Machine Learning applied to Planetary Sciences

PTYS 595B/495B

Leon Palafox

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

PSA

- Homework will be posted by the end of the week.
  - It will be shorter than previous ones.
  - It will cover validation and applications.
- Final Quiz:
  - Unsupervised Learning
  - Date: 12/7 (Last Session)
- Final Project
  - Last day of classes: December 7^{th}. (Only 4 sessions left)
  - Due Date: Either 12/12 or 12/14?
  - Groups?

Clustering

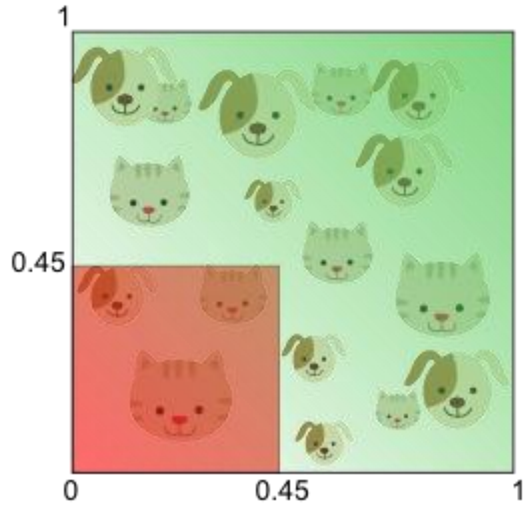
- Clustering algorithms group the data into K groups.
- While many algorithms need the number of groups K, there are some that find the best number.
- Popular Algorithms:
  - K-Means
  - Agglomerative Clustering
  - Density Estimation

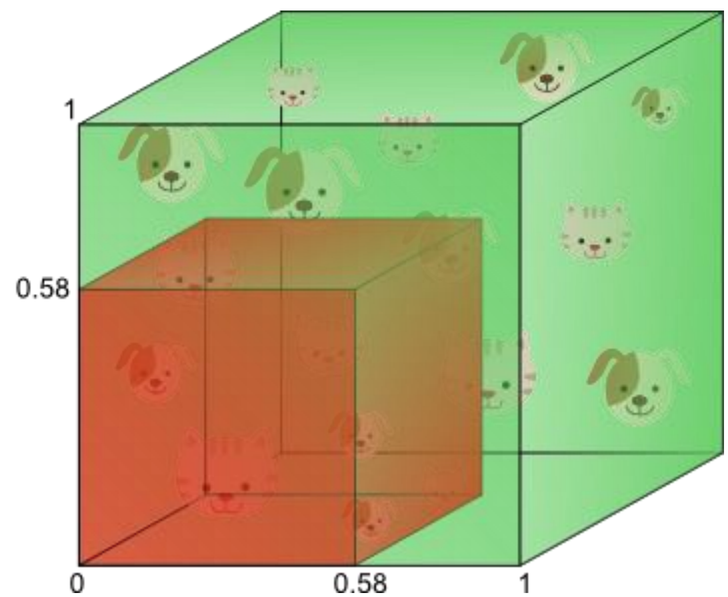
Curse of dimensionality

- 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

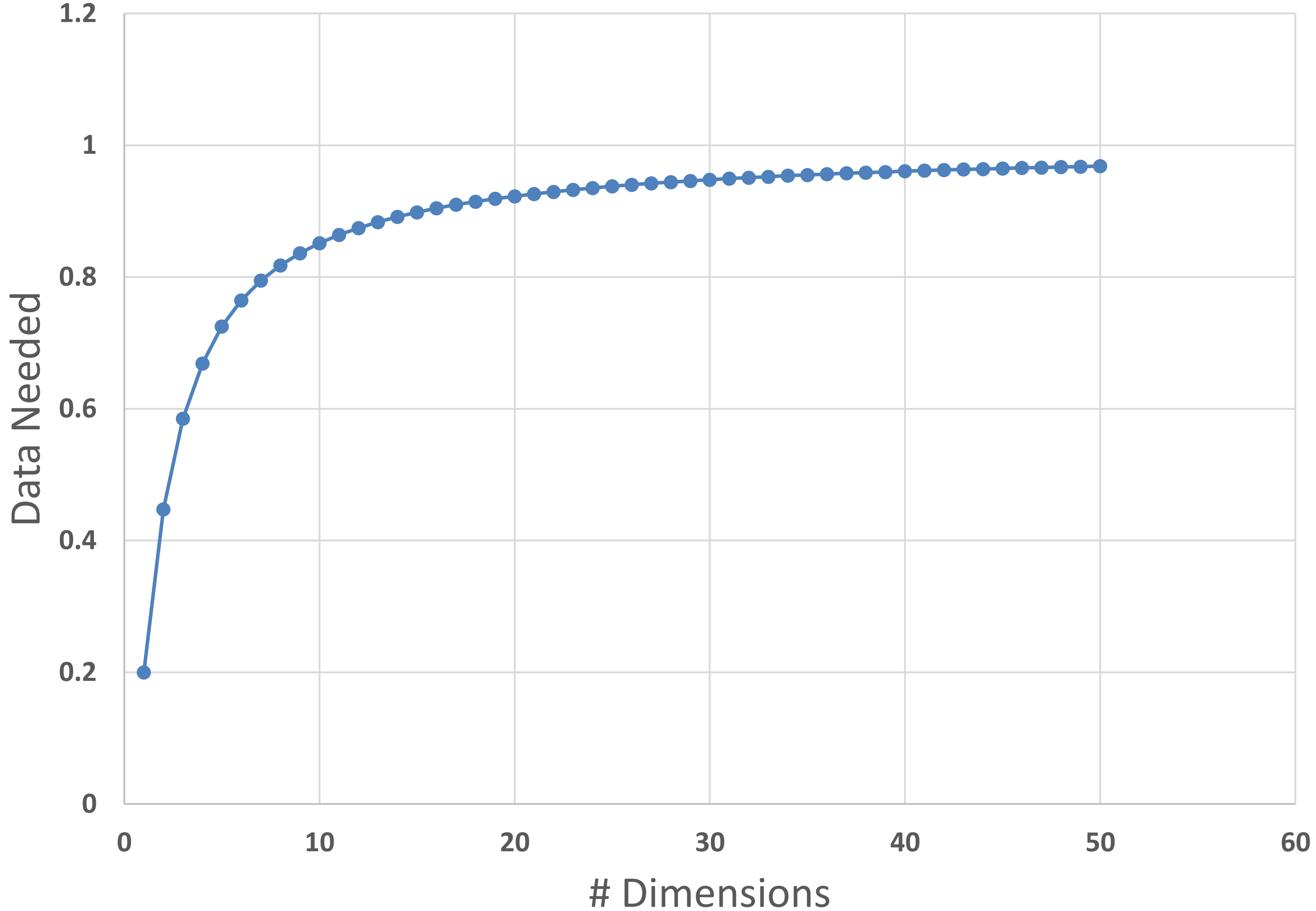




0.45^2 = 0.2

0.58^3 = 0.2

Curse of dimensionality



How to alleviate?

- We need to do feature selection, or feature reduction techniques.
  - NN do this “automatically” since the dimension in each unit is reduced as we go deeper in the layers.
- Feature Reduction Techniques:
  - Principal Component Analysis.
  - Independent Component Analysis.

K-Means

- This is the most popular and most widely used algorithm.

- It has few (if none) parameters to tune and play around with.

- It suffers greatly from the curse of dimensionality.

- It still does a pretty solid job once the dimensions have been reduced.

K-Means

1. Initialize **cluster centroids** $\mu_1, \mu_2, \ldots, \mu_k \in \mathbb{R}^n$ randomly.

2. Repeat until convergence: {

   For every i, set
   $$c^{(i)} := \arg\min_j ||x^{(i)} - \mu_j||^2.$$

   For each j, set
   $$\mu_j := \frac{\sum_{i=1}^m 1\{c^{(i)} = j\} x^{(i)}}{\sum_{i=1}^m 1\{c^{(i)} = j\}}.$$

   }

What is it doing?

- It’s minimizing the objective function:

$$J(c, \mu) = \sum_{i=1}^{m} ||x^{(i)} - \mu_{c^{(i)}}||^2$$

- Where J is a cost function between each point and its assigned centroid.

How do we evaluate K-Means

- If we have labels of what the clusters should be (at least some examples):
  - We can calculate TP, FP, TN and FN

$$RI = \frac{TP + TN}{TP + FP + FN + TN}$$

  - This is our accuracy. (That Rand guy was nifty)

How do we evaluate K-Means

- If we don’t have labels:
  - Davies-Boudin Index

$$DB = \frac{1}{n}\sum_{i=1}^{n}\max_{j\neq i}\left(\frac{\sigma_i + \sigma_j}{d(c_i, c_j)}\right)$$

  - σ is the average distance of the elements in the cluster to the centroid.
  - d(c,c) is distance between centroids.
  - It provides low scores for low intra-disctance and high inter-distance.

How do we use K-Means

- Matlab:
  - idx = kmeans(X,k)

- Python:
  - Kmeans(n_clusters, random_state)