Week 11 Interactive Session

INF3050/INF4050
2021

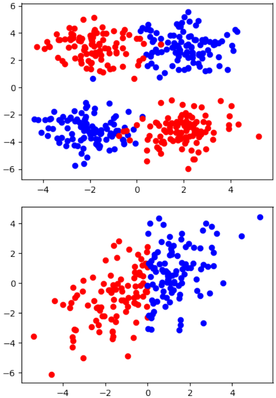
1. Summary of Week 11
2. Quiz
3. Demos
4. Questions?

1. Summary of Week 11

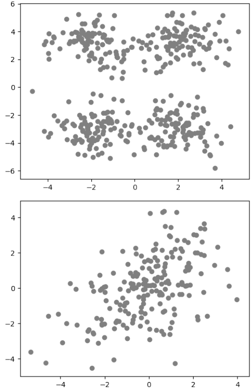
Unsupervised learning

Unsupervised learning is concerned with learning when no labels are given.

Supervised setting



Unsupervised setting



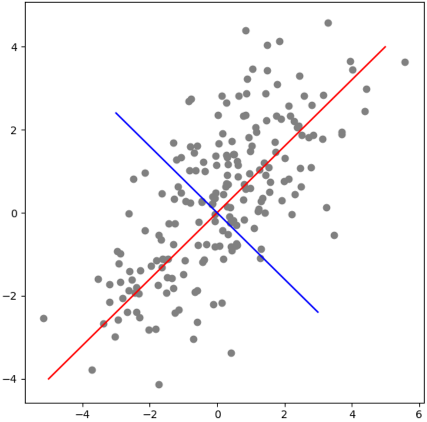
Key Concepts about Unsupervised learning

1. We want to transform data into **new representations**
   (e.g.: we want to project 256D data in 20D)
2. We try to preserve some **relevant structures** in the data
   (e.g.: we want to keep clusters of data)
3. We make **assumptions** on what is relevant
   (e.g.: Euclidean distance captures similarity)

Types of Unsupervised Learning

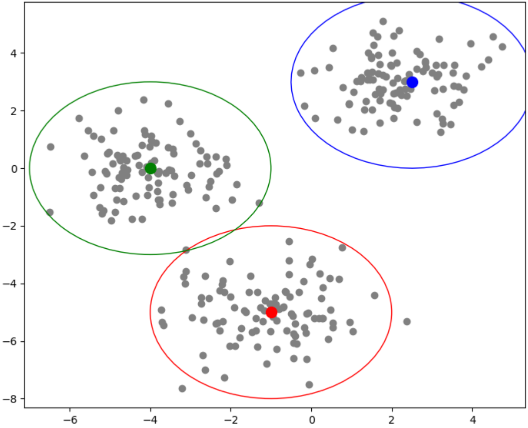
- *Clustering*
- *Dimensionality reduction / visualization*
- *Dimensionality reduction / manifold learning*
- *Dimensionality reduction / compression*
- *Anomaly detection*
- *Generative models*

PCA



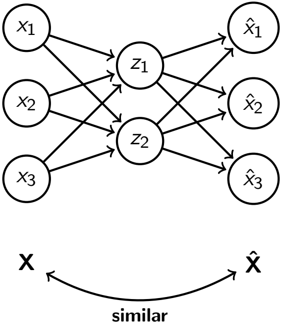
PCA projects data on the axes of maximal variance.

K-means



K-means finds clusters in the data through successive iterations.

Autoencoder



Autoencoders learn representations by compressng and reconstructing inputs.

2. Quiz

Question 1

Why is unsupervised learning important?

1. Data are often unlabelled
2. Labels are unreliable
3. Labels are costly to acquire
4. Labels are hard to agree upon
5. Updating labels is difficult
6. Humans learn without labels
7. Human labels are relative
8. ....

Bottom line: this is more an open opinion poll with no single right answer. (Ideally it invites students to think about why they do unsupervised learning. I do not know if Mentimeter supports it, though.)

Question 2

You are given a large data set of network data. The network administrator suspects that the data set contains samples of intrusion detection. What sort of UL algorithm would best fit your problem?

1. A clustering algorithm
2. A visualization algorithm
3. An anomaly detection algorithm
4. A generative model

Bottom line: ideal answer is 3, under the assumption of similarity/dissimilarity in a suitable space. Other algorithms *may* allow one to solve the problem, although possibly in an indirect way.

Question 3

You are given a large data set of unlabelled music tracks. Given a song, you would like to recommend a song of a similar genre. What sort of UL algorithm would best fit your problem?

1. A compresssion algorithm
2. A clustering algorithm
3. A visualization algorithm
4. A generative model

Bottom line: ideal answer is 2, under the assumption of closeness in a suitable space. Other algorithms *may* allow one to solve the problem, although possibly in an indirect way.

Question 4

You are given a large data set of unlabelled digits (from 0 to 9) as images. You want to cluster the images by their value (e.g.: all 1s in a cluster, all 2s in another cluster...). Your algorithm clusters images by similarity. Which similarity property would you like NOT to hold?

1. Background change
2. Rigid translation
3. Rotation
4. Color change

Bottom line: you would like an algorithm that capture similarity independently of rigid translations and changes in background or color, but in which similarity is not invariant to rotation, as a rotation may transform a 6 into 9. (Other invariances may be considered.)

Question 5

You are given a large data set of unlabelled black-and-white images. Each image is represented by a real-value for each pixel. You want to cluster the images by similarity. Does it make sense to evaluate similarity between images as a difference between corresponding pixels (0 being perfect similarity)?

1. Yes
2. No

Bottom line: no, such a measure of similarity may return a high value (low similarity) for images that are just slightly translated.

Question 6

You are given a set of unlabelled recordings of telephone calls to a customer service number. Recordings contain information such as pitch, tone, or other vocal features of the caller. You want to separate customers in groups: angry, frustrated, happy, and neutral. Which algorithm would you consider?

1. PCA
2. K-means
3. Autoencoder

Bottom line: K-means, as this sounds like a clustering task.

Question 7

You download an autoencoder pre-trained to process images. You want to use it to generate representations of pictures of animals. Would you expect that the generated representations would be lossless representations of your data?

1. Yes
2. No

Bottom line: no, in general autoencoders learn compressed lossy representations.

3. Demos

PCA for genetic data

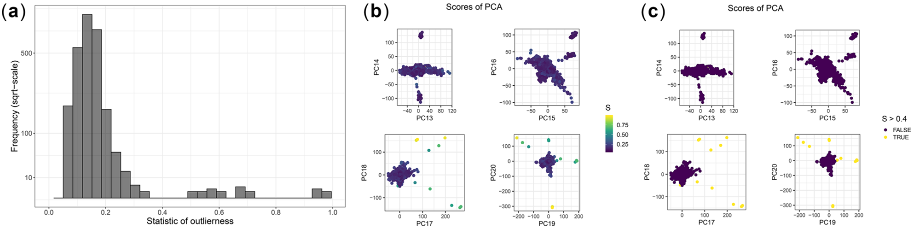
PCA may look like a simple algorithm, but it has important applications.

PCA turns out to be useful when you have *very high-dimensional data* and *few datapoints*, in order to find a few meaningful dimensions.

In **genetic studies** you may have genetic profiles composed of many genes from few individuals.

Demo

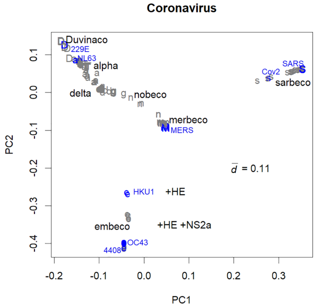
Principal component analysis (PCA) has been widely used in genetics for many years and in many contexts. [...] However, conducting PCA analyses can be complicated and has several potential pitfalls.



Paper (with data): Priv, Florian, et al. "Efficient toolkit implementing best practices for principal component analysis of population genetic data." Bioinformatics 36.16 (2020): 4449-4457.

Demo

All nucleotide sequences were analysed [...] Aligned data [...] were further processed to observe the relationships among virus samples by using the direct PCA method [...] The coronaviruses separated into distinct classes.



Paper (with data and R code): Konishi, Tomokazu. "Principal component analysis of coronaviruses reveals their diversity and seasonal and pandemic potential." PloS one 15.12 (2020): e0242954.

Word embedding

Words are notorious examples of *discrete data*.

Many learning algorithms exploit *continuous spaces* in which you can define distances, interpolate points, or compute gradients.

Word embedding allows one to learn in an unsupervised way a continuous representation of words by forcing co-occurring words to have small distances.

Word embedding

Word2vec

- *Demo:* `http://projector.tensorflow.org/`
- *Explanation:* `https://www.tensorflow.org/tutorials/text/word_embeddings`
- *Explanation:* `https://en.wikipedia.org/wiki/Word2vec`
- *Source:* `https://github.com/tensorflow/tensorflow/blob/r1.1/tensorflow/examples/tutorials/word2vec/word2vec_basic.py`

Text generation/prediction

To generate meaningful sentences you need a *model* that produces one word after another word.

There is no clearly defined label for when a word is correct or wrong.

Learn a simple model for text by computing probability distributions over *sequences of words*.

Text generation/prediction

n-grams model

- *Demo:* https://books.google.com/ngrams
- *Explanation:* https://books.google.com/ngrams/info
- *Explanation:* https://en.wikipedia.org/wiki/N-gram
- *Framework:* https://www.nltk.org/api/nltk.html
- *Source:*
  http://www.nltk.org/_modules/nltk/model/ngram.html

Text generation/prediction

GPT-2

- *Demo:* https://transformer.huggingface.co/doc/distil-gpt2
- *Explanation:* https://openai.com/blog/better-language-models/
- *Source:* https://github.com/huggingface/transformers

Other fun applications

- Image generation with generative adversarial networks or variational auto-encoders;
- Transform or color images;
- Code-to-code translation;
- Topic identification with Latent Dirichlet Allocation;
- Coresets for data reduction;
- Probabilistic data visualization with t-SNE;
- ...

4. Questions?