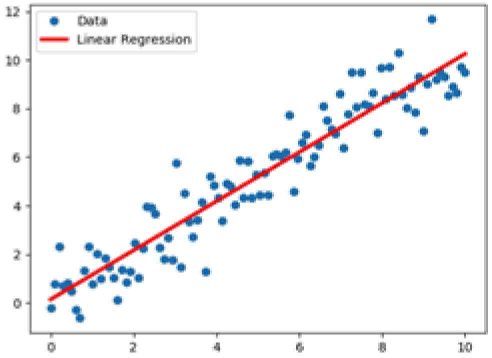
DATA8015 Math Foundation of Data Science

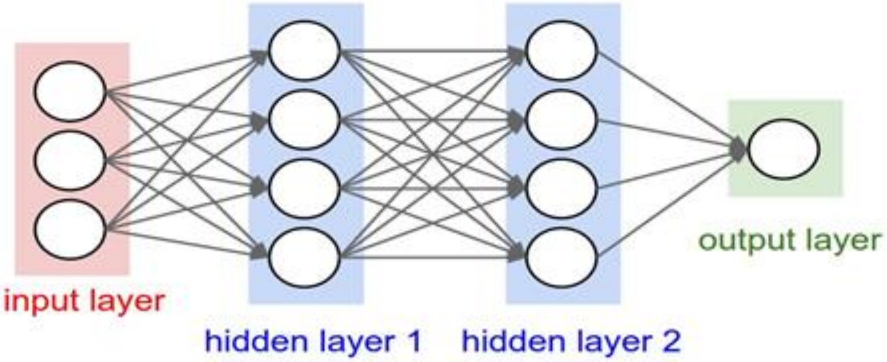
Linear Algebra Basics

Linear Algebra: What is it good for?

- Study of Linear functions: simple, tractable
- In AI/ML: building blocks for **all models**
  - e.g., linear regression; part of neural networks



Hieu Tran

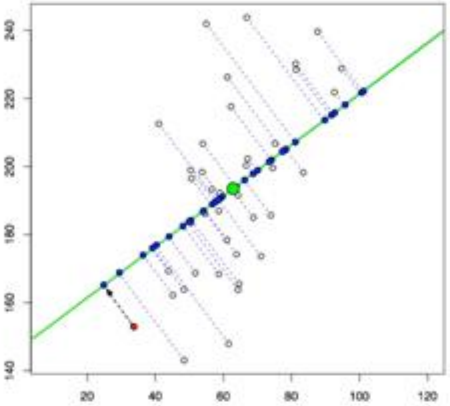


Stanford CS231n

Outline

- Basics: vectors, matrices, operations

- Examples in data science

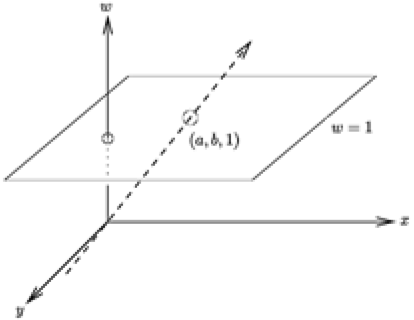


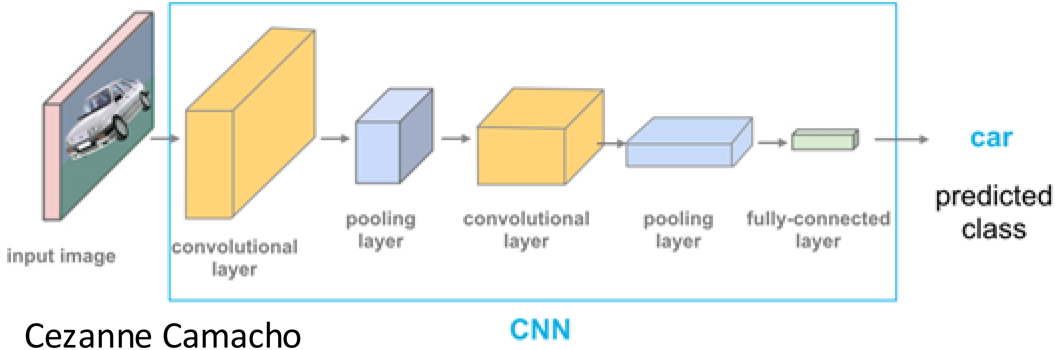
Lior Pachter

Basics: **Vectors**

$$x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix} \in \mathbb{R}^5$$

- Many interpretations
  - List of values (represents information)
  - Point in a space
- Dimension: number of values: $x \in \mathbb{R}^d$
- AI/ML: often use **very high dimensions**:
  - Ex: images!





Cezanne Camacho

Basics: **Matrices**

- Many interpretations
  - Table of values; list of vectors
  - Represent linear transformations
  - Apply to a vector, get another vector

$$A = \begin{bmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \\ A_{31} & A_{33} & A_{33} \\ A_{41} & A_{43} & A_{43} \end{bmatrix}$$

- Dimensions: #rows $\times$ #columns, $A \in \mathbb{R}^{m \times n}$
  - Indexing!

Basics: **Transposition**

- Transposes: flip rows and columns
  - Vector: standard is a column. Transpose: row vector
  - Matrix: go from $m \times n$ to $n \times m$

$$x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \quad x^T = \begin{bmatrix} x_1 & x_2 & x_3 \end{bmatrix}$$

$$A = \begin{bmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \end{bmatrix} \quad A^T = \begin{bmatrix} A_{11} & A_{21} \\ A_{12} & A_{22} \\ A_{13} & A_{23} \end{bmatrix}$$

Matrix & Vector **Operations**

- **Vectors**
  - **Addition:** component-wise
    - Commutative: $x + y = y + x$
    - Associative: $(x + y) + z = x + (y + z)$

$$x + y = \begin{bmatrix} x_1 + y_1 \\ x_2 + y_2 \\ x_3 + y_3 \end{bmatrix}$$

  - **Scalar Multiplication**
    - Uniform stretch / scaling

$$cx = \begin{bmatrix} cx_1 \\ cx_2 \\ cx_3 \end{bmatrix}$$

Matrix & Vector **Operations**

- **Vector products**
  - **Inner product** (e.g., dot product)

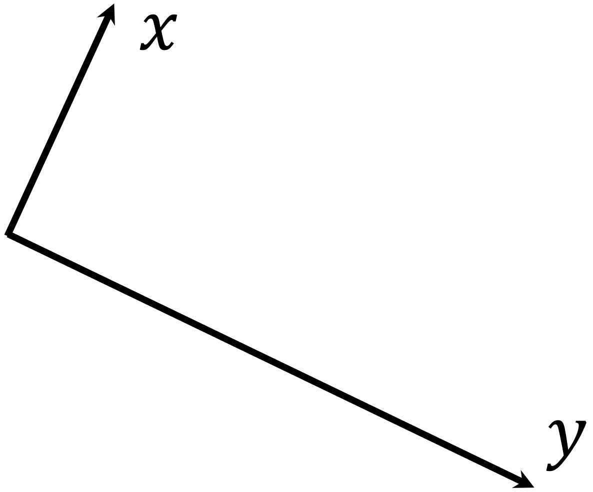
$$< x, y > := x^T y = \begin{bmatrix} x_1 & x_2 & x_3 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = x_1 y_1 + x_2 y_2 + x_3 y_3$$

  - **Outer product**

$$xy^T = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \begin{bmatrix} y_1 & y_2 & y_3 \end{bmatrix} = \begin{bmatrix} x_1 y_1 & x_1 y_2 & x_1 y_3 \\ x_2 y_1 & x_2 y_2 & x_2 y_3 \\ x_3 y_1 & x_3 y_2 & x_3 y_3 \end{bmatrix}$$

Matrix & Vector **Operations**

- x and y are **orthogonal** if $\langle x, y \rangle$=0



- Vector **norms**: "length"

$$\|x\|_2 = \sqrt{\sum_{i=1}^{n} x_i^2}$$

Matrix & Vector **Operations**

- **Matrices:**
  - **Addition:** Component-wise
  - Commutative, Associative

$$A + B = \begin{bmatrix} A_{11} + B_{11} & A_{12} + B_{12} \\ A_{21} + B_{21} & A_{22} + B_{22} \\ A_{31} + B_{31} & A_{32} + B_{32} \end{bmatrix}$$

  - **Scalar Multiplication**
  - “Stretching” the linear transformation

$$cA = \begin{bmatrix} cA_{11} & cA_{12} \\ cA_{21} & cA_{22} \\ cA_{31} & cA_{32} \end{bmatrix}$$

Matrix & Vector **Operations**

- **Matrix-Vector multiplication**
  - Linear transformation; plug in vector, get another vector
  - Each entry in Ax is the inner product of a row of A with x

$$x \in \mathbb{R}^n, A \in \mathbb{R}^{m \times n}$$

$$Ax = \begin{bmatrix} \langle A_{1:}, x \rangle \\ \langle A_{2:}, x \rangle \\ \vdots \\ \langle A_{m:}, x \rangle \end{bmatrix} = \begin{bmatrix} A_{11}x_1 + A_{12}x_2 + \cdots + A_{1n}x_n \\ A_{21}x_1 + A_{22}x_2 + \cdots + A_{2n}x_n \\ \vdots \\ A_{m1}x_1 + A_{m2}x_2 + \cdots + A_{mn}x_n \end{bmatrix}$$

Matrix & Vector **Operations**

Ex: feedforward neural networks. Input *x.*

- Output of layer *k* is

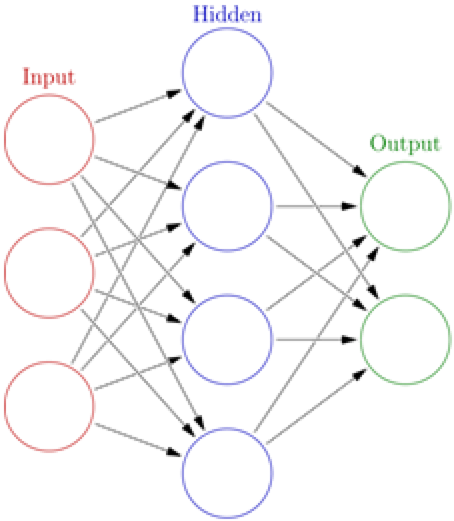
nonlinearity

$$f^{(k)}(x) = \sigma(W_k^T f^{(k-1)}(x)))$$

Output of layer k-1: **vector**

Output of layer k: vector

Weight **matrix** for layer k:
Note: linear transformation!



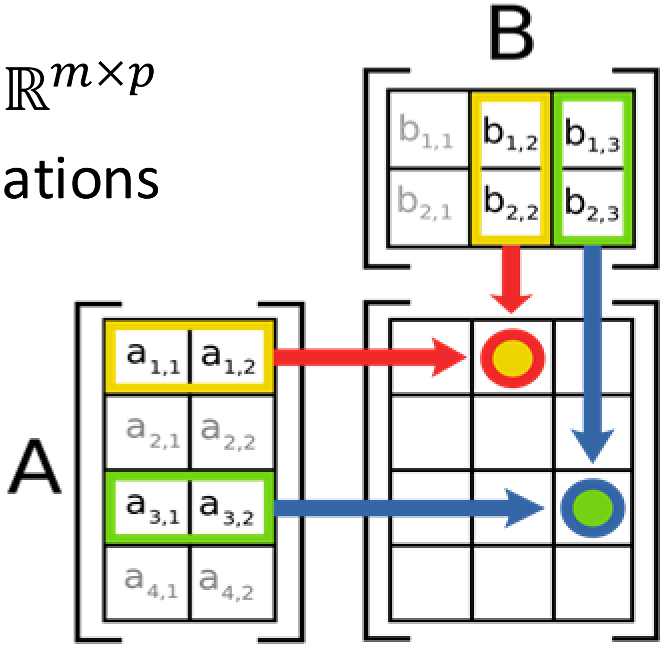
Wikipedia

Matrix & Vector **Operations**

- **Matrix multiplication**
  - $A \in \mathbb{R}^{m \times n}, B \in \mathbb{R}^{n \times p}$, then $AB \in \mathbb{R}^{m \times p}$
  - "Composition" of linear transformations
  - **Not commutative** in general!

$$AB \neq BA$$

  - Lots of interpretations



Wikipedia

Identity Matrix

- Like "1"
- Multiplying by it gets back the same matrix or vector

- Rows & columns are the "**standard basis vectors**" e_i

$$I = \begin{bmatrix} \boxed{\begin{matrix} 1 \\ 0 \\ \vdots \\ 0 \end{matrix}} & \boxed{\begin{matrix} 0 \\ 1 \\ \vdots \\ 0 \end{matrix}} & \begin{matrix} \dots \\ \dots \\ \\ \dots \\ \dots \end{matrix} & \boxed{\begin{matrix} 0 \\ 0 \\ \vdots \\ 1 \end{matrix}} \end{bmatrix}$$

$$\begin{matrix} \downarrow & \downarrow & & \downarrow \\ e_1 & e_2 & & e_n \end{matrix}$$

Matrix Inverse

- If there is a B such that $AB = BA = I$
  - Then A is invertible/nonsingular, B is its **inverse**
  - Some matrices are **not** invertible!

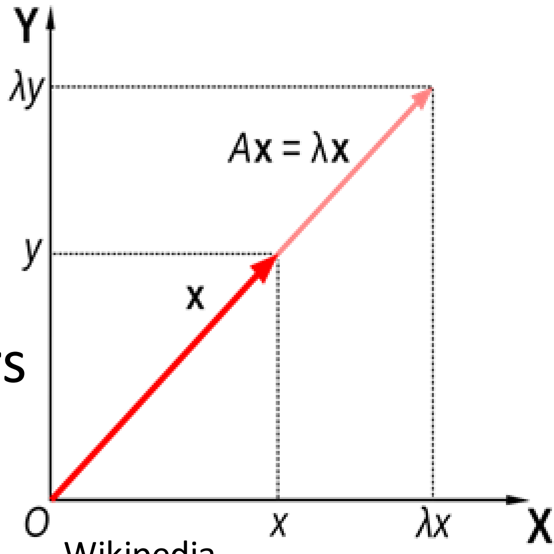
- Notation: A^{-1}

$$\begin{bmatrix} 1 & 1 \\ 2 & 3 \end{bmatrix} \times \begin{bmatrix} 3 & -1 \\ -2 & 1 \end{bmatrix} = I$$

Eigenvalues & Eigenvectors

- For a square matrix A, solutions to $Av = \lambda v$
  - v is a (nonzero) vector: **eigenvector**
  - λ is a scalar: **eigenvalue**

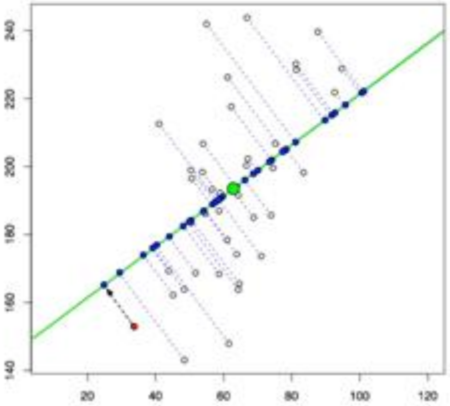
- Intuition
  - Multiplying by A can stretch/rotate vectors
  - Eigenvectors v: only stretched (by λ)



Wikipedia

Outline

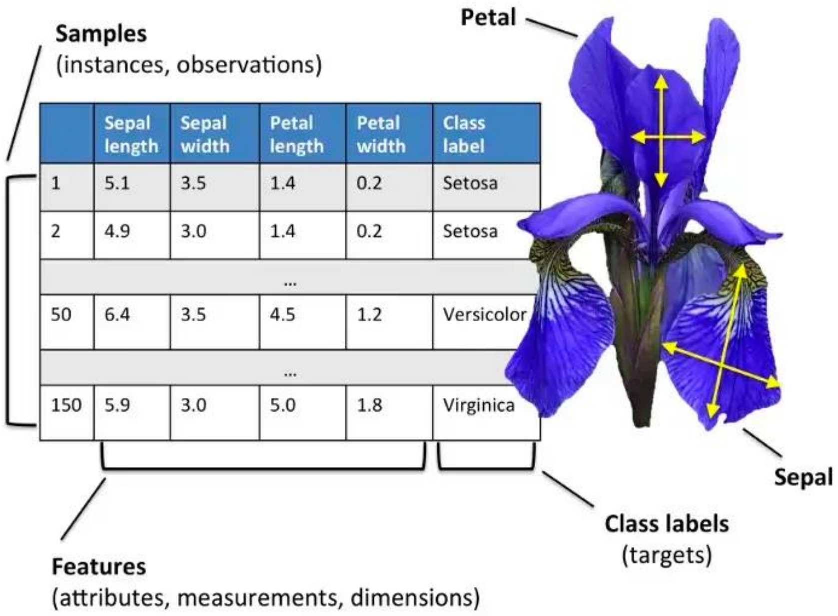
- Basics: vectors, matrices, operations
- Examples in data science



Lior Pachter

Represent Data

- Feature vectors
  - Objects have many relevant features
  - Each object is represented by a feature vector

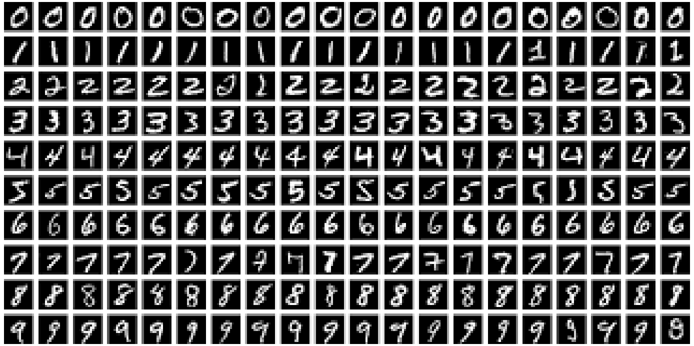


	Sepal length	Sepal width	Petal length	Petal width	Class label
1	5.1	3.5	1.4	0.2	Setosa
2	4.9	3.0	1.4	0.2	Setosa
			...		
50	6.4	3.5	4.5	1.2	Versicolor
			...		
150	5.9	3.0	5.0	1.8	Virginica

Iris dataset. Image from https://eminebozkus.medium.com/exploring-the-iris-flower-dataset-4e000bcc266c

Represent Data

- Images/Videos: Array of Pixels



Represent Data

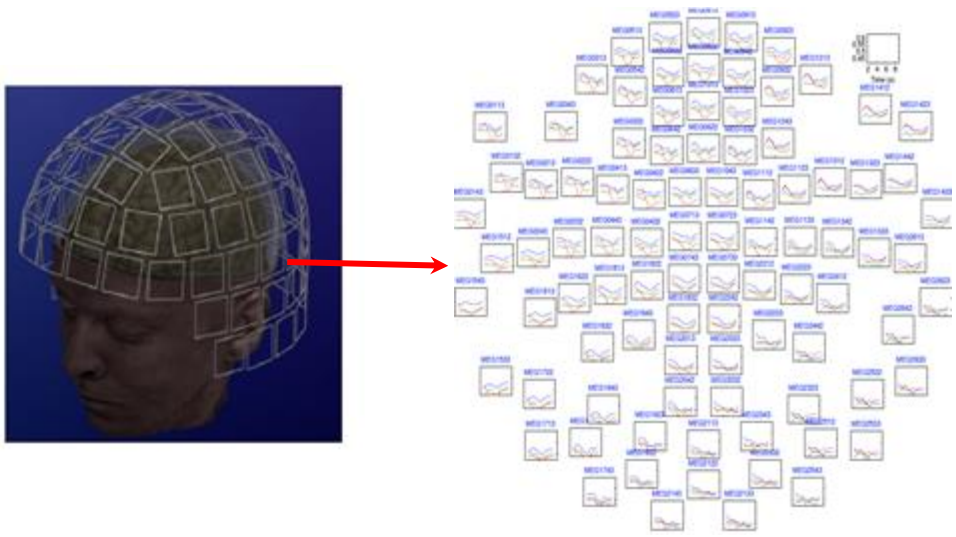
- Netflix surveys: 480189 users x 17770 movies

	movie 1	movie 2	movie 3
Tom	5	?	?
George	?	?	3
Susan	4	3	1
Beth	4	3	?

	movie 1	movie 2	movie 3	movie 4	movie 5	movie 6	movie 7	movie 8	movie 9	movie 10	...	movie 17770
user 1			1		2							3
user 2		2		3	3			4				
user 3							5	3		4		
user 4	2				3			2				2
user 5		4				5			3			4
user 6			2									
user 7			2					4	2	3		
user 8	3	4				4	?					
user 9									3			
user 10			1		2							2
...												
user 480189		4			3			3				

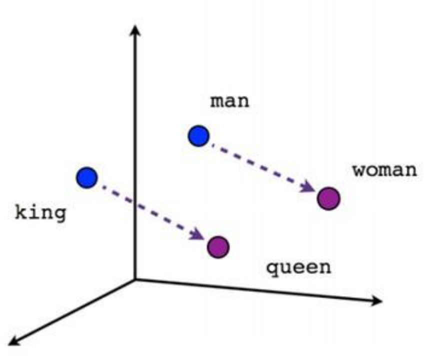
Represent Data

- MEG Brain Imaging: 120 locations x 500 time points x 20 objects

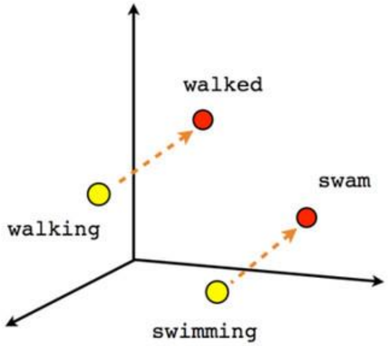


Represent Data

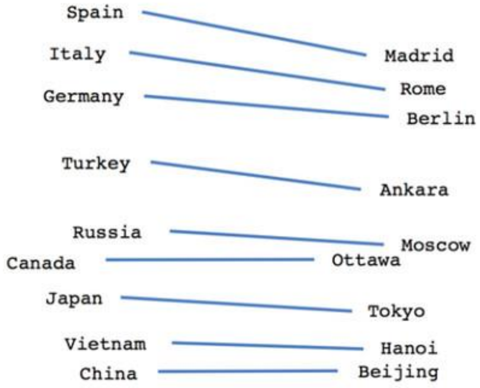
- What about text? Embed as vectors



Male-Female



Verb tense

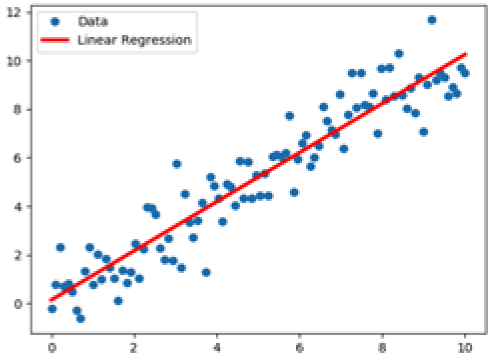


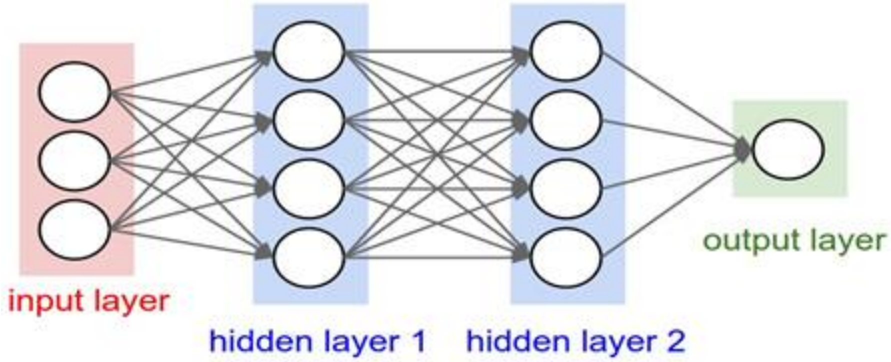
Country-Capital

Efficient Estimation of Word Representations in Vector Space. Tomas Mikolov, Kai Chen, Greg Corrado, Jeffrey Dean. 2013.

Representing Linear Functions

- Linear models
- Part of neural networks





Stanford CS231n