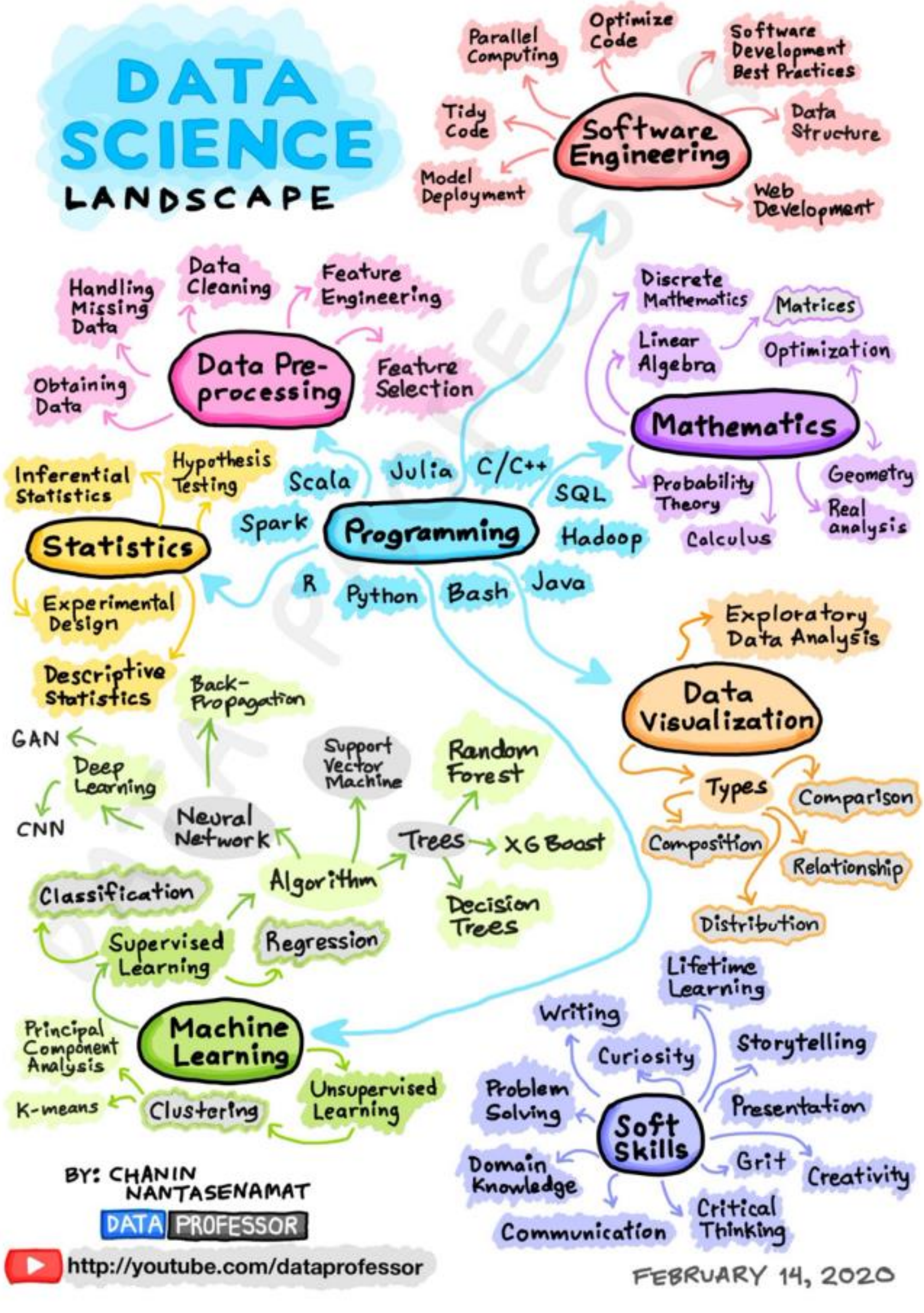

DATA SCIENCE
LANDSCAPE
Software Engineering
Parallel Computing
Optimize Code
Software Development Best Practices
Tidy Code
Data Structure
Model Deployment
Web Development
Data Pre-processing
Handling Missing Data
Data Cleaning
Feature Engineering
Feature Selection
Obtaining Data
Mathematics
Discrete Mathematics
Matrices
Linear Algebra
Optimization
Probability Theory
Calculus
Geometry
Real analysis
Statistics
Inferential Statistics
Hypothesis Testing
Experimental Design
Descriptive Statistics
Programming
Scala
Julia
C/C++
SQL
Spark
Hadoop
R
Python
Bash
Java
Data Visualization
Exploratory Data Analysis
Types
Comparison
Composition
Relationship
Distribution
Machine Learning
Back-Propagation
GAN
Deep Learning
CNN
Neural Network
Support Vector Machine
Random Forest
Trees
XG Boost
Decision Trees
Algorithm
Classification
Supervised Learning
Regression
Principal Component Analysis
Unsupervised Learning
Clustering
K-means
Soft Skills
Lifetime Learning
Writing
Curiosity
Storytelling
Problem Solving
Presentation
Domain Knowledge
Grit
Creativity
Communication
Critical Thinking
BY: CHANIN NANTASENAMAT
DATA PROFESSOR
http://youtube.com/dataprofessor
FEBRUARY 14, 2020

Administrative Issues

1. Professors and students must sign a Consent/asent IRB document
2. A code of conduct for our Institute is also important. (Carpentries Code of Conduct as a reference: https://docs.carpentries.org/topic_folders/policies/code-of-conduct.html)
3. Our Schedule

JUNE Summer Institute 2021 (DS-INTER)				
Monday	Tuesday	Wednesday	Thursday	Friday
21 9 AM – 12:00 PM • A: **IM-RRivera** • B: **ML1-IGriva** 1 PM – 4:00 PM • A: **ML2-IGriva** • B: **S -MEPerez**	22	23	24	25 9- 12:00 PM Group Work 1 – 4:00 PM Group A,B Presentations (15 min x 8 approx 2 hours)
28 9 AM – 12:00 PM • A: **RW-ERodriguez** • B: **V - RAcosta** 1 PM – 4:00 PM • A: **WP-RAcosta** • B: **R – Erodriguez**	29	30	1	2 9- 12:00 PM Group Work 1 – 4:00 PM Group A,B Presentations (15 min x 8 approx 2 hours)

Courses	Speaker
Introduction to DS and R (2 meetings)	**Dr. Alvaro Lecompte**
Math for DS and R (1 meeting)	**Dr. Carmen Caiseda**
Python Programming Part I (3 meetings) **Python Programming Part II Google Collab, Git**	**Prof. Wilson Lozano**

JUNE PRE- Institute 2021 (DS-INTER)				
Monday	Tuesday	Wednesday	Thursday	Friday
7 9 AM – 12:00 PM DS and R **ALecompte**	8 Software Installation and Python Programming I **WLozano**	9 Python Programming I **WLozano**	10 Python Programming I **WLozano**	11
14 9 AM – 12:00 PM DS-Math and R **Carmen Caiseda**	15 DS and R **ALecompte**	16 Python Part II, Google Collab, G **WLozano**	17 Cohort A Presentations!	18

Objectives

1. Introduction to DS and Math
2. Linear Algebra
3. Regression Problem with Linear Algebra
4. Multivariate Calculus - for Machine Learning II
5. R in Google Collab – preview of Wednesday's work

10 Things I wished I knew about Learning Data Science by Chanin Nantasenamat – Thailand BioInformatics Professor

Reference: https://towardsdatascience.com/10-things-i-wish-i-knew-about-learning-data-science-7a30bfb91759

1. **Your DS journey is personal** – embrace the "imposture syndrome". Craft your own list of things to learn and do. Focus on learning 1 new thing a day of the "bucket list".
2. **How to learn DS** – the best way to learn is DOING DS. You can learn from books, videos, podcasts, lectures, teaching and DOING. Apply your new knowledge to your data science project. Reinforce by teaching others = reorganize your thoughts and better understand. (Recommended articles on learning: *Learning From the Feynman Technique*, YouTube video on *The 25 Best Scientific Study Tips*, *Ultralearning* from Scott Young, Josh Kaufman delivered a TED talk and described in his book *The First 20 Hours*
3. **Resource**s for Learning Data Science (Fee vs Free) - For Ex. THIS INSTITUTE!
4. **Why Data Science?** What is my purpose and reason to learn DS? Then you can better understand which area you need to focus on learning first. How will you use it? (exploratory, developing regression/classification/clustering, develop chat bot or recommendation system….). What value can DS bring to YOUR WORK

a. **We can ALL be educated either as DS-Tools …**
    i. **Users**
    ii. **Builders**
    iii. **and a "Translators" between both**

5. **Keep Yourself Accountable and Be Productive! Here are some basic advice for being productive:**
    a. Set aside dedicated time every day (preferably 1–2 hours or at least 45 minutes everyday) that you can spend learning and doing Data science
    b. Avoid distractions (Turn off your phones, avoid checking social media, etc.). If you cannot stop distractions from reaching you then maybe it may be a better idea to move yourself from a distractive environment. This means that you should find someplace quiet where you can put your undivided attention to focus.
    c. Don't procrastinate, don't over think, and *just do it!* (like Nike) To help you overcome this, try applying the 2-minute rule (read this Medium article on How to Stop Procrastinating by Using the '2-Minute Rule') to help keep you in motion.
6. **Embrace Failure and Learn to Love Debugging: "**If you don't fail, you don't learn. It is perfectly okay to get stuck, it is okay to don't understand algorithm X, and it is okay to not know how to debug your failed code. You can take a break to refresh your mind before getting back into tackling your challenge. Sometimes your mind gets clogged and get sluggish and so taking a break may help to rejuvenate and refresh the mind."
7. **Don't Worry About Trying to Learn Everything.** "The only thing that remains constant is change" A new and dynamic field
    a. **Focus on Basics:**
        i. Data wrangling (Python — pandas, R — dplyr)
        ii. Read up on statistics so that you can apply them in your models. For example, applying proper statistics to compare models (parametric vs non-parametric).
        iii. Exploratory data analysis and descriptive statistics for gaining an overview of the data
        iv. Start with building simple and interpretable machine learning models (linear regression, tree-based methods)
        v. Use machine learning approaches that you are confident in using (knowing the math behind it)

    b. ***Focus on the Project and Not on the Technology****:* Don't over think. Overcome the "What language should I learn?" dilemma, choose one and move on.
        i. The Python vs. R vs. Matlab Issue→ use your favorite programming language for DS. What is your Community of Practice using? -- THAT'S YOUR LANGUAGE.

Communities of Practice

- people that care about a domain
- collaborative teaching/learning by working through problems
- interaction and group-work

Why engage in Communities of Practice-DS Education

- DS often left out of core curricula
- DS has a strong existing community
- DS tasks are explicitly coop (eg. Data sharing, open source)
- *practice reinforces conventional learning

ii. Examples of Communities of Practice and DS Hubs

- Software Carpentries **https://cookbook.carpentries.org/**
- eScience Institute – University of Washington https://escience.washington.edu/
- West Big Data Innovation Hub: https://westbigdatahub.org/
- Pangeo Project- Big Data Geoscience
- Academic Data Science Alliance - https://academicdatascience.org/
- DS for Social Good - https://data.berkeley.edu/academics/resources/data-science-education-resources/data-science-ethos-lifecycle
- HOPEFULLY: DS-INTER may develop into one or more communities of Practice

c. *Some clues*
   - *R - FOR STATISTICS COMMUNITY*
   - *PYTHON – FOR ENGINEERS, CS, GENERAL USAGE*
   - *MATLAB – MATH MODELING*

8. Make Your Projects Reproducible

Some of the benefits of making your data science project**s reproducible** are as follows:

a. Others can help you
b. People will propagate an error- http://albertocairo.com/, Book: How Charts Lie. Entrevista a Alberto Caito (data journalist) sobre el **problema de los sesgos ideológicos, religiosos o de otro tipo,** y como afecta la objetividad.
   i. “cuando vemos un gráfico en redes sociales podemos evitar compartirlo inmediatamente simplemente porque el gráfico confirma lo que queremos creer”
   ii. “Una de las razones por la que este ecosistema informativo se está deteriorando se debe a que los lectores creamos cámaras de eco. **Compartimos entre nosotros informaciones y gráficos que ya conocemos o, más peligroso todavía, que confirman lo que queremos saber**.”
   iii. “En definitiva, conocerse a uno mismo y ser más consciente de que esas creencias (ser más de izquierdas o de derechas, por ejemplo) nos llevan muchas veces a engañarnos a nosotros mismos. “
c. Important to be productive in a **Community of Practice**!
d. Save time for you and others: Use “docker containers” and Export your project as Python’s and Conda’s environments, because what works today may not work in 6 months.

```
conda env export > environment.yml
```

9. **Learning Success Starts from Within:** Curiosity, Love the process, Growth Mindset and Grit = "tendency to sustain interest in and effort toward very long-term goals" Angela Duckworth *Grit: The Power of Passion and Perseverance* (YouTube video)

10. **Taking Full Responsibility**

Success is not something you pursue, success is something you become.
—Jim Rohn

"Until you accept responsibility for your life, someone else runs your life."
—Orrin Woodward

"Take full responsibility for what happens to you, it is one of the highest form of human maturity. Accepting full responsibility, it's the day you know you have pass from childhood to adulthood."
—Jim Rohn

"The next best thing to KNOWING, is KNOWING WHERE TO FIND IT"
—Samuel Johnson

Now, take a moment and reflect. Let's start taking accountability and taking full responsibility, you'll be amazed at how much you can achieve in your data science journey. Only if we can be objective and take full responsibility for our actions and lack of progress, will we be empowered to do something about it.

SHARE - Groups of 3:

- How does DS fits into what I am doing or will do?
- What activities you could do at IAUPR that will help build a DS-Community of Practice?

Machine Learning Syllabus

Mathematical concepts to be refreshed: chain rules for derivatives, matrix-matrix product, matrix-vector product, scalar product (inner product, dot product), hyperplanes (generalization of the line), **Euclidean distance, probability, expected values of a random variable, confidence intervals, Bayes theorem.**

Review of Matrix Operations using as Motivation Linear Regression (“Trendlines” in Excel)

(Good Reference: Python Linear Algebra numpy functions: https://towardsdatascience.com/linear-algebra-cheat-sheet-for-deep-learning-cd67aba4526c)

Please notice the use of Mathematics Notation in programming languages. For example the ‘assignment’ operator gives a value to a variable

weight_kg ‘=’ 50

weight_kg ‘←’ 50

The notation for multiple observations (i.e. DATA) is given by Linear Algebra. Data Scientists use (and abuse) linear algebra operations and teach it. BEWARE!! NOT ALL THAT IS PUBLISHED IN THE INTERNET IS HEALTHY. A LOT OF JUNK-FOOD!

(Other internet sites: https://opendatascience.com/linear-algebra-cheat-sheet-for-deep-learning/)

Linear algebra cheat sheet for Deep Learning

BLOG DEEP LEARNING DEEP LEARNING posted by Brendan Fortuner June 12, 2017

Deep Learning 48

Like Dislike

Beginner’s guide to commonly used operations

During Jeremy Howard’s excellent **deep learning course** I realized I was a little rusty on the prerequisites and my fuzziness was impacting my ability to understand concepts like backpropagation. I decided to put together a few wiki pages on these topics to improve my understanding. Here is a very basic intro to some of the more common linear algebra operations used in deep learning.

What is linear algebra?

In the context of deep learning, linear algebra is a mathematical toolbox that offers helpful techniques for manipulating groups of numbers simultaneously. It provides structures like vectors and matrices (spreadsheets) to hold these numbers and new rules for how to add, subtract, multiply, and divide them.

Why is it useful?

It turns complicated problems into simple, intuitive, efficiently calculated problems. Here is an example of how linear algebra can achieve greater speed and simplicity.

```
# Multiply two arrays
x = [1,2,3]
y = [2,3,4]
product = []
for i in range(len(x)):
    product.append(x[i]*y[i])
```

```
# Linear algebra version
x = numpy.array([1,2,3])
y = numpy.array([2,3,4])
x * y
```

After initializing the arrays, the linear algebra approach was 3x faster.

How is it used in deep learning?

Neural networks store weights in matrices. Linear algebra makes matrix operations fast and easy, especially when training on GPUs. In fact, GPUs were created with vector and matrix operations in mind. Similar to how images can be represented as arrays of pixels, video games generate compelling gaming experiences using enormous, constantly evolving matrices. Instead of processing pixels one-by-one, GPUs manipulate entire matrices of pixels in parallel.

GPU = graphics processing unit (faster than CPU, although we are building multi-core).

Vectors

Vectors are 1-dimensional arrays of numbers or terms. In geometry, vectors store the magnitude and direction of a potential change to a point. The vector [3, -2] says go right 3 and down 2. A vector with more than one dimension is called a matrix.

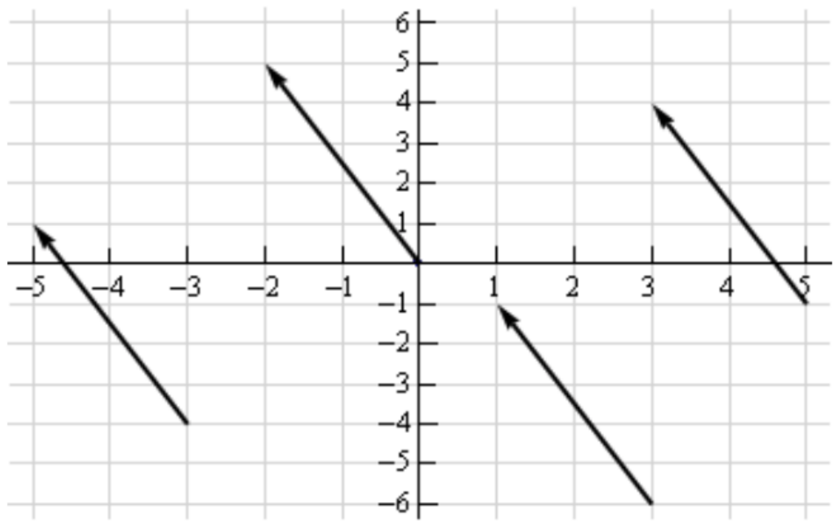
Vector notation

There are a variety of ways to represent vectors. Here are a few you might come across in your reading.

$$v = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix} = \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix} = \begin{bmatrix} 1 & 2 & 3 \end{bmatrix}$$

Vectors in geometry

Vectors typically represent movement from a point. They store both the **magnitude** and **direction** of potential changes to a point. The vector [-2,5] says move left 2 units and up 5 units. **Source**.



v = [-2, 5]

A vector can be applied to any point in space. The vector's direction equals the slope of the hypotenuse created moving up 5 and left 2. Its magnitude equals the length of the hypotenuse.

Elementwise operations

In elementwise operations like addition, subtraction, and division, values that correspond positionally are combined to produce a new vector. The 1st value in vector A is paired with the 1st value in vector B. The 2nd value is paired with the 2nd, and so on. This means the vectors must have equal dimensions to complete the operation.*

$$\begin{bmatrix} a_1 \\ a_2 \\ a_n \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2 \\ b_n \end{bmatrix} = \begin{bmatrix} a_1 + b_1 \\ a_2 + b_2 \\ a_n + b_n \end{bmatrix}$$

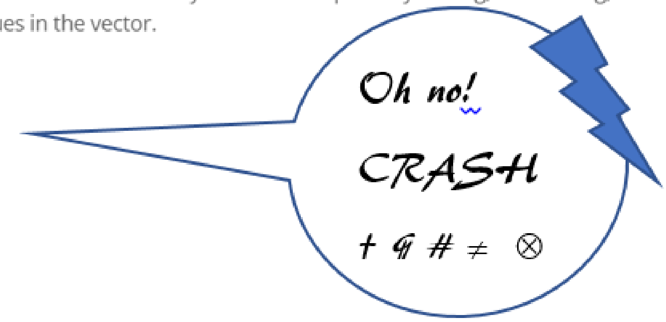
Vector addition

Scalar operations

Scalar operations involve a vector and a number. You modify the vector in-place by adding, subtracting, or multiplying the number from all the values in the vector.

$$\begin{bmatrix}2\\2\\2\end{bmatrix} + 1 = \begin{bmatrix}3\\3\\3\end{bmatrix}$$

Scalar addition



Bad Use of the "Addition" concept, why? These are NOT similar objects. You cannot add a "vector" and a "number or scalar".

CONSEQUENCE:

If I can ADD a vector and a number, then I can add two different objects:

2 Apples + 1 Mango = 3 ?! VS. 2 Apples + 1 Apple = 3 Apples

Pepito has 2 apples and 1 sato dog, how many XYZ Pepito has? Poor Pepito!

2/5 + 1 = 3/ ?!

2x + sin(x) = 3 ?!

We know 2x + sin(x) = 2x + sin(x) BECAUSE WE DO NOT ADD (=COMBINE) DIFFERENT OBJECTS!

Correction:

$$\begin{bmatrix}2\\2\\2\end{bmatrix} + \begin{bmatrix}1\\1\\1\end{bmatrix} = \begin{bmatrix}3\\3\\3\end{bmatrix}$$

Be aware, that our "higher level programming" does not confuse our well known understanding of addition only defined among similar.

Multiplication is Different

Nonetheless, a matrix, a numbers (scalar), and vectors can be **multiplied** in diverse forms. Multiplication is different! I have NO problem with "scalar Multiplication" because multiplication can be defined among any two objects in various way. It "changes the scale" of an object. Typically

2[3,5,7] = [6,10,14]

(2/5)(3) = 6/5

3(sin(x)) = 3sin(x)

Dot product

The dot product of two vectors is a scalar. Dot product of vectors and matrices (matrix multiplication) is one of the most important operations in deep learning.

Using Matrix Notation

$$\begin{bmatrix} a_1 \\ a_2 \end{bmatrix} \cdot \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} = a_1 b_1 + a_2 b_2$$

$$= \boldsymbol{a}^{\mathrm{T}}\mathbf{b} = \begin{bmatrix} a_1 & a_2 \end{bmatrix} \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}$$

```
y = np.array([1,2,3])
x = np.array([2,3,4])
np.dot(y,x) = 20
```

Hadamard product

Hadamard Product is elementwise multiplication and it outputs a vector.

$$\begin{bmatrix} a_1 \\ a_2 \end{bmatrix} \odot \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} = \begin{bmatrix} a_1 \cdot b_1 \\ a_2 \cdot b_2 \end{bmatrix}$$

```
y = np.array([1,2,3])
x = np.array([2,3,4])
y * x = [2, 6, 12]
```

REVIEW OF MATRIX MULTIPLICATION AS THE DOT PRODUCT OF ROW AND COLUMN

IF AB = C, then the element $c_{i,j}$ = [ith-row of A] • [jth-col of B]

$$
\begin{array}{c}
\\ \\ \text{row } i \hookrightarrow \\ \\ \\
\end{array}
\begin{bmatrix}
a_{11} & a_{12} & a_{13} & \cdots & a_{1n} \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
a_{i1} & a_{i2} & a_{i3} & \cdots & a_{in} \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
a_{n1} & a_{n2} & a_{n3} & \cdots & a_{nn}
\end{bmatrix}
\cdot
\begin{array}{c}
\begin{array}{cccccc} & & & \text{column } j & & \end{array} \\
\begin{bmatrix}
b_{11} & b_{12} & \cdots & b_{1j} & \cdots & b_{1n} \\
\vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\
b_{i1} & b_{i2} & \cdots & b_{ij} & \cdots & b_{in} \\
\vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\
b_{n1} & b_{n2} & \cdots & b_{nj} & \cdots & b_{nn}
\end{bmatrix}
\end{array}
=
$$

$$
=
\begin{bmatrix}
c_{11} & c_{12} & \cdots & c_{1j} & \cdots & c_{1n} \\
\vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\
c_{i1} & c_{i2} & \cdots & \boxed{\mathbf{c_{ij}}} & \cdots & c_{in} \\
\vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\
c_{n1} & c_{n2} & \cdots & c_{nj} & \cdots & c_{nn}
\end{bmatrix}
\begin{array}{l}
\\ \\ \text{entry on row } i \\ \text{column } j \\ \\
\end{array}
$$

Also using the SUMMATION NOTATION: $c_{i,j} = a_{i,1}b_{1,j} + a_{i,2}b_{2,j} + \cdots + a_{i,n}b_{n,j} = \sum_{k=1}^{n} a_{i,k}b_{k,j}$

MATRIX ALGEBRA "CHEAT-SHEETS" (SUGGESTIONS?)

https://www.slideshare.net/spoonfeedme/eng1091-cheat-sheet-monash-university

Group Work about Matrix Mathematics

Instructions:

- Click on the Anaconda Navigator
- See the following document to explore Matrix Multiplication First: https://github.com/ccaiseda/ccaiseda/blob/DS-INTER/MatrixMult.ipynb
- Continue working in groups your problems 2-6 for 15 minutes

GROUP WORK

1. What are the dimensions of : $\begin{bmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \end{bmatrix} \cdot \begin{bmatrix} 1 & 2 \\ 5 & 6 \\ 3 & 0 \\ 2 & 1 \end{bmatrix}$

2. What is the matrix product? $\begin{bmatrix} 4 & 3 \\ 3 & 2 \end{bmatrix} \begin{bmatrix} -2 & 3 \\ 3 & -4 \end{bmatrix}$

3. What is the matrix product? $\begin{bmatrix} 3 \\ 5 \end{bmatrix} \cdot \begin{bmatrix} 1 & 2 & 3 \end{bmatrix} =$

4. What is the matrix product? $\begin{bmatrix} a_1 & a_2 & a_3 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix}$

5. The Inverse of $A = \begin{bmatrix} 4 & 3 \\ 3 & 2 \end{bmatrix}$ $or \;\; A^{-1} =$

6. What is the matrix product? $AB \;\; where \;\; A = \begin{bmatrix} 4 \\ 2 \end{bmatrix} \; and \;\; B = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 1 \end{bmatrix}$

Mathematics generalizes from basic principles ("scalability")

From linear equations to systems of linear equations $\rightarrow$ From scalars to matrices

Data is given as vectors that measure a variable, usually huge vectors.

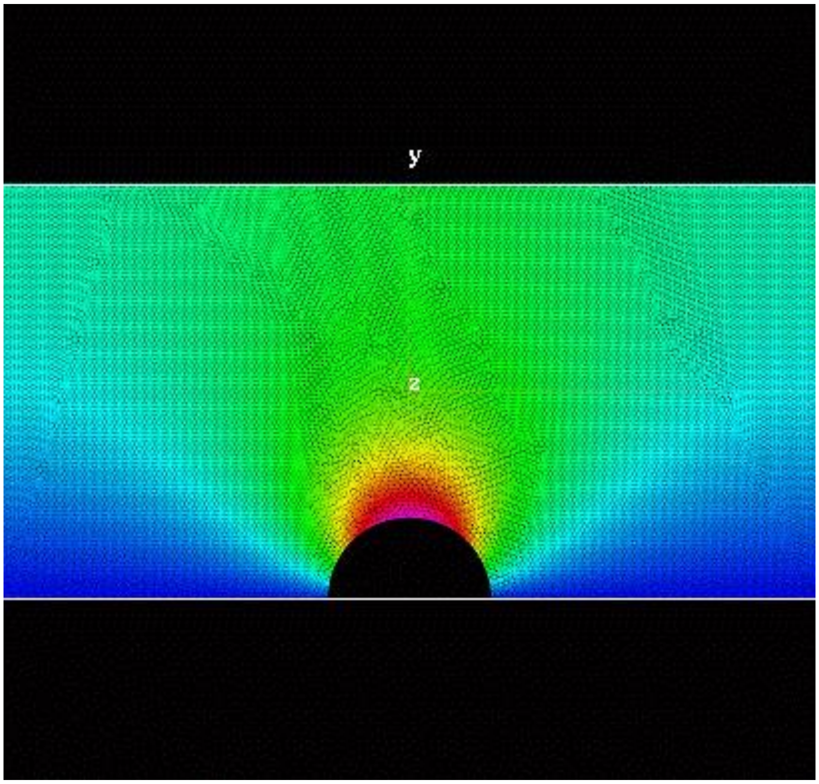
Linear Algebra is an instrument to operate on all data at once using simple notation that generalizes **from 1 to n dimensions**.

1-dimensional x	n-dimensional **x**
$ax = b$ $a \neq 0$ (*cancelation of a using the inverse operation* a^{-1})	$A\mathbf{x} = \mathbf{b}$ *A is* n x n *matrix, non-singular* ($\vert A\vert \neq 0$), b n x 1 *vector* operation A^{-1}
$a^{-1}a\, x = a^{-1}b$ *multiply both sides by constant*	$A^{-1}A\, \mathbf{x} = A^{-1}\, \mathbf{b}$, *multiply both sides from the left by the same matrix*
$1\, x = b/a$ *cancelation of* a *using the inverse operation* a^{-1}	$I\, \mathbf{x} = A^{-1}\, \mathbf{b}$ cancelation of A using the inverse
$x = a^{-1}b$ 1 *is the multiplication identity unique solution*	$\mathbf{x} = A^{-1}\, \mathbf{b}$ I *is the matrix product identity matrix*

Warnings: we cannot divide a matrix but we can find its inverse, and multiplication is NOT commutative: $AB \neq BA$ in general

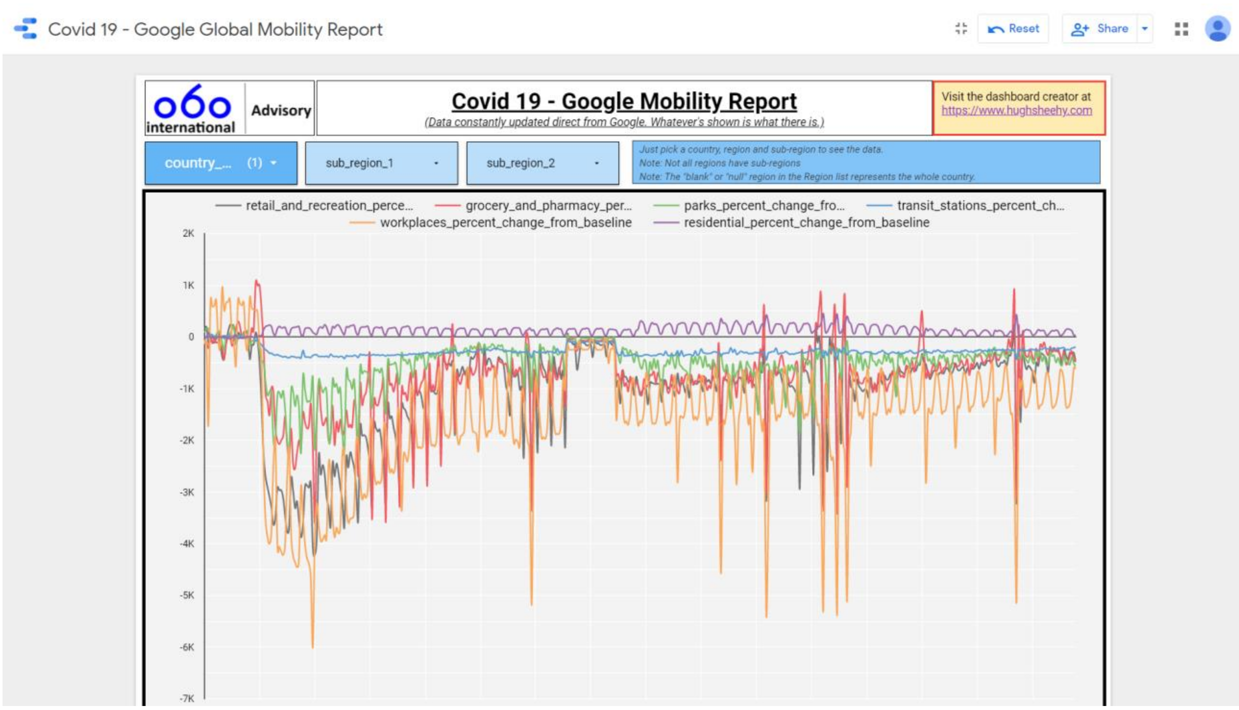
Most numerical methods and work can be reduced as matrix operations on vectors:

- Correlation matrices
- Transformation matrices: such as rotations and animation
- Deep Learning and Machine learning optimization (minimize errors)
- Numerical methods to solve differential equations such as Finite Difference, Finite Elements (see the mesh)



A = 30,000 x 30,000 FEM Matrix to solve the Heat Equation, 2012, CCaiseda- Computational Project

Motivational Example:



Reference: https://datastudio.google.com/u/0/reporting/a529e043-e2b9-4e6f-86c6-ec99a5d7b9a4/page/yY2MB?s=ho2bve3abdM

Use mobility data during Covid in PR, to study the relationship between "stay at home" and "domestic violence" variables.

Regression – (Regression Course)

y = **Xv**+ **e** , where **X** correlation matrix that consist n data of k-independent variables (prior domestic violence index, percentage of families staying at home per city) to estimate/predict the value of the "domestic violence index" in n-cities= data points **y** = $[y_1, y_2, \ldots y_n]^T$ during Lockdown. Find vector **v** that best fits the data, with **e** is the approximation error we want to minimize (assume **e** = **0**).

Equation (linear regression without error term)	**Dot Product (inner, scalar product) Notation**	**Vector Form of a linear system of equations (Call X = A)**
$y_1 = a + x_{11} b_1 + x_{12} b_2 + \ldots + x_{1,k} b_k$	$y_1 = \begin{bmatrix} 1 & x_{1,1} & \cdots & x_{1,k} \end{bmatrix} \begin{bmatrix} a \\ b_1 \\ \vdots \\ b_k \end{bmatrix}$	**y** = **Av**
$y_2 = a + x_{21} b_1 + x_{22} b_2 + \ldots + x_{2,k} b_k$	$y_2 = \begin{bmatrix} 1 & x_{2,1} & \cdots & x_{2,k} \end{bmatrix} \begin{bmatrix} a \\ b_1 \\ \vdots \\ b_k \end{bmatrix}$	
$\vdots$	$\vdots$	
$y_n = a + x_{n1} b_1 + x_{n2} b_2 + \ldots + x_{n,k} b_k$	$y_n = \begin{bmatrix} 1 & x_{n,1} & \cdots & x_{n,k} \end{bmatrix} \begin{bmatrix} a \\ b_1 \\ \vdots \\ b_k \end{bmatrix}$	

We want to find the linear coefficients k-dimensional vector $\mathbf{b} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_k \end{bmatrix}$, and a scalar intercept a. Let our variable be vector $\mathbf{v} = [a \;\; \mathbf{b}]^T = \begin{bmatrix} a \\ b_1 \\ \vdots \\ b_k \end{bmatrix}$ (Also transpose is denoted as $[a \;\; \mathbf{b}]'$)

Then define the general problem in Matrix Form of the linear form without error:

y = A **v**

where $\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$ $A = \begin{bmatrix} 1 & x_{1,1} & \cdots & x_{1,k} \\ 1 & x_{2,1} & \cdots & x_{2,k} \\ \vdots & \vdots & \vdots & \vdots \\ 1 & x_{n,1} & \cdots & x_{n,k} \end{bmatrix}$, $\mathbf{v} = \begin{bmatrix} a \\ b_1 \\ \vdots \\ b_k \end{bmatrix}$,

$n \times 1$ $n \times (k+1)$ $(k+1) \times 1$

Note that A is not square, (n = no. of data points, k = No. of independent variables)!

Nonetheless for any m x n matrix A, the **Transpose** is A^T (or A')   is n x m.   Then we can get a square matrix by multiplying (on any side) by the transpose:

The product of right side: AA' =   m x m

The product of left side: A'A = n x n

Consider this 4x3 matrix.

$$\mathbf{X} = \begin{bmatrix} a_1 & b_1 & c_1 \\ a_2 & b_2 & c_2 \\ a_3 & b_3 & c_3 \\ a_4 & b_4 & c_4 \end{bmatrix}$$

Now consider the matrix product, X'X.

$$\begin{bmatrix} a_1 & a_2 & a_3 & a_4 \\ b_1 & b_2 & b_3 & b_4 \\ c_1 & c_2 & c3 & c_4 \end{bmatrix} \begin{bmatrix} a_1 & b_1 & c_1 \\ a_2 & b_2 & c_2 \\ a_3 & b_3 & c_3 \\ a_4 & b_4 & c_4 \end{bmatrix}$$

The result (product) is a square matrix, whose elements are,

$$\begin{bmatrix} \sum a_i^2 & \sum a_i b_i & \sum a_i c_i \\ \sum b_i a_i & \sum b_i^2 & \sum b_i c_i \\ \sum c_i a_i & \sum c_i b_i & \sum c_i^2 \end{bmatrix}$$

Why square matrices n = m?

- To find inverse of a matrix A = n $\times$ m , and have a unique solution for a system of equations, (n = number of equations or data points, m = number of independent variables)
    - If n > m , we have more equations than variables (overdetermined problem, infinite or no-solution)
    - If n < m, then we have more variables than equations (infinite solutions)
- A^{-1} requires that the determinant of the matrix |A| or det(A) $\neq$ 0
    - When is the determinant |A| = 0?
        - When one column is a multiple of other (one variable is dependent on another, and therefore it is redundant: ex. x1 = 2x2 , then if x2 = 3 → x1 = 6) ,

- Or one row is a multiple of another, (have a redundant equation)
- Practice: Python code to find the inverse **y** of the matrix **x** $= \begin{bmatrix} \mathbf{1} & \mathbf{2} \\ \mathbf{3} & \mathbf{4} \end{bmatrix}$

```
import numpy as np

x = np.array([[1,2],[3,4]])
y = np.linalg.inv(x)
print x
print y
print np.dot(x,y)
```

NOTATION PROBLEMS FOR QUADRATIC Equations $v^T A v$

If A is n x n, define a system of QUADRATIC Equations as **y =** $\mathbf{v^T}$ **Av**	Obtained as the dot product of $\mathbf{v^T}$ and	**Av**
$y_1 = x_{11} b^2_1 + x_{12} b_1 b_2 + \dots + x_{1n} b_1 b_n$	$\begin{bmatrix} b_1 & b_2 & \dots & b_n \end{bmatrix}$	$\begin{bmatrix} x_{11} b_1 + x_{12} b_2 + \dots + x_{1n} b_n \\ x_{21} b_1 + x_{22} b_2 + \dots + x_{2,n} b_n \\ \vdots \\ x_{n1} b_1 + x_{n2} b_2 + \dots + x_{n,n} b_n \end{bmatrix}$
$y_2 = x_{21} b_2 b_1 + x_{22} b^2_2 + \dots + x_{2n} b_2 b_n$		
$\vdots$		
$y_n = x_{n1} b_n b_1 + x_{n2} b_n b_2 + \dots + x_{nn} b^2_n$		

Eucledian Distance: For n-dimensional points in Cartesian coordinates

A. k-nearest neighbor – (Machine Learning Part I)

p = ($p_1, p_2, \dots p_n$), **q** = ($q_1, q_2, \dots q_n$), **the distance between p and q is defined**

d(p,q) = d(q,p) = $\sqrt{(q_1-p_1)^2+(q_2-p_2)^2+(q_3-p_3)^2+\cdots+(q_n-p_n)^2}$

Or $\sqrt{\sum_{k=1}^{n}(q_k-p_k)^2}$ (Also known as Pythagorean Formula)

One dimension $n = 1$: d(p,q) = $\sqrt{(q-p)^2} = |q-p|$

Two dimensions $n = 2$: **d(p,q)** = $\sqrt{(q_1-p_1)^2+(q_2-p_2)^2}$

Motivation: Pattern Recognition Dissimilarities

Calculus for Machine Learning Algorithms Part II

A. The Chain Rule

Usage: Rule for derivation of a composition of functions f(u)

1. Explains how to compute the derivative of a composition of two functions. f ο u = f(u), where u=u(x)
   a. functions is a relationship between two variable. Example: How does the distance (d) changes when time changes (t) is a function: d(t)
   b. Derivative measures the *Instant rate of change (vs. average rate of change)* among the variables. Examples:
   - Velocity = instantaneous (=limit) change of distance over time
   - Slope of a tangent line to a curve = change of the function curve at a point

Example of the Chain Rule	Derivative using the Chain Rule
f(x) = (sin x)^2	f ’(x) = 2 (sin x)· cos(x)
Color Coding Functions from the **outermost** to “**innermost**” of x : the “**quadratic**” → “**trigonometric**” → x . The composition says: that “**green** changes by **purple** that changes by red.”	
Let **u** = **sin** (x) be function of x Then f(x) = (**sin** x)^2 can be rewritten as f(u) = (u)^2	General Change Rule Formula Derivative of f(u) is **f ’(u) u’** ”
To apply the Chain Rule we need the derivatives (References: derivative tables) **u** = **sin** (x) → **u’** = **cos**(x) f(u) = (u)^2 → **f ’(u) = 2 u**	Then in our example: **f ’(u)** · **u’** = **2 u** · **cos**(x) = **2** (**sin** x)· **cos**(x)
The Chain Rule answers the final question: How does **f changes with respect to** x **f (u (x))**	Answer: “derivative of **f with respect to u times the derivative of u with respect to** x”

The Chain Rule

df/dx = df/du*du/dx

also

f ’(x) = df/dx = f ’(u)*u’

Reference:

https://www.khanacademy.org/math/ap-calculus-bc/bc-differentiation-2-new/bc-3-1a/v/chain-rule-introduction?modal=1

B. Higher Order Derivatives Multi-variable Derivatives: $f(x_1, x_2, \ldots, x_k)$ - (Support Vector Machines and Neural Networks- Machine Learning Part II)

The Gradient ∇f = Vector Defined as: $\begin{bmatrix} \frac{\partial f}{\partial x_1} \\ \frac{\partial f}{\partial x_2} \\ \vdots \\ \frac{\partial f}{\partial x_k} \end{bmatrix}$	A Vector that gives the direction of **maximum** change of the function over each variable (for **minimum** use the opposite sign)
The Hessian H = a Square Matrix (k x k) Defined as $\begin{bmatrix} \frac{\partial^2 f}{\partial x^2_1} & \frac{\partial^2 f}{\partial x_1 \partial x_2} & \ddots & \frac{\partial^2 f}{\partial x_1 \partial x_k} \\ \frac{\partial f}{\partial x_2 \partial x_1} & \frac{\partial f}{\partial x^2_2} & \vdots & \frac{\partial f}{\partial x_2 \partial x_k} \\ \vdots & \vdots & \vdots & \vdots \\ \frac{\partial f}{\partial x_k \partial x_1} & \frac{\partial f}{\partial x_k \partial x_2} & & \frac{\partial f}{\partial x^2_k} \end{bmatrix}$	A Matrix that gives the direction of **maximum** change of the gradient vector over each variable (for **minimum** use the opposite sign) Remember the relationship between first ("increase-> decrease" the f) and second derivatives ("concave downward" of the f as it changes from "increase → decrease")

The HESSIAN in neural Networks Optimization (the derivative of the Gradient)
https://stackoverflow.com/questions/23297090/how-calculating-hessian-works-for-neural-network-learning

Random Variables X – (Statistics and Inference)

- Random variables are numeric outcomes resulting from random processes.
    - Examples: X = pick a card from the deck, X = the height of a random person, X = the mean of a random sample
    - Not random: constants, height of Lebron James, yesterday's Max temperature

A. EXPECTED VALUE E[X]

E[X] = expectation in probability, the expected value of a random variable X (X a function), "is a generaltization of a **weighted average**, and intuitively the "**arithmetic mean**" of a large number of independent realizations of the variable." (https://en.wikipedia.org/wiki/Expected_value)

A random variable will vary around its expected value in a way that if you take the average of many, many draws, the average of the draws will approximate the expected value, getting closer and closer the more draws you take.

Let X be a random variable with a finite number of finite outcomes $x_1, x_2, \ldots, x_k$ occurring with probabilities $p_1, p_2, \ldots, p_k$, respectively. The **expectation** of X is defined as

$$\mathrm{E}[X] = \sum_{i=1}^{k} x_i\, p_i = x_1 p_1 + x_2 p_2 + \cdots + x_k p_k.$$

For equi-probable outcomes for X, then E[X] = μ_x the arithmetic mean.

B. Bayes Theorem – (Naïve-Bayes Classifiers in Machine Learning-Part I)

Bayes's theorem is stated mathematically as the following equation:[3]

$$P(A \mid B) = \frac{P(B \mid A) P(A)}{P(B)}$$

where A and B are events and $P(B) \neq 0$.

- $P(A \mid B)$ is a conditional probability: the likelihood of event A occurring given that B is true.
- $P(B \mid A)$ is also a conditional probability: the likelihood of event B occurring given that A is true.
- $P(A)$ and $P(B)$ are the probabilities of observing A and B respectively; they are known as the marginal probability.

Reference: (https://en.wikipedia.org/wiki/Bayes%27_theorem)

THE BAYESIAN MODELING PARADIGM SHIFT

Translate Bayes' theorem:

$$\text{P(hypothesis | data)} = \frac{P(data\ |hypothesis)P(hypothesis)}{P(data)}$$

Therefore Bayes rule tells us how to do inference about hypotheses from data.

Reference: http://mlg.eng.cam.ac.uk/zoubin/talks/lect1bayes.pdf

Introduction to R in Collab Links

- **Instructions**
- **My experiment https://github.com/ccaiseda/ccaiseda/blob/main/FirstR_Python.ipynb**
- **More R? Try the following instructions**
  - **Run Rstudio or use google collab in the previous experiments**

```r
#Libraries needed
library(tidyverse)

# Simulate an election poll with only two parties, with probability of
being a Democrat equal to p = 0.45 (1 – p for Republicans). The size
of a poll is N = 1000, and determine the mean (x_hat).  Repeat more
than once and check and compare the value of x_hat to show that the
“mean” of a poll is a Random Variable.
p <- 0.45  #the proportion of Democrats in the population
N <- 1000   #Sample_size

# Simulate ONE poll of size N and determine x_hat in the for loop 10
times
x <- sample(c(0,1), size=N, replace=TRUE, prob=c(1-p,p))
x_hat <- mean(x)

# Use a for-loop to repeat 10 times the finding the mean and saving in
the vector x
times <- 1:10  #repeat 10 times
for (ii in times)
{x <- sample(c(0,1), size=N, replace=TRUE, prob=c(1-p,p))
x_hat <- mean(x)
print(x_hat)
}
# Much better and advanced instructions using “replicate” function
that repeats the instructions B times, similar to a loop

p = 0.45
B = 10  #number of replicates
N <- 1000  #sample size per replicate
x_hat <- replicate(B,{
   x <- sample(c(0,1), size=N, replace = TRUE, prob =c(1-p,p))
   print(mean(x))
})
```

Observe why the mean of different samples is a Random Variable

References

TEXTBOOK

Rafael Irizarry's Textbook: https://rafalab.github.io/dsbook/

DS Training

- https://datacarpentry.org/ Building a "Community of Interest"
- Data Camp
- EdX, Coursera
- **MIT OpenCourseWare** (longer than 45 minutes).

DS for Social Good –

- University of Washington eScience Institute
- UC Berkeley https://data.berkeley.edu/academics/resources/data-science-education-resources/data-science-ethos-lifecycle

Communities/Groups of Interest

- Software Carpentries **https://cookbook.carpentries.org/**
- West Big Data Innovation Hub: https://westbigdatahub.org/
- Pangeo Project- Big Data Geoscience
- Towards Data Science Community https://towardsdatascience.com/
    - https://towardsdatascience.com/tagged/puerto-rico
    - https://towardsdatascience.com/mathematics-for-data-science-e53939ee8306
    - https://towardsdatascience.com/10-things-i-wish-i-knew-about-learning-data-science-7a30bfb91759

Stanford Linear Algebra Review with Vector Calculus

- http://cs229.stanford.edu/section/cs229-linalg.pdf

Correlation Reference

- https://www.youtube.com/watch?v=4EXNedimDMs

Linear Regression

- https://www.youtube.com/watch?v=ZkjP5RJLQF4
- http://faculty.cas.usf.edu/mbrannick/regression/regma.htm

http://philender.com/courses/multivariate/notes/sscp2.html

Naive-Bayes Classifier

- **https://en.wikipedia.org/wiki/Naive_Bayes_classifier**
- http://mlg.eng.cam.ac.uk/zoubin/talks/lect1bayes.pdf

YouTube Channels

- YouTube Channel Brandon Foltz Statistics 101, https://www.youtube.com/watch?v=ZkjP5RJLQF4
- YouTube for Siraj Ravel Discrete Mathematics https://www.youtube.com/watch?v=LGt4PE7-ATI

AP Calculus Khan Academy and BC Calculus Khan Academy

1) Power Rule – Khan Academy

   Justifying the Power Rule - graphical approach

2) Basic derivative Rules

3) Worked example: Derivative of sin(x) and cos(x) - show in desmos.com

4) Derivative of e^x

5) Derivative of ln(x)

APPENDIX: Linear Algebra Problem to find the Regression Coefficients

Linear Systems of Equations can be solved using the Inverse of a square matrix B →

$$\mathbf{y} = A\,\mathbf{v}$$

We multiply-left by A'

$$A'\mathbf{y} = A'A\,\mathbf{v}$$

A'A = B a square matrix, we can find the Inverse B^{-1}

$$A'\mathbf{y} = B\,\mathbf{v}$$

$$B^{-1}A'\mathbf{y} = (B^{-1}B)\,\mathbf{v}$$

$$B^{-1}A'\mathbf{y} = I\,\mathbf{v}$$

I the Identity square matrix (similar properties as number 1)

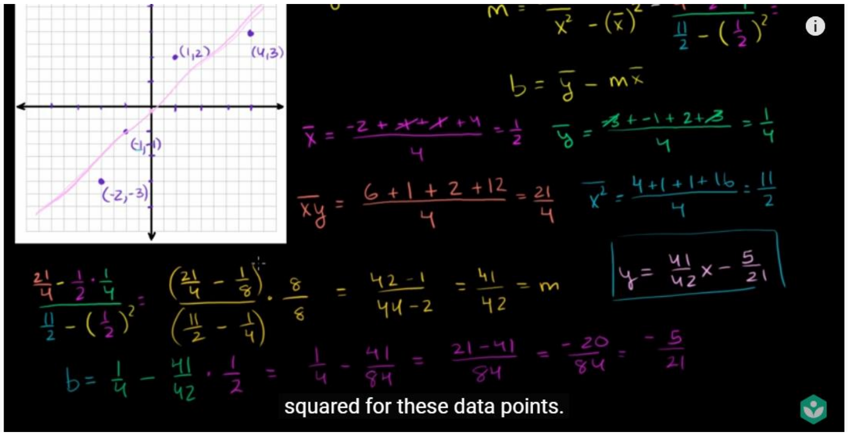
$$B^{-1}A'\mathbf{y} = \mathbf{v}$$

The solution to find the linear trendline/regression coefficients using Matrix Algebra is

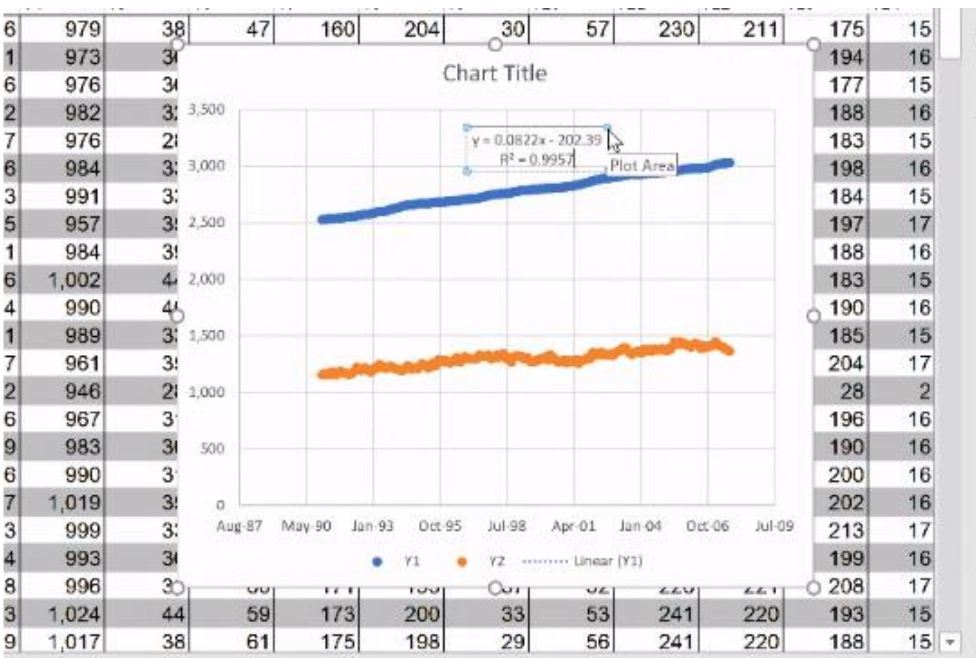
$$\boxed{\mathbf{v} = B^{-1}A'\,\mathbf{y}}$$

$$\mathbf{v} = \begin{bmatrix} a \\ b_1 \\ \vdots \\ b_k \end{bmatrix} = B^{-1}A'\,\mathbf{y}$$

Compare with other “messy” multiple formula-based solutions



Reference: http://faculty.cas.usf.edu/mbrannick/regression/regma.htm



Find the **b vector of the “Indicadores Economicos” Excel Trendline, for the date serial number = x , y1.**

Using the formula: b = $[X'X]^{-1} X'y$, * Uses date serial number where Jan/1/1900 = 1.