# IntroML Tutorial

## Clustering and Dimensionality Reduction

Vincent Fortuin (*fortuin@inf.ethz.ch*)

April 2019

Institute of Machine Learning, ETH Zürich

# Table of contents


1. Clustering
    - The k-means problem
    - Lloyd's heuristic
    - Issues and Caveats
2. Dimensionality Reduction
    - Use cases
    - Principal Component Analysis
    - Issues and Caveats
3. Connections between Clustering and Dimensionality Reduction
4. Exam Tasks 2016

# Clustering

# Clustering

- Let us have a data set $X = [x_1, \ldots, x_n]^\top \in \mathbb{R}^{n \times d}$
- The data points could be documents, articles, customers, patients, etc.
- We want to separate the data points into different groups (think of it as unsupervised classification)
- Those groups/clusters can be used for data exploration, assigning different treatments, or for compression

# Clustering

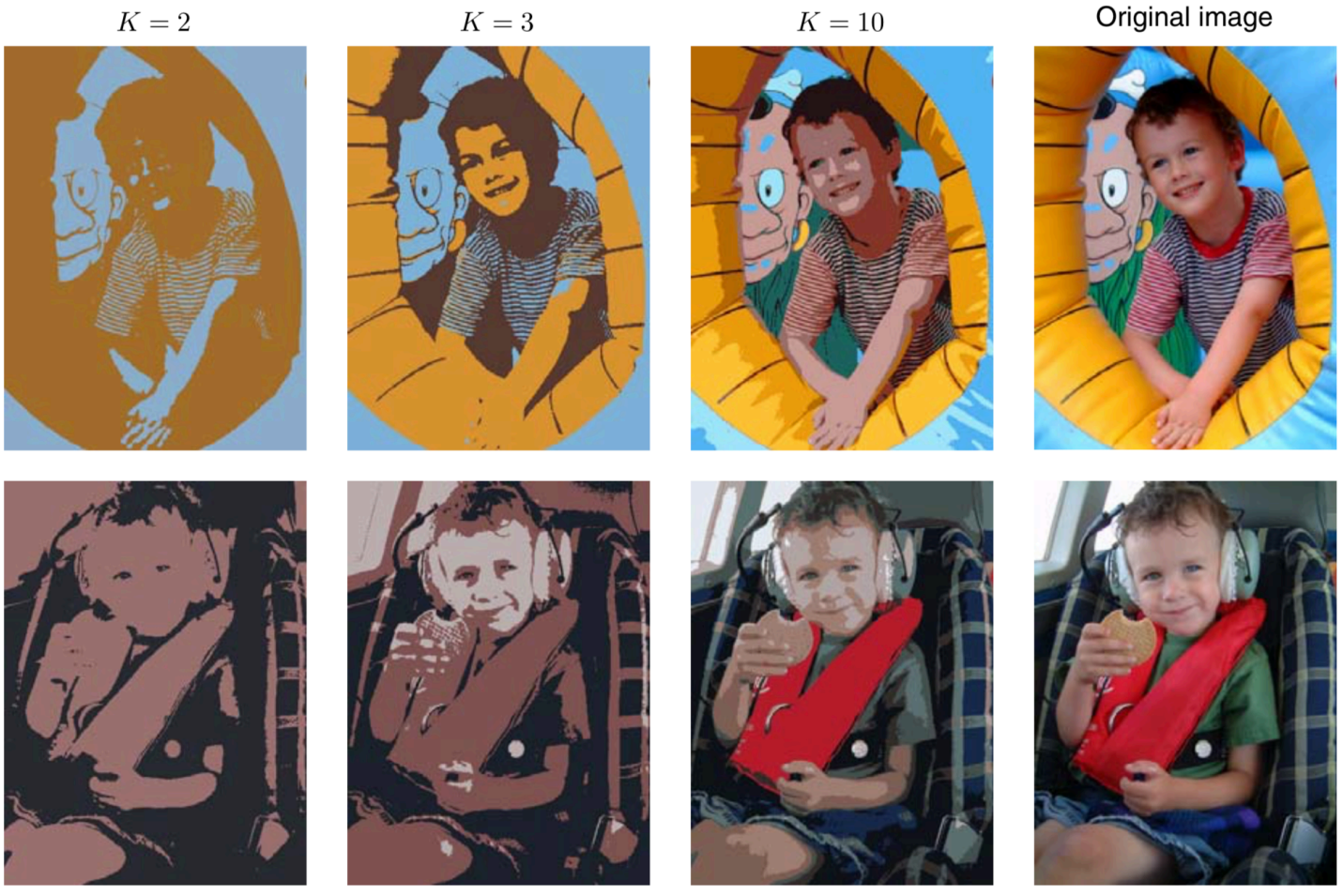



**Figure 1:** Clustering of colors in an image with different numbers of clusters.

Taken from Bishop 2006, *Pattern Recognition and Machine Learning*.

# The k-means problem

- We define the assignments of our data points to the clusters as additional variables $Z = [z_1, \ldots, z_n]^\top \in \{1, \ldots, k\}^n$
- Those variables are initially unobserved, i.e., latent
- We want to find values for those variables, such that the following objective is minimized:

$$\mathcal{L}(\mu, Z) := \sum_{i=1}^{n} \sum_{j=1}^{k} \delta(j, z_i) \|x_i - \mu_j\|_2^2$$

where $\mu = [\mu_1, \ldots, \mu_k]^\top \in \mathbb{R}^{k \times d}$ are the so-called cluster centroids

# Lloyd's heuristic

- Bad news: finding the global optimum of the k-means objective in $\mu$ and $Z$ is NP-hard
- Idea (Lloyd 1982): Let's optimize $\mu$ and $Z$ iteratively!

$$
\begin{aligned}
z_i^* &= \arg\min_{z_i} \mathcal{L}(\mu, Z) \\
&= \arg\min_{z_i} \sum_{j=1}^{k} \delta(j, z_i) \|x_i - \mu_j\|_2^2 \\
&= \arg\min_{j} \|x_i - \mu_j\|_2^2
\end{aligned}
$$

# Lloyd's heuristic

$$
\begin{aligned}
\mu_j^* &= \arg\min_{\mu_j} \mathcal{L}(\mu, Z) \\
&= \arg\min_{\mu_j} \sum_{i=1}^{n} \delta(j, z_i) \|x_i - \mu_j\|_2^2 \\
\frac{\partial}{\partial \mu_j} \mathcal{L}(\mu, Z) &= -2 \sum_{i=1}^{n} \delta(j, z_i)(x_i - \mu_j) = 0 \\
\iff \quad \sum_{i=1}^{n} \delta(j, z_i)\mu_j &= \sum_{i=1}^{n} \delta(j, z_i) x_i \\
\iff \quad \mu_j &= \frac{\sum_{i=1}^{n} \delta(j, z_i) x_i}{\sum_{i=1}^{n} \delta(j, z_i)} \\
\iff \quad \mu_j &= \frac{\sum_{i:z_i=j} x_i}{|\{i | z_i = j\}|}
\end{aligned}
$$

# Lloyd's heuristic

- This heuristic is guaranteed to converge ($\rightarrow$ homework), but only to a local optimum
- Which local optimum the algorithm converges to is sensitive to initialization ($\rightarrow$ homework)
- In practice, it is often helpful to run the optimization several times with different random initializations

# Smarter initialization: k-means++

- The `k-means++` algorithm (Arthur and Vassilvitskii 2007) is supposed to overcome the sensitivity to random initialization by initializing the cluster centroids in a smarter way
- It proceeds as follows:
  - Choose the first centroid $\mu_1$ uniformly at random from $X$
  - For each $x \in X$ compute $D(x) := \min_j \|x - \mu_j\|_2$
  - Sample the next $\mu_j$ from $X$ with probability $P(\mu_j = x) \propto D(x)^2$
  - Repeat the last two steps until $k$ centroids are chosen
- This can be shown to yield an $\mathcal{O}(\log k)$ approximation to the optimal centroids in expectation

# Choosing k

Before running the k-means algorithm, the number of clusters $k$ has to be chosen. This can be done according to

- prior knowledge
- some form of regularization (e.g., the “elbow criterion”)
- some information criterion (e.g., Akaike Information Criterion, Bayesian Information Criterion)

# Elbow criterion

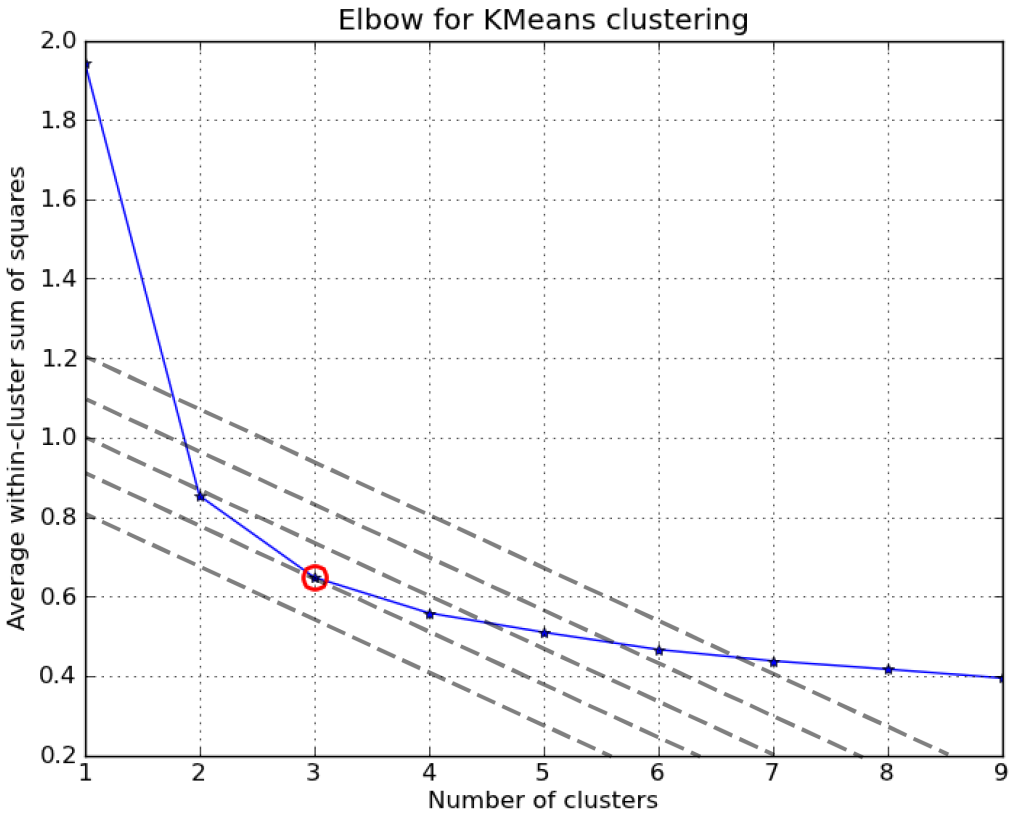



**Figure 2:** Loss curve for different values of k. The grey lines show points where $\mathcal{L}(\mu, Z) + \lambda k = \texttt{const.}$

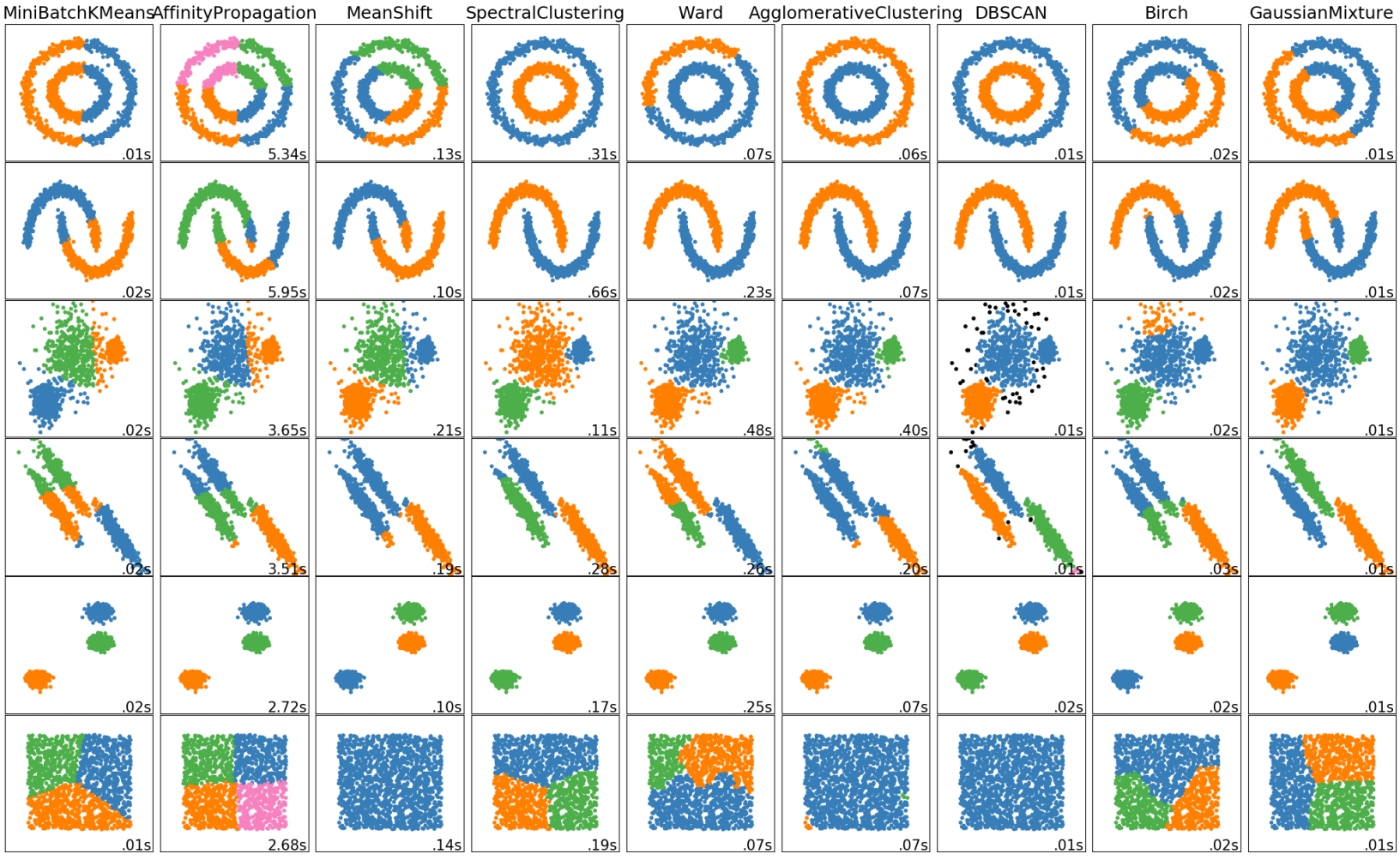
Adapted from `https://stackoverflow.com/questions/6645895/calculating-the-percentage-of-variance-measure-for-k-means`.

# Caveats

The k-means algorithm (especially using Lloyd's heuristic) comes with a few caveats:

- It does not guarantee to find the global optimum (actually, it can be arbitrarily bad)
- It is sensitive to initialization (although one can use multiple starts and k-means++)
- There is no principled way to choose $k$ (but several heuristics exist)
- There are no uncertainties about the cluster assignments (no probabilistic "soft assignments")

# Alternative clustering methods




Taken from `https://scikit-learn.org/stable/auto_examples/cluster/plot_cluster_comparison.html`.

# Dimensionality Reduction

# Dimensionality Reduction

- We want to take some high-dimensional data $X = [x_1, \ldots, x_n]^\top \in \mathbb{R}^{n \times d}$ and embed it into a lower-dimensional space $\mathbb{R}^k$ with $k \ll d$
- We can then use the representation of the data points $Z = [z_1, \ldots, z_n]^\top \in \mathbb{R}^{n \times k}$ for visualization, compression, or down-stream tasks (e.g., supervised learning)
- We believe that a lot of the information should be preserved due to the *manifold hypothesis*: real-world data lives on low-dimensional manifolds in high-dimensional spaces (e.g., images)
- We can preserve distances between points, if $k$ is of order $\mathcal{O}(\log d)$ (Johnson and Lindenstrauss 1984)

# Dimensionality reduction

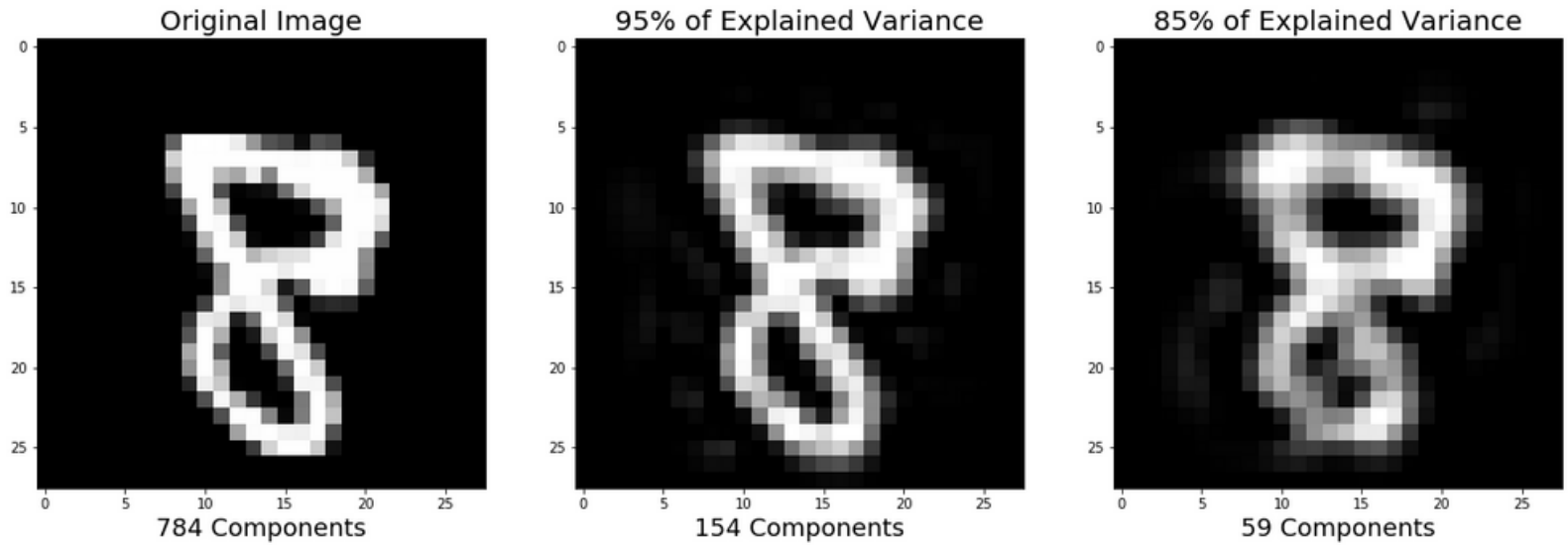



**Figure 3:** Reconstruction of MNIST digit with different numbers of principal components.

Taken from `https://towardsdatascience.com/pca-using-python-scikit-learn-e653f8989e60`.

# Principal Component Analysis

- PCA is a linear dimensionality reduction, i.e., $z_i = w^\top x_i$
- You have seen in the lecture that it minimizes the reconstruction error $\|wz_i - x_i\|_2^2$
- Let's see which $w$ we should choose, if instead we want to maximize the variance of the projected data

# Maximum variance projection

The projected variance is

$$\frac{1}{n}\sum_{i=1}^{n}\|w^\top x_i - w^\top \bar{x}\|_2^2 = w^\top \Sigma w \qquad [= Q_R(\Sigma, w)]$$

with the data mean $\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$

and data covariance $\Sigma = \frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})(x_i - \bar{x})^\top$

If $\bar{x} = 0$, then $\Sigma = X^\top X$

# Maximum variance projection

$$
\begin{aligned}
w^* &= \arg\max_w \mathcal{L}(w) \\
&= \arg\max_w w^\top \Sigma w + \lambda(1 - w^\top w) \\
\frac{\partial}{\partial w}\mathcal{L}(w) &= 2\Sigma w - 2\lambda w = 0 \\
\iff \quad \Sigma w = \lambda w &\implies w \text{ is eigenvector of } \Sigma \\
\iff \quad w^\top \Sigma w = \lambda &\implies \text{variance is eigenvalue of } \Sigma
\end{aligned}
$$

$\implies$ To maximize the projected variance $w^\top \Sigma w$, we just have to choose $w$ to be the eigenvector of $\Sigma$ with the largest eigenvalue $\lambda$. This is exactly the PCA solution!

# The k-dimensional case

$$
\begin{aligned}
W^* &= \arg\max_W \mathcal{L}(W) \\
&= \arg\max_W \operatorname{Tr}\left[W^\top \Sigma W + \Lambda(I - W^\top W)\right] \\
\frac{\partial}{\partial W}\mathcal{L}(W) &= 2\Sigma W - 2\Lambda W = 0 \\
\iff \quad \Sigma W &= \Lambda W \implies W \text{ are eigenvectors of } \Sigma \\
\iff \quad W^\top \Sigma W &= \Lambda \implies \text{variances are eigenvalues of } \Sigma
\end{aligned}
$$

# Maximum variance = minimum error?

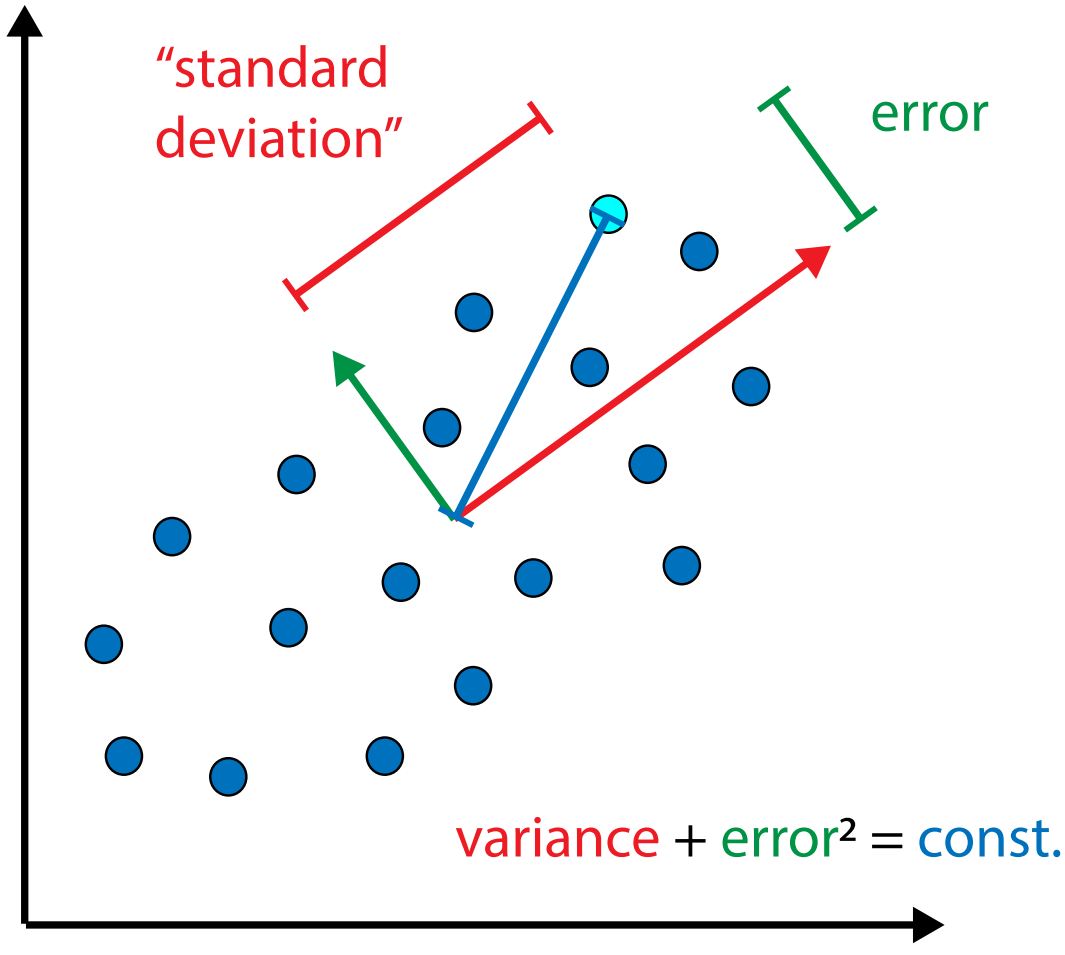



**Figure 4:** Visual illustration of the relationship between variance and error in Principal Component Analysis.

# Computing the principal components

We can compute the eigenvectors and eigenvalues of $\Sigma$ using an eigendecomposition:

$$\Sigma = W \Lambda W^\top$$

Recall that $\bar{x} = 0 \implies \Sigma = X^\top X$

We can then use a singular value decomposition (SVD) on $X$ directly:

$$X = USW^\top \implies \Sigma = X^\top X = WS^\top U^\top USW^\top = WS^2W^\top$$

# Caveats

As every method, PCA comes with a few caveats:

- It is linear (i.e., cannot model nonlinear manifold structures)
- It does not take supervised information into account (c.f. LDA)
- Computing via SVD assumes centering

There are a couple of alternative methods to overcome these issues:

- Autoencoders (allow nonlinear mappings, but expensive to train)
- t-SNE (nonlinear, but stochastic and parameter-dependent)
- Linear Discriminant Analysis (supervised)
- many others (MDS, UMAP, etc.)

# Connections between Clustering and Dimensionality Reduction

# PCA ≈ k-means?

PCA:

$$\mathcal{L}(W, Z) = \sum_{i=1}^{n} \|W z_i - x_i\|_2^2$$

with $W$ orthogonal and $z_i \in \mathbb{R}^k$

k-means:

$$\mathcal{L}(W, Z) = \sum_{i=1}^{n} \|W z_i - x_i\|_2^2$$

with $W$ arbitrary and $z_i \in E_k$ (unit vectors)

# k-means as approximate PCA

- One can show that PCA solves a relaxed version of the k-means problem (Ding and He 2004)
- The k-means cluster centroids therefore span a constrained approximation of the principal subspace
- The line between the centroids for $k = 2$ approximates the first PC, the plane between the centroids for $k = 3$ the first two PCs, etc.

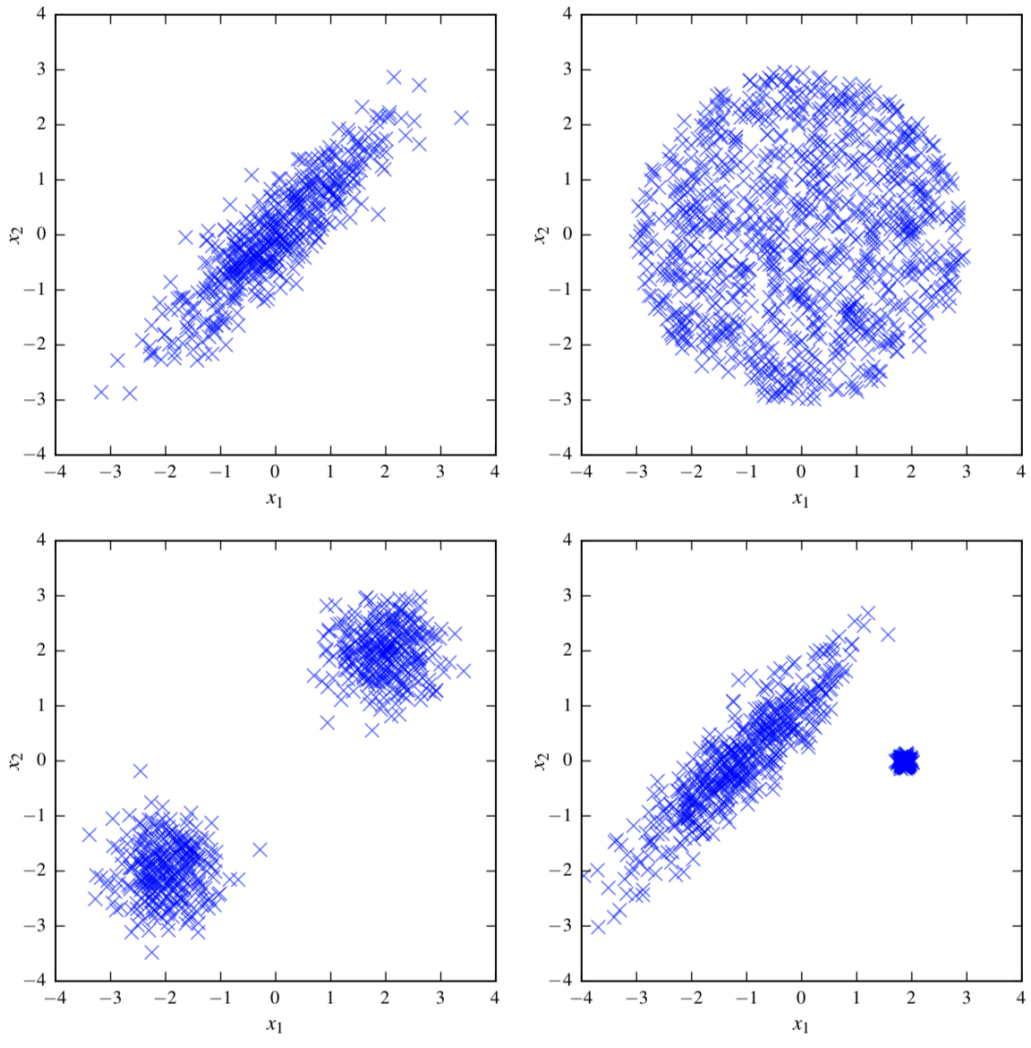
# First PCA, then k-means

- One can also show that one can speed up k-means by first using PCA (Cohen *et al.* 2015)
- Projecting the data onto the first $\mathcal{O}(\frac{k}{\epsilon})$ principal components and then using Lloyd's heuristic, yields a $(1+\epsilon)$-optimal solution

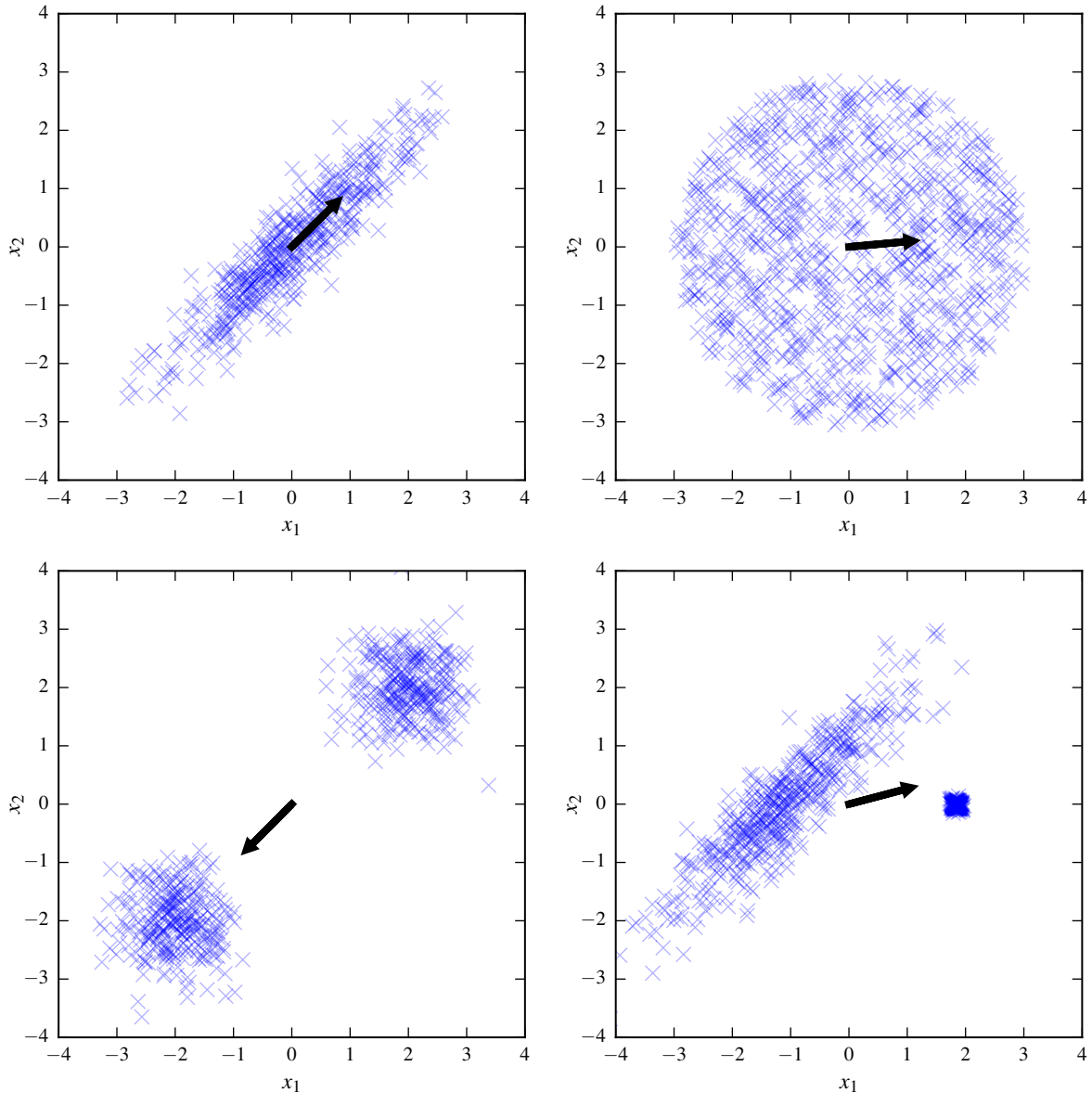
# Exam Tasks 2016

# PCA TASK

**(4 points)** (ii) In each figure below, draw the first principal component of the data. The data is centered. In the two figures at the bottom both clusters have the same number of data points.



# PCA TASK



# K-MEANS TASK 1

Assume we have three one-dimensional data points $x_1 = 0, x_2 = 2, x_3 = 3$, which we want to cluster using the $k$-means algorithm. Note that, if at any iteration no point is assigned to some center $\mu_j$, this center is not updated during that iteration.

**(4 points)** (i) For $k = 1$, compute the global minimizer of the $k$-means objective.

# K-MEANS TASK 1

Assume we have three one-dimensional data points $x_1 = 0, x_2 = 2, x_3 = 3$, which we want to cluster using the $k$-means algorithm. Note that, if at any iteration no point is assigned to some center $\mu_j$, this center is not updated during that iteration.

**(4 points)** (i) For $k = 1$, compute the global minimizer of the $k$-means objective.

$$\mu_1 = \frac{5}{3}$$

$$\forall i \in \{1, 2, 3\} : z_i = 1$$

# K-MEANS TASK 2

**(4 points)** (ii) For $k = 2$, if we initialize one cluster center as $\mu_1 = 0$ and the other as $\mu_2 = 10$, what will happen in the subsequent iterations of the $k$-means algorithm?

# K-MEANS TASK 2

**(4 points)** (ii) For $k = 2$, if we initialize one cluster center as $\mu_1 = 0$ and the other as $\mu_2 = 10$, what will happen in the subsequent iterations of the $k$-means algorithm?

Since $\mu_1$ is closer to all of the data points than $\mu_2$, we will converge to the same solution as in the previous task ($\mu_1 = \frac{5}{3}$, $\forall i \in \{1, 2, 3\} : z_i = 1$), while $\mu_2 = 10$ will not get updated.

# K-MEANS TASK 3

**(5 points)** (iii) For $k = 3$, what is a global minimizer of the $k$-means objective? Is there a local minimizer that is not global?

# K-MEANS TASK 3

**(5 points)** (iii) For $k = 3$, what is a global minimizer of the $k$-means objective? Is there a local minimizer that is not global?

The global minimizer is

$$\mu_1 = x_1 \qquad z_1 = 1$$
$$\mu_2 = x_2 \qquad z_2 = 2$$
$$\mu_3 = x_3 \qquad z_3 = 3$$

As seen in the previous task, there can be pathological initializations that lead to spurious local minima, such as

$$\mu_1 = \frac{5}{3}, \quad \mu_2 = 1000, \quad \mu_3 = 1001$$
$$\forall i \in \{1, 2, 3\} : z_i = 1$$

Questions?

# References


- Bishop, C. M. (2006). Pattern recognition and machine learning. Springer.
- Lloyd, S. (1982). Least squares quantization in PCM. IEEE transactions on information theory, 28(2), 129-137.
- Arthur, D., & Vassilvitskii, S. (2007, January). k-means++: The advantages of careful seeding. In Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms (pp. 1027-1035). Society for Industrial and Applied Mathematics.
- Johnson, W. B., & Lindenstrauss, J. (1984). Extensions of Lipschitz mappings into a Hilbert space. Contemporary mathematics, 26(189-206), 1.
- Ding, C., & He, X. (2004, July). K-means clustering via principal component analysis. In Proceedings of the twenty-first international conference on Machine learning (p. 29). ACM.
- Cohen, M. B., Elder, S., Musco, C., Musco, C., & Persu, M. (2015, June). Dimensionality reduction for k-means clustering and low rank approximation. In Proceedings of the forty-seventh annual ACM symposium on Theory of computing (pp. 163-172). ACM.