MATHEMATICS FOR MACHINE LEARNING

Nagoya University, Fall 2023

Lecture 12

Principal Component Analysis (PCA)

This week Tutorial: Friday 19th Dec. 6th period

https://www.henrikbachmann.com/mml2023.html

Semester project

Objective: Choose a project topic related to Nagoya or Japan more broadly that can be addressed using machine learning algorithms. Your task is to develop a machine learning model to solve a specific problem or provide insights into an aspect of life, business, environment, culture, etc., in Nagoya/Japan.

Group Size: 1-3 members

Code: Preferably a Google Colab notebook. Exceptions are possible; please provide full documentation for any different technology or package used. If you plan not to submit a Google Colab, please contact us in advance.

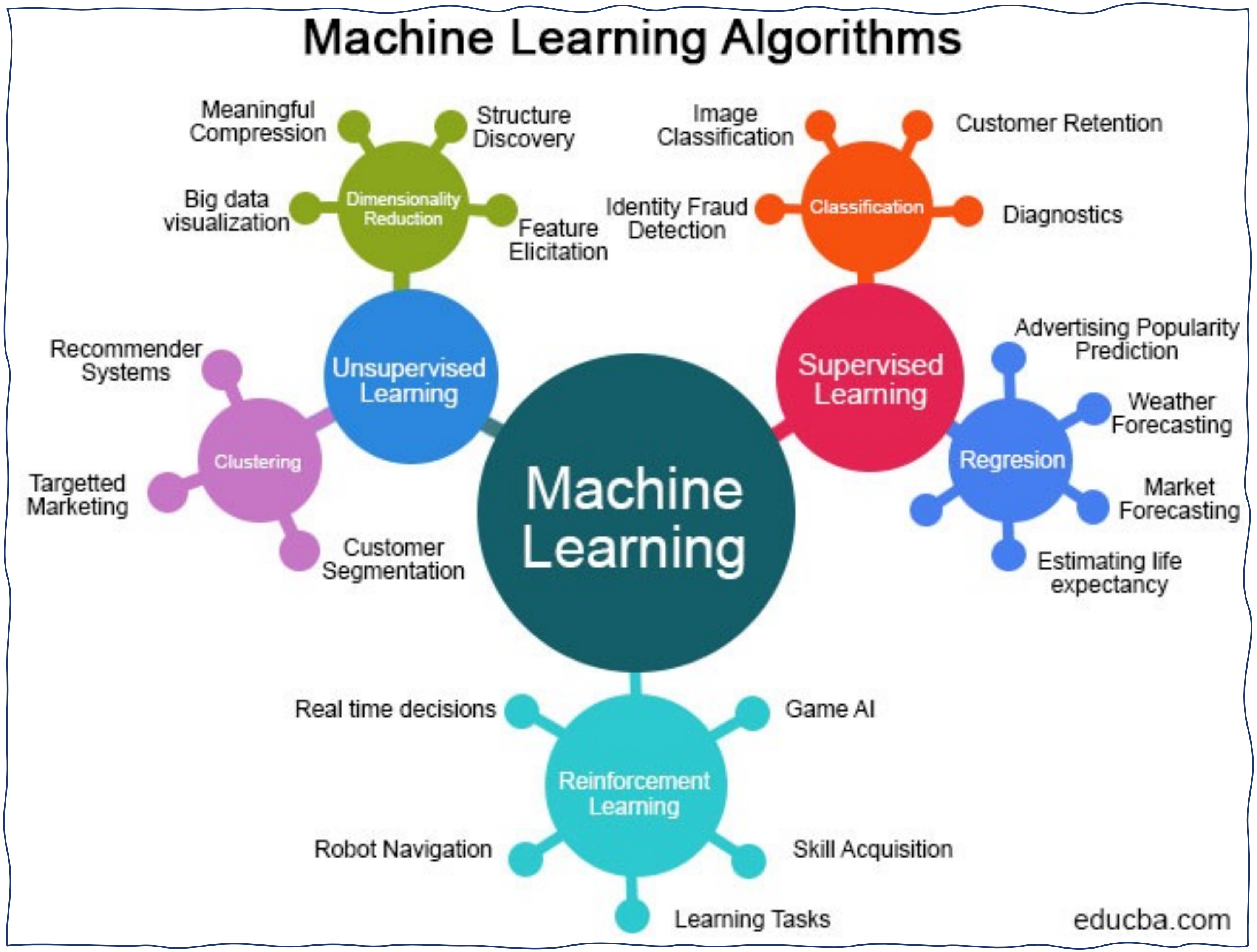
Documentation: 5-10 slides as if you were going to present the project.

Your slides should cover (for example):

- Problem Statement
- Data Collection
- Data Exploration and Visualization
- Model Building and Evaluation
- Conclusion

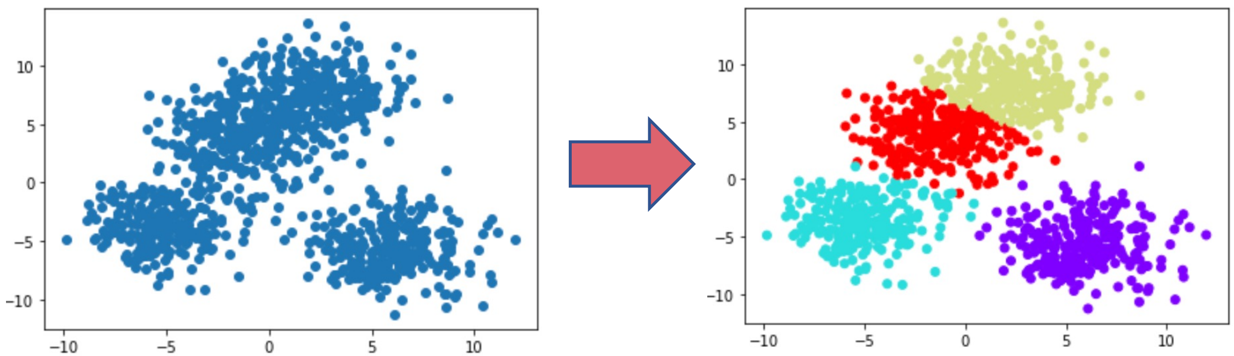
- **The slides are the documentation and should explain what you did.**
- **Do not just include examples and diagrams.**
- **It should be possible to understand your project by just reading the slides.**

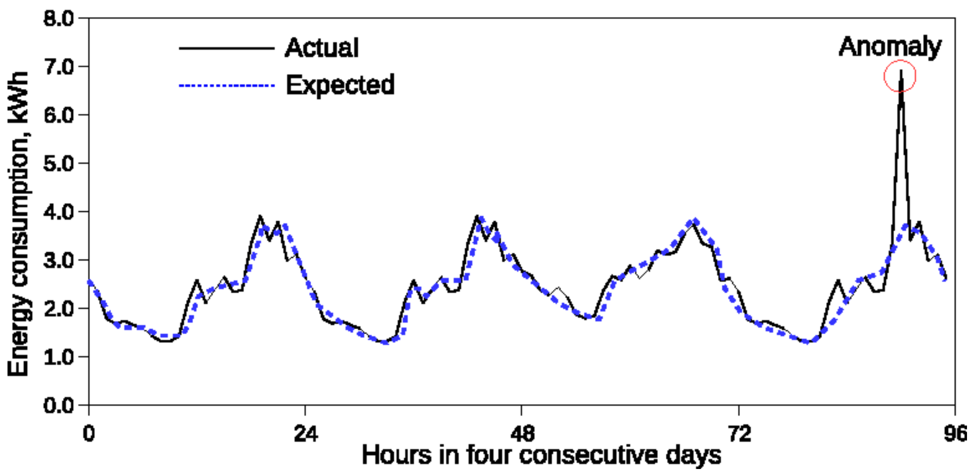
Overview

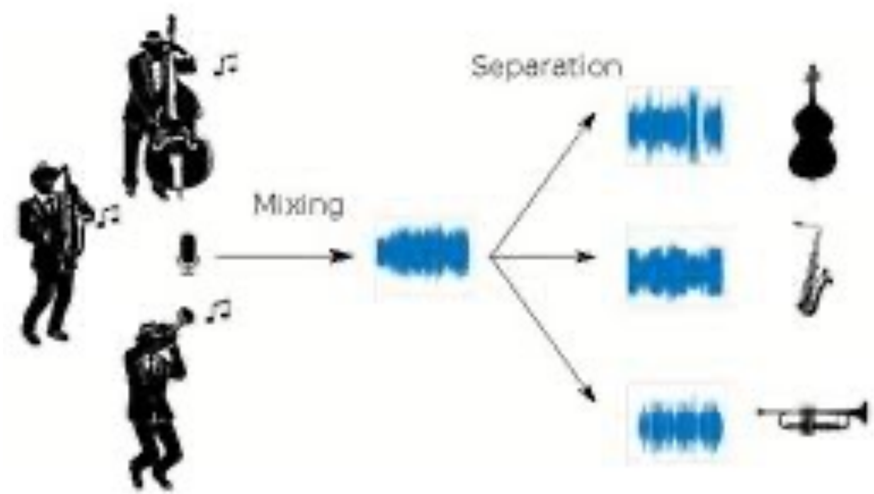

Machine Learning Algorithms
Machine Learning
Unsupervised Learning
Dimensionality Reduction
Meaningful Compression
Structure Discovery
Big data visualization
Feature Elicitation
Clustering
Recommender Systems
Targetted Marketing
Customer Segmentation
Supervised Learning
Classification
Image Classification
Customer Retention
Identity Fraud Detection
Diagnostics
Regresion
Advertising Popularity Prediction
Weather Forecasting
Market Forecasting
Estimating life expectancy
Reinforcement Learning
Real time decisions
Game AI
Robot Navigation
Skill Acquisition
Learning Tasks
educba.com

Unsupervised learning

- Clustering (Lecture 11)
- Anomaly detection
- Signal separation
- …



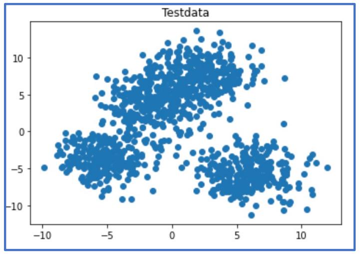


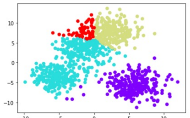


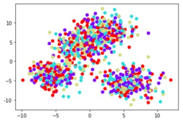
https://en.wikipedia.org/wiki/Signal_separation

https://en.wikipedia.org/wiki/Cocktail_party_effect

Lecture 11: k-means clustering







k-means algorithm

1. Initialize the means $\mu_1, \dots, \mu_k \in \mathbb{R}^d$ with some starting value.
   (a) **Forgy method:** Choose randomly k different numbers $\{s_1, \dots, s_k\} \subset \{1, \dots, n\}$ and set

   $$\mu_i = p_{s_i}$$

   for $i = 1, \dots, k$

   (b) **Random partition:** Choose $c : P \to \{1, \dots, k\}$ randomly and set for $i = 1, \dots, k$

   $$\mu_i = \frac{1}{|C_i|} \sum_{p \in C_i} p\,.$$

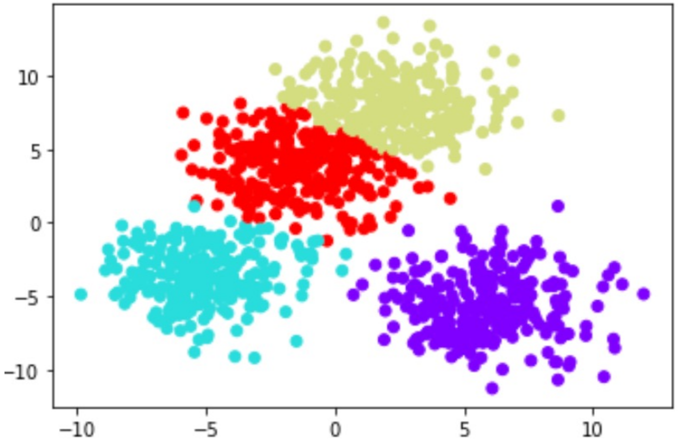
2. Define the clusters C_i for $i = 1, \dots, k$ by

   $$C_i = \{p \in P \mid \|p - \mu_i\| \leq \|p - \mu_j\| \text{ for } j = 1, \dots, k\}$$

3. Recalculate the means μ_i for $i = 1, \dots, k$ by

   $$\mu_i = \frac{1}{|C_i|} \sum_{p \in C_i} p\,.$$

4. Repeat with step 2.

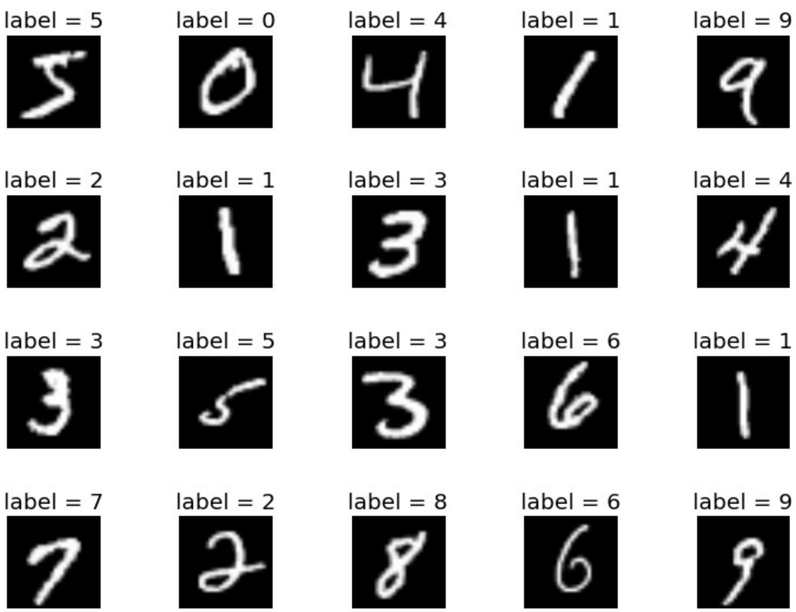


Result after 20 steps

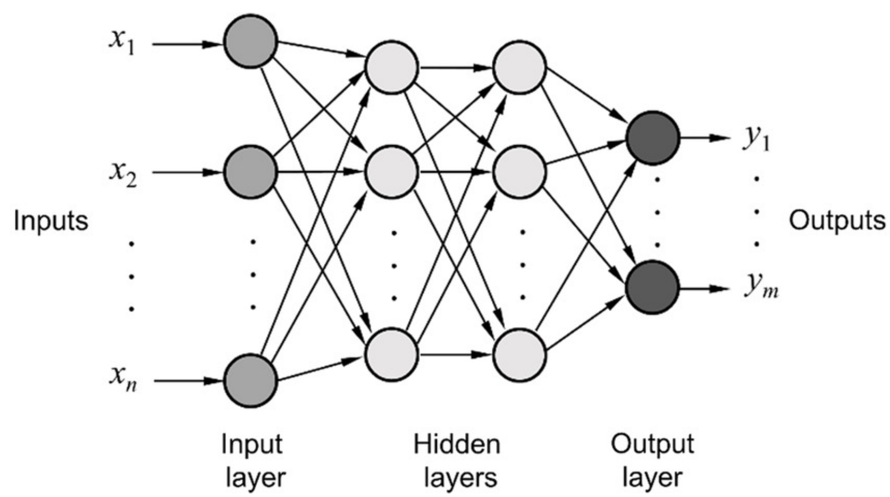
Unsupervised learning: Dimensionality reduction

Example: Digit recognition (MNIST Dataset)

Pictures of size 28 x 28 = 784 pixels



Neural network with input layer size 784

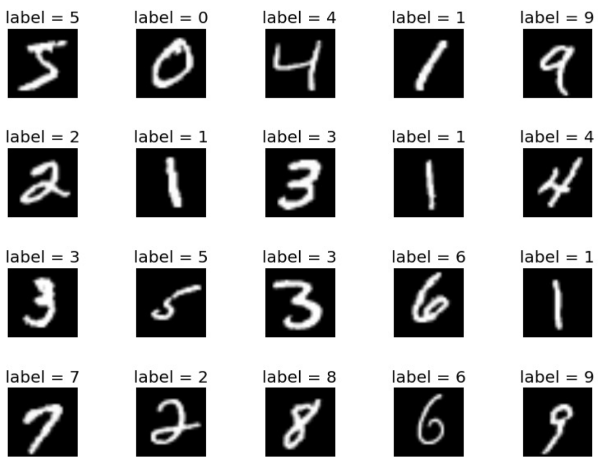


Question: How can we reduce the size of the input layer without losing a lot of information?

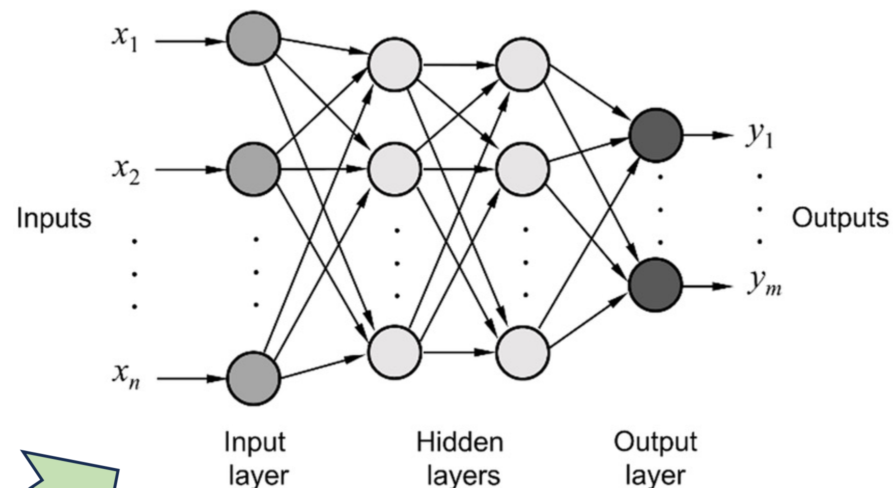
Unsupervised learning: Dimensionality reduction

Example: Digit recognition (MNIST Dataset)

Pictures of size 28 x 28 = 784 pixels



Neural network with input layer size 64



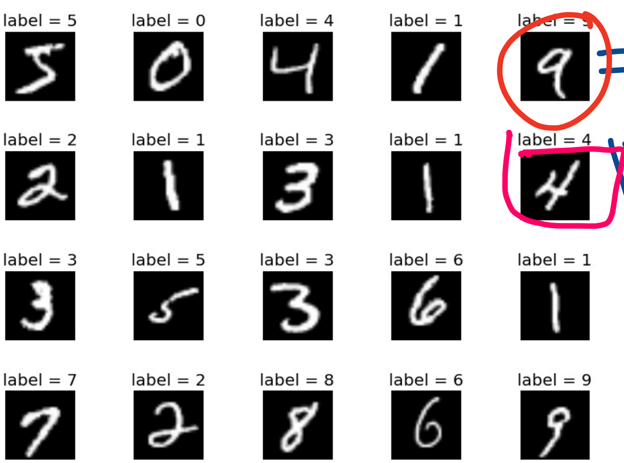
Pictures of size 8 x 8 = 64 pixels

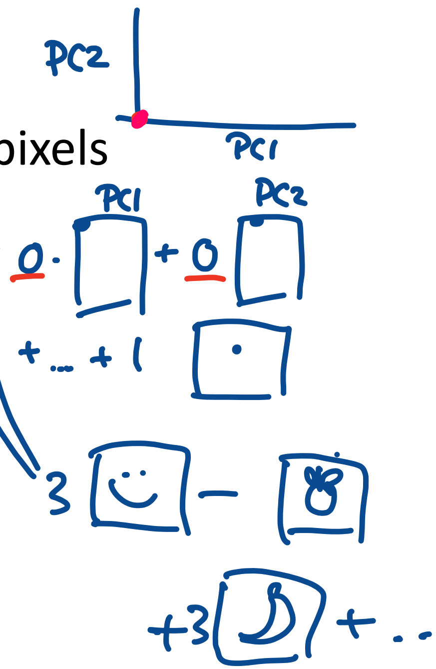
Naive approach: Scale the image down or ignore pixels

Unsupervised learning: Dimensionality reduction

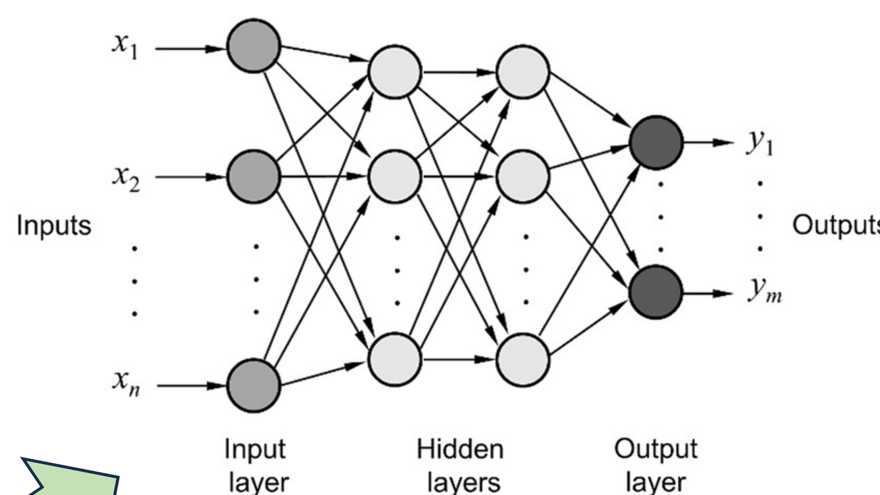
Example: Digit recognition (MNIST Dataset)

Pictures of size 28 x 28 = 784 pixels





Neural network with input layer size X

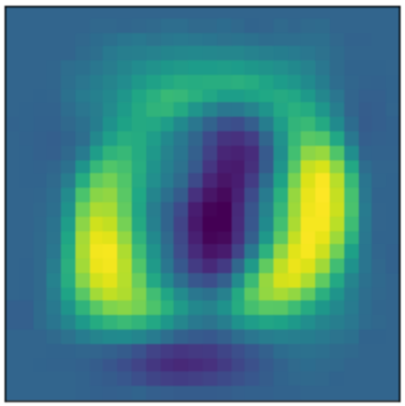


784 Datapoints ordered by "importance".
Choose the first X of them.

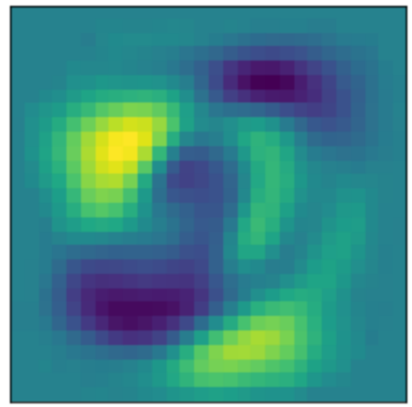
Better approach: Find a new representation of the picture into principal components.

Unsupervised learning: Dimensionality reduction

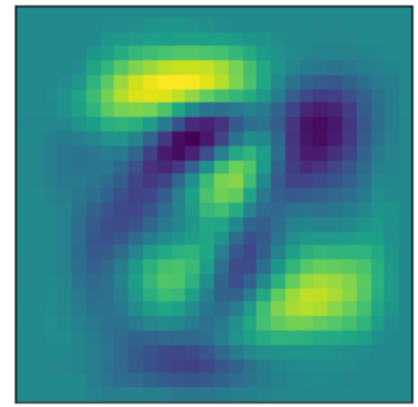
The first three principal components (PC) of the MNIST dataset



PC1

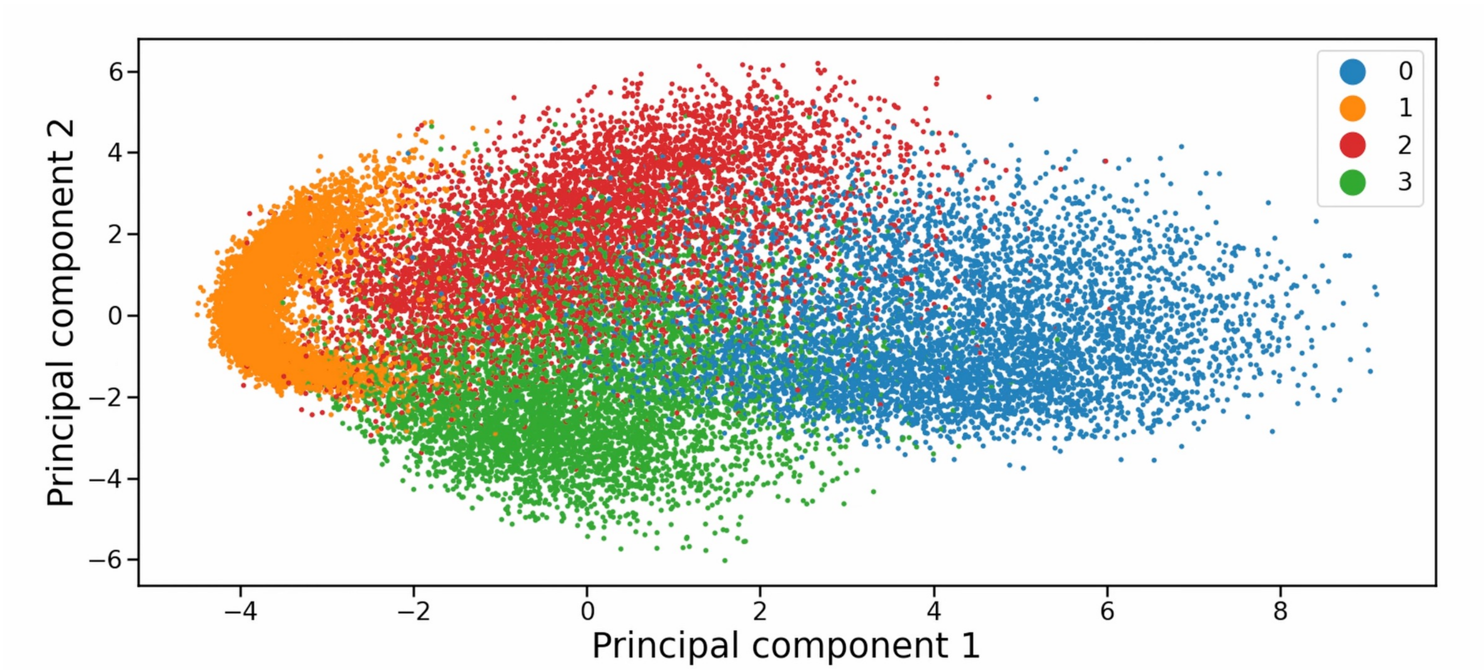


PC2



PC3

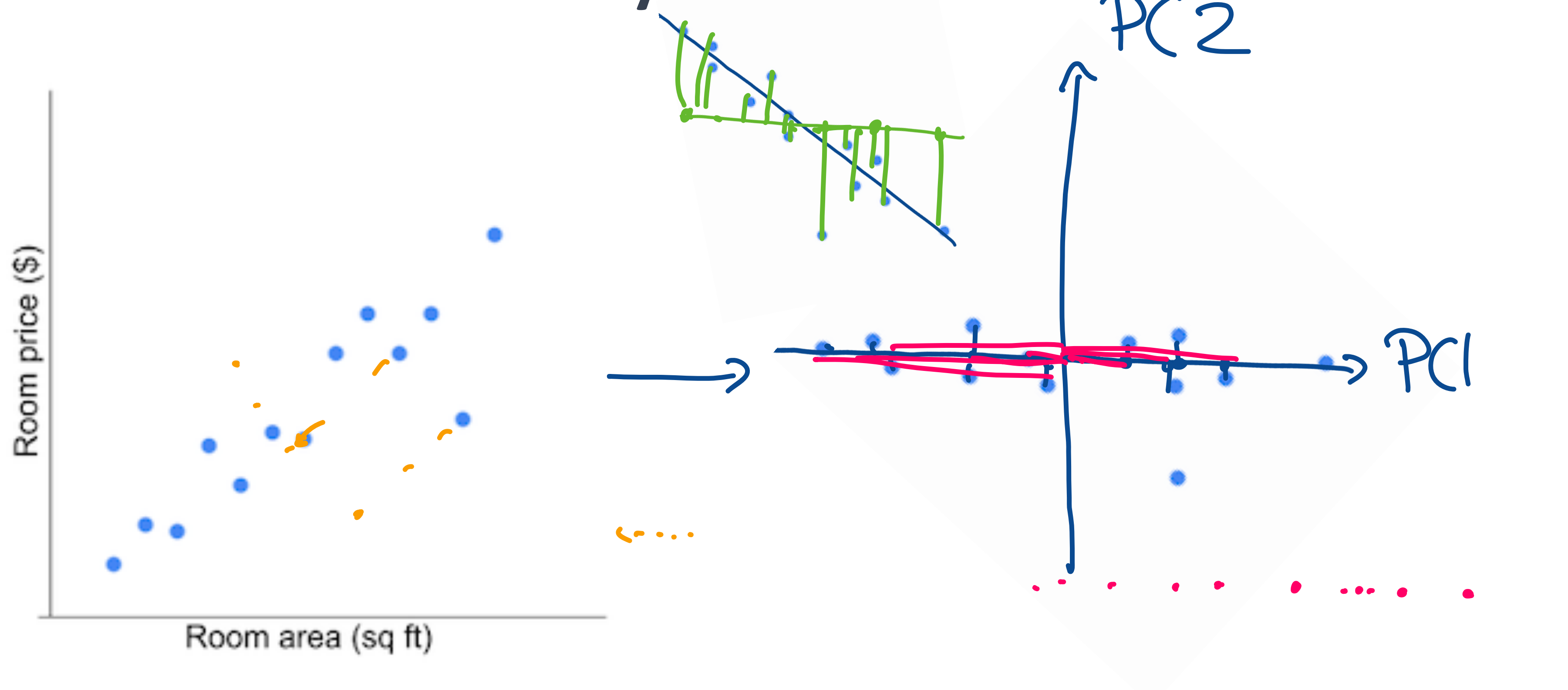
...



Principal Component Analysis (PCA)

Principal Component Analysis (PCA) is a method that reduces the dimensionality of data by transforming it into principal components, each representing unique variance, while retaining the most significant information.

“Finds a better coordinate system for given data, such that the axes are ordered by importance”

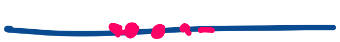


PCA – Variance

Given numbers $X = \{x_1, \ldots, x_n\} \subset \mathbb{R}$

average: $\mu = \frac{1}{n} \sum_{i=1}^{n} x_i$

variance: $\mathrm{Var}(X) = \frac{1}{n} \sum_{i=1}^{n} (x_i - \mu)^2$

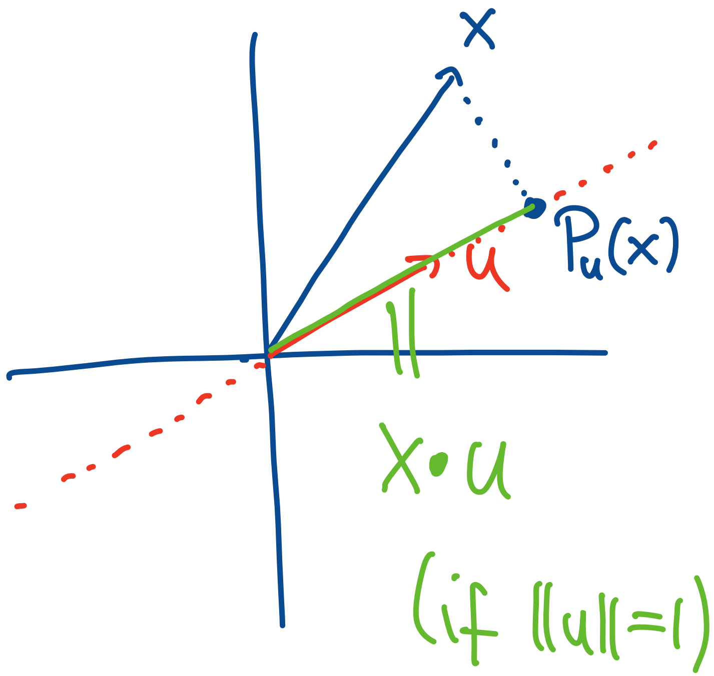
Var() > Var()

PCA – Orthogonal projection

Given $u \in \mathbb{R}^n$ $\qquad x \cdot u = x^T u$

orthogonal projection:

$$P_u(x) = \frac{x \cdot u}{u \cdot u} u$$



If $\|u\| = 1$ $\quad (u \cdot u = 1)$

$$P_u(x) = (x \cdot u) u$$

PCA – First principal component

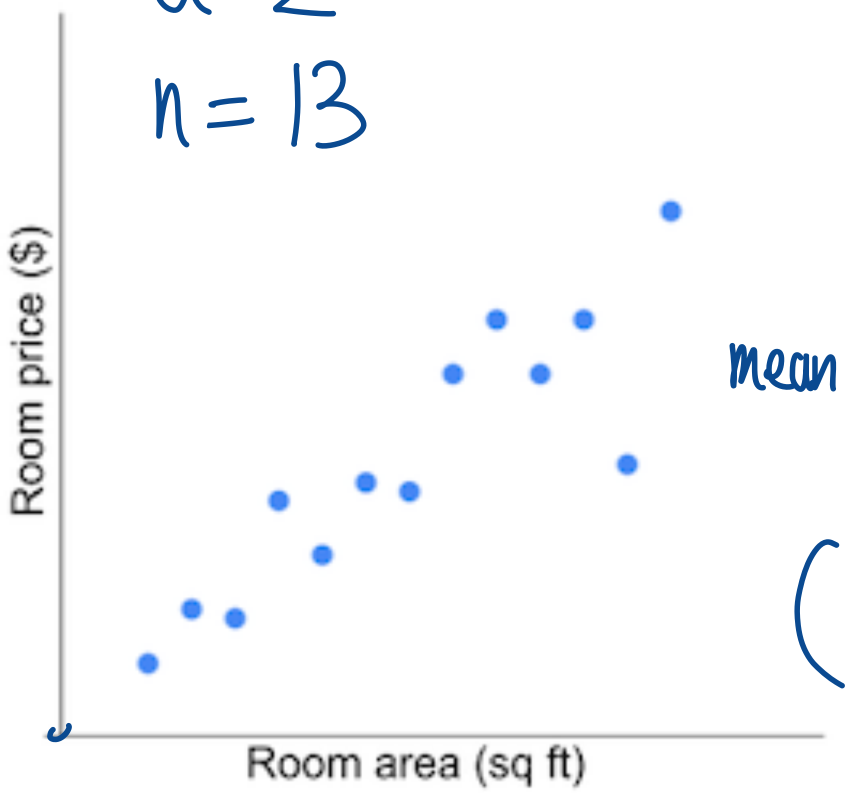
How to find the first "principal component"

Given $x^{(i)} \in \mathbb{R}^d$ $1 \le i \le n$

$X = \{x^{(i)}\}$

$d=2$

$n=13$



Step 1: Scale s.th.

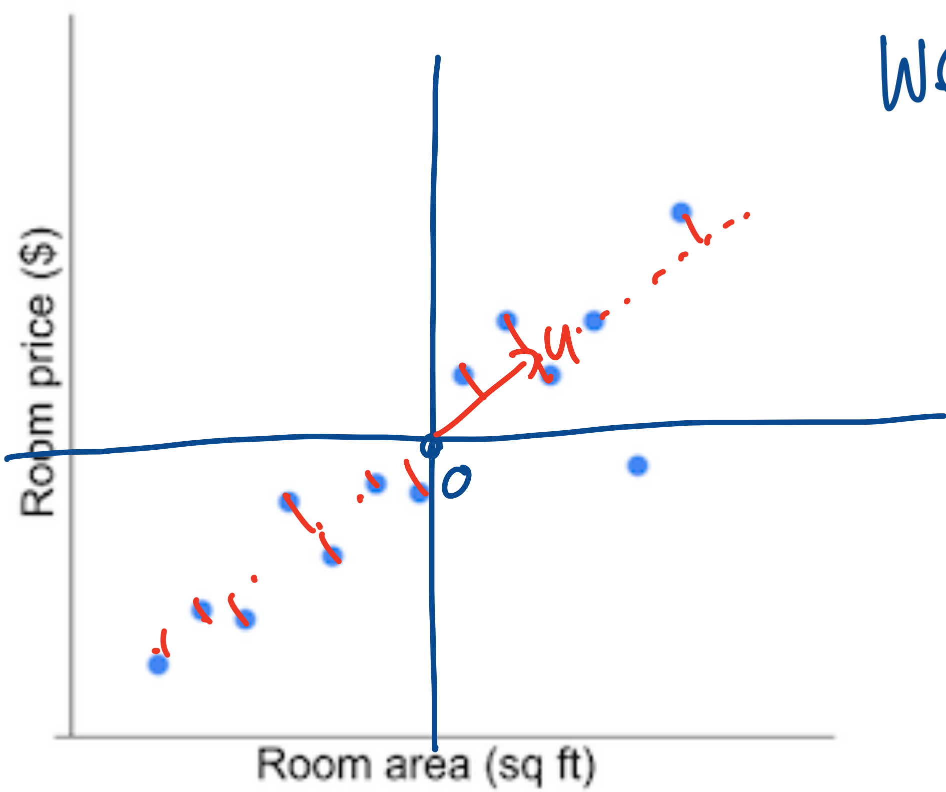
mean $\frac{1}{n}\sum_{i=1}^{n} x^{(i)} = 0$

(standard deviation = 1)

PCA -

How to find the first "principal component" $\|u\|=1$

after scaling:



Want $u \in \mathbb{R}^d$ such that

we maximize:

$$\frac{1}{n}\sum_{i=1}^{n}\left(x^{(i)}\cdot u\right)^2$$

$$\left(x^{(i)T}u\right)^T x^{(i)T} u$$

$$u^T x^{(i)} x^{(i)T} u$$

$$\mathbb{R}^{d\times d} \qquad U$$

$$= u^T \boxed{\frac{1}{n}\sum_{i=1}^{n} x^{(i)} x^{(i)T}}\, u$$

$$= u^T \boxed{\frac{1}{n}\sum_{i=1}^{n} x^{(i)} x^{(i)T}} u \qquad X = n \begin{pmatrix} \overset{d}{\overline{\quad}} \\ — x^{(1)} — \\ — x^{(2)} — \\ \vdots \\ — x^{(n)} — \end{pmatrix}$$

$$= \Sigma : \text{covariance matrix}$$

$$= \frac{1}{n} X X^T$$

In example

$$\Sigma = \begin{pmatrix} & \circ \\ \circ & \end{pmatrix}$$

covariance of room size & price

$$\boxed{\begin{array}{l} A \in \mathbb{R}^{n \times n},\ x \neq 0 \\ \qquad Ax = \lambda x \\ \lambda : \text{eigenvalue},\ x : \text{eigenvector} \end{array}}$$

$$A = \begin{pmatrix} 1 & 0 \\ 0 & 2 \end{pmatrix}, \quad A\begin{pmatrix} 1 \\ 0 \end{pmatrix} = 1 \cdot \begin{pmatrix} 1 \\ 0 \end{pmatrix}$$

$$A\begin{pmatrix} 0 \\ 1 \end{pmatrix} = 2\begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

Summary: For symmetric M we want to maximize $u^T M u$.

Theorem: For any sym. $M \in \mathbb{R}^{d \times d}$ the term $\frac{u^T M u}{u^T u}$ (Rayleigh quotient) is maximal if u is an eigenvector of M for the largest eigenvalue.

Proof: By the spectral theorem

$$M = QDQ^T \qquad \lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_d$$

$$D = \begin{pmatrix} \lambda_1 & & & 0 \\ & \lambda_2 & & \\ & & \ddots & \\ 0 & & & \lambda_d \end{pmatrix}$$

Assume $\|u\|=1$

$$u^T M u = u^T Q D Q^T u$$

$u = Qy \qquad = (\underbrace{Q^T u}_{y})^T D\, Q^T u$

$Q^T u = y$

$$= y^T D y$$

$y = \begin{pmatrix} y_1 \\ \vdots \\ y_d \end{pmatrix}$ $\qquad \|y\| = 1 \qquad y_1^2 + y_2^2 + \dots + y_d^2 = 1$

$$\underset{\begin{pmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_d \end{pmatrix}}{y^T D y} = \sum_{i=1}^{d} \lambda_i y_i^2$$

maximal for $y = \begin{pmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix}$

(then it is λ_1)

$$u = Qy$$

$$\| \qquad Q \begin{pmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix} = \text{first column of } Q$$

$$= \text{eigenvector for } \lambda_1.$$