Real World Applications of Data Science

In partnership with:
Proscia Inc, Betamore, Spark B-more

Lecture 4: Topics in Unsupervised Learning Pt. 2

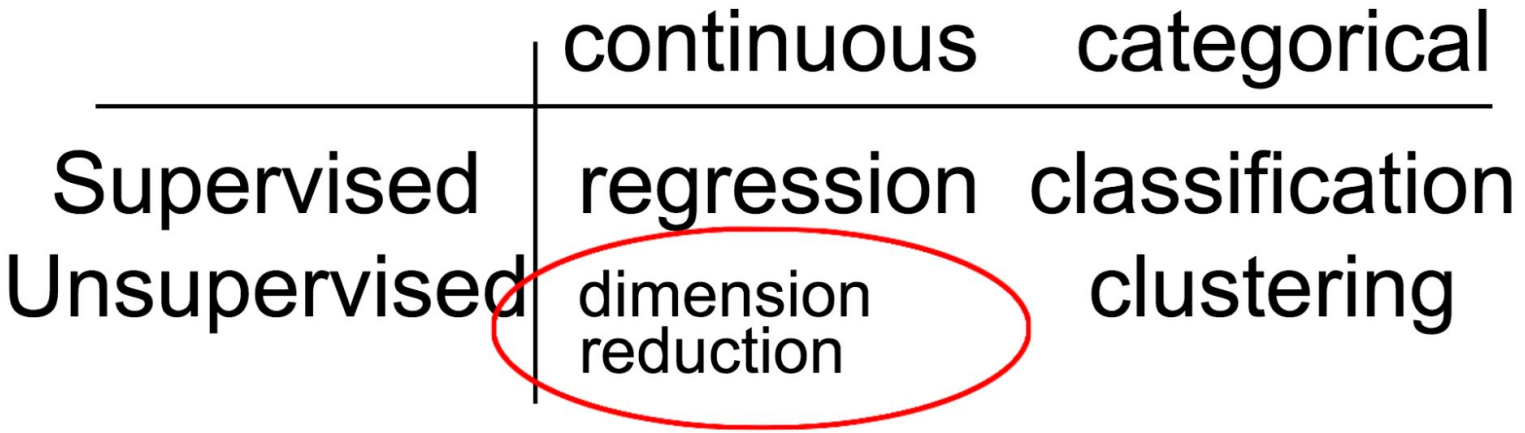
Topics in Unsupervised Learning

1) Clustering + Visualization
2) **Principal Component Analysis**
3) **Dimensionality Reduction**

Dimensionality Reduction

Topics

Topics


continuous
categorical
Supervised
regression
classification
Unsupervised
dimension reduction
clustering

Dimensionality reduction

Q: What is dimensionality reduction?

Dimensionality reduction

Q: What is dimensionality reduction?
A: A set of techniques for reducing the size (in terms of features,
records, and/or bytes) of the dataset under examination.

In general, the idea is to regard the dataset as a matrix and to decompose the matrix into simpler, meaningful pieces.

Dimensionality reduction

Q: What are the motivations for dimensionality reduction?

Dimensionality reduction

Q: What are the motivations for dimensionality reduction?

The number of features in our dataset can be difficult to manage, or even misleading (eg, if the relationships are actually simpler than they appear).

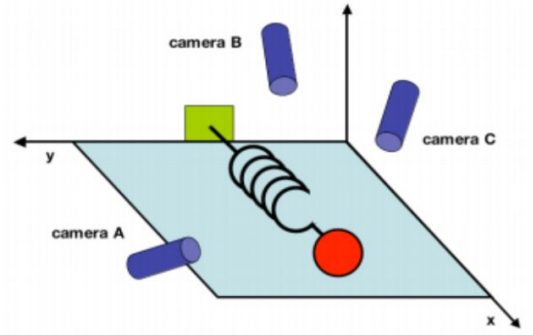
Dimensionality reduction

For example, suppose we have a dataset with some features that are related to each other.

Ideally, we would like to eliminate this redundancy and consolidate the number of variables we're looking at.

If these relationships are *linear*, then we can use well-established techniques like PCA/SVD.

Example: 1d harmonic oscillator



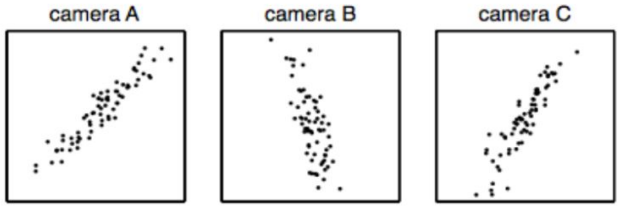
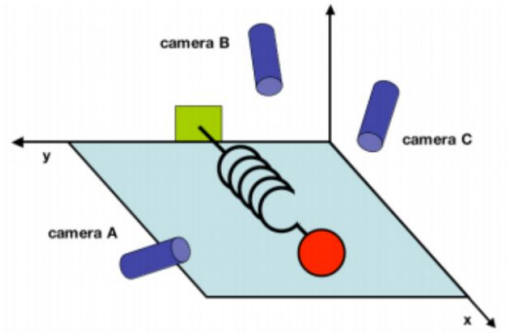


FIG. 1 A toy example. The position of a ball attached to an oscillating spring is recorded using three cameras A, B and C. The position of the ball tracked by each camera is depicted in each panel below.

source: http://www.snl.salk.edu/~shlens/pca.pdf

Example: 1d harmonic oscillator



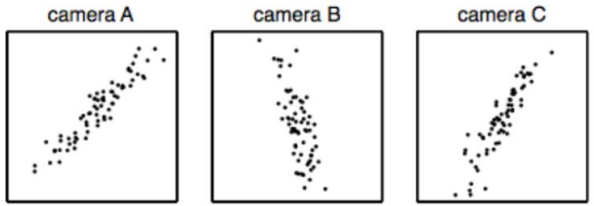


FIG. 1 A toy example. The position of a ball attached to an oscillating spring is recorded using three cameras A, B and C. The position of the ball tracked by each camera is depicted in each panel below.

source: http://www.snl.salk.edu/~shlens/pca.pdf

NOTE

In this case the "truth" is (nearly) one-dimensional. We don't generally know what the "truth" is, but the same techniques can apply.

Curse of dimensionality

The complexity that comes with a large number of features is due in part to the curse of dimensionality.

Namely, the sample size needed to accurately estimate a random variable taking values in a *d*-dimensional feature space grows exponentially with *d* (almost).

(More precisely, the sample size grows exponentially with $l \leq d$, the dimension of the manifold *embedded* in the feature space).

Curse of dimensionality

Another way of characterizing this is to say that high-dimensional spaces are inherently sparse.

ex: A high-dimensional orange contains most of its volume in the rind!

ex: A high-dimensional hypercube contains most of its volume in the corners!

Curse of dimensionality

In either case, most of the points in the space are “far” from the center.

This illustrates the fact that local methods will break down in these circumstances (eg, in order to collect enough neighbors for a given point, you need to expand the radius of the neighborhood so far that locality is not preserved).

Dimensionality reduction

Q: What is the goal of dimensionality reduction?

Dimensionality reduction

Q: What is the goal of dimensionality reduction?

We'd like to analyze the data using the most meaningful basis (or coordinates) possible.

More precisely: given an n x d matrix X (encoding n observations of a d-dimensional random variable), we want to find a k-dimensional representation of X ($k < d$) that captures the information in the original data, according to some criterion.

Dimensionality reduction

Q: What is the goal of dimensionality reduction?

- reduce computational expense
- reduce susceptibility to overfitting
- reduce noise in the dataset
- enhance our intuition

Dimensionality reduction

Q: How is dimensionality reduction performed?

Dimensionality reduction

Q: How is dimensionality reduction performed?
A: There are two approaches: feature selection and feature extraction.

feature selection – selecting a subset of features using an external criterion (*filter*) or the learning algo accuracy itself (*wrapper*)

feature extraction – mapping the features to a lower dimensional space

Dimensionality reduction

Feature selection is important, but typically when people say dimensionality reduction, they are referring to *feature extraction*.

The goal of feature extraction is to create a new set of coordinates that *simplify the representation* of the data.

Dimensionality reduction

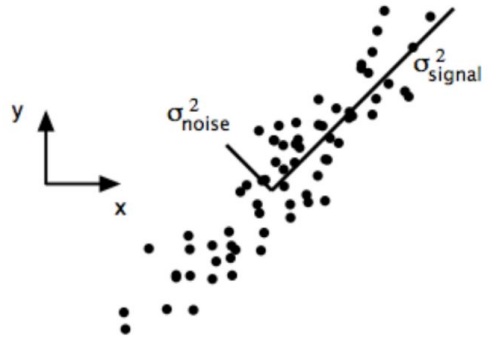


FIG. 2 Simulated data of (x, y) for camera A. The signal and noise variances σ^2_{signal} and σ^2_{noise} are graphically represented by the two lines subtending the cloud of data. Note that the largest direction of variance does not lie along the basis of the recording (x_A, y_A) but rather along the best-fit line.

source: http://www.snl.salk.edu/~shlens/pca.pdf

Dimensionality reduction

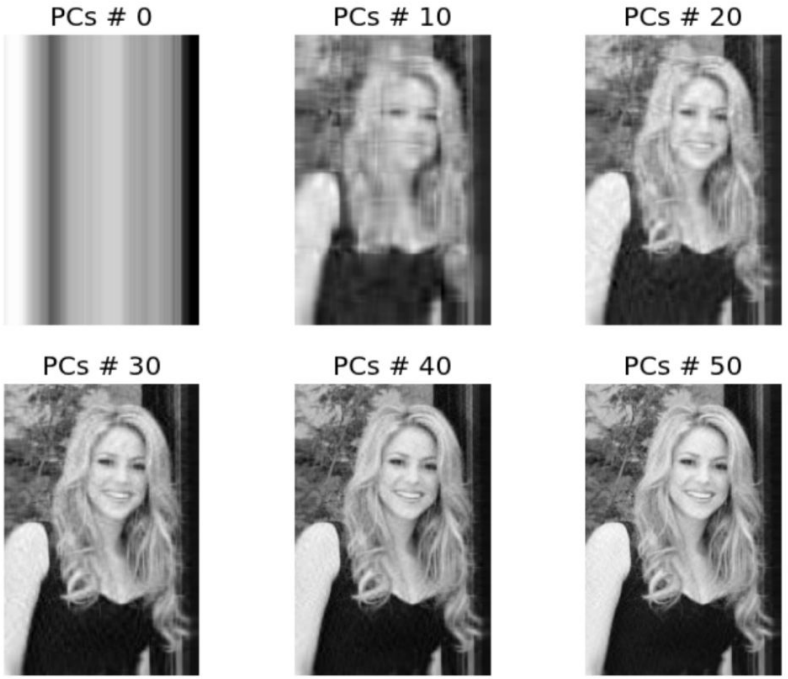
Q: What are some applications of dimensionality reduction?

Dimensionality reduction

Q: What are some applications of dimensionality reduction?

- topic models (document clustering)
- image recognition/computer vision
- bioinformatics (microarray analysis)
- speech recognition
- astronomy (spectral data analysis)
- recommender systems

Dimensionality reduction



Principal Component Analysis

PCA

Principal component analysis is a dimension reduction technique that can be used on a matrix of any dimensions.

This procedure produces a new basis, each of whose components retain as much variance from the original data as possible.

The PCA of a matrix X boils down to the eigenvalue decomposition of the covariance matrix of X.

Covariance matrices

The covariance matrix *C* of a matrix *X* is always square:

$$C = \begin{bmatrix} \mathrm{E}[(X_1 - \mu_1)(X_1 - \mu_1)] & \mathrm{E}[(X_1 - \mu_1)(X_2 - \mu_2)] & \cdots & \mathrm{E}[(X_1 - \mu_1)(X_n - \mu_n)] \\ \mathrm{E}[(X_2 - \mu_2)(X_1 - \mu_1)] & \mathrm{E}[(X_2 - \mu_2)(X_2 - \mu_2)] & \cdots & \mathrm{E}[(X_2 - \mu_2)(X_n - \mu_n)] \\ \vdots & \vdots & \ddots & \vdots \\ \mathrm{E}[(X_n - \mu_n)(X_1 - \mu_1)] & \mathrm{E}[(X_n - \mu_n)(X_2 - \mu_2)] & \cdots & \mathrm{E}[(X_n - \mu_n)(X_n - \mu_n)] \end{bmatrix}.$$

off-diagonal elements C_{ij} give the *covariance* between X_i, X_j $(i \neq j)$

diagonal elements C_{ii} give the *variance* of X_i

Eigenvalue decomposition

The *eigenvalue decomposition* of a square matrix C is given by:

$$C = Q \Lambda Q^{-1}$$

The columns of Q are the eigenvectors of C, and the values in Λ are the associated eigenvalues of C.

For an eigenvector v of C and its eigenvalue λ, we have the important relation:

$$Cv = \lambda v$$

Eigenvalue decomposition

The *eigenvalue decomposition* of a square matrix C is given by:

$$C = Q \Lambda Q^{-1}$$

The columns of Q are the eigenvectors of C, and the values in Λ are the associated eigenvalues of C.

For an eigenvector v of C and its eigenvalue λ, w
the important relation:

$$Cv = \lambda v$$

NOTE

This relationship *defines* what it means to be an eigenvector of C.

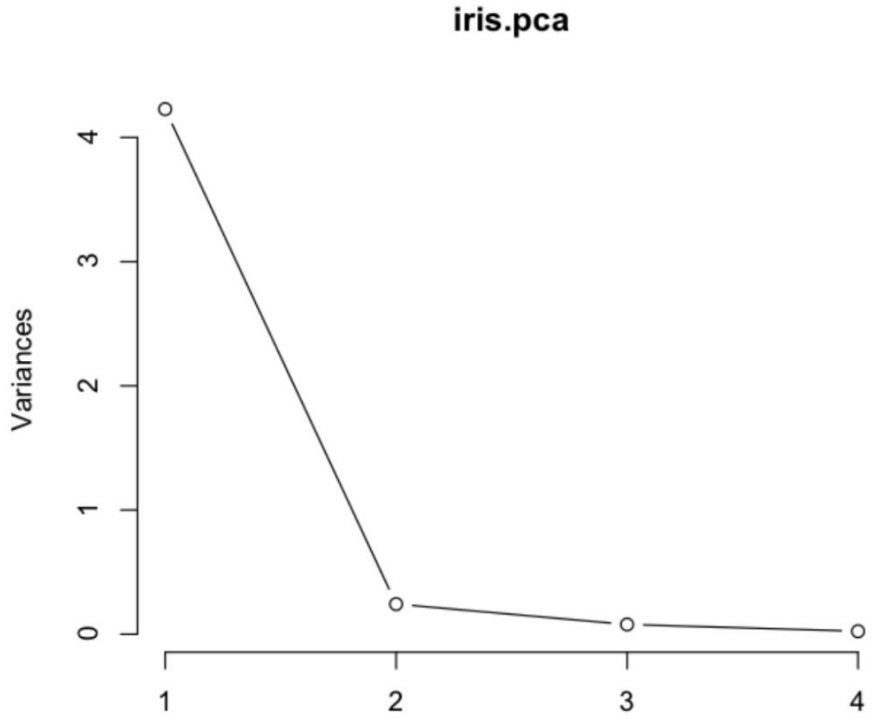
PCA

The eigenvectors form a basis of the vector space on which *C* acts (eg, they are orthogonal).

Furthermore the basis elements are ordered by their eigenvalues (from largest to smallest), and these eigenvalues represent the amount of variance explained by each basis element.

This can be visualized in a scree plot, which shows the amount of variance explained by each basis vector.

PCA



NOTE

Looking at this plot also gives you an idea of how many principal components to keep.

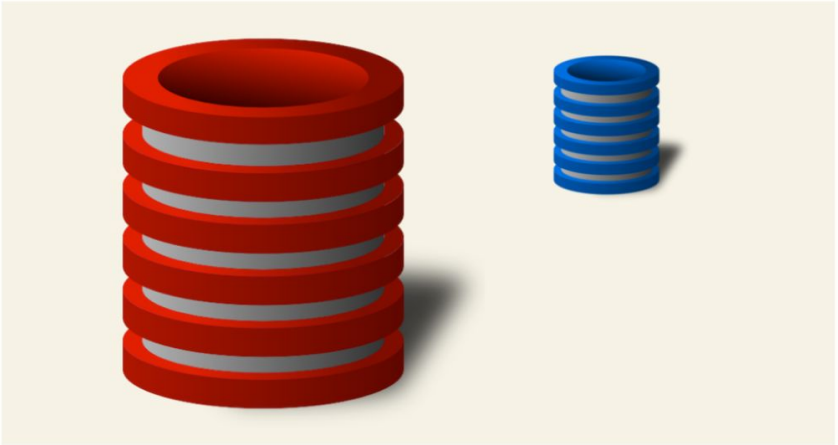
Apply the *elbow test*: keep only those pc's that appear to the left of the elbow in the graph.

Extra Topics!

Databases

Databases

- An organized collection of data
- Organized using a schema (like a blueprint of a database)
- Organized into tables with different sets of data

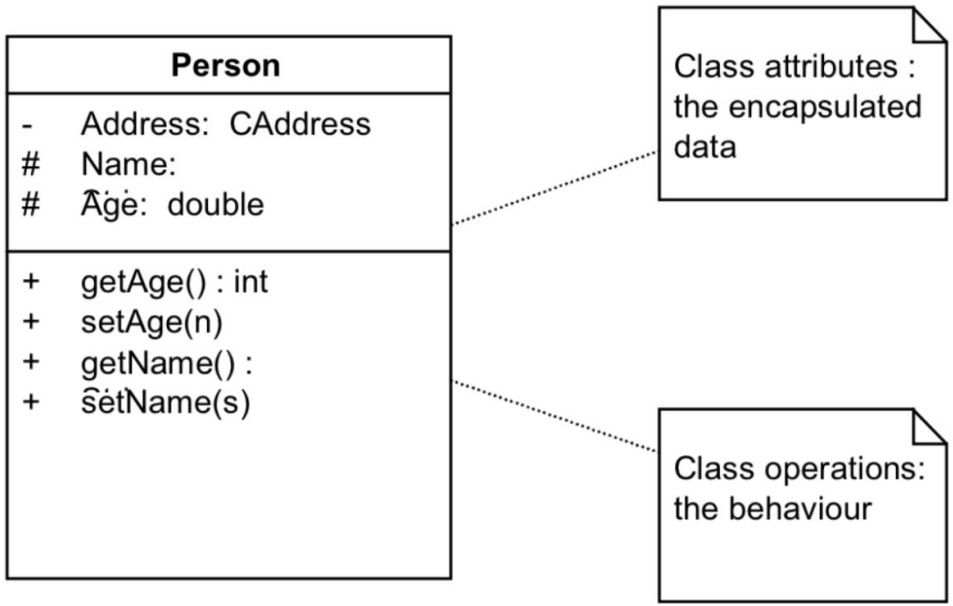


Databases

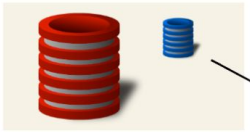
WHY EVEN USE A DATABASE?

- Easy to store and more importantly, retrieve data
- Generally has a structured language for interacting with the data
- Reliable and **scalable**
- Access large amounts of data relatively quickly

Databases


Person
- Address: CAddress
Name:
Age: double
+ getAge() : int
+ setAge(n)
+ getName() :
+ setName(s)
Class attributes : the encapsulated data
Class operations: the behaviour

Sql v NoSql



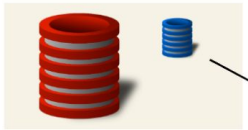
Relational

- Traditional rows and columns data
- **Strict** structure / Primary Keys
- Entire column for each feature
- Industry standard

NoSql

- No well defined data structure
- Works better for unstructured data
- Cheaper hardware
- Popular among Startups

Sql v NoSql



Relational

Examples

- MySQL
- Oracle
- Postgres
- SQLite

NoSql

Examples

- MongoDB
- CouchDB
- Redis
- Casssandra

SQL

Structured

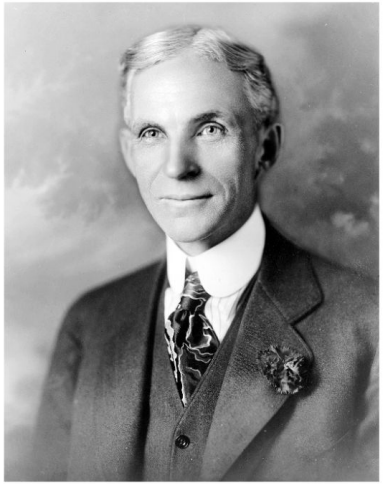
Query

Language

Is a language for database communication

MapReduce

MapReduce



"Nothing is particularly hard if you divide it into small jobs."

- Henry Ford

MapReduce

MapReduce is broken up into steps:

1. Map: produces key value pairs depending on the task
2. Shuffle/Sort (*optional*) sorts the key value pairs, could make new ones
3. Reduce combines the key value pairs into a single output

MapReduce

MapReduce is broken up into steps:

1. Map: produces key value pairs depending on the task
2. Shuffle/Sort (*optional*) sorts the key value pairs
3. Reduce combines the key value pairs into a single output

NOTE

Ideally the map function is run over a *cluster* of computers in *parallel*

MapReduce Example

Temperatures

- **Given**: several data tables consisting of data organized in a (City: Temperature) over several days
- **Goal**: Find the max temperature for each city

File 1

New York City: 32
Chicago: 22
New York City: 36
Miami: 67
…

File 2

Miami: 77
New York City: 32
New Haven: 29
…

MapReduce Example

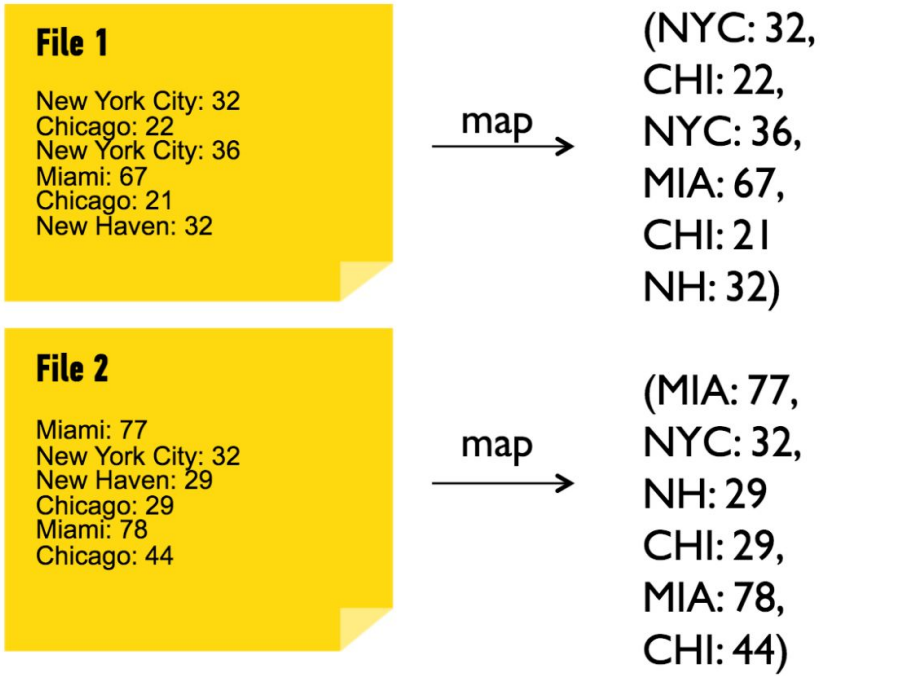
File 1

New York City: 32
Chicago: 22
New York City: 36
Miami: 67
Chicago: 21
New Haven: 32

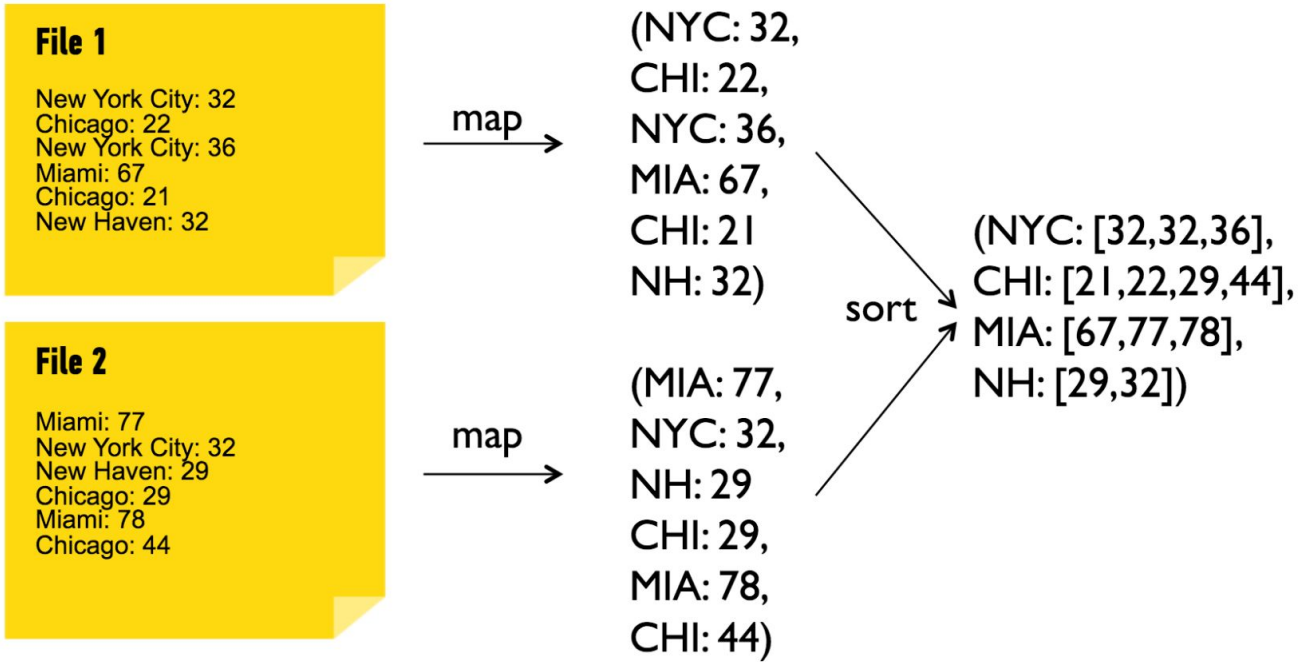
File 2

Miami: 77
New York City: 32
New Haven: 29
Chicago: 29
Miami: 78
Chicago: 44

MapReduce Example



MapReduce Example



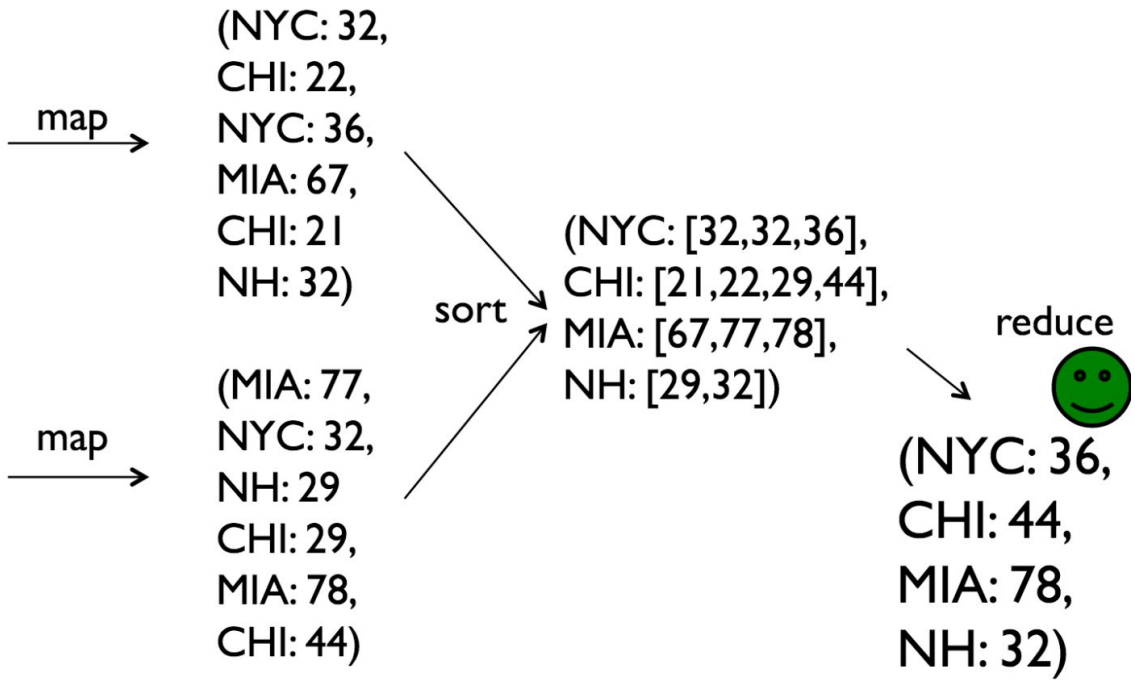
MapReduce Example

File 1

New York City: 32
Chicago: 22
New York City: 36
Miami: 67
Chicago: 21
New Haven: 32

File 2

Miami: 77
New York City: 32
New Haven: 29
Chicago: 29
Miami: 78
Chicago: 44



Benefits of MapReduce

MapReduce Benefits

- Extremely scalable: algorithm for a MB of Data will work for a PetaByte of Data
- Many implementations with documentation exist for Java, C, Python, etc..
- Relatively fast compared to straight though processing

MapReduce Example: Word Count

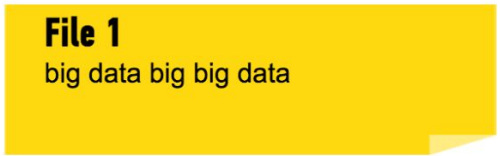
MECHANICAL TURK ALERT!!!

You are all about to become a MapReduce cluster. All of you will be either a:

- Mapper
- Sorter
- Reducer

MapReduce Example: Word

- Mappers

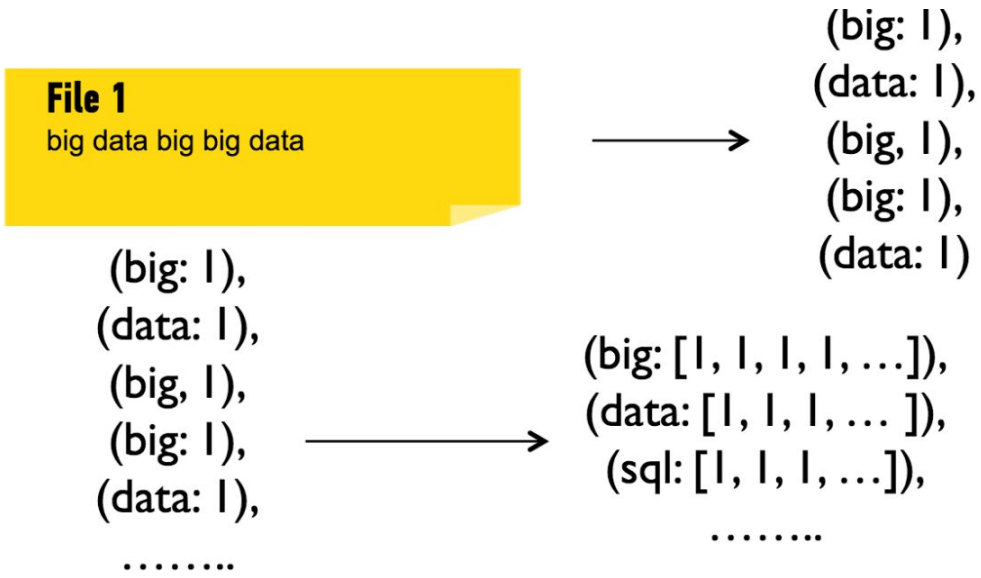


(big: 1),
(data: 1),
(big, 1),
(big: 1),
(data: 1)

- Sorters
- Reducer

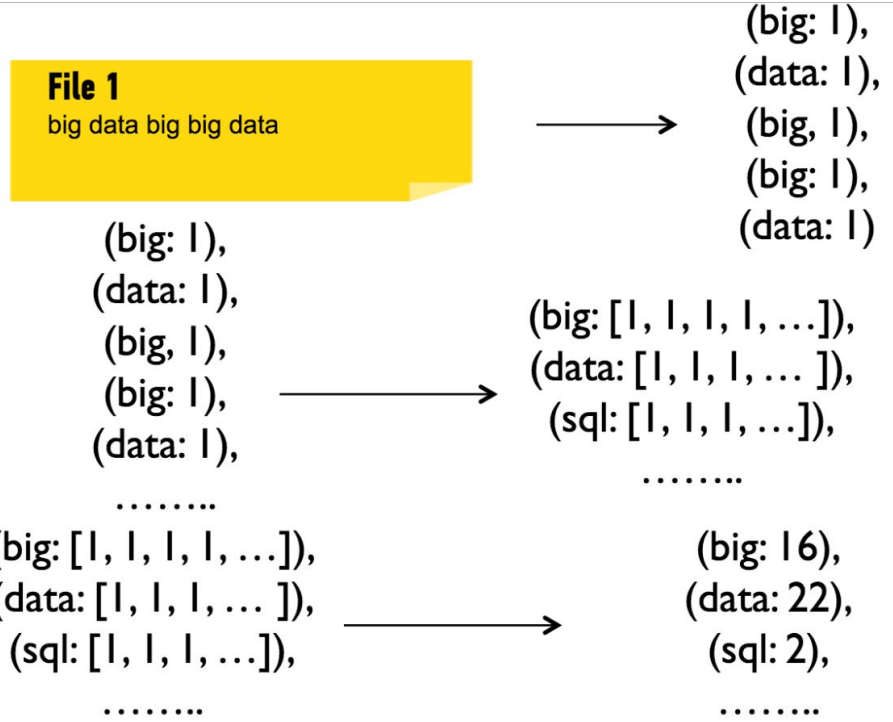
MapReduce Example: Word Count

- Mappers
- Sorters
- Reducer



MapReduce Example: Word

- Mappers
- Sorters
- Reducer



Implementations of MapReduce

- Creator: Apache
- Implemented in: Java
- Has API for python
- Open source

- Creator: UC Berkley
- Owned by: Apache
- Implemented in: Java
- Built in API, ML, Graphing capabilities

- Creator: Nokia
- Implemented in: Python
- Open source on Github

Any Questions?