DIMENSIONALITY REDUCTION

And representation learning

DATA TRANSFORMATION

- Our data is provided in a given form
  - Tabular (vectors)
  - Network
  - Time series
  - Text
  - Images
  - ….

- To use the full potential of data mining, you might want to study it from multiple angles
  - How to convert from tabular to graph?
  - From Graph to Tabular?
  - From images/text to tabular (embedding)?

DIMENSIONALITY REDUCTION

Low dimensionality embedding

DIMENSIONALITY REDUCTION

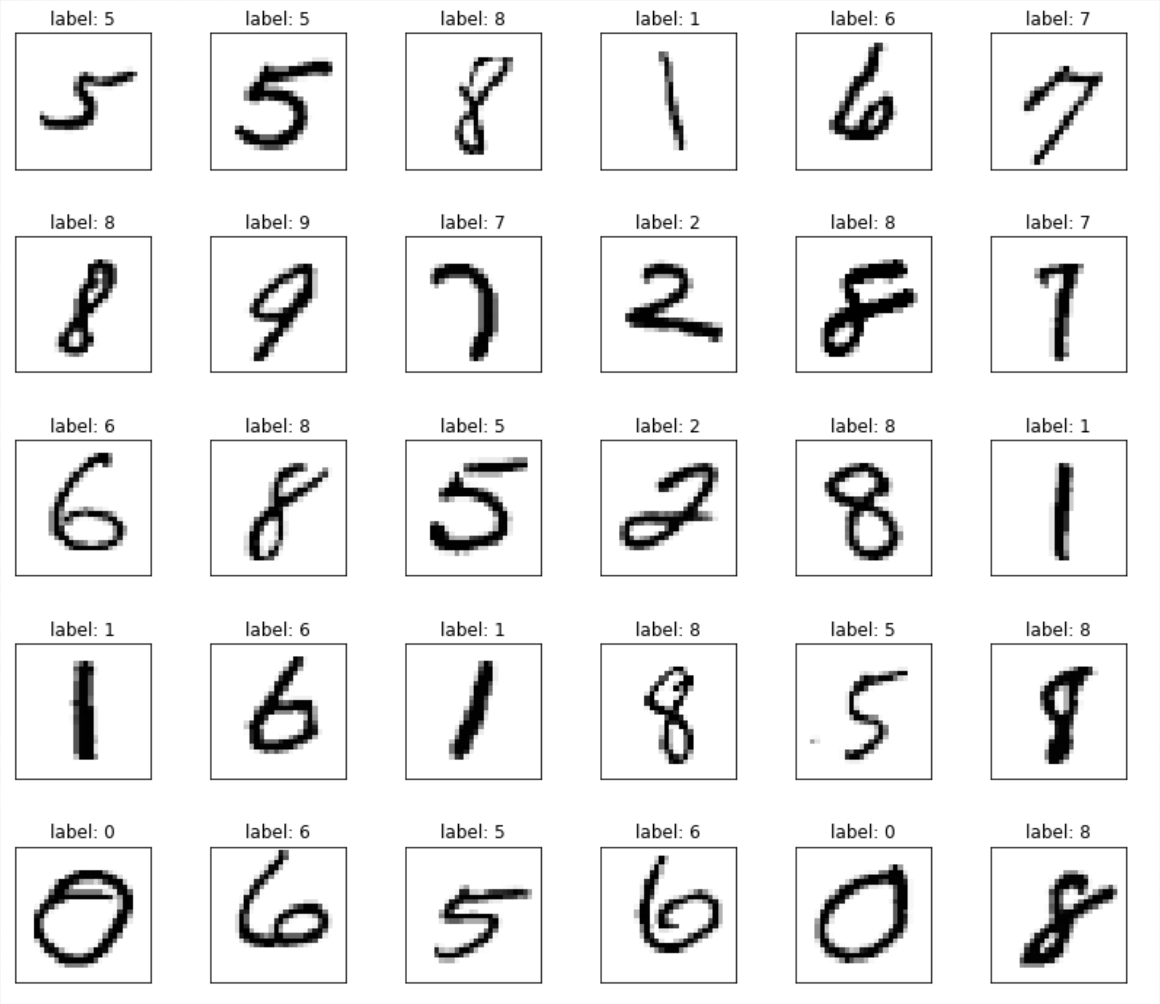
- Data Mining objective: understand our data
  - We get a dataset composed of many features
    - Or worst, complex object (image, sound, graph…)
  - How to understand the organization of our data?
  - How to perform clustering?

VISUALIZATION

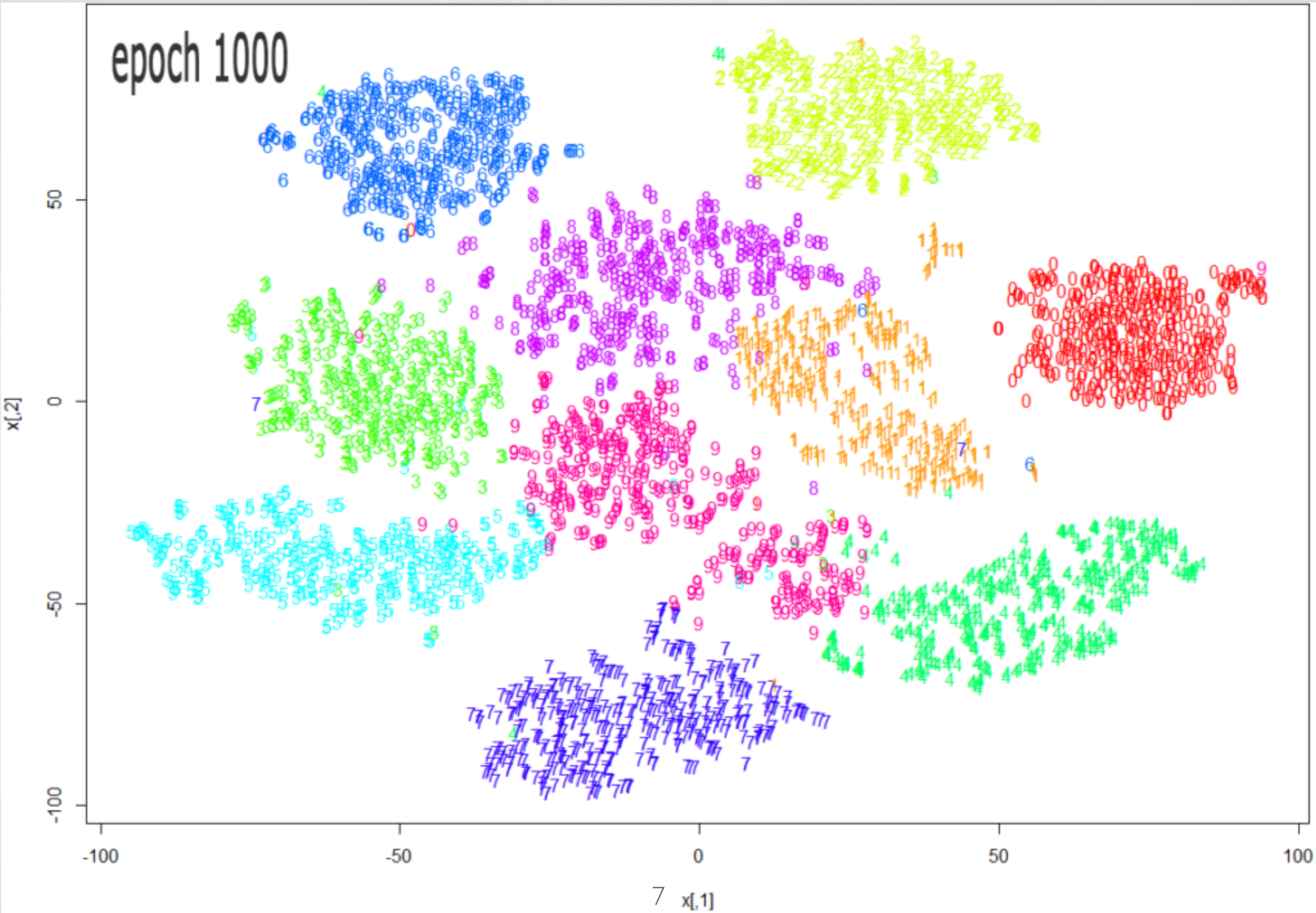
- Your data is perfectly fine, but you want to intuitively understand how it is organized
  - Are there groups of similar objects?
  - Are my clusters meaningful?
  - Is my classification/clustering on some types of elements and not others.

VISUALIZATION

Example: MNIST Dataset
Each pixel is a variable



t-SNE embedding



CURSE OF DIMENSIONALITY

- Having hundreds/thousands of attributes is a problem for data analysis.
  - e.g.: medicine: blood analysis, genomics….
  - e.g.: cooking recipes: each column an ingredient…
- We want to reduce the number of attributes while keeping most of the information
- Also helps with scalability

CORRELATION

- Assume that you have correlated features such as age, height and weight.
  - Redundancy ! Computational Inefficiency
    - e.g., Decision tree will spend a lot of time choosing between them for no reason
  - Risk of overfitting
    - noise between correlated variables used to distinguish individuals
  - Model interpretability
    - e.g., a model will say that y depends on x or w randomly, if x and w correlated

- Dimensionality reduction can create a single variable to capture what is common
  - The rest can be lost or captured by another feature,
    - Engine horsepower, Car weight, Fuel Consumption
    - =>Performance index (horsepower and weight)
    - =>Efficiency score (weight and fuel consumption)

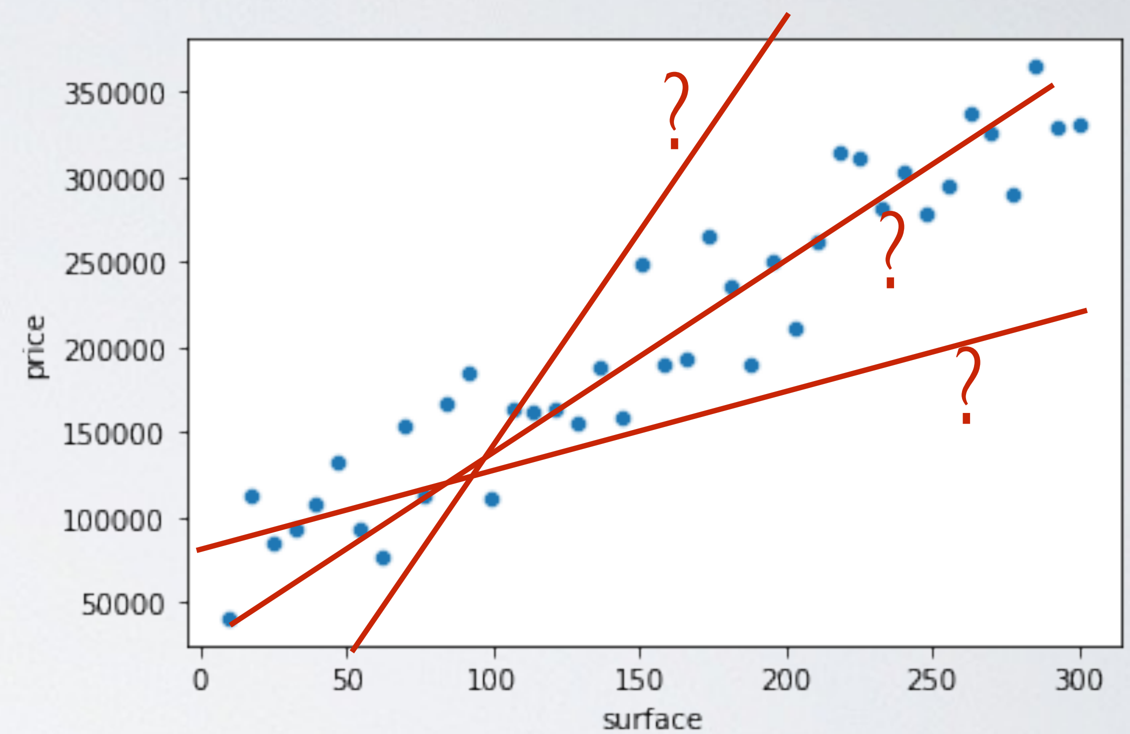
PCA

PCA

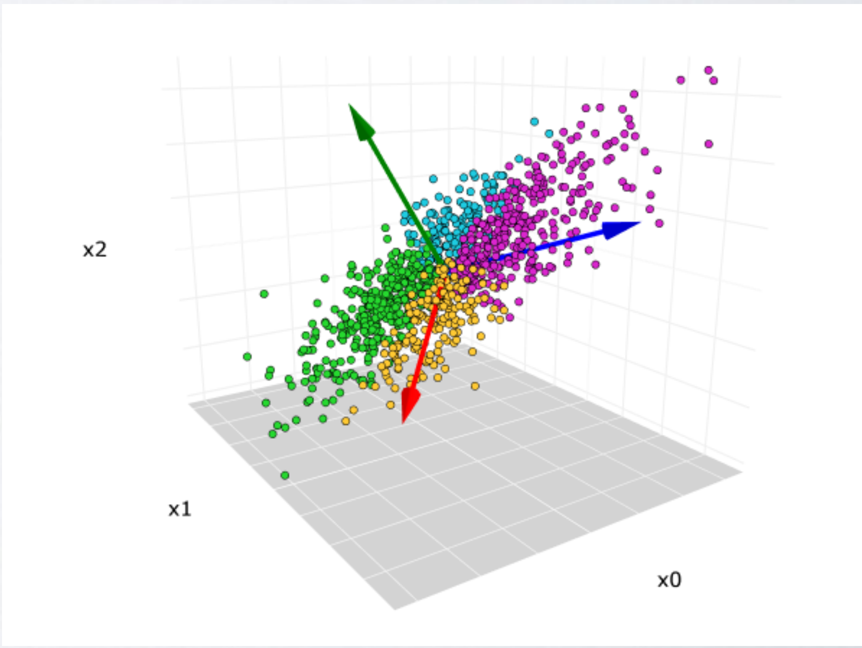
- PCA: Principal Component Analysis
- Defines new dimensions that are linear combinations of initial dimensions
  - Objective: concentrate the **variance** on some dimensions
    - So that we can keep only these ones.
    - Those we remove contain low variance, thus low information

PCA

- Algorithm:
  - 1)Find an "axis", a unit vector defining a line in the space
    - That minimizes the variance=>the squared distance from all points to that line

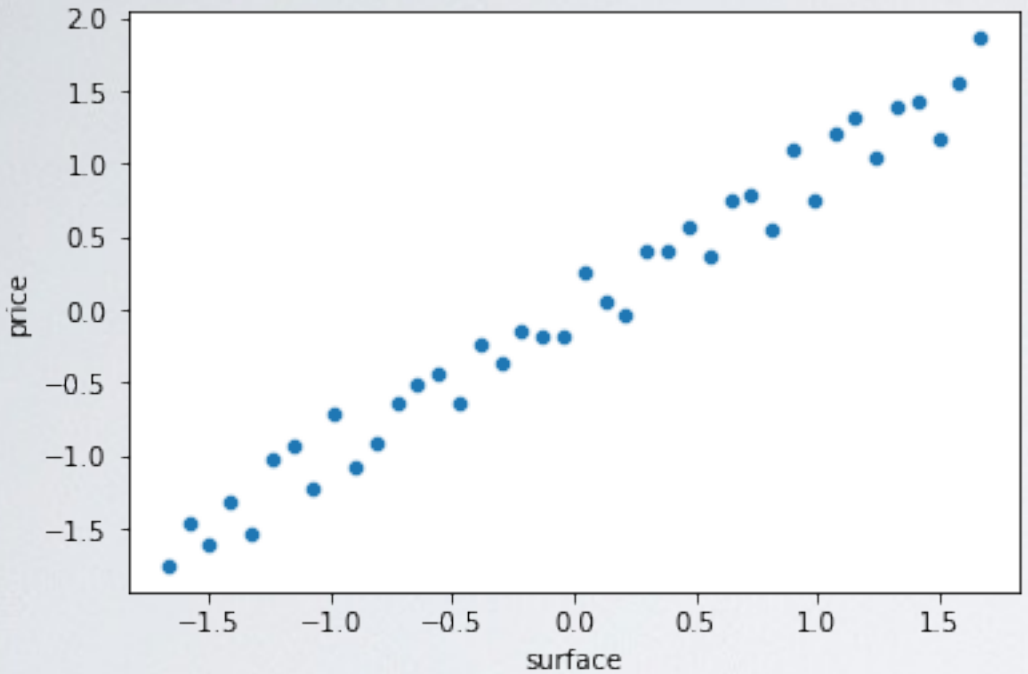


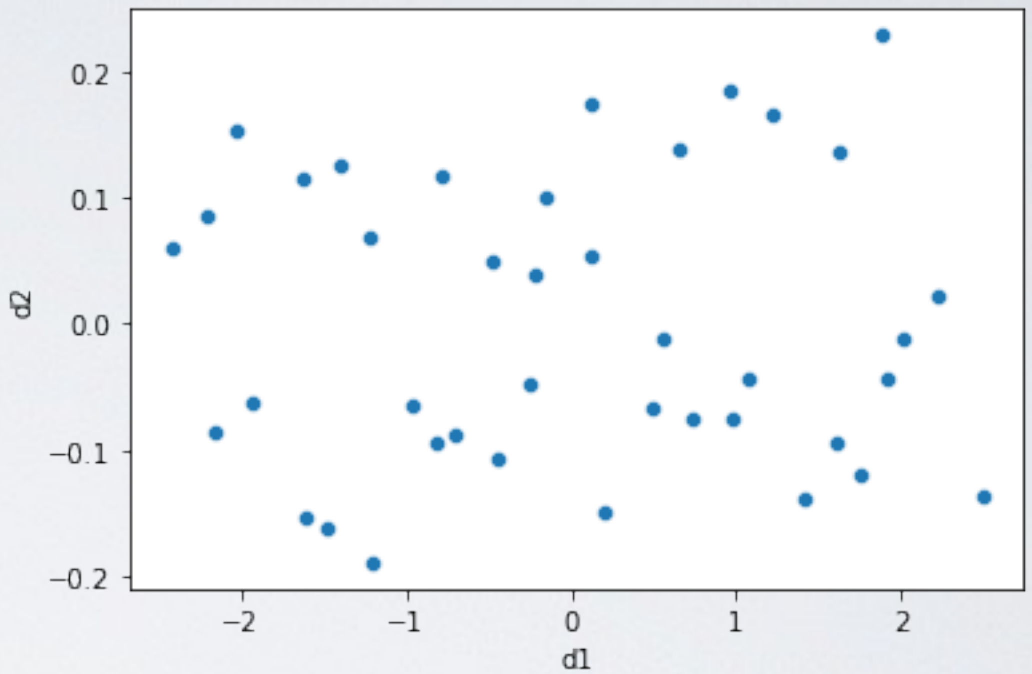
- 2)**For** d **in** [2:(initial_d)]
  - Find another axis, with two constraints:
    - Orthogonal to all previous axis
    - Among those, minimizing the variance



- 3)At the end, keep the first k dimensions
  - Some information is lost

EXAMPLE PCA 2D





Covariance matrix (original)

```
[1.        , 0.98675899],
       [0.98675899, 1.        ]
```

Sum of variance

2

Variance by dimension

1 1

Covariance matrix (pca)

```
[ 1.98675899e+00, 0],
[0,  1.32410092e-02]
```

Sum of variance

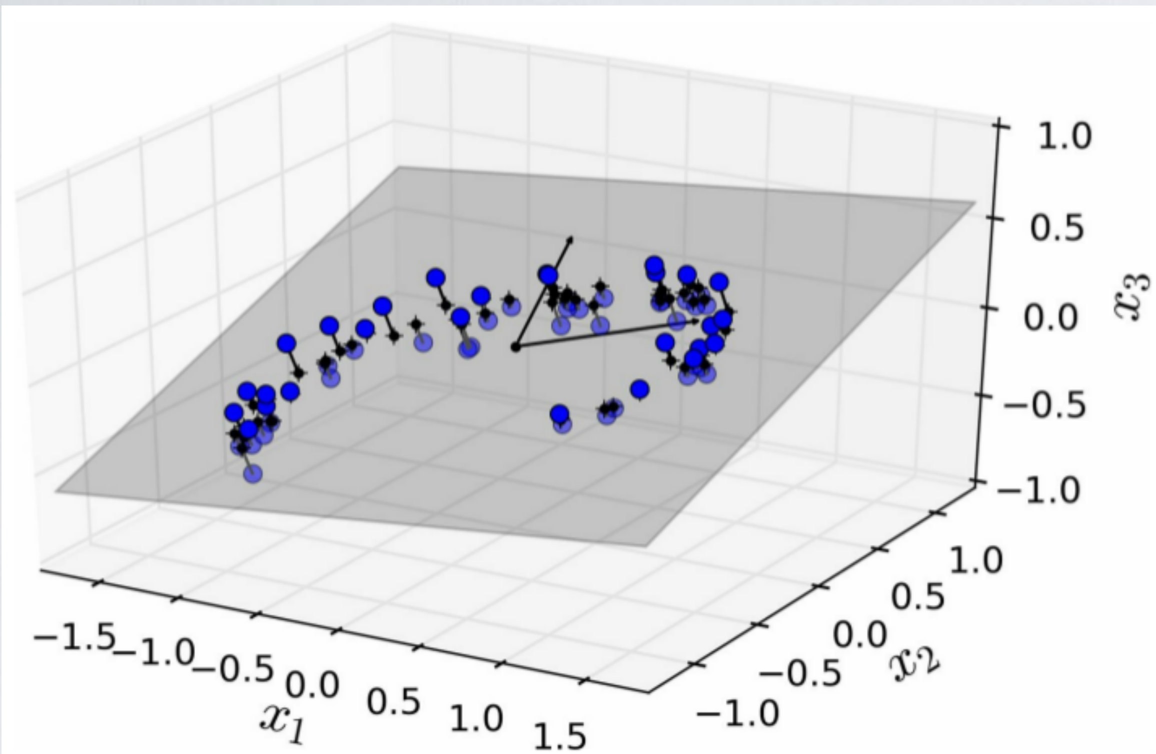
2

Variance by dimension

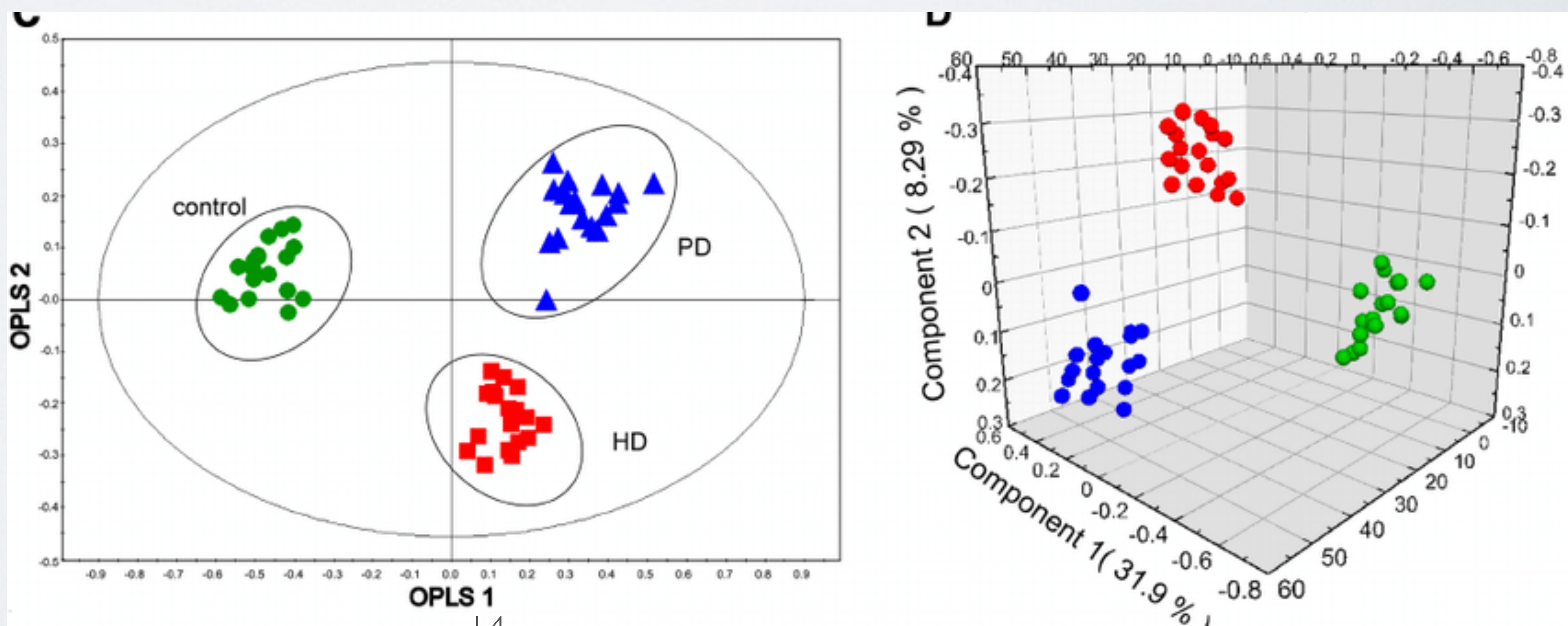
1.98675899 0.01324101

Explained variance(ratio) [0.9933795, 0.0066205]

3D=>2D



x_1
x_2
x_3
−1.5 −1.0 −0.5 0.0 0.5 1.0 1.5
−1.0 −0.5 0.0 0.5 1.0
1.0 0.5 0.0 −0.5 −1.0

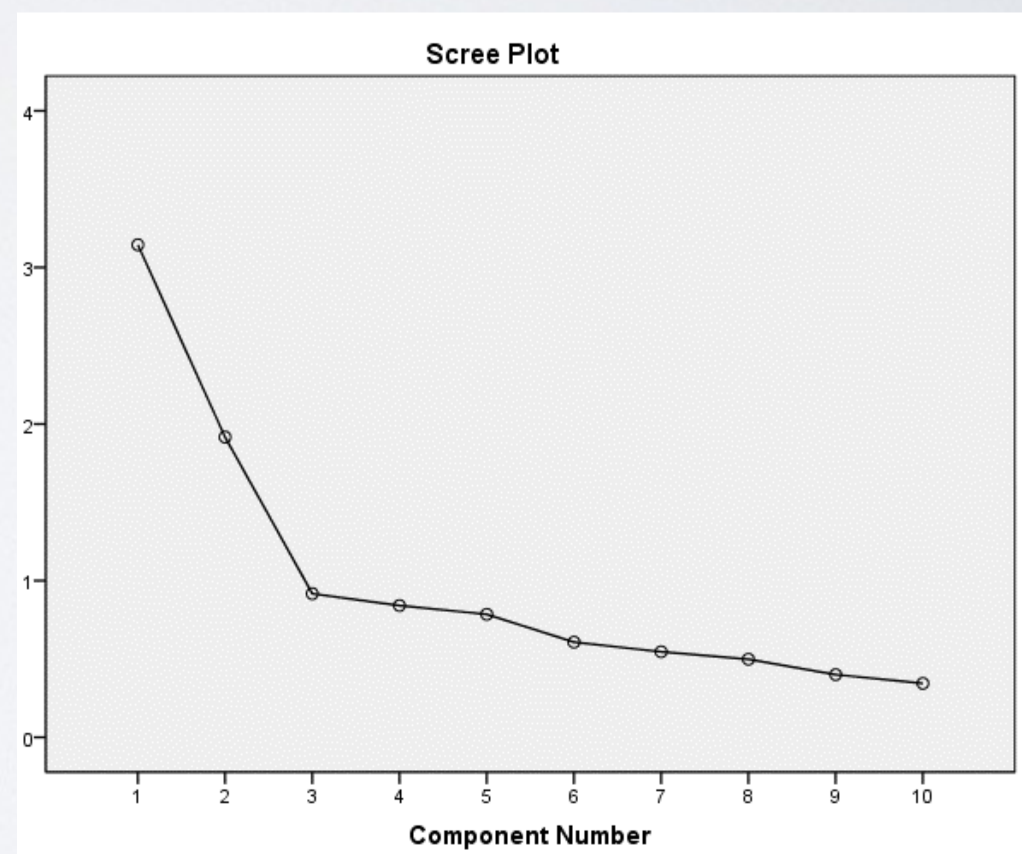


C
control
PD
HD
OPLS 1
OPLS 2
D
Component 1(31.9 %)
Component 2 (8.29 %)

CHOOSING COMPONENTS

- How to choose k?
  - Elbow method… BIC/AIC…
  - OR fix beforehand a min threshold of explained variance, e.g.: 80%
    - We are fine with losing 20% of information
  - If there is a downstream task, cross-validation

Explained
variance



COMPUTATION IN PRACTICE

- From standardized dataset X

- Method 1:
  - 1)Compute the Covariance Matrix ($X^T X$)
    - => Linear Correlation Matrix
  - 2) Find the eigenvectors of this matrix
    - $X^T X = V \Lambda V^T$
    - V: eingenvectors = Pincipal components, Λ: Eigenvalues, = explained variance

- Method 2:
  - Apply SVD matrix decomposition
  - $X = U \Sigma V^T$
    - U: left singular vectors. Σ: diagonal matrix with the singular values, V^T:right singular vectors (the principal components)

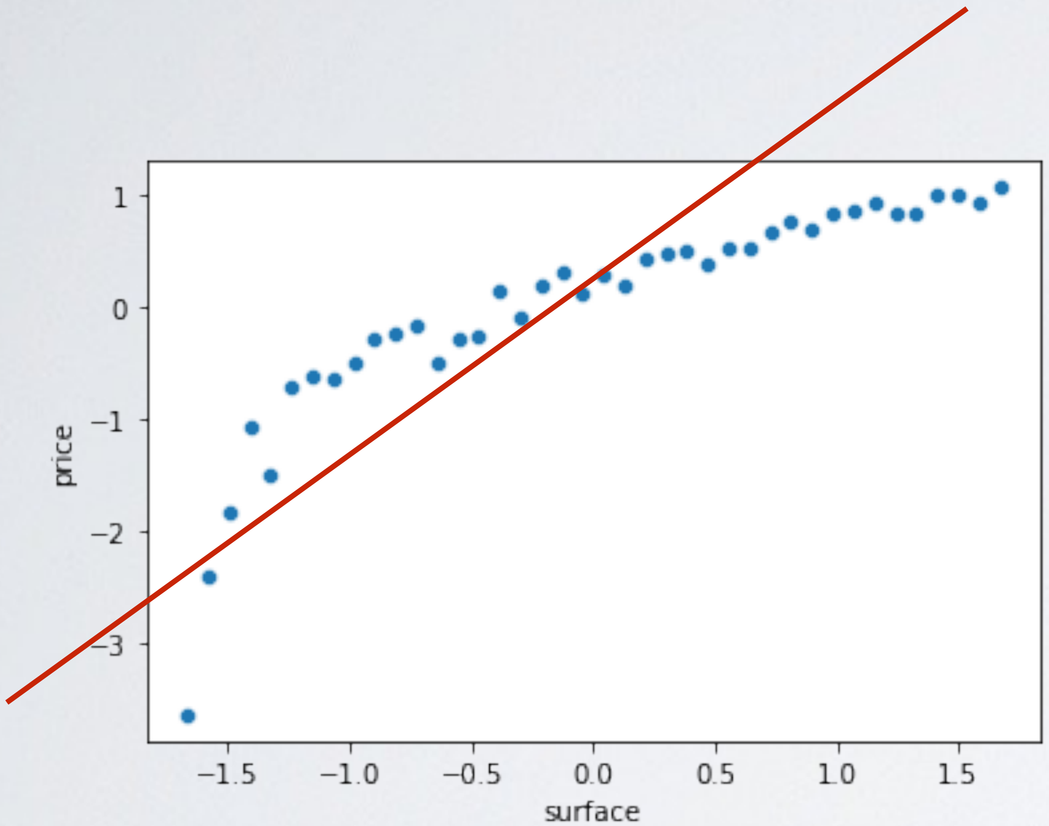
COMPUTATION IN PRACTICE

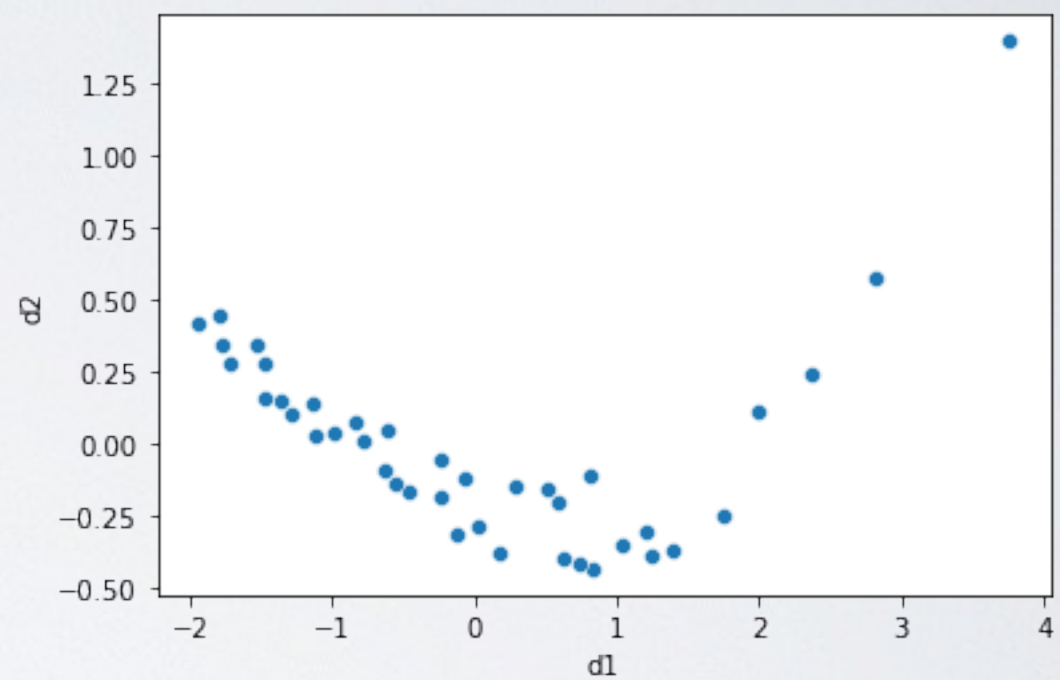
- V are the principal components
- Computing the new positions for each observation:
  - XV

PCA POPULARITY

- Why is PCA popular?
- Similar reasons than linear regression:
  - Useful
    - Eliminate correlations
  - Analytical solutions
    - Guarantee to find the global minimum of the objective
    - Could be done before modern computers
  - Interpretable solution
  - Intuitively pleasant
- No reason to consider it "better" than other methods for demensionality reduction…

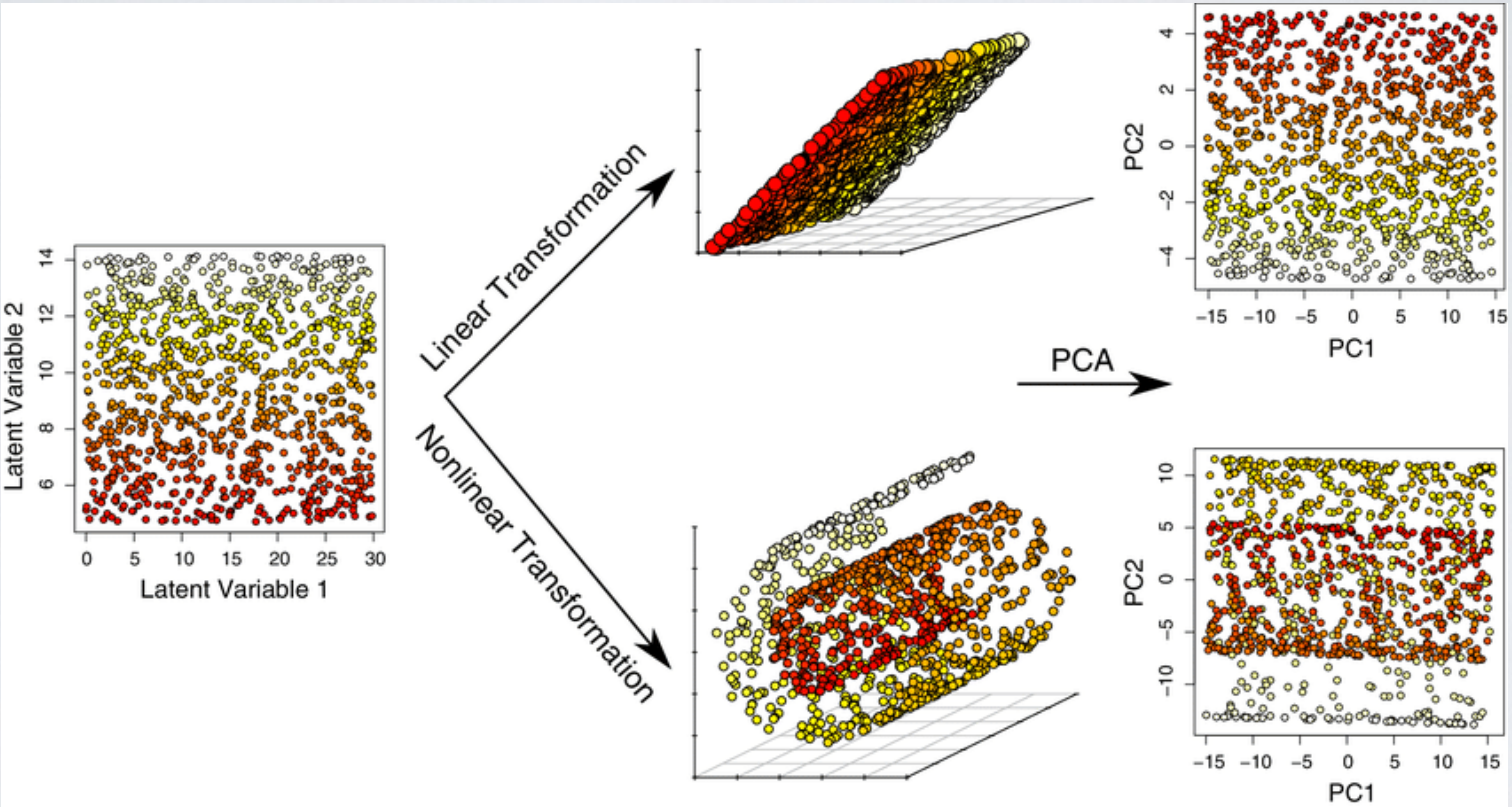
NON-LINEAR SITUATIONS





`Pearson correlation(d1,d2): 0`

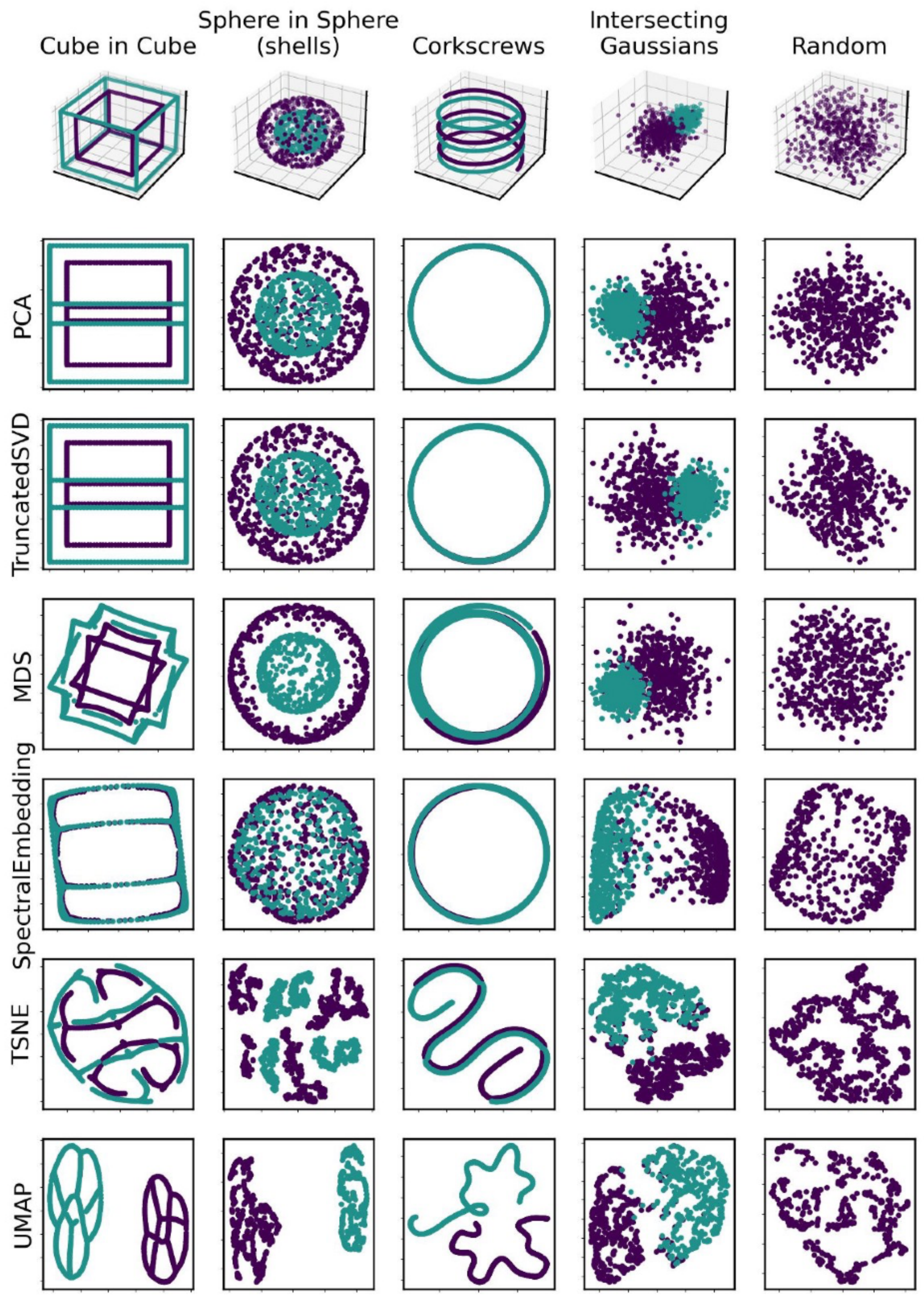
NONLINEAR DATA



MANIFOLDS

MANIFOLDS

- Manifolds are another approach to dimensionality reduction
- The general principle is to
  - 1)Define a notion of distance between elements in the original space
  - 2)Define a notion of distance between elements in a reduced, target space
  - 3)Minimize the difference between distances in original and target space
- In many cases, the process is nonlinear, i.e., we choose distances such as
  - We care more about preserving the distance for items “close” in space than for those “far” from each other


Cube in Cube
Sphere in Sphere (shells)
Corkscrews
Intersecting Gaussians
Random
PCA
TruncatedSVD
MDS
SpectralEmbedding
TSNE
UMAP

MDS

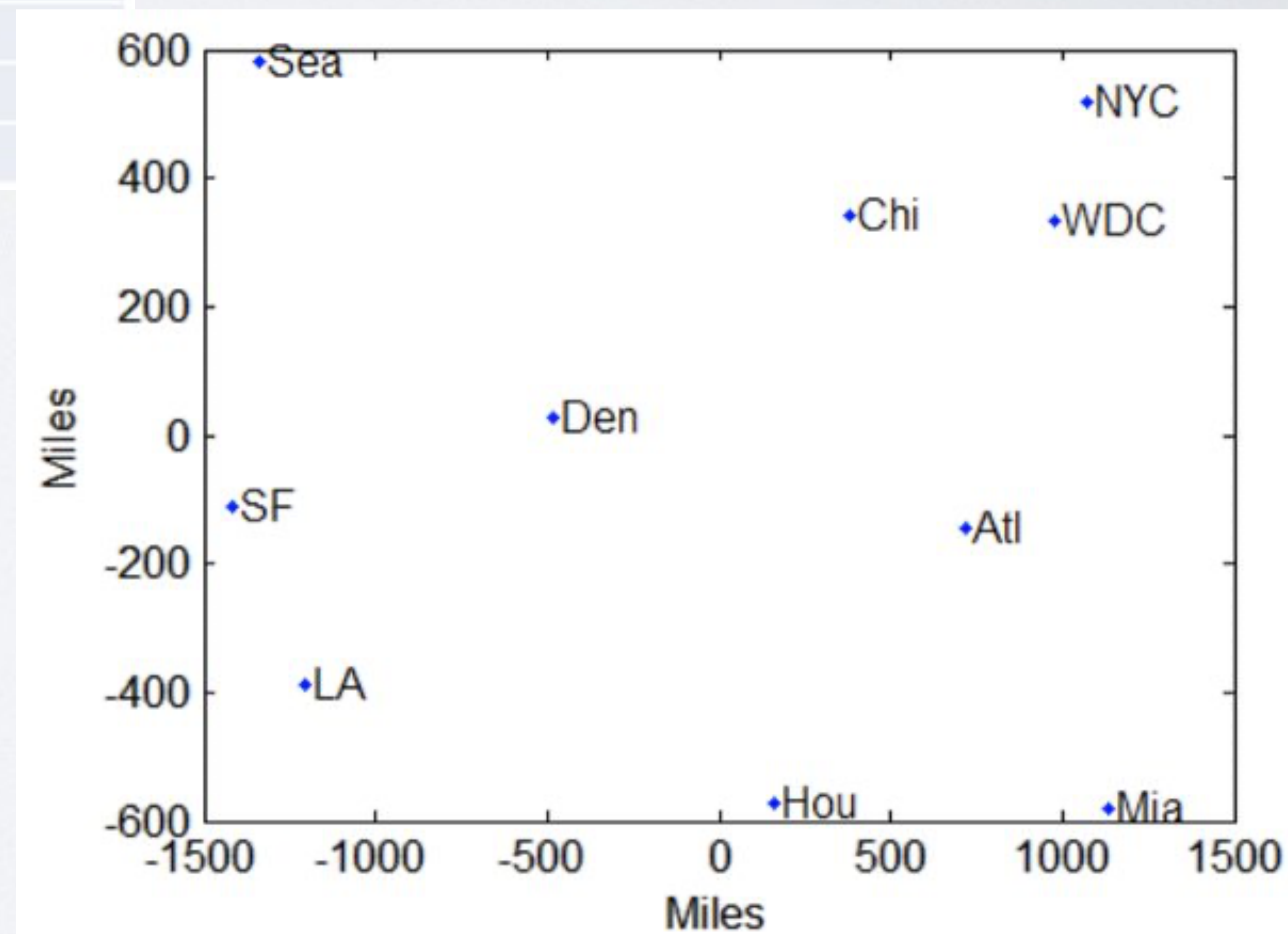
- MDS: Multi-dimensional Scaling:
  - Simply minimize distance between original space and target space
    - e.g., d-dimensional forced to 2-dimensional

- How to do it?
  - 1)Compute all (squared Euclidean) pairwise distances between items=>**Similarity matrix**
    - n x f matrix => n x n matrix
    - Apply double-centering (remove row and column means)
  - 2)Compute PCA on this similarity matrix

- Problems:
    - Very costly (nb features=nb elements), n^2
    - Try to preserve all distances, therefore extremely constrained

MDS

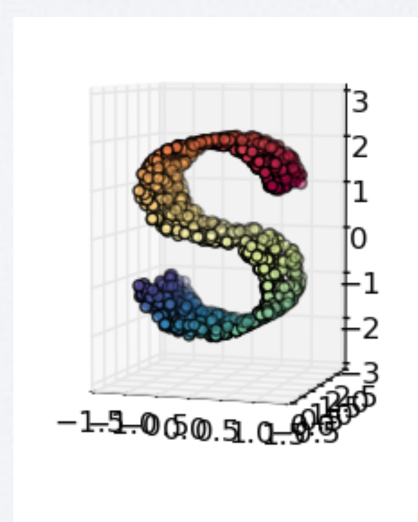
	Atl	Chi	Den	Hou	LA	Mia	NYC	SF	Sea	WDC
Atl	0	587	1212	701	1936	604	748	2139	2182	543
Chi	587	0	920	940	1745	1188	713	1858	1737	597
Den	1212	920	0	879	831	1726	1631	949	1021	1494
Hou	701	940	879	0	1374	968	1420	1645	1891	1220
LA	1936	1745	831	1374	0	2339	2451	347	959	2300
Mia	604	1188	1726	968	2339	0	1092	2594	2734	923
NYC	748	713	1631	1420	2451	1092	0	2571	2408	205
SF	2139	1858	949	1645	347	2594	2571	0	678	[illegible]
Sea	2182	1737	1021	1891	959	2734	2408	678	0	[illegible]
WDC	543	597	1494	1220	2300	923	205	2442	2329	[illegible]

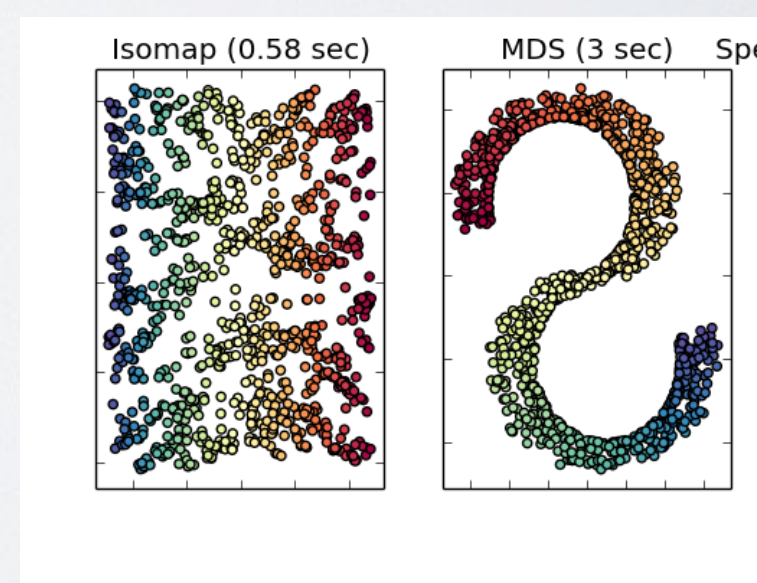


ISOMAP

- Variation of MDS
  - 1)We define a **graph** such as two elements are connected if they are at distance<threshold. (Alternative: fixed number of neighbors)
    - Put a weight on edges=euclidean distance
  - 2)Compute a **similarity** matrix, such as distance = weighted shortest path distance
  - 3)Apply MDS on it

- Non-linear distances





T-SNE

T-SNE

- t-SNE : t-distributed stochastic neighbor embedding
- Non-linear dimensionality reduction
- One of the most popular method for visualizing data in low dimensions

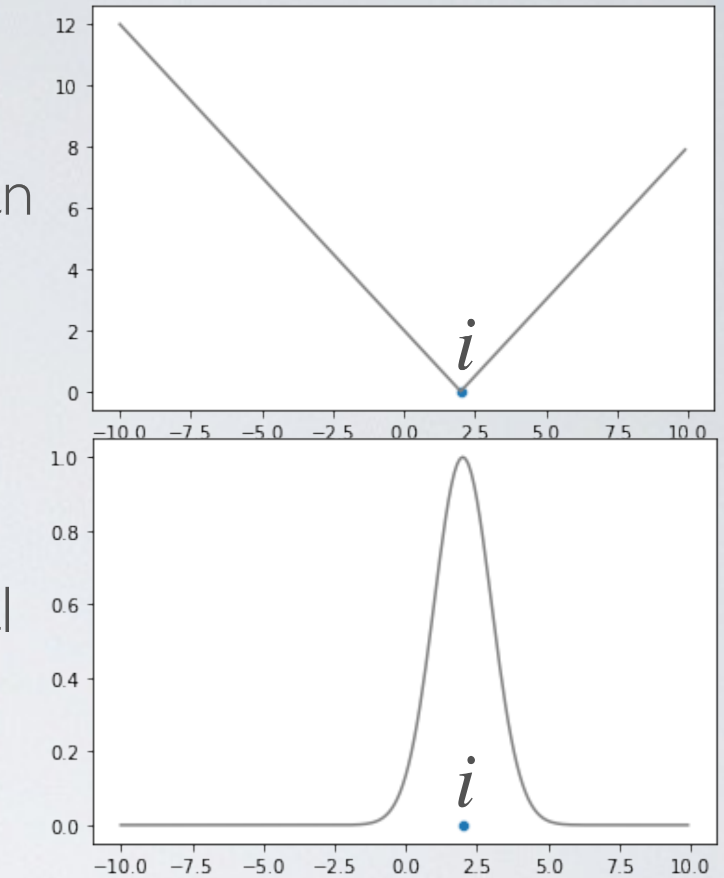
SNE

- General principle:
  - Define a notion of similarity $p_{j|i}$ in the high dimensional space P
    - Based on normal distribution
  - Define a notion of similarity $q_{j|i}$ in the low dimensional space Q
    - Based on student-t distribution, tends to "exaggerate" differences
  - For each point of initial coordinates x_i, find a new coordinate y_i in the lower dimensional space, such as to minimize the difference between P and Q
    - $\forall_{i,j} p_{j|i} \approx q_{j|i}$

SNE

Euclidean

Normal

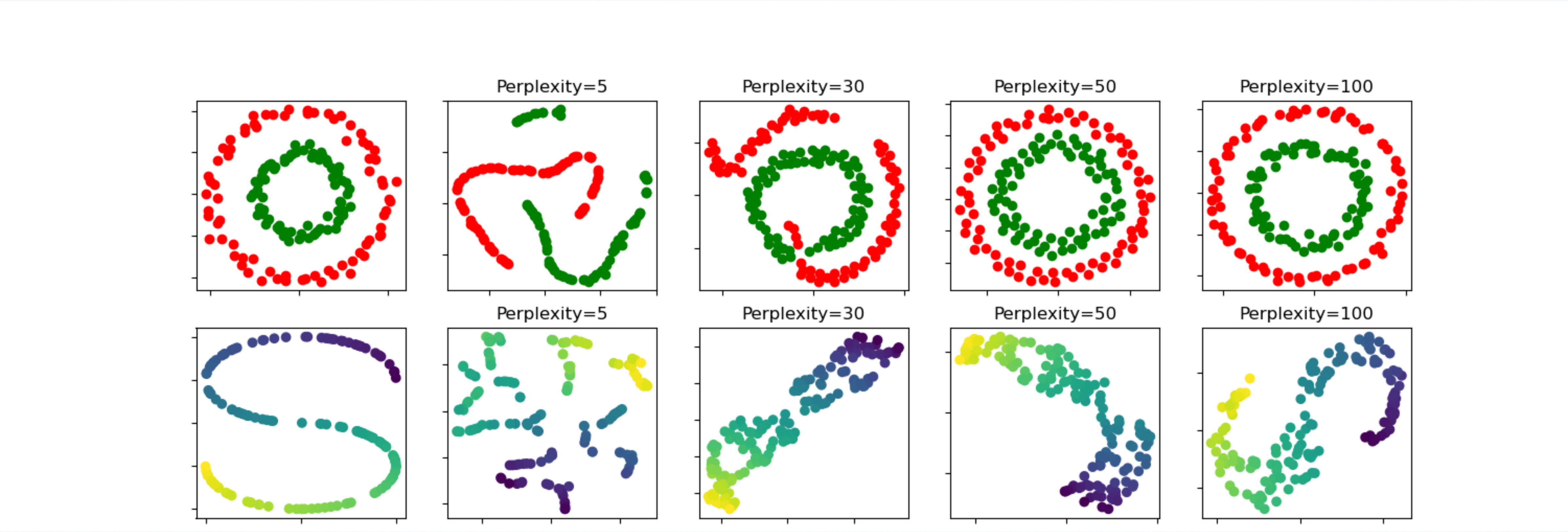


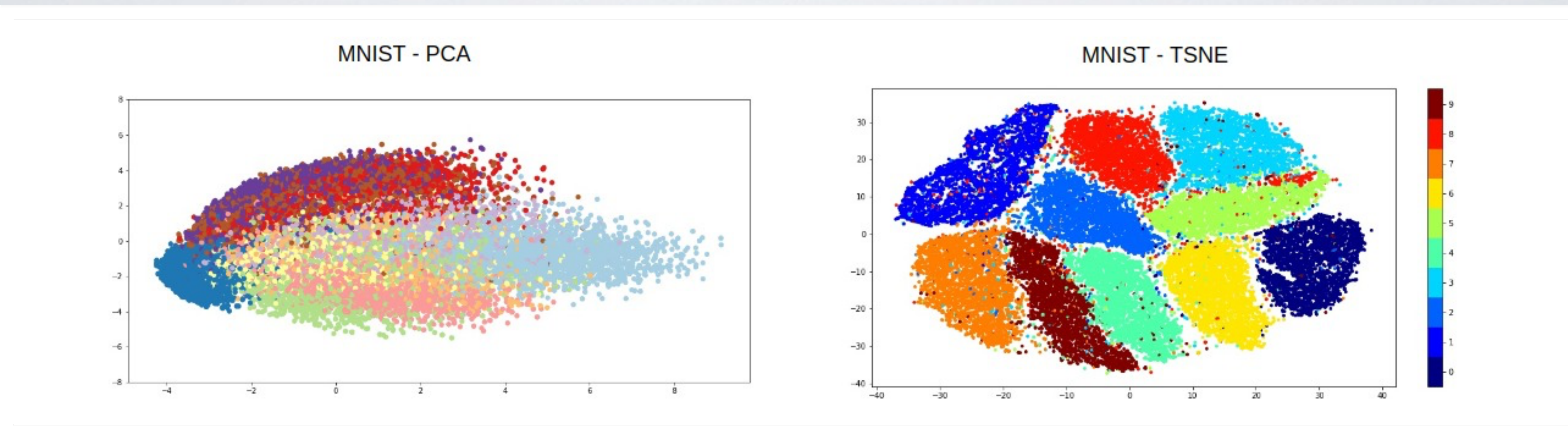
- Distance in the original space P
  - To compute how far j is from i, consider a normal distribution centered in j with variance σ
  - Mathematically: the raw distance is given as $s^P_{j|i} = e^{-\frac{\|x_i - x_j\|^2}{2\sigma^2}}$
  - $p_{j|i} = \dfrac{s^P_{j|i}}{\sum_{k \neq i} s^P_{j|k}}$
    - Normalizes the similarity by sum of similarity to all other points.
    - With proper σ, local definition of similarity

T-SNE: PERPLEXITY

- There is a perplexity parameter σ: it controls how much each point cares more about close neighbors compared with farther neighbors
  - Low σ: Preserve mostly local distances
  - High σ: Give more importance to long-range distances
    - More expensive, more similar to MDS

INFLUENCE OF PERPLEXITY




MNIST - PCA
8
6
4
2
0
−2
−4
−6
−8
−4
−2
0
2
4
6
8
MNIST - TSNE
30
20
10
0
−10
−20
−30
−40
−40
−30
−20
−10
0
10
20
30
40
9
8
7
6
5
4
3
2
1
0

LOW DIMENSIONAL EMBEDDINGS

EMBEDDINGS

- A recent usage of low dimensional embeddings is to encode complex objects as vectors
  - Words as Vector => Word2Vec
  - Nodes (of graph) as Vectors => Node2Vec
  - Documents as Vectors => Doc2Vec
  - ….

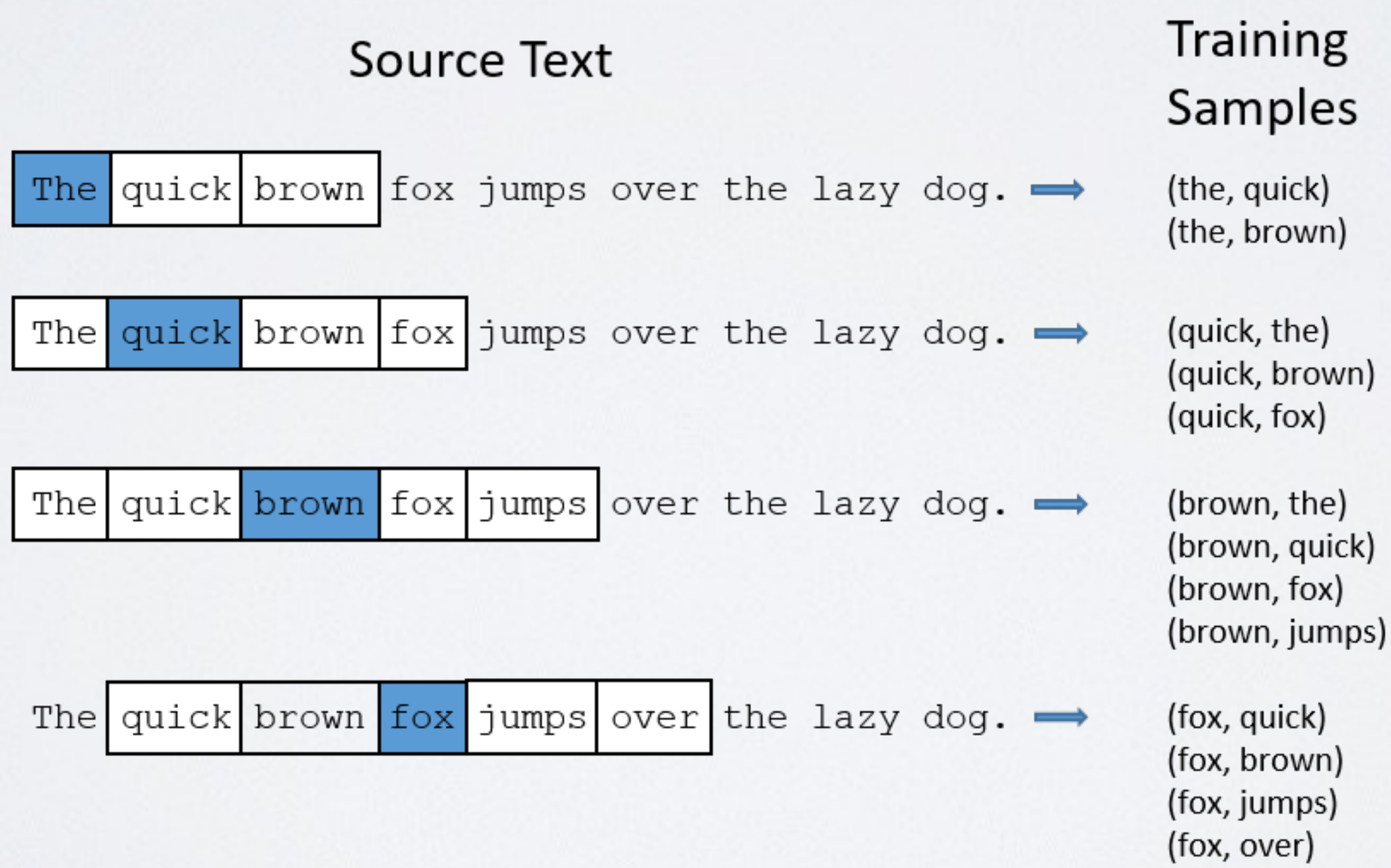
WORD EMBEDDING

WORD EMBEDDING

- Words can be understood as a (very) high dimensional space
  - Using One Hot encoding: vocabulary of 1000 words=>1000 columns
- Could we assign a vector in "low dimension", encoding the "semantic" of a word?
  - Two words with similar meanings should be close

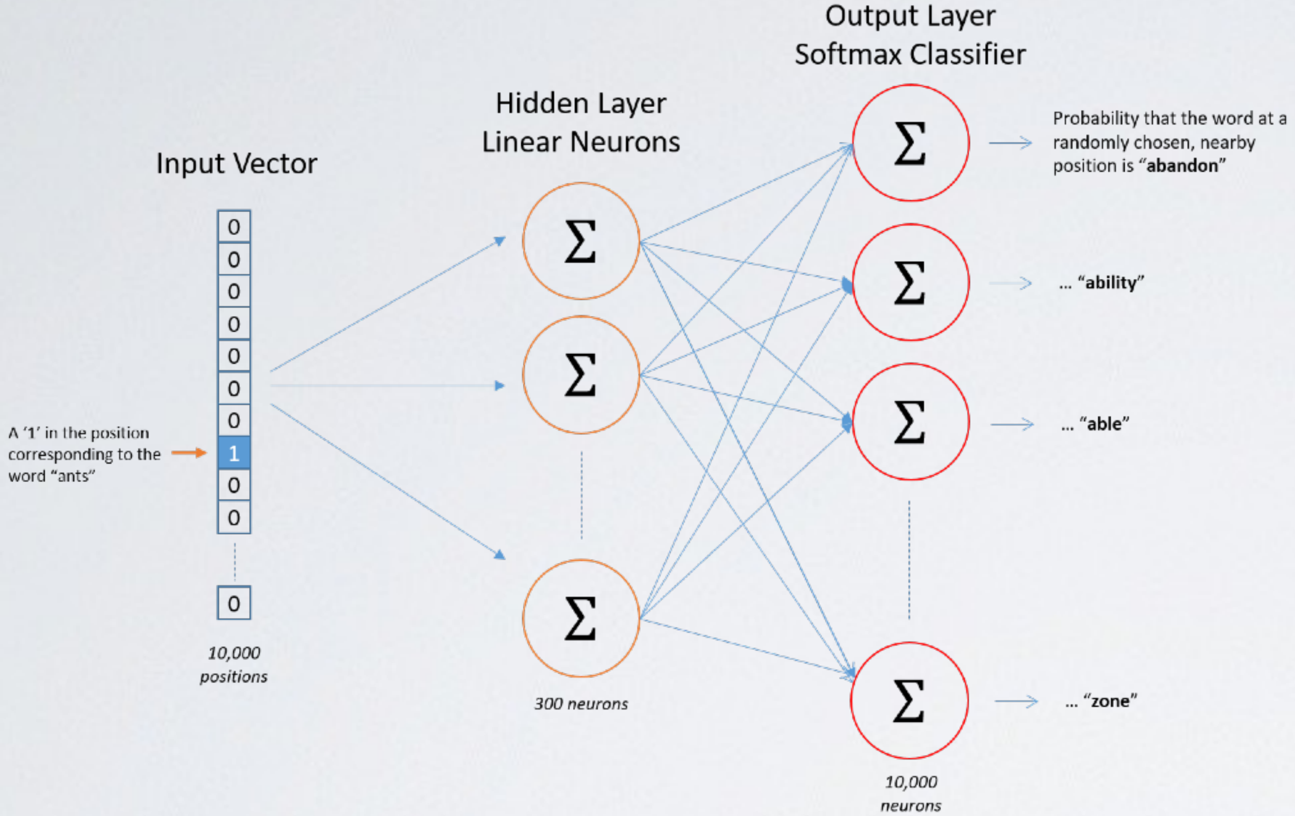
SKIPGRAM

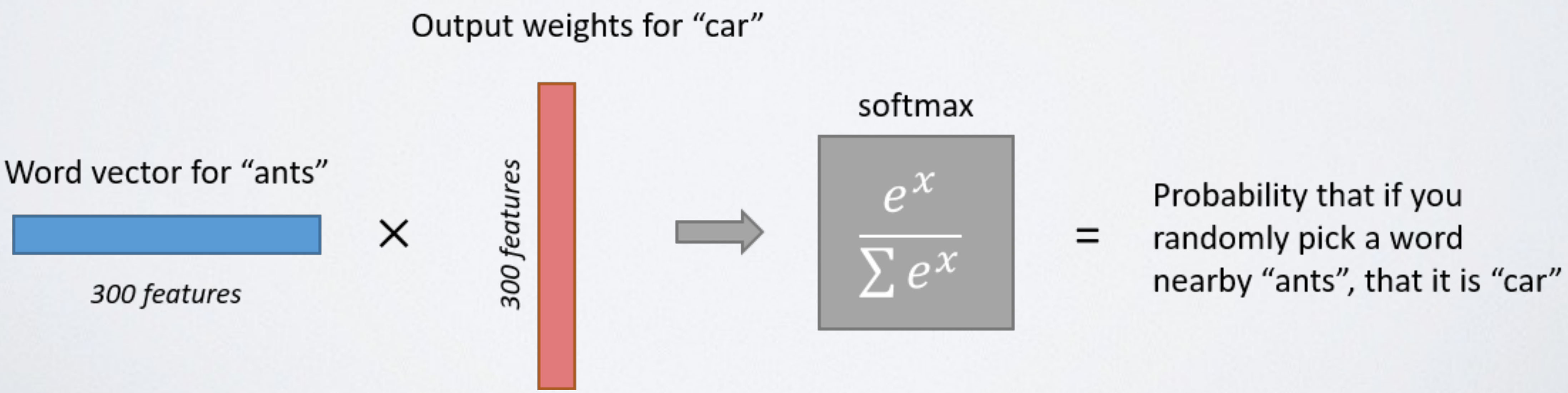
Word embedding
Corpus => Word = vectors
Similar embedding= similar **context**



[http://mccormickml.com/2016/04/19/word2vec-tutorial-the-skip-gram-model/]

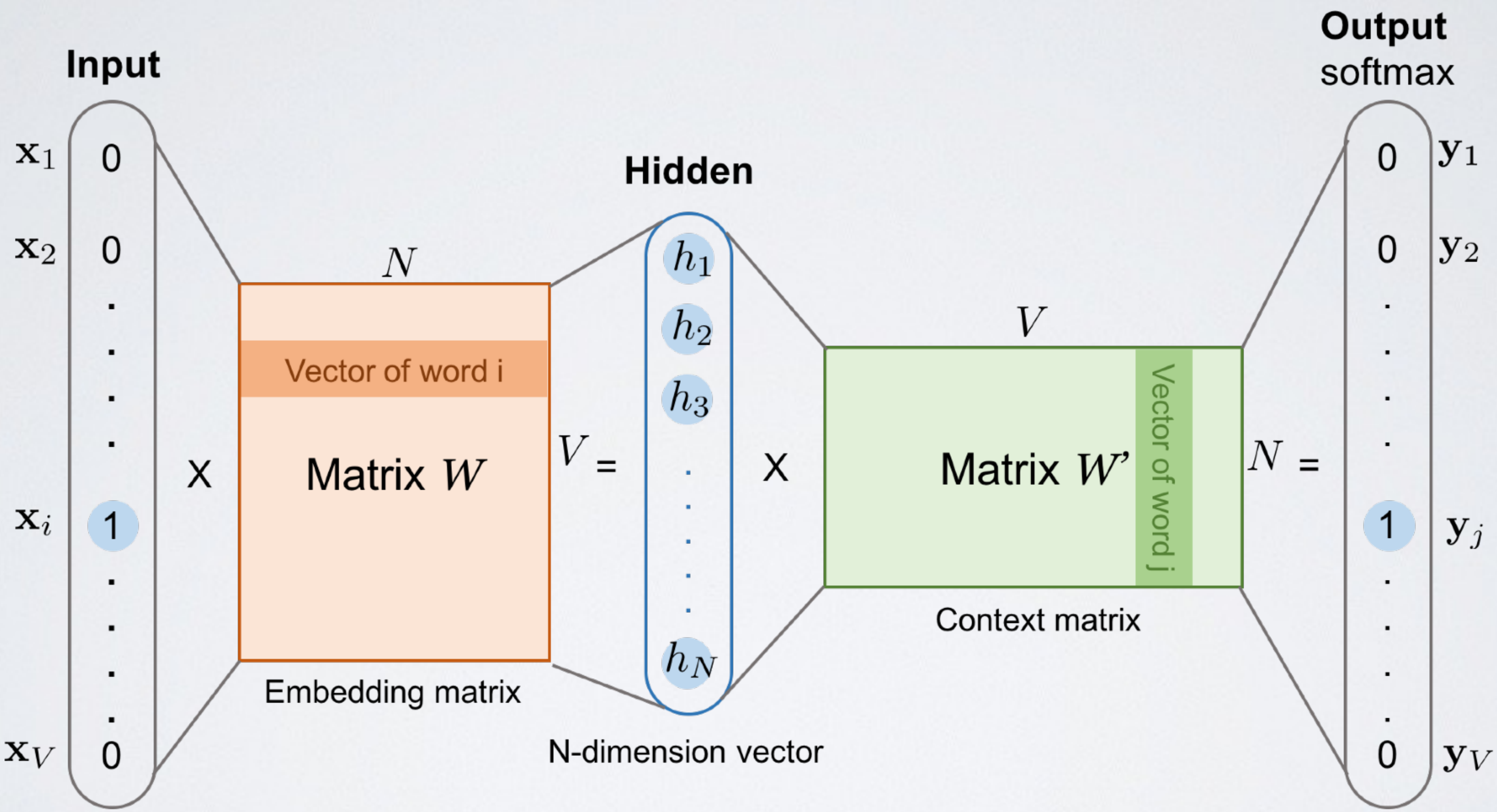
SKIPGRAM





https://towardsdatascience.com/word2vec-skip-gram-model-part-1-intuition-78614e4d6e0b

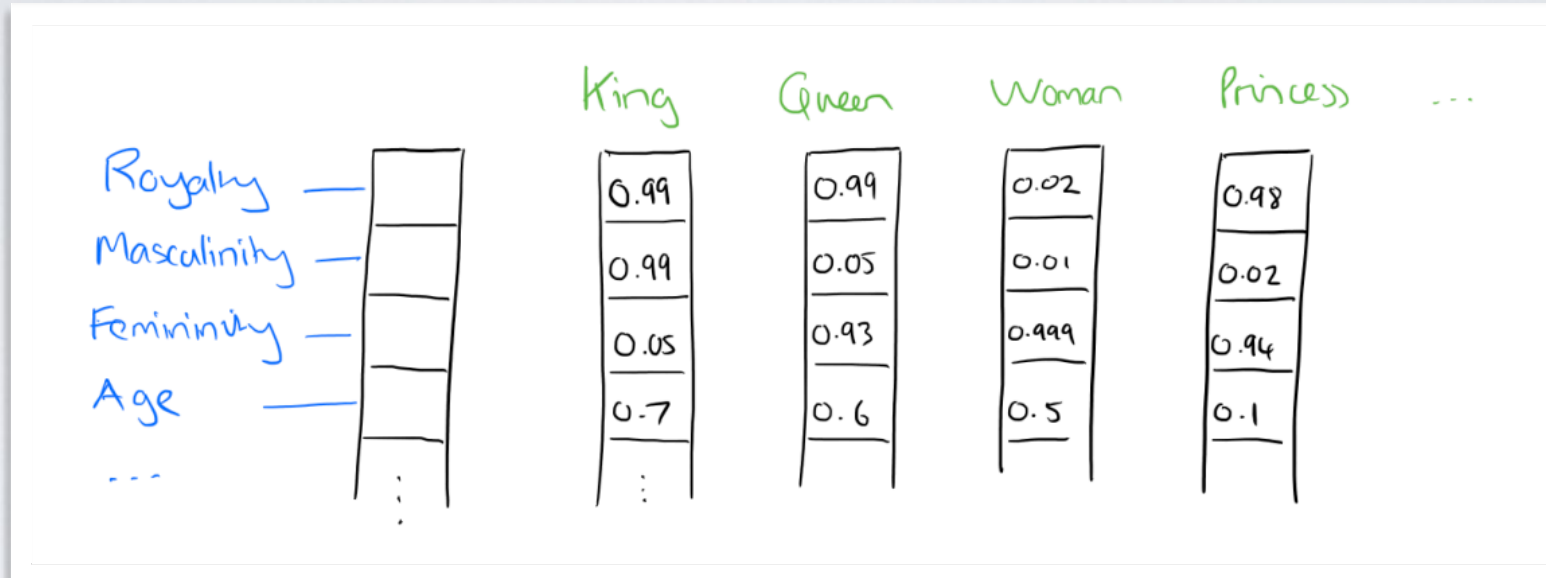
SKIPGRAM

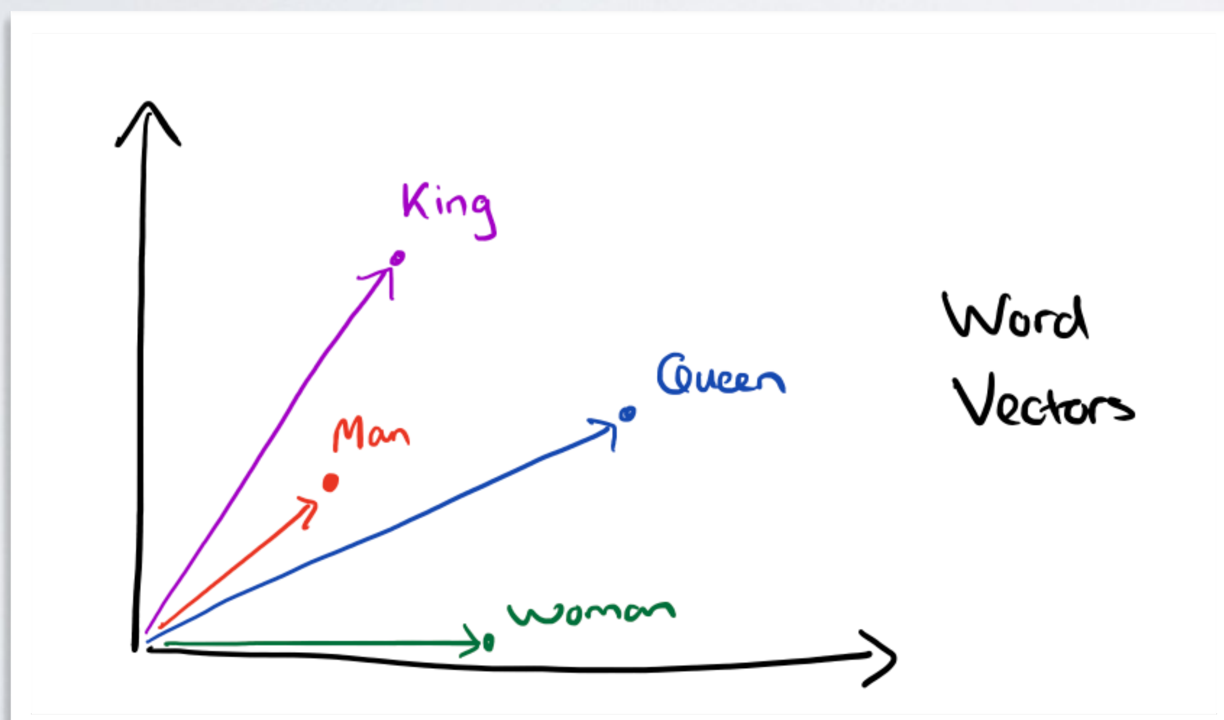


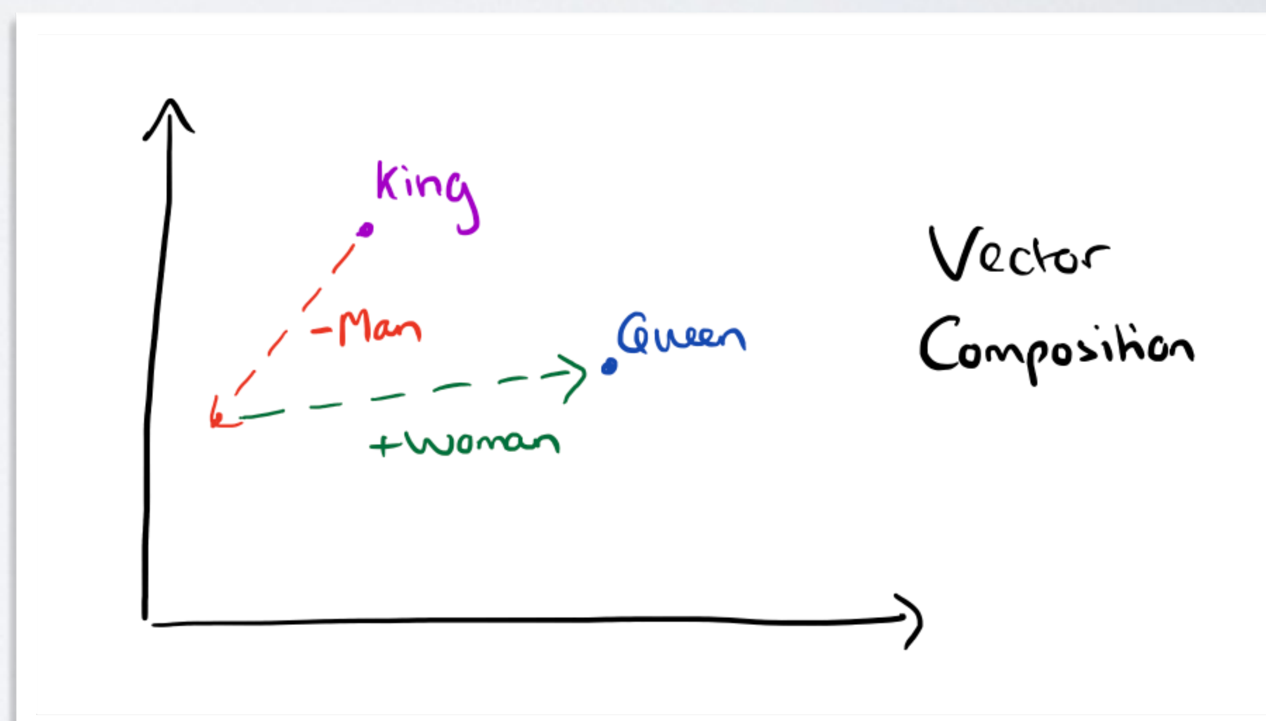
N=embedding size. **V**=vocabulary size

https://towardsdatascience.com/word2vec-skip-gram-model-part-1-intuition-78614e4d6e0b

SKIPGRAM







[https://blog.acolyer.org/2016/04/21/the-amazing-power-of-word-vectors/]

SKIPGRAM

Table 8: *Examples of the word pair relationships, using the best word vectors from Table 4 (Skip-gram model trained on 783M words with 300 dimensionality).*

Relationship	Example 1	Example 2	Example 3
France - Paris	Italy: Rome	Japan: Tokyo	Florida: Tallahassee
big - bigger	small: larger	cold: colder	quick: quicker
Miami - Florida	Baltimore: Maryland	Dallas: Texas	Kona: Hawaii
Einstein - scientist	Messi: midfielder	Mozart: violinist	Picasso: painter
Sarkozy - France	Berlusconi: Italy	Merkel: Germany	Koizumi: Japan
copper - Cu	zinc: Zn	gold: Au	uranium: plutonium
Berlusconi - Silvio	Sarkozy: Nicolas	Putin: Medvedev	Obama: Barack
Microsoft - Windows	Google: Android	IBM: Linux	Apple: iPhone
Microsoft - Ballmer	Google: Yahoo	IBM: McNealy	Apple: Jobs
Japan - sushi	Germany: bratwurst	France: tapas	USA: pizza

[https://blog.acolyer.org/2016/04/21/the-amazing-power-of-word-vectors/]

PRE-TRAINED

- One can train word2vec on their own dataset, but it needs to be large enough (and is costly)
  - https://radimrehurek.com/gensim/models/word2vec.html
- You can use pre-trained embeddings, trained on very large corpus (Twitter, Wikipedia…)
  - e.g., Glove: https://nlp.stanford.edu/projects/glove/

USAGE

- Single words=> Use directly vectors
- Short texts=> Weighted average vectors (more weights to more important words, e.g., rare words: TF-IDF…)
- Long texts=> More tricky. Needs BERT/LLM

USAGE

- Parameters:
  - Embedding dimensions *d*
  - Context size

EXTENSIONS

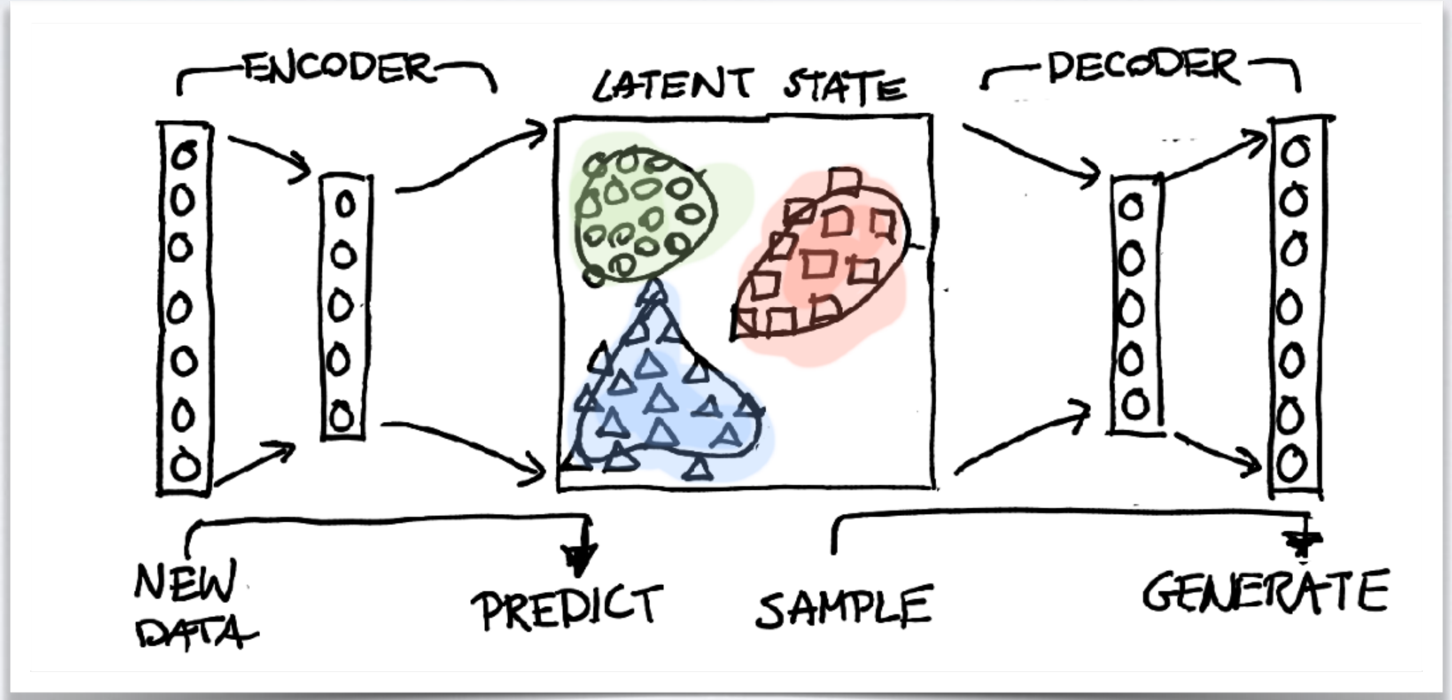
- Note: LLM works in a similar way, but:
  - using a deep, transformer architecture instead of a single-layer
- LLM also provide contextual embeddings
  - The embedding of a word is different based on the sentence.

DEEP LEARNING AND EMBEDDINGS

SHALLOW TO DEEP

- Deep neural networks are also commonly used to produce complex data embedding
  - Skipgram/Word2Vec is just particular cases of a general principle
- After each layer of a DNN, items are represented as vectors
  - Usually, at some steps, those layers are low-dimensional
  - Often, the last step or the middle step
  - These can be used as embedding for other tasks

SHALLOW TO DEEP



APPLICATIONS

- Image modification: modify some values of the embedding of an object (image, music, graph…) to reconstruct a slightly different version of it

- Clustering
  - Train a DNN on image classification task, then use clustering on the embeddings to discover similar images

- Visualization
  - Using Tsne on an embedding, we can have a global view of the organization of our data
    - Music, photos, graphs, books…

GRAPH EMBEDDING

GENERIC "SKIPGRAM"

- Algorithm that takes an input:
  - The element to embed
  - A list of "context" elements
- Provide as output:
  - An embedding with interesting properties
    - Works well for machine learning
    - Similar elements are close in the embedding
    - Somewhat preserves the overall structure

DEEPWALK

- Skipgram for graphs:
  - 1)Generate "sentences" using random walks
  - 2)Apply Skipgram
- Parameters:
  - Same as Skipgram
    - Embedding dimensions *d*
    - Context size
  - Parameters for "sentence" generation: length of random walks, number of walks starting from each node, etc.

Perozzi, B., Al-Rfou, R., & Skiena, S. (2014, August). Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 701-710). ACM.

NODE2VEC

- Use biased random walk to tune the context to capture *what we want*
  - "Breadth first" like RW => local neighborhood (edge probability ?)
  - "Depth-first" like RW => global structure ? (Communities ?)
  - 2 parameters to tune:
    - **p**: bias towards revisiting the previous node
    - **q**: bias towards exploring undiscovered parts of the network

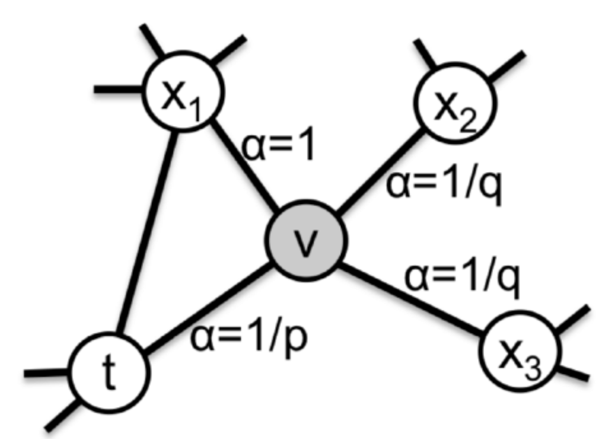


Figure 2: Illustration of the random walk procedure in *node2vec*. The walk just transitioned from t to v and is now evaluating its next step out of node v. Edge labels indicate search biases α.

Grover, A., & Leskovec, J. (2016, August). node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 855-864). ACM.

EMBEDDING ROLES

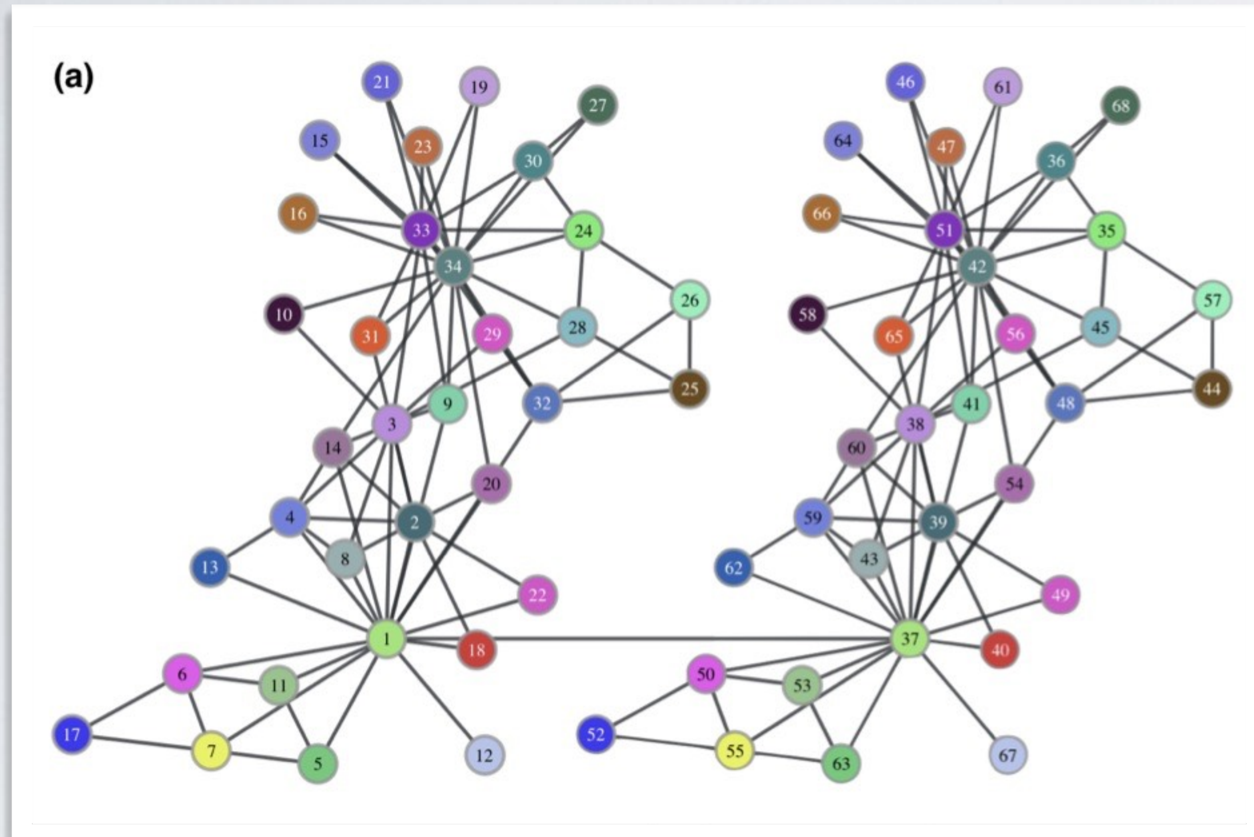
STRUC2VEC/ROLE2VEC

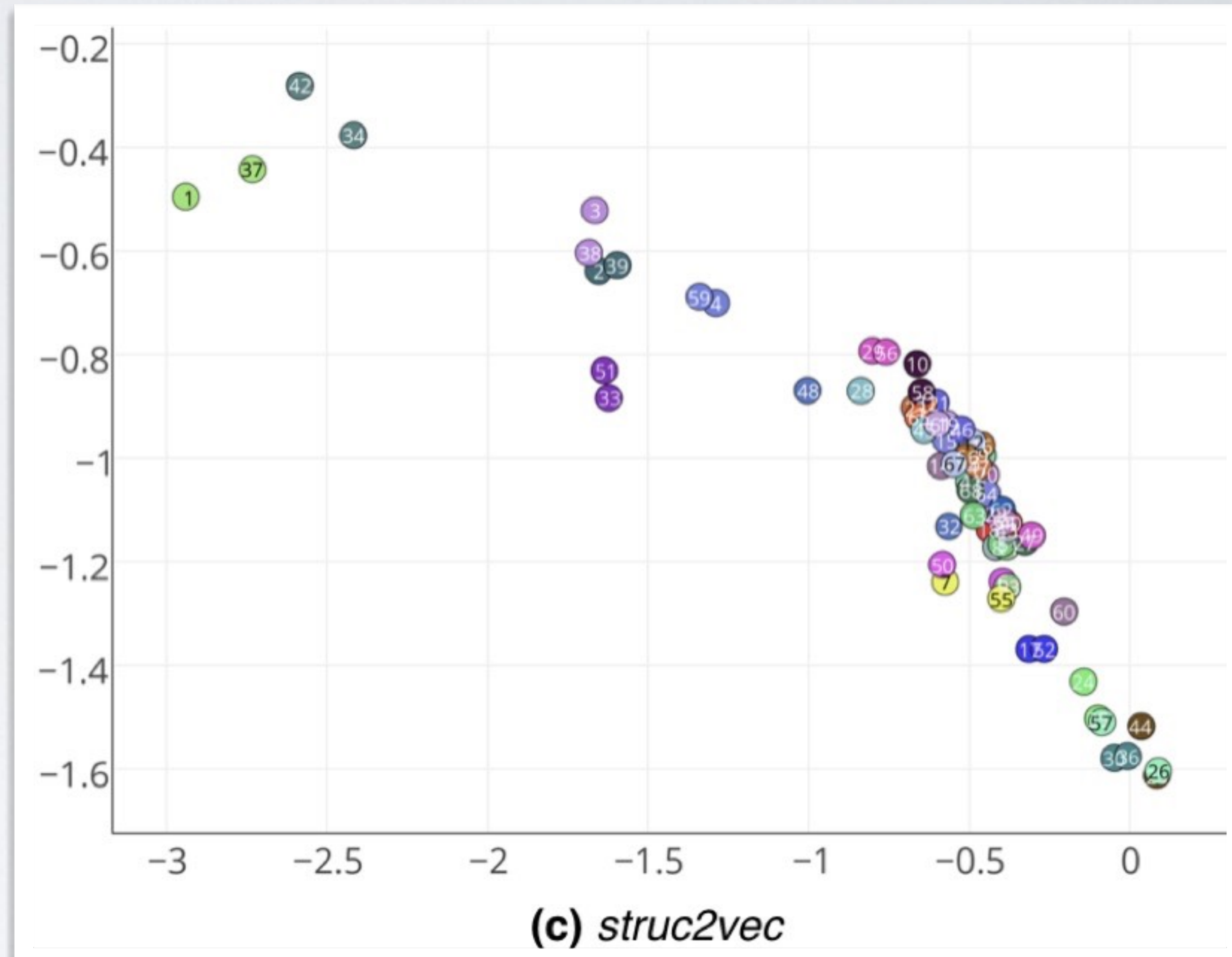
(Intuition)

- In node2vec/Deepwalk, the context collected by RW contains the **labels** of encountered nodes
- Instead, we could memorize the **properties** of the nodes: attributes if available, or computed attributes (degrees, CC, …)
- =>Nodes with a same context will be nodes in a same “position” in the graph
- =>Capture the role of nodes instead of proximity

Ribeiro, L. F., Saverese, P. H., & Figueiredo, D. R. (2017, August). struc2vec: Learning node representations from structural identity. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 385-394). ACM.

STRUCT2VEC : DOUBLE ZKC





Ribeiro, L. F., Saverese, P. H., & Figueiredo, D. R. (2017, August). struc2vec: Learning node representations from structural identity. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 385-394). ACM.