Unsupervised Machine Learning with Python

Section 3.0: Review of Mathematical Concepts

Review of Mathematical Concepts

Section	Contents
3.1	What is the Data in Unsupervised Machine Learning? -Demo on text processing using sklearn
3.2	Computational Complexity -Discussion describing language of complexity, used to quantify resources required by algorithms -Demo on measuring complexity using numpy
3.3	Distance Measures -Review of formulas for distance between points and sets of points -Demo on using numpy for computing distances between data points
3.4	Singular Value Decomposition -Math underlying principal component analysis for reducing number of dimensions in data -Demo on computing and visualizing SVD

See UnsupervisedML_Resources.pdf for links to additional resources

Unsupervised Machine Learning with Python

Section 3.1: What is the Data in Unsupervised Learning?

Unsupervised Machine Learning

- Goal of Unsupervised ML is to find patterns in data
- Question: what is the data?

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 data points, then dataset is represented as a matrix of dimensions d rows and M columns

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

Throughout course we will call X the dataset or the feature matrix

Example: Customer Segmentation

Data point consists of features of customer

Example: 4 features

- age = 27
- gender = female (0 for male and 1 for female)
- salary = 60,000
- # of purchases = 10
- Data point represented as:

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

- Combine feature vectors for multiple customers to create feature matrix

Example: Natural Language Processing

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

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

Words | Feature Matrix

$$
\begin{array}{c c}
call & \\
me & \\
pick & \\
soon & \\
to & \\
up & \\
win &
\end{array}
\begin{bmatrix}
1 & 1 & 0 \\
1 & 0 & 1 \\
0 & 0 & 1 \\
1 & 0 & 1 \\
0 & 1 & 0 \\
0 & 0 & 1 \\
0 & 1 & 0
\end{bmatrix}
$$

Example: Natural Language Processing

- Term Frequency Inverse Document Frequency (Tfidf) approach
  - Term frequency: number of times word appears in document
  - Inverse document frequency: inverse of number of documents in which word appears
  - Tfidf is term frequency multiplied by inverse document frequency (with scaling)
  - Logic: if word appears in many documents, then its importance/weighting is lowered
- Messages:

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

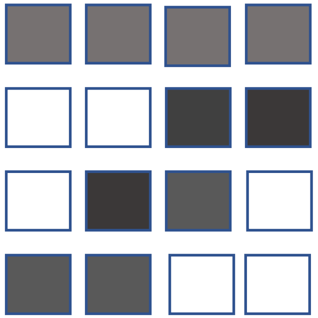
$$\begin{array}{cc} \text{Words} & \text{Feature Matrix} \\ \begin{array}{c} call \\ me \\ pick \\ soon \\ to \\ up \\ win \end{array} & \begin{bmatrix} 0.58 & 0.47 & 0.00 \\ 0.58 & 0.00 & 0.43 \\ 0.00 & 0.00 & 0.56 \\ 0.58 & 0.00 & 0.43 \\ 0.00 & 0.62 & 0.00 \\ 0.00 & 0.00 & 0.56 \\ 0.00 & 0.62 & 0.00 \end{bmatrix} \end{array}$$

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 between 0 and 255 (0=white, 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 Feature Vector

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

sklearn Toy Datasets

- https://scikit-learn.org/stable/datasets/toy_dataset.html
- 7 easy to use datasets

University of California, Irvine Machine Learning Data Repository

- https://archive.ics.uci.edu/ml/index.php
- Contains 100s of freely available machine learning datasets

Kaggle

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

3.1 sklearn Text Processing DEMO

Jupyter Notebook for demo:

- UnsupervisedML/Examples/Section03/SklearnText.ipynb

Course Resources at:

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

Unsupervised Machine Learning with Python

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 amount of time to run clustering algorithms, usually in terms of number of data points

Language of Complexity: Big O Notation

Example:

- $f(M) = O(M^2)$ as $M \to \infty$, means that $|f(M)| \approx CM^2$ as $M \to \infty$

General Definition:

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

$$|f(M)| \approx C|g(M)| \quad as\ M \to \infty$$

See Resource document Section 3 for link for more details

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 M, then computation of dot product X^TY requires M multiplications and M-1 additions, hence it requires $O(M)$ operations as $M \to \infty$

Work Complexity	Implication
$O(M)$ as $M \to \infty$	If M increases by factor of 2, then amount of work increases by factor of 2
$O(M^2)$ as $M \to \infty$	If M increases by factor of 2, then amount of work increases by factor of 4
$O(M^3)$ as $M \to \infty$	If M increases by factor of 2, then amount of work increases by factor of 8

Estimating Complexity Power from Data

- Let us assume the amount of work for an algorithm is $O(M^p)$ as $M \to \infty$. How can we estimate p?
- Note: $W = CM^p$, then $\log W = \log C + p \log M$

Estimate p as follows:

Assume work is measured by amount of time to run algorithm

(1) Collect data for Time as a function of M for the algorithm to run

(2) Take log of M and log of Time data

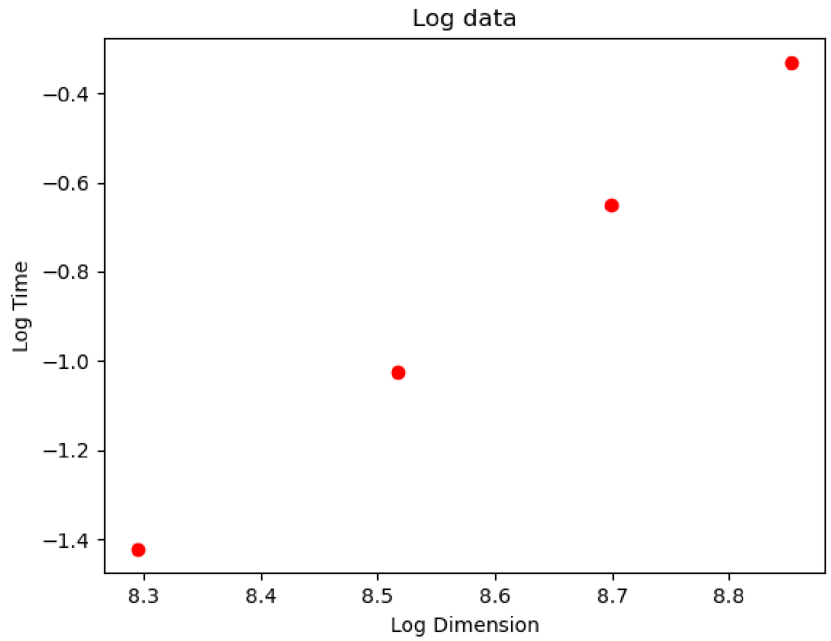
(3) Fit a straight line to log M / log Time data

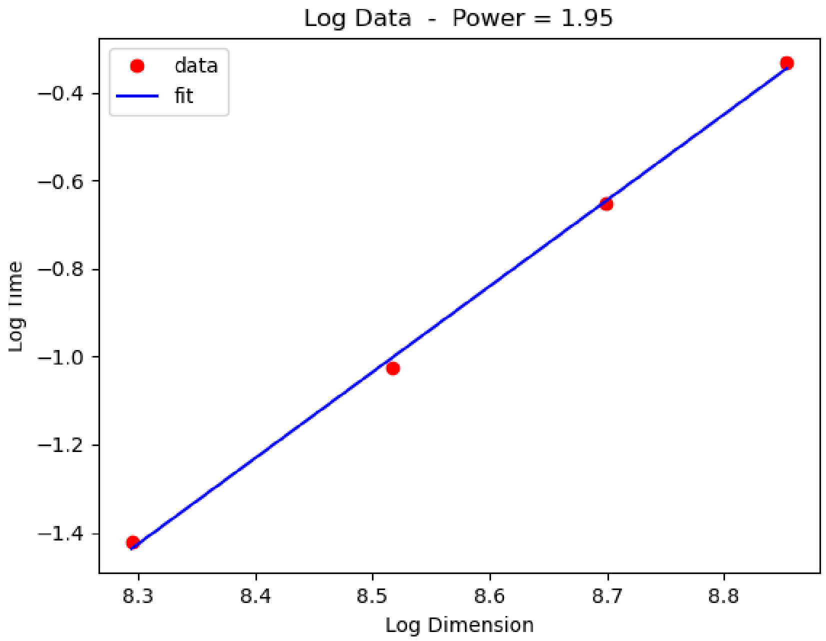
(4) Slope of line is p

(5) Intercept is $\log C$

Example: Estimating Complexity Power from Data

Dimension (M)	4000	5000	6000	7000
Time (T)	0.2413	0.3590	0.5216	0.7181
Log Dimension	8.294	8.517	8.700	8.854
Log Time	-1.422	-1.024	-0.651	-0.331





- Here: $T = O(M^{1.95})$
- This is only an estimate of behaviour as $M \to \infty$, as test M values only go to 7000.
- Timings may be affected by other processes taking place, vectorization versus looping, etc, memory issues

3.2 Computing Complexity DEMO

Jupyter Notebook for demo:

- UnsupervisedML/Examples/Section03/Complexity.ipynb

Course Resources at:

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

Unsupervised Machine Learning with Python

Section 3.3: Distance Measures

Why is a Distance Measure Needed?

- For clustering algorithms, one needs to compute distances between data points or distances between groups of data points
- We need a formula to compute such distances

Euclidean Distance Formula

- Define:

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

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

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

L1 and Lp Distance Formulas

- 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 ($p \geq 1$) 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 and memory to compute L1, L2, or Lp distance between 2 vectors of dimension d are both $O(d)$ as $d \to \infty$

Distance Between Clusters

- Suppose $\{X_j\}$ j=0,…,M-1 is a set of points in a cluster

$$X = [X_0 \quad \dots \quad X_{M-1}] = \begin{bmatrix} X_{00} & \cdots & X_{0,M-1} \\ \vdots & \cdots & \vdots \\ X_{d-1,0} & \cdots & X_{d-1,M-1} \end{bmatrix}$$

- Define cluster mean as

$$C = \frac{1}{M}\sum_{j=0}^{M-1} X_j \qquad or \quad C_i = \frac{1}{M}\sum_{j=0}^{M-1} X_{ij} \quad i = 0, \dots, d-1$$

- Suppose $\{X_j\}$ 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:

$$dist(\{X_j\}, \{Y_j\}) = dist(C_X, C_Y)$$

3.3 Distance Computation DEMO

Jupyter Notebook for demo:

- UnsupervisedML/Examples/Section03/Distance.ipynb

Course Resources at:

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

Unsupervised Machine Learning with Python

Section 3.4: Singular Value Decomposition

Singular Value Decomposition (Compact)

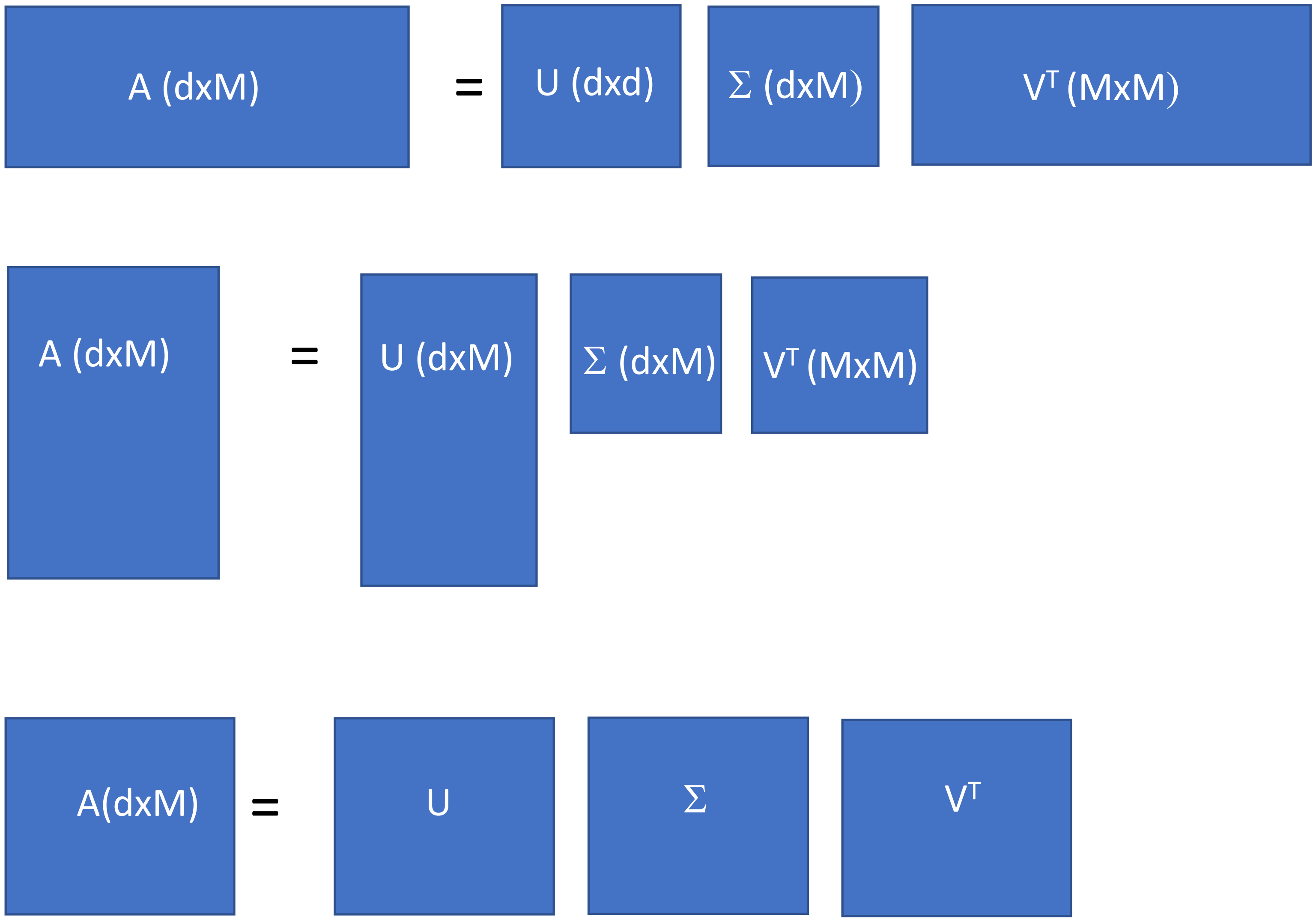
Compact version of SVD

- EVERY MATRIX A (dxM) can be decomposed as A = UΣV^T
- N = min(d,M)
- $U = [u_0 \quad \dots \quad u_{N-1}]$ U is dxN
  - Vectors u_0, …, u_{N-1} are in d dimensions, have L2 length =1 and are orthogonal (pairwise dot products = 0)
- $\Sigma = diag(\sigma_0, \dots, \sigma_{N-1})$

Singulars values σ_0, …, σ_{N-1} are non-negative and arranged in descending order

- $V^T = \begin{bmatrix} v_0^T \\ \dots \\ v_{N-1}^T \end{bmatrix}$ V is NxM
  - Vectors v_0, …, v_{N-1} are in M dimensions, have L2 length=1 and are orthogonal (pairwise dot products = 0)

Singular Value Decomposition (Compact)



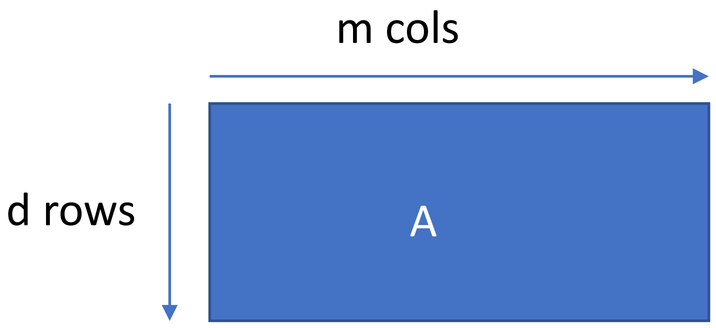
$d < M$

$d > M$

A (dxM) $d = M$

Matrix as a Mapping

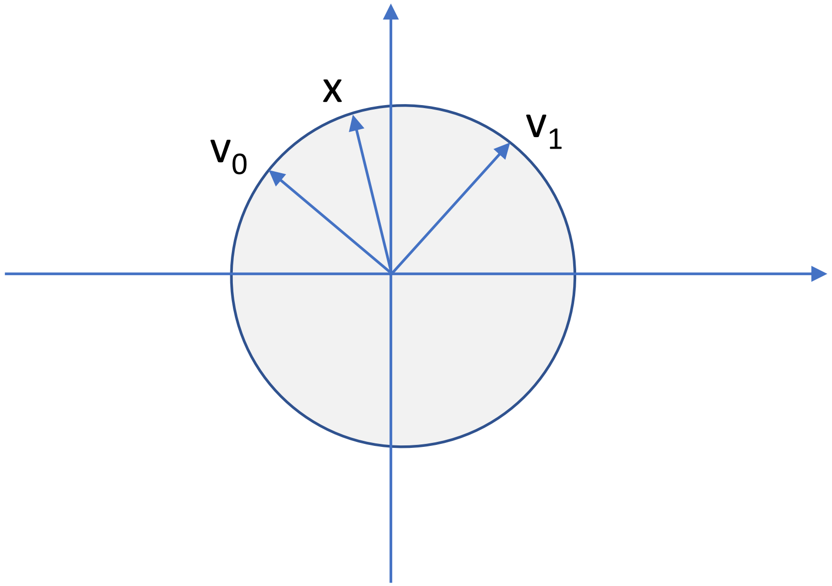
- Consider a matrix A (d rows and M columns)



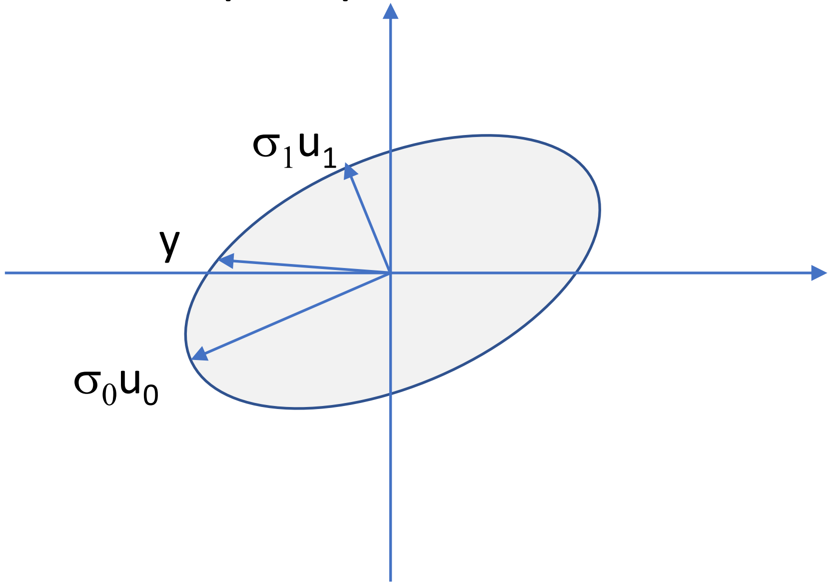
- If $y = Ax$, then x point in R^M (M dimensional space) is mapped to y point in R^d (d dimensional space)
- A represents mapping from R^M to R^d

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 elliptical region in output space



y=Ax

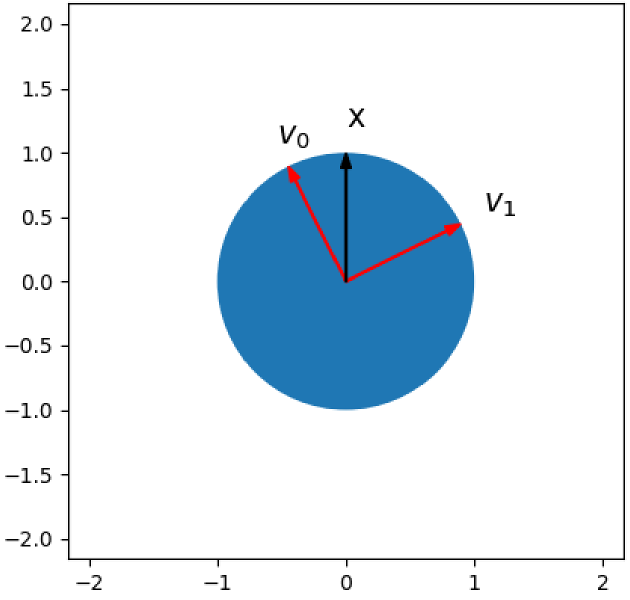


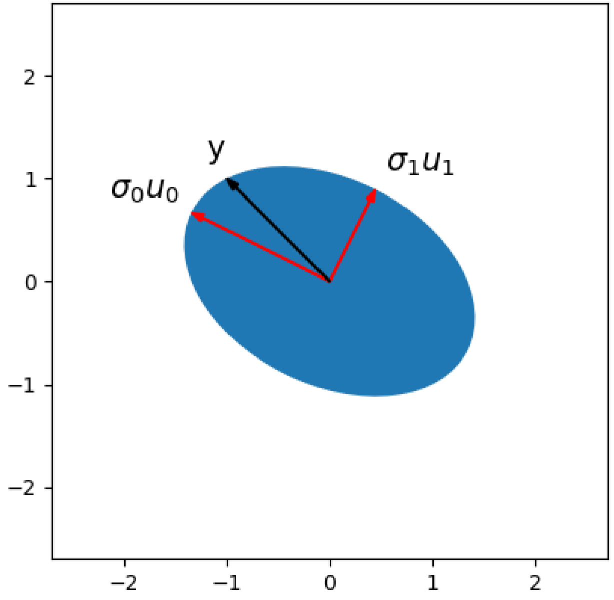
Example: 2x2 matrix

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

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

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





Singular Value Decomposition: Computation

- Eigenvalues of A^TA are squares of singular values of A
- Usually one computes singular values, U, and V using a numerical approach without directly computing A^TA
- No exact formula for SVD so use iterative approach
- Use numpy.linalg.svd() function in numpy with appropriate settings to get compact version of SVD
- For dxd matrix SVD computation requires $O(d^3)$ operations as $d \to \infty$

Singular Value Decomposition: Applications

In this course, we will use SVD for

- Visualization/Animation of contours of normal probability density function for the Gaussian Mixture Model
  - Take SVD of covariance matrix
- Dimension Reduction using Principal Component Analysis
  - Take SVD of feature matrix X

3.4 Singular Value Decomposition DEMO

Jupyter Notebook for demo:

- UnsupervisedML/Examples/Section03/SVD.ipynb

Course Resources at:

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