Machine Learning applied to Planetary Sciences

PTYS 595B/495B

Leon Palafox

https://leonpalafox.github.io/MLClass/

PSA

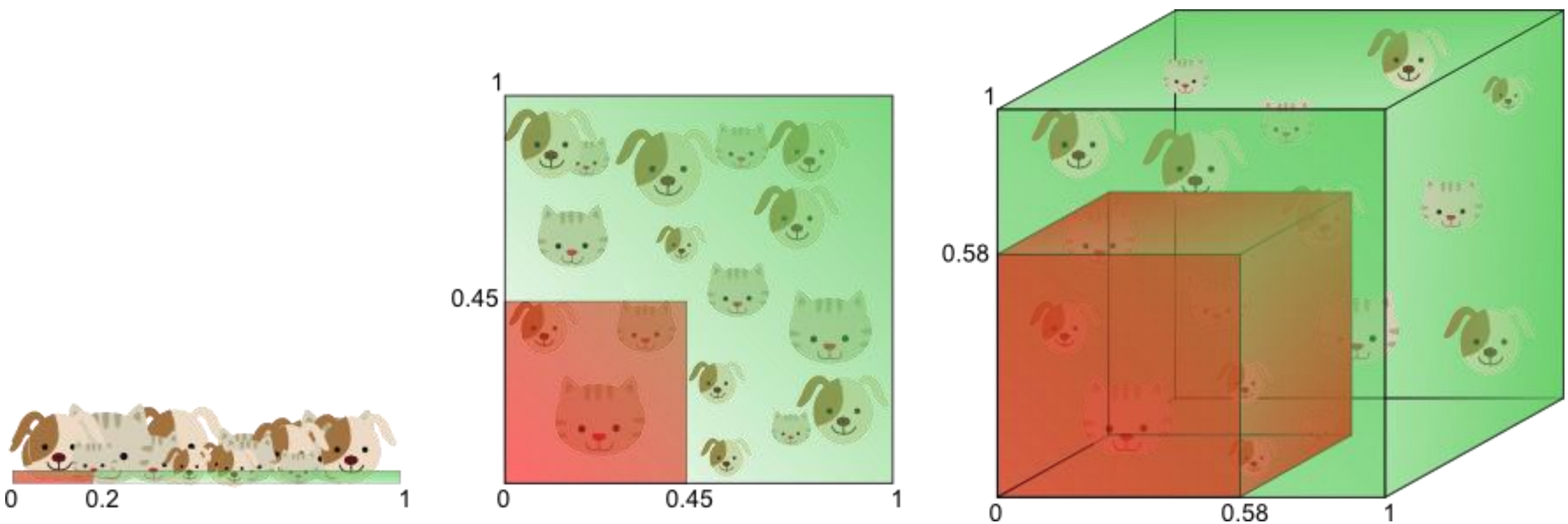
- Homework questions?
- Final Quiz:
  - Unsupervised Learning
  - Date: 12/7 (Last Session)
- Final Project
  - Due date: 12/14
  - Questions?

Curse of dimensionality

- As we have more dimensions, finding the underlying manifold becomes more difficult
  - Also applies to classifiers
  - ConvNets are surprisingly robust to this problem.
- It affects the concept of Euclidean distance.
  - If you have an algorithm, or data processing technique that uses this, beware.

Curse of dimensionality

Imagine you want to train a classifier, and want to cover 20% of the population of cat's and dogs

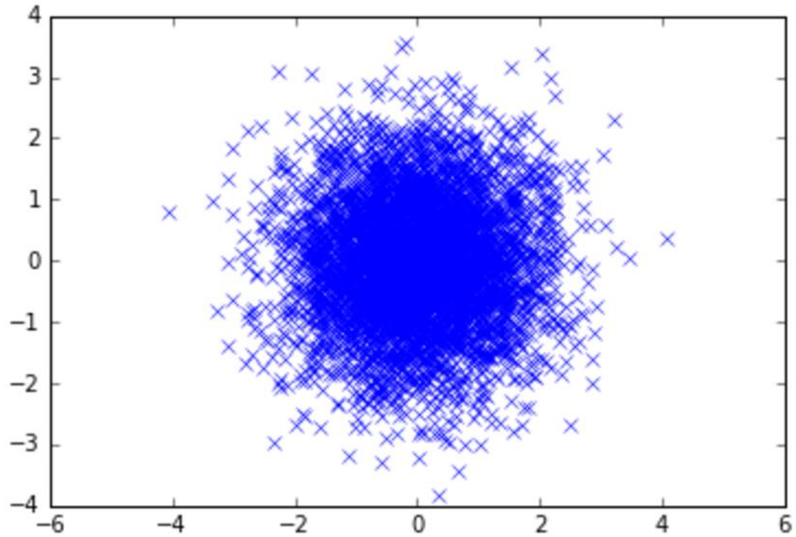


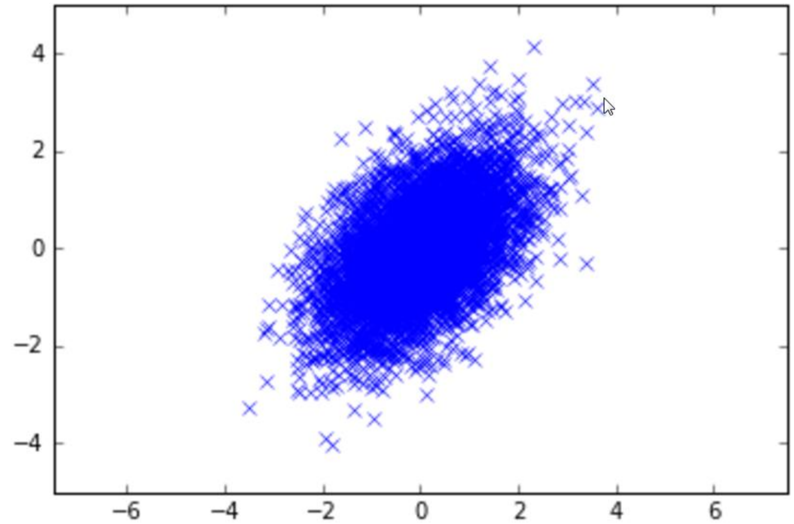
$0.45^2 = 0.2$ $0.58^3 = 0.2$

Feature decomposition

- When we have a very large number of features:
  - Many of them are redundant.
    - Speed in different dimensions
    - Different electrodes in an EEG signal
- People usually calculate the Correlation matrix.
  - It shows which features change in tandem.

Covariance Matrix



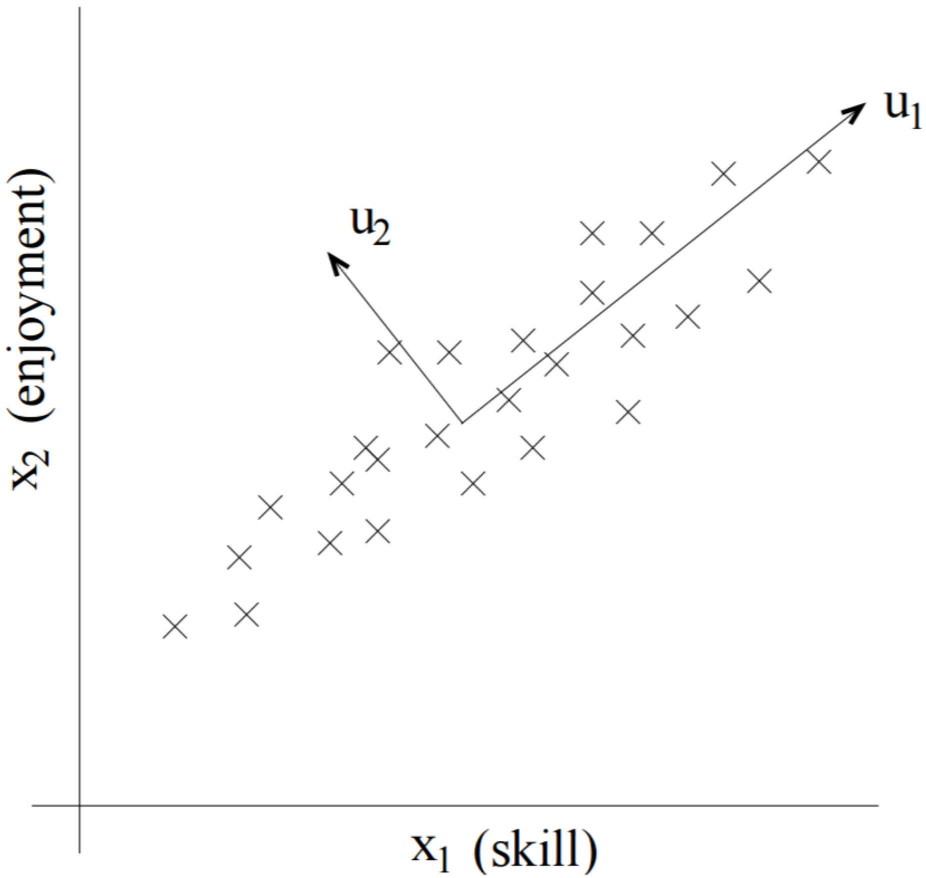


There should be a more elegant way

- Remember eigenvalues?
  - What did we use them for.
- Calculating the eigenvalues on the data, can be an indicator of covariance.
- If we know how the eigenvalues:
  - We know the transformations in the data.
  - We can gain insight on which features affect the data in a similar way.

Scenario

- We are trying to detect whether a user tagging galaxy shapes (or craters) is good, average or outright bad.
- Crate a survey:
  - Measure different variables:
    - Skill (based on previous tagging)
    - Enjoyment (perhaps based on time)
    - Accuracy
    - Age, Gender, Race, etc



- If we create new axes.
  - U is a new space that captures the “karma” of each user.
  - The axes of U capture the variability of the data.

PCA algorithm

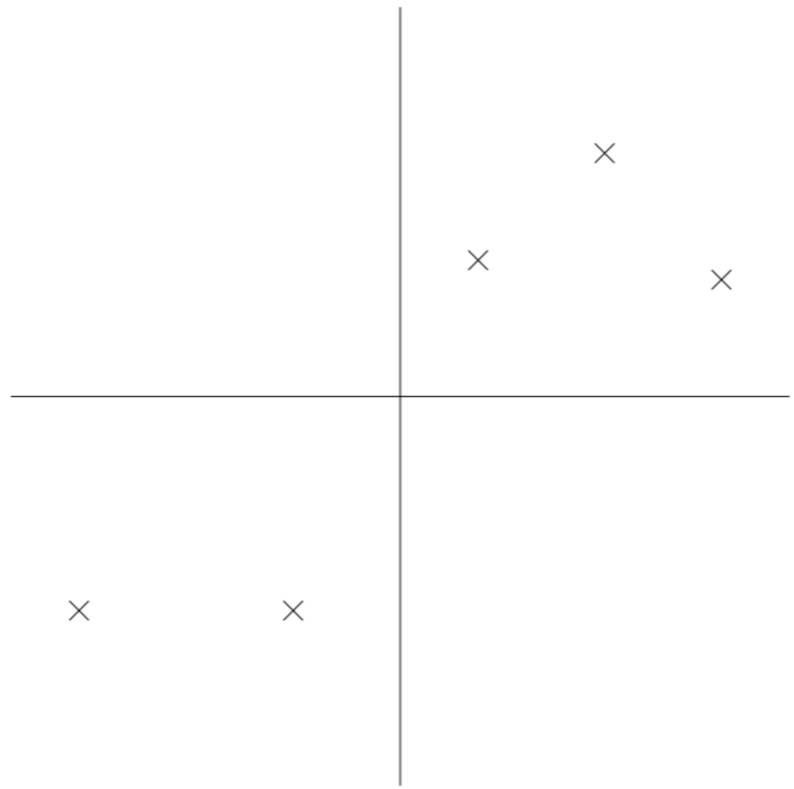
- First we need to normalize the data.
  - We want the sources of variability be described by the data itself, not by the units.
  - Normalization is not unique to PCA, and can be used in regressions and classifications.
- It forces the data to have zero mean and standard deviation of 1.
  - How would you do that?

Normalization

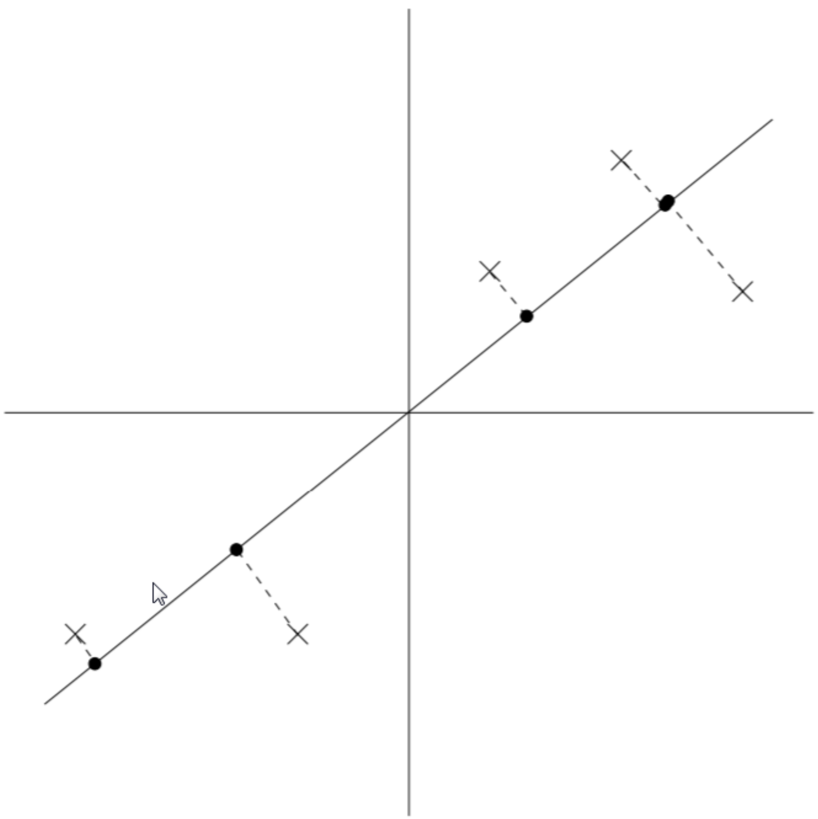
1. Let $\mu = \frac{1}{m}\sum_{i=1}^{m} x^{(i)}$.
2. Replace each $x^{(i)}$ with $x^{(i)} - \mu$.
3. Let $\sigma_j^2 = \frac{1}{m}\sum_i (x_j^{(i)})^2$
4. Replace each $x_j^{(i)}$ with $x_j^{(i)}/\sigma_j$.

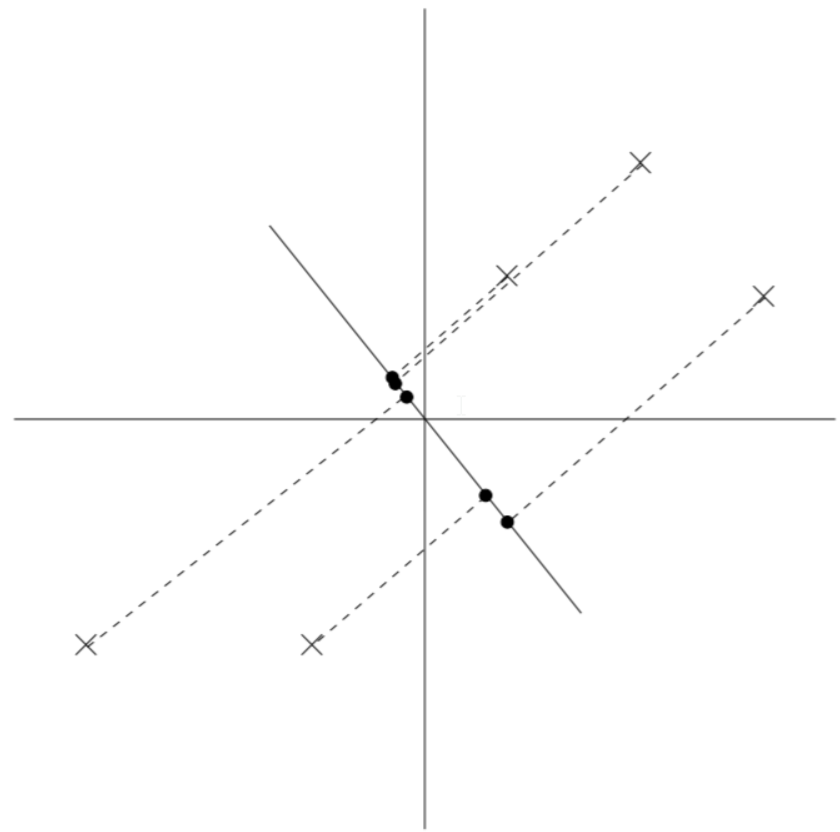
Now with PCA

- Let’s find the directions with more “variation”.



Options, options!





We need to maximize the covariance.

$$\begin{aligned}
\frac{1}{m}\sum_{i=1}^{m}({x^{(i)}}^T u)^2 &= \frac{1}{m}\sum_{i=1}^{m} u^T x^{(i)} {x^{(i)}}^T u \\
&= u^T \left(\frac{1}{m}\sum_{i=1}^{m} x^{(i)} {x^{(i)}}^T\right) u.
\end{aligned}$$

- If we maximize ||U|| = 1, the result is the covariance of the data.
  - Or the principal eigenvector.

PCA

$$y^{(i)} = \begin{bmatrix} u_1^T x^{(i)} \\ u_2^T x^{(i)} \\ \vdots \\ u_k^T x^{(i)} \end{bmatrix} \in \mathbb{R}^k.$$

- If we want to re-project the data, we just need to calculate the eigenvectors and multiply.
- If we calculate k eigenvectors, we can reproject in k-dimensions.

Python/Matlab Code

- Sklearn.decomposition.pca
  - n_components = # of components to keep.
  - whiten = normalization
  - tol = tolerance for the calculation of the eigenvalues
- PCA
  - NumComponents