Lecture 8

Scalable PCA/SVD

Dimensionality Reduction & Factor Analysis

Haiping

http://www.dcs.shef.

COM6012 Scalable Machine Learning

Spring 2018

Week 8 Contents

- **Unsupervised Learning**

- PCA - Dimen

- SVD – Factor Analysis

- Scalable PCA in Spark

Unsupervised Learning

Supervised methods

Unsupervised methods

$y = f($

$f(X)$

predict
our data

as a function of
other data

find structure in the data on its own

Three Topics

- Principal component analysis (PCA) & SVD
  - Dimensionality reduction & factor analysis

- K-means
  - Clustering

- Matrix factorisation (with ormation)

  - Collaborative filtering →Re system

- **Scale these algorithms for big data**

Week 8 Contents

- Unsupervised Learning

- **PCA - Dime**

- SVD – Factor Analysis

- Scalable PCA in Spark

Dimensionality Reduction

- **Assumption:** Data lies on or near a low *d*-dimensional subspace
- Axes of this subspace are effective representation of the data

Why Reduce Dimensions?

Why reduce dimensions?

- Discover hidden correlations/topics

  - Words that o
- Remove redun s

  - Not all words are useful

- Interpretation and visualization
- Easier storage and processing of the data

Dimensionality Reduction

- Raw data is complex and high-dimensional

- Dimensionali the data using a simpler, mor on

- This representation may make interesting patterns in the data clearer or easier to see

Dimensionality Reduction

- Goal: Find a ‘better’ representation for data

- How do we d

- For example
  - Minimise reconstruction err

  - Maximise variance
  - **They give the same solution → PCA!**

PCA Algorithm

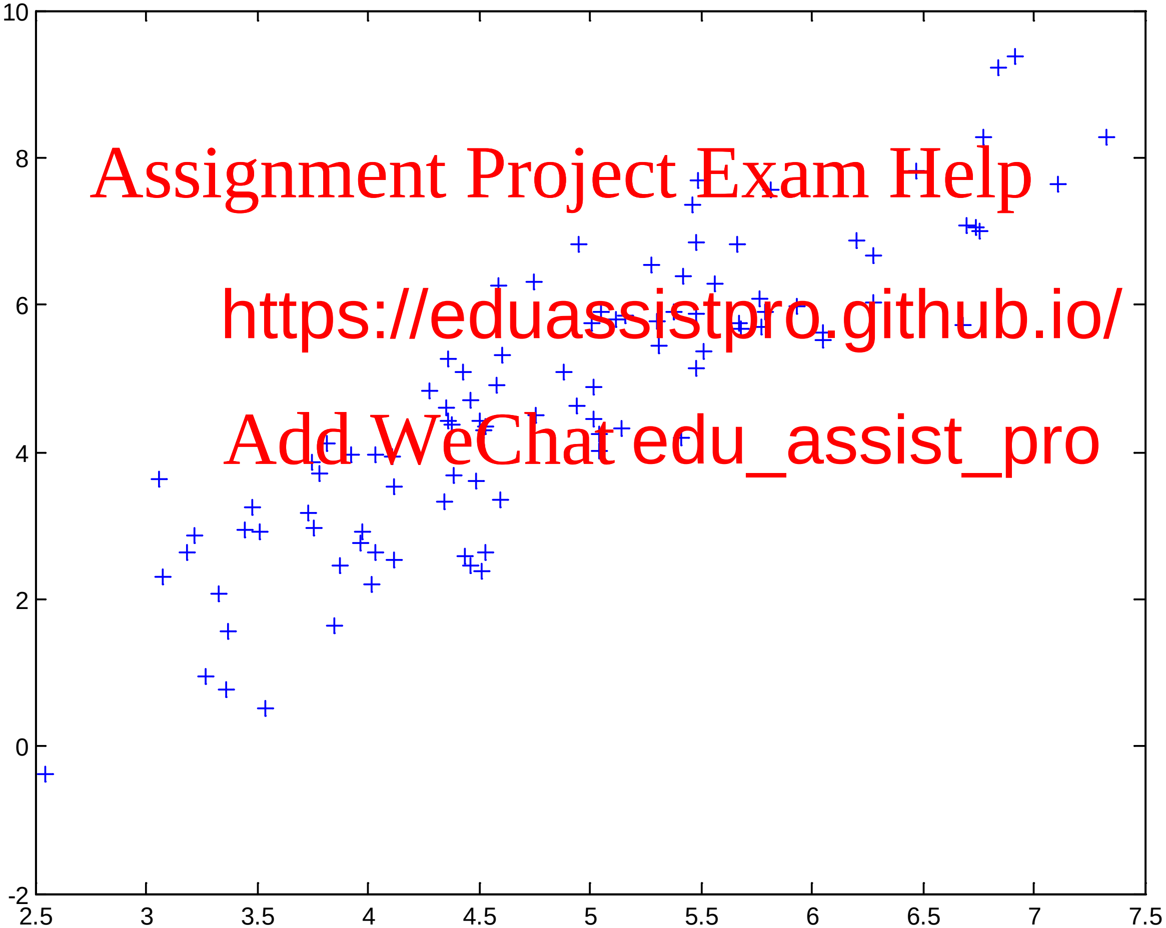
- Input: N data points, each → D-dimensional vector
- PCA algorithm

  - 1. $\mathbf{X_0}$ ← For h one row vector $\mathbf{x}_n$ per data poi

  - 2. $\mathbf{X}$**:** subtract mean $\bar{\mathbf{x}}$ from tor $\mathbf{x}_n$ in $\mathbf{X_0}$

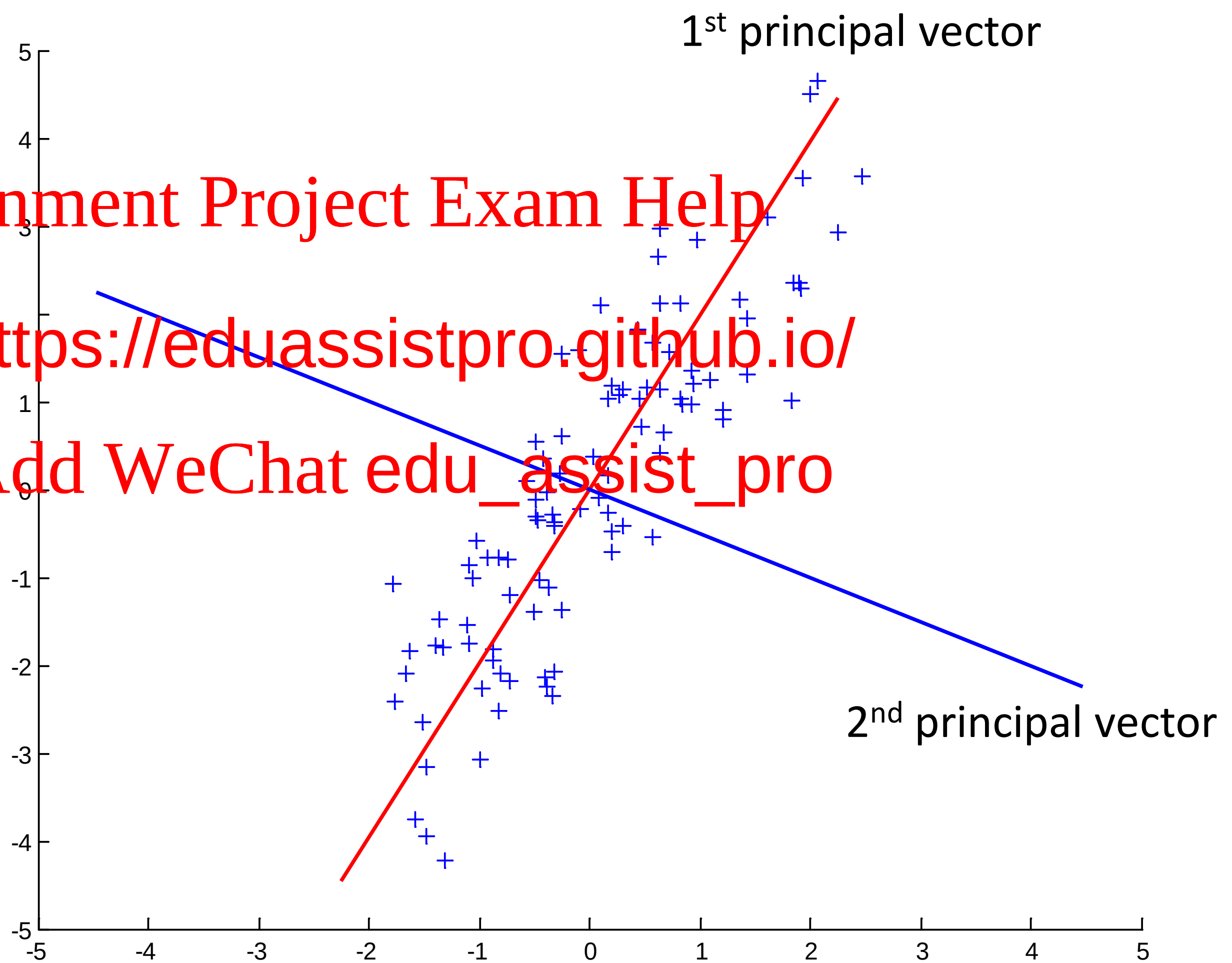
  - 3. $\mathbf{\Sigma} \leftarrow \mathbf{X}^T\mathbf{X}$ Gramian (scatt r $\mathbf{X}$
  - Find eigenvectors and eigenvalues of $\mathbf{\Sigma}$
  - PCs $\mathbf{U}$ ($D \times d$) ← the d eigenvectors with largest eigenvalues
- PCA feature for $\mathbf{y}$ D-dim: $\mathbf{U}^T\mathbf{y}$ (d-dimensional)
  - Zero correlations, ordered by variance

2D Data



Principal Components

- The best axis to project
- Minimum RMS error
- Principal vectors are orthogonal



How Many Components?

- Check the distribution of eigen-values
- Take enough many eigen-vectors to cover 80-90% of the variance

Other Practical Tips

- PCA assumptions (linearity, orthogonality) not always appropriate
- Various extensions to PCA with different underlying assumptions, , Kernel PCA, ICA
- Centring is crucial, i.e., we rocess data so that all features have zero re applying PCA
- PCA results dependent on scaling of data
- Data is sometimes rescaled in practice before applying PCA

Problems and Limitations

- What if very large dimensional data?
  - e.g., Images ($D \geq 10^4 = 100\text{x}100$)

- Problem:
  - Gramian mat
  - $D=10^4 \rightarrow |\Sigma| = 10^8$

- Singular Value Decomposition (SVD)!
  - Efficient algorithms available
  - Some implementations find just top *d* eigenvectors

Week 8 Contents

- Unsupervised Learning

- PCA - Dimen

- **SVD – Factor Analysis**
- Scalable PCA in Spark

Singular Value Decomposition

- Factorization (decomposition) problem
  - #1: Find concepts/topics/genres → Factor Analysis
  - #2: Reduce dimensionality

The above matrix is actually "2-dimensional." All rows can be reconstructed by scaling [1 1 1 0 0] or [0 0 0 1 1]: D=5→d=2

SVD - Definition

$$\mathbf{A}_{[\mathbf{n} \times \mathbf{m}]} = \mathbf{U}_{[\mathbf{n} \times \mathbf{r}]} \, \boldsymbol{\Lambda}_{[\mathbf{r} \times \mathbf{r}]} \, (\mathbf{V}_{[\mathbf{m} \times \mathbf{r}]})^{\mathrm{T}}$$

- **A**: $n \times m$ ma , m terms)
- **U**: $n \times r$ matrix (n docume epts)
- **Λ**: $r \times r$ diagonal matrix (s each ‘concept’) (r: rank of the matrix)
- **V**: $m \times r$ matrix (m terms, r concepts)

SVD - Properties

Always possible to decompose matrix $\mathbf{A}$ into $\mathbf{A} = \mathbf{U}\,\boldsymbol{\Lambda}\,\mathbf{V}^T$, where

- **U, Λ, V**: uniqu

- **U**, **V**: column o are unit vectors, orthogonal to each other)

  - $\mathbf{U}^T\mathbf{U} = \mathbf{I}$; $\mathbf{V}^T\mathbf{V} = \mathbf{I}$ (**I:** identity ma
- **Λ:** singular value are positive, and sorted in decreasing order

SVD ←→ Eigen-decomposition

- SVD gives us:
  - $\mathbf{A} = \mathbf{U}\,\Lambda\,\mathbf{V}^T$
- Eigen-decomposition:
  - $\mathbf{B} = \mathbf{W}\,\Sigma\,\mathbf{W}^T$
    - $\mathbf{U}$, $\mathbf{V}$, $\mathbf{W}$ are
    - Λ, Σ are diagonal
- Relationship:
  - $\mathbf{AA}^T = \mathbf{U}\,\Lambda\,\mathbf{V}^T(\mathbf{U}\,\Lambda\,\mathbf{V}^T)^T = \mathbf{U}\,\Lambda\,\mathbf{V}^T(\mathbf{V}\,\Lambda^T\mathbf{U}^T) = \mathbf{U}\,\Lambda\,\Lambda^T\,\mathbf{U}^T$
  - $\mathbf{A}^T\mathbf{A} = \mathbf{V}\,\Lambda^T\,\mathbf{U}^T(\mathbf{U}\,\Lambda\,\mathbf{V}^T) = \mathbf{V}\,\Lambda\,\Lambda^T\,\mathbf{V}^T = \mathbf{V}\,\Lambda^2\,\mathbf{V}^T$
  - $\mathbf{B} = \mathbf{A}^T\mathbf{A} = \mathbf{W}\,\Sigma\,\mathbf{W}^T$

SVD for PCA

- PCA by SVD:
  - 1. $\mathbf{X}_0 \leftarrow$ Form $N \times d$ data matrix, with one row vector $\mathbf{x}_n$ per data point
  - 2. $\mathbf{X}$ subtra ector $\mathbf{x}_n$ in $\mathbf{X}_0$
  - 3. $\mathbf{U}\ \mathbf{\Lambda}\ \mathbf{V}^T$
  - The right singular vectors $\mathbf{V}$ uivalent to the eigenvectors of $\mathbf{X}^T\mathbf{X} \rightarrow$ the
  - The singular values in $\mathbf{\Lambda}$ are equal to the square roots of the eigenvalues of $\mathbf{X}^T\mathbf{X}$

SVD - Properties

'spectral decomposition' of the matrix:

$$\begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 2 & 2 & 2 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 0 & 0 & 2 & 2 \\ 0 & 0 & 0 & 3 & 3 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix} = \begin{bmatrix} | & | \\ u_1 & u_2 \\ | & | \end{bmatrix} \times \begin{bmatrix} \sigma_1 & \emptyset \\ \emptyset & \sigma_2 \end{bmatrix} \times \begin{bmatrix} \text{---} & v_1 & \text{---} \\ \text{---} & v_2 & \text{---} \end{bmatrix}$$

SVD - Interpretation

‘documents’, ‘terms’ and ‘concepts’:

- **U**: document-to-concept similarity matrix

- **V**: term-to-c x
- **Λ**: its diagon of each concept

Projection:

- Best axis to project on: (‘best’ = min sum of squares of projection errors)

SVD - Example

- $\mathbf{A} = \mathbf{U}\, \boldsymbol{\Lambda}\, \mathbf{V}^T$ - example:

$$\begin{array}{c} \\ \text{CS} \\ \\ \text{MD} \end{array} \begin{array}{c} \text{data} \quad \text{inf.} \quad \text{retrieval} \quad \text{brain} \\ \begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 2 & 2 & 2 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 0 & 0 & 2 & 2 \\ 0 & 0 & 0 & 3 & 3 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix} \end{array} = \begin{bmatrix} 0.18 & 0 \\ 0.36 & 0 \\ 0.18 & 0 \\ 0.90 & 0 \\ 0 & 0.53 \\ 0 & 0.80 \\ 0 & 0.27 \end{bmatrix} \mathrm{x} \begin{bmatrix} 9.64 & 0 \\ 0 & 5.29 \end{bmatrix} \mathrm{x} \begin{bmatrix} 0.58 & 0.58 & 0.58 & 0 & 0 \\ 0 & 0 & 0 & 0.71 & 0.71 \end{bmatrix}$$

SVD - Example

- $\mathbf{A} = \mathbf{U}\, \mathbf{\Lambda}\, \mathbf{V}^T$ - example:

doc-to-concept similarity matrix

CS-concept

ept

data inf. retrieval brain l

$$
\begin{array}{c}
\text{CS} \\ \\ \\ \text{MD} \\ \\
\end{array}
\begin{bmatrix}
1 & 1 & 1 & 0 & 0 \\
2 & 2 & 2 & 0 & 0 \\
1 & 1 & 1 & 0 & 0 \\
5 & 5 & 5 & 0 & 0 \\
0 & 0 & 0 & 2 & 2 \\
0 & 0 & 0 & 3 & 3 \\
0 & 0 & 0 & 1 & 1
\end{bmatrix}
=
\begin{bmatrix}
0.18 & 0 \\
0.36 & 0 \\
0.18 & 0 \\
0.90 & 0 \\
0 & 0.53 \\
0 & 0.80 \\
0 & 0.27
\end{bmatrix}
\times
\begin{bmatrix}
9.64 & 0 \\
0 & 5.29
\end{bmatrix}
\times
\begin{bmatrix}
0.58 & 0.58 & 0.58 & 0 & 0 \\
0 & 0 & 0 & 0.71 & 0.71
\end{bmatrix}
$$

SVD - Example

- $\mathbf{A} = \mathbf{U}\,\boldsymbol{\Lambda}\,\mathbf{V}^T$ - example:

-concept

$$\begin{array}{cc} & \begin{array}{ccccc} \text{data} & \text{inf.} & \text{retrieval} & \text{brain} & \text{lu} \end{array} \\ \begin{array}{c} \\ \text{CS} \\ \\ \\ \\ \text{MD} \\ \\ \end{array} & \begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 2 & 2 & 2 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 0 & 0 & 2 & 2 \\ 0 & 0 & 0 & 3 & 3 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix} \end{array} = \begin{bmatrix} 0.18 & 0 \\ 0.36 & 0 \\ 0.18 & 0 \\ 0.90 & 0 \\ 0 & 0.53 \\ 0 & 0.80 \\ 0 & 0.27 \end{bmatrix} \times \begin{bmatrix} 9.64 & 0 \\ 0 & 5.29 \end{bmatrix} \times \begin{bmatrix} 0.58 & 0.58 & 0.58 & 0 & 0 \\ 0 & 0 & 0 & 0.71 & 0.71 \end{bmatrix}$$

SVD - Example

- $\mathbf{A} = \mathbf{U}\,\mathbf{\Lambda}\,\mathbf{V}^T$ - example:

term-to-concept
milarity matrix

$$
\begin{array}{c c}
 & \begin{array}{ccccc} \text{data} & \text{inf.} & \text{retrieval} & \text{brain} & \text{lu} \end{array} \\
\begin{array}{c} \text{CS} \\ \\ \\ \\ \text{MD} \\ \\ \\ \end{array} &
\begin{bmatrix}
1 & 1 & 1 & 0 & 0 \\
2 & 2 & 2 & 0 & 0 \\
1 & 1 & 1 & 0 & 0 \\
5 & 5 & 5 & 0 & 0 \\
0 & 0 & 0 & 2 & 2 \\
0 & 0 & 0 & 3 & 3 \\
0 & 0 & 0 & 1 & 1
\end{bmatrix}
\end{array}
=
\begin{bmatrix}
0.18 & 0 \\
0.36 & 0 \\
0.18 & 0 \\
0.90 & 0 \\
0 & 0.53 \\
0 & 0.80 \\
0 & 0.27
\end{bmatrix}
\times
\begin{bmatrix}
9.64 & 0 \\
0 & 5.29
\end{bmatrix}
\times
\begin{bmatrix}
0.58 & 0.58 & 0.58 & 0 & 0 \\
0 & 0 & 0 & 0.71 & 0.71
\end{bmatrix}
$$

SVD – Dimensionality Reduction

- Q: how exactly is (**further**) dim. reduction done?
- A: set the smallest singular values to zero:
- Note: **3 zero** ... y removed

$$\begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 2 & 2 & 2 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 0 & 0 & 2 & 2 \\ 0 & 0 & 0 & 3 & 3 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix} = \begin{bmatrix} 0.18 & \cancel{0} \\ 0.36 & \cancel{0} \\ 0.18 & \cancel{0} \\ 0.90 & \cancel{0} \\ 0 & \cancel{0.53} \\ 0 & \cancel{0.80} \\ 0 & \cancel{0.27} \end{bmatrix} \times \begin{bmatrix} 9.64 & 0 \\ 0 & \cancel{5.29} \end{bmatrix} \times \begin{bmatrix} 0.58 & 0.58 & 0.58 & 0 & 0 \\ \cancel{0} & \cancel{0} & \cancel{0} & \cancel{0.71} & \cancel{0.71} \end{bmatrix}$$

SVD - Dimensionality Reduction

$$\begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 2 & 2 & 2 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 0 & 0 & 2 & 2 \\ 0 & 0 & 0 & 3 & 3 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix} \sim \begin{bmatrix} 0.36 \\ 0.18 \\ 0.90 \\ 0 \\ 0 \\ 0 \end{bmatrix} \text{ x } \begin{bmatrix} \end{bmatrix} \text{ x } \begin{bmatrix} 0.58 & 0.58 & 0.58 & 0 & 0 \end{bmatrix}$$

SVD - Dimensionality Reduction

- Best rank-1 approximation

$$\begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 2 & 2 & 2 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 0 & 0 & 2 & 2 \\ 0 & 0 & 0 & 3 & 3 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix} \sim \begin{bmatrix} 2 & 2 & 2 & & \\ 1 & 1 & 1 & & \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Week 8 Contents

- Unsupervised Learning

- PCA - Dimen

- SVD – Factor Analysis

- **Scalable PCA in Spark**

PCA & SVD in Spark MLlib

- Not scalable: computePrincipalComponents( ) from RowMatrix
- **Scalable**: computeSVD( ) from RowMatrix

- Code:

https://github.com/a src/main/scala/org/apache/spark/mllib/linalg/distributed/Row

- Documentation: https://spark.apache.org/docs/2.1.0/api/scala/index.html#org.apache.spark.mllib.linalg.distributed.RowMatrix

PCA in Spark MLlib (RDD)

- https://spark.apache.org/docs/2.1.0/mllib-dimensionality-reduction.html

- Not scalable, local computation

```
val brzSvd.SVD(u: BDM[Double], s: BDV[Double], _) = brzSvd(Cov)
```

- Notebook 8

PCA in Spark ML (DF)

- Now in https://spark.apache.org/docs/2.1.0/ml-features.html#pca
- Under feature
- Scalable? Not

SVD in Spark MLlib (RDD)

- https://spark.apache.org/docs/2.1.0/mllib-dimensionality-reduction.html
- With distribu

SVD in Spark MLlib (RDD)

- An $m \times n$ data matrix **A** with $m > n$ (note different notations)

- For large mat t need the complete fact top k singular values and its ctors.

- Save storage, de-noise and e low-rank structure of the matrix (dimensionality reduction)

SVD in Spark MLlib (RDD)

- An $m \times n$ data matrix **A**
- Assume $m > n$, SVD $\mathbf{A} = \mathbf{U}\,\mathbf{\Lambda}\,\mathbf{V}^{T}$

- The singular gular vectors are derived fr d the eigenvectors of $\mathbf{A}^{T}\mathbf{A}$ (whic r than **A**)

- The left singular vectors ar d via matrix multiplication as $\mathbf{U}=\mathbf{A}\mathbf{V}\,\mathbf{\Lambda}^{-1}$, if requested by the user via the computeU parameter

Selection of SVD Computation

- Auto
- If *n* is small (*n*<100) or *k* is large compared with *n* (*k*>*n/2*), com T compute its top eigenvalues a y on the driver
- Otherwise, compute $\mathbf{A}^T\mathbf{A}$ v utive way and send it to ARPACK to co p eigenvalues and eigenvectors on the driver node

Selection of SVD Computation

- Auto (default)

Selection of SVD Computation

- Specify computeMode (private)

Selection of SVD Computation

- computeMode (note brzSvd.SVD is local)

Remark

- Acknowledgement
  - Some slides are adapted from slides by Jure Leskovec et al. http://www.mmds.org

- References
  - http://infolab.stanford.edu/~ s/ch11.pdf
  - http://www.mmds.org
  - https://en.wikipedia.org/wiki/Principal_component_analysis
  - https://spark.apache.org/docs/2.1.0/mllib-dimensionality-reduction.html