Unsupervised Machine Learning

Section 3: Review of Mathematical Concepts

Mathematical Concepts

Section	Contents
3.1	What is the Data in Unsupervised Learning?
3.2	Distance Measures -Review of formulas for distance between points and sets of points
3.3	Computational Complexity -Discussion of how to measure how much work that an algorithm requires
3.4	Singular Value Decomposition -Theory underlying principal component analysis for reducing number of dimensions in data

Section 3.1: What is the Data in Unsupervised Learning?

Data and Datasets

- Typically, a data point is a vector in d dimensions

$$\begin{bmatrix} x_0 \\ \dots \\ x_{d-1} \end{bmatrix}$$

Data point often called feature vector as each entry represents a feature

- Let $X_0, X_1, \dots, X_{M-1}$ denote the m datapoints, then dataset is represented as a matrix of dimensions d rows and M columns

$$X = [X_0 \quad \dots \quad X_{m-1}]$$

X often called feature matrix

Example: Customer Segmentation

Data point consists of features of customer

- Consider case where features age, gender, salary, # of purchases

age = 27

gender = female (0 for male and 1 for female)

salary = 60,000

# of purchases = 10

- Data point:

$$\begin{bmatrix} 27 \\ 1 \\ 60000 \\ 10 \end{bmatrix}$$

Example: Natural Language Processing

- Simple approach is word count
  - Create dictionary of all words
  - Count number of times each word appears in each document
- Consider 3 messages:

"Call me soon", "CALL to win", "Pick me up soon"

Dictionary

$$\begin{matrix} \boxed{call} \\ \boxed{me} \\ pick \\ \boxed{soon} \\ to \\ up \\ win \end{matrix}$$

Feature Matrix

$$\begin{bmatrix} \boxed{1} & 1 & 0 \\ \boxed{1} & 0 & 1 \\ 0 & 0 & 1 \\ \boxed{1} & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

Example: Natural Language Processing

- Term Frequency Inverse Document Frequency (Tfidf)
  - Term frequency: number of times word appears in document
  - Inverse document frequency: inverse of number of documents word appears
  - Tfidf is term frequency multiplied by inverse document frequency (with scaling)
- Messages:

"Call me soon", "CALL to win", "Pick me up soon"

Dictionary / Feature Matrix

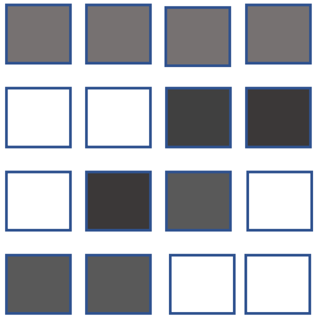
$$\begin{array}{c} call \\ me \\ pick \\ soon \\ to \\ up \\ win \end{array} \quad \begin{bmatrix} 0.58 & 0.47 & 0.00 \\ 0.58 & 0.00 & 0.43 \\ 0.00 & 0.00 & 0.56 \\ 0.58 & 0.00 & 0.43 \\ 0.00 & 0.62 & 0.00 \\ 0.00 & 0.00 & 0.56 \\ 0.00 & 0.62 & 0.00 \end{bmatrix}$$

Example: Images

- Images typically are composed of rectangular arrays of pixels
- For black and white images, intensity of greyscale for each pixel is represented by a number (white = 0 to 255 = black)
- Feature vector for image is vector of intensities for all pixels
- For colour images, each pixel represented by 3 values – intensities of red, blue, and green components for that pixel – feature vector in colour case vector will be 3 times longer than in black and white case

Converting Image to Matrix

Original Image:
Greyscale 4x4 =16 pixels



Intensity Matrix
4x4 (white=0 to 255=black)

$$\begin{bmatrix} 190 & 190 & 190 & 190 \\ 0 & 0 & 220 & 220 \\ 0 & 220 & 200 & 0 \\ 200 & 200 & 0 & 0 \end{bmatrix}$$

Feature Vector 16x1
Standard to divide by 255

$$\begin{bmatrix} 190 \\ 190 \\ 190 \\ 190 \\ 0 \\ 0 \\ 220 \\ 220 \\ 0 \\ 220 \\ 200 \\ 0 \\ 200 \\ 200 \\ 0 \\ 0 \end{bmatrix}$$

Websites for Data

Kaggle

- www.kaggle.com
- Site for data science competitions (often with prize money)
- Each competition comes with freely available data
- Can learn from tutorials, practice competitions, and notebooks created by participants
- You will need to create a free account to access the resources (not needed for this course)

University of California, Irvine Machine Learning Data Repository

- https://archive.ics.uci.edu/ml/index.php
- Contains 100s of machine learning datasets
- No account required

3.1 sklearn Text Processing DEMO

Jupyter Notebook for demo:

- UnsupervisedML/Examples/Section03/SklearnText.ipynb

Course Resources at:

- https://github.com/satishchandrareddy/UnsupervisedML/

Section 3.2: Computational Complexity

Computational Complexity

- Complexity of an algorithm is amount of resources (number of operations, memory, etc) to run it
- Typically, represent complexity as a function of the size of the input
  - For sorting, represent complexity in terms of number of elements in list
  - For matrix multiplication, represent complexity in terms of size of input matrices
- In this course, we provide complexity estimates for clustering algorithms

Big O Notation

- A function $f(M) = O\big(g(M)\big)$ as M$\rightarrow \infty$ if

$$|f(M)| \leq C|g(M)| \quad as\ M \rightarrow \infty$$

Here C is a constant

Examples

- Well known result from computer science is that sorting of a list of M elements can be done in $O(MlogM)$ operations as $M \to \infty$
- If X and Y are vectors of length d, then computation of dot product $X^T Y$ requires d multiplications and d-1 additions, hence it requires $O(d)$ operations as $d \to \infty$

Section 3.3: Distance Measures

Why is a Distance Measure Needed?

- Throughout Unsupervised Learning, one needs to compute distances between data points or distances between clusters of data points
- For example: Clusters are often defined in terms of points within a distance of other points
- Let us define

$$\mathrm{X} = \begin{bmatrix} x_0 \\ \cdots \\ x_{d-1} \end{bmatrix} \qquad \mathrm{Y} = \begin{bmatrix} y_0 \\ \cdots \\ y_{d-1} \end{bmatrix}$$

Examples of Distance Measure

- L2 or Euclidean distance measure between X and Y defined as

$$dist(X,Y) = \left[\sum_{i=0}^{d-1} |x_i - y_i|^2\right]^{1/2}$$

In this course, we will use the Euclidean distance measure

- L1 or Taxicab distance measure between X and Y defined as

$$dist(X,Y) = \sum_{i=0}^{d-1} |x_i - y_i|$$

- p norm or Minkowski distance between X and Y is a general distance measure that incorporates L1 and L2 measures as special cases:

$$dist(X,Y) = \left[\sum_{i=0}^{d-1} |x_i - y_i|^p\right]^{1/p}$$

Computational Complexity

- Confirm for yourself that number of operations to compute L1, L2, or Lp distance between 2 vectors of length requires $O(d)$ operations and memory as $d \to \infty$

Distance Between Clusters

- Suppose $\{X_i\}$ i=0,…,m-1 is the set of points in a cluster
- Define cluster mean as

$$C = \frac{1}{m}\sum_{i=0}^{m-1} X_i$$

- If $\{X_i\}$ and $\{Y_j\}$ are two clusters and let C_X and C_Y denote their means

- Distance between clusters defined as distance between the cluster means:

$$dist\left(\{X_i\},\{Y_j\}\right) = dist(C_X, C_Y)$$

3.3 Distance Computation DEMO

Jupyter Notebook for demo:

- UnsupervisedML/Examples/Section03/Distance.ipynb

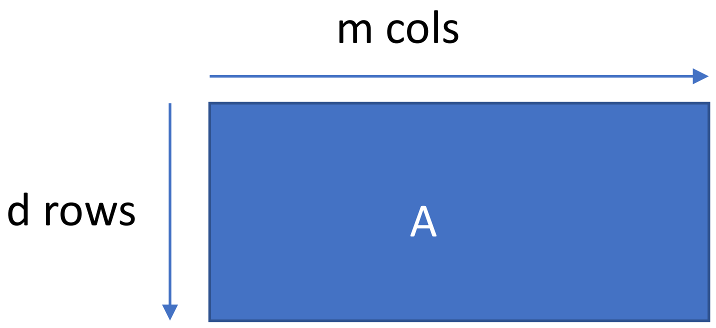
Course Resources at:

- https://github.com/satishchandrareddy/UnsupervisedML/

Section 3.4: Singular Value Decomposition

Looking at a Matrix

- Consider a matrix A (d rows and M columns)



- A is can considered as mapping from R^M (M dimensional space) to R^d (d dimensional space)
- If $y = Ax$, then x point in R^M (M dimensional space) and y is a point in R^d (d dimensional space)

Singular Value Decomposition

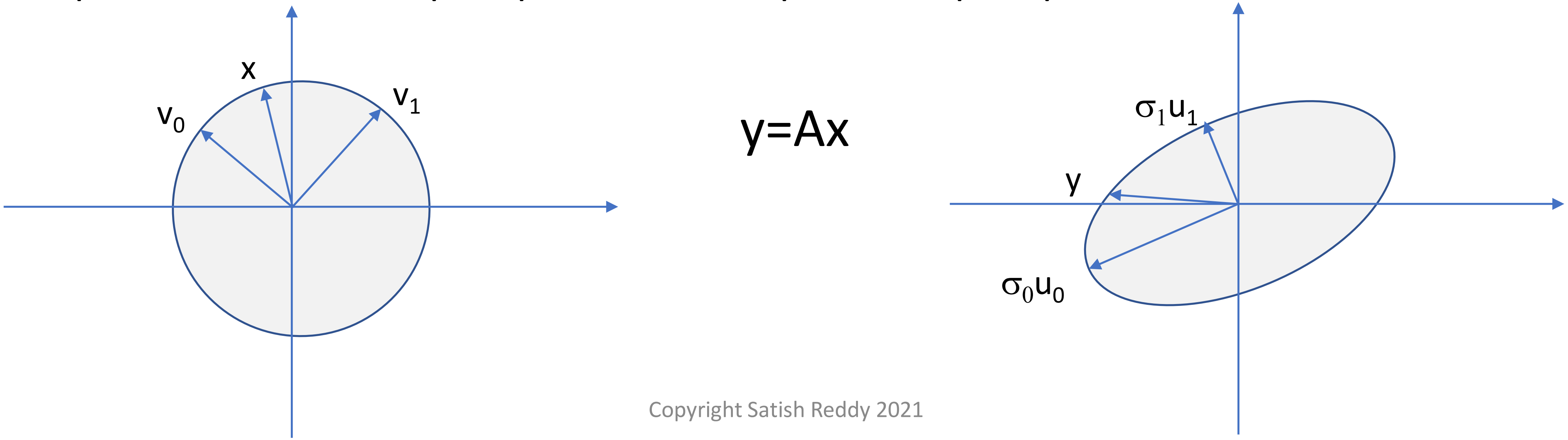
- A can be decomposed as A = $U\Sigma V^T$, where (here d<=M)

$$U = [u_0 \quad \dots \quad u_{d-1}] \quad \Sigma = diag(\sigma_0, \dots, \sigma_{d-1}) \quad V = \begin{bmatrix} v_0^T \\ \dots \\ v_{M-1}^T \end{bmatrix}$$

- U is dxd (orthogonal), Σ dxM, V is MxM (orthogonal)
- Singulars values σ_0, …, σ_{d-1} are positive and arranged in descending order
- u_0, …, u_{d-1} are d orthogonal vectors spanning d dimensional space
- v_0, …, v_{M-1} are M orthogonal vectors spanning M dimensional space

Matrix A as a Mapping and SVD

- Consider A is 2x2
- $A = \begin{bmatrix} u_0 & u_1 \end{bmatrix} \begin{bmatrix} \sigma_0 & 0 \\ 0 & \sigma_1 \end{bmatrix} \begin{bmatrix} v_0^T \\ v_1^T \end{bmatrix}$
- Av_0 mapped to $\sigma_0 u_0$
- Av_1 mapped to $\sigma_1 u_1$
- x can be decomposed as a linear combination of v_0 and v_1
- y=Ax can be decomposed as a linear combination of $\sigma_0 u_0$ and $\sigma_1 u_1$
- A maps the unit disk in input space to the ellipse in output space

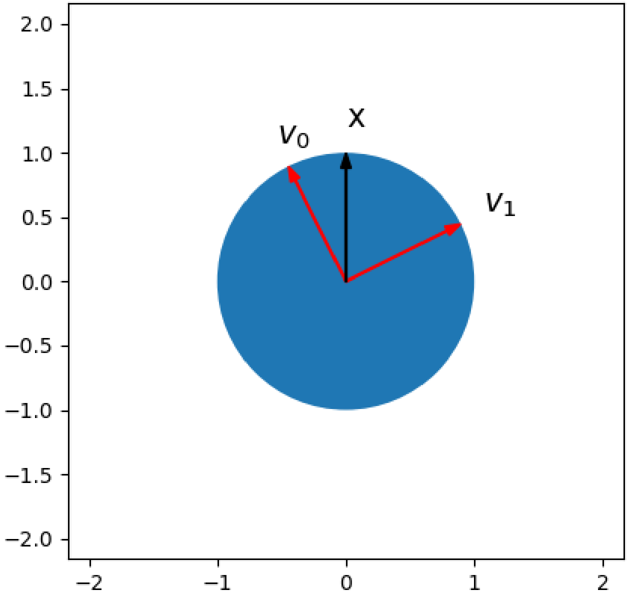


Example: 2x2 matrix

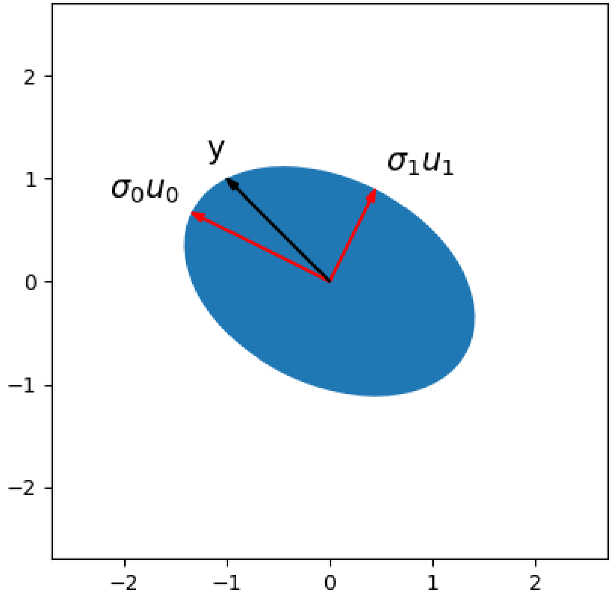
$$A = \begin{bmatrix} 1 & -1 \\ 0.5 & 1 \end{bmatrix}$$

$$A = U\Sigma V^T = \begin{bmatrix} -0.8944 & 0.4472 \\ 0.4472 & 0.8944 \end{bmatrix} \begin{bmatrix} 1.5 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} -0.4472 & 0.8944 \\ 0.8944 & 0.4472 \end{bmatrix}$$

$$Ax = \begin{bmatrix} 1 & -1 \\ 0.5 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} -1 \\ 1 \end{bmatrix}$$



: S



Singular Value Decomposition: Computation

- Eigenvalues of $A^T A$ are squares of singular values of A
- Usually one computes singular values U and V using a numerical approach without directly computing $A^T A$
- Underlying algorithm is an iterative approach
- Use numpy.linalg.svd() function in numpy
- If A is dxd, SVD computation requires $O(d^3)$ operations as $d \to \infty$

3.4 Singular Value Decomposition DEMO

Jupyter Notebook for demo:

- UnsupervisedML/Examples/Section03/SVD.ipynb

Course Resources at:

- https://github.com/satishchandrareddy/UnsupervisedML/