Mining Real World Data

Vikram Pudi
vikram@iiit.ac.in
IIIT Hyderabad

1

What is Preprocessing?

- Prepare input data for data mining
  - Clean: noise, inconsistency, missing values
  - Transform: discretize, normalize, generalize,...
  - Feature selection and feature engineering
- The kind of transformations to apply depend on our end goals, i.e. what are we trying to achieve by doing data mining. E.g. prediction, summarization, optimization, ...

2

Scenario 1

- Suppose you want to develop a media player that can provide recommendations to users on what items to play based on their taste.

3

Available data

- Song
- Artist
- Album
- Genre
- Rating
- User profile

This data may be split across many tables.

4

Strategies

1. Find similar songs and recommend them.
2. Find similar users and recommend their highly-rated songs.

- In strategy 1, our preprocessing requires to find features of songs and store them; e.g. artist, song length, frequency spectrum, tempo, etc.
- In strategy 2, our preprocessing requires to find features of users and store them; e.g. nationality, age, lenient/strict, etc.

5

Scenario 2

- Given sales transactions in a supermarket, suppose we want to determine the factors that lead to increase in sales of “Nirma washing powder”.

6

Available data

- Item-id
- Amount purchased
- Price
- Total
- User profile (more details in other tables)
- Date
- Type of item (more details in other tables)

Strategies

1. Determine what kind of users buy Nirma.
2. Determine what other items are purchased frequently with Nirma.
3. Determine what *kind* of items are purchased frequently with Nirma.
4. Determine on which kind of days people buy more Nirma.

What preprocessing is required?

Scenario 3

- Suppose you want to write an application that searches for faculty web-pages from all over the world, extracts information from these pages and determine which of these faculty do top-class research in the area of "data science".

What kind of data is available?

What preprocessing is needed?

Data Cleaning

Noise, inconsistency, missing vales

Handling noise (inaccuracies)

- Binning: Sort values of an attribute and partition into ranges/bins. Replace all values within a bin by its mean/median/...
- Regression: Fit the data into a function such as linear or non-linear regression.
- Outliers: Treat outliers as noise and ignore them.

Inconsistency

- Avoid by using good integrity constraints when designing database.
- If inconsistency still arises, cure using same approaches as for handling noise.

Missing Values

1. Ignore missing values
2. Most common value
3. Concept most common value
4. All possible values
5. Missing values as special values
6. Use classification techniques

Data Transformation

Format conversion, discretization, normalization, …

Data format conversion

- Needed usually when combining data from multiple sources.
- Needs major manual programming effort
- Remember Y2K problem!

Discretization (Numeric → Categorical)

- Sort values of numeric attribute
- Divide sorted values into ranges
  - Equi-depth
  - Clustering
  - Information gain

Normalization

- Some attributes may have large ranges.
- Bring all attributes to common range.
- Scale values to lie within, say, -1.0 to +1.0

Generalization

- Categorical attributes (like name, location) contain too many values.
- Attributes like name can be ignored.
- Attributes like location can be generalized (e.g. instead of using address, use only the city/town name).

Dimensionality Reduction

Feature selection
Feature engineering

Reducing Dimensions to Visualize

- Feature Selection
  Choose the "best" features from your data, which you then visualize.
- Feature Extraction
  Initial set of measured data and builds derived features intended to be informative and non-redundant, facilitating.

Feature selection

- Which features are likely to be relevant for a given task?
- E.g. for detecting spam emails, some words such as "free university degree", "easy loan", etc. may be more relevant than others.

Approach: Find discriminating features.

Selecting Features as Matrix Multiplication

$$X = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} \qquad Z = UX$$

- Select first and third feature

$$\begin{bmatrix} z_1 \\ z_2 \end{bmatrix} = \begin{bmatrix} x_1 \\ x_3 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}$$

- Select first and fourth feature

$$\begin{bmatrix} z_1 \\ z_2 \end{bmatrix} = \begin{bmatrix} x_1 \\ x_4 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}$$

Selecting Features as Matrix Multiplication

- A "new" set of features are selected/extracted from original one by a matrix multiplication.
- Rows of **U** decide what the new features are. (They need not be 0 and 1).
- Often rows of **U** is smaller than column of **U**. This is also called **dimensionality reduction**.
- We find dot product of existing features with row-vectors of **U**.

Feature Selection and Extraction

- Selection:
  Select some features out of a pool (Simple U with 0/1).
  Eg. Select best one by one.
- Extraction:
  Extract a set of new features (elements of U need not be 0/1).

Feature Selection and Extraction

Extraction is often required:

- To visualize in 2D/3D.
- To remove some “useless” or “less useful” features.
- Make computations efficient.
  (Note: original data could be 1000s of dimensions!!)

Principal Component Analysis

Simplifying representations

Three View Points

- Maximal variance on the new features.
- Data Compression and Minimal Reconstruction Error.
- Orthogonal Line Fitting.

Dimensionality and Representation

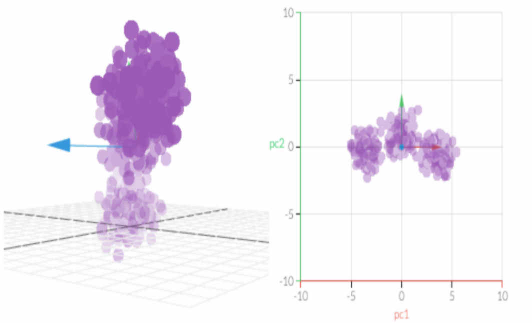
The Algorithm

- Find the prominent direction u (a vector)
- Project all samples x on this to get z
  – (dot product of x and u)

The Algorithm

- Find the prominent directions u1 and u2 (two vectors)
  – (Plane is defined by the two vectors/lines in 3D.)
- Project all samples x on these to get z1 and z2
  – (dot product of x and u1 and u2)

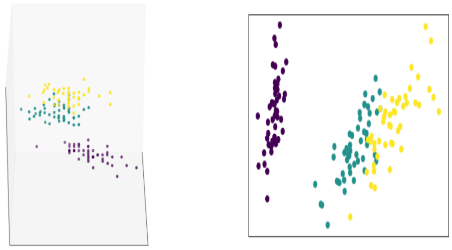
3D to 2D



X1, X2, X3

Z1, Z2

3D to 2D



PCA based Feature Extraction

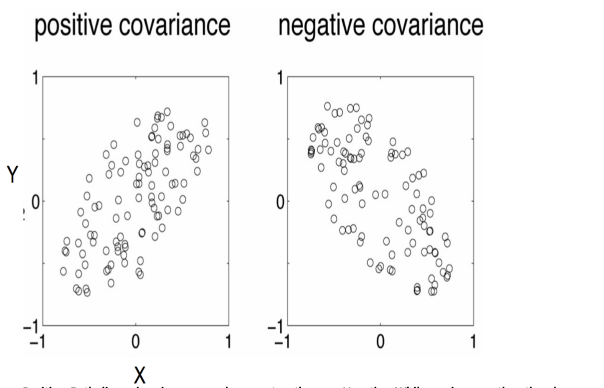
Covariance

- Variance and Covariance:
  - Measure of the “spread” of a set of points around their center of mass(mean)
- Variance:
  - Measure of the deviation from the mean for points in one dimension
- Covariance:
  - Measure of how much each of the dimensions vary from the mean with **respect to each other**

$$covariance(X,Y) = \frac{1}{n-1}\sum_{i=1}^{n}(X_i - \bar{X})(Y_i - \bar{Y})$$

$$\sum = \frac{1}{N}\sum_{i=1}^{N}(x_i - \mu)(x_i - \mu)^T$$

- **Covariance is measured between two dimensions**
- **Covariance sees if there is a relation between two dimensions**



Positive: Both dimensions increase or decrease together

Negative: While one increase the other decrease

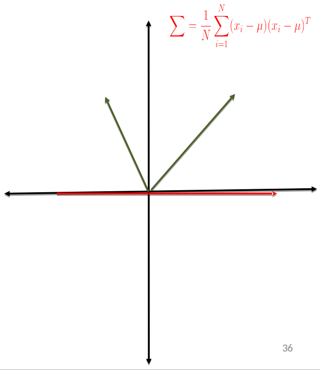
Principal Component Analysis

$$\mu = \frac{1}{N}\sum_{i=1}^{N} x_i$$

$$\sum = \frac{1}{N}\sum_{i=1}^{N}(x_i - \mu)(x_i - \mu)^T$$

Goal: Find r-dim projection that best preserves variance

- Compute the Mean Vector μ and Covariance Matrix $\sum$
- Compute Eigen vectors and Eigen values
- Select top r-Eigen Vectors
- Project the points on the subspace spanned by them



$$z = U[x - \mu]$$

Measurement of similarity

- Nominal (categorical) variables
  d(x,y) = 1 - m/n
  m = no of matches among n attributes, or
  m = sum of weights of matching attributes, and n is the sum of weights of all attributes
- Numeric variables
  - Euclidean, manhattan, minkowski,…
  - Ordinal
    - z = (rank-1)/(M-1) where M is maximum rank
- Above are examples
  - Similarity is ultimately application dependent
  - Requires various kinds of preprocessing
    - Scaling: Convert all attributes to have same range
    - z-score: z = (value-mean)/m
      where m is the mean absolute deviation

Real World Data

- Relational
- Transactional
- Multi-dimensional
- Distributed
- Stream synopses
  - Random samples, histograms, sliding windows, snapshots
  - Snapshot timeframes: e.g. calendar, logarithmic
- Time series
  - Moving weighted averages, Cycle detection, Regression

Real World Data (contd)

- Spatial
  - Data types: Points, Polylines, Regions (polygons)
  - Predicates:
    - Topological: adjacent, inside, disjoint
    - Direction: above, below, left_of,…
    - Metric: distance < 10km
- Multimedia
  - What are the features?
  - What is the similarity metric?
- Text & web

IR Concepts

- Information need vs Query
  - Boolean vs Ranked Queries
- Data model
  - Text: Loose structure
  - Web: More structure – tags, links
- Bag of words
- Vector-space
- Cosine similarity
  – d1 . d2 / |d1| |d2|
- Entropy
  – $H = - \Sigma\, p_i \log p_i$
- Stop words
  - Zipf law: Frequency of word inversely proportional to rank.
- Stemming
- TF-IDF
  – TF-IDF = [tf / max(tf)] x [log (N+1) / df]
- Inverted index
- Precision = P(found results are correct)
- Recall = P(correct results are found)
- F-score = 2RP/(R+P)

Search Engine Architecture

- Documents, Users, Queries
- Crawling
- Keyword Extraction (Normalized Tokens)
  - Stop-word
  - Stemming
  - Normalization (equivalence classes): U.S.A, C.A.T
- Indexing
- Ranking
  - Page-rank(x) = SUM [rank(y) / |links(y)|]
    - All y that link to x