Machine Learning

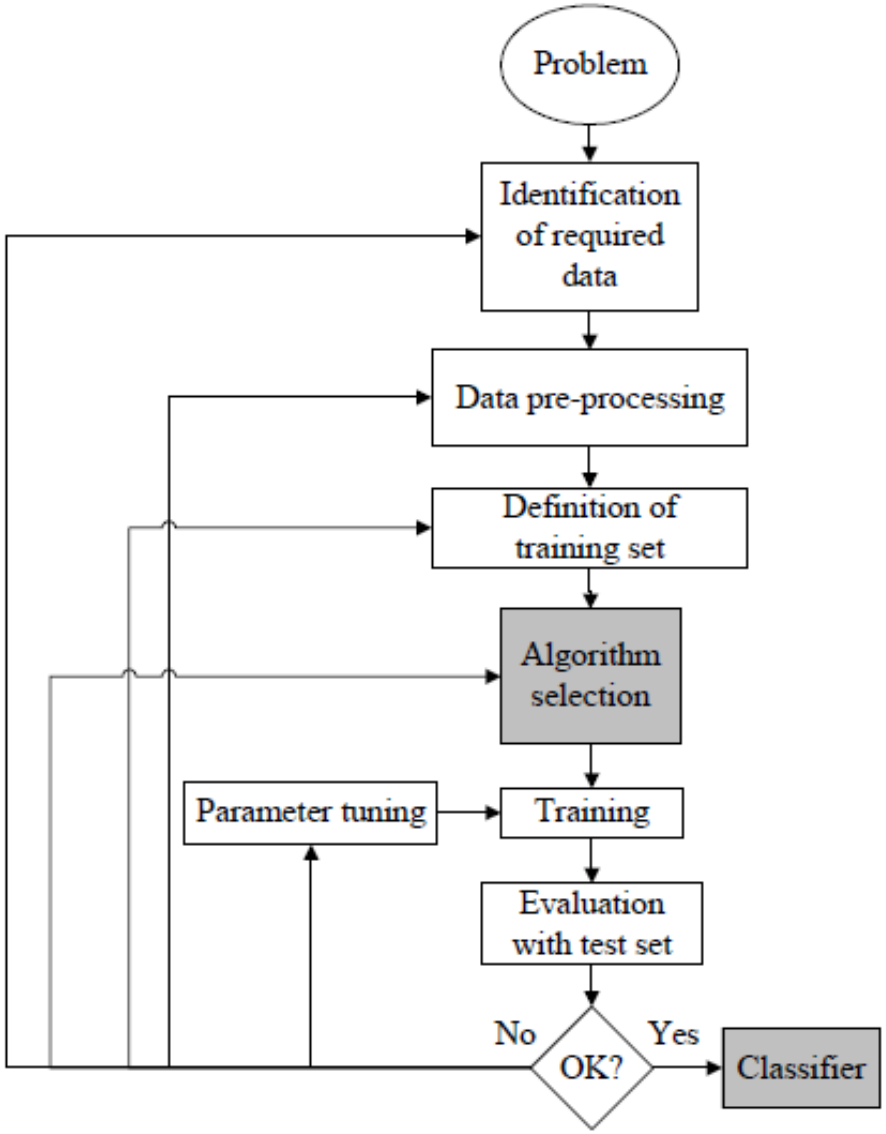
BS/MS (Computer Science)

IQRA UNIVERSITY IU

Lecture: 05 and 06

Designing a Learning System
(Issues Related to Learning System)

The Process of Learning Model



Issues Related to Learning Model

1. Feature Selection
2. Correlation and Multicollinearity among the data variables
3. Imbalance Data Set
4. Algorithm Selection (minimize cost function)

Issue No. 1: Feature Selection

- Given a set of features $F = \{ f_1, \ldots, f_i, \ldots, f_n \}$
  the Feature Selection problem is
  to find a subset $F' \subseteq F$ that "maximizes the learners ability to classify patterns".

Kind of Feature Selection

1. Feature Subset Selection
2. Rank wise Individual Feature Selection

- Methods:
  - Wrapper methods
    - Make use of the algorithm that will be used to build the final classifier
  - Filter methods

A Wrapper Method – Example

- Say we have predictors A, B, C and classifier M. We want to predict T given the smallest possible subset of {A,B,C}, while achieving maximal performance

FEATURE SET	CLASSIFIER	PERFORMANCE
{A,B,C}	*M*	**98%**
{A,B}	*M*	**98%**
{A,C}	*M*	77%
{B,C}	*M*	56%
{A}	*M*	89%
{B}	*M*	90%
{C}	*M*	91%
{.}	*M*	85%

A Filter Method (Variable Ranking Methods)

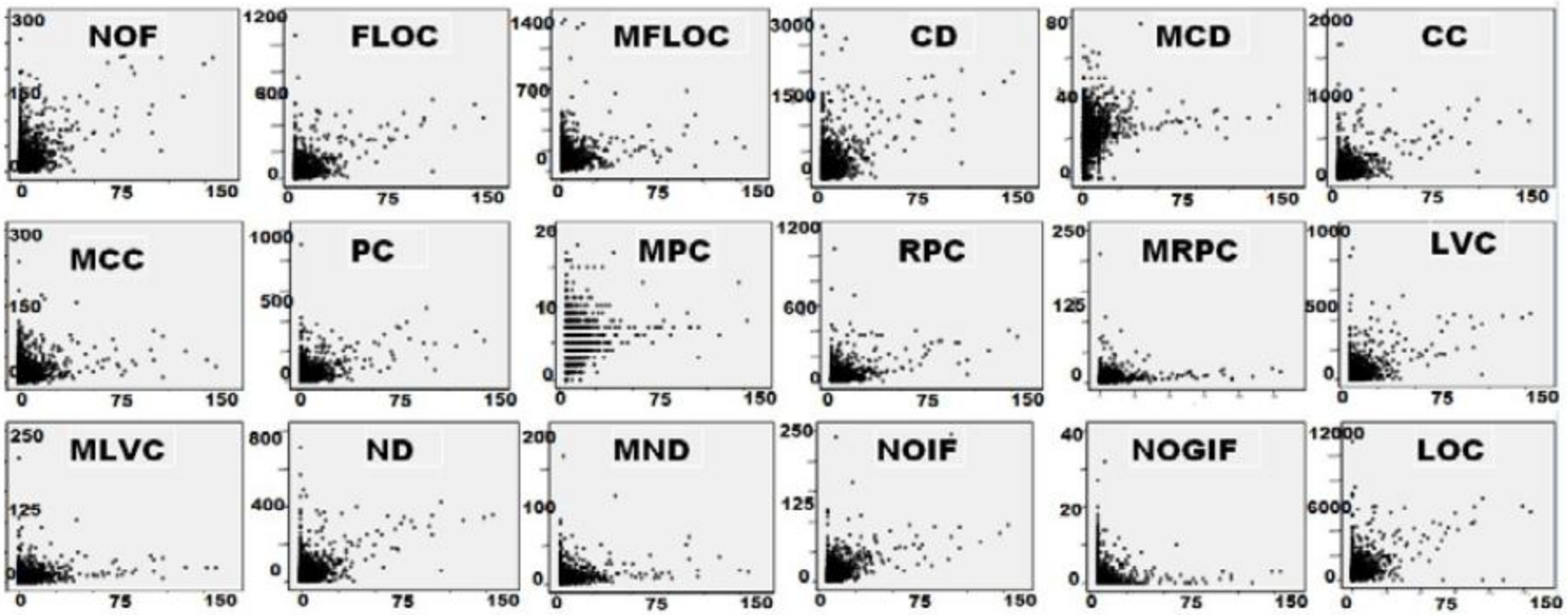
- A filter method does not make use of C, but rather attempts to find predictive subsets of the features by making use of simple statistics computed from the empirical distribution.
  - Ranks features in terms of the mutual information between the features and the class label

A Filter Method – Example

- In the filter approach we do not rely on running a particular classifier and searching in the space of feature subsets; instead we select features on the basis of statistical properties. A classic example is univariate associations:

FEATURE	ASSOCIATION WITH TARGET	
{A}	89%	Threshold gives suboptimal solution
{B}	90%	Threshold gives optimal solution
{C}	91%	Threshold gives suboptimal solution

Issue No. 2: Multi Collinearity Among Features (Scatter Plot)



Important Points

- Feature selection can significantly increase the performance of a learning algorithm (both accuracy and computation time) – but it is not easy!
- One can work on problems with very high- dimensional feature-spaces
- Relevance <-> Optimality
- Correlation and Mutual information between single variables and the target are often used as Ranking-Criteria of variables.

Issue No. 3: The Class Imbalance Problem

- Data sets are said to be balanced if there are, approximately, as many positive examples of the concept as there are negative ones.
- There exist many domains that do not have a balanced data set.
- ***Examples:***
  - Helicopter Gearbox Fault Monitoring
  - Discrimination between Earthquakes and Nuclear Explosions
  - Document Filtering
  - Detection of Oil Spills
  - Detection of Fraudulent Telephone Calls

The Class Imbalance Problem

- The problem with class imbalances is that standard learners are often biased towards the majority class.
- That is because these classifiers attempt to reduce global quantities such as the error rate, not taking the data distribution into consideration.
- As a result examples from the overwhelming class are well-classified whereas examples from the minority class tend to be misclassified.

How to Deal with Class Imbalance

There are four main ways to deal with class imbalances:

1. At the Data Level: Re-sampling,
2. At the Algorithmic Level: Re-weighing/Adjusting the cost of various classes, (modification of existing learning algorithms)
3. Combining Methods,
4. Evaluation Metrics

Data Level Methods for Handling Imbalance

1. Undersampling: Random and Direct Methods
2. Oversampling: Random and Direct Methods
3. Feature Selection for Imbalance Data Sets

Advantage of Resampling

- Re-sampling provides a simple way of biasing the generalization process.
- It can do so by:
  - Generating synthetic samples accordingly biased
  - Controlling the amount and placement of the new samples

SMOTE: A State-of-the-Art Resampling Approach

- SMOTE stands for **Synthetic Minority Oversampling Technique.**
- It is a technique designed by Chawla, Hall, & Kegelmeyer in 2002.
- It combines Informed Oversampling of the minority class with Random Undersampling of the majority class.
- SMOTE currently yields the best results as far as re-sampling and modifying the probabilistic estimate techniques go (Chawla, 2003).

SMOTE's Informed Oversampling Procedure

- For each minority Sample
  - Find its k-nearest minority neighbours
  - Randomly select j of these neighbours
  - Randomly generate synthetic samples along the lines joining the minority sample and its j selected neighbours

  (j depends on the amount of oversampling desired)

SMOTE's Shortcomings

- **Overgeneralization**
  - SMOTE's procedure is inherently dangerous since it blindly generalizes the minority area without regard to the majority class.
  - This strategy is particularly problematic in the case of highly skewed class distributions since, in such cases, the minority class is very sparse with respect to the majority class, thus resulting in a greater chance of class mixture.
- **Lack of Flexibility**
  - The number of synthetic samples generated by SMOTE is fixed in advance, thus not allowing for any flexibility in the re-balancing rate.

Issue No. 4: Algorithm Selection

- Type of learning problem (supervised/unsupervised)
- Input Data
- Number of Input parameters
- Type of class variable
- Cost function

Feature Reduction Using Principal Component Analysis (PCA)

Feature Reduction

- What is feature reduction?
- Why feature reduction?
- Feature reduction algorithms
- Principal Component Analysis (PCA)

What is feature reduction?

- Feature reduction refers to the mapping of the original high-dimensional data onto a lower-dimensional space.
  - Criterion for feature reduction can be different based on different problem settings.
    - Unsupervised setting: minimize the information loss
    - Supervised setting: maximize the class discrimination
- Given a set of data points of p variables
    - Compute the linear transformation (projection)

Why Feature Reduction?

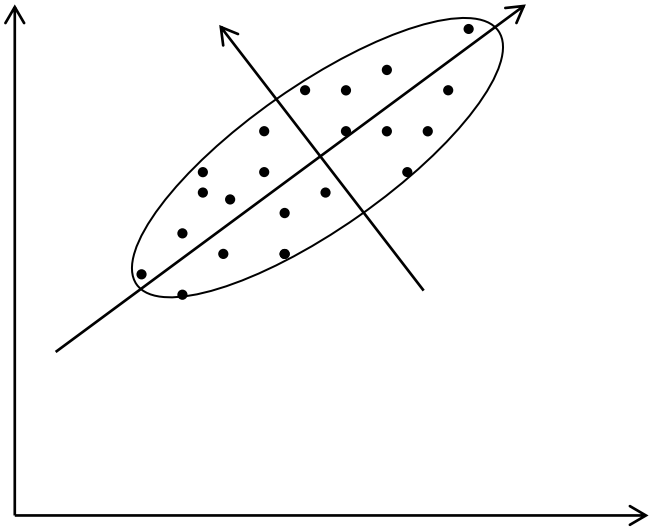
- Most machine learning and data mining techniques may not be effective for high-dimensional data
  - Curse of Dimensionality
  - Query accuracy and efficiency degrade rapidly as the dimension increases.
- The intrinsic dimension may be small.
  - For example, the number of genes responsible for a certain type of disease may be small.

Feature Reduction Algorithms

- Unsupervised
  - Latent Semantic Indexing (LSI): truncated SVD
  - Independent Component Analysis (ICA)
  - **Principal Component Analysis (PCA)**
  - Canonical Correlation Analysis (CCA)
- Supervised
  - Linear Discriminant Analysis (LDA)

Principal Component Analysis

- Eigen Vectors show the direction of axes of a fitted ellipsoid
- Eigen Values show the significance of the corresponding axis
- The larger the Eigen value, the more separation between mapped data
- For high dimensional data, only few of Eigen values are significant



What is PCA?

- Finding Eigen Values and Eigen Vectors
- Deciding on which are significant
- Forming a new coordinate system defined by the significant Eigen vectors
  (→lower dimensions for new coordinates)
- Mapping data to the new space

→ Compressed Data

How is PCA used in Classification/Regression?

- A training set is used for LEARNING phase
  - Applying PCA to training data to form a new coordinate system defined by significant Eigen vectors
  - Representing each data in PCA coordinate system (weights of Eigen vectors)
- A test set is used for TESTING phase
  - Same PCA coordinate system is used
  - Each new data is represented in PCA coordinates
  - New data is recognized as the closest training data (Euclidean distance)

The Objective of Principal Component Analysis (PCA)

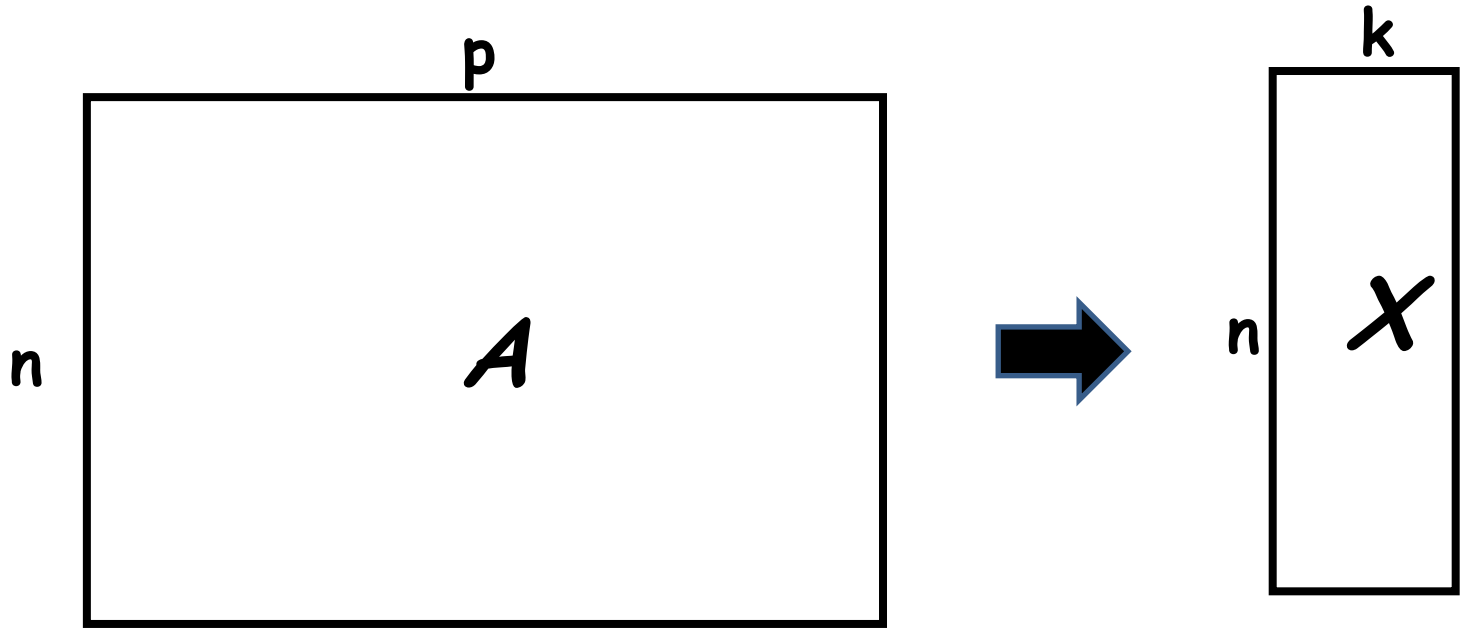
- To discover or to reduce the dimensionality of the data set.
- To remove the collinearity among the data
- To identify new meaningful underlying variables.

Principal Component Analysis (PCA)

- Principal component analysis (PCA) is a mainstay of modern data analysis - a black box that is widely used but poorly understood.
- In PCA, we eliminate those components from high dimensional data set which contribute low variance

Data Reduction

- summarization of data with many (p) variables by a smaller set of (k) derived (synthetic, composite) variables.



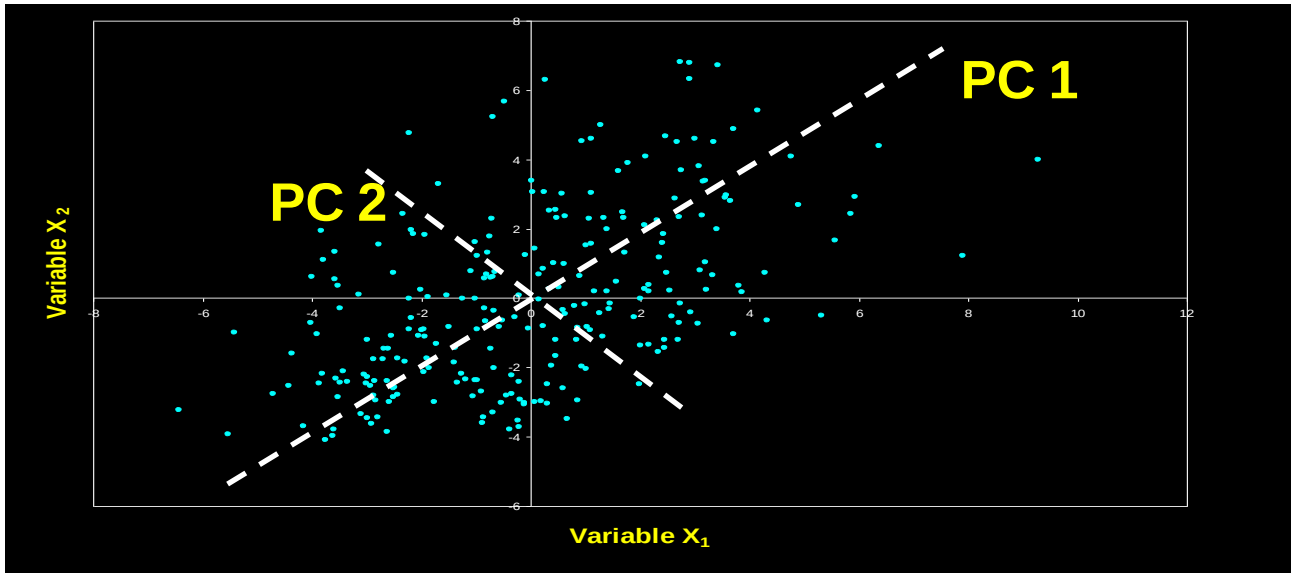
Principal Component Analysis (PCA)

- Takes a data matrix of *n* objects by *p* variables, which may be correlated, and summarizes it by uncorrelated axes (principal components or principal axes) that are linear combinations of the original *p* variables
- The first *k* components display as much as possible of the variation among objects.

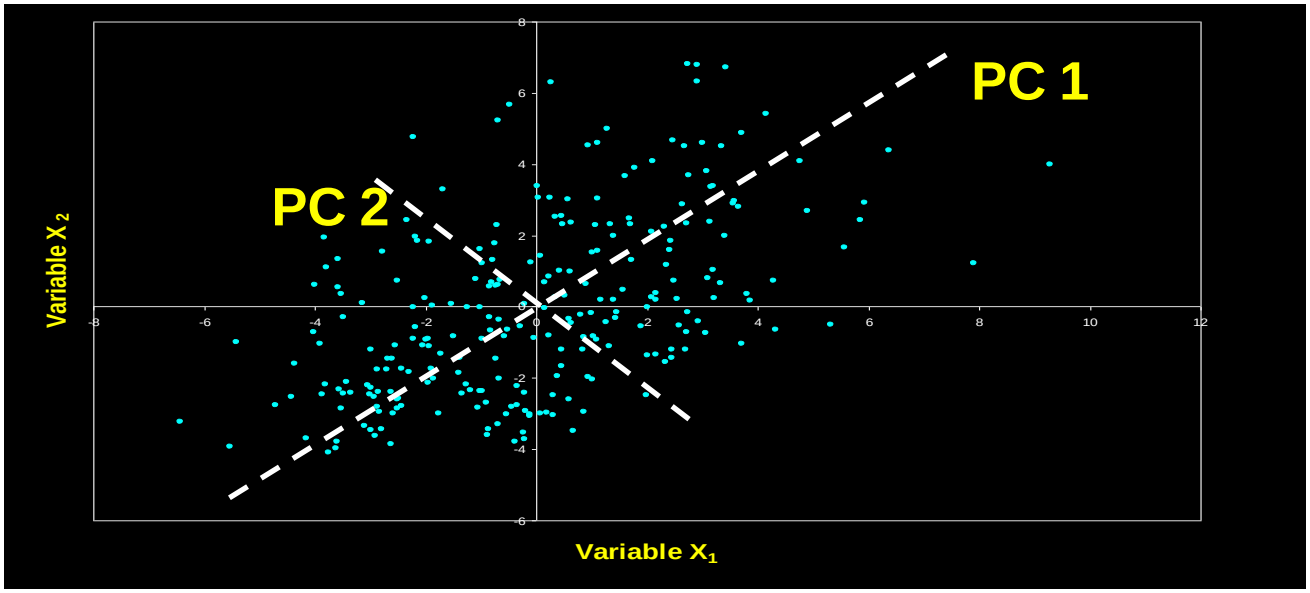
Generalization to *p*-dimensions

- In practice nobody uses PCA with only 2 variables
- The algebra for finding principal axes readily generalizes to *p* variables
- PC 1 is the direction of maximum variance in the *p*-dimensional cloud of points
- PC 2 is in the direction of the next highest variance, subject to the constraint that it has zero covariance with PC 1.
- PC 3 is in the direction of the next highest variance, subject to the constraint that it has zero covariance with both PC 1 and PC 2
- and so on... up to PC *p*

- each principal axis is a linear combination of the original two variables
- $PC_j = a_{i1}Y_1 + a_{i2}Y_2 + \ldots a_{in}Y_n$
- a_{ij}'s are the coefficients for factor i, multiplied by the measured value for variable j



- PC axes are a rigid rotation of the original variables
- PC 1 is simultaneously the direction of maximum variance and a least-squares "line of best fit" (squared distances of points away from PC 1 are minimized).



Background for PCA

- Suppose attributes are A_1 and A_2, and we have n training examples. x's denote values of A_1 and y's denote values of A_2 over the training examples.
- Variance of an attribute:

$$\text{var}(A_1) = \frac{\sum_{i=1}^{n} (x_i - \bar{x})^2}{(n-1)}$$

- Covariance of two attributes:

$$\text{cov}(A_1, A_2) = \frac{\sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y})}{(n-1)}$$

- If covariance is positive, both dimensions increase together. If negative, as one increases, the other decreases. Zero: independent of each other.

- **Covariance matrix**
  - Suppose we have *n* attributes, A_1, ..., A_n.
  - Covariance matrix:

$$C^{n\times n} = (c_{i,j}), \text{ where } c_{i,j} = \text{cov}(A_i, A_j)$$

	Hours(H)	*Mark(M)*
Data	9	39
	15	56
	25	93
	14	61
	10	50
	18	75
	0	32
	16	85
	5	42
	19	70
	16	66
	20	80
Totals	167	749
Averages	13.92	62.42

Covariance:

H	M	$(H_i - \bar{H})$	$(M_i - \bar{M})$	$(H_i - \bar{H})(M_i - \bar{M})$
9	39	-4.92	-23.42	115.23
15	56	1.08	-6.42	-6.93
25	93	11.08	30.58	338.83
14	61	0.08	-1.42	-0.11
10	50	-3.92	-12.42	48.69
18	75	4.08	12.58	51.33
0	32	-13.92	-30.42	423.45
16	85	2.08	22.58	46.97
5	42	-8.92	-20.42	182.15
19	70	5.08	7.58	38.51
16	66	2.08	3.58	7.45
20	80	6.08	17.58	106.89
Total				1149.89
Average				104.54

Table 2.2: 2-dimensional data set and covariance calculation

$$\begin{pmatrix} \text{cov}(H,H) & \text{cov}(H,M) \\ \text{cov}(M,H) & \text{cov}(M,M) \end{pmatrix}$$

$$= \begin{pmatrix} \text{var}(H) & 104.5 \\ 104.5 & \text{var}(M) \end{pmatrix}$$

$$= \begin{pmatrix} 47.7 & 104.5 \\ 104.5 & 370 \end{pmatrix}$$

Covariance matrix

Eigenvectors:

- Let **M be** an $n \times n$ matrix.
  - **v** is an *eigenvector* of M if $M \times \mathbf{v} = \lambda \mathbf{v}$
  - λ is called the *eigenvalue* associated with **v**
- For any eigenvector **v** of **M** and scalar a,

$$\mathbf{M} \times a\mathbf{v} = \lambda a\mathbf{v}$$

- Thus you can always choose eigenvectors of length 1:

$$\sqrt{v_1^2 + \ldots + v_n^2} = 1$$

- If **M** has any eigenvectors, it has n of them, and they are orthogonal to one another.
- Thus eigenvectors can be used as a new basis for a n-dimensional vector space.

PCA

1. Given original data set S = {$\mathbf{x}^1$, ..., $\mathbf{x}^k$}, produce new set by subtracting the mean of attribute A_i from each x_i.

Data =

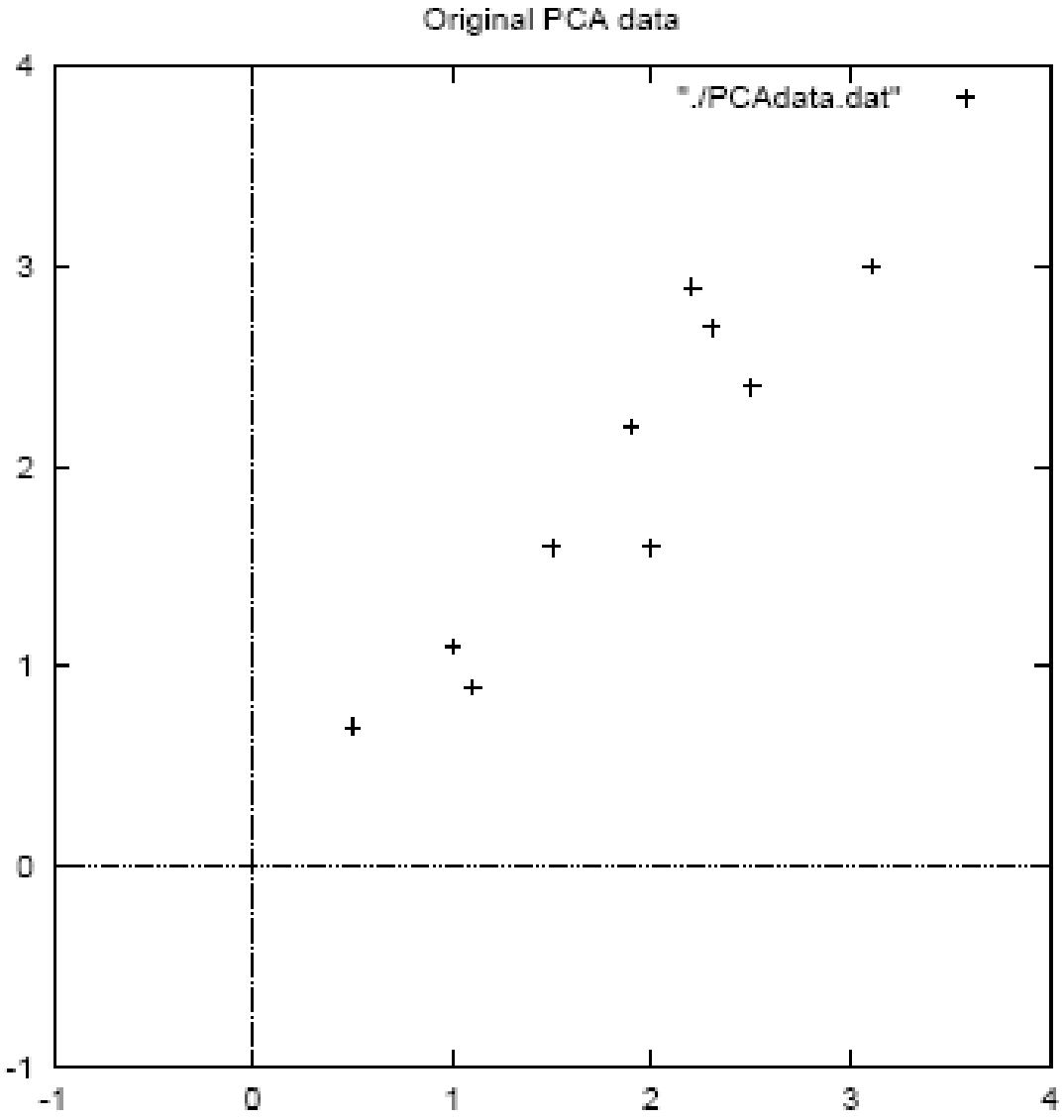
x	y
2.5	2.4
0.5	0.7
2.2	2.9
1.9	2.2
3.1	3.0
2.3	2.7
2	1.6
1	1.1
1.5	1.6
1.1	0.9

Mean: 1.81 1.91

DataAdjust =

x	y
.69	.49
-1.31	-1.21
.39	.99
.09	.29
1.29	1.09
.49	.79
.19	-.31
-.81	-.81
-.31	-.31
-.71	-1.01

Mean: 0 0


Original PCA data
"./PCAdata.dat"
4
3
2
1
0
-1
-1
0
1
2
3
4

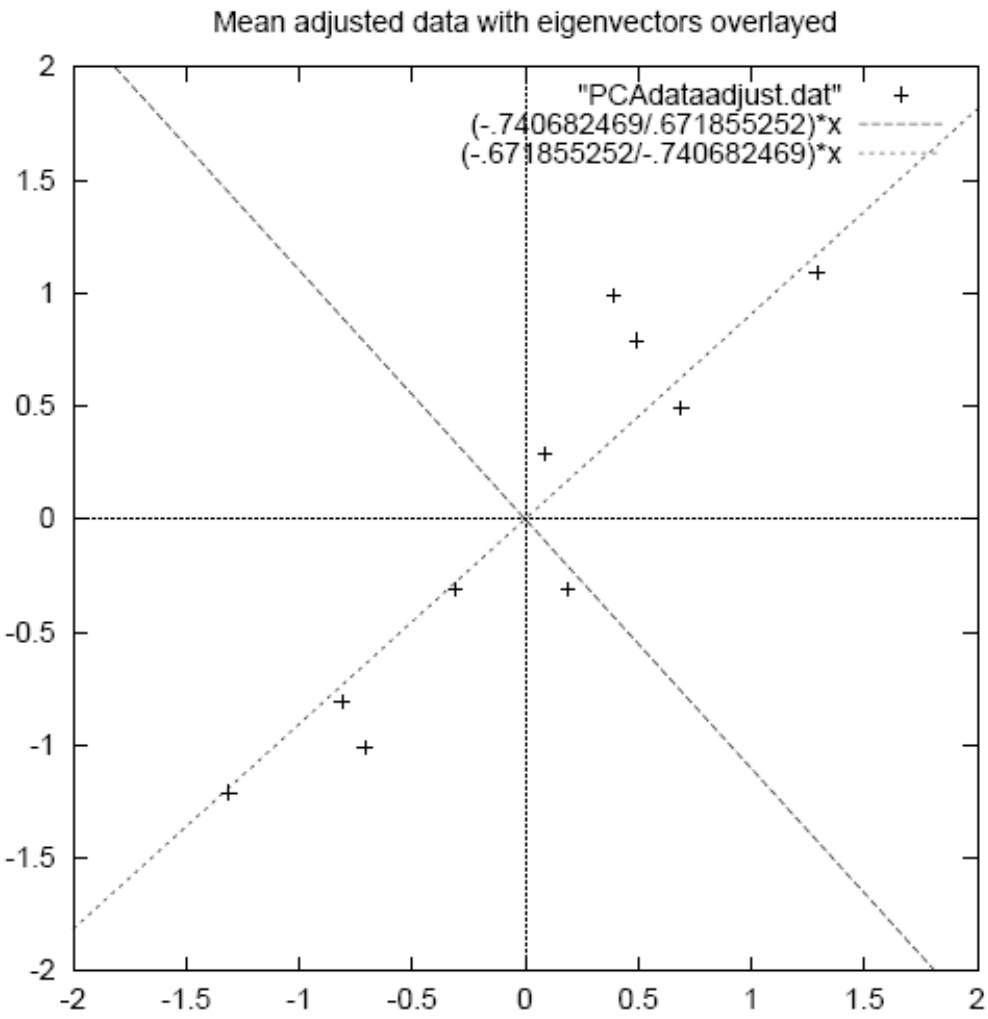
2. Calculate the covariance matrix:

$$cov = \begin{array}{c} \\ x \\ y \end{array} \begin{array}{c} \begin{array}{cc} x & y \end{array} \\ \begin{pmatrix} .616555556 & .615444444 \\ .615444444 & .716555556 \end{pmatrix} \end{array}$$

2. Calculate the (unit) eigenvectors and eigenvalues of the covariance matrix:

$$eigenvalues = \begin{pmatrix} .0490833989 \\ 1.28402771 \end{pmatrix}$$

$$eigenvectors = \begin{pmatrix} -.735178656 & -.677873399 \\ .677873399 & -.735178656 \end{pmatrix}$$



Eigenvector with largest eigenvalue traces linear pattern in data

Figure 3.2: A plot of the normalised data (mean subtracted) with the eigenvectors of the covariance matrix overlayed on top.

4. Order eigenvectors by eigenvalue, highest to lowest.

$$\mathbf{v}_1 = \begin{pmatrix} -.677873399 \\ -.735178956 \end{pmatrix} \quad \lambda = 1.28402771$$

$$\mathbf{v}_2 = \begin{pmatrix} -.735178956 \\ .677873399 \end{pmatrix} \quad \lambda = .0490833989$$

In general, you get n components. To reduce dimensionality to p, ignore $n-p$ components at the bottom of the list.

Construct new feature vector.

Feature vector = ($\mathbf{v}_1$, $\mathbf{v}_2$, ...$\mathbf{v}_p$)

$$FeatureVector1 = \begin{pmatrix} -.677873399 & -.735178956 \\ -.735178956 & .677873399 \end{pmatrix}$$

or reduced dimension feature vector:

$$FeatureVector2 = \begin{pmatrix} -.677873399 \\ -.735178956 \end{pmatrix}$$

Mining Association Rules

Association Rules

- Mining Association Rules between Sets of Items in Large Databases
  (R. Agrawal, T. Imielinski & A. Swami) 1993.

- Fast Algorithms for Mining Association Rules
  (R. Agrawal & R. Srikant) 1994.

Basket Data

Retail organizations, e.g., supermarkets, collect and store massive amounts sales data, called *basket data*.

A record consist of

- transaction date
- items bought

Or, basket data may consist of items bought by a customer over a period.

Example Association Rule

90% of transactions that purchase bread and butter also purchase milk

Antecedent: bread and butter

Consequent: milk

Confidence factor: 90%

Example Queries

- Find all rules that have “Diet Coke” in the antecedent.
- Find all rules that have “bread” in the antecedent and “egg” in the consequent.
- Find all the rules relating items located on shelves A and B in the store.

Formal Model

- $\mathcal{I} = i_1, i_2, \ldots, i_m$: set of literals (items)
- D : database of transactions
- $T \in D$: a transaction. $T \subseteq \mathcal{I}$
  - *TID: unique identifier, associated with each T*
- X: a subset of $\mathcal{I}$
  - T contains X if $X \subseteq T$.

Formal Model (Cont.)

- *Association rule*: $X \Rightarrow Y$
  here $X \subset I$, $Y \subset I$ and $X \cap Y = \emptyset$.
- Rule $X \Rightarrow Y$ has a confidence c in D
  if c% of transactions in D that contain X also contain Y.
- Rule $X \Rightarrow Y$ has a support s in D
  if s% of transactions in D contain $X \cup Y$.

Problem

- Given a set of transactions,
- Generate all association rules
- that have the support and confidence greater than the user-specified **minimum support** (*minsup*) and **minimum confidence** (*minconf*).

How Good is an Association Rule?

Customer	Items Purchased
1	OJ, soda
2	Milk, OJ, window cleaner
3	OJ, detergent
4	OJ, detergent, soda
5	Window cleaner, soda

← POS Transactions

Co-occurrence of Products

	OJ	Window cleaner	Milk	Soda	Detergent
OJ	4	1	1	2	2
Window cleaner	1	2	1	1	0
Milk	1	1	1	0	0
Soda	2	1	0	3	1
Detergent	2	0	0	1	2

How Good is an Association Rule?

	OJ	Window cleaner	Milk	Soda	Detergent
OJ	4	1	1	2	2
Window cleaner	1	2	1	1	0
Milk	1	1	1	0	0
Soda	2	1	0	3	1
Detergent	2	0	0	1	2

Simple patterns:

1. OJ and soda are more likely purchased together than any other two items
2. Detergent is never purchased with milk or window cleaner
3. Milk is never purchased with soda or detergent

How Good is an Association Rule?

Customer	Items Purchased
1	OJ, soda
2	Milk, OJ, window cleaner
3	OJ, detergent
4	OJ, detergent, soda
5	Window cleaner, soda

← POS Transactions

- What is the confidence for this rule:
  - If a customer purchases soda, then customer also purchases OJ
  - 2 out of 3 soda purchases also include OJ, so 67%
- What about the confidence of this rule reversed?
  - 2 out of 4 OJ purchases also include soda, so 50%
- **Confidence** = Ratio of the number of transactions with all the items to the number of transactions with just the "if" items

Association Rule Mining

- Given a set of transactions, find rules that will predict the occurrence of an item based on the occurrences of other items in the transaction

Market-Basket transactions

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

Example of Association Rules

{Diaper} → {oil},
{Milk, Bread} → {Eggs,Coke},
{oil, Bread} → {Milk},

Implication means co-occurrence, not causality!

Definition: Frequent Itemset

- **Itemset**
  - A collection of one or more items
    - Example: {Milk, Bread, Diaper}
  - k-itemset
    - An itemset that contains k items
- **Support count (σ)**
  - Frequency of occurrence of an itemset
  - E.g. σ({Milk, Bread,Diaper}) = 2
- **Support**
  - Fraction of transactions that contain an itemset
  - E.g. s({Milk, Bread, Diaper}) = 2/5
- **Frequent Itemset**
  - An itemset whose support is greater than or equal to a *minsup* threshold

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

Definition: Association Rule

- **Association Rule**
  - An implication expression of the form $X \rightarrow Y$, where X and Y are itemsets
  - Example:
    {Milk, Diaper} $\rightarrow$ {oil}

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

- **Rule Evaluation Metrics**
  - Support (s)
    - Fraction of transactions that contain both X and Y
  - Confidence (c)
    - Measures how often items in Y appear in transactions that contain X

Example:

$$\{\text{Milk}, \text{Diaper}\} \Rightarrow \text{Beer}$$

$$s = \frac{\sigma(\text{Milk}, \text{Diaper}, \text{Beer})}{|\text{T}|} = \frac{2}{5} = 0.4$$

$$c = \frac{\sigma(\text{Milk}, \text{Diaper}, \text{Beer})}{\sigma(\text{Milk}, \text{Diaper})} = \frac{2}{3} = 0.67$$

Mining Association Rules

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

Example of Rules:

{Milk,Diaper} → {oil} (s=0.4, c=0.67)
{Milk,oil} → {Diaper} (s=0.4, c=1.0)
{Diaper,oil} → {Milk} (s=0.4, c=0.67)
{Beer} → {Milk,Diaper} (s=0.4, c=0.67)
{Diaper} → {Milk,oil} (s=0.4, c=0.5)
{Milk} → {Diaper,oil} (s=0.4, c=0.5)

Observations:

- All the above rules are binary partitions of the same itemset:
  {Milk, Diaper, oil}
- Rules originating from the same itemset have identical support but can have different confidence
- Thus, we may decouple the support and confidence requirements

Mining Association Rules

- Two-step approach:
  1. Frequent Itemset Generation
     - Generate all itemsets whose support $\geq$ minsup
  2. Rule Generation
     - Generate high confidence rules from each frequent itemset, where each rule is a binary partitioning of a frequent itemset
- Frequent itemset generation is still computationally expensive

Power set

- Given a set S, power set, P is the set of all subsets of S
- Known property of power sets
  - If S has n number of elements, P will have $N = 2^n$ number of elements.
- Examples:
  - For S = {}, P={{}}, $N = 2^0 = 1$
  - For S = {Milk}, P={{}, {Milk}}, $N=2^1=2$
  - For S = {Milk, Diaper}
  - P={{},{Milk}, {Diaper}, {Milk, Diaper}}, $N=2^2=4$
  - For S = {Milk, Diaper, Beer},
  - P={{},{Milk}, {Diaper}, {Beer}, {Milk, Diaper}, {Diaper, Beer}, {Beer, Milk}, {Milk, Diaper, Beer}}, $N=2^3=8$

Brute Force approach to Frequent Itemset Generation

- For an itemset with 3 elements, we have 8 subsets
  - Each subset is a candidate frequent itemset which needs to be matched against each transaction

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

1-itemsets

Itemset	Count
{Milk}	4
{Diaper}	4
{oil}	3

2-itemsets

Itemset	Count
{Milk, Diaper}	3
{Diaper, oil}	3
{oil, Milk}	2

3-itemsets

Itemset	Count
{Milk, Diaper, oil}	2

Important Observation: Counts of subsets can't be smaller than the count of an itemset!

Reducing Number of Candidates

- Apriori principle:
  - If an itemset is frequent, then all of its subsets must also be frequent

- Apriori principle holds due to the following property of the support measure:

$$\forall X, Y : (X \subseteq Y) \Rightarrow s(X) \geq s(Y)$$

  - Support of an itemset never exceeds the support of its subsets
  - This is known as the anti-monotone property of support

Illustrating Apriori Principle

Item	Count
Bread	**4**
Coke	**2**
Milk	**4**
Oil	**3**
Diaper	**4**
Eggs	**1**

Items (1-itemsets)

Itemset	Count
{Bread,Milk}	**3**
{Bread,Oil}	**2**
{Bread,Diaper}	**3**
{Milk,Beer}	**2**
{Milk,Diaper}	**3**
{Oil,Diaper}	**3**

Pairs (2-itemsets)

(No need to generate candidates involving Coke or Eggs)

Triplets (3-itemsets)

Itemset	Count
{Bread,Milk,Diaper}	**3**

Write all possible 3-itemsets
and prune the list based on infrequent 2-itemsets

Apriori Algorithm

- Method:
  - Let k=1
  - Generate frequent itemsets of length 1
  - Repeat until no new frequent itemsets are identified
    - Generate length (k+1) candidate itemsets from length k frequent itemsets
    - Prune candidate itemsets containing subsets of length k that are infrequent
    - Count the support of each candidate by scanning the DB
    - Eliminate candidates that are infrequent, leaving only those that are frequent

Note: This algorithm makes several passes over the transaction list

Rule Generation

- Given a frequent itemset L, find all non-empty subsets $f \subset L$ such that $f \rightarrow L - f$ satisfies the minimum confidence requirement
  - If {A,B,C,D} is a frequent itemset, candidate rules:

    $ABC \rightarrow D$, $ABD \rightarrow C$, $ACD \rightarrow B$, $BCD \rightarrow A$,
    $A \rightarrow BCD$, $B \rightarrow ACD$, $C \rightarrow ABD$, $D \rightarrow ABC$
    $AB \rightarrow CD$, $AC \rightarrow BD$, $AD \rightarrow BC$, $BC \rightarrow AD$,
    $BD \rightarrow AC$, $CD \rightarrow AB$,

- If $|L| = k$, then there are $2^k - 2$ candidate association rules (ignoring $L \rightarrow \emptyset$ and $\emptyset \rightarrow L$)
- Because, rules are generated from frequent itemsets, they automatically satisfy the minimum support threshold
  - Rule generation should ensure production of rules that satisfy only the minimum confidence threshold

Computing Confidence for Rules

- Unlike computing support, computing confidence does not require several passes over the transaction list
- Supports computed from frequent itemset generation can be reused
- Tables on the side show support values for all (except null) the subsets of itemset {Bread, Milk, Diaper}
- Confidence values are shown for the $(2^3-2) = 6$ rules generated from this frequent itemset

Itemset	Support
{Milk}	4/5=0.8
{Diaper}	4/5=0.8
{Bread}	4/5=0.8

Itemset	Support
{Milk, Diaper}	3/5=0.6
{Diaper, Bread}	3/5=0.6
{Bread, Milk}	3/5=0.6

Itemset	Support
{Bread, Milk, Diaper}	3/5=0.6

Example of Rules:

{Bread, Milk} → {Diaper} (s=0.6, c=1.0)
{Milk, Diaper} → {Bread} (s=0.6, c=1.0)
{Diaper, Bread} → {Milk} (s=0.6, c=1.0)
{Bread} → {Milk, Diaper} (s=0.6, c=0.75)
{Diaper} → {Milk, Bread} (s=0.6, c=0.75)
{Milk} → {Diaper, Bread} (s=0.6, c=0.75)

Rule generation in Apriori Algorithm

- For each frequent k-itemset where k > 2
  - Generate high confidence rules with one item in the consequent
  - Using these rules, iteratively generate high confidence rules with more than one items in the consequent
  - if any rule has low confidence then all the other rules containing the consequents can be pruned (not generated)

{Bread, Milk, Diaper}

1-item rules

{Bread, Milk} $\rightarrow$ {Diaper}

{Milk, Diaper} $\rightarrow$ {Bread}

{Diaper, Bread} $\rightarrow$ {Milk}

2-item rules

{Bread} $\rightarrow$ {Milk, Diaper}
{Diaper} $\rightarrow$ {Milk, Bread}

{Milk} $\rightarrow$ {Diaper, Bread}