Going Big in Data Dimensionality:
Challenges and Solutions for Mining High Dimensional Data

Peer Kröger
Lehrstuhl für Datenbanksysteme
Ludwig-Maximilians-Universität München
The three V’s of big data
- Volume
- Velocity
- Variety

Let’s talk about variety before talking about velocity and volume

An aspect of variety (and volume) is high dimensionality
- An archaelogical finding can have 10+ attributes/dimensions
- Recommendation data can feature 100+ dimensions per object
- Micro array data may contain 1000+ dimensions per object
- TF vectors may contain 10,000+ dimensions per object
- ...

Note: we are talking about structured data!!!
Outline

• Why Bother?

• Solutions

• Perspectives – Open Issues
Why Bother?

• Clustering
  – Automatically partition the data into clusters of similar objects
  – Diverse applications ranging from business applications (e.g. customer segmentation) to science (e.g. molecular biology)

• Similarity?
  – Objects are points in some feature spaces (let us assume an Euclidean space for convenience)
  – Similarity can be defined as the vicinity of two feature vectors in the feature space, usually applying a distance function like some Lp norm, the cosine distances, etc.
Why Bother?

• But:
  – In the early days of data mining:
    • The relevance of features was carefully analyzed before recording them because data acquisition was costly
    • So far so good: only relevant features were recorded
  
  – Nowadays:
    • Data acquisition is cheap and easy (mobile devices, sensors, modern machines, etc. – everyone measures everything, everywhere)
    • Consequence: sure big data, but what bothers?
      – The relevance of a given feature to the analysis task is not necessarily clear
      – Data is high dimensional containing a possibly large number of irrelevant attributes
    • OK, but why does this bother?
Why Bother?

- **High-dimensional data problems**
  - General: “the curse of dimensionality”
    \[
    \forall \varepsilon > 0 : \lim_{d \to \infty} P\left( \text{dist}\left( \frac{D_{\max_d} - D_{\min_d}}{D_{\min_d}}, 0 \right) \leq \varepsilon \right) = 1
    \]
  - Relative contrast of distances decrease
  - No cluster to find any more, only noise
    - Special/Additional: Clusters in subspaces
      - *Local* feature relevance
      - *Local* feature correlation

\[D_{\max} = \text{Distance to the farthest neighbor}\]
\[D_{\min} = \text{Distance to the nearest neighbor}\]
\[d = \text{dimensionality of the data space}\]
Why Bother?

- **Example: local feature correlation**
  - customers (C1 – C10) rate products (P1 – P3)

<table>
<thead>
<tr>
<th></th>
<th>P1</th>
<th>P2</th>
<th>P3</th>
</tr>
</thead>
<tbody>
<tr>
<td>C1</td>
<td>3</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>C2</td>
<td>2</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>C3</td>
<td>0</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>C4</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>C5</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>C6</td>
<td>1</td>
<td>3</td>
<td>10</td>
</tr>
<tr>
<td>C7</td>
<td>4</td>
<td>6</td>
<td>8</td>
</tr>
<tr>
<td>C8</td>
<td>0</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>C9</td>
<td>7</td>
<td>9</td>
<td>1</td>
</tr>
<tr>
<td>C10</td>
<td>3</td>
<td>5</td>
<td>5</td>
</tr>
</tbody>
</table>

relevant subspace (2D plane perpendicular to the line)

\[ P1 - 2 \cdot P2 + P3 = 0 \]

relevant subspace (1D line perpendicular to the plane)

\[ P1 - P2 + 2 = 0 \]
Why Bother?

- High-dimensional data problems

Consequences:

We should care about accuracy before caring about scalability

If we take a traditional clustering method and optimize it for efficiency, we will most likely not get the desired results

Dmax = Distance to the farthest neighbor
Dmin = Distance to the nearest neighbor

d = dimensionality of the data space

Specific/Additional: Clusters in subspaces

Relative contrast of distances decrease
Why Bother?

• General Solution: “correlation clustering”
  – Search for clusters in arbitrarily oriented subspaces
  – Affine subspace $S+a$, $S \subseteq \mathbb{R}^d$, affinity $a \in \mathbb{R}^d$, is interesting if a set of points clusters (are dense) within this subspace
  – The points of the cluster may exhibit high variance in the perpendicular subspace $(\mathbb{R}^d \setminus S)+a$

  $\rightarrow$ points form a hyper-plane along this subspace
Why Bother?

• Back to the definition of clustering
  – Clustering: partition the data into clusters of similar objects

  – Traditionally, similarity means similar values in all attributes
    (this obviously does not account for high dimensional data)

  – Now, similarity is defined as a common correlation in a given (sub)set of features (actually, that is not too far apart)
Why Bother?

- Why not feature selection?
  - (Unsupervised) feature selection (e.g. PCA, SVD, ...) is *global*; it always returns only one (reduced) feature space
  - The *local* feature relevance/correlation problem states that we usually need multiple feature spaces (possibly one for each cluster)
  - Example: Simplified metabolic screening data (here: 2D, 43D in reality)
Why Bother?

- Use feature selection before clustering

Symposium on Scalable Analytics, 22. – 23. November 2012, Garching bei München
Why Bother?

• Two tasks:

  1. We still need to search for clusters (depends on cluster model)
     • E.g. minimal cut of the similarity graph is NP-complete

  2. But now, we also need to search for arbitrarily oriented subspaces
     (search space probably infinite)
     • Naïve solution:
       – Given a cluster criterion and a database of \( n \) points
       – Compute for each subset of \( k \) points the subspace in which these points
         cluster and test the cluster criterion in this subspace
       – Search space:
         \[
         \sum_{k=1}^{n} \binom{n}{k} = 2^n - 1 = O(2^n)
         \]

• BTW:
  – How can we compute the subspace of the cluster? => see later
  – What is a cluster criterion? => see task 1
Why Bother?

• Even worse: *Circular Dependency*
  – Both tasks depend on each other
    • In order to determine the correct subspace of a cluster, we need to know (at least some) cluster members
    • In order to determine the correct cluster memberships, we need to know the subspaces of all clusters

• How to solve the circular dependency problem?
  – Integrate subspace search into the clustering process
  – Due to the complexity of both tasks, we need **heuristics**
  – These heuristics should *simultaneously* solve
    • the clustering problem
    • the subspace search problem
Outline

• Why Bother?

• Solutions

• Perspectives – Open Issues
• Finding clusters in arbitrarily oriented subspaces
  - Given a set $D$ of points (e.g. a potential cluster); how can we determine the subspace in which these points cluster?
  - Principal Component Analysis (PCA) determines the directions of highest variance
    - Compute Covariance-matrix $\Sigma_D$ für $D$

\[
\Sigma_D = \frac{1}{|D|} \sum_{x \in D} (x - x_D)(x - x_D)^T
\]

• Obtain Eigenvalue-Matrix and Eigenvector-Matrix
  - $V_D$: new basis, first Eigenvector = direction of the highest variance
  - $E_D$: covariance-matrix of $D$ in the new coordinate system $V_D$
• If the points in $D$ form a $\lambda$-dimensional hyper-plane then this hyper-plane is spanned by the first $\lambda$ Eigenvectors.

• The relevant subspace in which the points cluster is spanned by the remaining $d-\lambda$ Eigenvectors $\hat{V}_D$.

The sum of the smallest $d-\lambda$ Eigenvalues

$$\sum_{i=\lambda+1}^{d} e_{D_i}$$

is minimal w.r.t. all possible transformations $\rightarrow$ points cluster optimal in this subspace.

• Model for Correlation Cluster

  – The $\lambda$-dimensional hyper-plane accommodating the cluster $C \subseteq R^d$ is defined by a system of $d-\lambda$ equations for $d$ variables and an affinity (e.g. the centroid of the cluster $x_C$):

$$\hat{V}_C^T x = \hat{V}_C^T x_C$$

  – The equation system is approximately fulfilled by all $x \in C$.
• Correlation clustering methods based on PCA
  – Integrate (local) PCA into existing clustering algorithms
  – Learn a distance measure that reflects the subspace of points and/or parts of clusters (typically: specialized Mahalanobis distance)
  – Conquer the circular dependency of the two tasks by the so-called „Locality Assumption“
    • A local selection of points (e.g. the \( k \)-nearest neighbors of a potential cluster center) represents the hyper-plane of the corresponding cluster
    • The application of PCA on this local selection yields the subspace of the corresponding cluster
  
• *Curse of dimensionality*???
• Many methods rely on a local application of PCA to sets of potential cluster members
  – Rely on locality assumption
  – Alternative: random sampling

• How can we avoid the Locality Assumption/Random Sampling???
  – CASH (Clustering in Arbitrary Subspaces based on the Hough transform)
• Basic idea of CASH
  – Transform each object into a so-called *parameter space* representing all possible subspaces accommodating this object (i.e. all hyper-planes through this object)
  – This parameter space is a *continuum* of all these subspaces
  – The subspaces are represented by a considerably small number of parameters
  – This transform is a generalization of the Hough Transform (which is designed to detect linear structures in 2D images) for arbitrary dimensions
Transform

For each $d$-dimensional point $p$ there is an infinite number of $(d-1)$-dimensional hyper-planes through $p$.

Each of these hyper-planes $s$ is defined by $(p, \alpha_1, \ldots, \alpha_{d-1})$, where $\alpha_1, \ldots, \alpha_{d-1}$ is the normal vector $n_s$ of the hyper-plane $s$.

The function $f_p(\alpha_1, \ldots, \alpha_{d-1}) = \delta_s = \langle p, n_s \rangle$ maps $p$ and $\alpha_1, \ldots, \alpha_{d-1}$ onto the distance $\delta_s$ of the hyper-plane $s$ to the origin.

The parameter space plots the graph of this function.
Properties of this transform

- point in the data space = sinusoide curve in the parameter space
- point in the parameter space = hyper-plane in the data space
- points on a common hyper-plane in the data space (cluster) = sinusoide curves intersecting at one point in the parameter space
- intersection of sinusoide curves in the parameter space = hyper-plane accommodating the corresponding points in data space
• Detecting clusters
  – determine all intersection points of at least \( m \) curves in the parameter space
  => (d-1)-dimensional cluster
  – Exact solution (check all pair-wise intersections) is too costly
  – Heuristics are employed
    • Grid-based bisecting search
      => Find cells with at least \( m \) curves

😊 determining the curves that are within a given cell is in \( O(d^3) \)
😊 Number of cells \( O(r^d) \), where \( r \) is the resolution of the grid
😊 high value for \( r \) necessary
• **Complexity** \( (c = \text{number of cluster found} – \text{not an input parameter!!!}) \)
  - Bisecting search \( O(s \cdot c) \)
  - Determination of curves in a cell \( O(n \cdot d^3) \)
  - Over all \( O(s \cdot c \cdot n \cdot d^3) \)
  (algorithms for PCA are also in \( O(d^3) \))

• **Robustness against noise**
Outline

• Why Bother?

• Solutions

• Perspectives – Open Issues
• What is next?
  – Still a lot to take care about at the accuracy end!!!
    • Examining the results (Are they significant? How to evaluate?).
    • Novel heuristics with new assumptions (limitations?).
    • Other patterns like outlier detection
    • …

• Big Data?
  – **Variety:**
    non-linear correlations, non-numeric data, relational data, …
  – **Velocity:**
    dynamic data, data streams, …
  – **Volume:**
    scalability (size/dimensionality), approximate solutions, …
Schluss