

Data Mining Concepts

Summary

- Data Mining
- Knowledge Discovery in Databases (KDD)
- Goals of Data Mining and Knowledge Discovery
- Association Rules
- Additional Data Mining Algorithms
 - Sequential pattern analysis
 - Time Series Analysis

Definitions of Data Mining

- The discovery of new information in terms of patterns or rules from vast amounts of data.
- The process of finding interesting structure in data.
- The process of employing one or more computer learning techniques to automatically analyze and extract knowledge from data.

Knowledge Discovery in Databases (KDD)

- **Data Selection**
 - Identify target subset of data and attributes of interest
- **Data Cleaning**
 - Remove noise and outliers, unify units, create new fields, use denormalization if needed
- **Data Mining**
 - extract interesting patterns
- **Evaluation**
 - present the patterns to the end users in a suitable form, e.g. through visualization

Goals of Data Mining and Knowledge Discovery (PICO)

- **Prediction:**
 - Determine how certain attributes will behave in the future.
 - Example: How much sales volume a store will generate in a given period.
- **Identification:**
 - Identify the existence of an item, event, or activity.
 - Example: intruders trying to break into a system may be identified by the programs executed, files accessed and CPU time per session.
- **Classification:**
 - Partition data into classes or categories.
 - Example: Customers in a supermarket can be categorized into discount-seeking shoppers, loyal regular shoppers, shoppers attached to name brands and infrequent shoppers.
- **Optimization:**
 - Optimize the use of limited resources such as time, space, money or material to maximize output variables such as sales or profits under a given set of constraints.

Types of Discovered Knowledge - I

- Association Rules:
 - These rules correlate the presence of a set of items with another range of values for another set of variables.
 - Examples:
 - When a female shopper buys a handbag, she is likely to buy shoes.
 - An X-ray image containing characteristics a and b is likely to also exhibit characteristics c.
- Classification Hierarchies
 - The goal is to work from an existing set of events or transactions to create a hierarchy of classes.
 - Example: A population may be divided into five ranges of credit worthiness based on a history of previous credit transaction
- Sequential Patterns
 - A sequence of actions or events is sought.
 - Example: If a patient underwent cardiac bypass surgery for blocked arteries and later developed high blood pressure within a year of surgery, he or she is likely to suffer from kidney failure within the next 18 months.

Types of Discovered Knowledge - II

- Patterns Within Time Series
 - Detecting similarities within positions of a time series of data.
 - Example: Two products show the same selling pattern in summer but a different one in winter.
- Clustering
 - A given population of events or items can be partitioned/segmented into sets of “similar” elements.
 - Example: The adult population in Egypt can be categorized into five groups from most likely to buy to least likely to buy a new product.

Mining Association Rules

- Retailers can collect and store massive amounts of sales data
 - transaction date and list of items
- Association rules:
 - e.g. 98% customers who purchase “tires” and “auto accessories” also get “automotive services” done
 - Customers who buy mustard and ketchup also buy burgers
 - Goal: find these rules from just transactional data (transaction id + list of items)

Applications

- Can be used for
 - marketing program and strategies
 - cross-marketing
 - catalog design
 - add-on sales
 - store layout
 - customer segmentation

Notations

- Items $I = \{i_1, i_2, \dots, i_m\}$
- D : a set of transactions
- Each transaction $T \subseteq I$
 - has an identifier TID
- Association Rule
 - $X \rightarrow Y$
 - $X, Y \subset I$
 - $X \cap Y = \emptyset$

Confidence and Support

- Association rule $X \rightarrow Y$
- Confidence $c = |\text{Tr. with } X \text{ and } Y| / |\text{Tr. with } X|$
 - $c\%$ of transactions in D that contain X also contain Y
- Support $s = |\text{Tr. with } X \text{ and } Y| / |\text{all Tr.}|$
 - $s\%$ of transactions in D contain X and Y .

Support Example

TID	Cereal	Beer	Bread	Bananas	Milk
1	X		X		X
2	X		X	X	X
3		X			X
4	X			X	
5			X		X
6	X				X
7		X		X	
8			X		

- Support(Cereal)
 - $4/8 = .5$
- Support(Cereal → Milk)
 - $3/8 = .375$

Confidence Example

TID	Cereal	Beer	Bread	Bananas	Milk
1	X		X		X
2	X		X	X	X
3		X			X
4	X			X	
5			X		X
6	X				X
7		X		X	
8			X		

- Confidence(Cereal \rightarrow Milk)
 - $3/4 = .75$
- Confidence(Bananas \rightarrow Bread)
 - $1/3 = .33333...$

$X \rightarrow Y$ is not a Functional Dependency

For functional dependencies

- F.D. = two tuples with the same value of X must have the same value of Y
 - $X \rightarrow Y \Rightarrow XZ \rightarrow Y$ (concatenation)
 - $X \rightarrow Y, Y \rightarrow Z \Rightarrow X \rightarrow Z$ (transitivity)

For association rules

- $X \rightarrow A$ does not mean $XY \rightarrow A$
 - May not have the minimum support
 - Assume one transaction $\{AX\}$
- $X \rightarrow A$ and $A \rightarrow Z$ do not mean $X \rightarrow Z$
 - May not have the minimum confidence
 - Assume two transactions $\{XA\}, \{AZ\}$

Problem Definition

- Input
 - a set of transactions D
 - Can be in any form – a file, relational table, etc.
 - min support (minsup)
 - min confidence (minconf)
- Goal: generate all association rules that have
 - support \geq minsup and
 - confidence \geq minconf

Decomposition into two subproblems

- 1. Apriori and AprioriTID:
 - for finding “large” itemsets with support \geq minsup
 - all other itemsets are “small”
- 2. Then use another algorithm to find rules $X \rightarrow Y$ such that
 - Both itemsets $X \cup Y$ and X are large
 - $X \rightarrow Y$ has confidence \geq minconf

Basic Ideas - 1

- Q. Which itemset can possibly have larger support: ABCD or AB
 - i.e. when one is a subset of the other?
- Ans: AB
 - any subset of a large itemset must be large
 - So if AB is small, no need to investigate ABC, ABCD etc.

Basic Ideas - 2

- Start with individual (singleton) items {A}, {B}, ...
- In subsequent passes, extend the “large itemsets” of the previous pass as “seed”
- Generate new potentially large itemsets (candidate itemsets)
- Then count their actual support from the data
- At the end of the pass, determine which of the candidate itemsets are actually large
 - becomes seed for the next pass
- Continue until no new large itemsets are found
- Benefit: candidate itemsets are generated using the previous pass, without looking at the transactions in the database
 - Much smaller number of candidate itemsets are generated

Apriori vs. AprioriTID

- Both follow the basic ideas in the previous slides
- AprioriTID has the additional property that the database is not used at all for counting the support of candidate itemsets after the first pass
 - An “encoding” of the itemsets used in the previous pass is employed
 - Size of the encoding becomes smaller in subsequent passes – saves reading efforts

Notations

- Assume the database is of the form $\langle \text{TID}, i_1, i_2, \dots \rangle$ where items are stored in lexicographic order
- **TID** = identifier of the transaction
- Also works when the database is “normalized”: each database record is $\langle \text{TID}, \text{item} \rangle$ pair

k -itemset	An itemset having k items.
L_k	Set of large k -itemsets (those with minimum support). Each member of this set has two fields: i) itemset and ii) support count.
C_k	Set of candidate k -itemsets (potentially large itemsets). Each member of this set has two fields: i) itemset and ii) support count.

Algorithm Apriori

$L_1 = \{large\ 1\text{-itemsets}\}$

Count individual item occurrences

For ($k = 2; L_{k-1} \neq \phi; k++$) do begin

$C_k = \text{apriori-gen}(L_{k-1});$

Generate new k-itemsets candidates

forall transactions $t \in D$ do begin

count = 0

$C_t = \text{subset}(C_k, t)$

Find the support of all the candidates

forall candidates $c \in C_t$ do

$C_t =$ candidates contained in t

$c.count++;$

increment count

end

end

$L_k = \{ c \in C_k / c.count \geq minsup \}$

Take only those with support $\geq minsup$

end

$Answer = \bigcup_k L_k;$

Apriori-Gen

- Takes as argument L_{k-1} (the set of all large $k-1$ -itemsets)
- Returns a superset of the set of all large k -itemsets by augmenting L_{k-1}

● Join step

$L_{k-1} \bowtie L_{k-1}$

p and q are two large
(k-1)-itemsets identical in all k-2
first items.

insert into C_k

select $p.item_1, p.item_2, p.item_{k-1}, q.item_{k-1}$

from $L_{k-1} p, L_{k-1} q$

where $p.item_1 = q.item_1, \dots, p.item_{k-2} = q.item_{k-2}, p.item_{k-1} < q.item_{k-1}$

Join by adding the last item of q to p

● Prune step

forall *itemsets* c of C_k do

forall (k-1)-subsets s of c do

if (s not in L_{k-1}) then

delete c from C_k

Check all the subsets, remove all

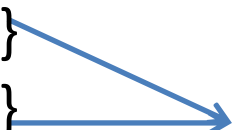
Apriori-Gen Example - 1

Step 1: Join ($k = 4$)

Assume numbers 1-5 correspond to individual items

L_3

C_4

- {1,2,3}
 - {1,2,4}
 - {1,3,4}
 - {1,3,5}
 - {2,3,4}
- 
- {1,2,3,4}

Apriori-Gen Example - 2

Step 1: Join ($k = 4$)

Assume numbers 1-5 correspond to individual items

L_3

- {1,2,3}
- {1,2,4}
- {1,3,4}
- {1,3,5}
- {2,3,4}

C_4

- {1,2,3,4}
- {1,3,4,5}



Apriori-Gen Example - 3

Step 2: Prune ($k = 4$)

- Remove itemsets that can't have the required support because there is a subset in it which doesn't have the level of support i.e. not in the previous pass ($k-1$)

L_3

- {1,2,3}
- {1,2,4}
- {1,3,4}
- {1,3,5}
- {2,3,4}

C_4

- {1,2,3,4}
- ~~{1,3,4,5}~~

No {1,4,5} exists in L_3
Rules out {1, 3, 4, 5}

Problem with Apriori

- Every pass goes over the entire dataset
- Database of transactions is massive
 - Can be millions of transactions added an hour
- Scanning database is expensive
 - In later passes transactions are likely NOT to contain large itemsets
 - Don't need to check those transactions
- Solutions
 - AprioriTID
 - Hybrid

Discovering Rules

(from the full version of the paper)

Naïve algorithm:

- For every large itemset p
 - Find all non-empty subsets of p
 - For every subset q
 - Produce rule $q \rightarrow (p-q)$
 - Accept if $\text{support}(p) / \text{support}(q) \geq \text{minconf}$

Checking the subsets

- For efficiency, generate subsets using recursive DFS
- If a subset q does not produce a rule, we do not need to check for subsets of q

Example

Given itemset : ABCD

If $ABC \rightarrow D$ does not have enough confidence
then $AB \rightarrow CD$ does not hold

minconf

\geq

$\text{Confidence}(ABC \rightarrow D) = \text{Support}(ABCD) / \text{Support}(ABC)$

\geq

$\text{Confidence}(AB \rightarrow CD) = \text{Support}(ABCD) / \text{Support}(AB)$

More Optimizations

Example:

If $AB \rightarrow CD$ holds

- $\text{conf} = \text{support}(ABCD) / \text{support}(AB) \geq \text{minconf}$

then so do $ABC \rightarrow D$ and $ABD \rightarrow C$

- $\text{conf} = \text{support}(ABCD) / \text{support}(ABC)$
- $\text{conf} = \text{support}(ABCD) / \text{support}(ABD)$

In general,

- If $(p-q) \rightarrow q$ holds then all the rules $(p-q') \rightarrow q'$ must hold
 - where $q' \subseteq q$ and is non-empty

Idea

- Start with 1-item consequent and generate larger consequents
- If a consequent does not hold, do not look for bigger ones
- The candidate set will be a subset of the simple algorithm

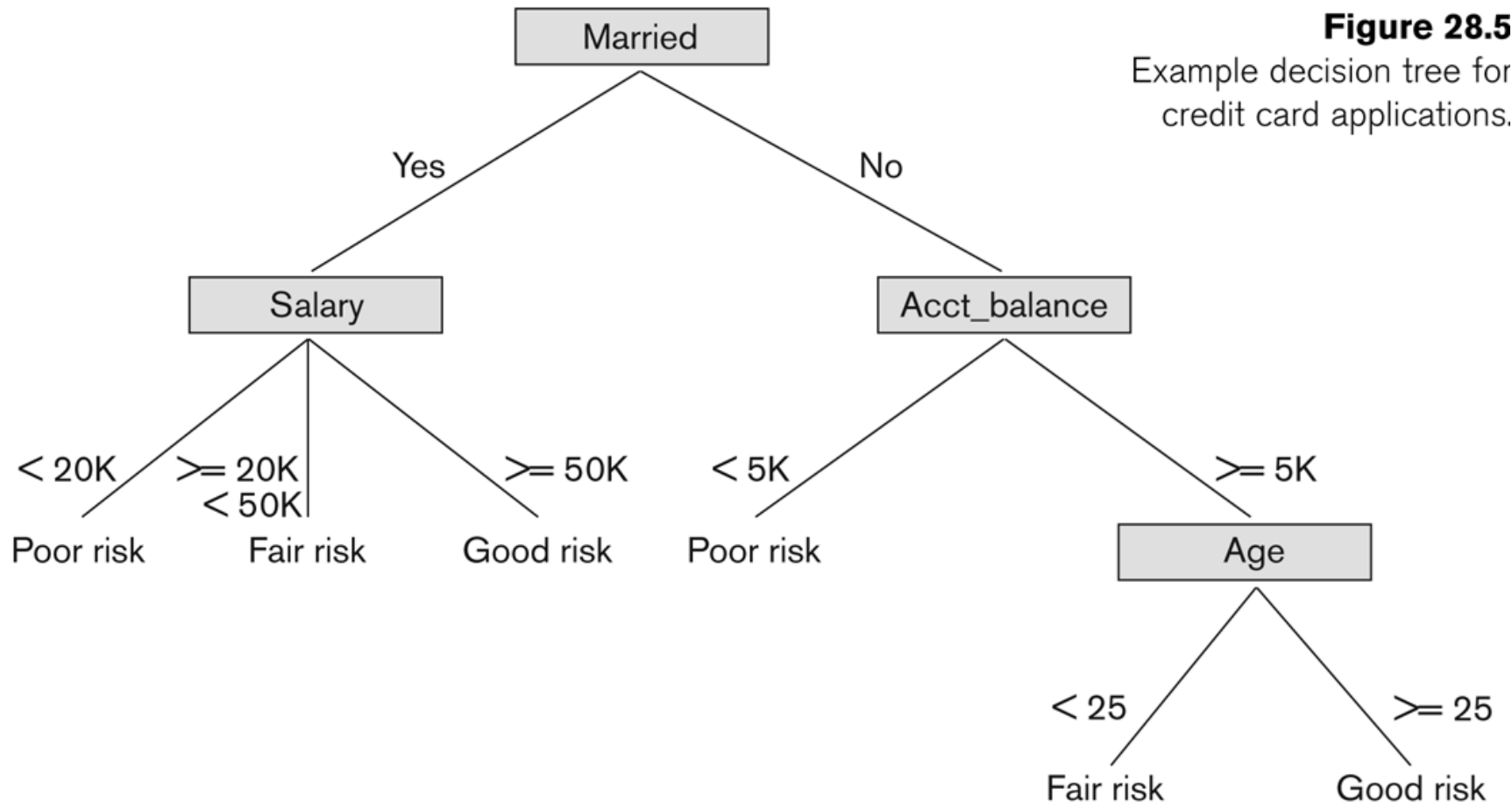
Complications seen with Association Rules

- The cardinality of itemsets in most situations is extremely large.
- Association rule mining is more difficult when transactions show variability in factors such as geographic location and seasons.
- Item classifications exist along multiple dimensions.
- Negative associations is more difficult. For example: 60% of customers who buy potato chips do not buy bottled water.
- Quality of data: missing, erroneous, conflicting and redundant.

Classification

- **Classification** is the process of learning a model that is able to describe different classes of data.
- Learning is **supervised** as the classes to be learned are predetermined.
- Learning is accomplished by using a training set of pre-classified data.
- The model produced is usually in the form of a decision tree or a set of rules.

Decision tree for credit card applications

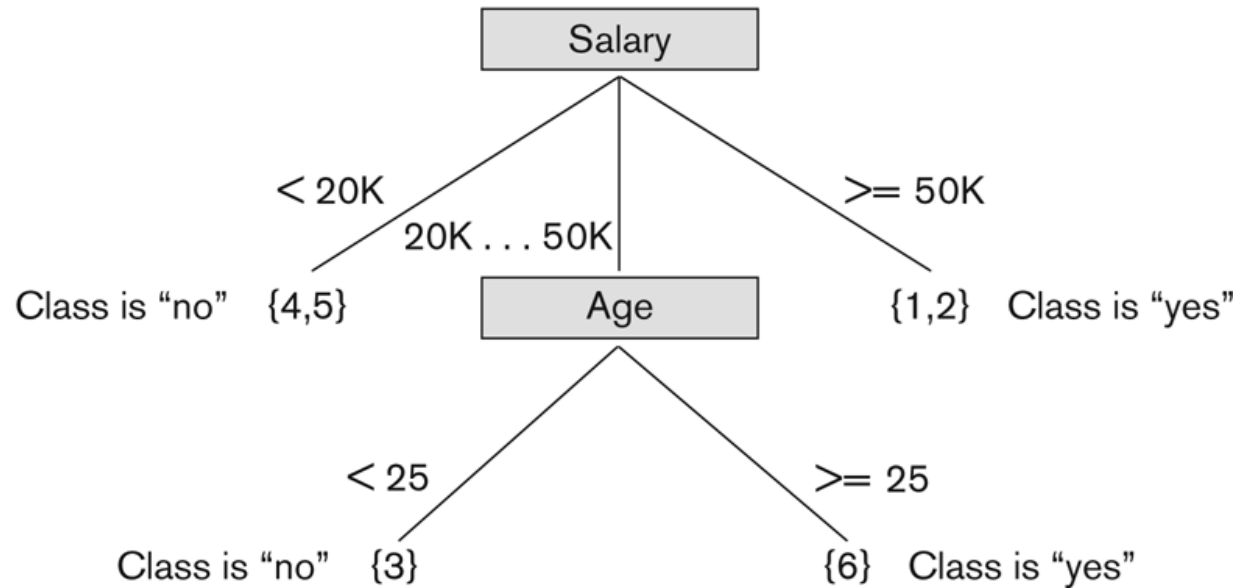


Sample training data for classification algorithm

RID	Married	Salary	Acct_balance	Age	Loanworthy
1	no	$\geq 50K$	$< 5K$	≥ 25	yes
2	yes	$\geq 50K$	$\geq 5K$	≥ 25	yes
3	yes	20K. . .50K	$< 5K$	< 25	no
4	no	$< 20K$	$\geq 5K$	< 25	no
5	no	$< 20K$	$< 5K$	≥ 25	no
6	yes	20K. . .50K	$\geq 5K$	≥ 25	yes

Figure 28.7

Decision tree based on sample training data where the leaf nodes are represented by a set of RIDs of the partitioned records.



An Example Rule

- | Here is one of the rules extracted from the decision tree of Figure 28.7.

IF 50K > salary >= 20K

AND age >=25

THEN class is “yes”

Clustering

- Unsupervised learning or clustering builds models from data without predefined classes.
- The goal is to place records into groups where the records in a group are highly similar to each other and dissimilar to records in other groups.
- The **k-Means** algorithm is a simple yet effective clustering technique.

K-means

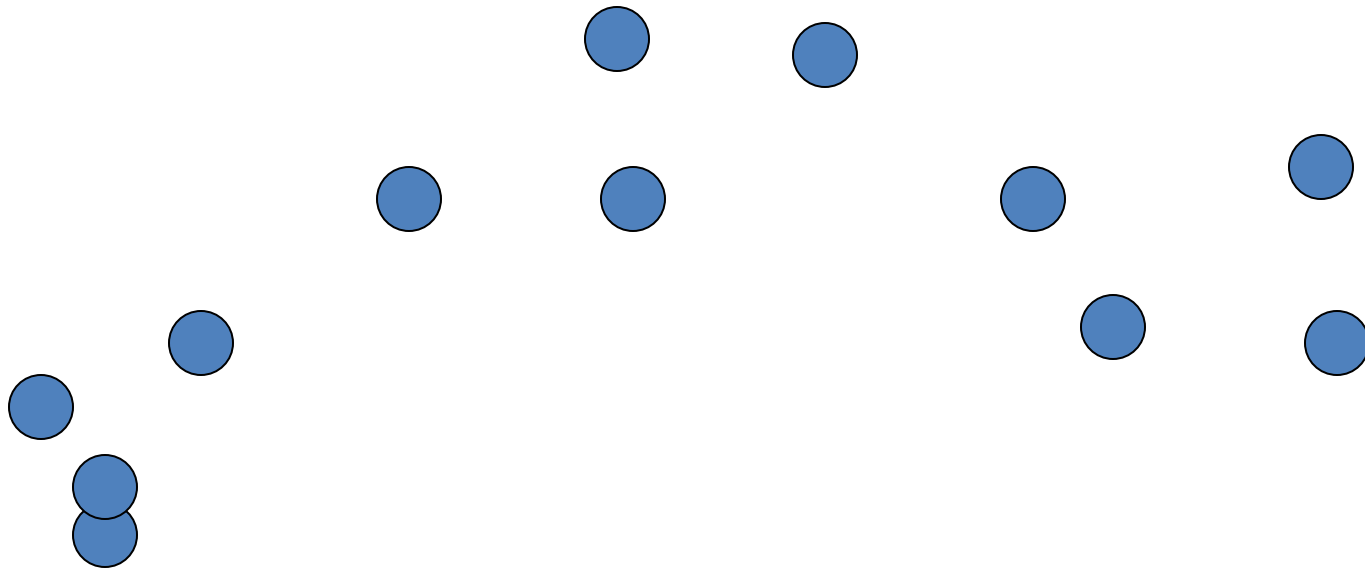
Most well-known and popular clustering algorithm:

Start with some initial cluster centers

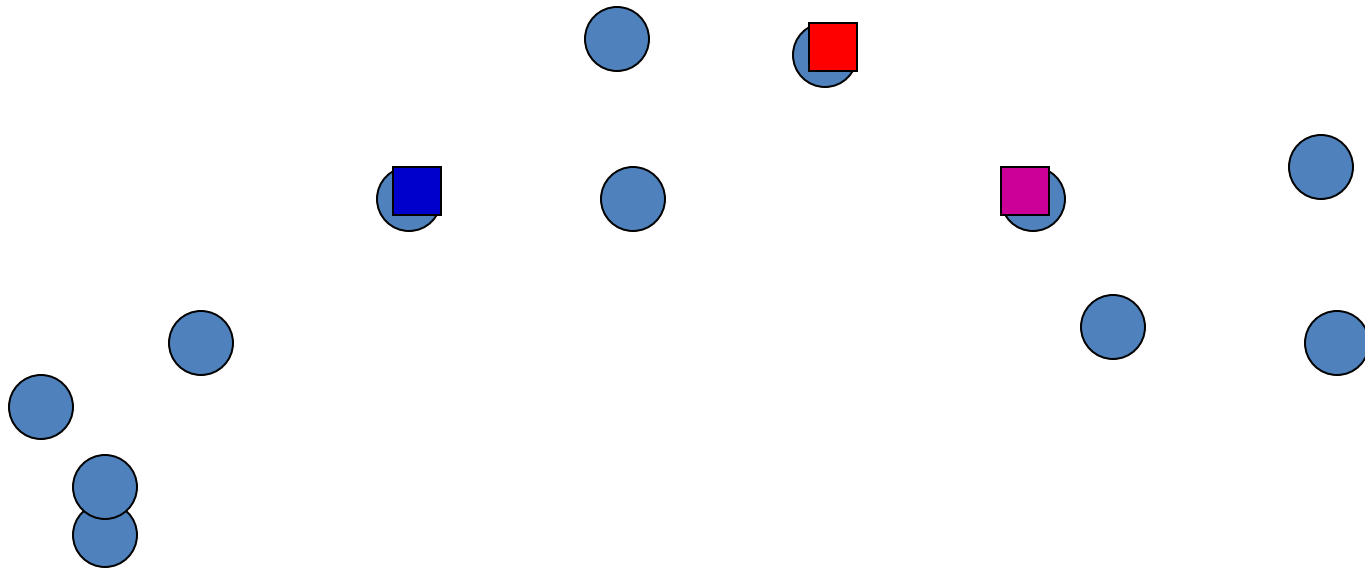
Iterate:

- Assign/cluster each example to closest center
- Recalculate centers as the mean of the points in a cluster

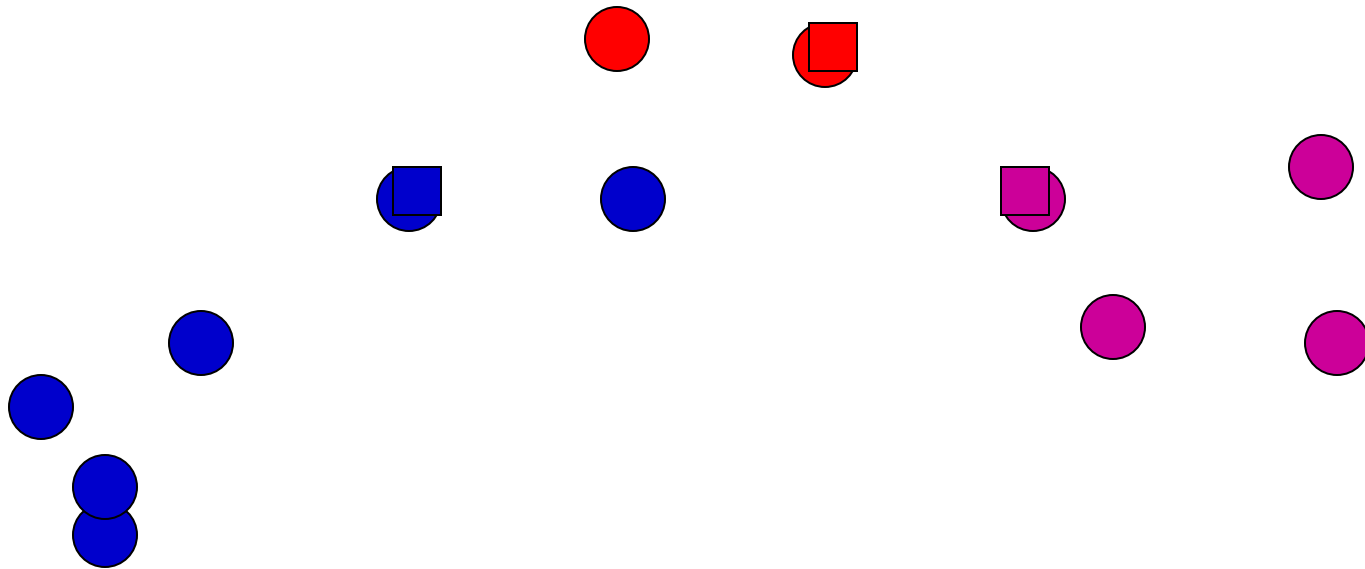
K-means: an example



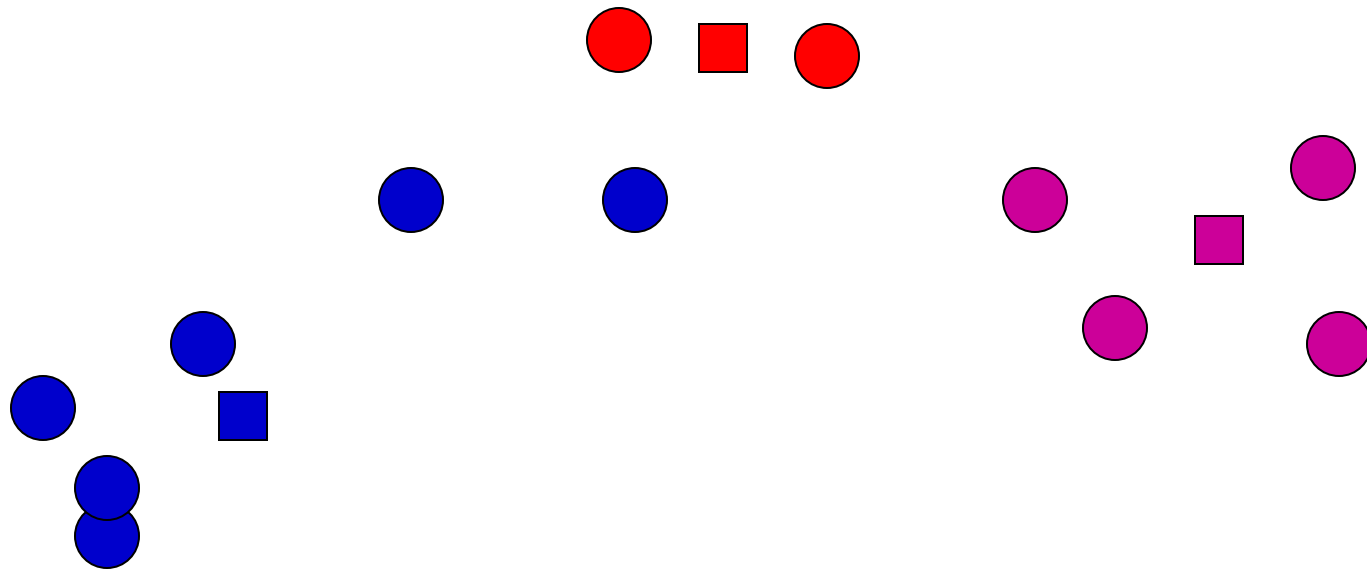
K-means: Initialize centers randomly



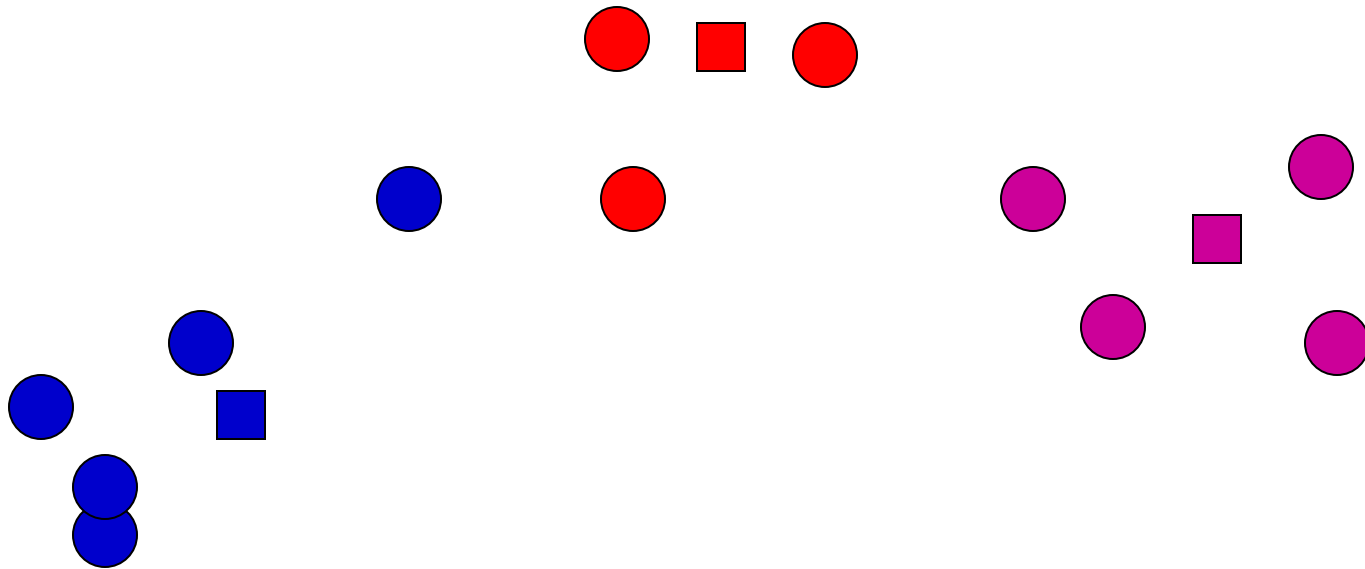
K-means: assign points to nearest center



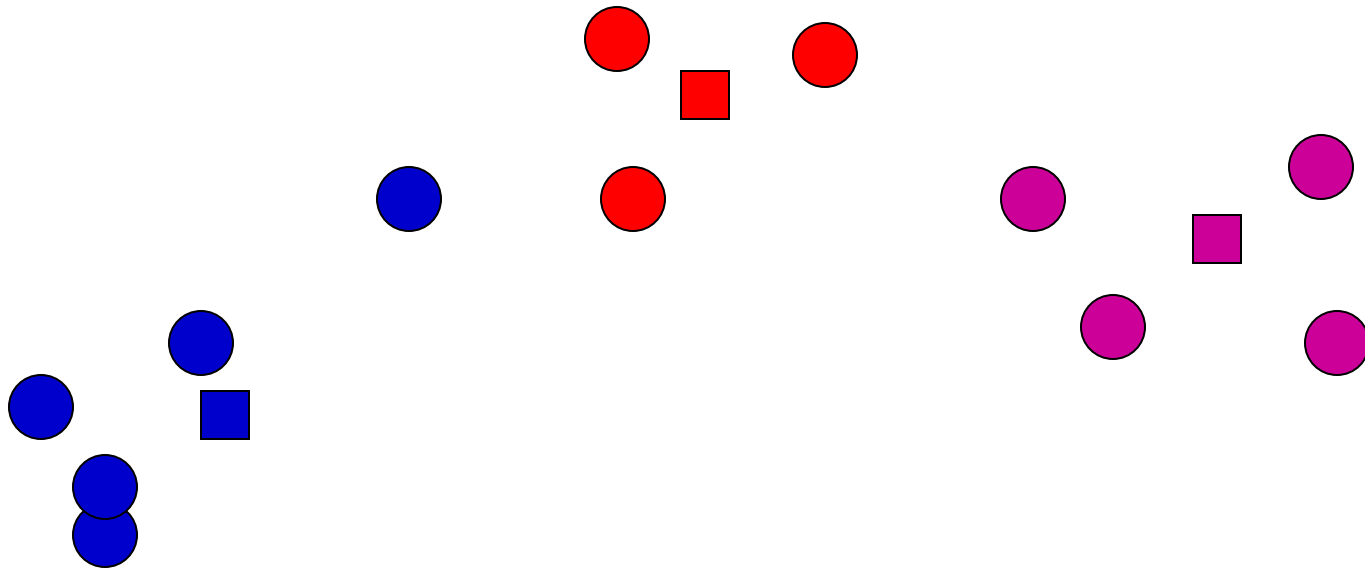
K-means: readjust centers



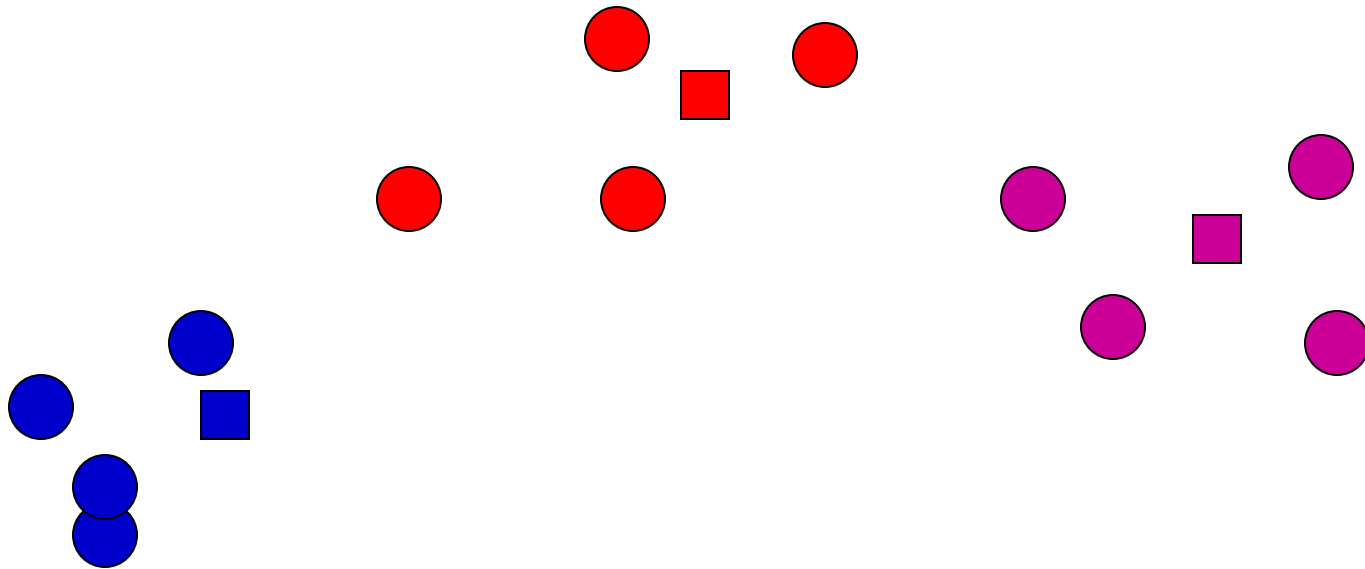
K-means: assign points to nearest center



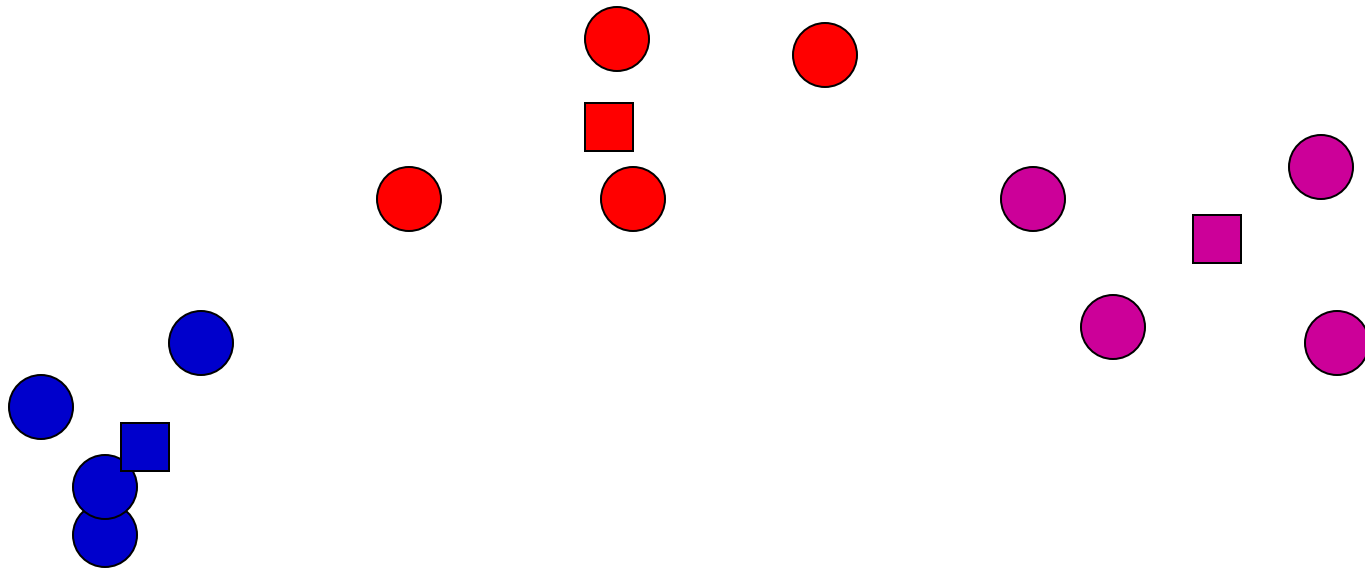
K-means: readjust centers



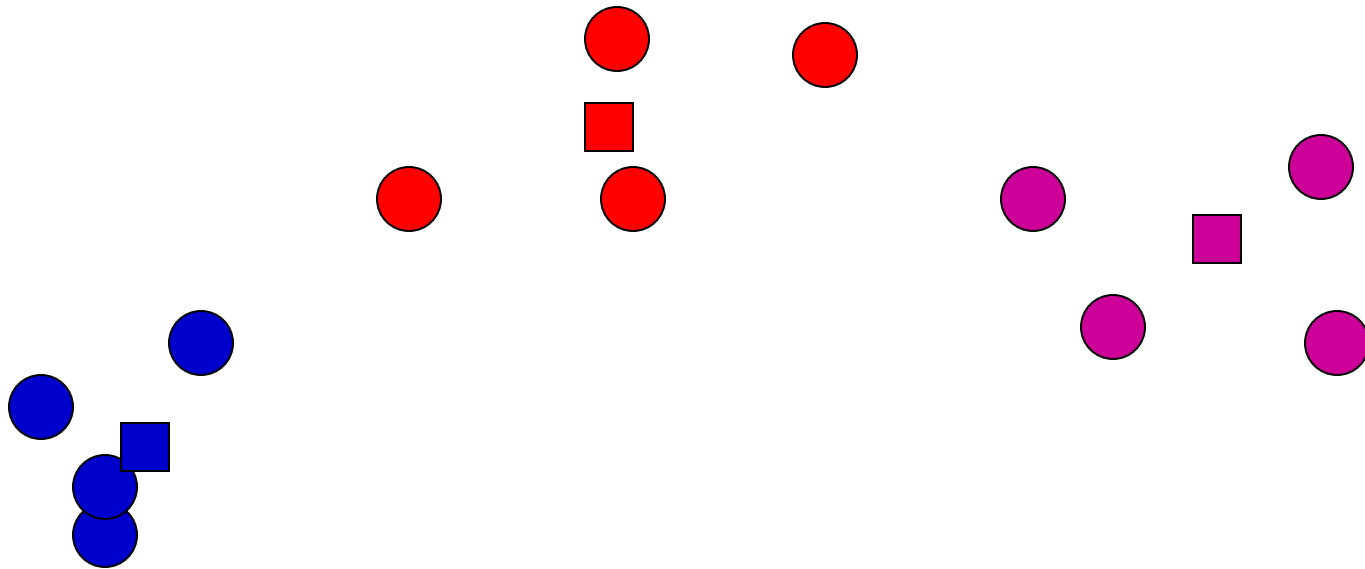
K-means: assign points to nearest center



K-means: readjust centers



K-means: assign points to nearest center



No changes: Done

Additional Data Mining Methods

- **Sequential pattern analysis**
- **Time Series Analysis**

Sequential Pattern Analysis

- Transactions ordered by time of purchase form a sequence of **itemsets**.
- The problem is to find all **subsequences** from a given set of sequences that have a minimum support.
- The sequence $S_1, S_2, S_3, ..$ is a predictor of the fact that a customer purchasing itemset S_1 is likely to buy S_2 , and then S_3 , and so on.

Time Series Analysis

- **Time series** are sequences of events. For example, the closing price of a stock is an event that occurs each day of the week.
- Time series analysis can be used to identify the price trends of a stock or mutual fund.
- Time series analysis is an extended functionality of **temporal** data management.
- Example: Find the quarter during which the stock had the most percentage gain or loss

Data Mining Applications

- **Marketing**

- Marketing strategies and consumer behavior

- **Finance**

- Fraud detection, creditworthiness and investment analysis

- **Manufacturing**

- Resource optimization

- **Health**

- Image analysis, side effects of drug, and treatment effectiveness

Commercial Data Mining Tool

Table 28.1

Some representative data mining tools

Company	Product	Technique	Platform	Interface*
Acknosoft	Kate	Decision trees, Case-based reasoning	Win NT UNIX	Microsoft Access
Angoss	Knowledge Seeker	Decision trees, Statistics	Win NT	ODBC
Business Objects	Business Miner	Neural nets, Machine learning	Win NT	ODBC
CrossZ	QueryObject	Statistical analysis Optimization algorithm	Win NT MVS UNIX	ODBC
Data Distilleries	Data Surveyor	Comprehensive, Can mix different types of data mining	UNIX	ODBC ODMG-compliant
DBMiner Technology Inc.	DBMiner	OLAP analysis, Associations, Classification, Clustering algorithms	Win NT	Microsoft 7.0 OLAP
IBM	Intelligent Miner	Classification, Association rules, Predictive models	UNIX (AIX)	IBM DB2
Megaputer Intelligence	Polyanalyst	Symbolic knowledge acquisition, Evolutionary programming	Win NT OS/2	ODBC Oracle DB2
NCR	Management Discovery Tool (MDT)	Association rules	Win NT	ODBC
SAS	Enterprise Miner	Decision trees, Association rules, Neural nets, Regression, Clustering	UNIX (Solaris) Win NT Macintosh	ODBC Oracle AS/400
Purple Insight	MineSet	Decision trees, Association rules	UNIX (Irix)	Oracle Sybase Informix

*ODBC: Open Data Base Connectivity

ODMG: Object Data Management Group

Summary

- | Data Mining
- | Knowledge Discovery in Databases (KDD)
- | Goals of Data Mining and Knowledge Discovery
- | Association Rules
- | Additional Data Mining Algorithms
 - Sequential pattern analysis
 - Time Series Analysis