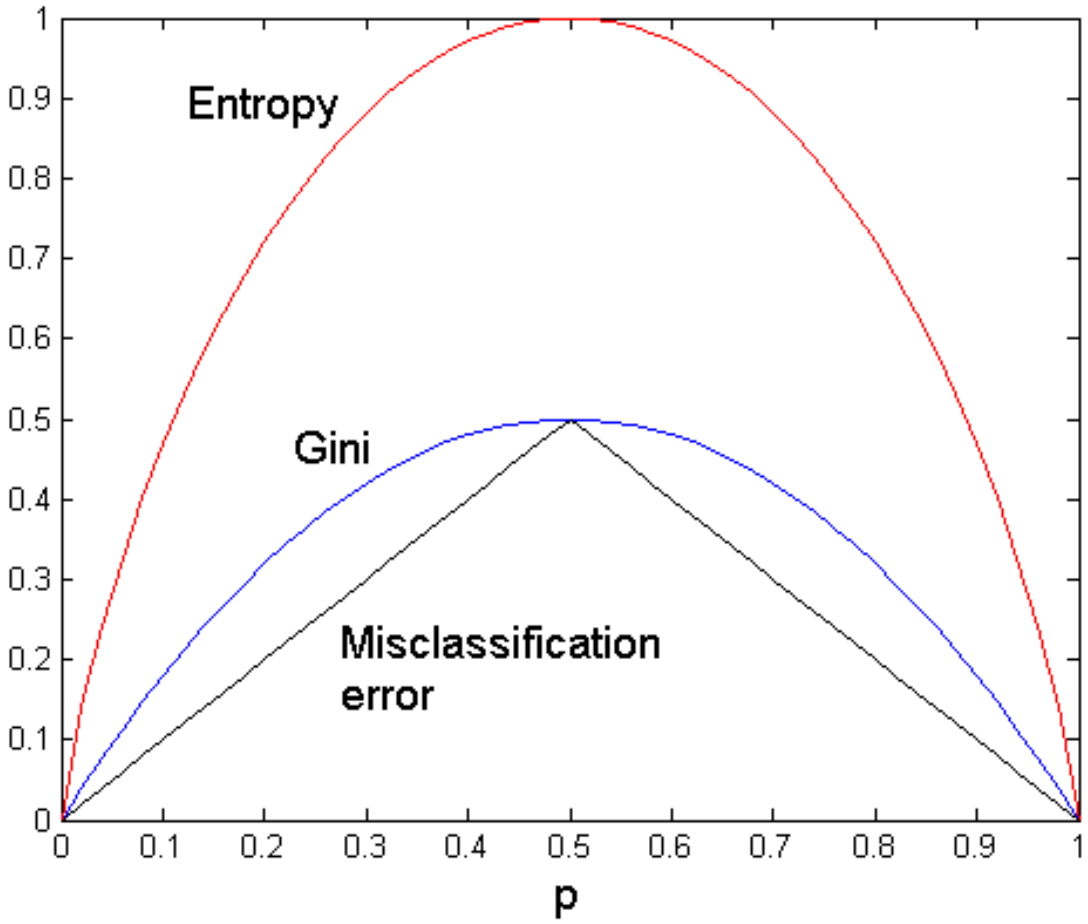
C1	**2**
C2	**4**

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

Error = 1 – max (2/6, 4/6) = 1 – 4/6 = 1/3

Comparison among Splitting Criteria

- For 2-Class Problem



Remark

- Greedy Algorithms
  - Grow a decision tree by making a series of locally optimum decisions on which attributes to use for partitioning the data
  - Top-down approach is by far the most common strategy
- Finding The Optimal Decision Tree is NP-Hard
  - Exponentially many decision trees can be constructed from a given set of attributes
- Fundamental Problems
  - Determine how to split the records
    - How to specify the attribute test condition?
    - How to determine the best split?
  - Determine when to stop splitting

Stopping Criteria for Tree Induction

- Stop expanding a node when all the records belong to the same class
- Stop expanding a node when no records left or all attributes have been used for splitting
- Stop expanding a node when none of the attributes provide gain
- Early termination (used by many decision tree tools to avoid overfitting); examples include:
  - Terminate if the number of examples left is less than x (e.g. x=4)
  - Terminate if the depth of the tree has reached a limit (e.g. limit=6)
  - Terminate if the purity of the examples associated with a node is x% or more (e.g. x=90)

Example: C4.5

- Simple top-down construction
- Uses Information Gain (Entropy)
- Sorts continuous attributes at each node
- Needs entire data to fit in memory
- One of the Top 10 Data Mining Algorithms in 2007

Decision Tree in Sklearn

```
class sklearn.tree.DecisionTreeClassifier(*, criterion='gini', splitter='best', max_depth=None, min_samples_split=2, min_samples_leaf=1, min_weight_fraction_leaf=0.0, max_features=None, random_state=None, max_leaf_nodes=None, min_impurity_decrease=0.0, class_weight=None, ccp_alpha=0.0, monotonic_cst=None)
```

criterion: {"gini", "entropy", "log_loss"}, default="gini"

https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html

```
>>> from sklearn.tree import DecisionTreeClassifier
>>> from sklearn.model_selection import train_test_split
>>> clf = DecisionTreeClassifier(criterion="entropy", max_depth=5, random_state=1)
>>> X_tr, X_te, y_tr, y_te = train_test_split(X, y, train_size=0.9)
>>> clf.fit(X_tr, y_tr)
>>> y_te = clf.predict(X_te)
```

How to handle categorical attributes?

- One-hot representation

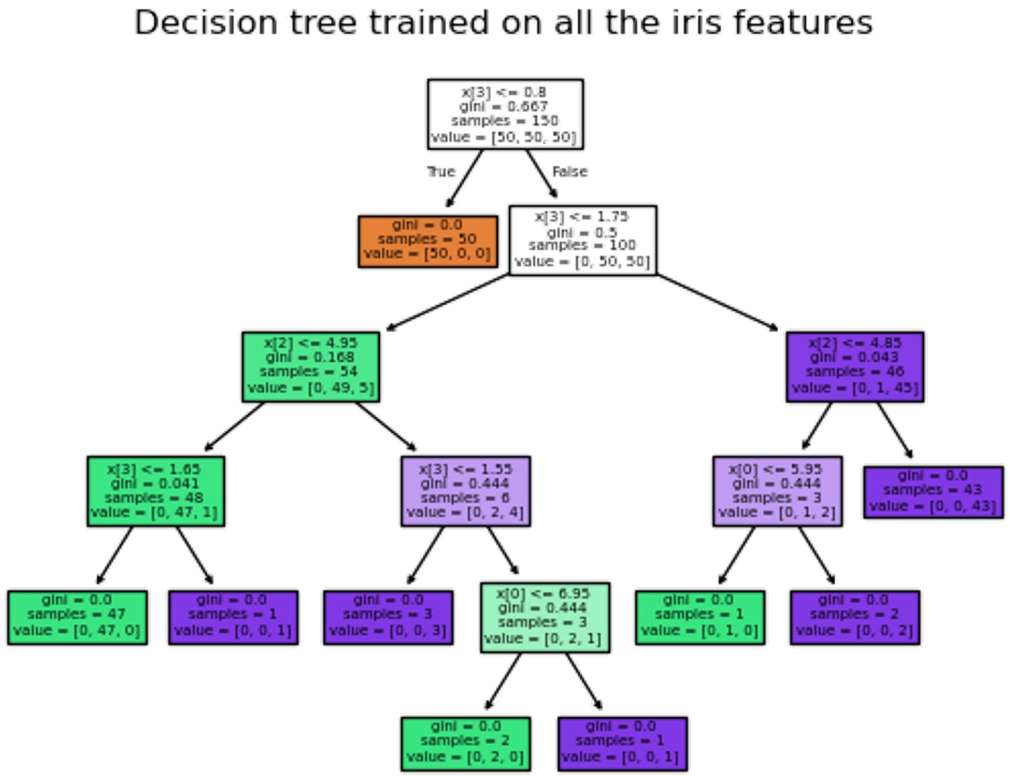
Tid	Marital Status
1	Single
2	Married
3	Single
4	Married
5	Divorced
6	Married
7	Divorced
8	Single
9	Married
10	Single

Tid	Single	Married	Divorced
1	1	0	0
2	0	1	0
3	1	0	0
4	0	1	0
5	0	0	1
6	0	1	0
7	0	0	1
8	1	0	0
9	0	1	0
10	1	0	0

Decision Tree in Sklearn

```
class sklearn.tree.DecisionTreeClassifier(*, criterion='gini', splitter='best', max_depth=None, min_samples_split=2, min_samples_leaf=1, min_weight_fraction_leaf=0.0, max_features=None, random_state=None, max_leaf_nodes=None, min_impurity_decrease=0.0, class_weight=None, ccp_alpha=0.0, monotonic_cst=None)
```

```
>>> from sklearn.tree import DecisionTreeClassifier
>>> from sklearn.model_selection import train_test_split
>>> clf = DecisionTreeClassifier(criterion="entropy", max_depth=5, random_state=1)
>>> X_tr, X_te, y_tr, y_te = train_test_split(X, y, train_size=0.9)
>>> clf.fit(X_tr, y_tr)
>>> y_te = clf.predict(X_te)

>>> from sklearn.tree import plot_tree
>>> import matplotlib as plt
>>> plt.figure()
>>> tree.plot_tree(clf, filled=True)
>>> plt.title("Decision tree trained on all the iris features")
>>> plt.show()
```



COSC 3337
Data Science I
Section 14623

Classification (contd.)

Instructor: Jingchao Ni
Fall 2024

Advantages of Decision Tree

- Extremely fast at classifying unknown records
- Easy to interpret for small-sized trees, strategy understandable to domain expert, Trees can be visualized. A white box model, in contrast to a black box model
- Requires little data preparation, Okay for noisy data
- Can handle both continuous and symbolic attributes, However, the scikit-learn implementation does not support categorical variables for now
- Useful for exploratory data analysis
- Accuracy is comparable to other classification techniques for many simple datasets
- Decent average performance over many datasets
- Kind of a standard—if you want to show that your “new” classification technique really “improves the world” -> e.g., compare its performance against decision trees using 10-fold cross-validation
- Does not need distance functions; only the order of attribute values is important for classification: 0.1,0.2,0.3 and 0.331,0.332, and 0.333 is the same for a decision tree learner

Disadvantages of Decision Tree

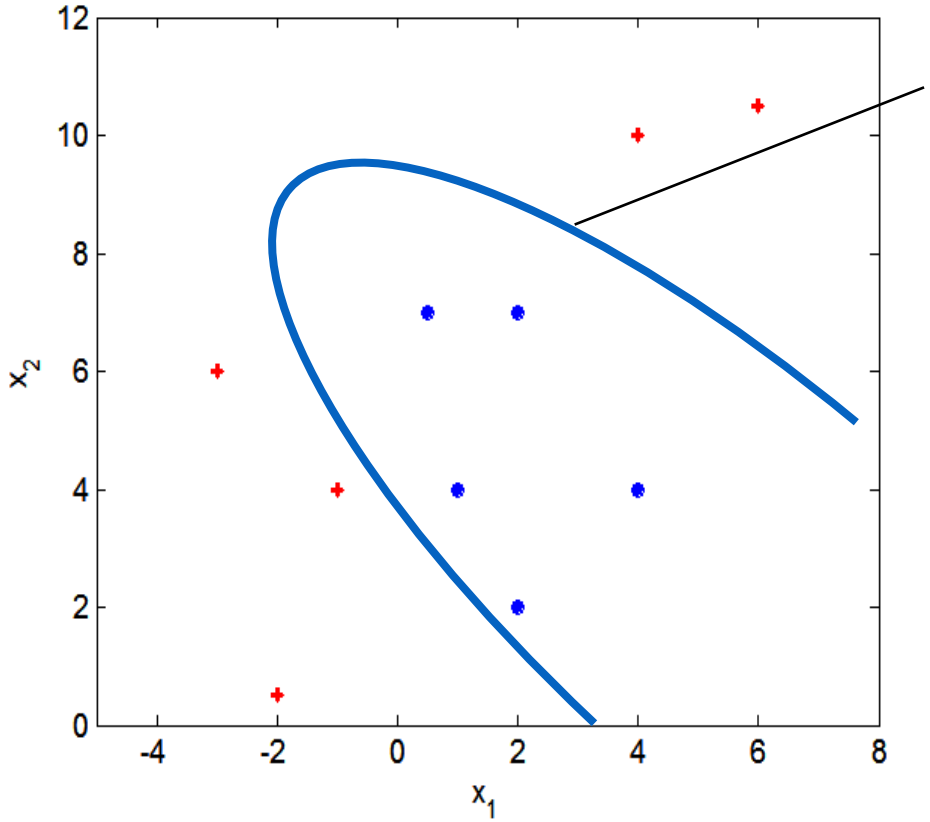
- Decision-tree learners can create over-complex trees that do not generalize the data well. This is called overfitting. Pruning, setting the minimum number of samples required at a leaf node or setting the maximum depth of the tree are necessary to avoid this problem
- Decision trees can be unstable because small variations in the data might result in a completely different tree being generated. This problem is mitigated by using decision trees within an ensemble
- Predictions of decision trees are neither smooth nor continuous, but piecewise constant approximations. Therefore, they are not good at extrapolation.
- The problem of learning an optimal decision tree is known to be NP-complete. Consequently, practical decision-tree learning algorithms are based on greedy algorithm where locally optimal decisions are made at each node. Such algorithms cannot guarantee global optimal decision tree. This can be mitigated by training multiple trees in an ensemble learner, where the features and samples are randomly sampled with replacement
- Decision tree learners create biased trees if some classes dominate. It is therefore recommended to balance the dataset prior to fitting with the decision tree.

Decision Tree Algorithms

- ID3 (Iterative Dichotomiser 3) creates a multiway tree, finding for each node (i.e. in a greedy manner) the categorical feature that will yield the largest information gain for categorical targets. Trees are grown to their maximum size and then a pruning step is usually applied to improve the ability of the tree to generalize to unseen data.
- C4.5 is the successor to ID3 and removed the restriction that features must be categorical by dynamically defining a discrete attribute (based on numerical variables) that partitions the continuous attribute value into a discrete set of intervals. C4.5 converts the trained trees (i.e. the output of the ID3 algorithm) into sets of if-then rules. The accuracy of each rule is then evaluated to determine the order in which they should be applied. Pruning is done by removing a rule's precondition if the accuracy of the rule improves without it.
- C5.0 is Quinlan's latest version release under a proprietary license. It uses less memory and builds smaller rulesets than C4.5 while being more accurate.
- CART (Classification and Regression Trees) is very similar to C4.5, but it differs in that it supports numerical target variables (regression) and does not compute rule sets. CART constructs binary trees using the feature and threshold that yield the largest information gain at each node.

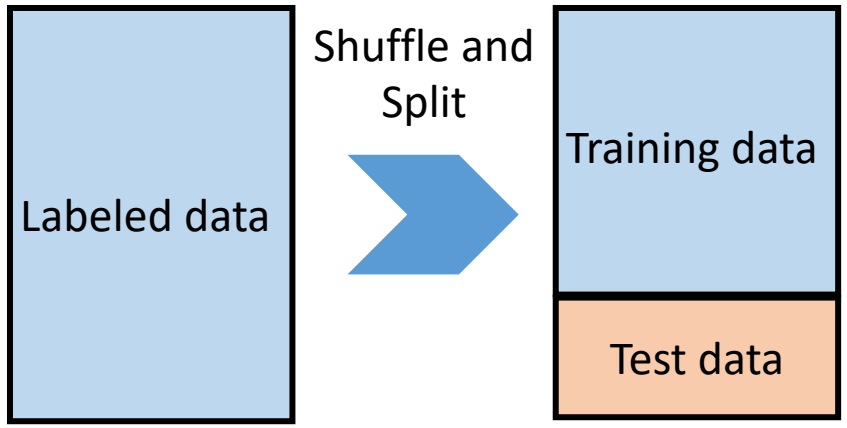
Classification and Decision Boundaries

- Classification can be viewed as “learning good decision boundaries” that separate the examples belonging to different classes in a data set

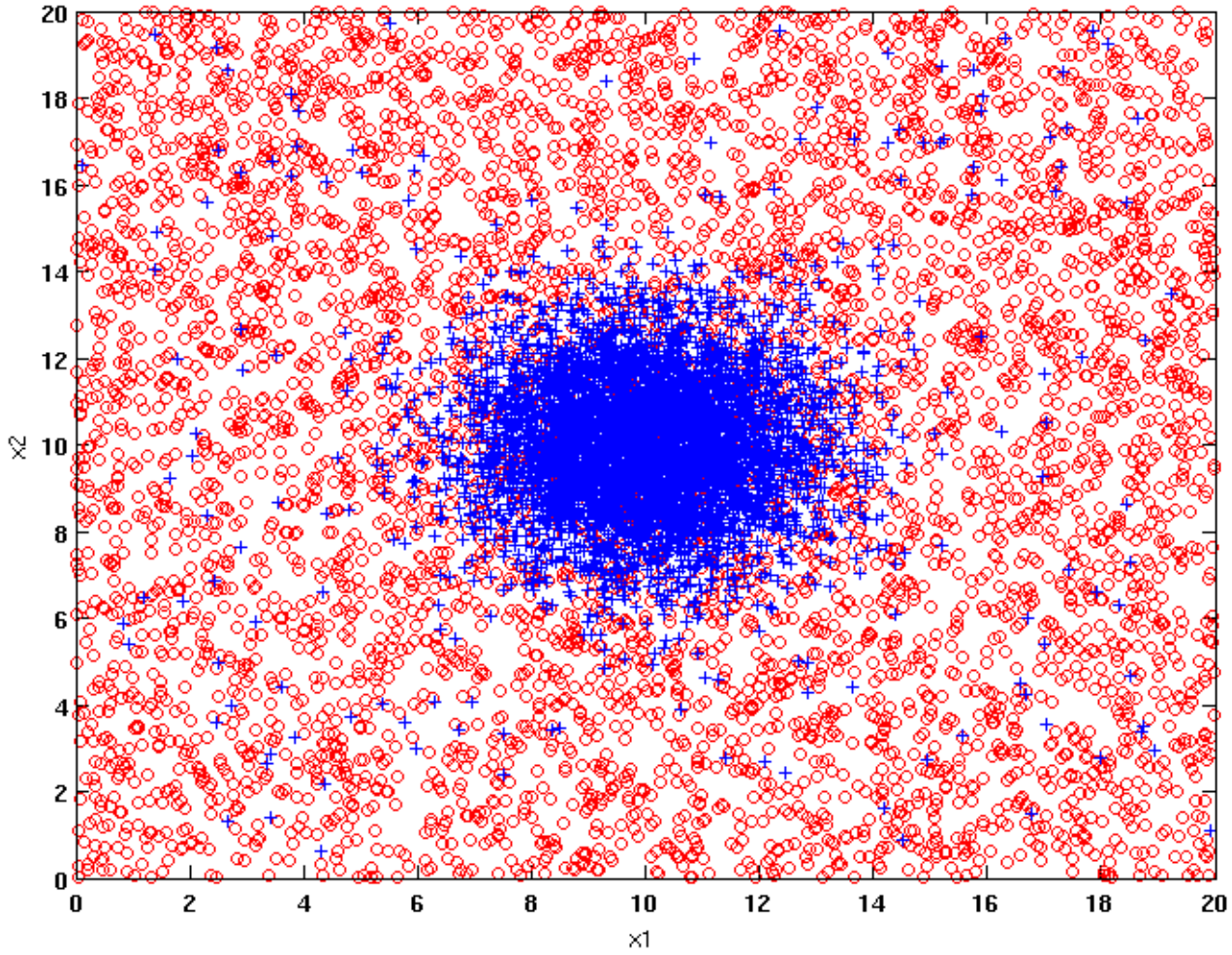


Types of Classification Errors

- Training errors (apparent errors)
  - Errors committed on the training set
- Test errors
  - Errors committed on the test set
- Generalization errors
  - Expected error of a model over a random selection of records from same distribution
  - It is estimated by test errors on a specific test set



Example Dataset



Two class problem:

+ : 5200 instances

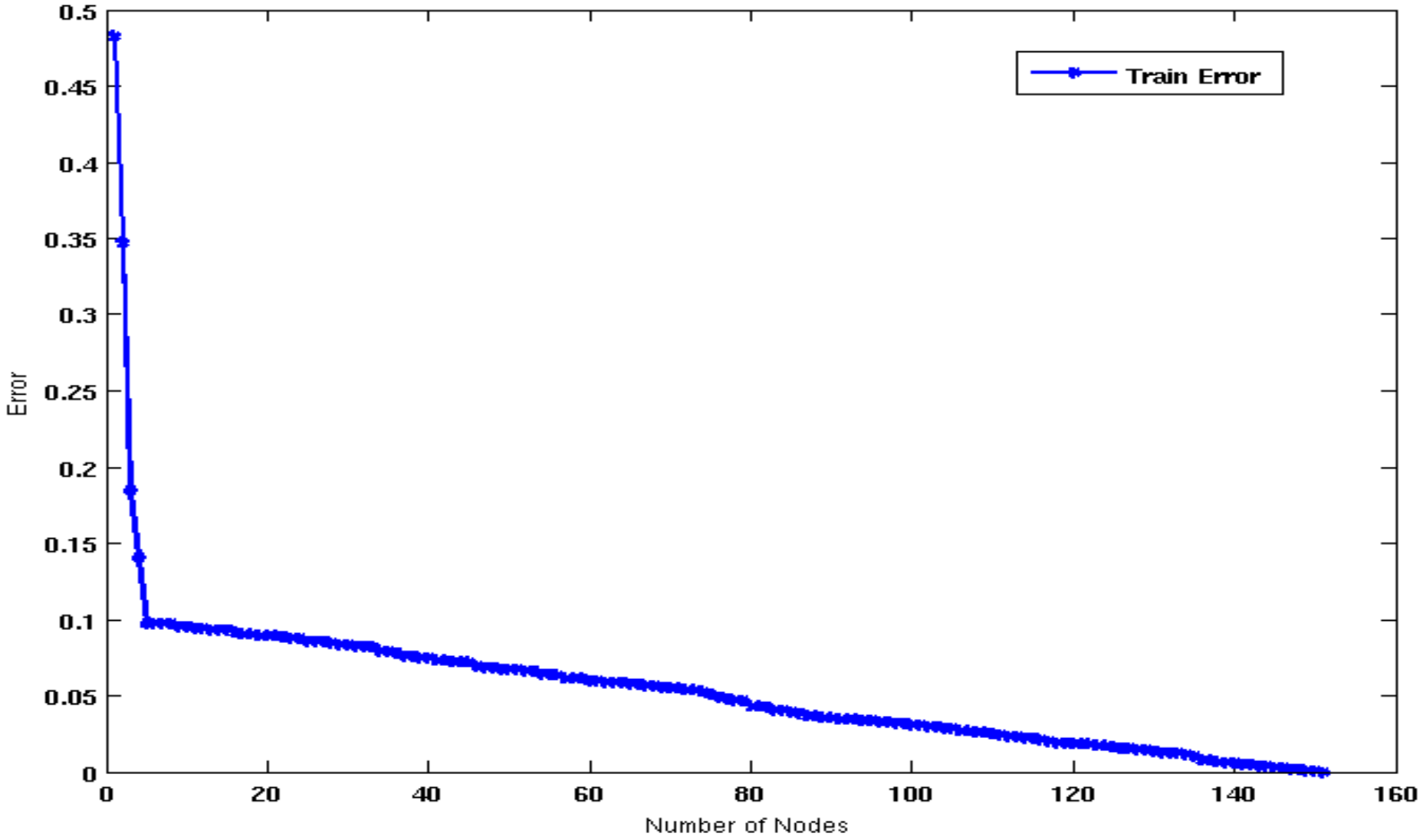
- **5000 instances generated from a Gaussian centered at (10,10)**
- **200 noisy instances added**

o : 5200 instances

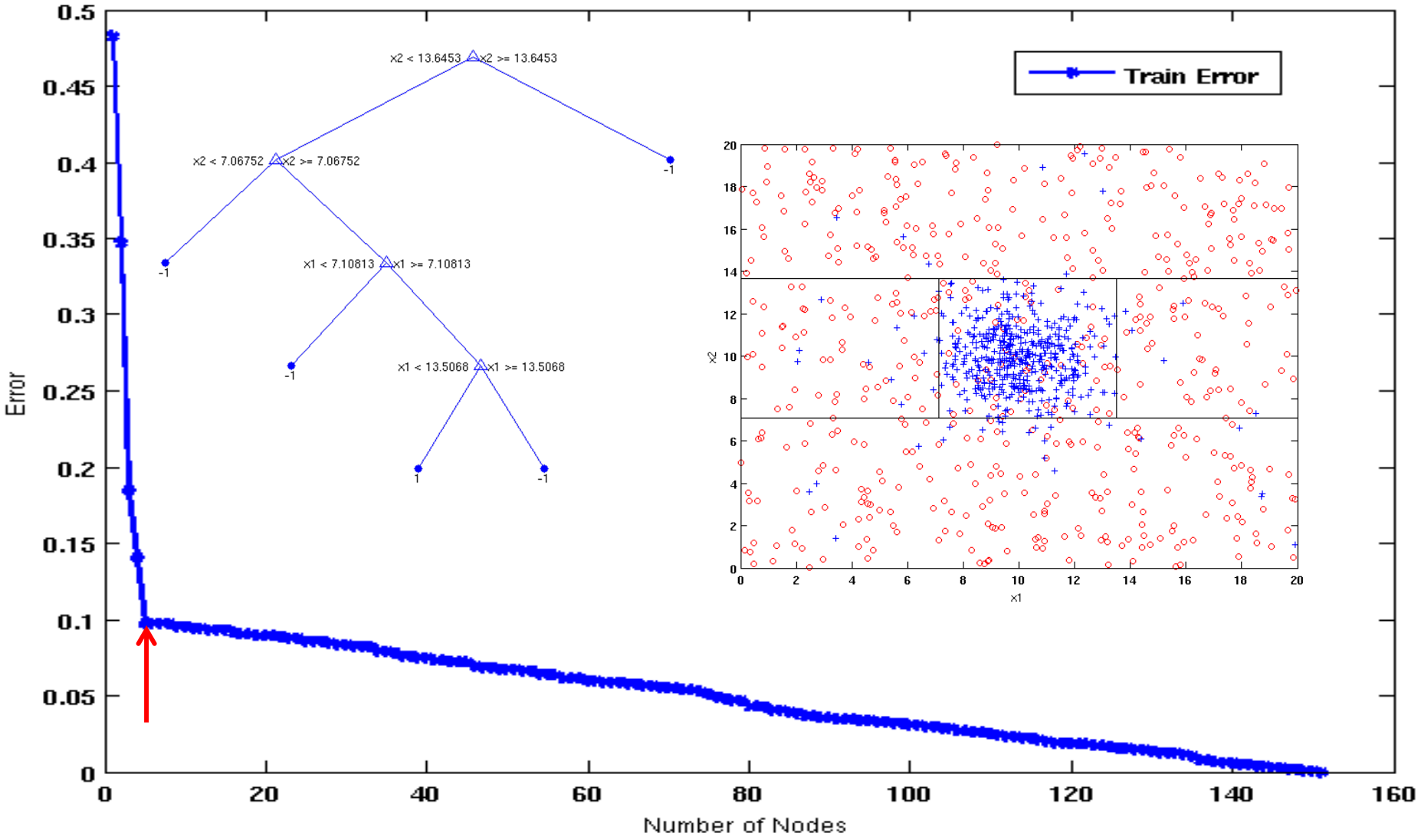
- **Generated from a uniform distribution**

10 % of the data used for training and 90% of the data used for testing

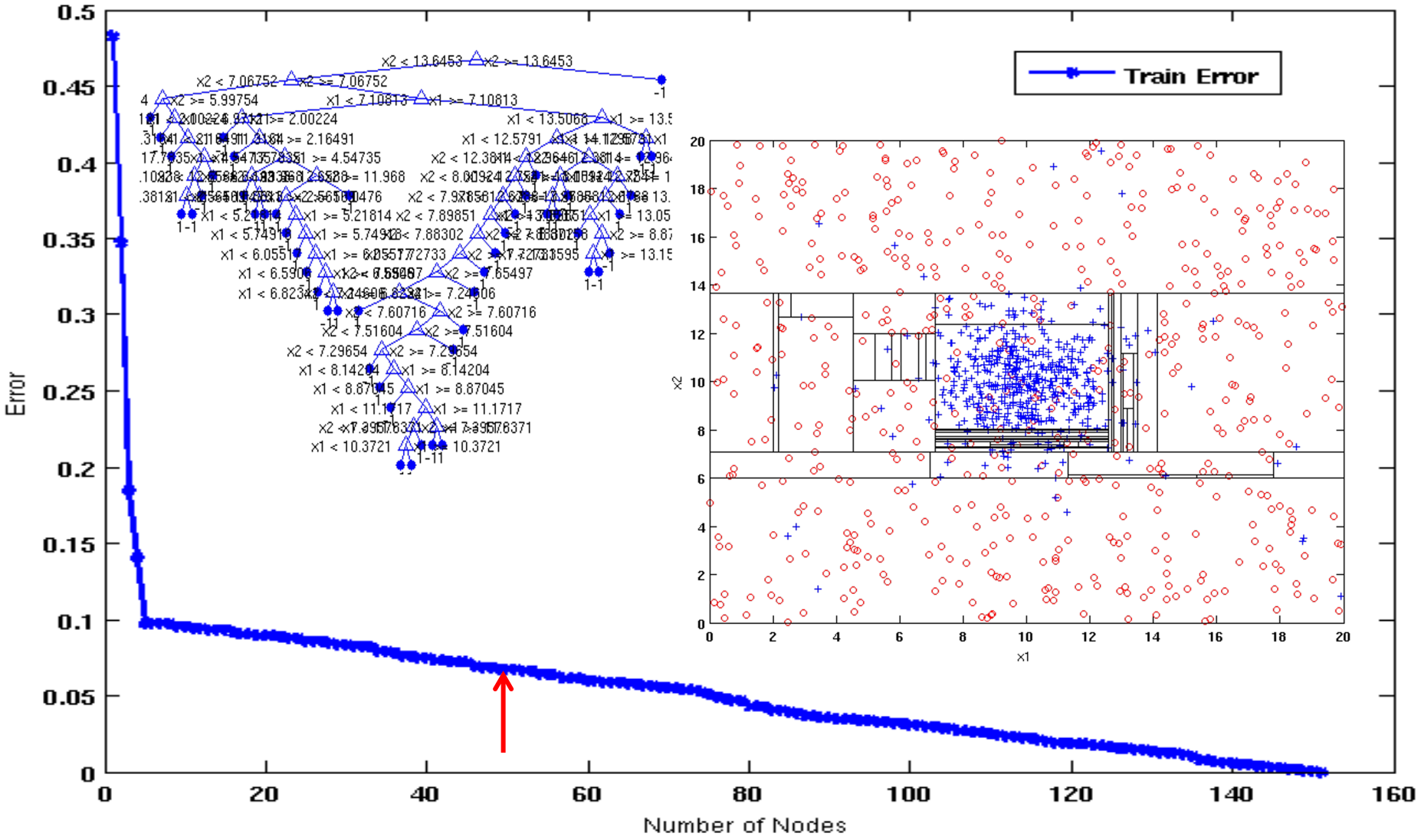
Decision Tree Classifier: Adding Nodes



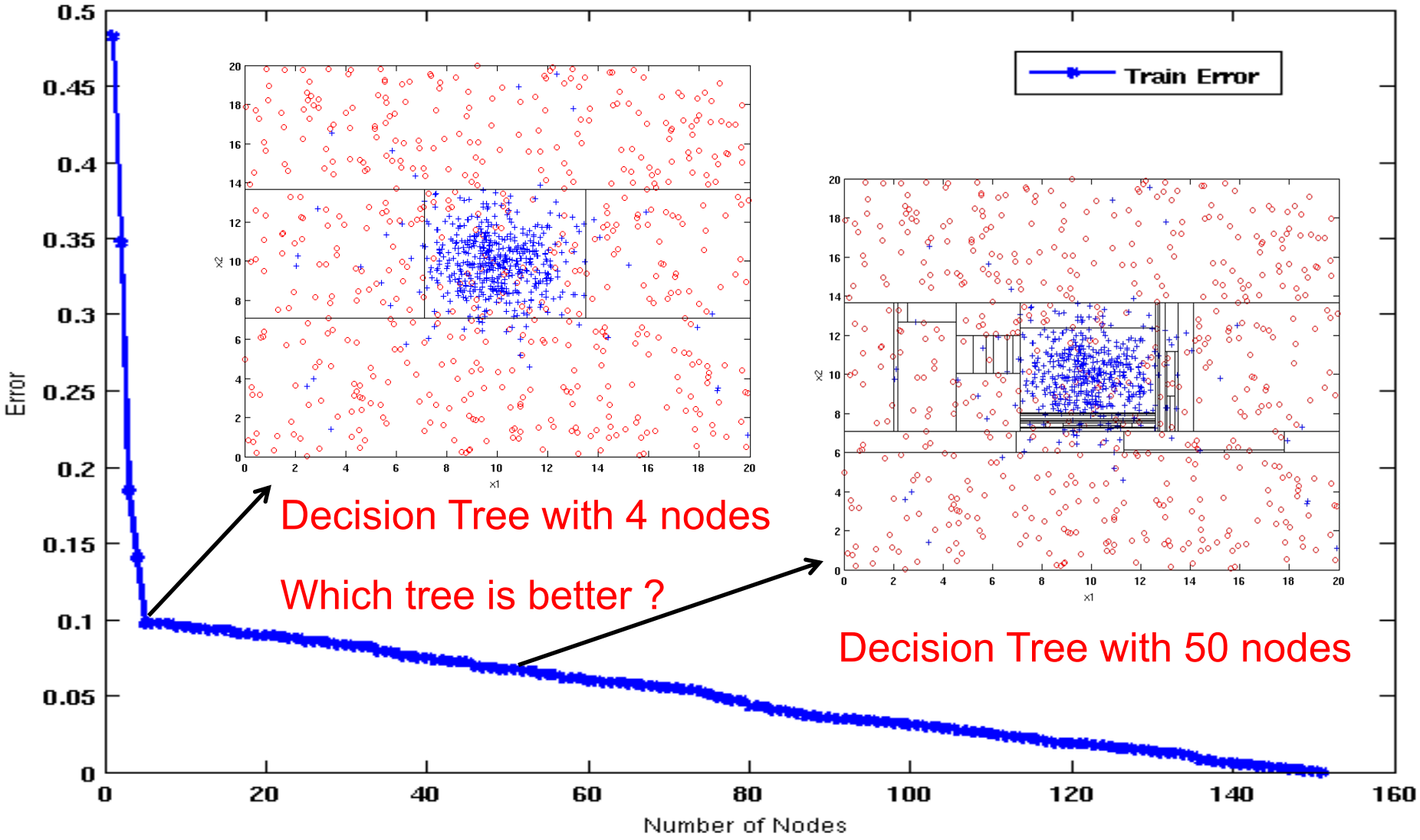
Decision Tree with 4 Nodes



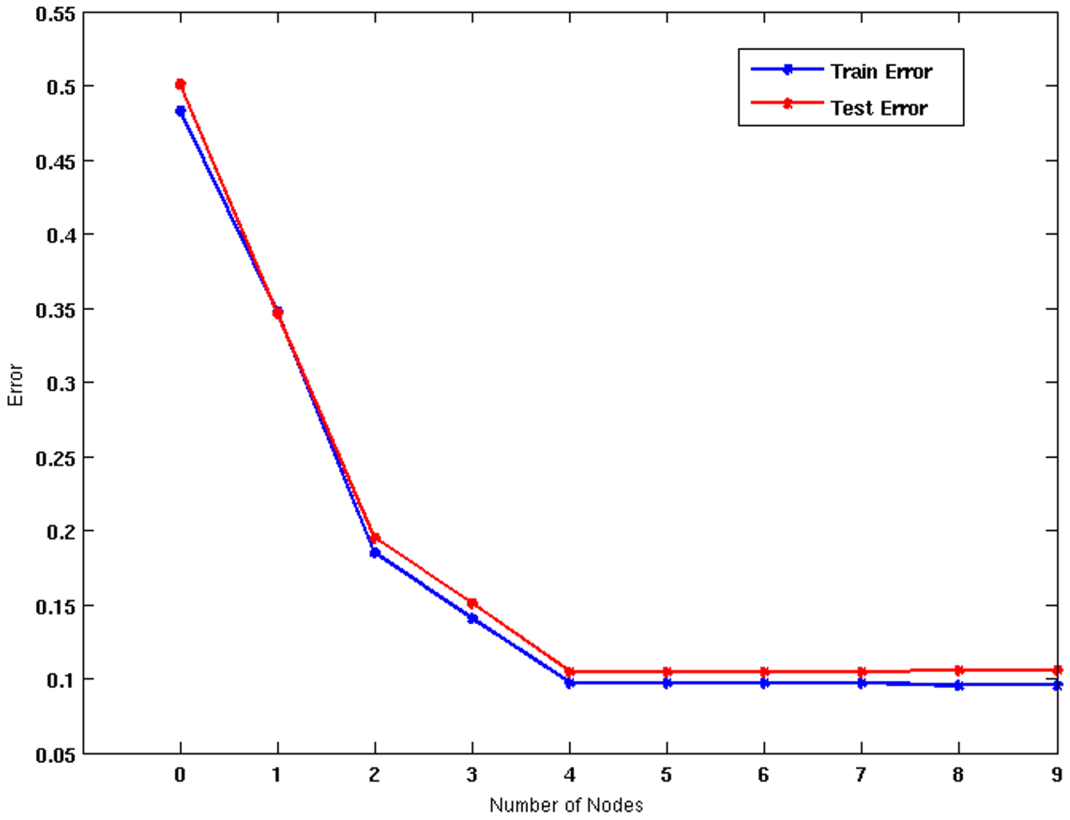
Decision Tree with 50 Nodes

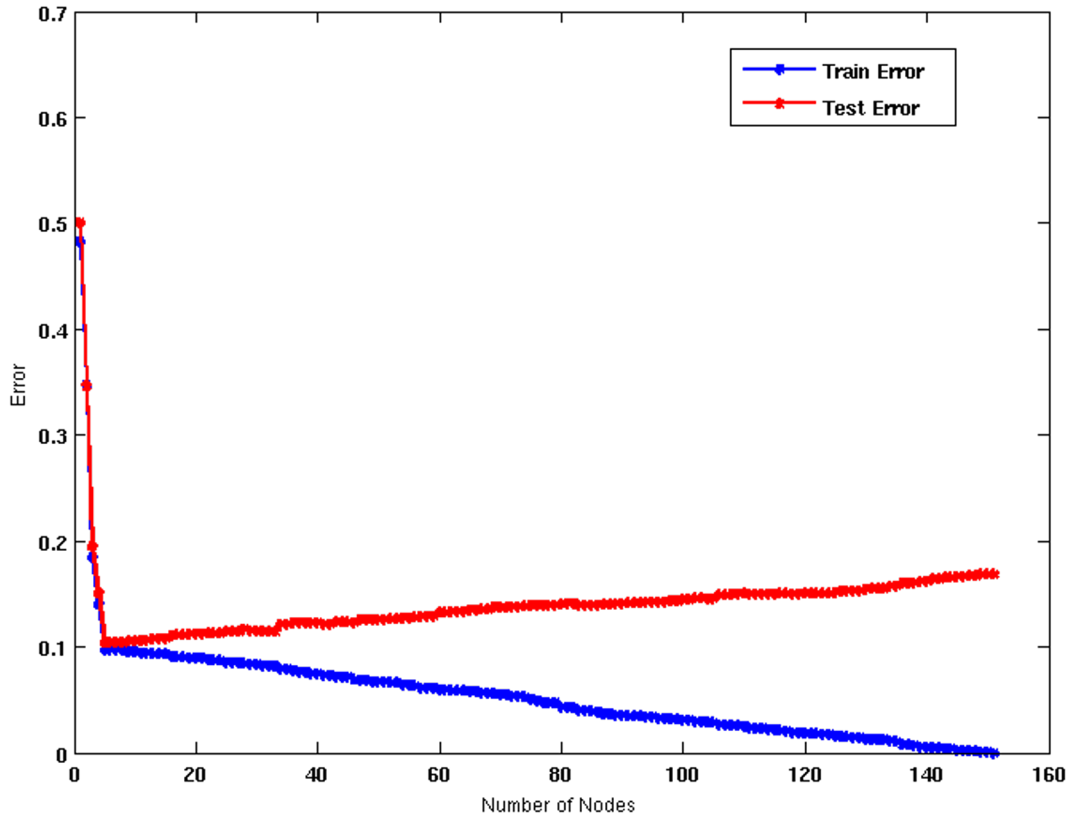


Which Tree is Better?



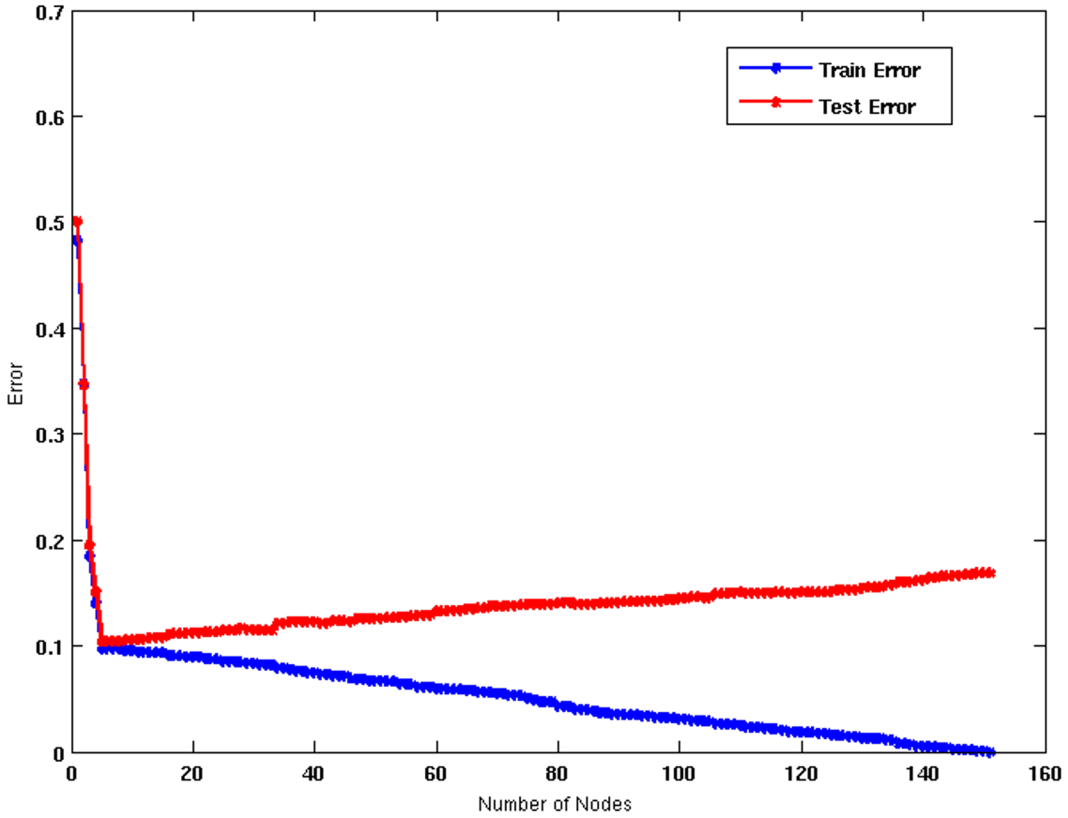
Model Overfitting

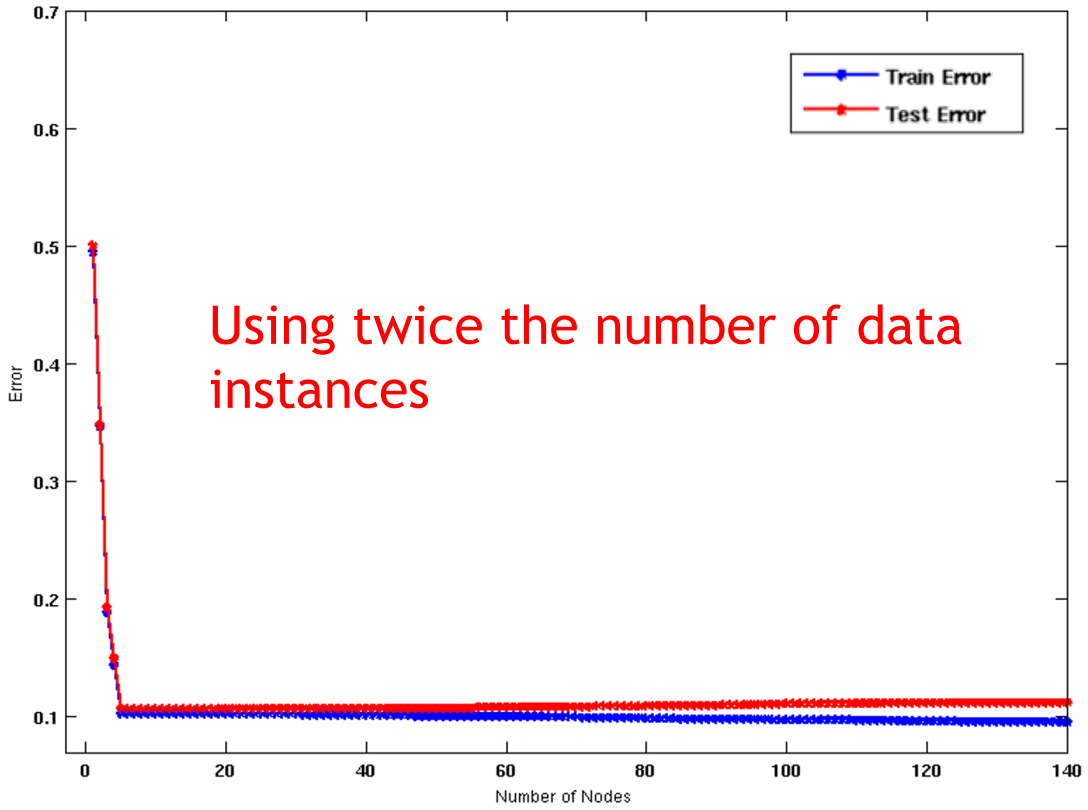




- **Underfitting**: when model is too simple, both training and test errors are large
- **Overfitting**: when model is too complex, training error is small but test error is large

Model Overfitting



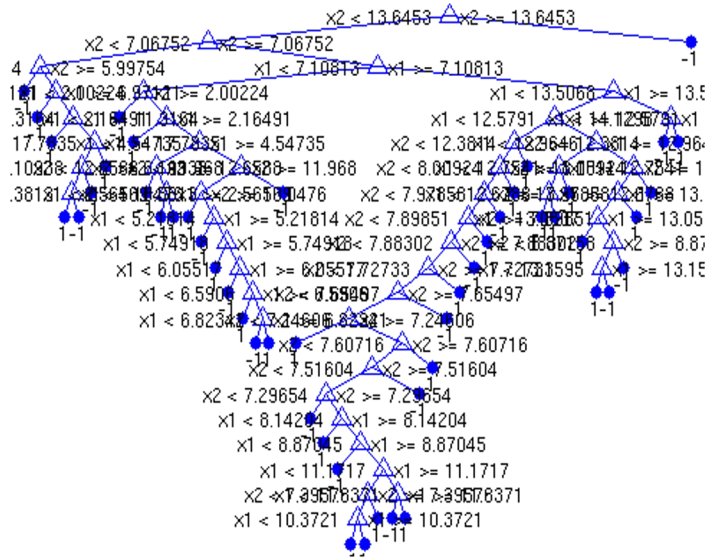


Using twice the number of data instances

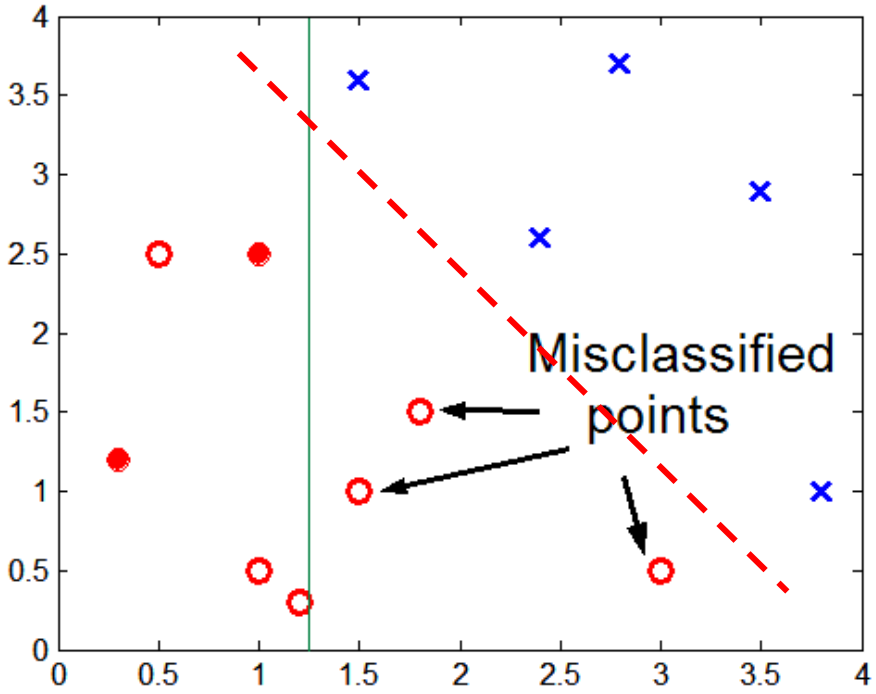
- If training data is under-representative, testing errors increase and training errors decrease on increasing number of nodes
- Increasing the size of training data reduces the difference between training and testing errors at a given number of nodes

Reasons for Model Overfitting

- Limited Training Set Size
  - Training set does not contain enough data samples to accurately represent all possible input data values
- Non-Representative Training Examples
  - Noisy Examples
- High Model Complexity
  - Multiple Comparison Procedure
- The model trains for too long on a single sample set of data
  - Fits noisy examples



Overfitting due to Insufficient Examples



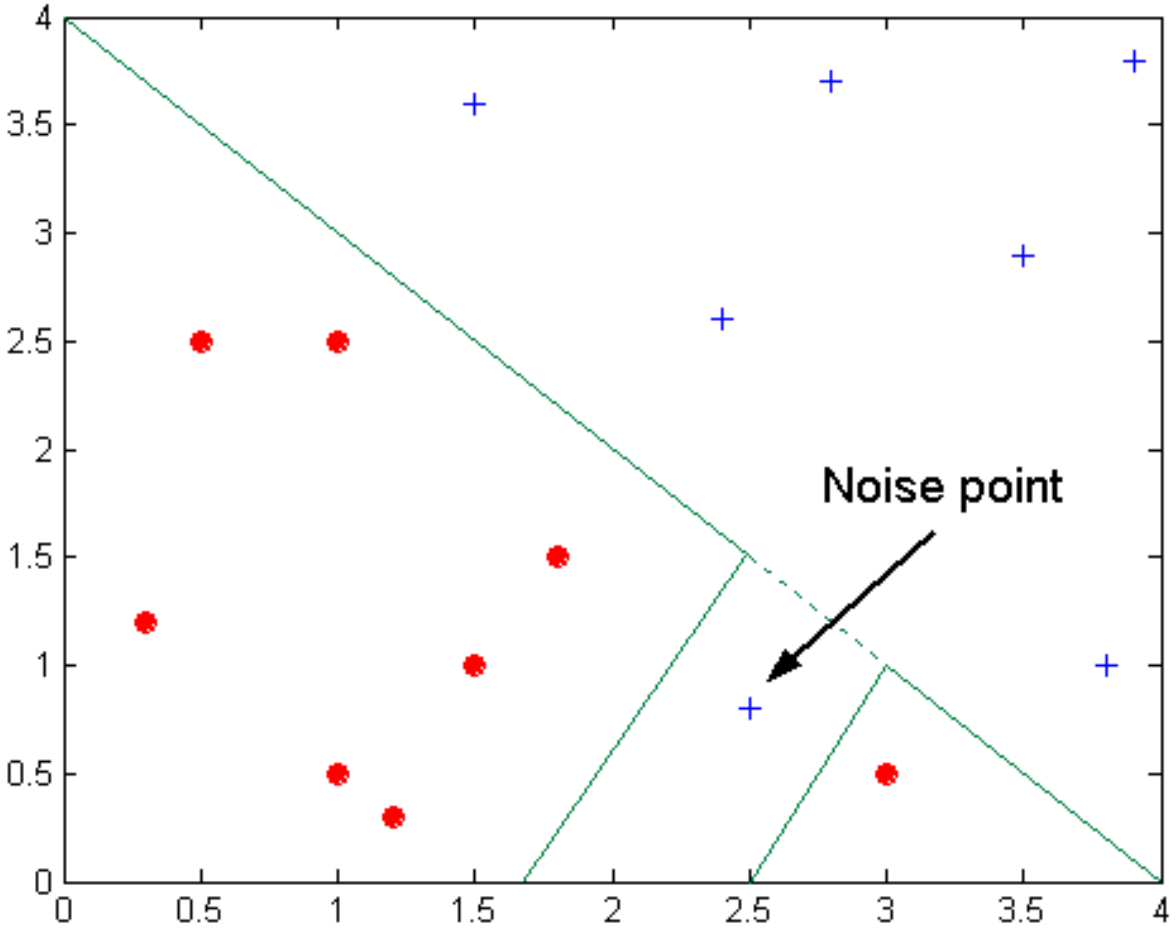
Training Examples

Training Examples

Testing Examples

- Lack of data points in the lower half of the diagram makes it difficult to predict correctly the class labels of that region
- Insufficient number of training records in the region causes the decision tree to predict the test examples without using the true data distribution

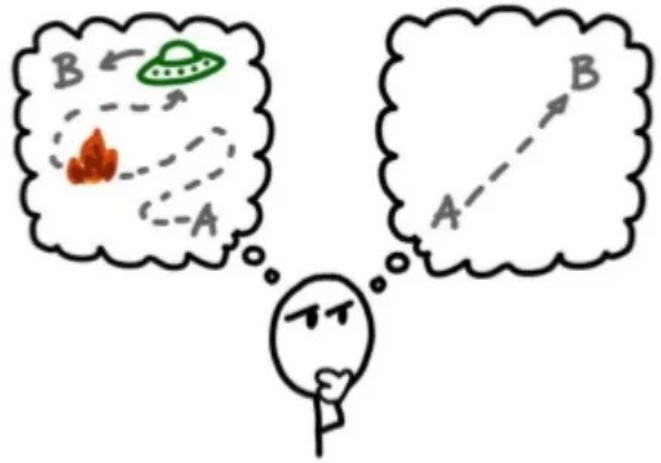
Overfitting due to Noise



Decision boundary is distorted by noise point

Occam's Razor

- Given two models of similar generalization errors, one should prefer the simpler model over the more complex model
- For complex models, there is a greater chance that it was fitted accidentally by errors in data
- Usually, simple models are more robust with respect to noise



"When faced with two equally good hypotheses, always choose the simpler."

How to Address Overfitting

- Pre-Pruning (Early Stopping Rule)
  - Stop the algorithm before it becomes a fully-grown tree
  - Typical stopping conditions for a node:
    - Stop if all instances belong to the same class
    - Stop if none of the attributes provide splitting gain
  - More restrictive conditions:
    - Stop if number of instances in a leaf node is less than some user-specified threshold
    - Stop if the depth of the tree reaches some user-specified threshold
    - Stop if the decreased impurity by expanding the current node is smaller than some user-specified threshold on certain impurity measure (e.g., Gini or information gain)

How to Address Overfitting

- Post-Pruning
  - Grow decision tree to its entirety
  - Trim the nodes of the decision tree in a bottom-up fashion
  - If generalization error improves after trimming, replace sub-tree by a leaf node
  - Class label of leaf node is determined by the majority class of instances in the sub-tree

Pre-Pruning in Sklearn

```
class sklearn.tree.DecisionTreeClassifier(*, criterion='gini', splitter='best', max_depth=None, min_samples_split=2, min_samples_leaf=1, min_weight_fraction_leaf=0.0, max_features=None, random_state=None, max_leaf_nodes=None, min_impurity_decrease=0.0, class_weight=None, ccp_alpha=0.0, monotonic_cst=None)
```

max_depth: *int, default=None*
The maximum depth of the tree.

min_samples_split: *int or float, default=2*
The minimum number of samples required to split an internal node

min_samples_leaf: *int or float, default=1*
The minimum number of samples required to be at a leaf node.

min_impurity_decrease: *float, default=0.0*
A node will be split if this split induces a decrease of the impurity greater than or equal to this value.

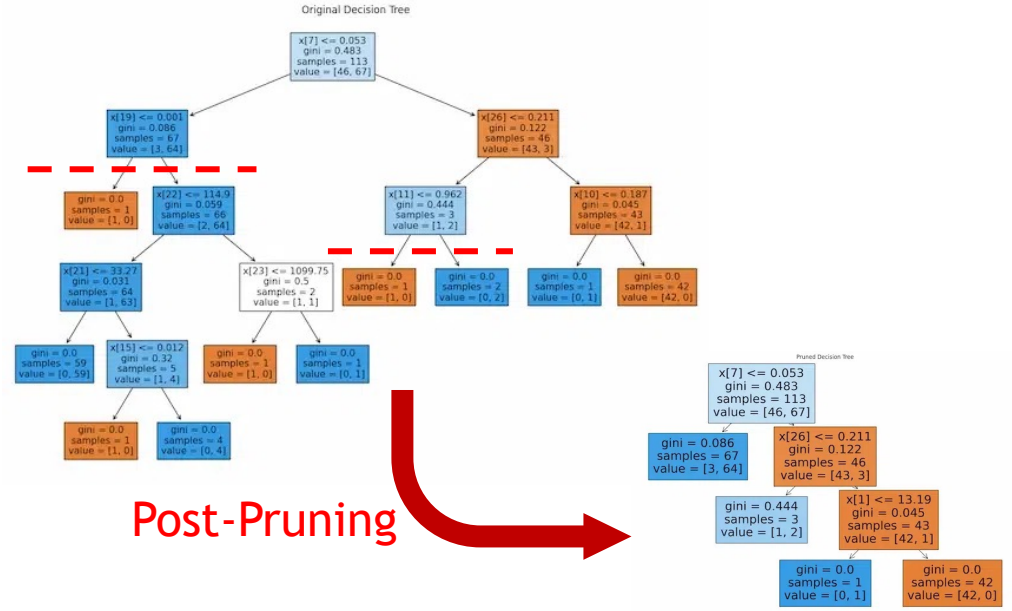
Post-Pruning in Sklearn

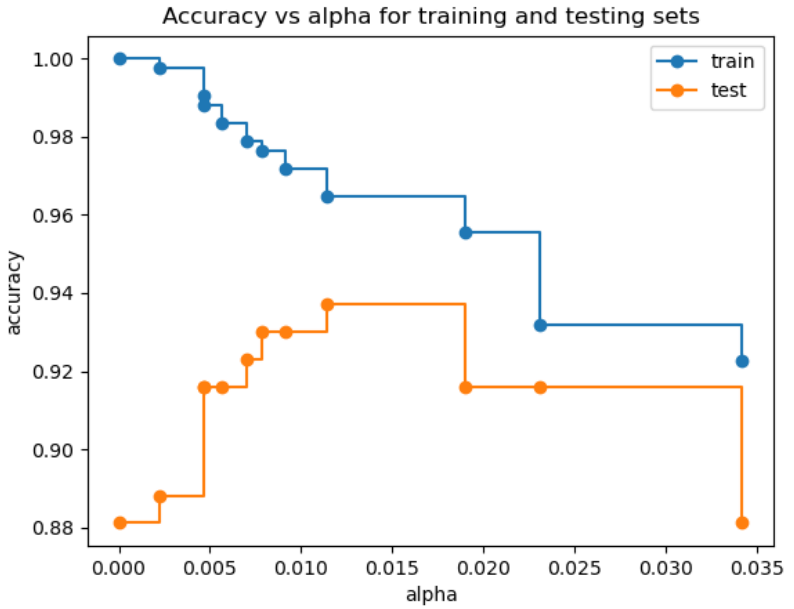
```
class sklearn.tree.DecisionTreeClassifier(*, criterion='gini', splitter='best', max_depth=None, min_samples_split=2, min_samples_leaf=1, min_weight_fraction_leaf=0.0, max_features=None, random_state=None, max_leaf_nodes=None, min_impurity_decrease=0.0, class_weight=None, ccp_alpha=0.0, monotonic_cst=None)
```

```
>>> from sklearn.tree import DecisionTreeClassifier
>>> from sklearn.model_selection import train_test_split
>>> clf = DecisionTreeClassifier(random_state=1)
>>> X_tr, X_te, y_tr, y_te = train_test_split(X, y, train_size=0.9)
>>> path = clf.cost_complexity_pruning_path(X_train, y_train)
>>> ccp_alphas, impurities = path.ccp_alphas, path.impurities

>>> pruned_models = []
>>> for ccp_alpha in ccp_alphas:
>>> ...  pruned_model = DecisionTreeClassifier(ccp_alpha=ccp_alpha)
>>> ...  pruned_model.fit(X_train, y_train)
>>> ...  pruned_models.append(pruned_model)
```

Find the pruned model with the best accuracy on test data

https://scikit-learn.org/stable/modules/tree.html#minimal-cost-complexity-pruning

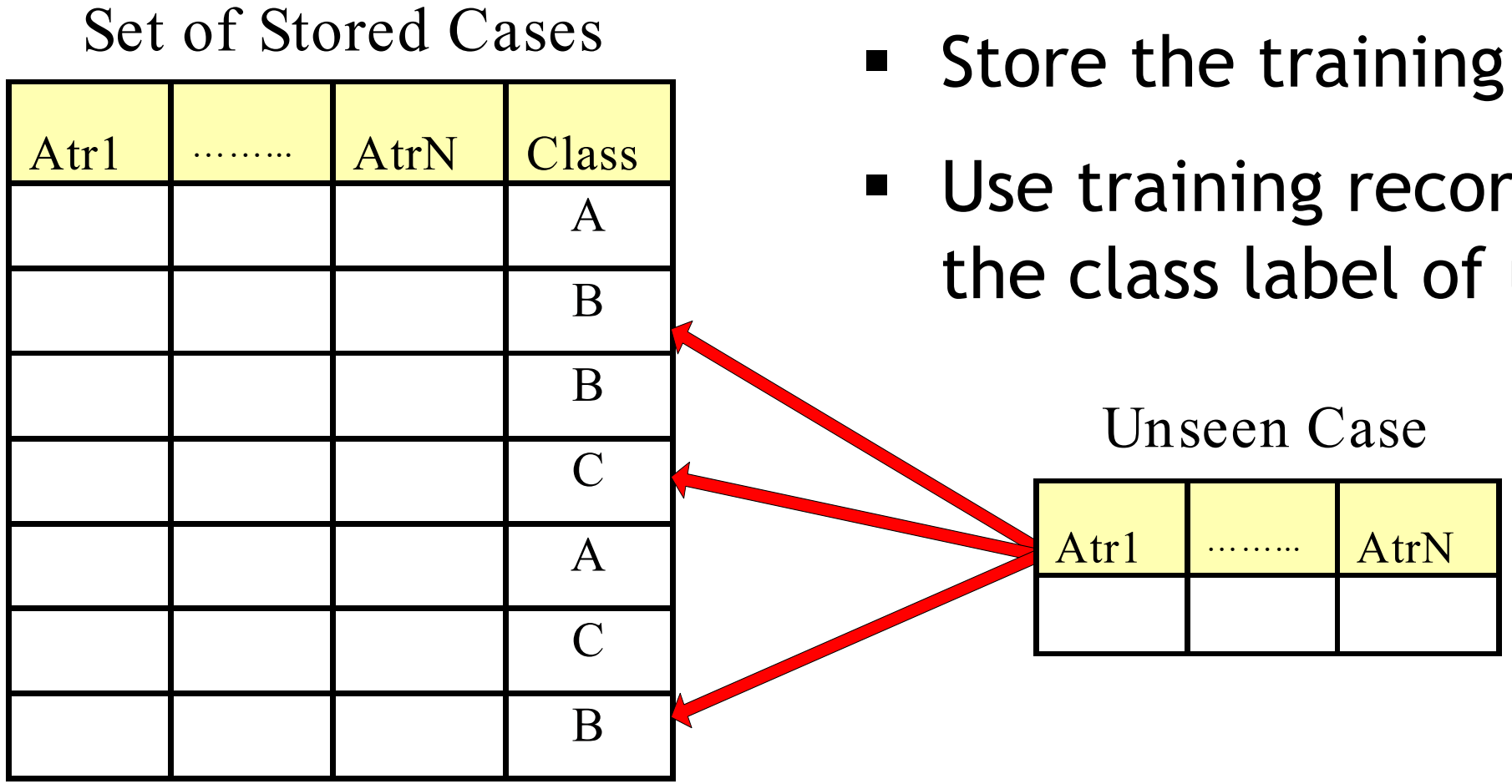




“Final” Notes on Overfitting

- Overfitting results in decision trees/models that are more complex than necessary: after learning knowledge they “tend to learn noise”
- More complex models tend to have more complicated decision boundaries and tend to be more sensitive to noise, missing examples,…
- When learning complex models, large representative training sets are needed
- For small datasets, simple models are often a “good choice”
- In summary, the two approaches to fight overfitting are:
  - Reduce model complexity
  - Increase the size of the training set

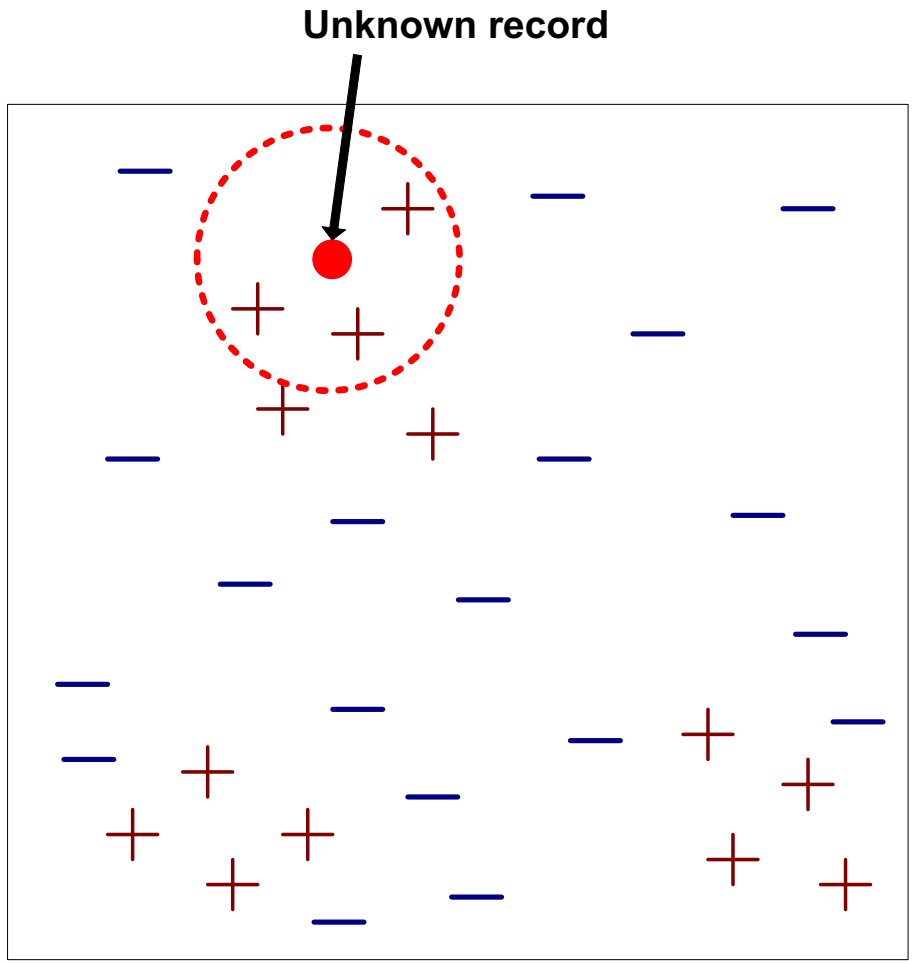
Instance-Based Classifiers



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

Related Topic: Case-based Reasoning (http://en.wikipedia.org/wiki/Case-based_reasoning)

K-Nearest-Neighbor Classifier

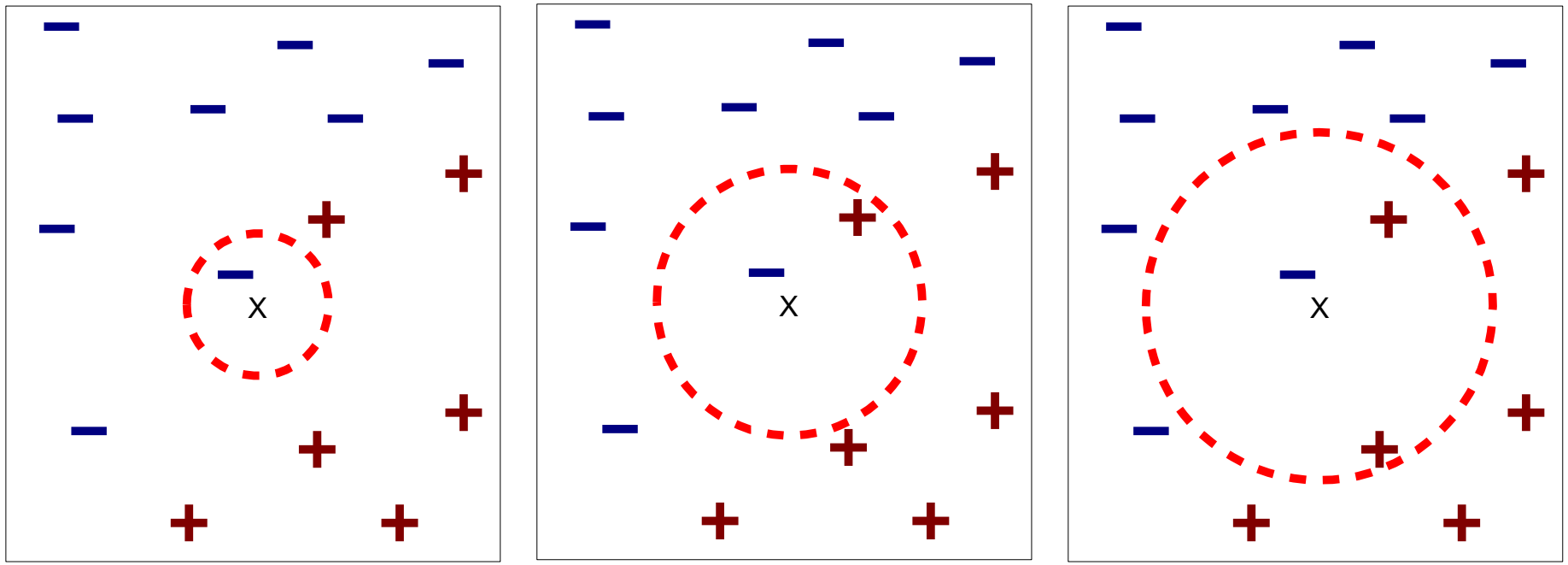


- Requires three things
  - The set of stored records
  - Distance Metric to compute distance between records
  - The value of k, the number of nearest neighbors to retrieve
- To classify an unknown record:
  - Compute distance to other training records
  - Identify k nearest neighbors
  - Use class labels of nearest neighbors to determine the class label of unknown record (e.g., by taking majority vote)

K-Nearest-Neighbor Classifier

- **Distance function**
  - defines similarity (dissimilarity) for pairs of objects
- K: number of neighbors considered
- **Decision Set**: set of k-nearest neighbors considered for classification
- **Decision rule**: how to determine the class of the unseen object from the classes of the decision set?
- The only parameter is K (number of neighbors)
- Weighting of neighbors (e.g. inverse distance)

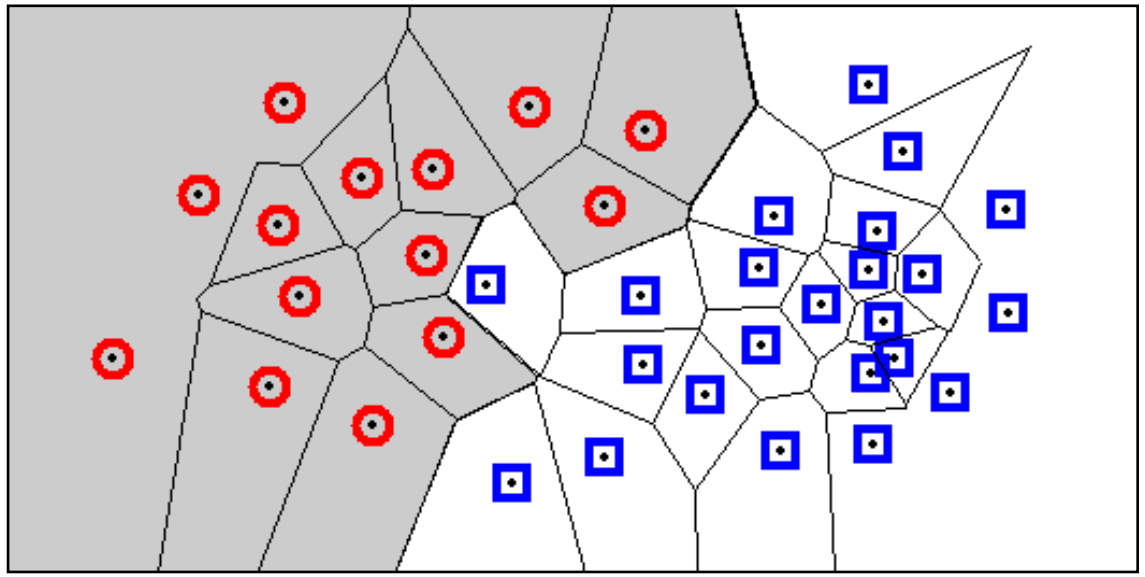
Definition of K-Nearest Neighbor



(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

Voronoi Diagrams for NN-Classifier



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 class 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.

Remarks: Voronoi diagrams can be computed in lower dimensional spaces; infeasible for higher dimensional spaces

K-Nearest-Neighbor Classification

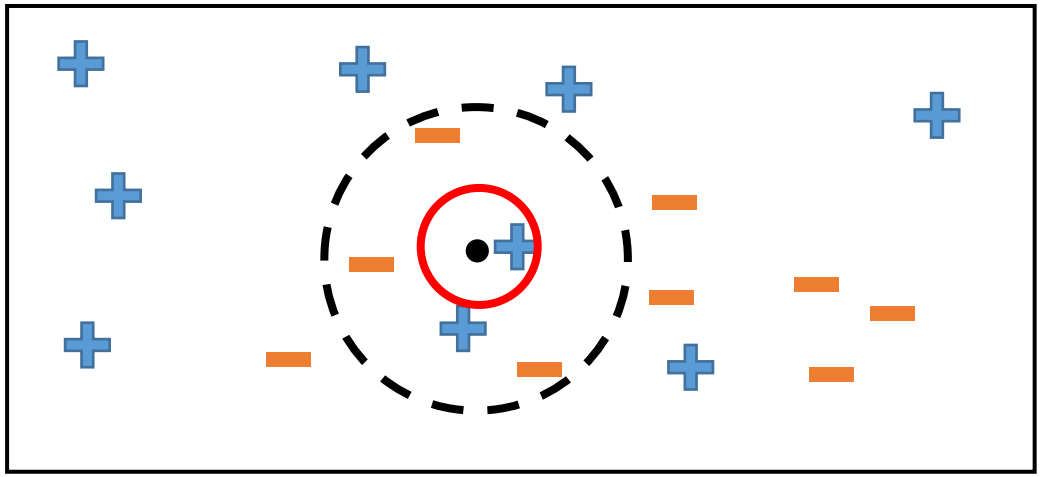
- Compute distance between two points
  - Euclidean distance

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

- Determine the class from nearest neighbor list
  - take the majority vote of class labels among the k-nearest neighbors
  - Weigh the vote according to distance
    - weight factor, $w = 1/d^2$

Decision Rule

- Standard rule
  - Choose the majority class within the decision set
- Weighted decision rule
  - Weight the classes of the decision set
    - By inverse distance
    - By class distribution (often skewed!)
      - class A: 95 %, class B 5 %
      - Decision set = {A, A, A, A, B, B, B}
      - Standard rule → A, Weighted rule → B



Uniform weight for the decision set

- K=1: classification as “+”, k = 5 classification as “-”

Inverse squared distance as weight for the decision set

- K=1 and K=5: classification as “+”

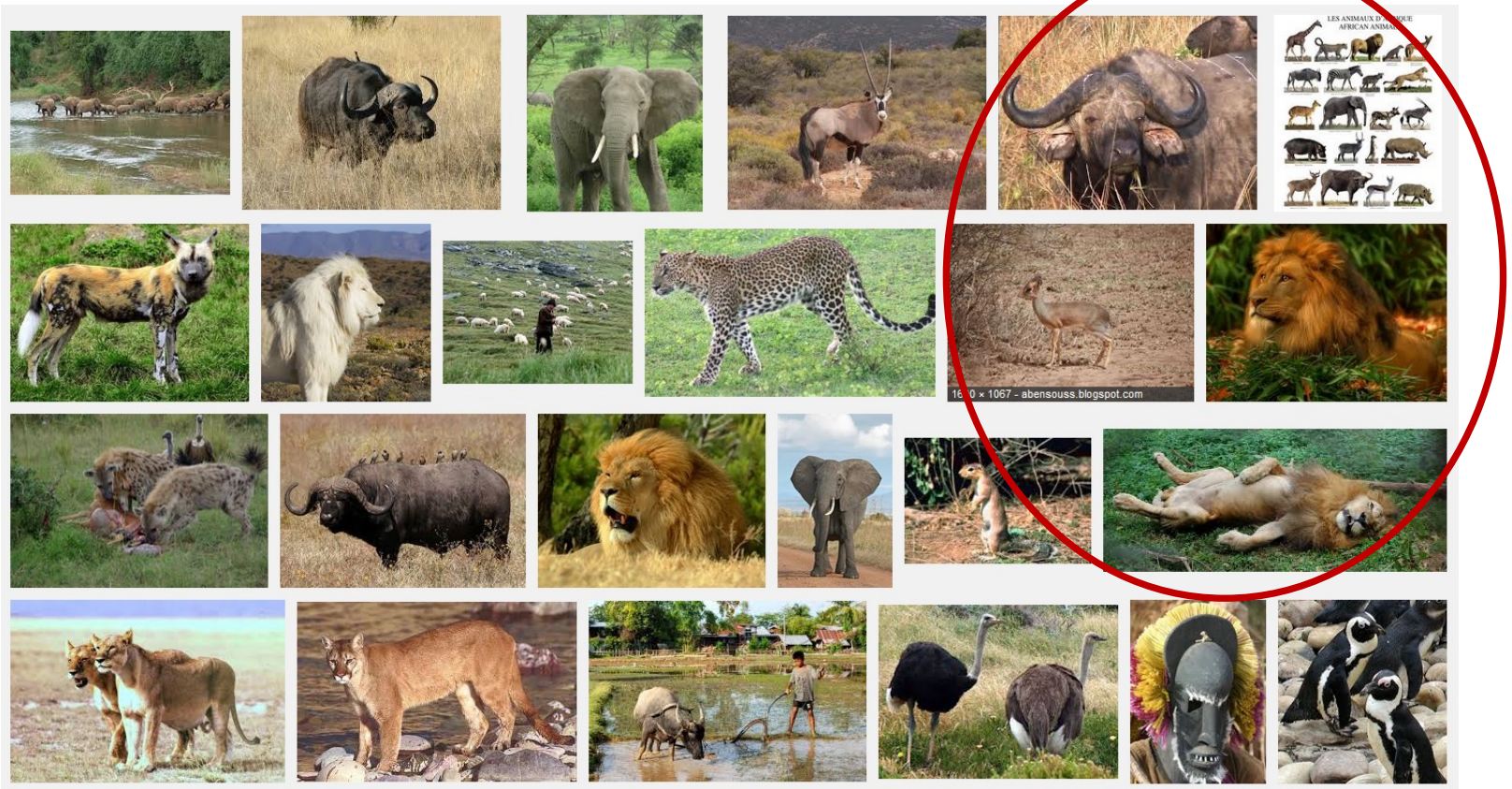
KNN in Image Classification

Top 5 nearest neighbors

- 2 cats
- 1 buffalo
- 1 deer
- 1 lion

The query

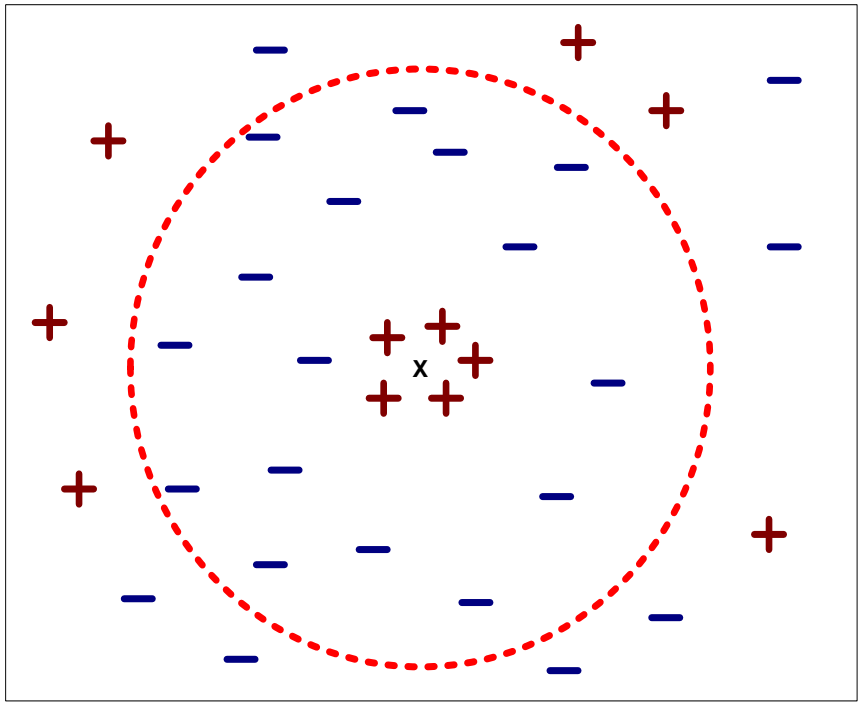
The decision set

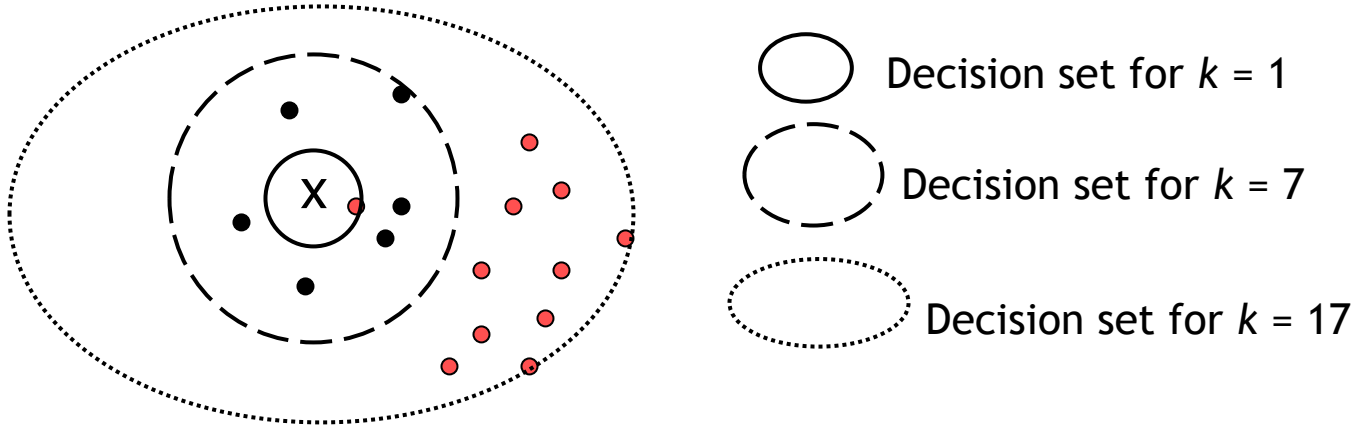


https://bcourses.berkeley.edu/courses/1267848/pages/cs-194-16-introduction-to-data-science-fall-2014

K-Nearest-Neighbor Classification

- 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
  - Medium k: highest classification accuracy, often 2 < k < 10





K-Nearest-Neighbor Classification

- Scaling issues
  - we already introduced z-scores to deal with 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

COSC 3337
Data Science I
Section 14623

Classification (contd.)

Instructor: Jingchao Ni
Fall 2024

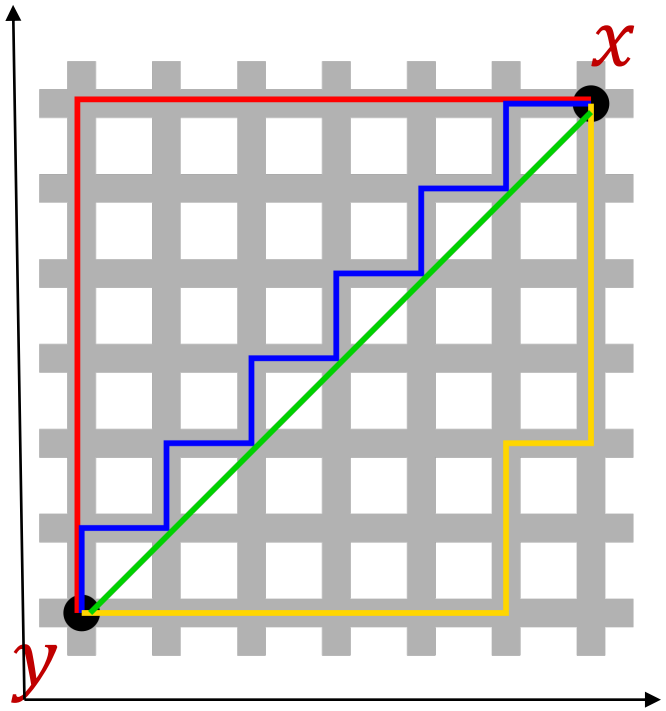
Distance Function

- Manhattan Distance $d(x,y) = \sum_{i=1}^{d} |x_i - y_i|$

- Cosine Distance $d(x,y) = 1 - \frac{\sum_{i=1}^{d} x_i y_i}{\sqrt[2]{\sum_{i=1}^{d} {x_i}^2}\ \sqrt[2]{\sum_{i=1}^{d} {y_i}^2}}$

- Euclidean Distance $d(x,y) = \sqrt[2]{\sum_{i=1}^{d} (x_i - y_i)^2}$

- Minkowski Distance $d(x,y) = \left(\sum_{i=1}^{d} |x_i - y_i|^p\right)^{\frac{1}{p}}$



Distance Function

- How to get a good distance function?
  - Seeks an “optimal” distance metric that maximizes the purity of a clustering result / the accuracy of a KNN-classifier
  - Distance Metric Learning (Mahalanobis distance)

$$d(x, y) = \sqrt{(x - y)^T M (x - y)}$$

A d-by-d parameter matrix

  - Solving M by optimizing a classification problem on a training dataset

KNN Classifier in Sklearn

```
class sklearn.neighbors.KNeighborsClassifier(n_neighbors=5, *,
weights='uniform', algorithm='auto', leaf_size=30, p=2, metric=
'minkowski', metric_params=None, n_jobs=None)
```

weights: *{'uniform', 'distance'}, callable or None, default='uniform'*
Weight function used in prediction.

metric: *str or callable, default='minkowski'*
Metric to use for distance computation.

```
>>> from sklearn.neighbors import KNeighborsClassifier
>>> from sklearn.model_selection import train_test_split
>>> clf = KNeighborsClassifier(n_neighbors=3)
>>> X_tr, X_te, y_tr, y_te = train_test_split(X, y,
train_size=0.9)
>>> clf.fit(X_tr, y_tr)
>>> y_te = clf.predict(X_te)
>>> prob = clf.predict_proba(X_te)
```

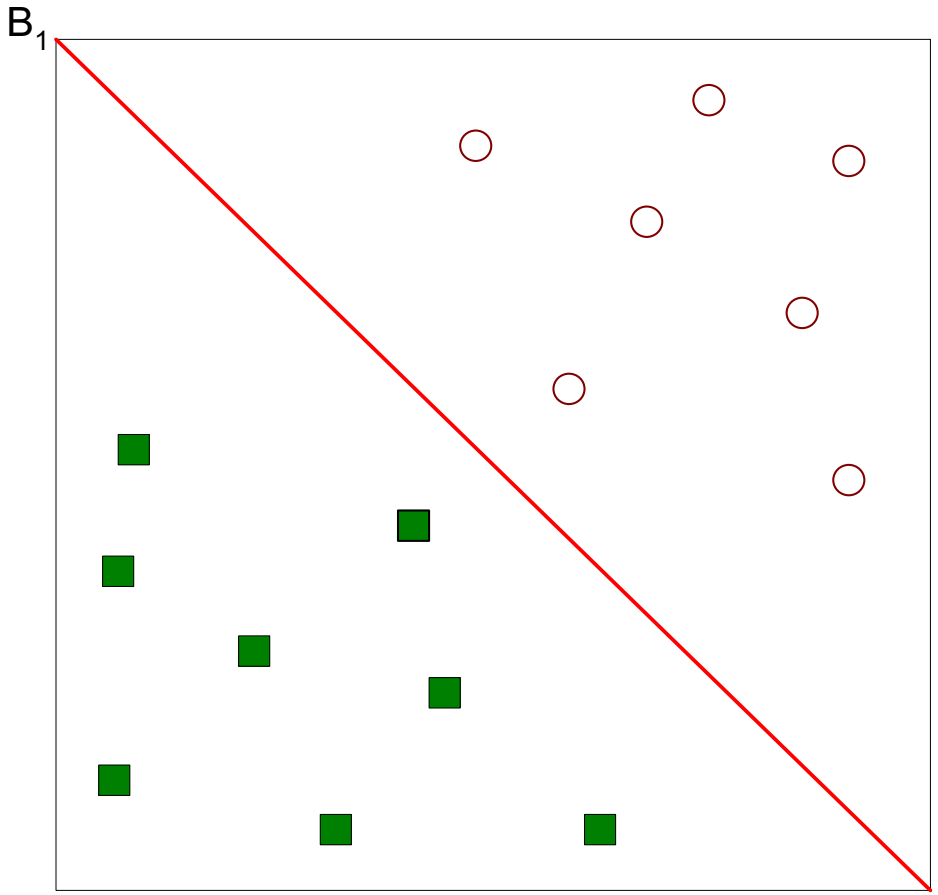
Probability is the proportion of the majority class in K nearest neighbors

metric	Function
'cityblock'	metrics.pairwise.manhattan_distances
'cosine'	metrics.pairwise.cosine_distances
'euclidean'	metrics.pairwise.euclidean_distances
'haversine'	metrics.pairwise.haversine_distances
'l1'	metrics.pairwise.manhattan_distances
'l2'	metrics.pairwise.euclidean_distances
'manhattan'	metrics.pairwise.manhattan_distances
'nan_euclidean'	metrics.pairwise.nan_euclidean_distances

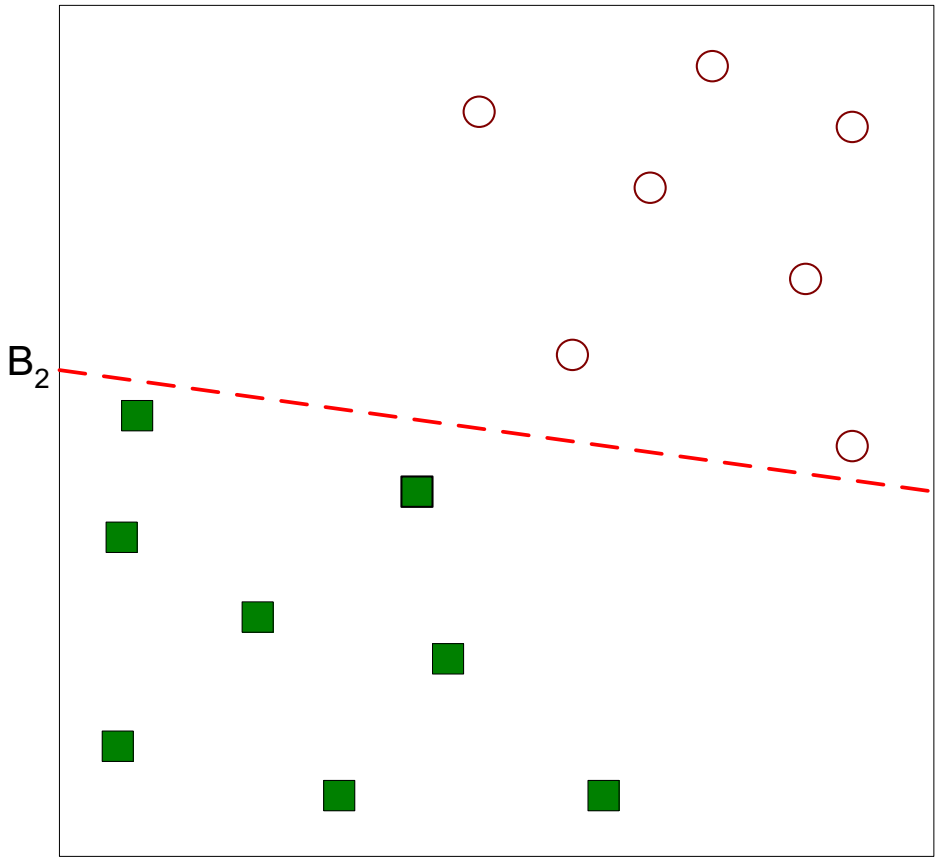
Summary of KNN Classifier

- k-NN classifiers are lazy learners
  - Unlike eager learners such as decision tree induction and rule-based systems, it does not build models explicitly
  - Classifying unknown records is relatively expensive
- k-NN classifiers rely on a distance function; the quality of the distance function is critical for the performance of a K-NN classifier
- k-NN classifiers can obtain high accuracies and are quite popular in some fields, such as text data mining and in information retrieval, and it provides interpretation

Support Vector Machine (SVM)

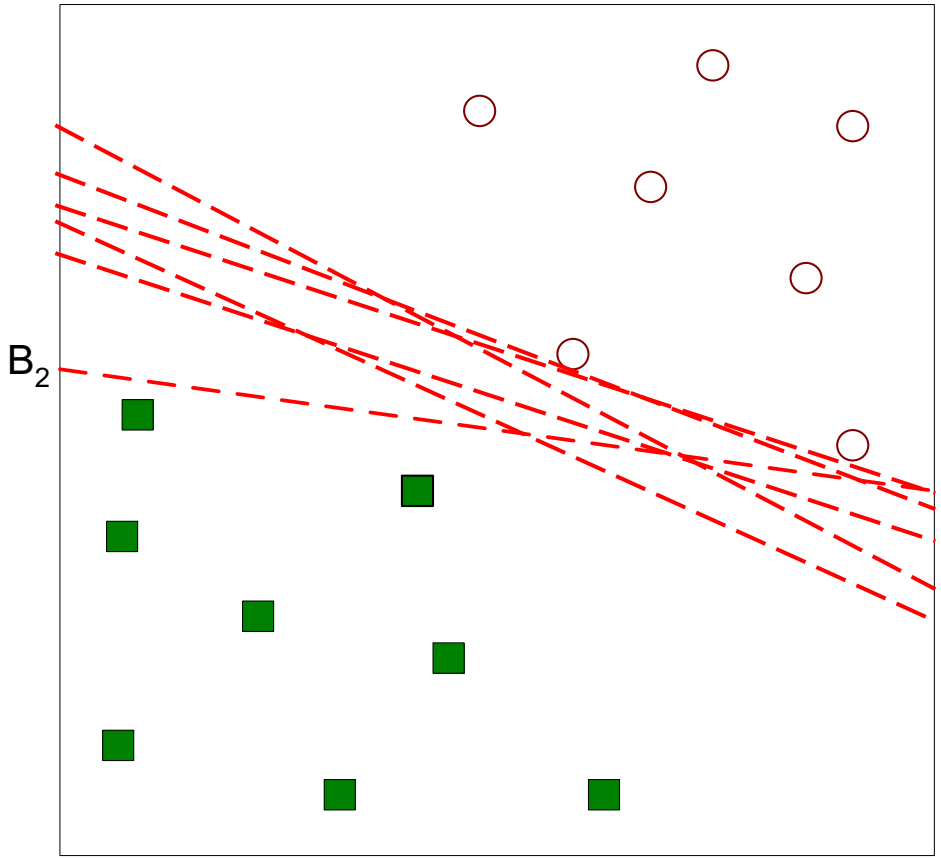


One Possible Solution

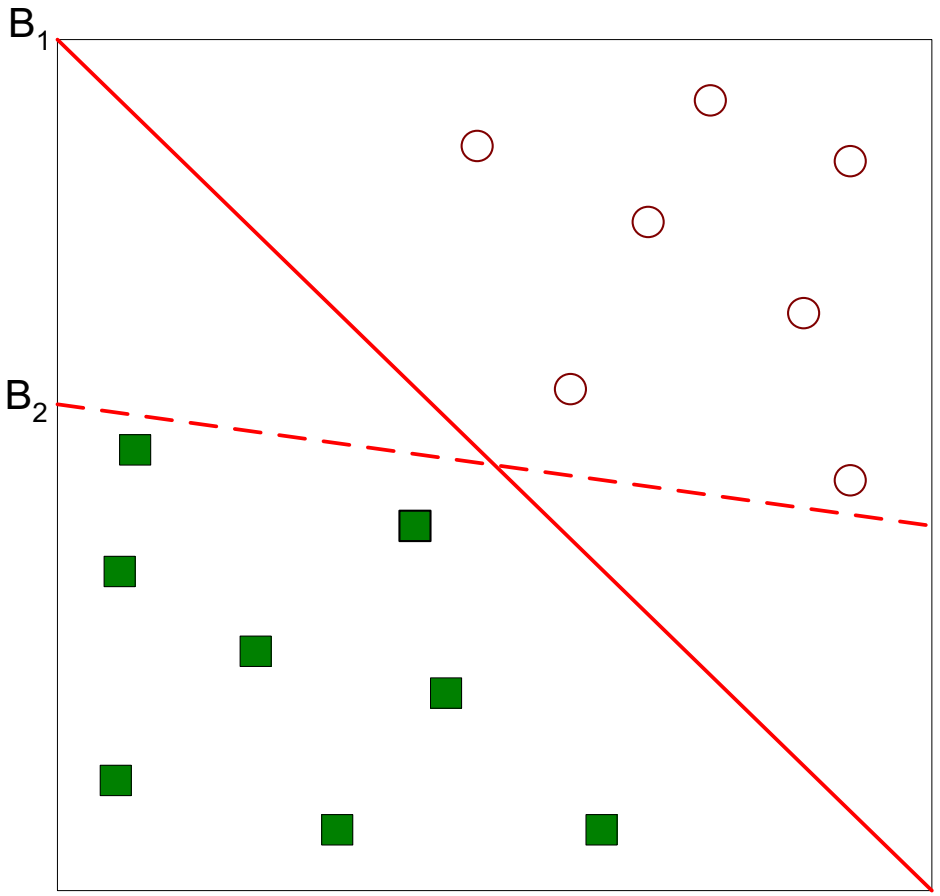


Another possible solution

Support Vector Machine (SVM)

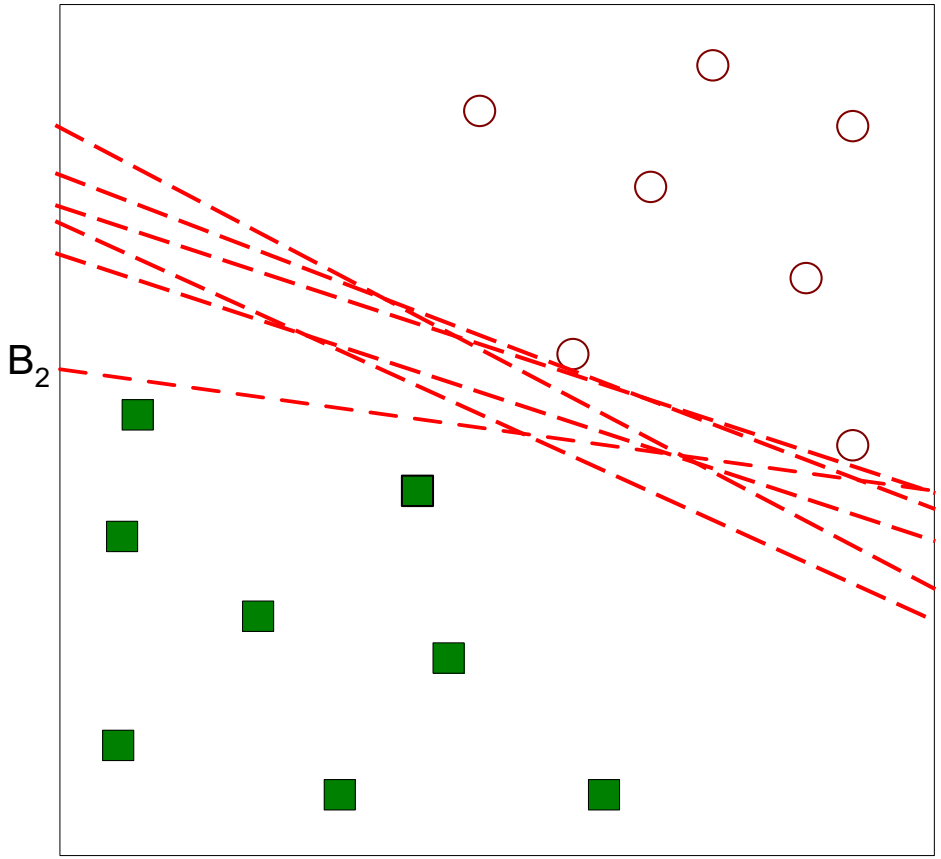


Other possible solutions

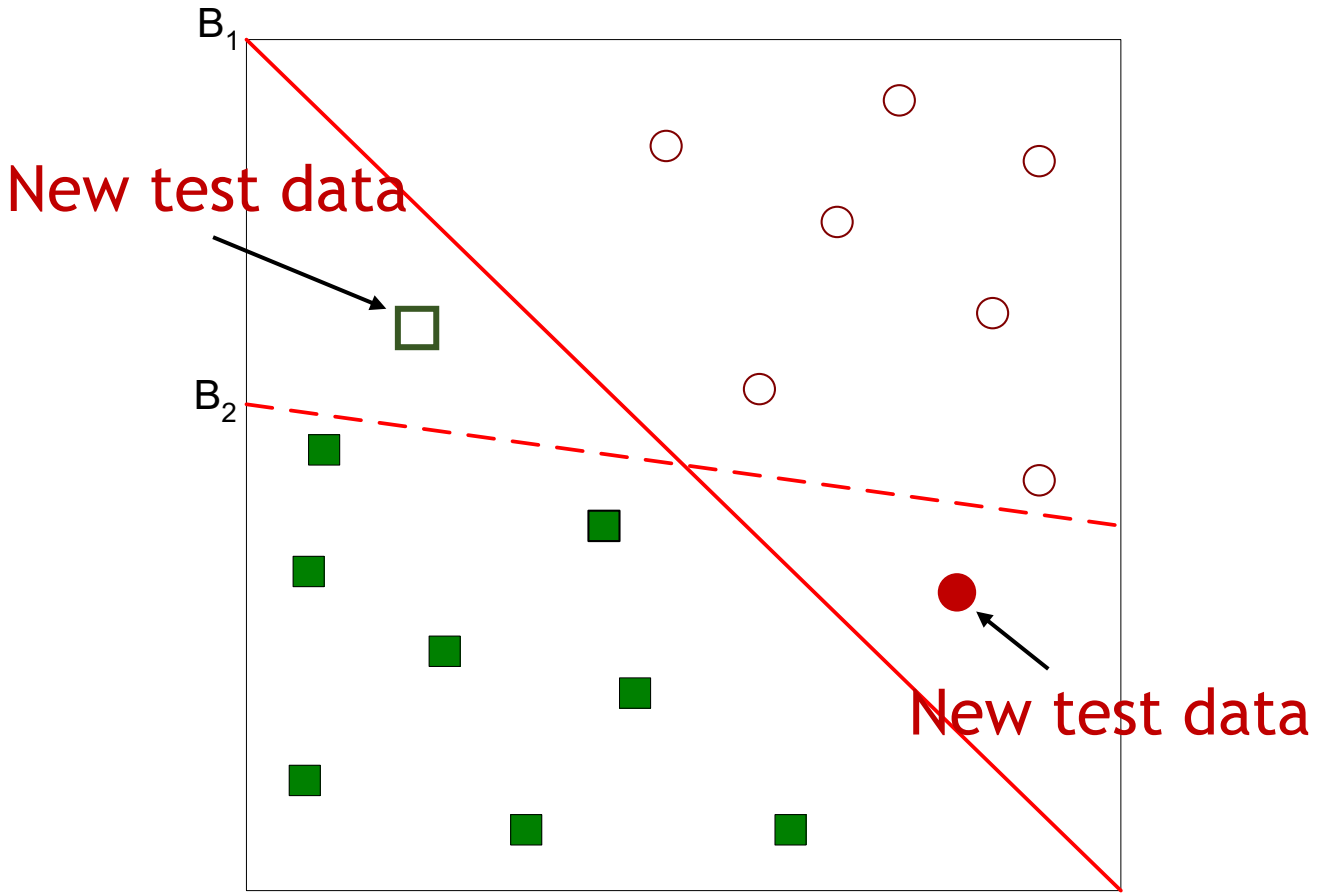


- Which one is better? B1 or B2?
- How do you define better?

Support Vector Machine (SVM)

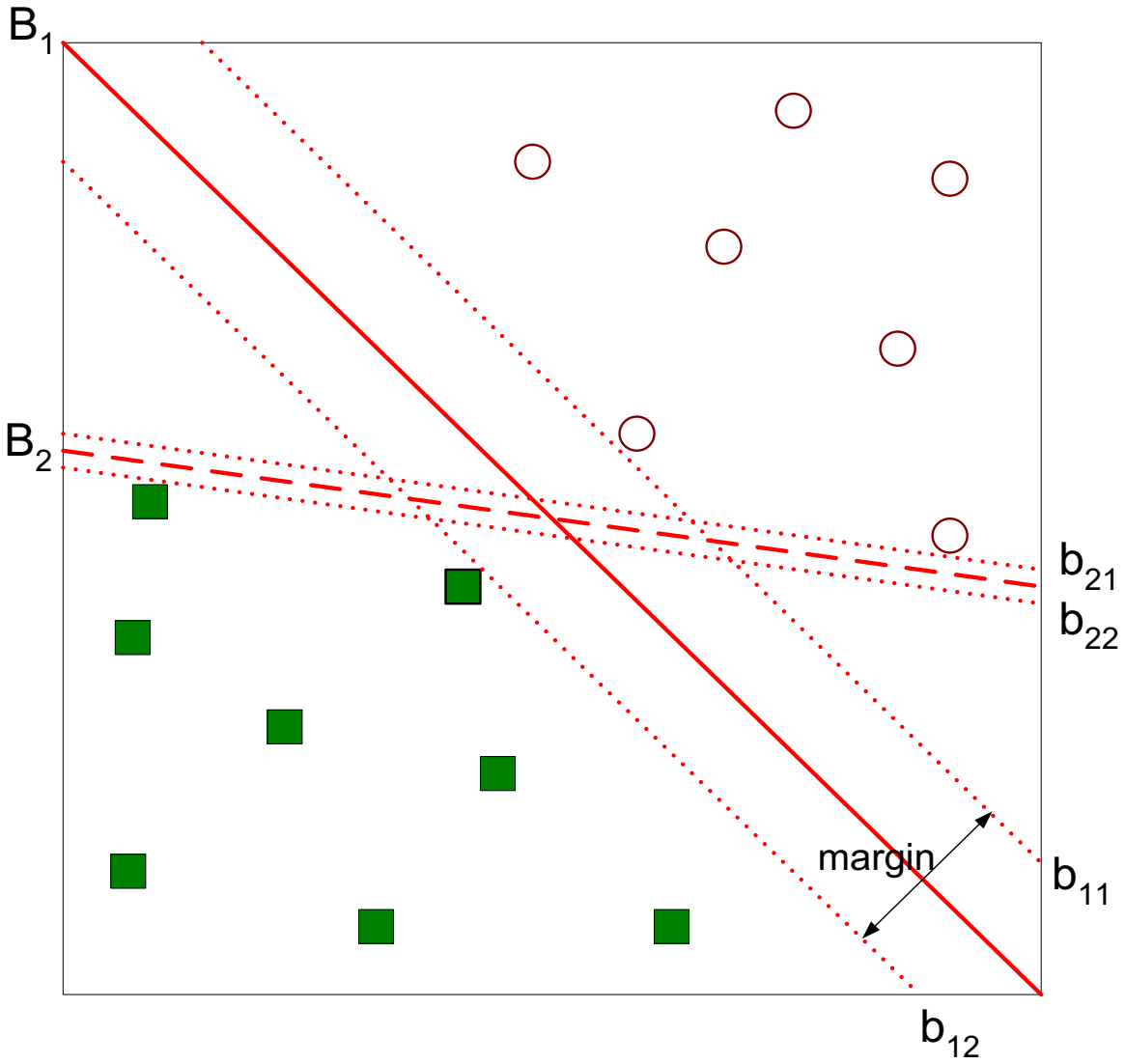


Other possible solutions



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

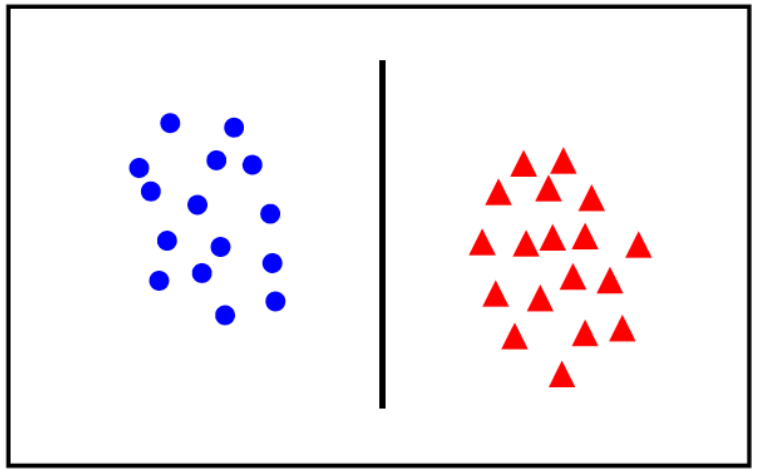
Support Vector Machine (SVM)

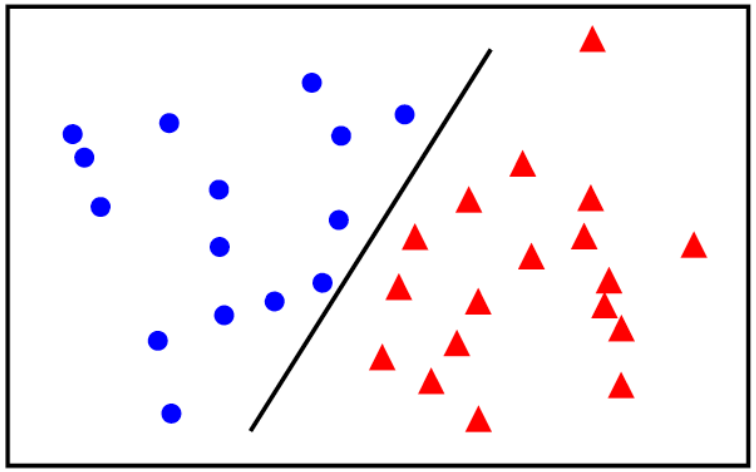


- Find a hyperplane that maximizes the margin between classes
- B1 is better than B2
- Maximum-margin hyperplane
  - Best generalization
  - Most stable under perturbations of the inputs

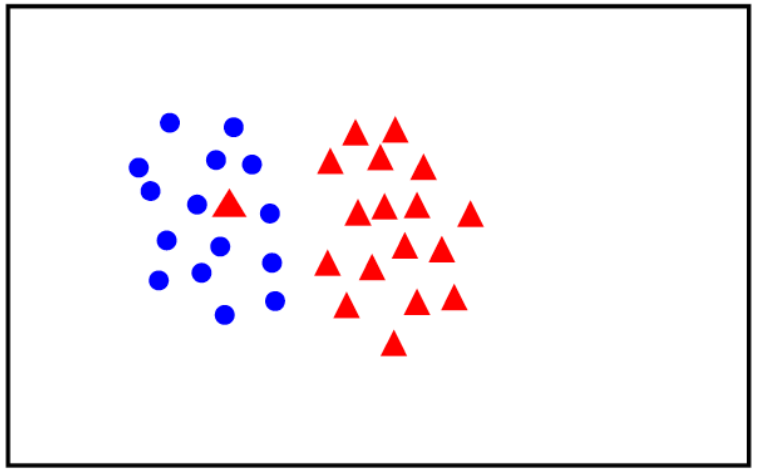
Linear Separability

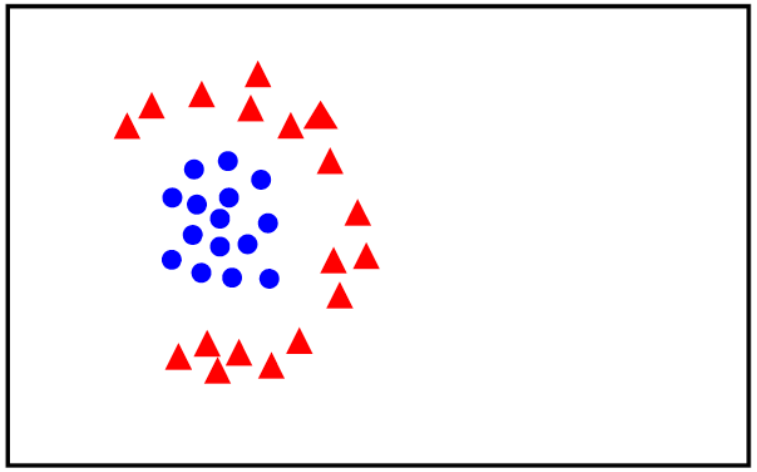
Linearly Separable





Not Linearly Separable



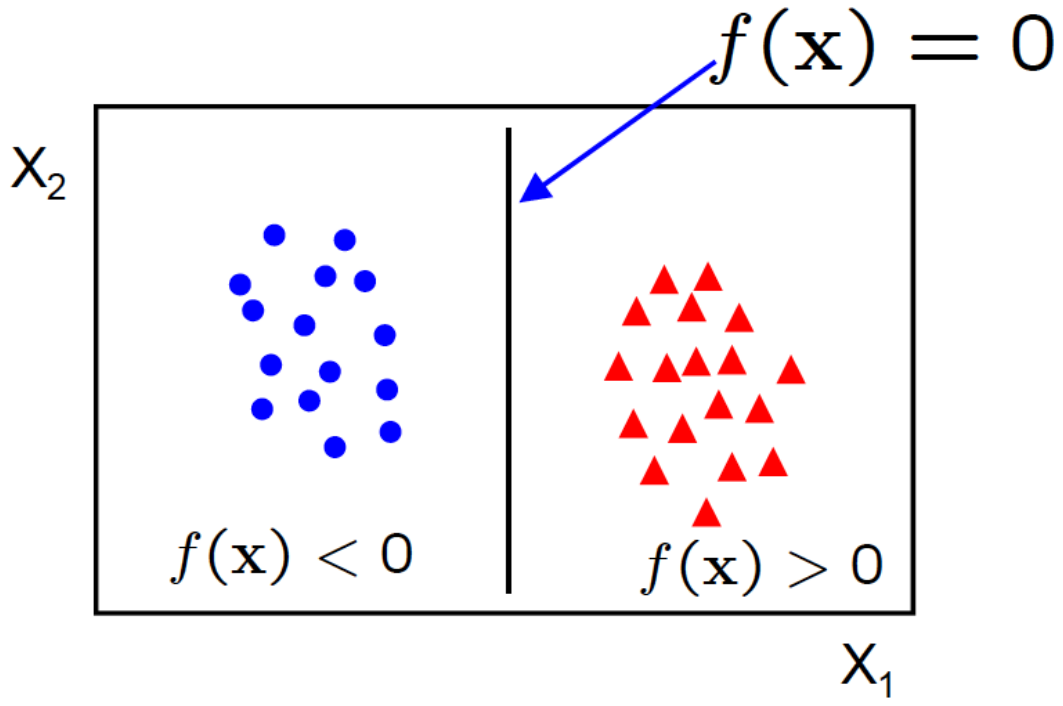


Linear Classifiers

- A linear classifier has the form

$$f(\mathbf{x}) = \mathbf{w}^\top \mathbf{x} + b$$

  - In 2D space, the boundary is a line
  - w is the weight vector
  - b is a scalar called bias



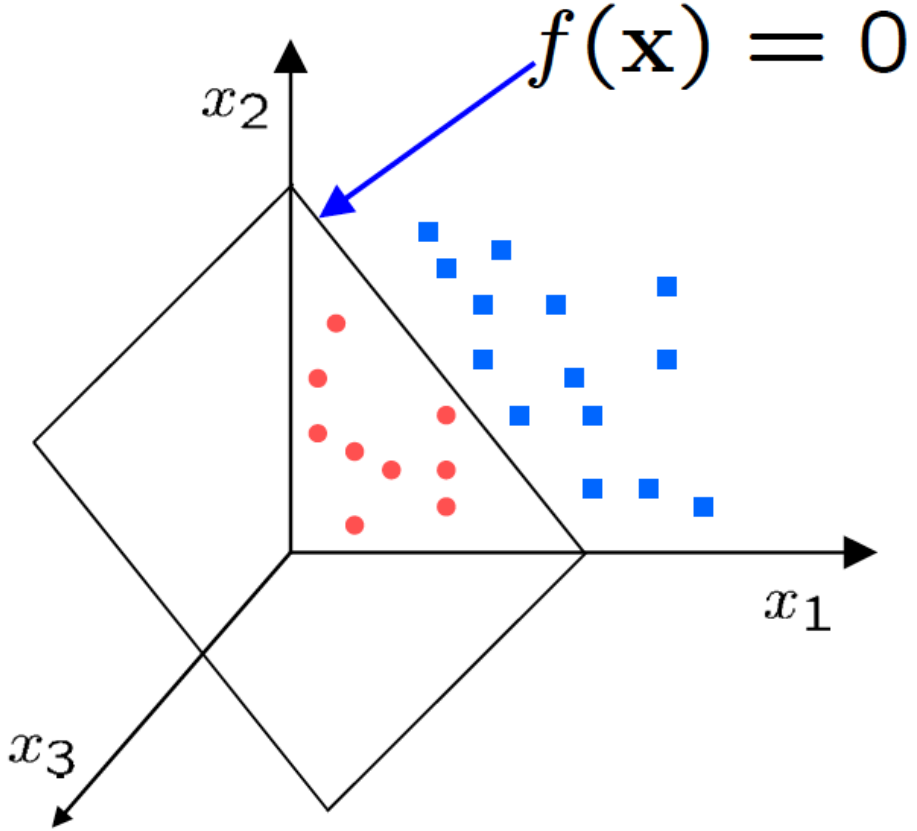
Linear Classifiers

- A linear classifier has the form

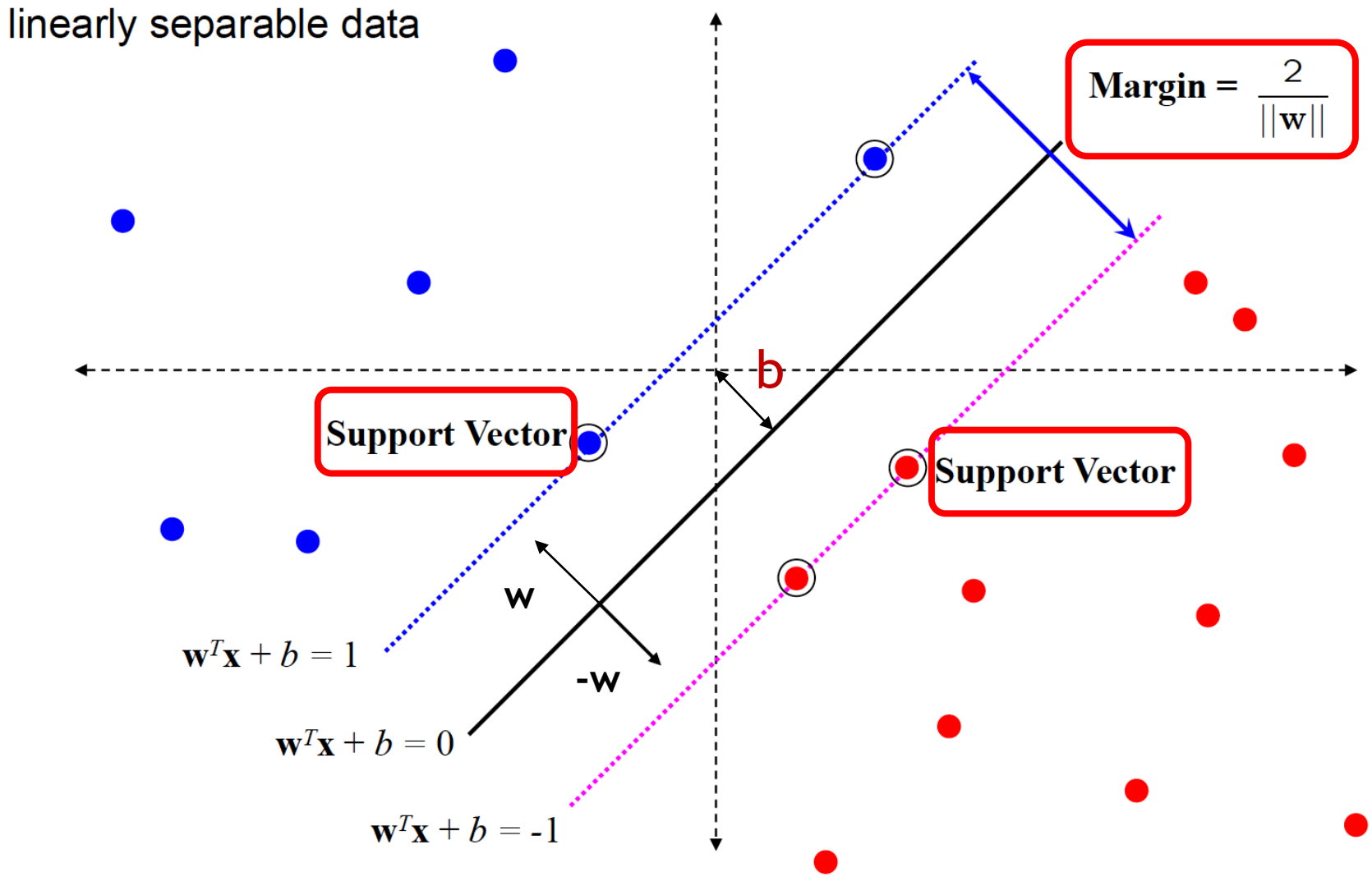
$$f(\mathbf{x}) = \mathbf{w}^\top \mathbf{x} + b$$

- In 3D space, the boundary is a plane
- In nD space, the boundary is a hyperplane

- For KNN classifier, it is necessary to "carry" the training data
- For a linear classifier, the training data is used to learn w and b, and then discarded
- Only w, b are needed for classifying new data



Support Vector Machine (SVM)



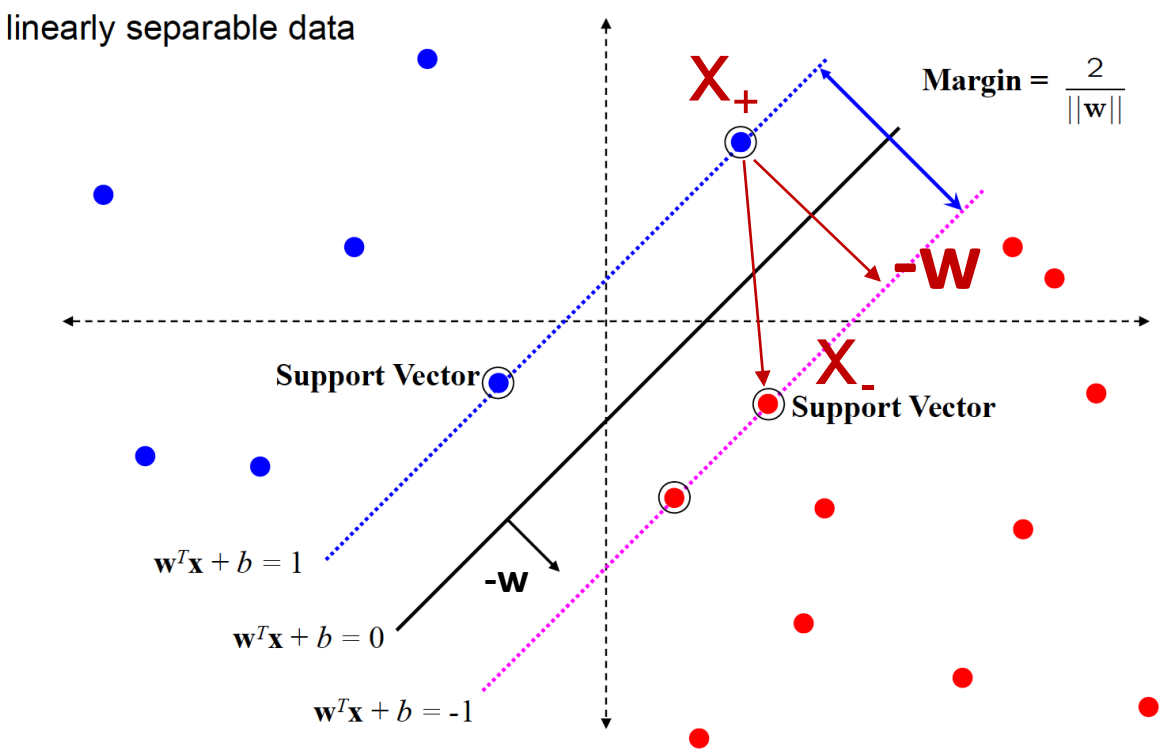
Examples are:
$(x_1, y_1), \ldots, (x_n, y_n)$

with
$y_i \in \{-1, +1\}$

w, b are parameters of the linear classifier

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

Derivation of Margin



Let x_+ be a positive example on blue plane

$$\mathbf{w}^T\mathbf{x}_+ + b = 1$$

Let x_- be a negative example on red plane

$$\mathbf{w}^T\mathbf{x}_- + b = -1$$

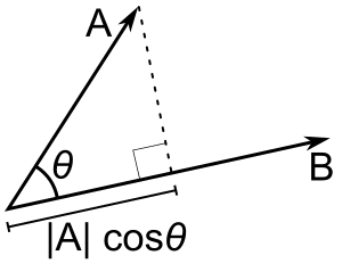
Then margin is a scalar projection of $(x_- - x_+)$ on -w

$$\text{margin} = (\mathbf{x}_- - \mathbf{x}_+)^T \left(-\frac{\mathbf{w}}{\|\mathbf{w}\|}\right) \quad \text{scalar projection}$$

$$= \frac{\mathbf{x}_+{}^T\mathbf{w} - \mathbf{x}_-{}^T\mathbf{w}}{\|\mathbf{w}\|}$$

$$= \frac{1 - b - (-1 - b)}{\|\mathbf{w}\|}$$

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



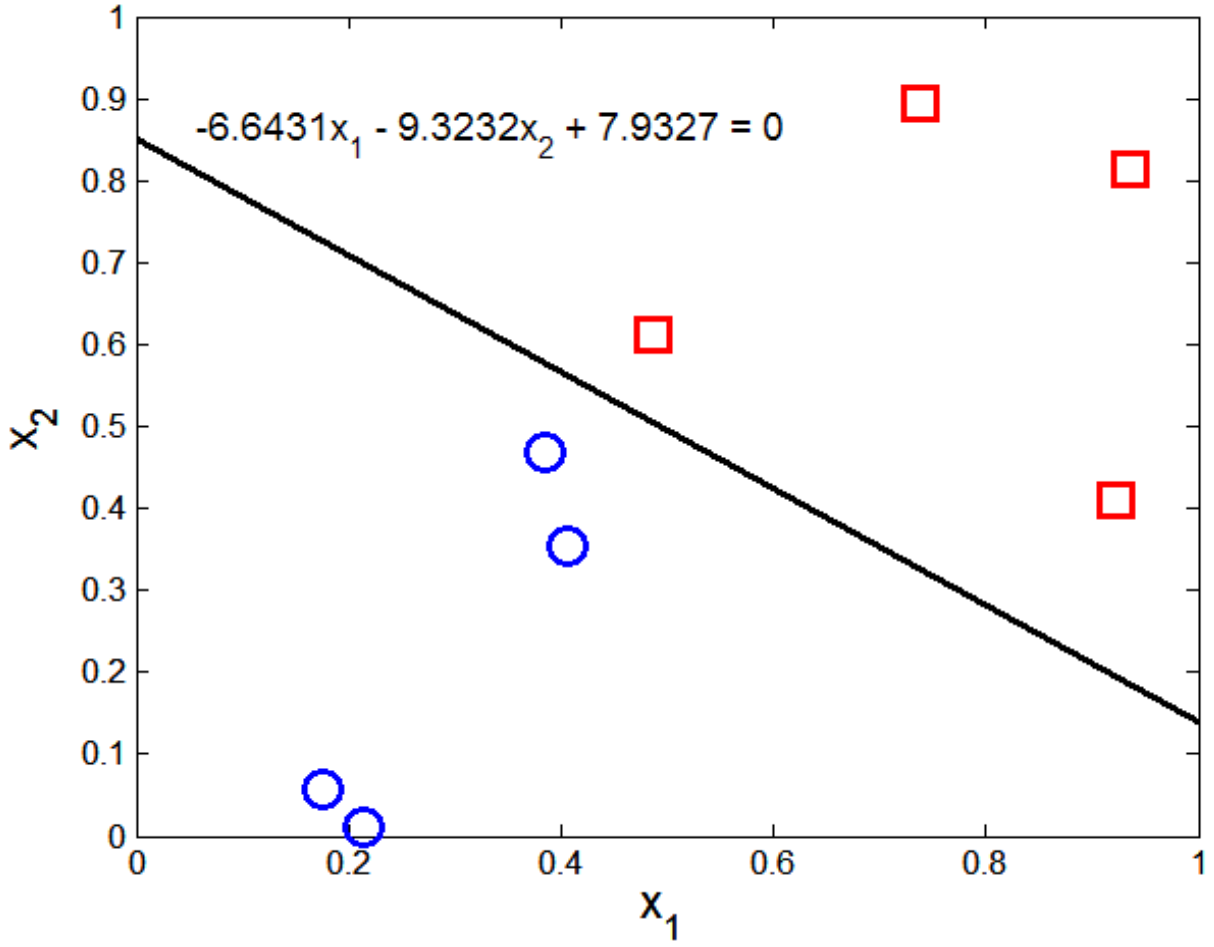
Learning Linear SVM

- The objective is to maximize $\text{Margin} = \frac{2}{\| \vec{w} \|}$
- Which is equivalent to minimize $L(\vec{w}) = \frac{\| \vec{w} \|^2}{2}$
- Subject to the following constraints

$$y_i = \begin{cases} 1 & \text{if } \vec{w} \bullet \vec{x}_i + b \geq 1 \\ -1 & \text{if } \vec{w} \bullet \vec{x}_i + b \leq -1 \end{cases} \quad \text{or} \quad y_i(\mathbf{w} \bullet \mathbf{x}_i + b) \geq 1, \quad i = 1,2,\ldots,N$$

  - This is a constrained convex quadratic optimization problem that can be solved in polynominal time
  - Numerical approaches to solve it (e.g., quadratic programming) exist
  - The function to be optimized has a unique minimum -> no local minimum problem

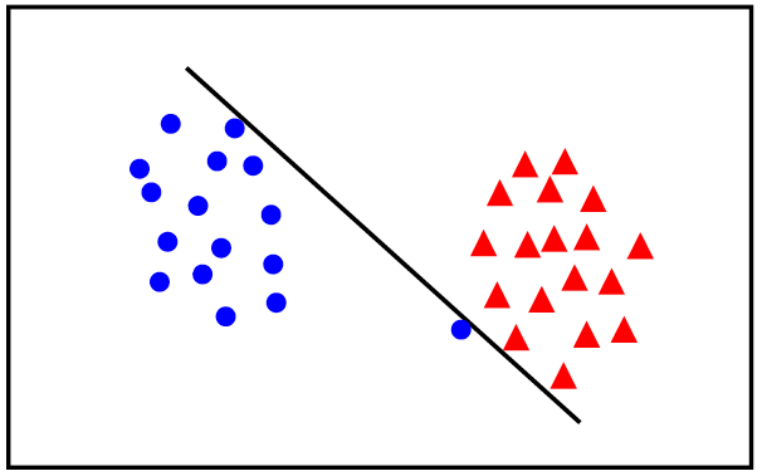
Example of Linear SVM

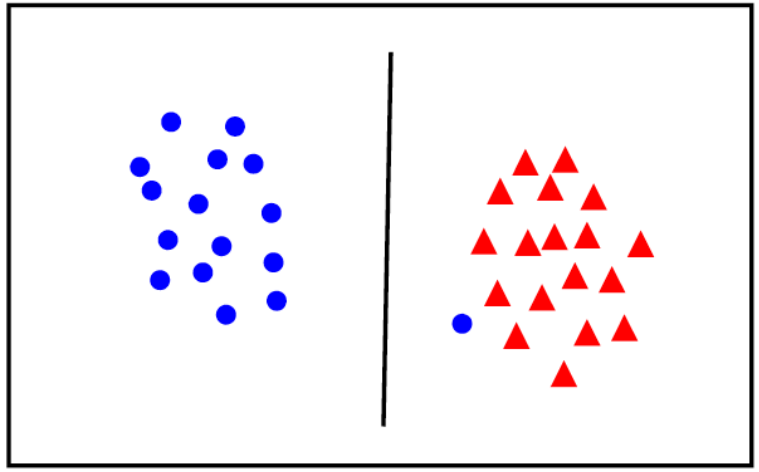


x1	x2	y
0.3858	0.4687	1
0.4871	0.611	-1
0.9218	0.4103	-1
0.7382	0.8936	-1
0.1763	0.0579	1
0.4057	0.3529	1
0.9355	0.8132	-1
0.2146	0.0099	1

Support vectors

Linear Separability: Best Boundary





- The points can be linearly separated but there is a very narrow margin

- A larger margin solution is better, even though one constraint is violated
  - In general, there is a trade-off between the margin and the number of mistakes on the training data

Introduce "Slack" Variables

- The optimization problem becomes $L(w) = \frac{\| \vec{w} \|^2}{2} + C\left(\sum_{i=1}^{N} \xi_i \right)$

Inverse size of margin between hyperplanes

Hyperparameter

Slack variable: parameter

- Subject to (i=1, …, N) $\begin{cases} (1) & y_i * (\vec{w} \bullet \vec{x}_i + b) \geq 1 - \xi_i \\ (2) & 0 \leq \xi_i \end{cases}$

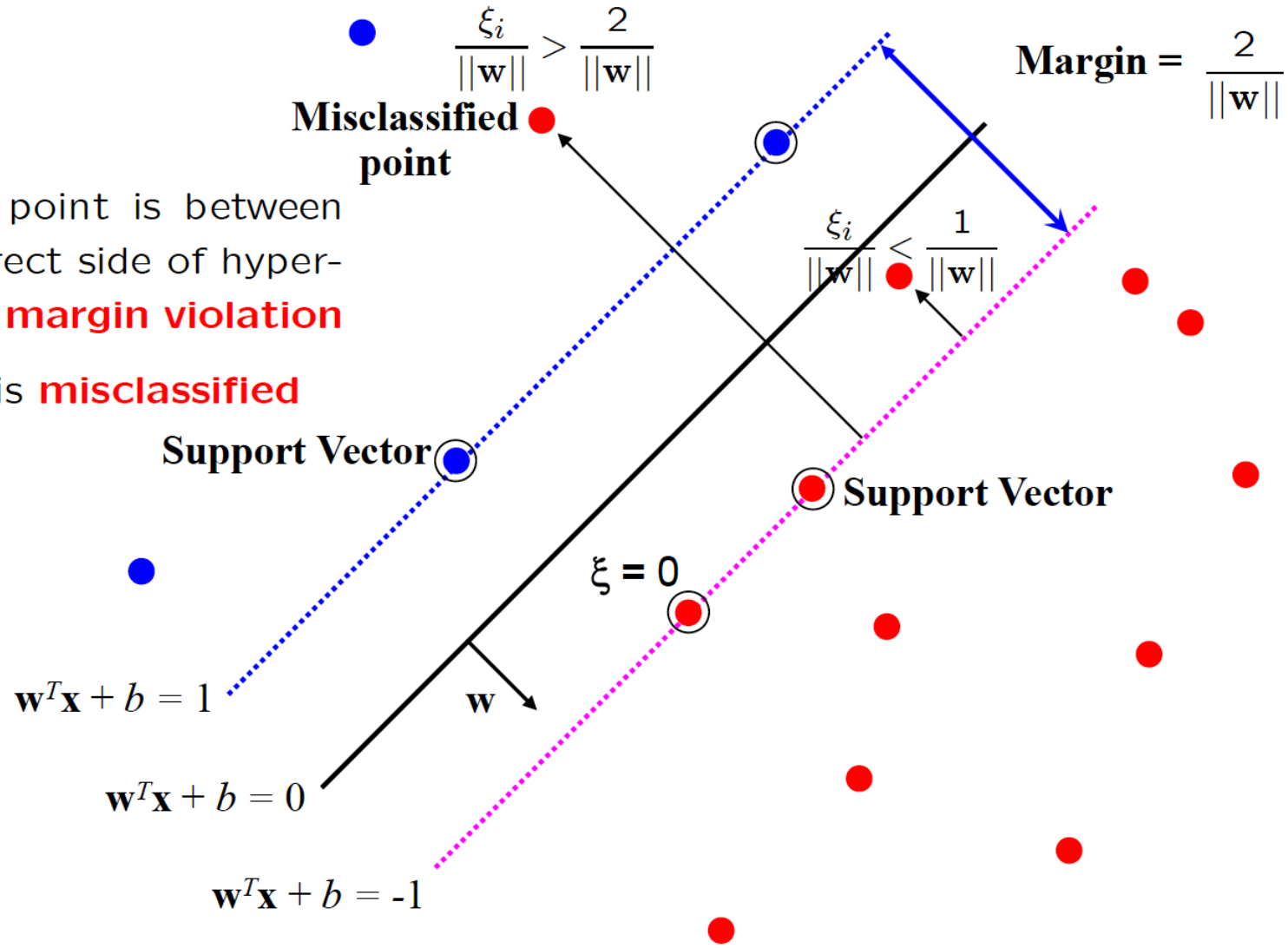
Allows constraint violation to a certain degree -> soft margin

- Every constraint can be satisfied if ξ_i is sufficiently large -> soft margin
- C is a regularization parameter (hyperparameter):
  - Small C allows constraints to be easily ignored -> large margin
  - Large C makes constraints hard to ignore -> narrow margin
  - $C = \infty$ enforces all constraints: hard margin
- This is still a quadratic optimization problem and there is a unique minimum. Note, there is only one hyperparameter, C (chosen using a validation set)

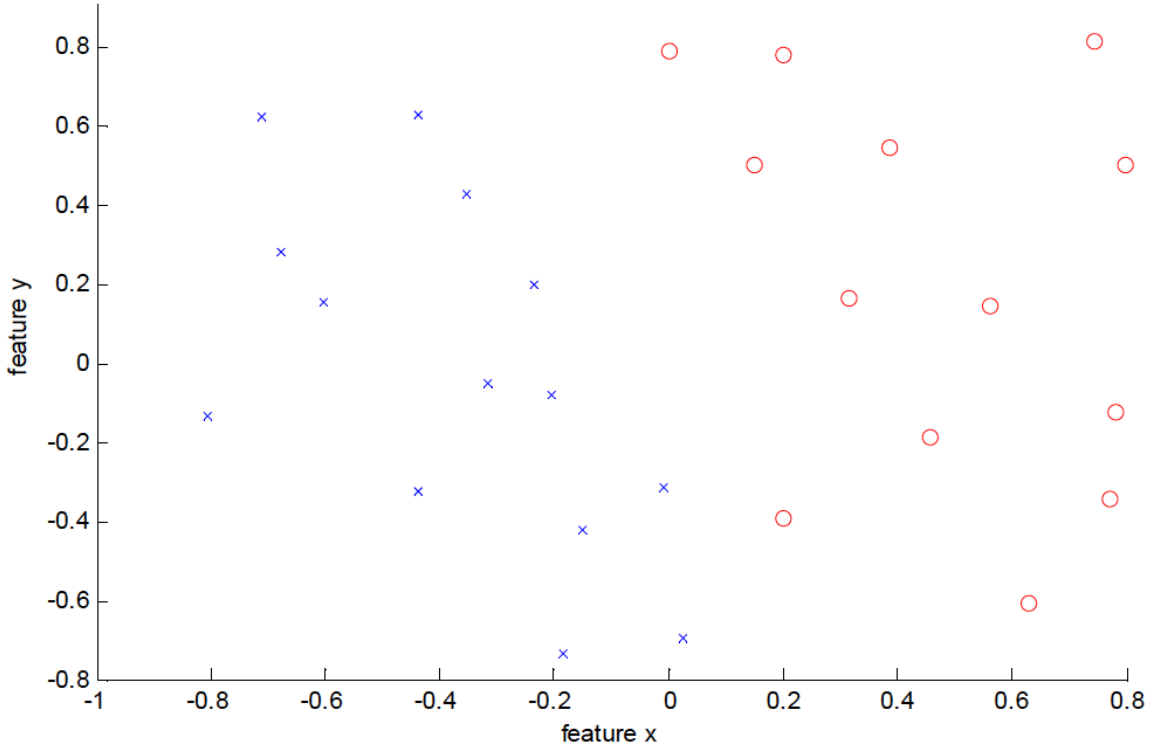
Introduce “Slack” Variables

$$\xi_i \geq 0$$

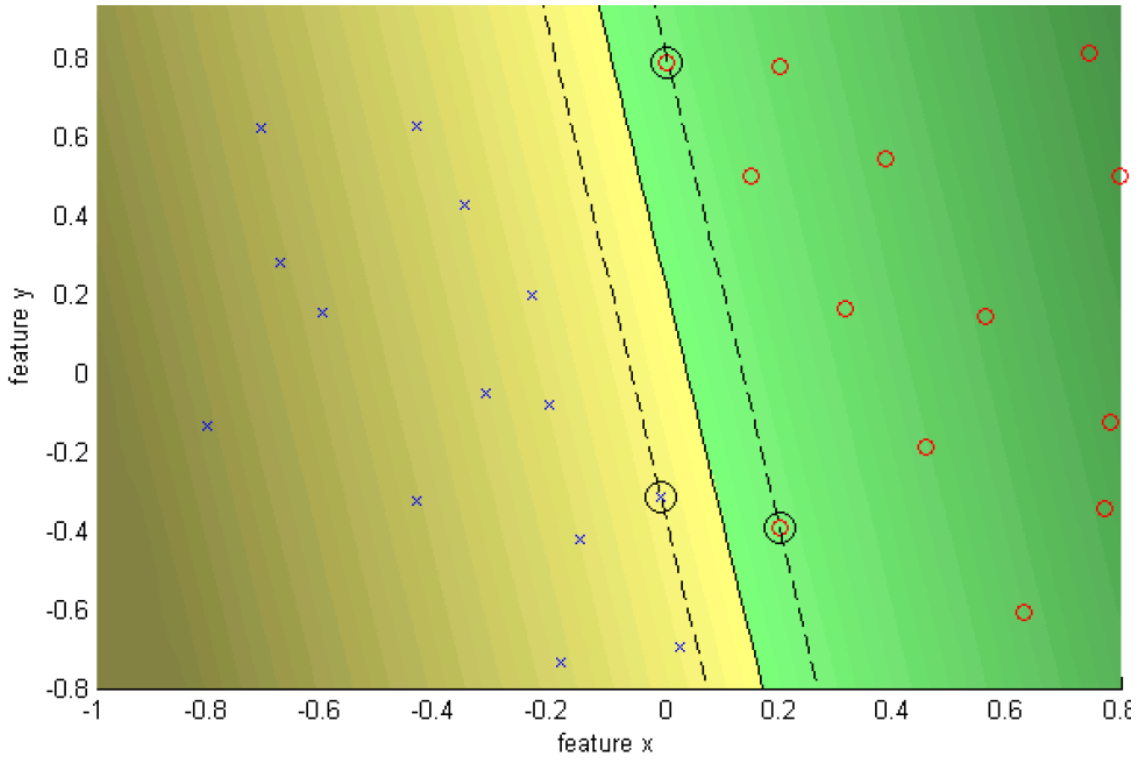
- for $0 < \xi \leq 1$ point is between margin and correct side of hyper-plane. This is a **margin violation**
- for $\xi > 1$ point is **misclassified**



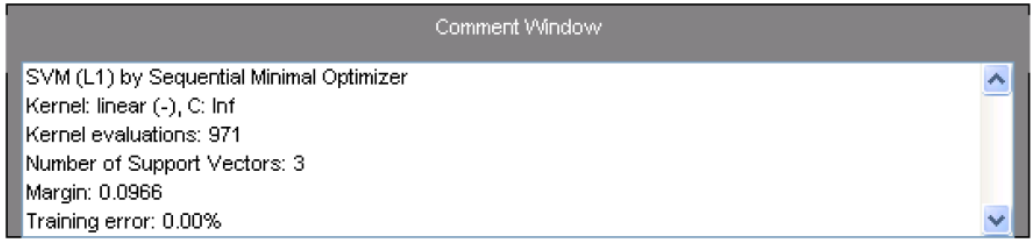
Example of Hard Margin



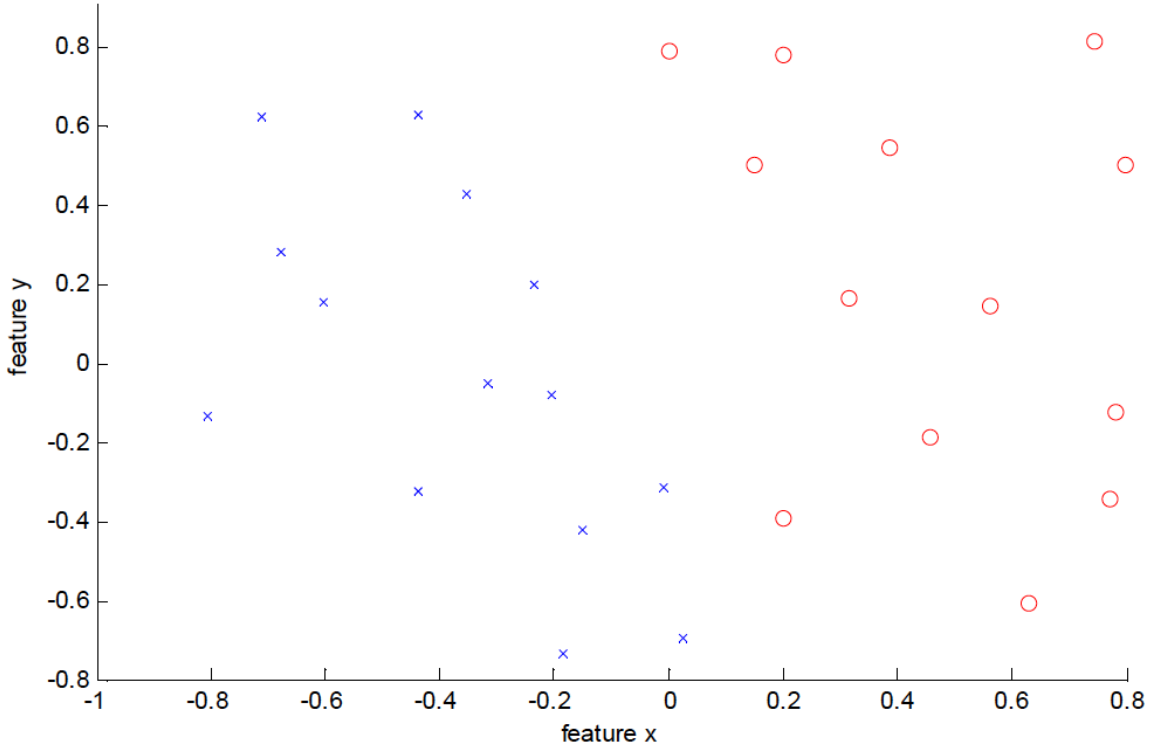
- C=Infinity -> hard margin



- Data is linearly separable
- But with a narrow margin

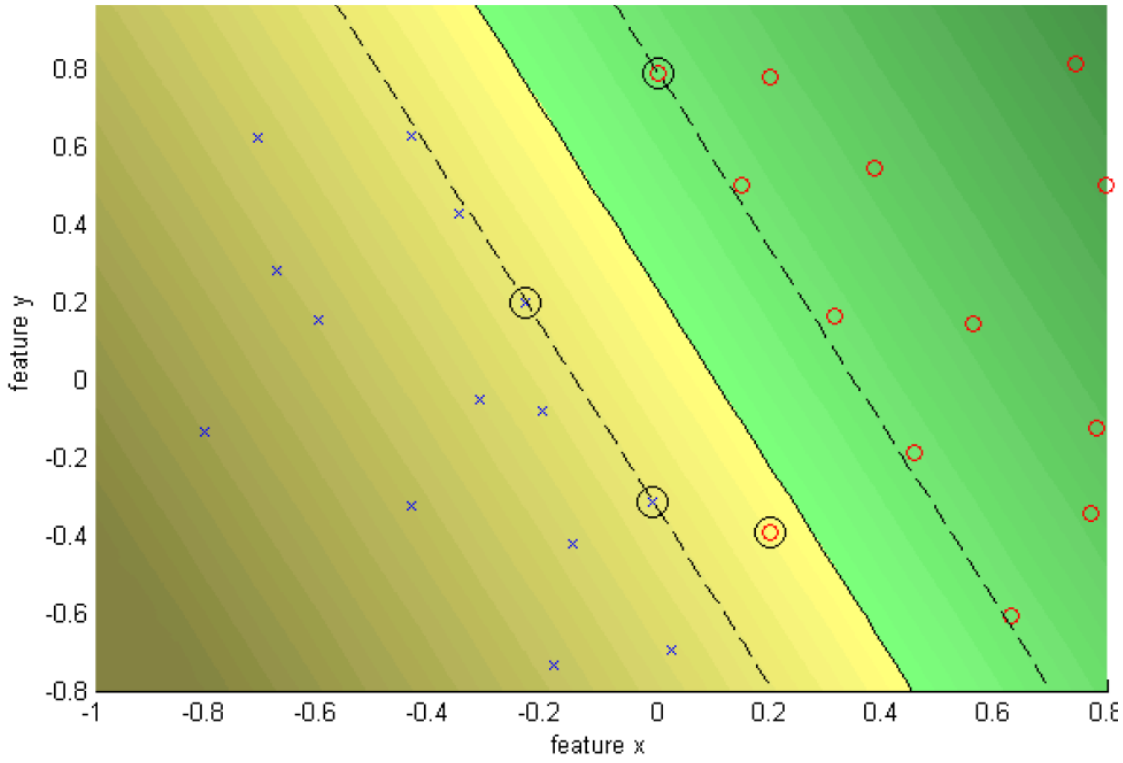


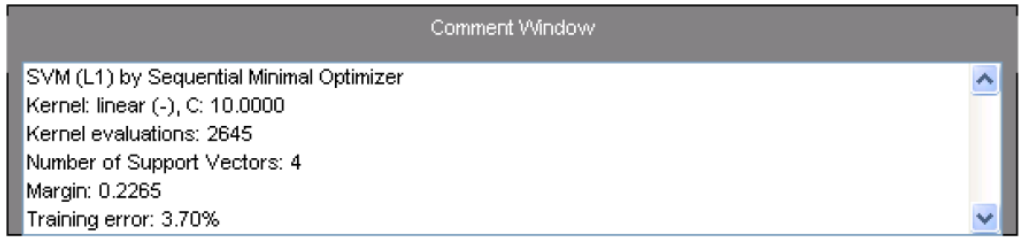
Example of Soft Margin



- Data is linearly separable
- But with a narrow margin

- C=10 -> soft margin





Loss Function

- Learning an SVM has been formulated as a constrained optimization over w and ξ

$$\min_{\mathbf{w}\in\mathbb{R}^d,\xi_i\in\mathbb{R}+} ||\mathbf{w}||^2 + C\sum_i^N \xi_i \text{ subject to } y_i\left(\mathbf{w}^\top \mathbf{x}_i + b\right) \geq 1 - \xi_i \text{ for } i = 1\ldots N$$

- The constraint can be written more concisely as $y_i f(\mathbf{x}_i) \geq 1 - \xi_i$

  which, together with $\xi_i \geq 0$, is equivalent to

$$\xi_i = \max(0, 1 - y_i f(\mathbf{x}_i))$$

Hence the learning problem is equivalent to the unconstrained optimization problem over w

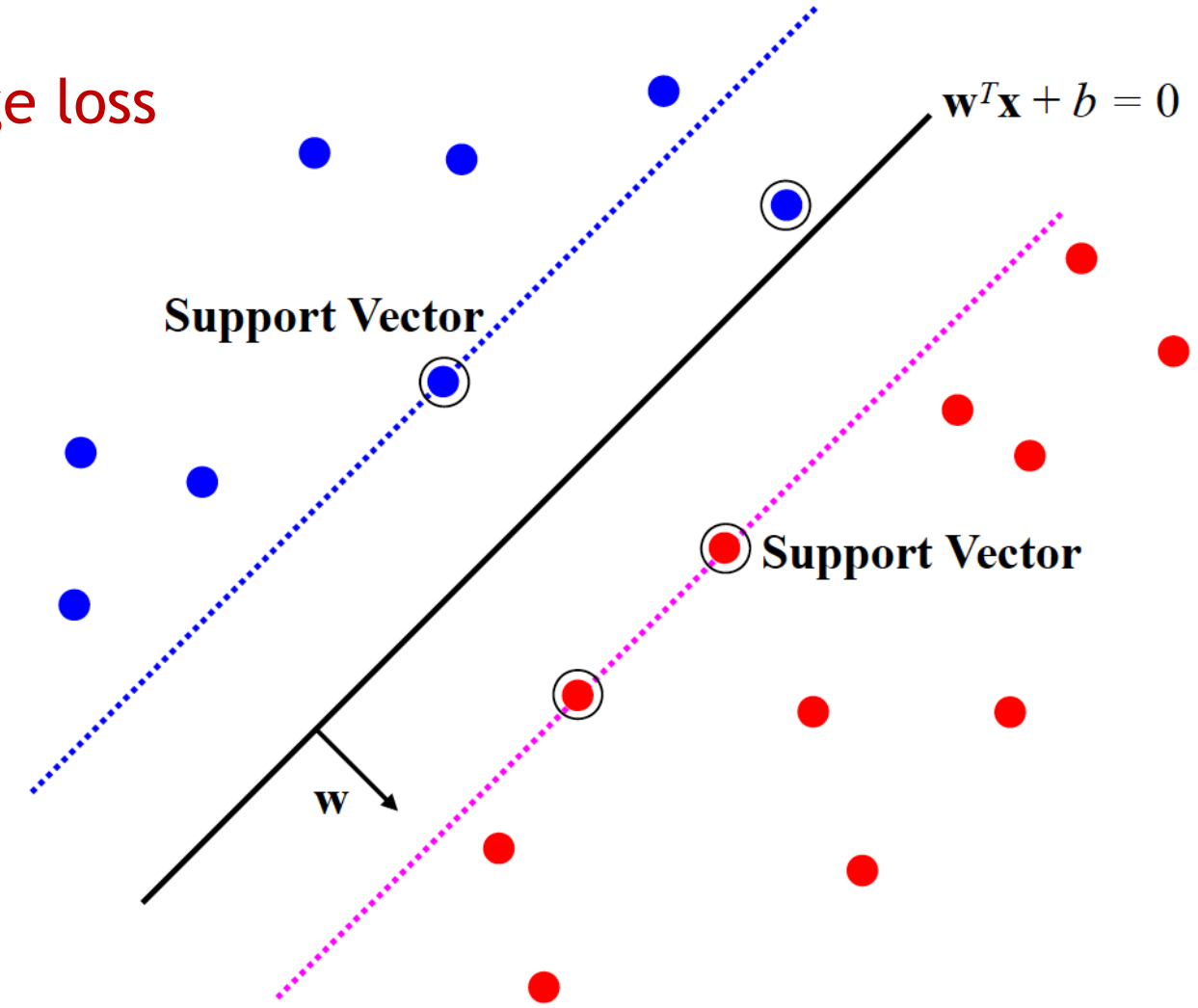
$$\min_{\mathbf{w}\in\mathbb{R}^d} \underbrace{||\mathbf{w}||^2}_{\text{regularization}} + C\sum_i^N \underbrace{\max(0, 1 - y_i f(\mathbf{x}_i))}_{\text{loss function}}$$

Convex function -> local minimum is global minimum

Can be solved by gradient descent algorithm

Loss Function

$$\min_{\mathbf{w}\in\mathbb{R}^d} ||\mathbf{w}||^2 + C\sum_i^N \underbrace{\max(0, 1 - y_i f(\mathbf{x}_i))}_{\text{loss function}}$$ Hinge loss



Points are in three categories:

- $y_i f(x_i) > 1$
  - Point is outside margin
  - No contribution to loss
- $y_i f(x_i) = 1$
  - Point is on margin
  - No contribution to loss
  - As in hard margin case
- $y_i f(x_i) < 1$
  - Point violates margin constraint
  - Contributes to loss

Loss Function

The 0-1 loss function

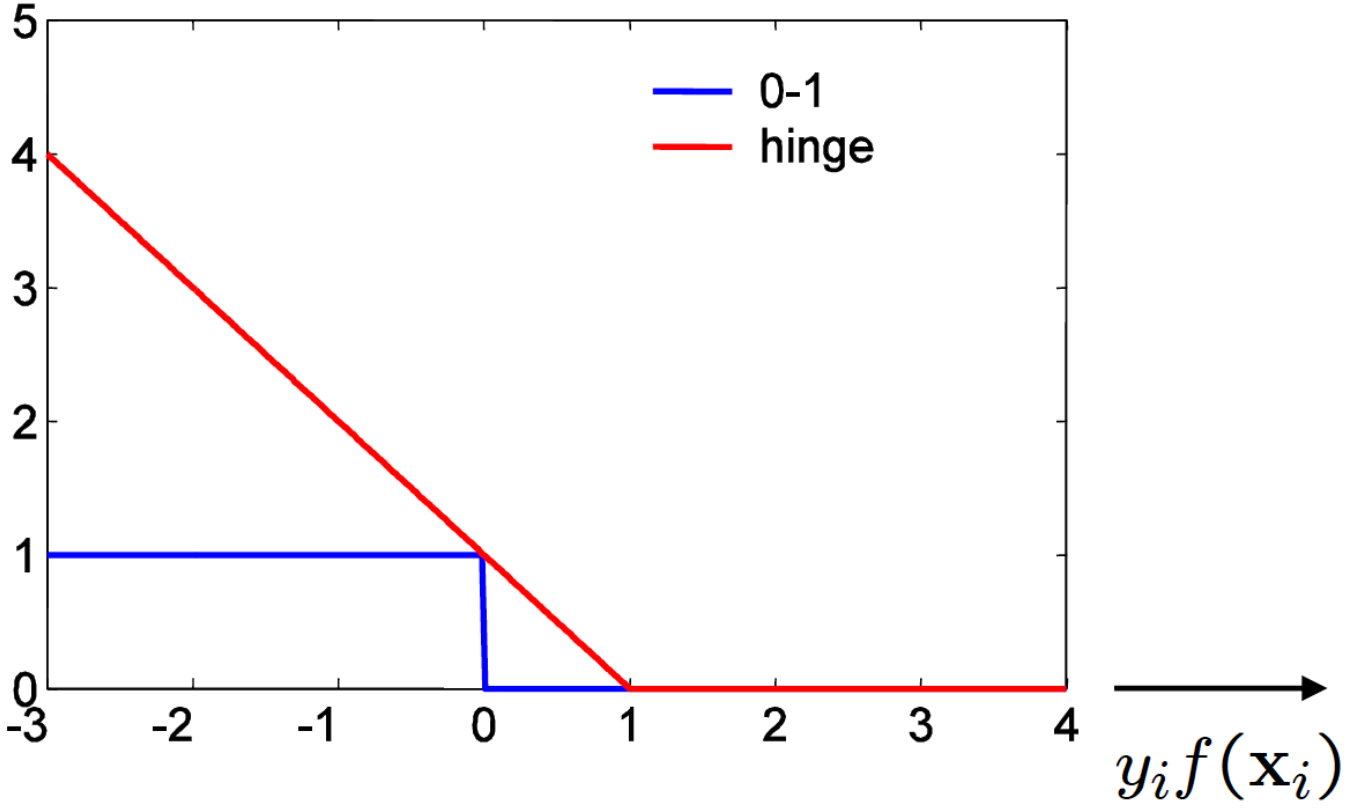
- Identify misclassified data points

$$L(y_i, \hat{y}_i) = \begin{cases} 1 & y_i \neq \hat{y}_i \\ 0 & y_i = \hat{y}_i \end{cases}$$

- SVM uses hinge loss
  - An approximation to the 0-1 loss

$$\max(0, 1 - y_i f(\mathbf{x}_i))$$

  - Minimizing hinge loss approximates minimizing the number of misclassified data points



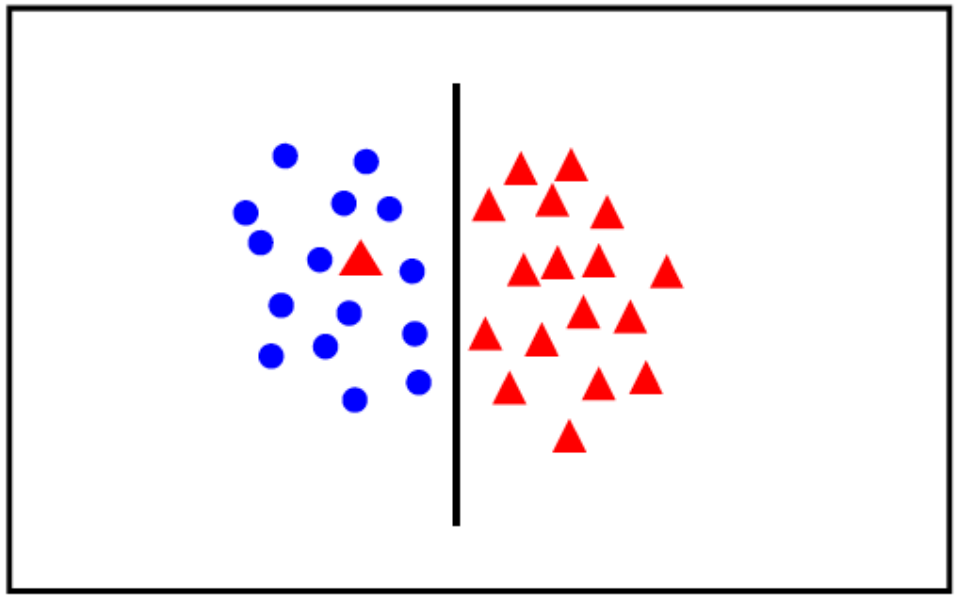
COSC 3337
Data Science I
Section 14623

Classification (contd.)

Instructor: Jingchao Ni
Fall 2024

Handling Non-Linearly Separable Data

- Introduce Slack Variables



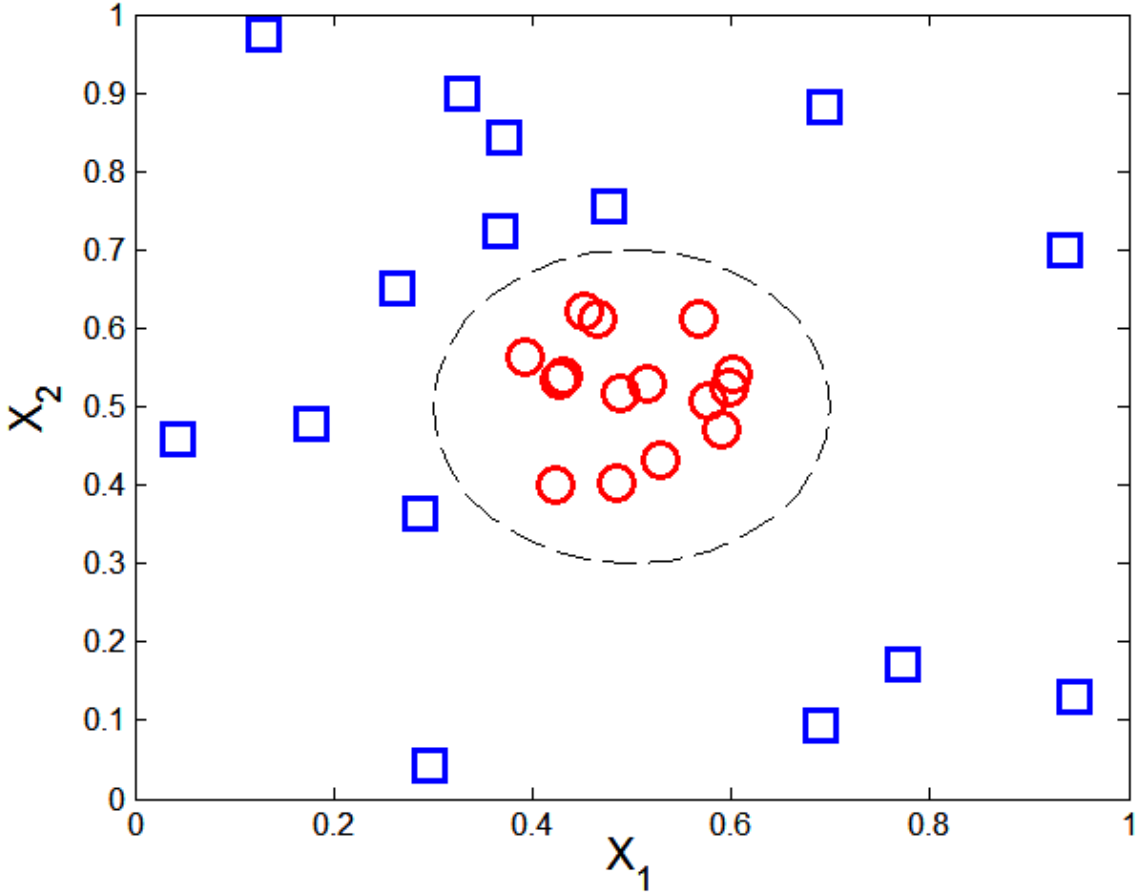
$$\min_{\mathbf{w}\in\mathbb{R}^d,\xi_i\in\mathbb{R}^+} ||\mathbf{w}||^2 + C\sum_i^N \xi_i$$

Subject to

$$y_i\left(\mathbf{w}^\top \mathbf{x}_i + b\right) \geq 1 - \xi_i \text{ for } i = 1\ldots N$$

Nonlinear Support Vector Machine

- What if decision boundary is not linear?



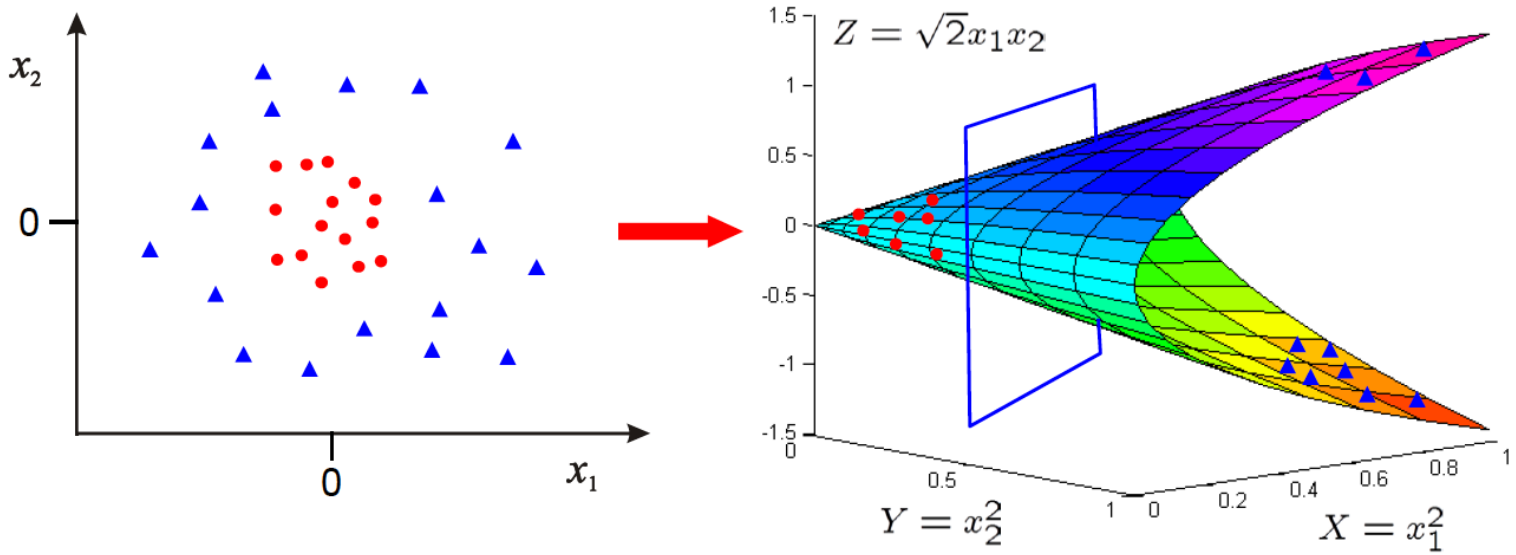
$$y(x_1, x_2) = \begin{cases} 1 & \text{if } \sqrt{(x_1 - 0.5)^2 + (x_2 - 0.5)^2} > 0.2 \\ -1 & \text{otherwise} \end{cases}$$

Linear classifier is not appropriate

Nonlinear Support Vector Machine

- Trick: map data to high dimensional space

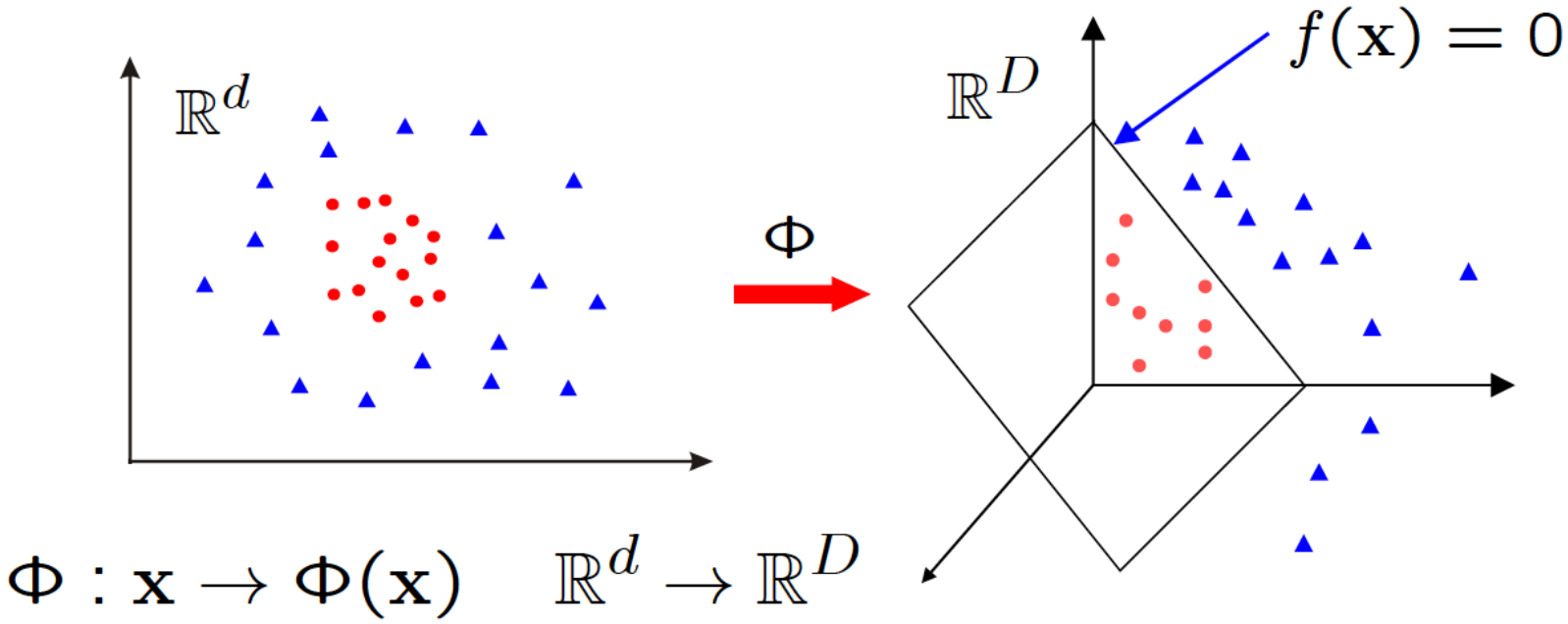
$$\Phi : \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \rightarrow \begin{pmatrix} x_1^2 \\ x_2^2 \\ \sqrt{2}x_1x_2 \end{pmatrix} \qquad \mathbb{R}^2 \rightarrow \mathbb{R}^3$$



- Data is linearly separable in 3D space
- This means the problem can be solved by a linear classifier in a high dimensional space

Nonlinear Support Vector Machine

- SVM classifiers in a transformed feature space



Learn a classifier that is linear in w for $\mathbb{R}^D$ $f(\mathbf{x}) = \mathbf{w}^\top \Phi(\mathbf{x}) + b$

Feature map

Nonlinear Support Vector Machine

- Classifier, with $\mathbf{w} \in \mathbb{R}^D$

$$f(\mathbf{x}) = \mathbf{w}^\top \Phi(\mathbf{x}) + b$$

- Learning, for $\mathbf{w} \in \mathbb{R}^D$

$$\min_{\mathbf{w} \in \mathbb{R}^D} ||\mathbf{w}||^2 + C \sum_i^N \max(0, 1 - y_i f(\mathbf{x}_i))$$

  - Simply map x to $\phi(x)$ where data is separable
  - Solve for w in high dimensional space $\mathbb{R}^D$
  - What type of mapping ϕ should be used?
  - If $D \gg d$, then there are many more parameters to learn for w, how to avoid expensive computation?

Dual Form of SVM Classifier

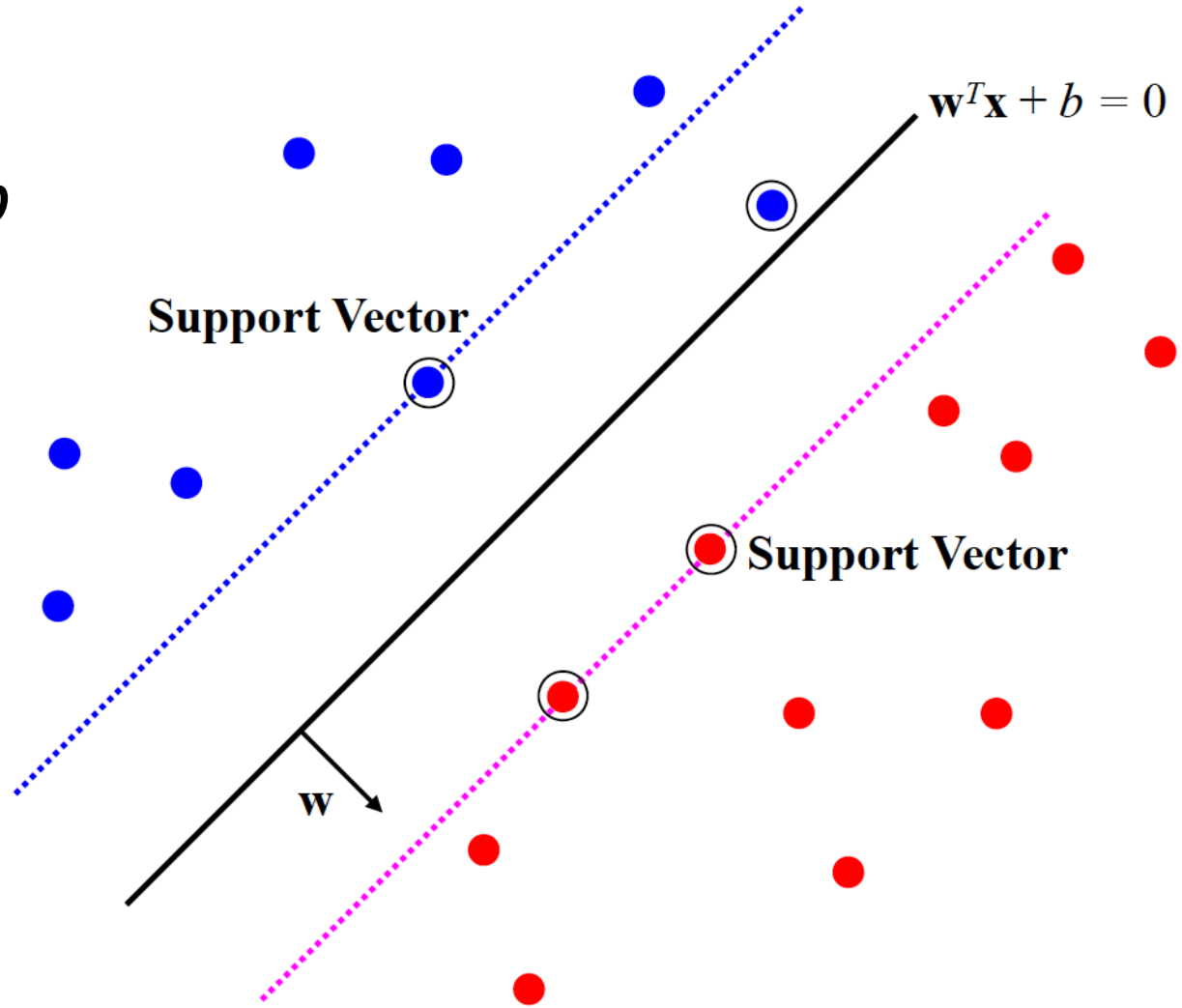
- Primal problem $\min_{\mathbf{w}\in\mathbb{R}^d} ||\mathbf{w}||^2 + C\sum_i^N \max(0, 1 - y_i f(\mathbf{x}_i))$
- Primal version of classifier $f(\mathbf{x}) = \mathbf{w}^\top \mathbf{x} + b$
- Dual problem $\max_{\alpha_i \geq 0} \sum_i \alpha_i - \frac{1}{2}\sum_{jk} \alpha_j \alpha_k y_j y_k (\mathbf{x}_j{}^\top \mathbf{x}_k)$ subject to $0 \leq \alpha_i \leq C$ for $\forall i$, and $\sum_i \alpha_i y_i = 0$
- Dual version of classifier $f(\mathbf{x}) = \sum_i^N \alpha_i y_i (\mathbf{x}_i{}^\top \mathbf{x}) + b$

- d parameters for primal, N for dual
- if N<<d, more efficient to solve α than w
- The dual form only involves $(x_j^T x_k)$, an advantage for using ϕ

Dual Form of SVM Classifier

$$f(x) = \sum_i \alpha_i y_i \left(x_i^T x\right) + b$$

Support Vectors



Many of the α_i's are zero, the ones that are non-zero define the support vectors

Dual Form in Transformed Feature Space

- Dual classifier

$$f(\mathbf{x}) = \sum_i^N \alpha_i y_i \, \mathbf{x}_i{}^\top \mathbf{x} + b \qquad \text{Original space}$$

$$\rightarrow f(\mathbf{x}) = \sum_i^N \alpha_i y_i \, \Phi(\mathbf{x}_i)^\top \Phi(\mathbf{x}) + b \qquad \text{Transformed space}$$

- Dual problem

$$\max_{\alpha_i \geq 0} \sum_i \alpha_i - \frac{1}{2} \sum_{jk} \alpha_j \alpha_k y_j y_k \mathbf{x}_j{}^\top \mathbf{x}_k \qquad \text{Original space}$$

$$\rightarrow \max_{\alpha_i \geq 0} \sum_i \alpha_i - \frac{1}{2} \sum_{jk} \alpha_j \alpha_k y_j y_k \Phi(\mathbf{x}_j)^\top \Phi(\mathbf{x}_k) \qquad \text{Transformed space}$$

- Subject to

$$0 \leq \alpha_i \leq C \text{ for } \forall i, \text{ and } \sum_i \alpha_i y_i = 0$$

Dual Form in Transformed Feature Space

- Note that $\phi(x)$ only occurs in pairs $\phi(x_j)^T \phi(x_i)$
- Once the scalar products are computed, only the N dimensional vector α needs to be learnt
- It is not necessary to learn in the D dimensional transformed space
- Write $k(x_i, x_j) = \phi(x_j)^T \phi(x_i)$. This is known as Kernel
- Dual classifier

$$f(\mathbf{x}) = \sum_i^N \alpha_i y_i \, k(\mathbf{x}_i, \mathbf{x}) + b$$

- Dual problem

$$\max_{\alpha_i \geq 0} \sum_i \alpha_i - \frac{1}{2} \sum_{jk} \alpha_j \alpha_k y_j y_k \, k(\mathbf{x}_j, \mathbf{x}_k)$$

- Subject to

$$0 \leq \alpha_i \leq C \text{ for } \forall i, \text{ and } \sum_i \alpha_i y_i = 0$$

Kernel Trick

- Example $\Phi : \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \rightarrow \begin{pmatrix} x_1^2 \\ x_2^2 \\ \sqrt{2}x_1x_2 \end{pmatrix} \quad \mathbb{R}^2 \rightarrow \mathbb{R}^3$

$$\begin{aligned} \Phi(\mathbf{x})^\top \Phi(\mathbf{z}) &= \left(x_1^2, x_2^2, \sqrt{2}x_1x_2\right) \begin{pmatrix} z_1^2 \\ z_2^2 \\ \sqrt{2}z_1z_2 \end{pmatrix} \\ &= x_1^2z_1^2 + x_2^2z_2^2 + 2x_1x_2z_1z_2 \\ &= (x_1z_1 + x_2z_2)^2 \\ &= (\mathbf{x}^\top \mathbf{z})^2 \end{aligned}$$

- Classifier can be learnt and applied without explicitly computing $\phi(x)$
- All that is required is the kernel $k(x, z) = \phi(x)^T \phi(z)$
- Complexity of learning depends on N (typically it is $O(N^3)$), not on D

Example Kernels

- Linear kernel: $k(x, z) = x^T z$ (This is equivalent to a linear SVM)
- Polynomial kernel $k(x, z) = (1 + x^T z)^d$ for any d>0
  - Contains all polynomials terms up to degree d
- Gaussian kernel $k(x, z) = \exp(-\|x - z\|^2 / 2\sigma^2)$ for $\sigma > 0$
  - Also called Radial Basis Function (RBF) kernel
  - $k(x, z) = \exp(-\gamma\|x - z\|^2)$ for $\gamma > 0$
- Sigmoid kernel $k(x, z) = \tanh(\gamma x^T z + 1)$
- The kernel computation is performed in the original feature space

SVM Classifier with Gaussian Kernel

Size of training dataset

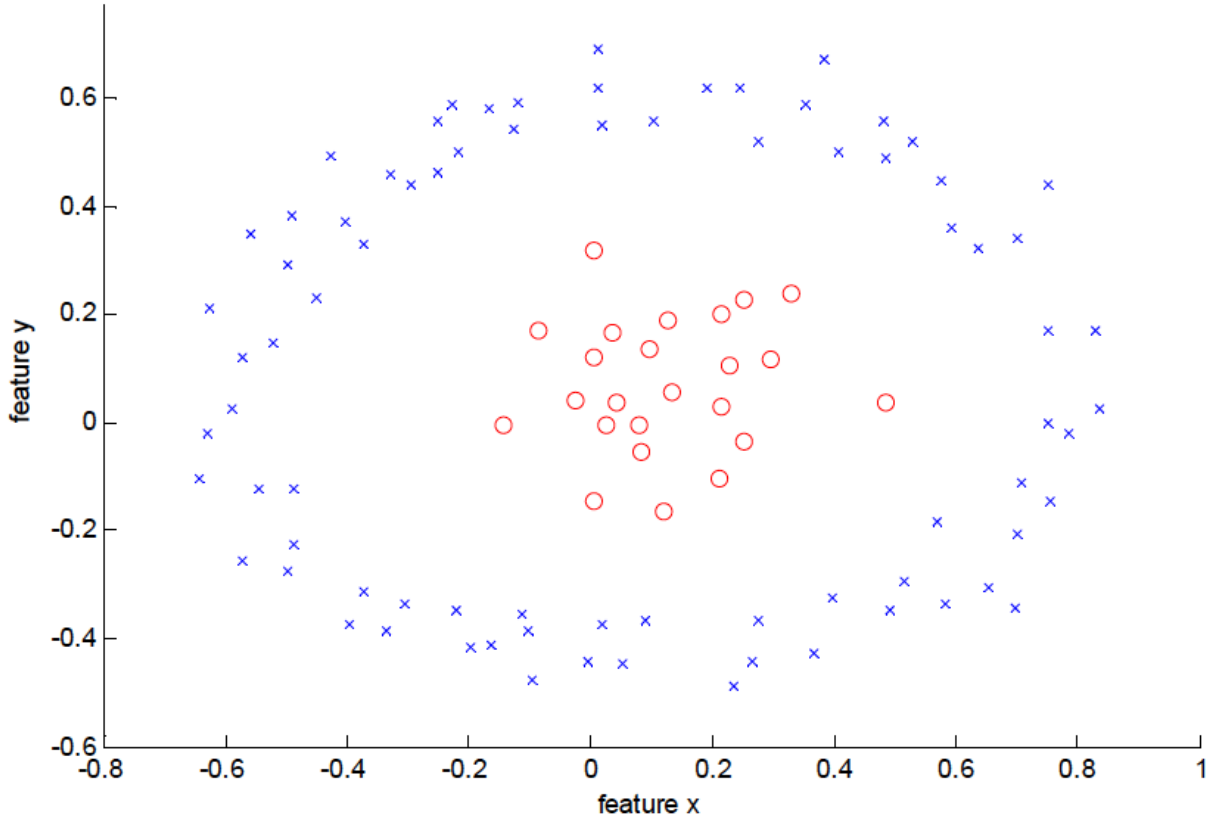
$$f(\mathbf{x}) = \sum_{i=1}^{N} \alpha_i y_i k(\mathbf{x}_i, \mathbf{x}) + b$$

Weight: non-zero α_i's define the support vectors

Support vector

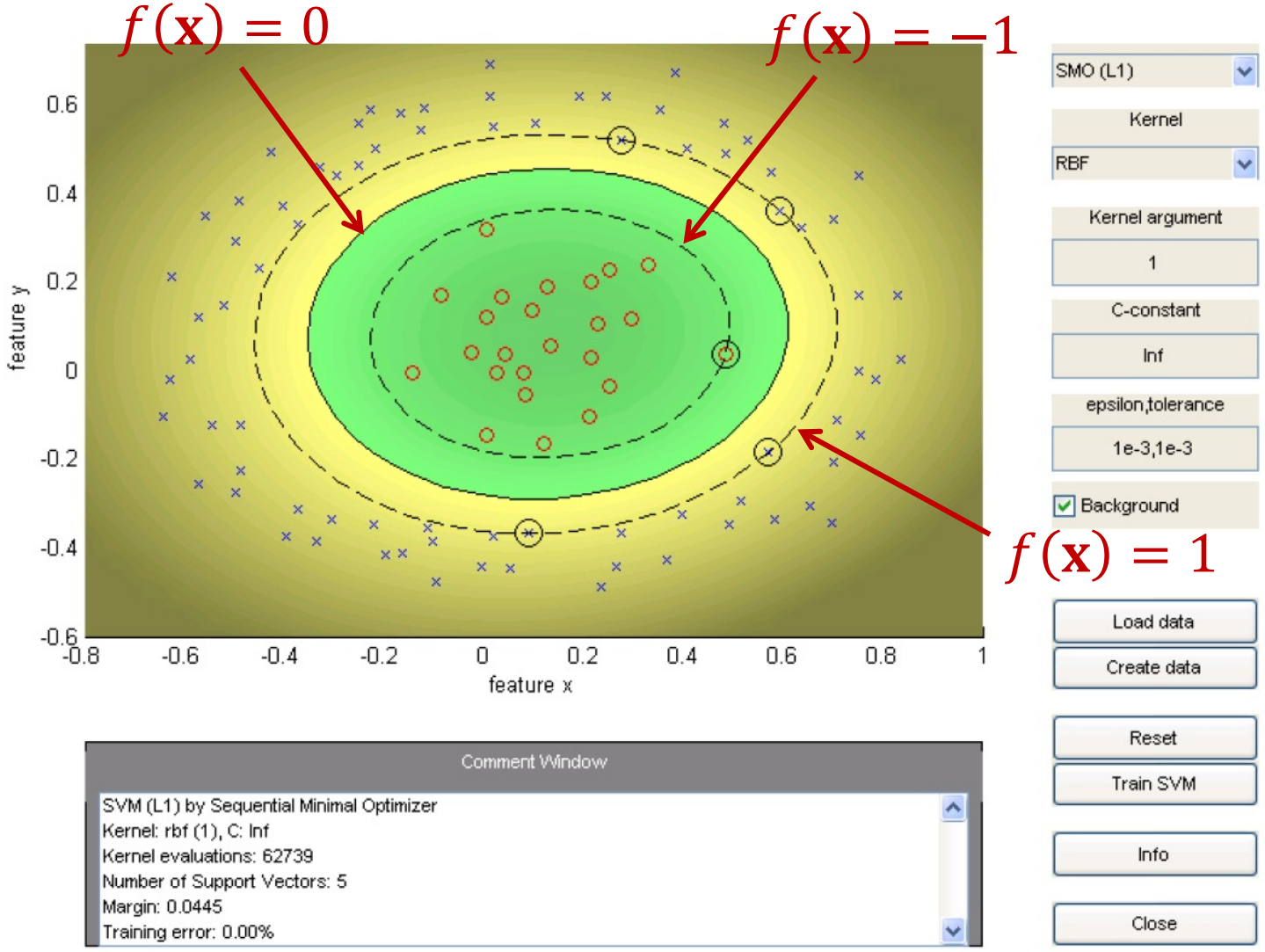
- Gaussian kernel $k(x, z) = \exp(-\|x - z\|^2 / 2\sigma^2)$ for $\sigma > 0$
  - Also called Radial Basis Function (RBF) kernel
- RBF kernel SVM $$f(\mathbf{x}) = \sum_{i=1}^{N} \alpha_i y_i \exp(-\|\mathbf{x} - \mathbf{x}_i\|^2 / 2\sigma^2) + b$$

RBF Kernel SVM Example



- The data is not linearly separable in the original 2D space

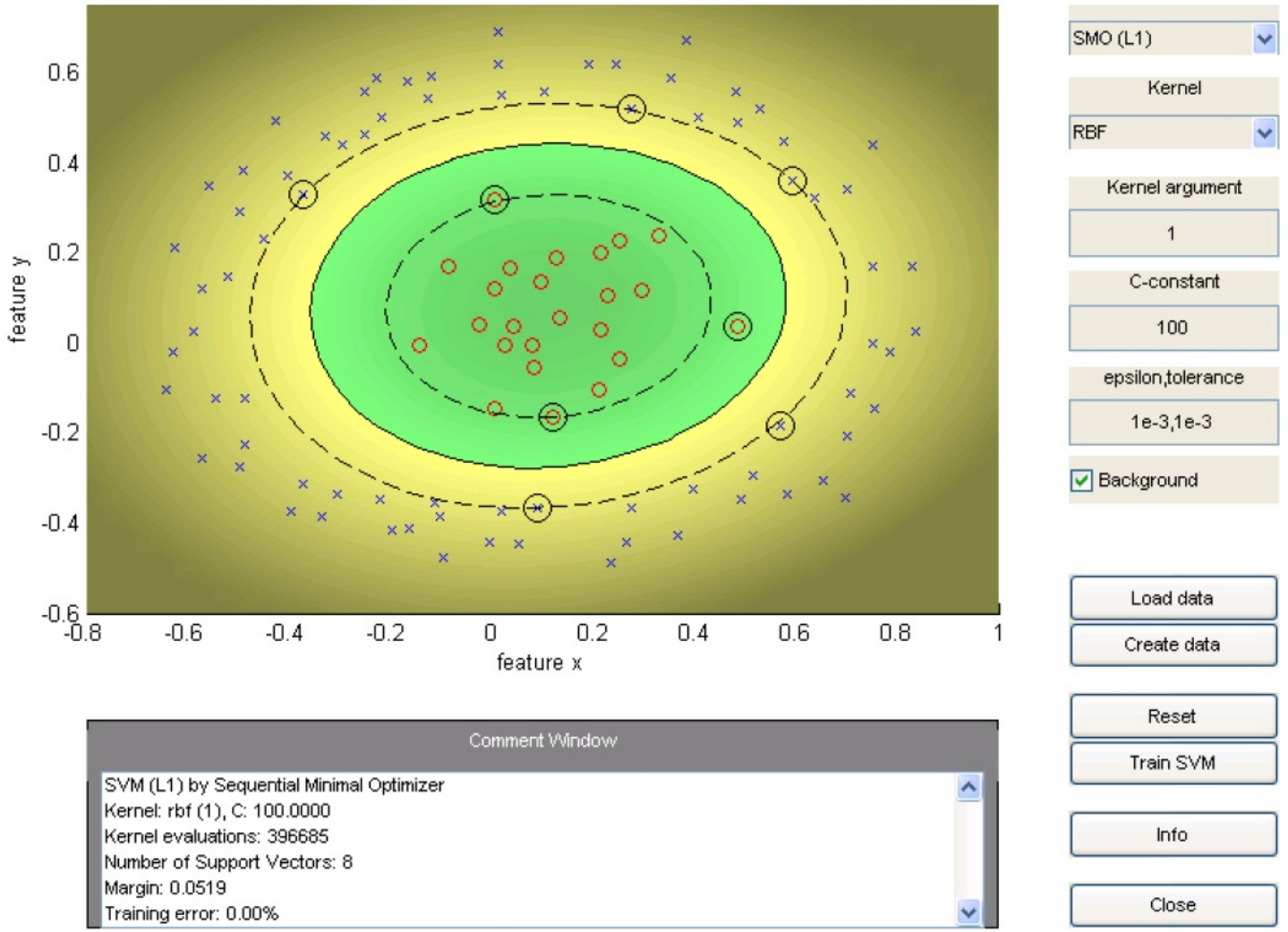
RBF Kernel SVM Example



$$\sigma = 1.0, C = \infty$$

$$f(\mathbf{x}) = \sum_{i=1}^{N} \alpha_i y_i \exp(-\|\mathbf{x} - \mathbf{x}_i\|^2 / 2\sigma^2) + b$$

RBF Kernel SVM Example

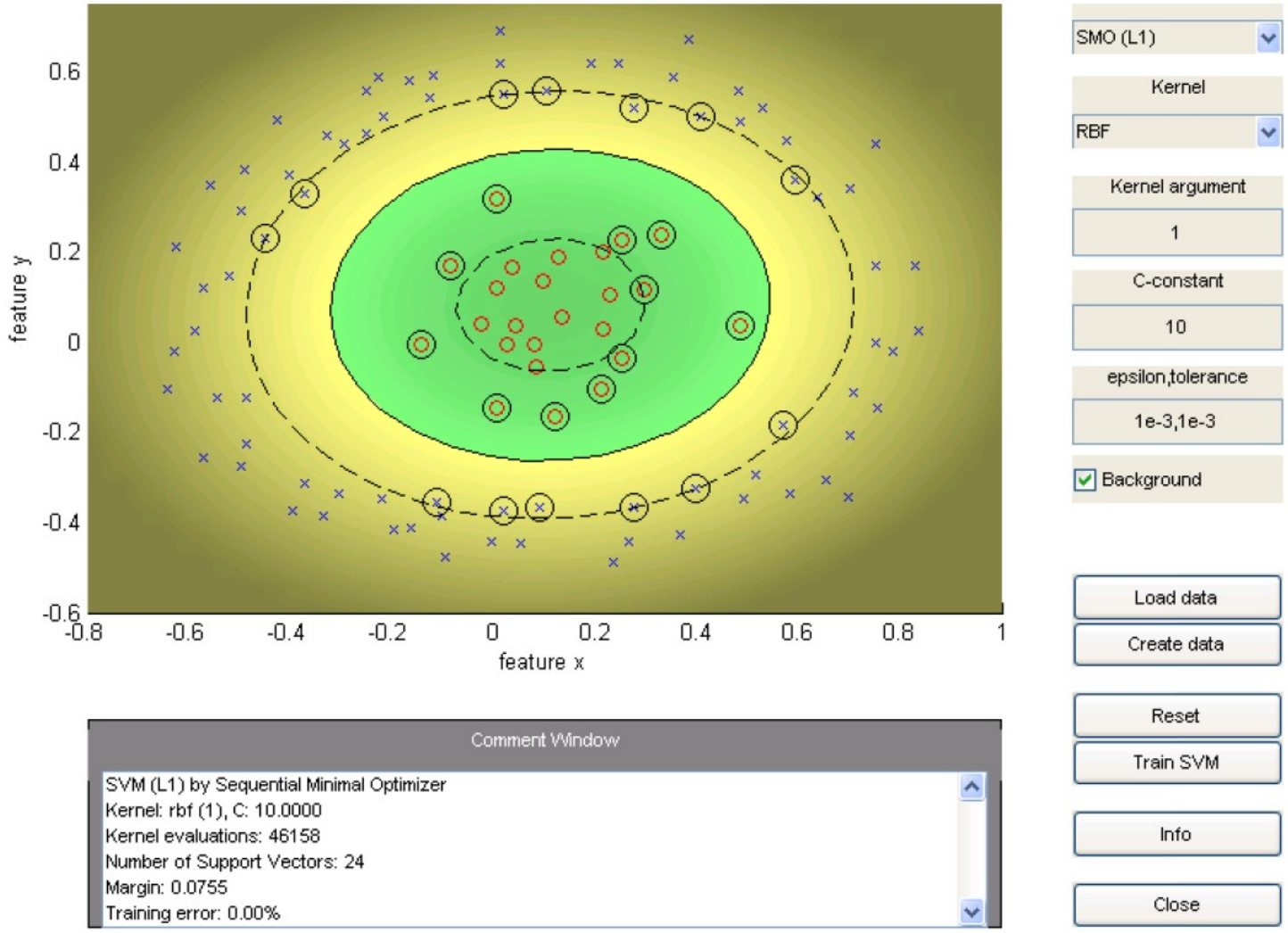


$$\sigma = 1.0, C = 100$$

$$f(\mathbf{x}) = \sum_{i=1}^{N} \alpha_i y_i \exp(-\|\mathbf{x} - \mathbf{x}_i\|^2 / 2\sigma^2) + b$$

Decrease C gives wider (soft) margin

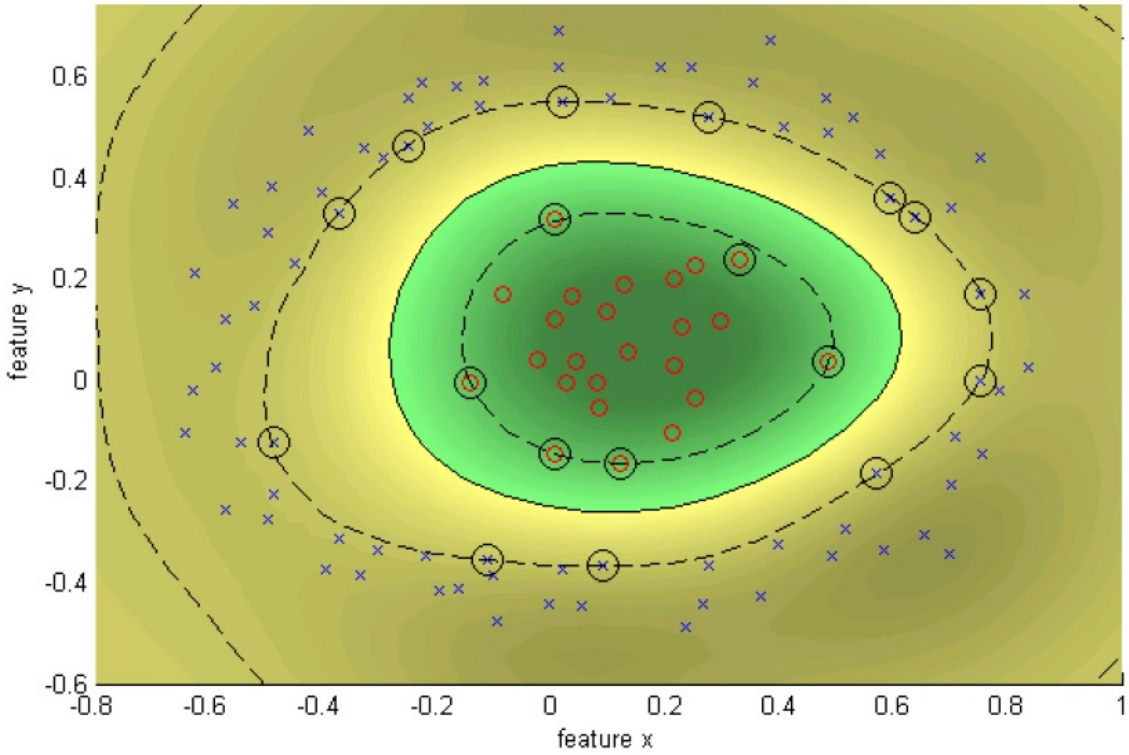
RBF Kernel SVM Example

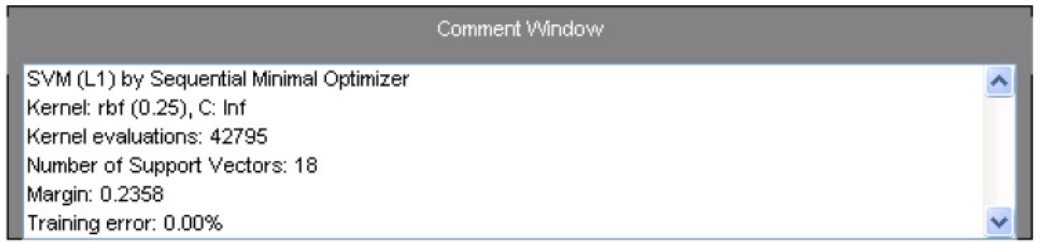


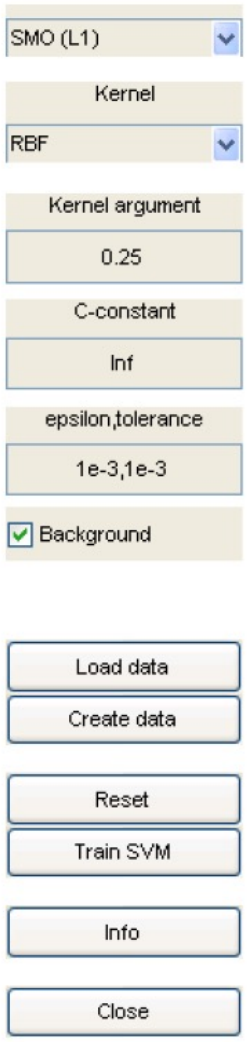
$$\sigma = 1.0, C = 10$$

$$f(\mathbf{x}) = \sum_{i=1}^{N} \alpha_i y_i \exp(-\|\mathbf{x} - \mathbf{x}_i\|^2 / 2\sigma^2) + b$$

RBF Kernel SVM Example





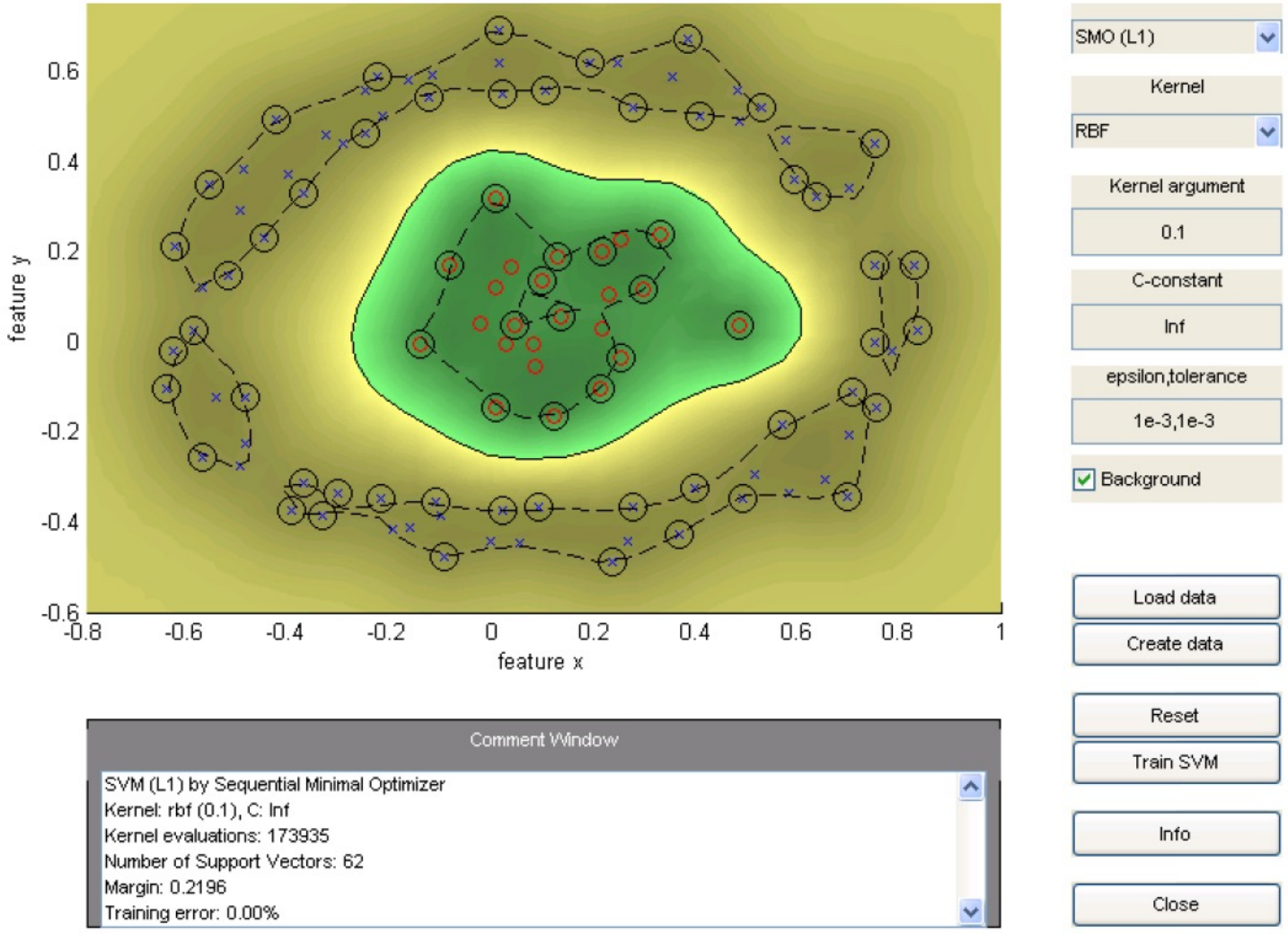


$$\sigma = 0.25, C = \infty$$

$$f(\mathbf{x}) = \sum_{i=1}^{N} \alpha_i y_i \exp(-\|\mathbf{x} - \mathbf{x}_i\|^2 / 2\sigma^2) + b$$

Decrease σ moves towards nearest neighbor classifier

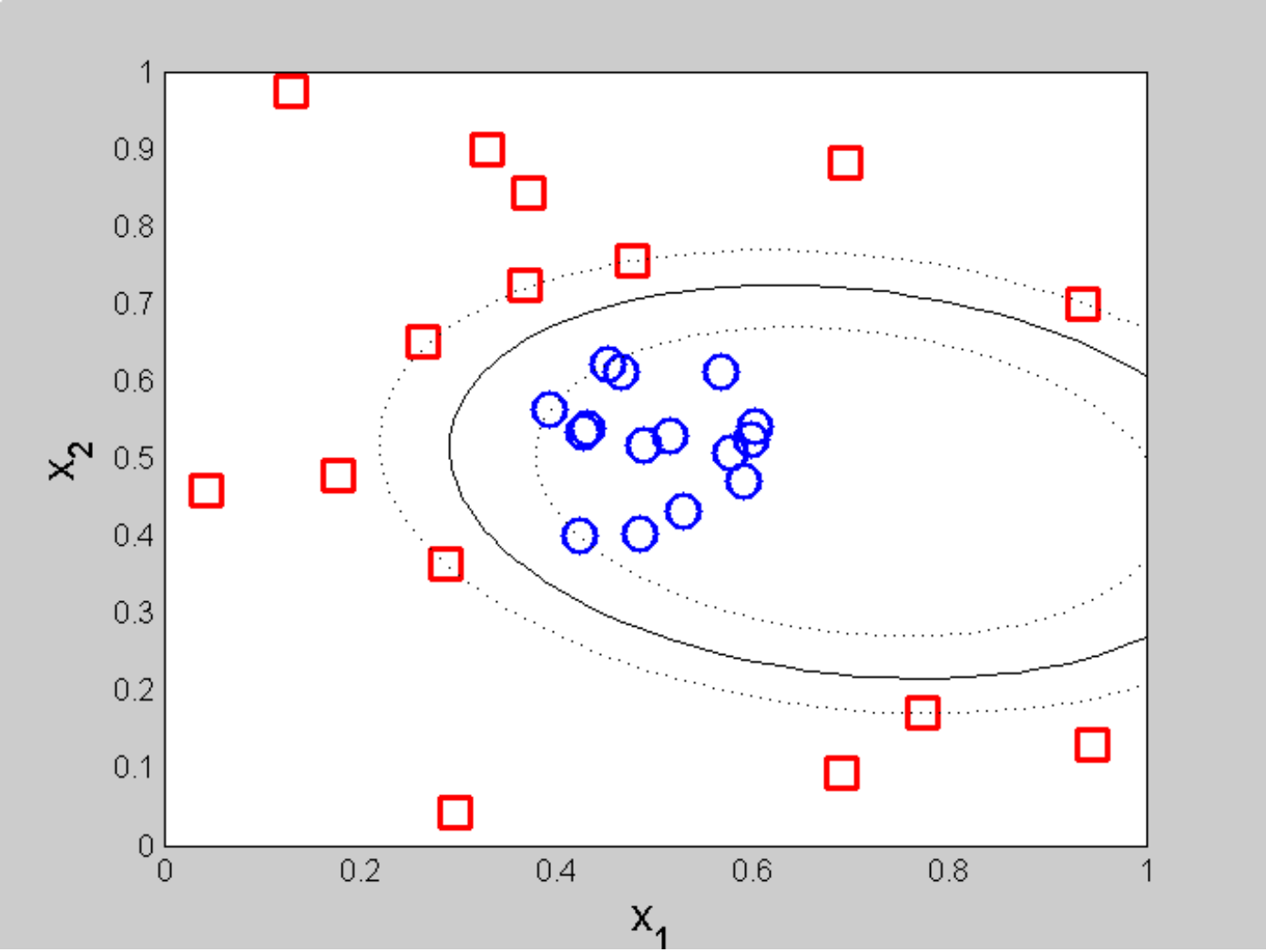
RBF Kernel SVM Example



$$\sigma = 0.1, C = \infty$$

$$f(\mathbf{x}) = \sum_{i=1}^{N} \alpha_i y_i \exp(-\|\mathbf{x} - \mathbf{x}_i\|^2 / 2\sigma^2) + b$$

Polynomial Kernel SVM Example



$$d = 2$$

$$f(\mathbf{x}) = \sum_{i=1}^{N} \alpha_i y_i (1 + \mathbf{x}^T \mathbf{x}_i)^d + b$$

Kernel Trick Summary

- Classifiers can be learnt for high dimensional feature spaces, without having to map the points to the high dimensional space
  - Don't have to know the mapping function ϕ
  - Kernel computation $k(\mathbf{x}, \mathbf{z})$ is performed in the original (low dimensional) space
- Data may be linearly sparable in the high dimensional space, but not linearly separable in the original feature space
- Kernels can be used for an SVM because of the scalar product in the dual form. They can also be used elsewhere. They are not tied to SVM
- Not all functions ϕ can be kernels. Finding $k(\mathbf{x}, \mathbf{z})$ is challenging. $k(\mathbf{x}, \mathbf{z})$ might not exist for a particular function ϕ

SVM Classifier in Sklearn

```
class sklearn.svm.SVC(*, C=1.0, kernel='rbf', degree=3, gamma='scale', coef0=0.0, shrinking=True, probability=False, tol=0.001, cache_size=200, class_weight=None, verbose=False, max_iter=-1, decision_function_shape='ovr', break_ties=False, random_state=None)
```

```
>>> from sklearn.svm import SVC
>>> from sklearn.model_selection import train_test_split
>>> clf = SVC(kernel='rbf')
>>> X_tr, X_te, y_tr, y_te = train_test_split(X, y,
train_size=0.9)
>>> clf.fit(X_tr, y_tr)
>>> y_te = clf.predict(X_te)
>>> prob = clf.predict_proba(X_te)
```

- Kernel functions

https://scikit-learn.org/stable/modules/svm.html#svm-kernels

- linear: $\langle x, x' \rangle$.
- polynomial: $(\gamma \langle x, x' \rangle + r)^d$, where d is specified by parameter `degree`, r by `coef0`.
- rbf: $\exp(-\gamma \|x - x'\|^2)$, where γ is specified by parameter `gamma`, must be greater than 0.
- sigmoid $\tanh(\gamma \langle x, x' \rangle + r)$, where r is specified by `coef0`.

C: float, default=1.0
Regularization parameter.

kernel: {'linear', 'poly', 'rbf', 'sigmoid', 'precomputed'} or callable, default='rbf'
Specifies the kernel type to be used in the algorithm.

degree: int, default=3
Degree of the polynomial kernel function ('poly').

gamma: {'scale', 'auto'} or float, default='scale'
Kernel coefficient for 'rbf', 'poly' and 'sigmoid'.
'scale' uses 1 / (n_features * X.var()).

coef0: float, default=0.0
Independent term in kernel function.

probability: bool, default=False
Whether to enable probability estimates. it internally uses 5-fold cross-validation.

SVM Classifier in Sklearn

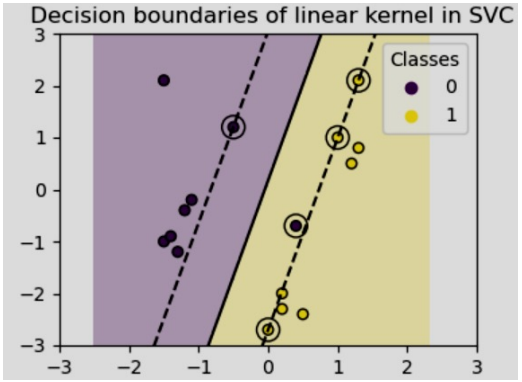
Plot classification boundaries with different SVM kernels

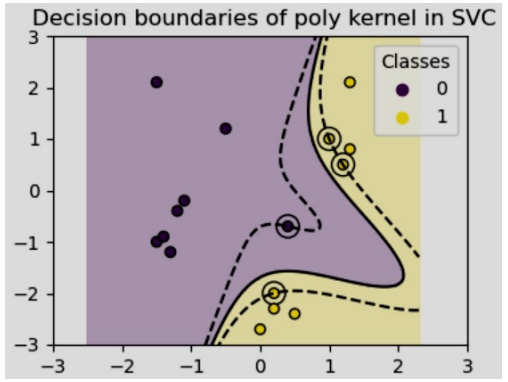
```
from sklearn import svm
from sklearn.inspection import DecisionBoundaryDisplay

def plot_training_data_with_decision_boundary( kernel, ax=None,
long_title=True, support_vectors=True ):
     # Train the SVC
     clf = svm.SVC(kernel=kernel, gamma=2).fit(X, y)
     ...
```

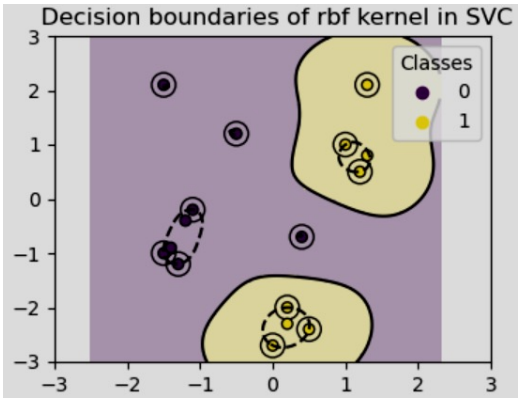
https://scikit-learn.org/dev/auto_examples/svm/plot_svm_kernels.html

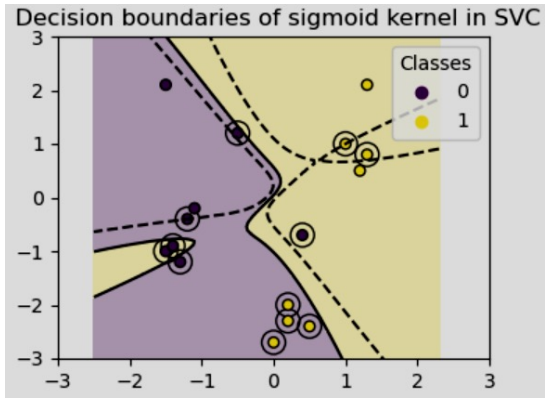
```
>>> plot_training_data_with_decision_boundary("linear")

>>> plot_training_data_with_decision_boundary("poly")

>>> plot_training_data_with_decision_boundary("rbf")

>>> plot_training_data_with_decision_boundary("sigmoid")
```

- Visualization of SVM decision boundaries









SVM Summary

- Support vector machines learn hyperplanes that separate two classes maximizing the margin between them (the empty space between the instances of the two classes)
- Support vector machines introduce slack variables, in the case that classes are not linear separable and trying to maximize margins while keeping the training error low
- The most popular versions of SVMs use non-linear kernel functions to map the feature space into a higher dimensional space to facilitate finding “good” linear decision boundaries in the modified space
- Support vector machines find “margin optimal” hyperplanes by solving a convex quadratic optimization problem. The complexity of this optimization problem is high; however, as computer got faster using SVMs on large datasets is no longer a major challenge
- In general, support vector machines accomplish quite high accuracies, if compared to other techniques

Bayes Classifiers

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

$$P(Y \mid X) = \frac{P(X,Y)}{P(X)}$$

$$P(X \mid Y) = \frac{P(X,Y)}{P(Y)}$$

- Bayes theorem:

$$P(Y \mid X) = \frac{P(X \mid Y)P(Y)}{P(X)}$$

Bayes Theorem in Classification

$$P(Y \mid X) = \frac{P(X \mid Y)P(Y)}{P(X)} \qquad \text{Posterior} = \frac{\text{Likelihood} \times \text{Prior}}{\text{Evidence}}$$

- X: evidence event
- Y: class event
- P(Y): prior, the probability of class Y before evidence is seen
- P(X): marginal probability, probability of observing evidence X
- P(X | Y): likelihood, the probability of observing X under the assumption of class Y
- P(Y | X): posterior, the probability of class Y after evidence is seen

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?

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

Using Bayes Theorem for Classification

- Consider each attribute and class label as random variables

- Given a record with attributes (X_1, X_2,…, X_d)
  - Goal is to predict class Y
  - Specifically, we want to find the value of Y that maximizes $P(Y| X_1, X_2,…, X_d)$

- Can we estimate $P(Y| X_1, X_2,…, X_d)$ directly from data?

Example Data

- Given a Test Record $X = (\text{Refund} = \text{No}, \text{Divorced}, \text{Income} = 120\text{K})$

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

- Can we estimate

  P(Evade = Yes | X) and P(Evade = No | X)?

- In the following we will replace

  Evade = Yes by Yes, and

  Evade = No by No

  -> P(Evade = Yes | X) = P(Yes | X)

  -> P(Evade = No | X) = P(No | X)

Using Bayes Theorem for Classification

- Approach:
  - Compute posterior probability P(Y | X_1, X_2, …, X_d) using the Bayes theorem

$$P(Y \mid X_1 X_2 \ldots X_n) = \frac{P(X_1 X_2 \ldots X_d \mid Y) P(Y)}{P(X_1 X_2 \ldots X_d)}$$

  - *Maximum a-posteriori*: Choose Y that maximizes
    P(Y | X_1, X_2, …, X_d)
  - Equivalent to choosing value of Y that maximizes
    P(X_1, X_2, …, X_d|Y) P(Y)

- How to estimate P(X_1, X_2, …, X_d | Y)?

COSC 3337
Data Science I
Section 14623

Classification (contd.)

Instructor: Jingchao Ni
Fall 2024

Example Data

- Given a Test Record $X = (\text{Refund} = \text{No}, \text{Divorced}, \text{Income} = 120\text{K})$

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

- Use Bayes Theorem

$$P(\text{Yes}|X) = \frac{P(X|\text{Yes})P(\text{Yes})}{P(X)}$$

$$P(\text{No}|X) = \frac{P(X|\text{No})P(\text{No})}{P(X)}$$

How to estimate P(X|Yes) and P(X|No)?

Naïve Bayes Classifier

- Assume independence among attributes X_i when class is given:
  - $P(X_1, X_2, \ldots, X_d \mid Y_j) = P(X_1 \mid Y_j)\, P(X_2 \mid Y_j) \ldots P(X_d \mid Y_j)$
  - Now we can estimate $P(X_i \mid Y_j)$ for all X_i and Y_j combinations from the training data
  - $$P(Y \mid X_1 X_2 \ldots X_n) = \frac{P(X_1 X_2 \ldots X_d \mid Y) P(Y)}{P(X_1 X_2 \ldots X_d)}$$
  - New point is classified to Y_j if $P(Y_j) \prod_{i=1}^{d} P(X_i \mid Y_j)$ is maximal

Naïve Bayes Classifier on Example Data

- Given a Test Record $X = (\text{Refund} = \text{No}, \text{Divorced}, \text{Income} = 120\text{K})$

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

- P(X | Yes) =
  P(Refund = No | Yes) x
  P(Divorced | Yes) x
  P(Income = 120K | Yes)

- P(X | No) =
  P(Refund = No | No) x
  P(Divorced | No) x
  P(Income = 120K | No)

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: $P(Y) = N_k/N$
  - E.g., P(No) = 7/10,
  - E.g., P(Yes) = 3/10
- For categorical attributes:

  $P(X_i \mid Y_k) = |X_{ik}| / N_k$

  - where $|X_{ik}|$ is number of instances having attribute value X_i and belonging to class Y_k
  - Examples:

  P(Status=Married|No) = 4/7

  P(Refund=Yes|Yes)=0

Estimate Probabilities from Data

- For continuous attributes:
  - Discretization: Partition the range into bins
    - Replace continuous value with bin value
      - Attribute changed from continuous to ordinal
  - 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, use it to estimate the conditional probability $P(X_i|Y)$

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

Naïve 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) = 2/3
P(Marital Status = Divorced | Yes) = 1/3
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 | No) = P(Refund=No | No)
  × P(Divorced | No)
  × P(Income=120K | No)
  = 4/7 × 1/7 × 0.0072 = 0.0006

- P(X | Yes) = P(Refund=No | Yes)
  × P(Divorced | Yes)
  × P(Income=120K | Yes)
  = 1 × 1/3 × 1.2 × 10^{-9} = 4 × 10^{-10}

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

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

Example of Naïve Bayes Classifier

- Given a Test Record $X = (\text{Refund} = \text{No}, \text{Divorced}, \text{Income} = 120\text{K})$

Naïve 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) = 2/3
P(Marital Status = Divorced | Yes) = 1/3
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(Yes) = 3/10
  P(No) = 7/10
- P(Yes | Divorced) = 1/3 x 3/10 / P(Divorced)
  P(No | Divorced) = 1/7 x 7/10 / P(Divorced)
- P(Yes | Refund = No, Divorced)
  = 1 x 1/3 x 3/10 / P(Divorced, Refund = No)
  P(No | Refund = No, Divorced)
  = 4/7 x 1/7 x 7/10 / P(Divorced, Refund = No)
- P(Yes | Married) = 0 x 3/10 / P(Married)
  P(No | Married) = 4/7 x 7/10 / P(Married)

Issues with Naïve Bayes Classifier

- Consider the table with Tid=7 deleted

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

Naïve Bayes Classifier:

P(Refund = Yes | No) = 2/6
P(Refund = No | No) = 4/6
→ P(Refund = Yes | Yes) = 0
P(Refund = No | Yes) = 1
P(Marital Status = Single | No) = 2/6
→ P(Marital Status = Divorced | No) = 0
P(Marital Status = Married | No) = 4/6
P(Marital Status = Single | Yes) = 2/3
P(Marital Status = Divorced | Yes) = 1/3
P(Marital Status = Married | Yes) = 0/3
For Taxable Income:
If class = No: sample mean = 91, sample variance = 685
If class = Yes: sample mean = 90, sample variance = 25

Given X = (Refund = Yes, Divorced, 120K)

P(X | No) = 2/6 X 0 X 0.0083 = 0

P(X | Yes) = 0 X 1/3 X 1.2 X 10^{-9} = 0

→ Naïve Bayes will not be able to classify X as Yes or No!

Issues with Naïve Bayes Classifier

- If one of the conditional probabilities is zero, then the entire expression becomes zero
  - This is problematic because it will wipe out all information in the other probabilities when they are multiplied
- Need to use other estimates of conditional probabilities than simple fractions -> add pseudocount

Original:

$$P(A_i|C) = \frac{N_{ic}}{N_c}$$

Laplace Smoothing:

$$P(A_i|C) = \frac{N_{ic} + \alpha}{N_c + \alpha d}$$

If $\alpha = 1$, each attribute value of A is added with a pseudocount. α can be a real number parameter.

- d: number of distinct values of attribute A
- N_c: number of instances in the class
- N_{ic}: number of instances having attribute value A_i in class c

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

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

A: attributes

M: mammals

N: non-mammals

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

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

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

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

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

=> Mammals

Naïve Bayes Classifier in Sklearn

```
class sklearn.naive_bayes.CategoricalNB(*, alpha=1.0, force_alpha=True, fit_prior=True, class_prior
=None, min_categories=None)

class sklearn.naive_bayes.BernoulliNB(*, alpha=1.0, force_alpha=True, binarize=0.0, fit_prior=True,
 class_prior=None)

class sklearn.naive_bayes.GaussianNB(*, priors=None, var_smoothing=1e-09)

class sklearn.naive_bayes.MultinomialNB(*, alpha=1.0, force_alpha=True, fit_prior=True, class_prior
=None)
```

```
>>> from sklearn.naive_bayes import GaussianNB
>>> from sklearn.model_selection import train_test_split
>>> clf = GaussianNB()
>>> X_tr, X_te, y_tr, y_te = train_test_split(X, y, train_size=0.9)
>>> clf.fit(X_tr, y_tr)
>>> y_te = clf.predict(X_te)
```

BernoulliNB	Naive Bayes classifier for multivariate Bernoulli models.
CategoricalNB	Naive Bayes classifier for categorical features.
ComplementNB	The Complement Naive Bayes classifier described in Rennie et al. (2003).
GaussianNB	Gaussian Naive Bayes (GaussianNB).
MultinomialNB	Naive Bayes classifier for multinomial models.

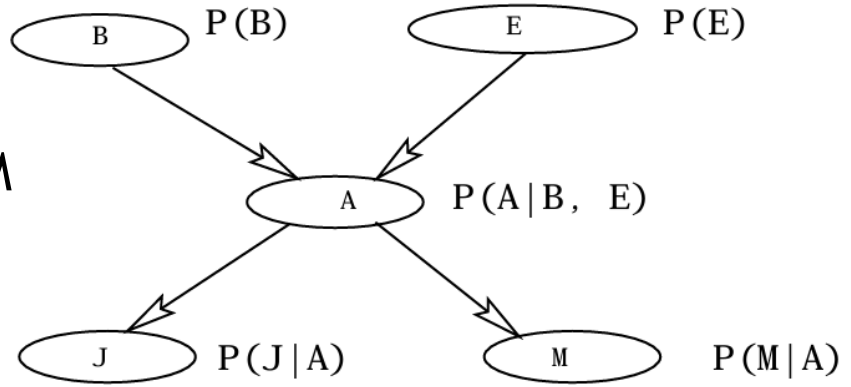
Summary of Naïve Bayes Classifier

- Easy to implement, good results obtained in most cases
- 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
  - E.g., Bayesian Networks

Definition

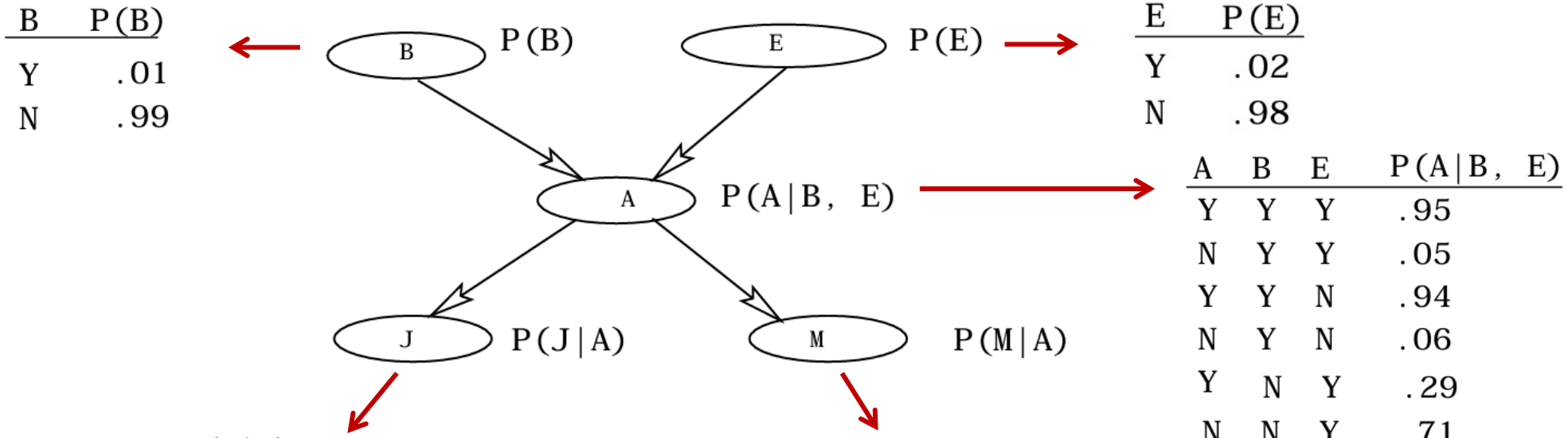
- A Bayesian network is:
  - A graphical representation of conditional independency relationships
    - A directed acyclic graph (DAG), where
    - Each node represents a random variable
    - And is associated with the conditional probability of the node given its parents
  - Also known as belief network, probabilistic network
  - Bayesian network allows conditionally independent variables

    - B, E, A, J, M: random variables
    - Arcs/edges: direct probabilistic dependence
    - B, E are the parents of A, A is the parent of J, M
    - Absence of edges indicates conditional independence: $B \perp E$, $J \perp M \mid A$, $J \perp B \mid A$, etc.
    - Has no loops or cycles



Bayesian Network

- Conditional Probability Tables (CPT)
  - Describe how parents of a variable influence the variable
  - CPT shows the conditional probability of all possible combinations of parent variables



B	P(B)
Y	.01
N	.99

E	P(E)
Y	.02
N	.98

A	B	E	P(A|B, E)
Y	Y	Y	.95
N	Y	Y	.05
Y	Y	N	.94
N	Y	N	.06
Y	N	Y	.29
N	N	Y	.71
Y	N	N	.001
N	N	N	.999

J	A	P(J|A)
Y	Y	.7
N	Y	.3
Y	N	.01
N	N	.99

M	A	P(M|A)
Y	Y	.9
N	Y	.1
Y	N	.05
N	N	.95

Bayesian Network

- As a whole:
  - A Bayesian network represents a factorization of a joint distribution

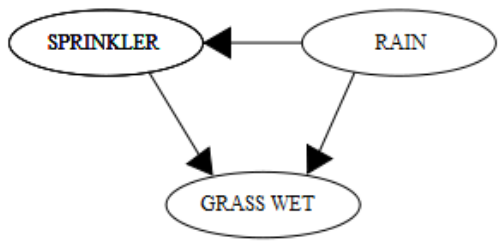
$$P(X_1, X_2, \ldots, X_n) = \prod_{i=1}^{n} P(X_i | pa(X_i))$$

  - Multiplying all the CPTs results in a joint distribution over all variables

Bayesian Network Example

- Two events could cause grass to be wet: either the sprinkler is on or it's raining
- The rain has a direct effect on the use of the sprinkler
  - when it rains, the sprinkler is usually not turned on

	SPRINKLER	
RAIN	T	F
F	0.4	0.6
T	0.01	0.99



RAIN	
T	F
0.2	0.8

		GRASS WET	
SPRINKLER	RAIN	T	F
F	F	0.0	1.0
F	T	0.8	0.2
T	F	0.9	0.1
T	T	0.99	0.01

Then the situation can be modeled with a Bayesian network
All three variables have two possible values, T and F
The joint probability function is:

$$P(G, S, R) = P(G \mid S, R)P(S \mid R)P(R)$$

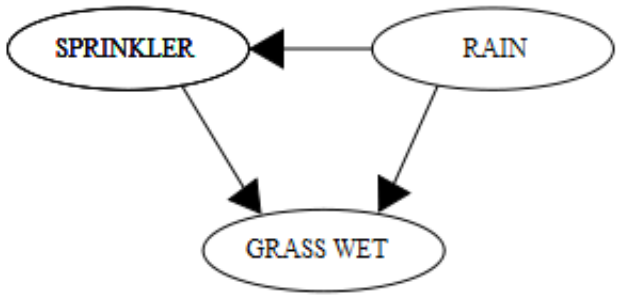
where *G* = *Grass wet*, *S* = *Sprinkler*, and *R* = *Rain*

Bayesian Network Example

- What is the probability that it is raining, given the grass is wet?
  - P(R=T | G=T)

- $P(G,S,R) = P(G \mid S,R)P(S \mid R)P(R)$
- $P(S,R), P(S|R), P(R|S)$ are known
- $P(R)$ is known
- $P(G,S,R), P(G|S,R), P(S,R|G)$ are known

RAIN	SPRINKLER T	SPRINKLER F
F	0.4	0.6
T	0.01	0.99



RAIN T	RAIN F
0.2	0.8

SPRINKLER	RAIN	GRASS WET T	GRASS WET F
F	F	0.0	1.0
F	T	0.8	0.2
T	F	0.9	0.1
T	T	0.99	0.01

$$P(R|G) = \frac{P(R,G)}{P(G)} = \frac{\sum_s P(R,G,S=s)}{\sum_{s,r} P(R=r,G,S=s)}$$

$$\mathrm{P}(R=T \mid G=T) = \frac{\mathrm{P}(G=T,R=T)}{\mathrm{P}(G=T)} = \frac{\sum_{S\in\{T,F\}} \mathrm{P}(G=T,S,R=T)}{\sum_{S,R\in\{T,F\}} \mathrm{P}(G=T,S,R)}$$

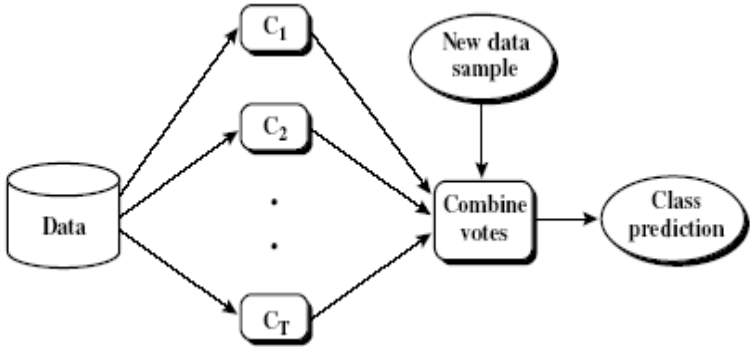
$$= \frac{(0.99 \times 0.01 \times 0.2 = 0.00198_{TTT}) + (0.8 \times 0.99 \times 0.2 = 0.1584_{TFT})}{0.00198_{TTT} + 0.288_{TTF} + 0.1584_{TFT} + 0_{TFF}} \approx 35.77\%.$$

Training Bayesian Networks

- Several scenarios:
  - Given both the network structure and all variables observable: learn only the CPTs
  - Network structure known, some hidden variables: gradient descent method, analogous to neural network learning
  - Network structure unknown, all variables observable: search through the model space to reconstruct network topology
  - Unknown structure, all hidden variables: No good algorithms known for this purpose
- Ref. D. Heckerman: Bayesian networks for data mining

Ensemble Methods

- Ensemble of predictors is a set of predictors whose individual decisions are combined in some way to classify new examples
- Simplest approach:
  - Generate multiple classifiers
  - Each votes on test instance
  - Take majority/average as prediction



- Classifiers are different due to different sampling of training data, or randomized parameters within the classification algorithm
- Aim: take simple mediocre algorithm and transform it into a super classifier without requiring any fancy new algorithm

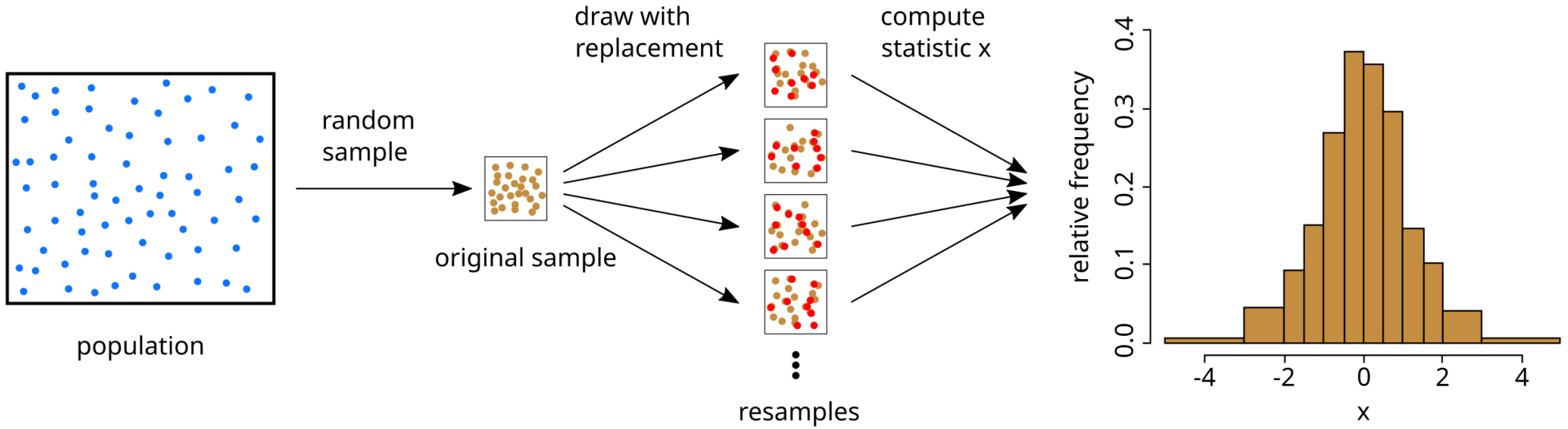
Ensemble Methods: Overview

- Differ in training strategy, and combination method
  - Parallel training with different training sets
    - Bagging (bootstrap aggregation): train separate models on overlapping training sets, average their predictions
  - Sequential training, iteratively re-weighting training examples so current classifier focuses on hard examples: boosting
  - Parallel training with objective encouraging division of labor: mixture of experts

- Notes: Also known as meta-learning

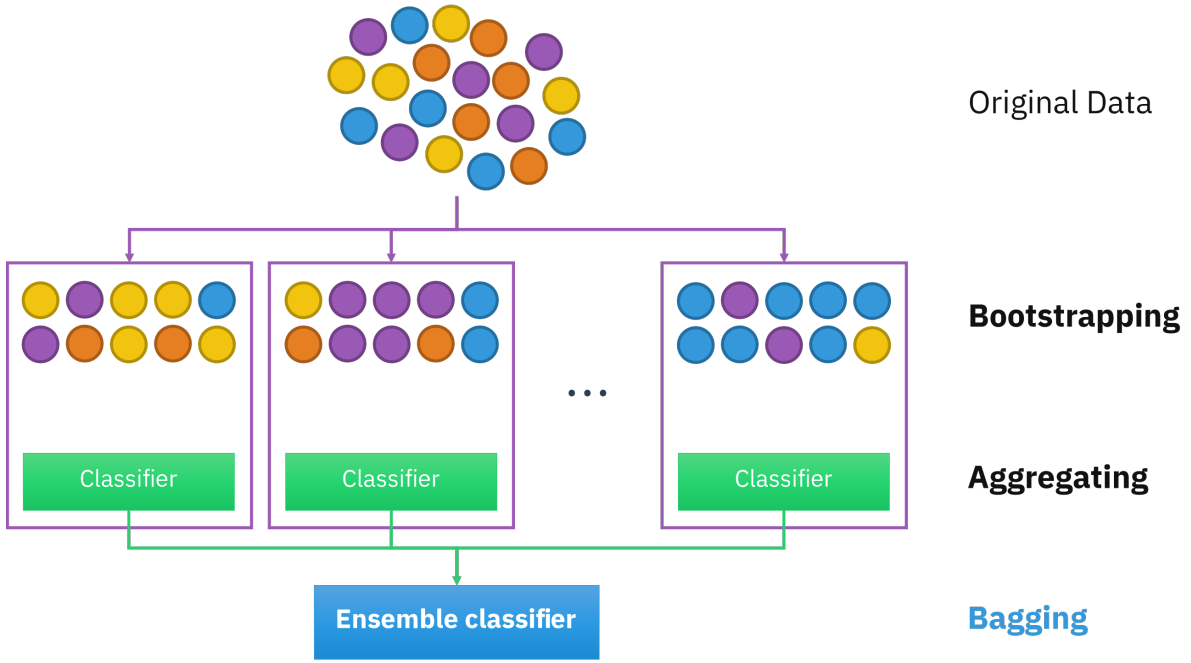
Bagging: Bootstrap Aggregation

- Bootstrap
  - A classical statistics technique
  - A procedure for estimating the distribution of an estimator by resampling (often with replacement) one's data or a model estimated from the data
  - This technique allows estimation of the sampling distribution of almost any statistic using random sampling methods



Bagging: Bootstrap Aggregation

- Bootstrap Aggregating or Bagging
  - A method in which we pick out random subsets from the original data and use them to train multiple different models
  - Once we have the results from each of these models, we let them "vote" upon the results to choose the best prediction



Bagging: Bootstrap Aggregation

- Training:
  1. Given a training set D of size n, bagging generates m new training sets D_i, each of size n_i, by sampling from D uniformly and with replacement (i.e., bootstrap)
     - Some observations may be repeated in each D_i
  2. m models are fitted using the above m bootstrap samples and returned as an ensemble classifier

- If n_i=n, then for large n the set D_i is expected to have the fraction (1 - 1/e) (~63.2%) of the unique examples of D, the rest being duplicates
- Sampling with replacement ensures each bootstrap is independent from its peers, as it does not depend on previous chosen samples when sampling

Bagging: Bootstrap Aggregation

- Inference/Classifying new data:
  1. Each model makes its prediction
  2. The m outputs from m models are combined by averaging the output (for regression) or voting (for classification) (aggregating)

  - Bagging leads to "improvements for unstable procedures" which include, for example, artificial neural networks, classification and regression trees
  - Usually significantly better than a single classifier derived from D
  - Proved improved accuracy in prediction

Bagging: Bootstrap Aggregation

- Random Forest
  - Bagging reduces overfitting by aggregating predictions
  - Works well with decision trees
    - They overfit easily (high variance to reduce)
    - Fast to perform inference
  - Random forest is decision trees + bagging + one more trick
  - To reduce correlation even more, each bootstrap set only considers a random subset of features