Computational Methods for Linguists

Ling 471

Olga Zamaraeva (Instructor)
Yuanhe Tian (TA)

05/11/21

Reminders
and announcements

- Assignment 4 published
  - due May 25
- Assignment 3 sample solution
  - Canvas —> Files
  - **Do not distribute**
- Entertaining video on the “reading list” for May 18
  - Ribeiro et al. 2020
    - Best paper at ACL conference!
    - (The paper itself is more substantial than the video but less fun.)

May 20	Working with linguistic corpora	TBA	
May 25	Visualization and Communication	TBA	Assignment 4
May 27	Visualization and Communication	TBA	Blogs 5
June 1	Presentations		
June 3	Presentations		
June 8			Assignment 5

From class syllabus

Today is going to be quite mathematical...
I promise we will talk about linguistics some!
(At the end of quarter)

Why so mathematical?

- This is a course on **CompLing**
  - ...**without** prerequisites
- **Realistically**, CompLing **today** is all **machine learning**
  - (**Tomorrow** this may change... But today that's the case)
- ML is all **math** :/
- **All** that math requires **multiple** prerequisites
- **Our goal**: **Start** thinking about **some** of the **underlying concepts**
- Become (**somewhat**) better at **actually using** ML **packages**
  - it's a fine way to start!
  - but we can't be too **ambitious**
  - (hence no exams etc)

Plan for today

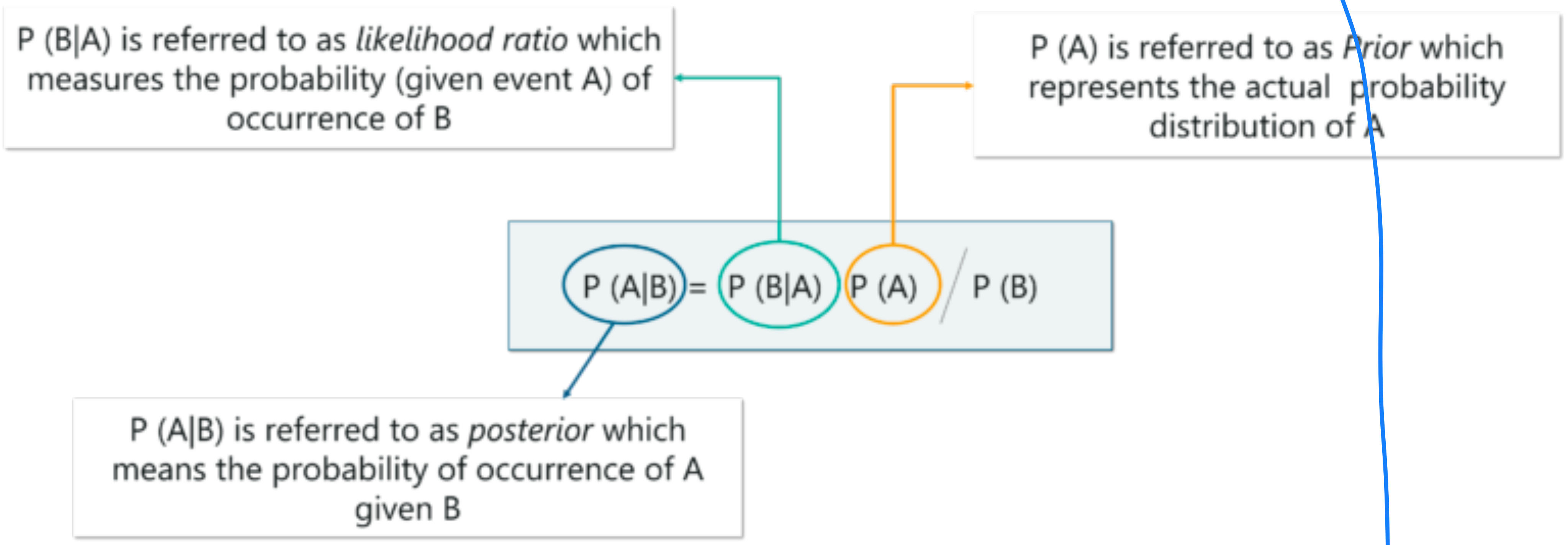
- Bayes Rule problem recap
- More dataframes!
  - access and manipulation
- Matrices and matrix multiplication
  - a “crash course” on linear algebra…
- Machine Learning:
  - Linear regression
  - A case for matrix multiplication
    - …and for **knowing** what matrices’ **dimensions** are!

Bayes Theorem

a classic example: solution

$$P(A \mid B) = \frac{\overset{0.8}{P(B \mid A)} \cdot \overset{0.01}{P(A)}}{\underset{0.1}{P(B)}} = 0.078$$

The Bayes Theorem



$P(B \mid \bar{A})$

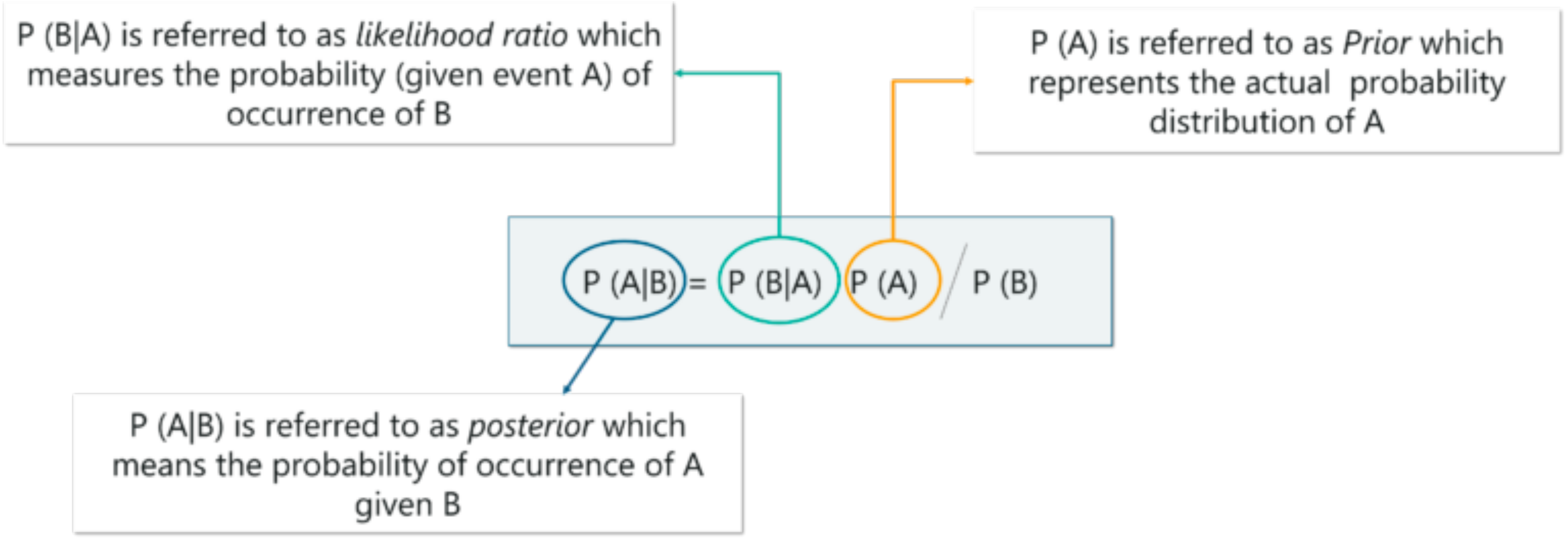
- Suppose:
  - 1% of population have cancer
  - 80% of tests detect it correctly while 20% of tests fail to detect it ("false negative")
  - 9.6% of tests detect it when it is not there ("false positive") while 90.4% correctly return negative
- Q: If you get a positive result, what is the probability of you having the disease?
  - Hint: "P(B) is the P(positive test). But P(positive test) is **not directly given** to you!
    - **Positive test outcome** means: [the test is positive AND person has cancer] **OR** [the test is positive and there is NO cancer!]
    - Use the marbles example: P(**two** events) is similar to P(**two** marbles)

$$P(B) = P(B, A) + P(B, \bar{A}) = P(B \mid A) \cdot P(A) + P(B \mid \bar{A}) \cdot P(\bar{A}) = 0.8 \times 0.01 + 0.096 \cdot 0.99 = 0.1$$

Bayes Theorem
a classic example: solution

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

The Bayes Theorem



- Suppose:
  - 1% of population have cancer
  - 80% of tests detect it correctly while 20% of tests fail to detect it ("false negative")
  - 9.6% of tests detect it when it is not there ("false positive") while 90.4% correctly return negative
- Q: If you get a positive result, what is the probability of you having the disease?
  - Hint: "P(B) is the P(positive test). But P(positive test) is **not directly given** to you!
    - **Positive test outcome** means: [the test is positive AND person has cancer] **OR** [the test is positive and there is NO cancer!]
    - Use the marbles example: P(**two** events) is similar to P(**two** marbles)

- **Need P(B)**
- **P(B) =**
  - **P(positive test)*P(have disease)**
    - **OR**
  - **P(positive test)*P(don't have disease)**
- **P(B) = 0.8*0.01 + 0.097*0.99 = 0.10304**
- **=>**
- **P(disease|positive) = 0.8* 0.01/0.10304 = 0.0776 = 7.8%**

More dataframes
with pandas

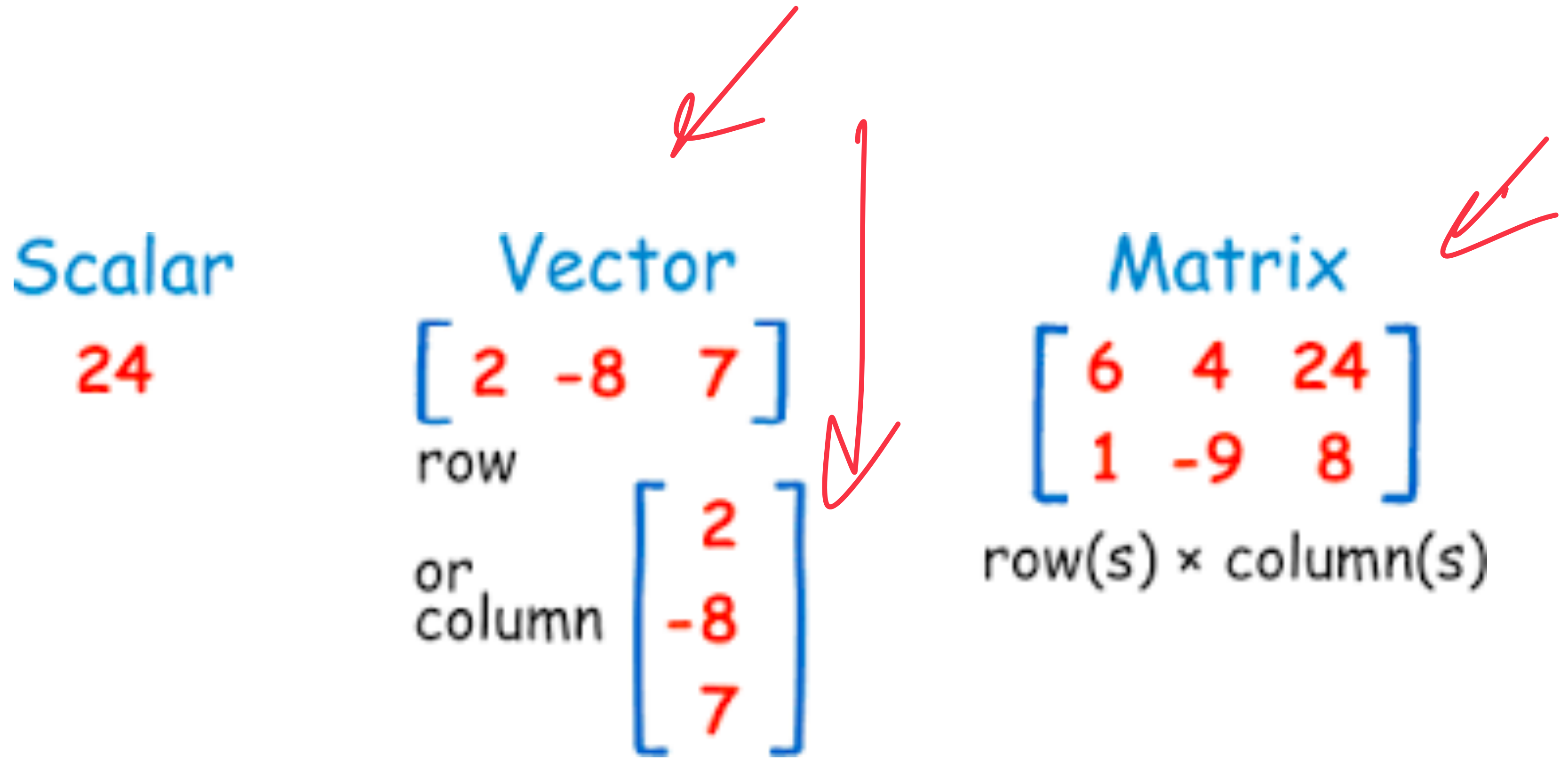
- Once a dataframe is created:
  - it can be accessed and manipulated
  - columns and cells can be accessed directly
    - without iteration!
  - dataframes can be concatenated
    - assuming the number of columns is the same
  - columns can be renamed, etc
- Crucially, matrix multiplication!

```
   label                                               text
0      1  For a movie that gets no respect there sure ar...
1      1  Bizarre horror movie filled with famous faces ...
2      1  A solid if unremarkable film Matthau as Einste...
3      1  Its a strange feeling to sit alone in a theate...
4      1  You probably all already know this by now but ...
```

pandas demo: more dataframes

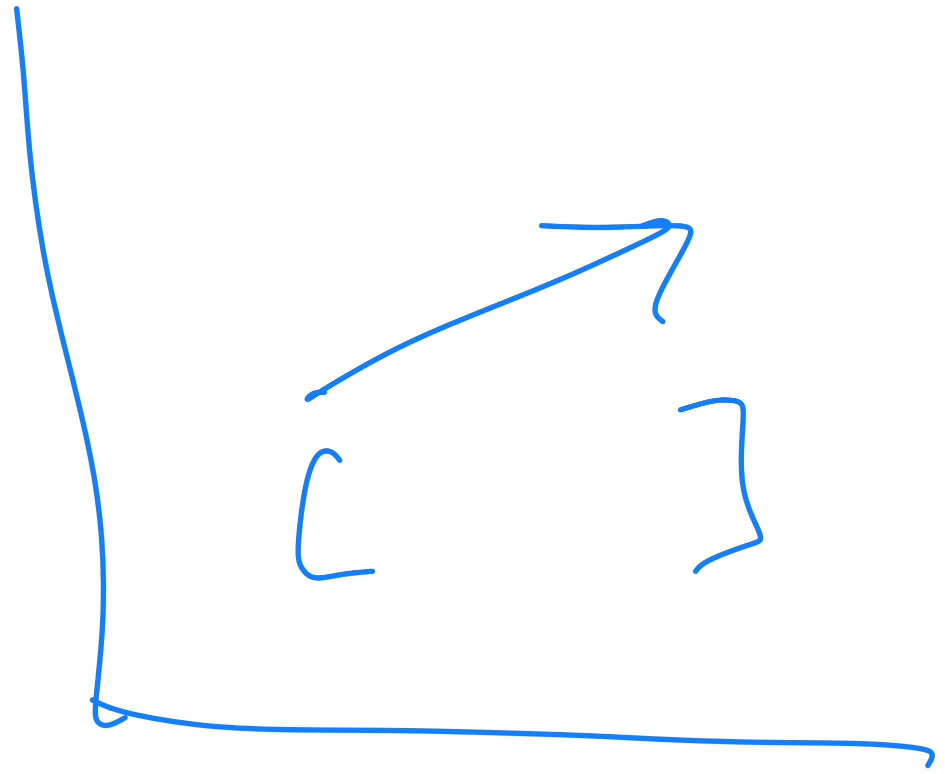
Linear Algebra

Linear Algebra



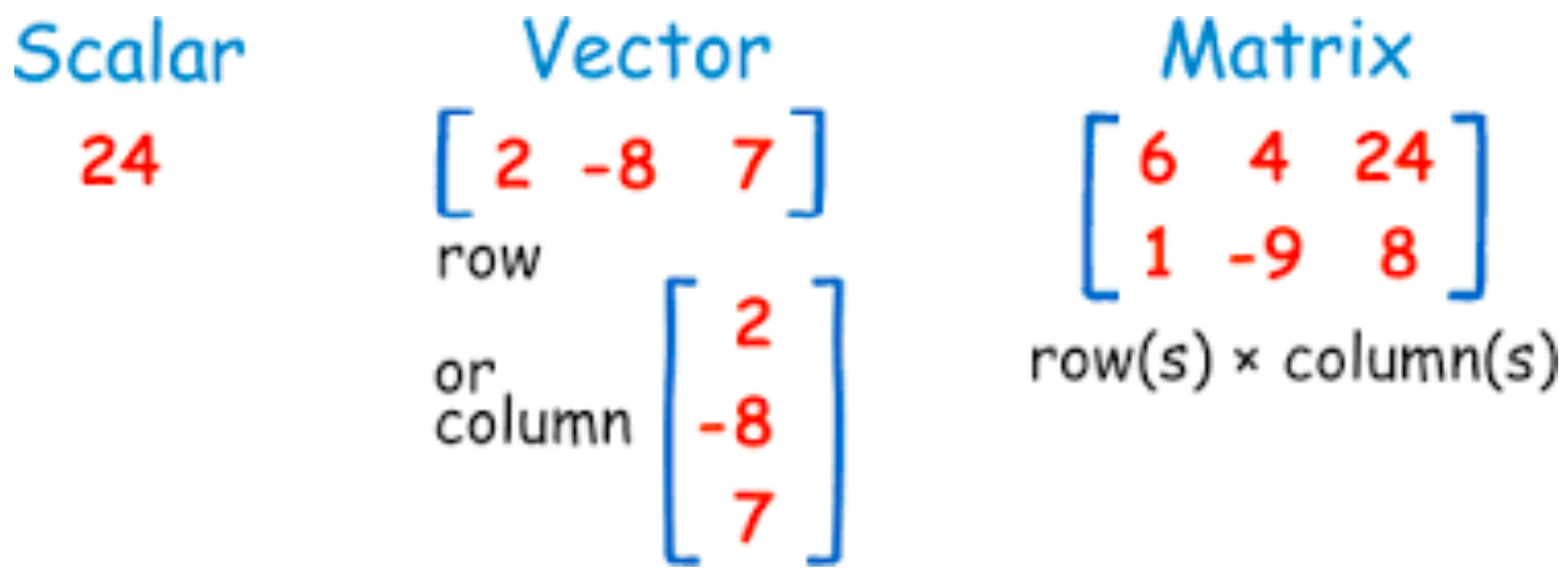
https://www.mathsisfun.com/algebra/scalar-vector-matrix.html

- A branch of mathematics
  - ...which deals in **vectors** and **matrices** :)
- **Vector:** a row of values ("scalars")
- **Matrix**: a bunch of vectors
  - aka a **table**
  - which has **rows** and **columns**



Linear Algebra

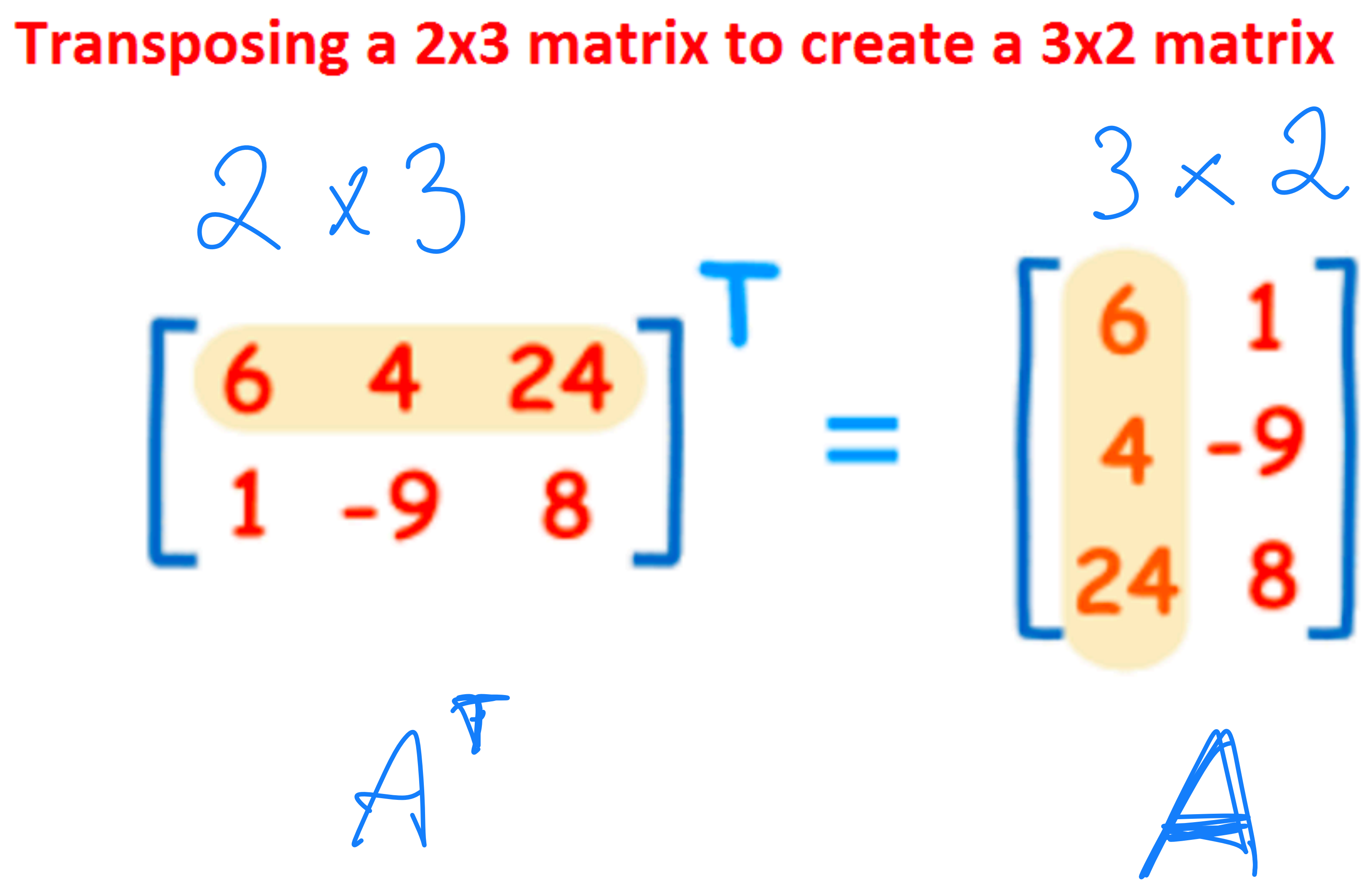
- Linear Algebra is pretty abstract
- Idea: **perform mathematics** on **entire matrices**
  - multiply them
  - prove **theorems** about complex expressions involving them
  - etc.



https://www.mathsisfun.com/algebra/scalar-vector-matrix.html

Linear Algebra

Sample definitions



- Matrix Transpose
  - columns and rows flipped
  - note the ^T notation...
  - if a matrix is denoted by a random variable X...
    - X^T

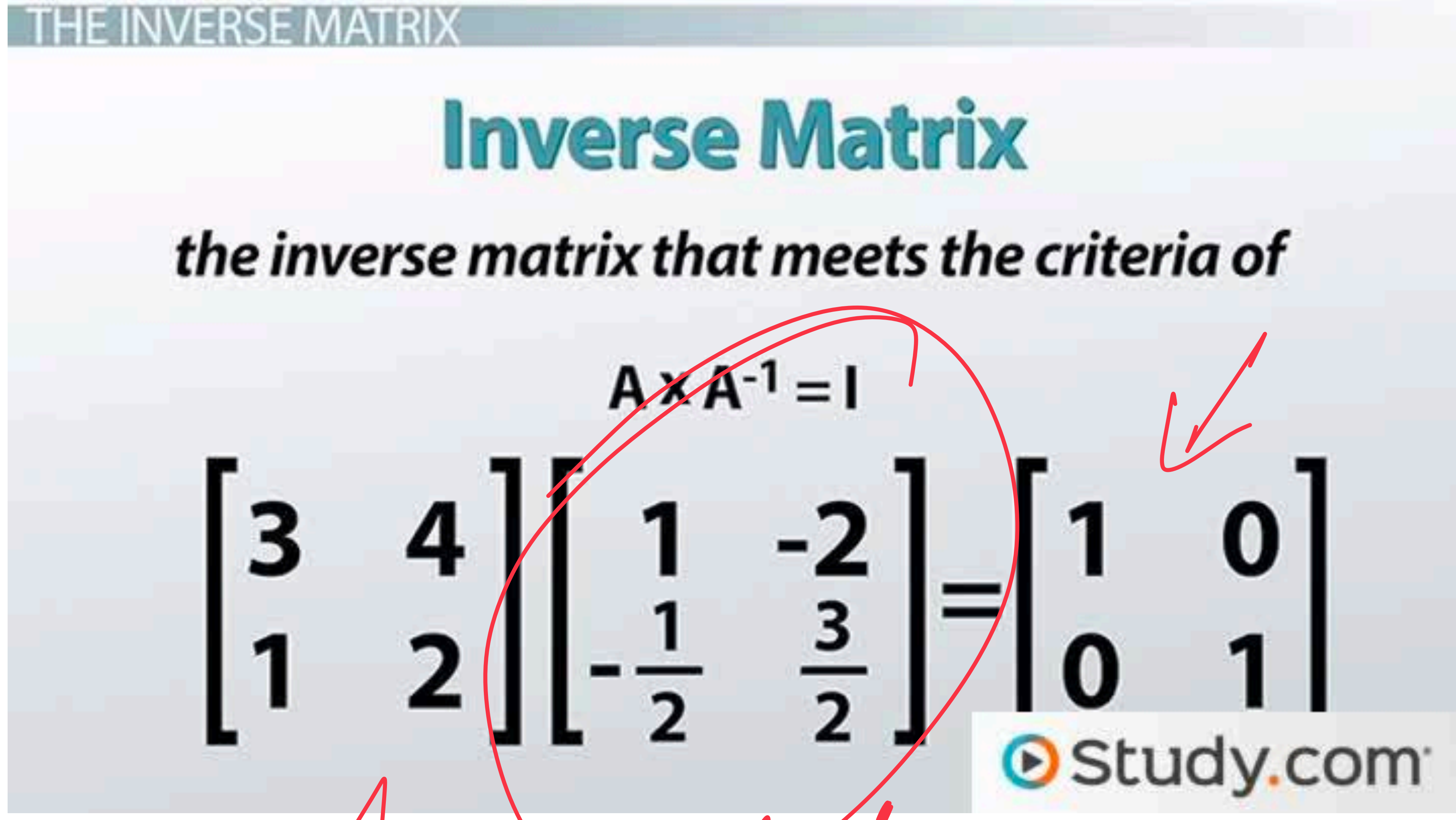
Linear Algebra

Sample definitions

- Matrix Inverse
  - Identity Matrix:
    - 1s on the diagonal, 0s elsewhere
  - **A**^-1 is a matrix such that its product with **A** is equal to the Identity Martrix…
    - Yes, pretty abstract!
    - But kind of like: 8 * 1/8 = 1
      - 1/8 = 8^-1
    - Similarly, **A** * **A**^-1 = **I**(dentity matrix)

Transposing a 2x3 matrix to create a 3x2 matrix

$$\begin{bmatrix} 6 & 4 & 24 \\ 1 & -9 & 8 \end{bmatrix}^T = \begin{bmatrix} 6 & 1 \\ 4 & -9 \\ 24 & 8 \end{bmatrix}$$



https://study.com/academy/lesson/how-to-solve-inverse-matrices.html

Matrix multiplication

linear algebra

- Definition:
  - **(But for now, look more at the example —>)**
  - For two matrices A ans B, with dimensions NxM and MxK:
    - The product is a **NxK** matrix
    - where each cell is a dot product of the i-th row of A and i-th column of B
    - N and K may be different but M must match

Example: The local shop sells 3 types of pies.

- Apple pies cost **$3** each
- Cherry pies cost **$4** each
- Blueberry pies cost **$2** each

1 x 4 [3, 4, 2]

And this is how many they sold in 4 days:

[83]

	Mon	Tue	Wed	Thu
Apple	13	9	7	15
Cherry	8	7	4	6
Blueberry	6	4	0	3

Now think about this ... the **value of sales** for Monday is calculated this way:

Apple pie value + Cherry pie value + Blueberry pie value

$3×13 + $4×8 + $2×6 = $83

So it is, in fact, the "dot product" of prices and how many were sold:

($3, $4, $2) • (13, 8, 6) = $3×13 + $4×8 + $2×6
= $83

We **match** the price to how many sold, **multiply** each, then **sum** the result.

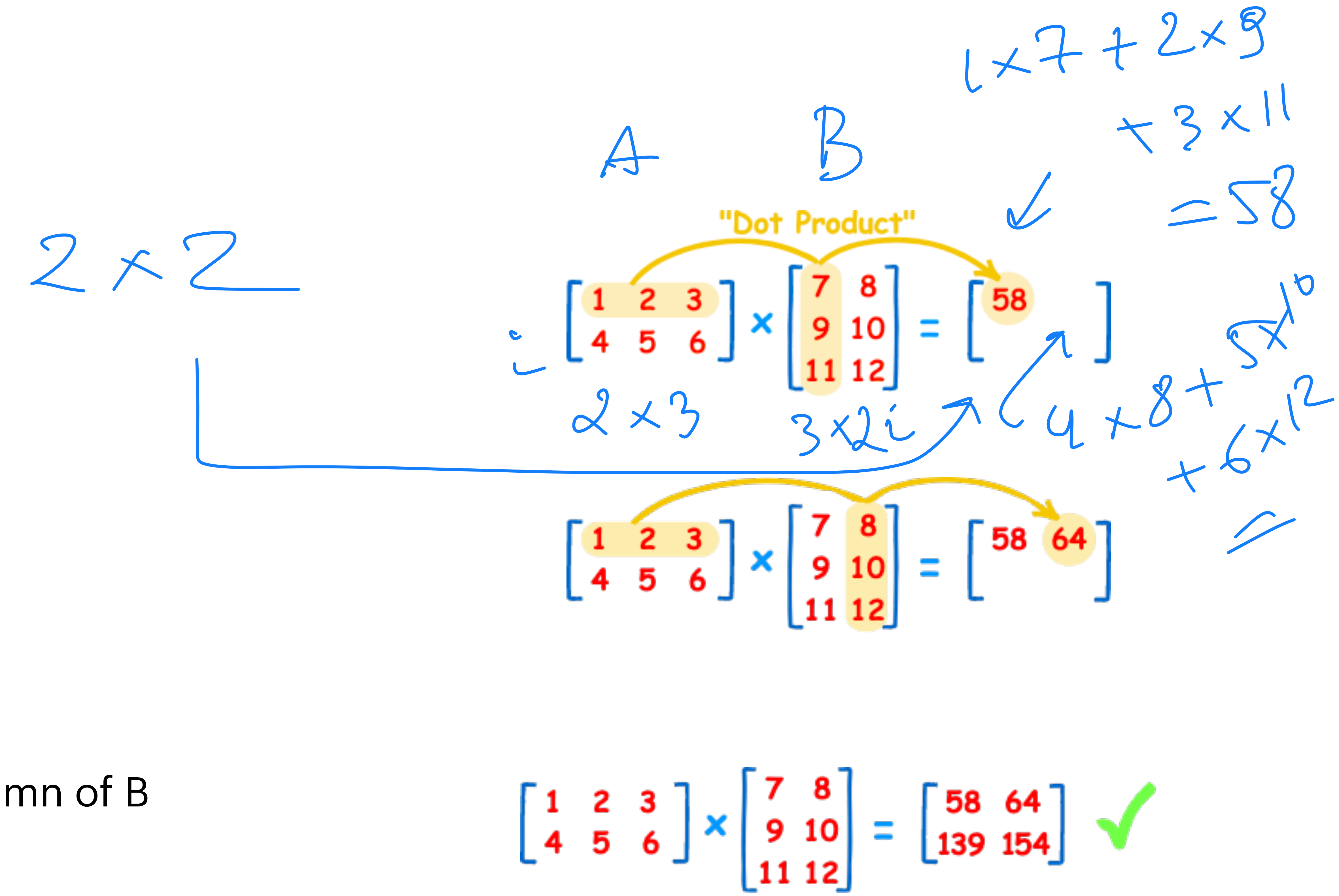
1x3 [ ] • 3x4 [ ] = 1x4 [ ]

https://www.mathsisfun.com/algebra/matrix-multiplying.html

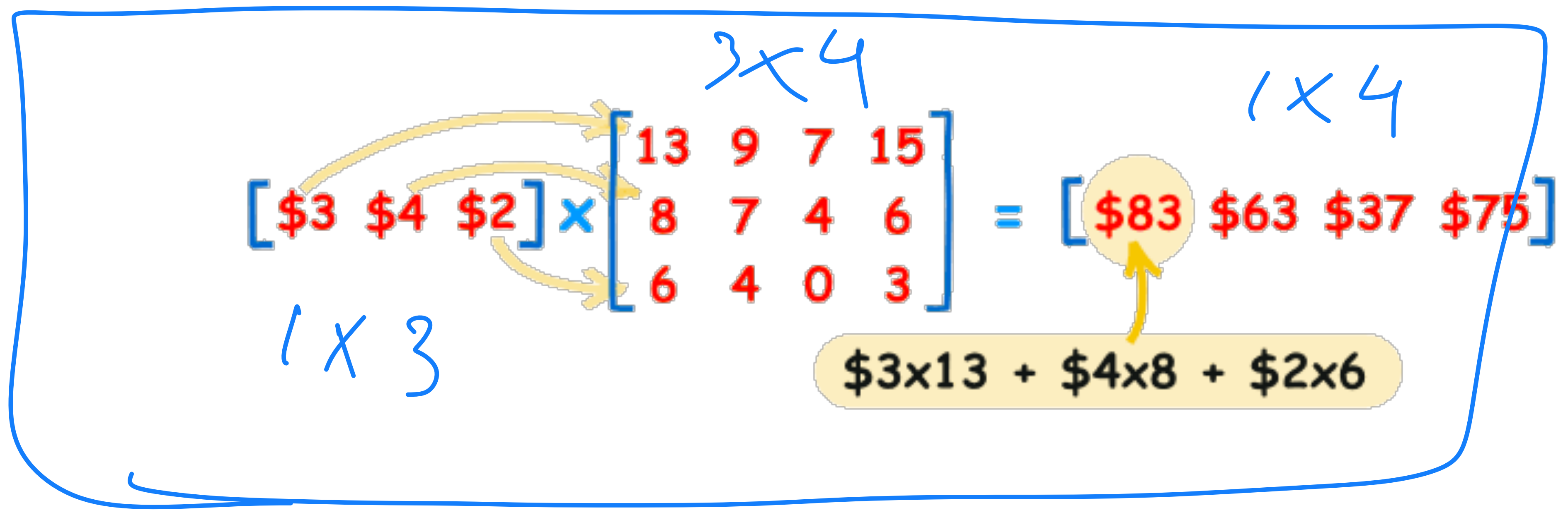
Matrix multiplication

linear algebra

- Definition:
  - For two matrices A ans B, with dimensions Nx**M** and **M**xK:
    - The product is a **NxK** matrix
    - where each cell is a dot product of the i-th row of A and i-th column of B
    - N and K may be different but M must match



https://www.mathsisfun.com/algebra/matrix-multiplying.html

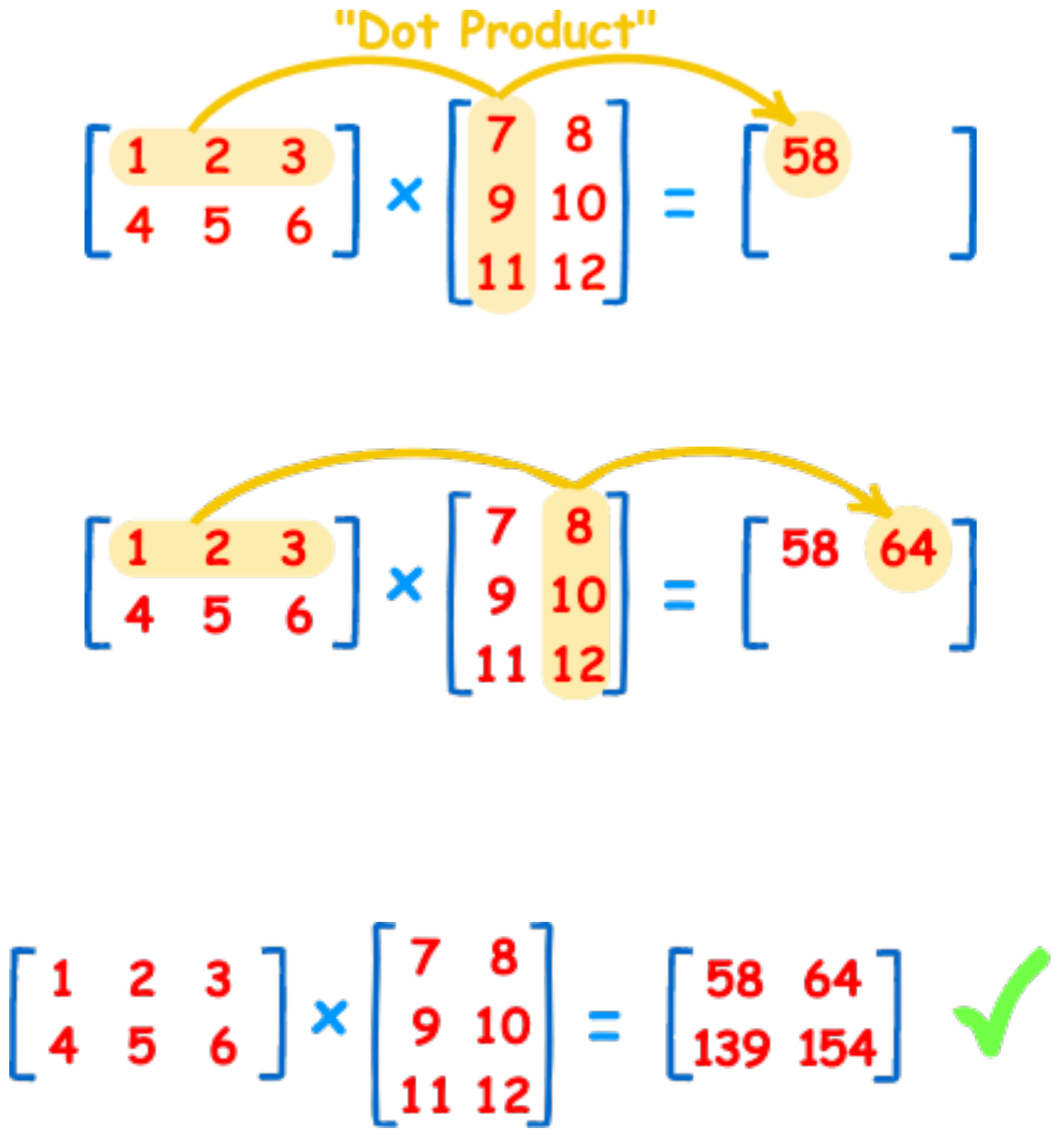


Matrix multiplication in python demo

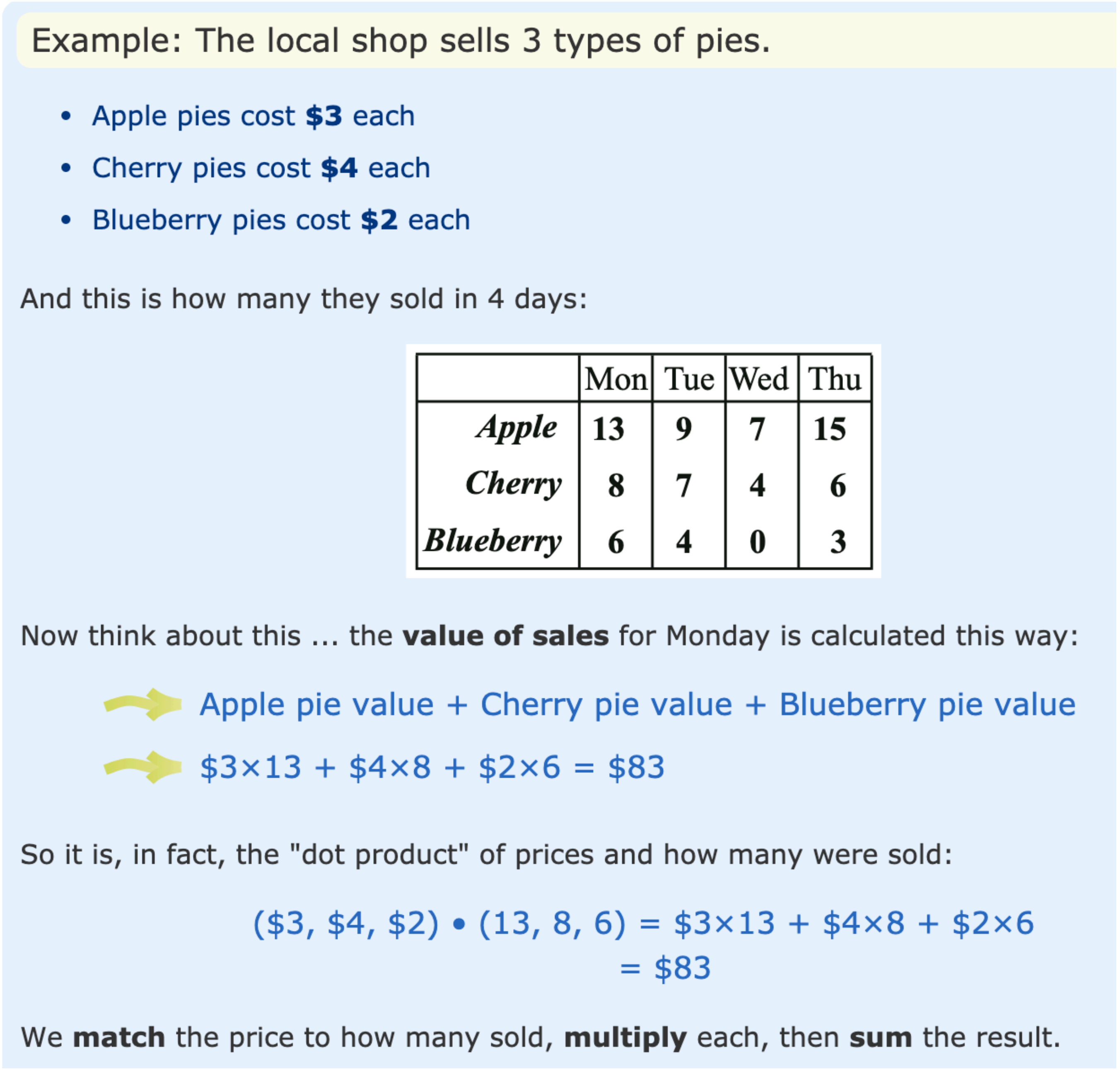
Matrix multiplication

Activity

- Go to link and implement the pie sales example:
  - https://olzama.github.io/Ling471/assignments/activity-May11.html
  - **Goal:** see the resulting 1x4 matrix as the result of multiplication!



https://www.mathsisfun.com/algebra/matrix-multiplying.html



Example: The local shop sells 3 types of pies.

- Apple pies cost **$3** each
- Cherry pies cost **$4** each
- Blueberry pies cost **$2** each

And this is how many they sold in 4 days:

	Mon	Tue	Wed	Thu
Apple	13	9	7	15
Cherry	8	7	4	6
Blueberry	6	4	0	3

Now think about this ... the **value of sales** for Monday is calculated this way:

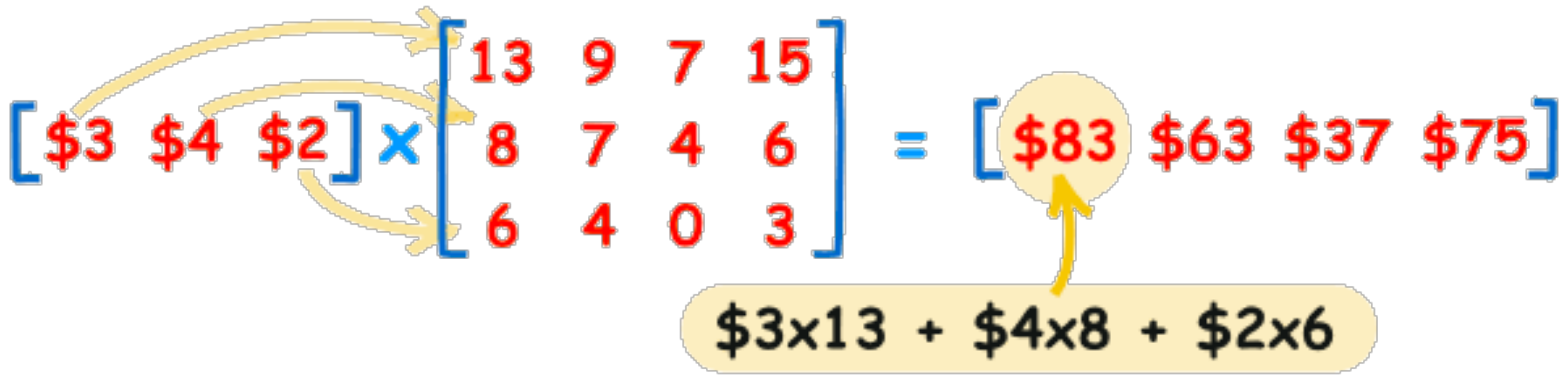
Apple pie value + Cherry pie value + Blueberry pie value

$3×13 + $4×8 + $2×6 = $83

So it is, in fact, the "dot product" of prices and how many were sold:

($3, $4, $2) • (13, 8, 6) = $3×13 + $4×8 + $2×6
= $83

We **match** the price to how many sold, **multiply** each, then **sum** the result.



https://www.mathsisfun.com/algebra/matrix-multiplying.html

Matrix multiplication

why does it matter?

1)

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix} \times \begin{bmatrix} 7 & 8 \\ 9 & 10 \\ 11 & 12 \end{bmatrix} = \begin{bmatrix} 58 & 64 \\ 139 & 154 \end{bmatrix} \checkmark$$

https://www.mathsisfun.com/algebra/matrix-multiplying.html

- It's because **much of ML** can be done by matrix multiplication
- All three pictures have something in common and the latter 2 illustrate **linear regression** which is a **basic ML algorithm**
  - ...but it is **not** easy to understand what they have in common
    - You will need **a course on linear algebra** for this
    - **Our goal:** Get used to the idea that **data** in linear regression **look** like **vectors/matrices**
    - The **"multiplication"** part is hard to understand, which is OK
    - ...suffice it to say that **if you have matrices**, you **can multiply** them
    - ...and that **solving a matrix equation** helps you **minimize errors** of your algorithm
      - ...which is to say it helps you **train** it

2)

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} = \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \vdots \\ 1 & x_n \end{bmatrix} \begin{bmatrix} \beta_0 \\ \beta_1 \end{bmatrix} + \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_n \end{bmatrix}$$

$$Y = X\beta + \varepsilon$$

3)

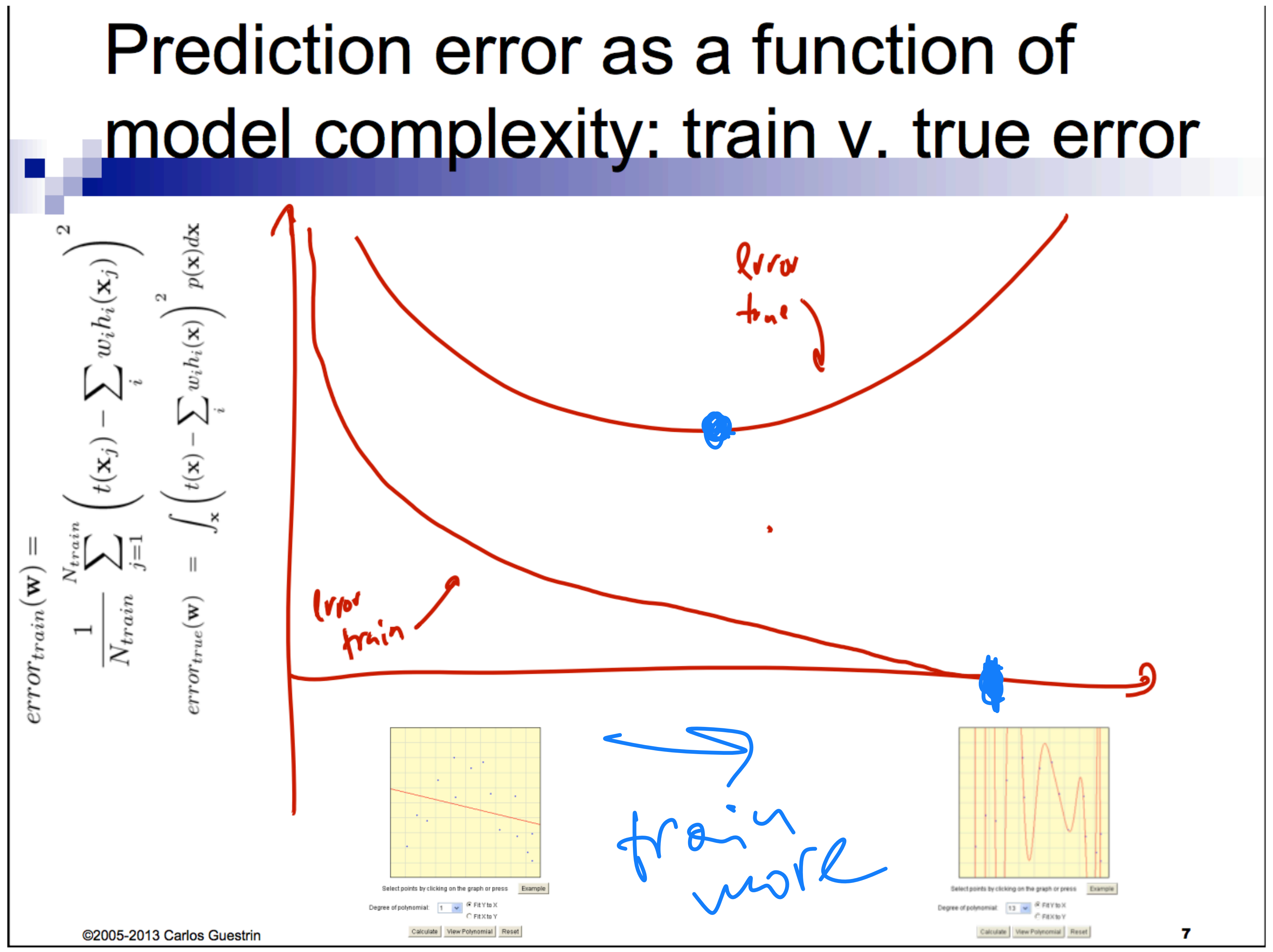
$$h_\theta(x) = \begin{bmatrix} \theta_0 & \theta_1 & \dots & \theta_n \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_n \end{bmatrix} = \theta^T x$$

https://www.programmersought.com/article/1216251344/

Machine learning and linear regression

Machine learning
bird's eye view

- Machine learning:
  - Want a **function** such that it predicts **y** given **x**
    - e.g. POS or NEG given a review
  - "Train" such a function **automatically** on **observations**
    - "training data"
    - What kind of function/**curve fits** the observations **best**?
      - **Option 1**: a curve which **minimizes training error**
      - ...actually, such a curve will go through every point!
      - **that's what we need to consider for now**
      - **Option 2:** a curve which **allows** for **some small** error in training
        - ...but results in **smaller test error** in practice
        - will talk about it next time



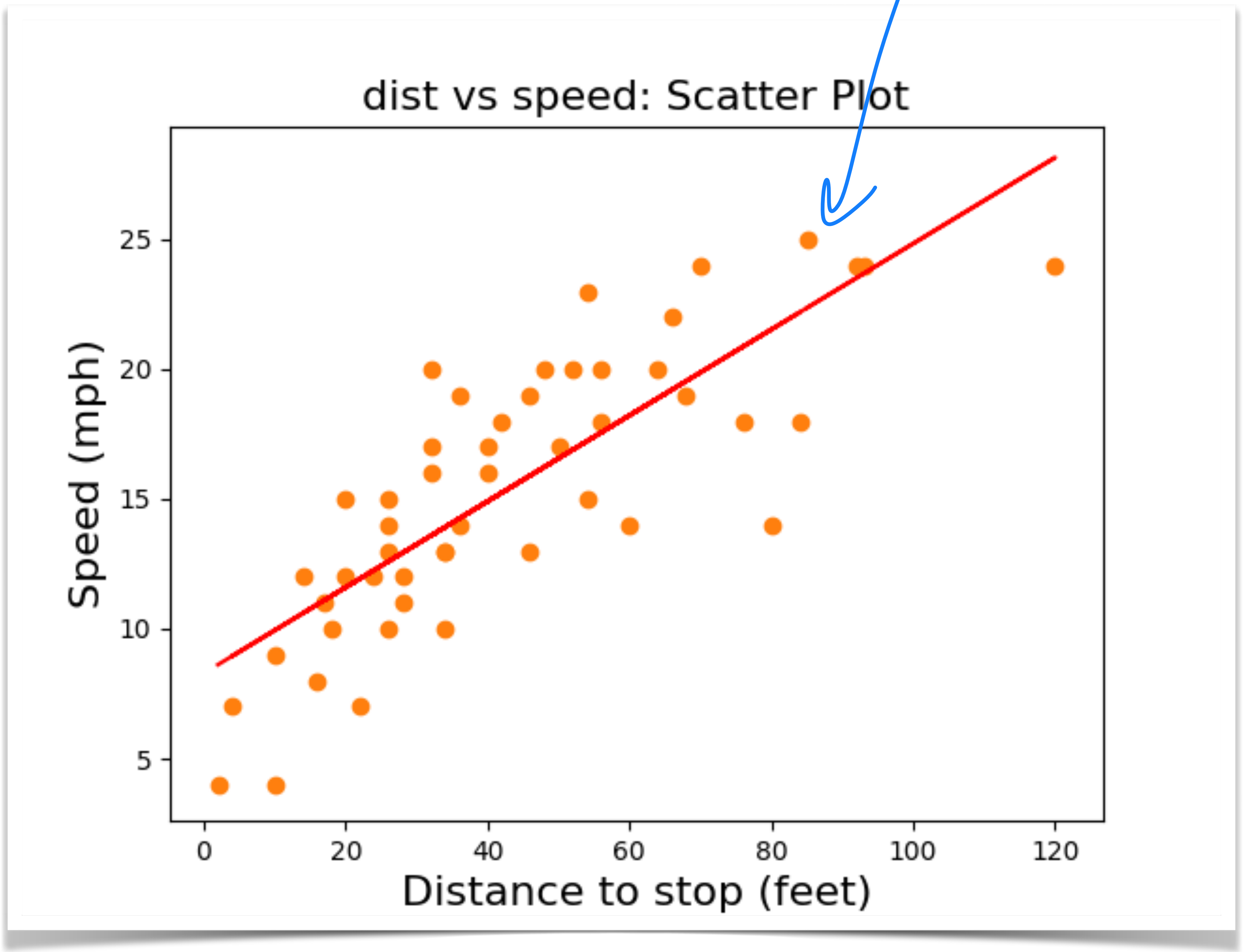
A picture from Carlos Guestrin's lecture on ML

$y = f(x)$

Machine learning
linear regression

- Problem: y = f(x)
  - Predict **y** given **x** where x is **continuous**
    - e.g. predict weather, cost, distance, salary…
  - **Why** do we bother with continuous varialbes in this class?
    - Sociolinguistic variables
    - …to understand the basics of ML
- The mechanics are **abstract** and **hard** to understand
- **Goal**: Next time you see them, you are at least somewhat familiar
  - **No** homework on **implementing** the mechanics
  - But need to understand some concepts to be able to use packages

training data

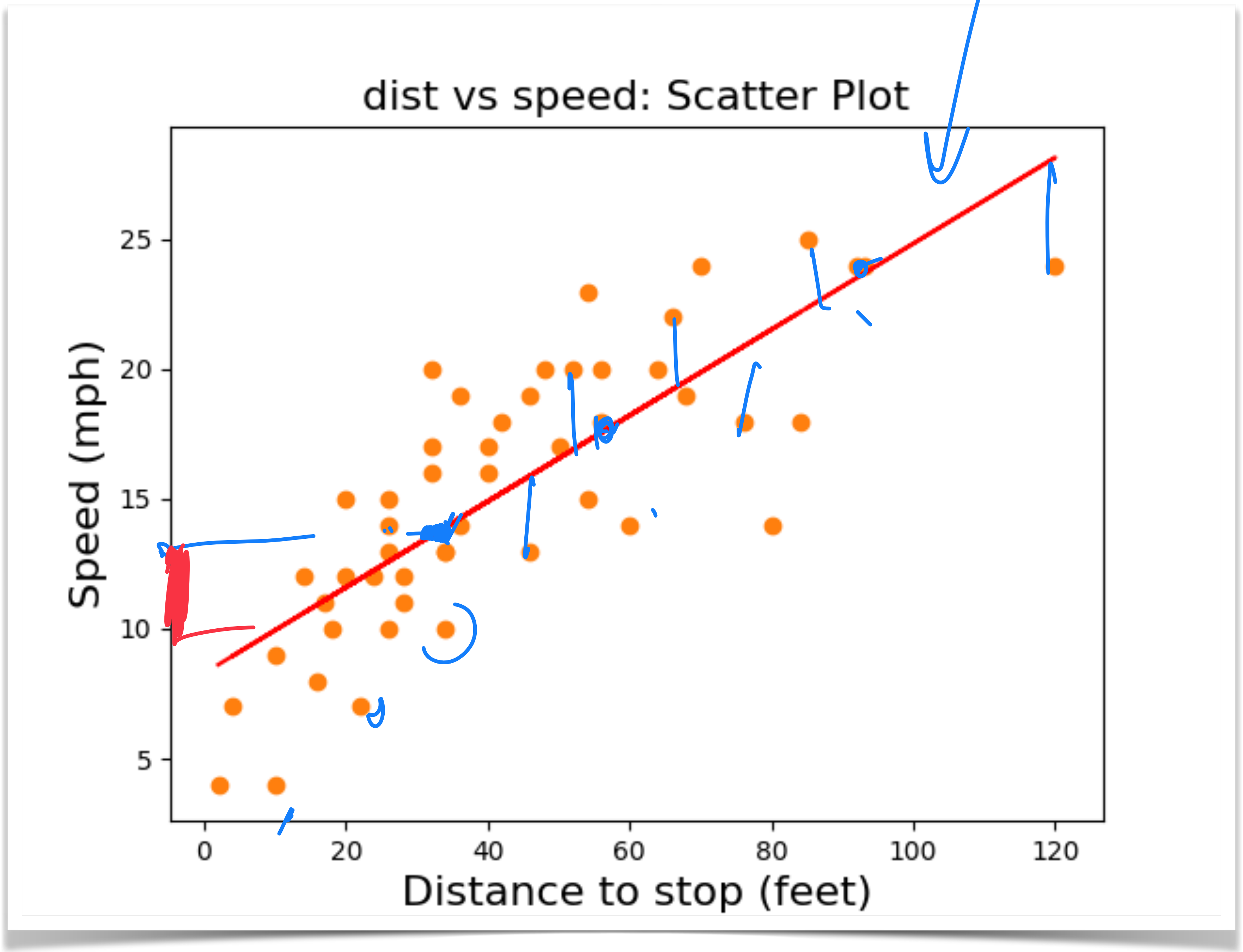


Linear regression
in a nutshell

straight

- **Goal:**
  - Fit a line to observed data
  - ...but **no** line is going to hit **ALL** points!
    - (in any interesting problem)
  - So, find a line such that the error is **minimal**
  - In other words:
    - Given set of **observations**, find **parameters** of the line equation which **minimize the error**
    - (Remember MLE?)
      - We were finding a **parameter** which **maximized likelihood** of observed data
      - This is similar to finding a parameter (or parameters) which **minimize the error** wrt observed data

$y = mx + b$

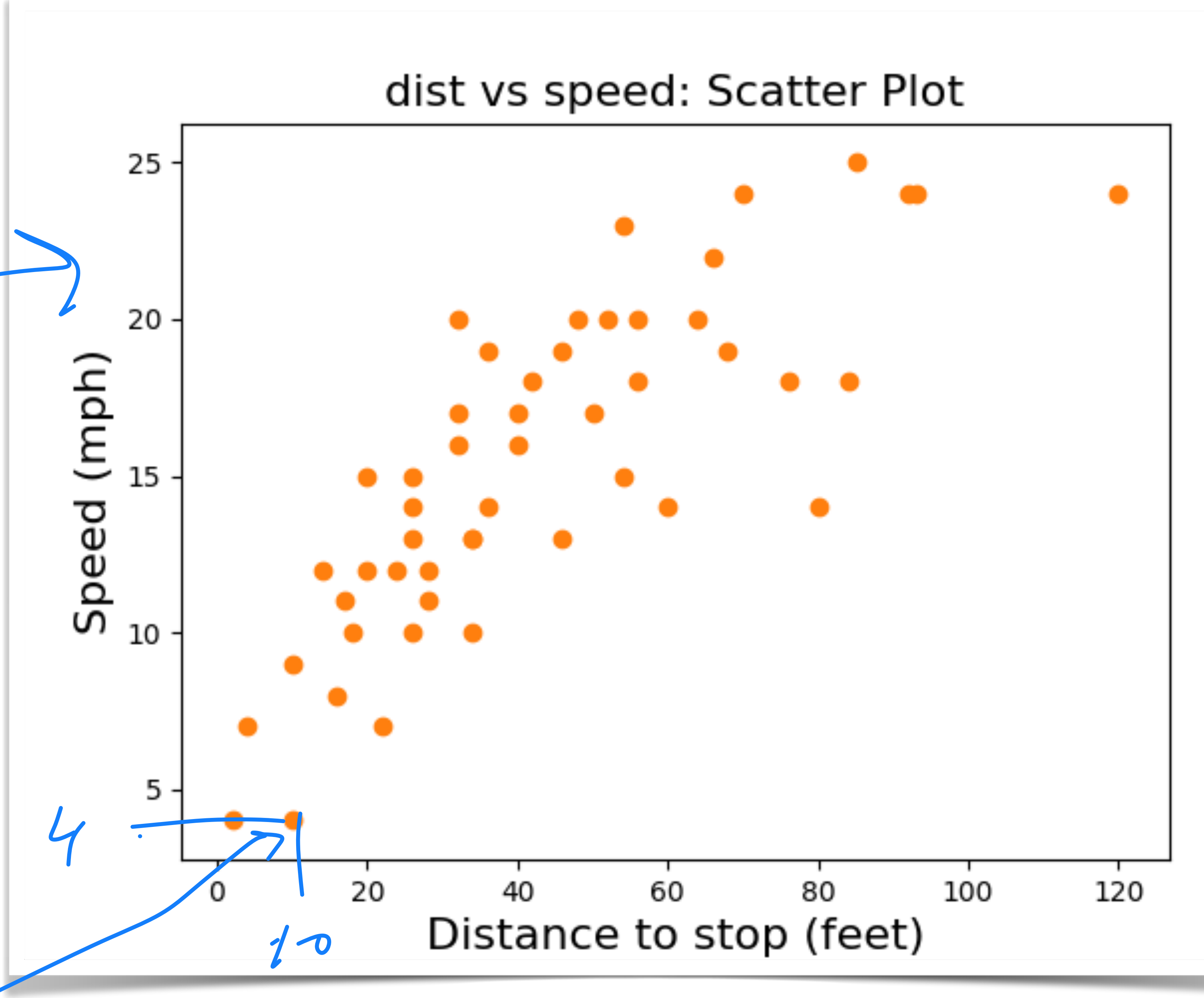


Linear regression

an example

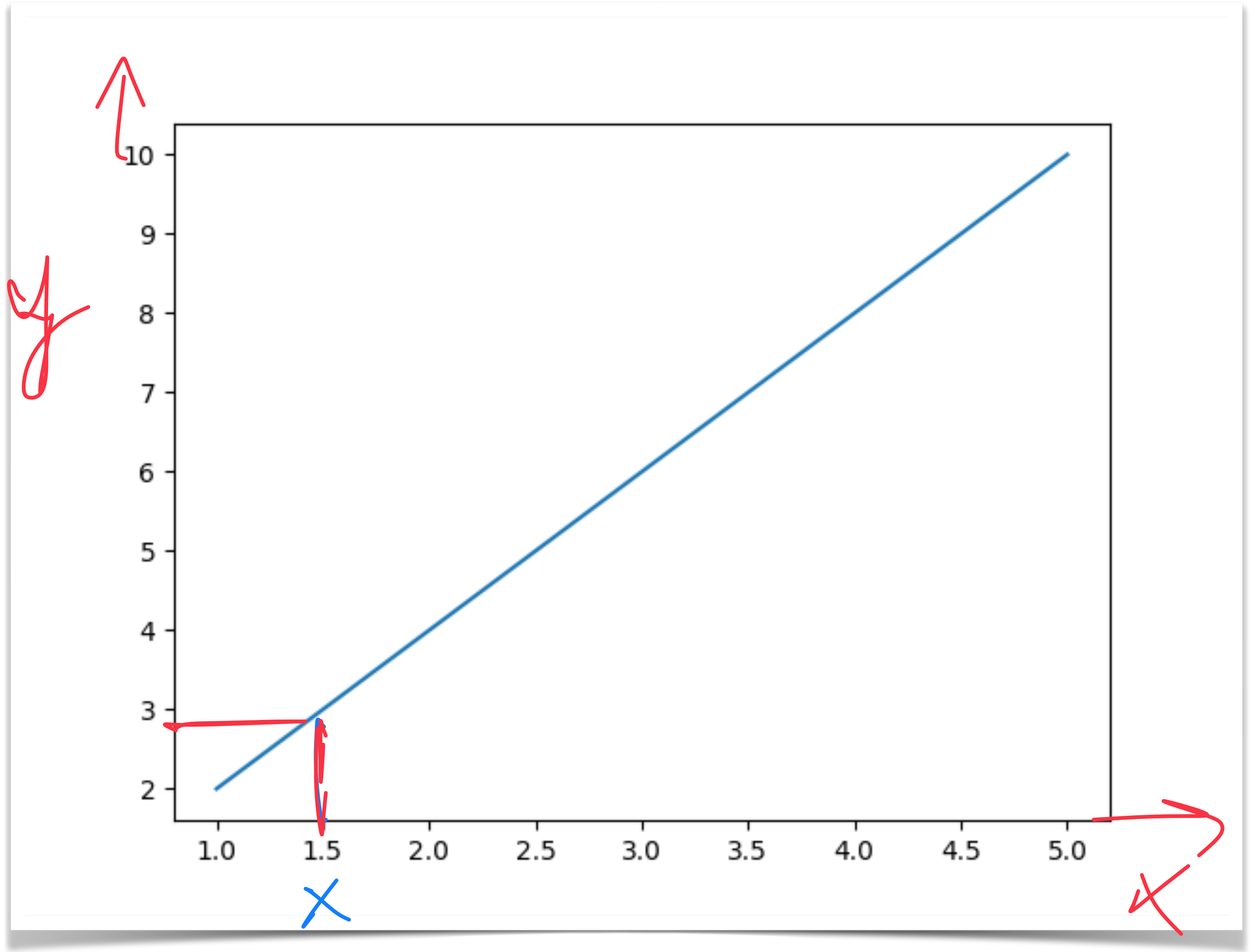
- **Suppose Data:**
  - moving **speed** in mph given **distance** taken to stop
  - (assume 1900 rows here!)
  - **Observations**: (4,2), (4,10), (7,4), (7,22)...
    - (x1,y1), (x2,y2), (x3,y3)...

speed (mph)	distance to stop
4	2
4	10
7	4
7	22
8	16



Linear regression
review of what a line is:

- The **line** equation:
  - **y** = m**x** + b
    - m: coefficient
      - the line **slope**
    - b: intercept
      - where the line **crosses** the y-axis
- (x1=1; y1=2), (x2=2; y2=4)



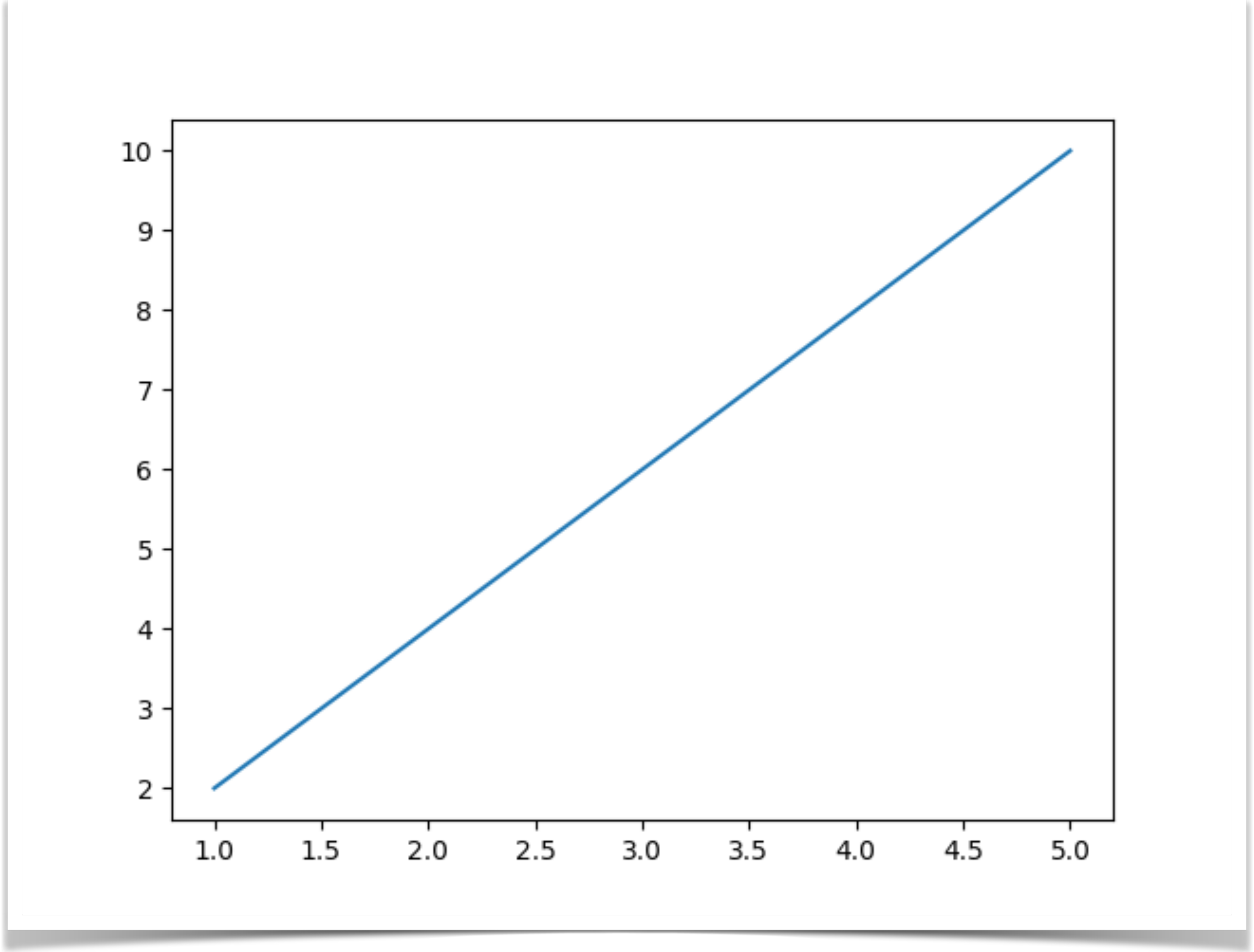
y = 2*x + 0

Linear regression

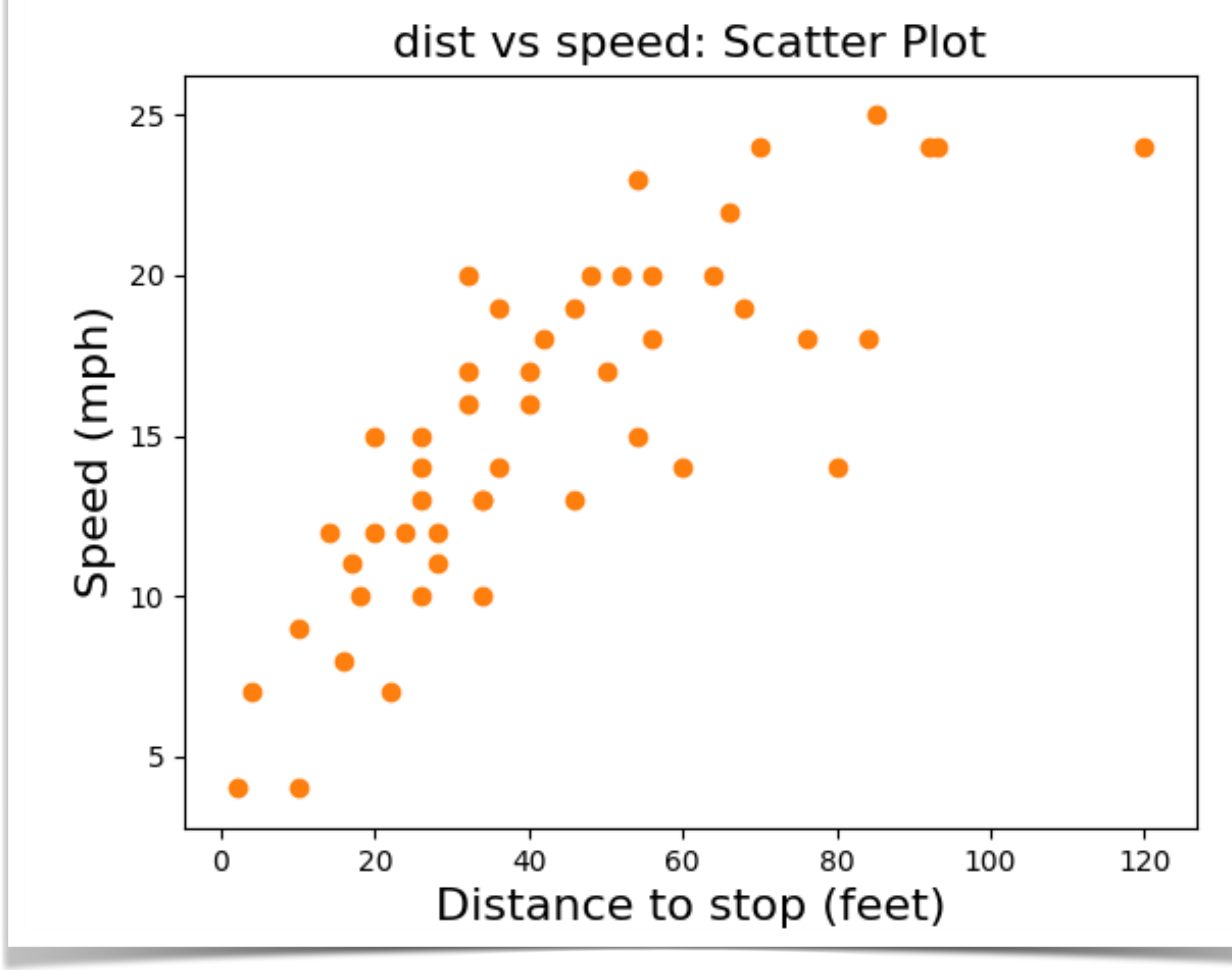
an example

- The **line** equation:
  - **y** = m**x** + b
    - m: coefficient
      - the line **slope**
    - b: intercept
      - where the line **crosses** the y-axis

speed (mph)	distance to stop
4	2
4	10
7	4
7	22
8	16



y = 2*x + 0



Linear regression

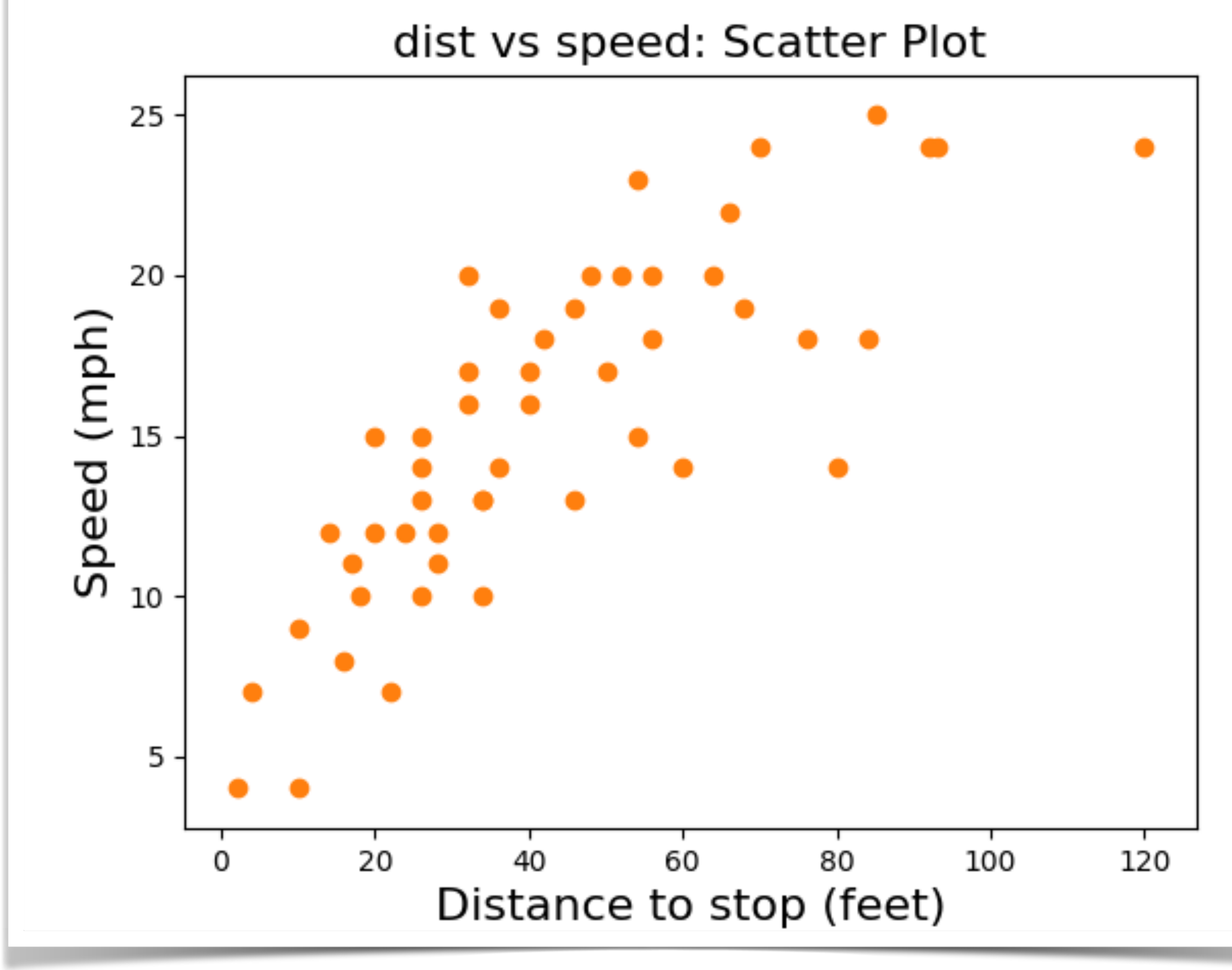
"Errors" and "Least squares"

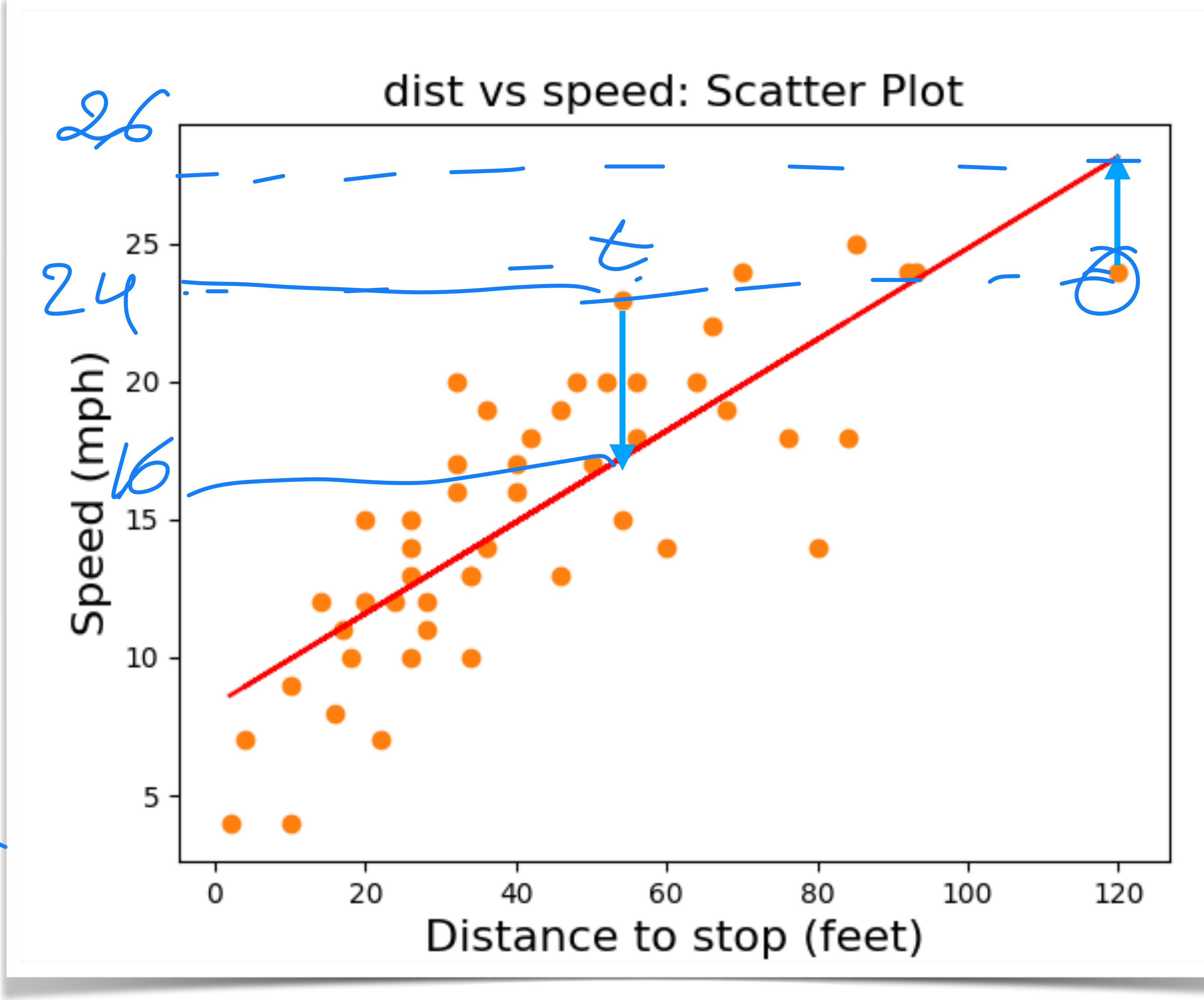
- How many points does our line predict correctly?
- What about the ones it doesn't?
    - Drop a vertical line from each
    - Its length is the "error"
        - how far is the prediction from the truth?
        - ...now, when choosing **m** and **b**, minimize the **sum of errors**
            - ...furthermore, minimize the sum of **squared** errors
            - ...because you care about the absolute distance from the truth, not whether it is above or below the actual point

speed (mph)	distance to stop
4	2
4	10
7	4
7	22
8	16

$24 - 16 = 8$

$24 - 26 = -2$





Linear regression

“Errors” and “Least squares”

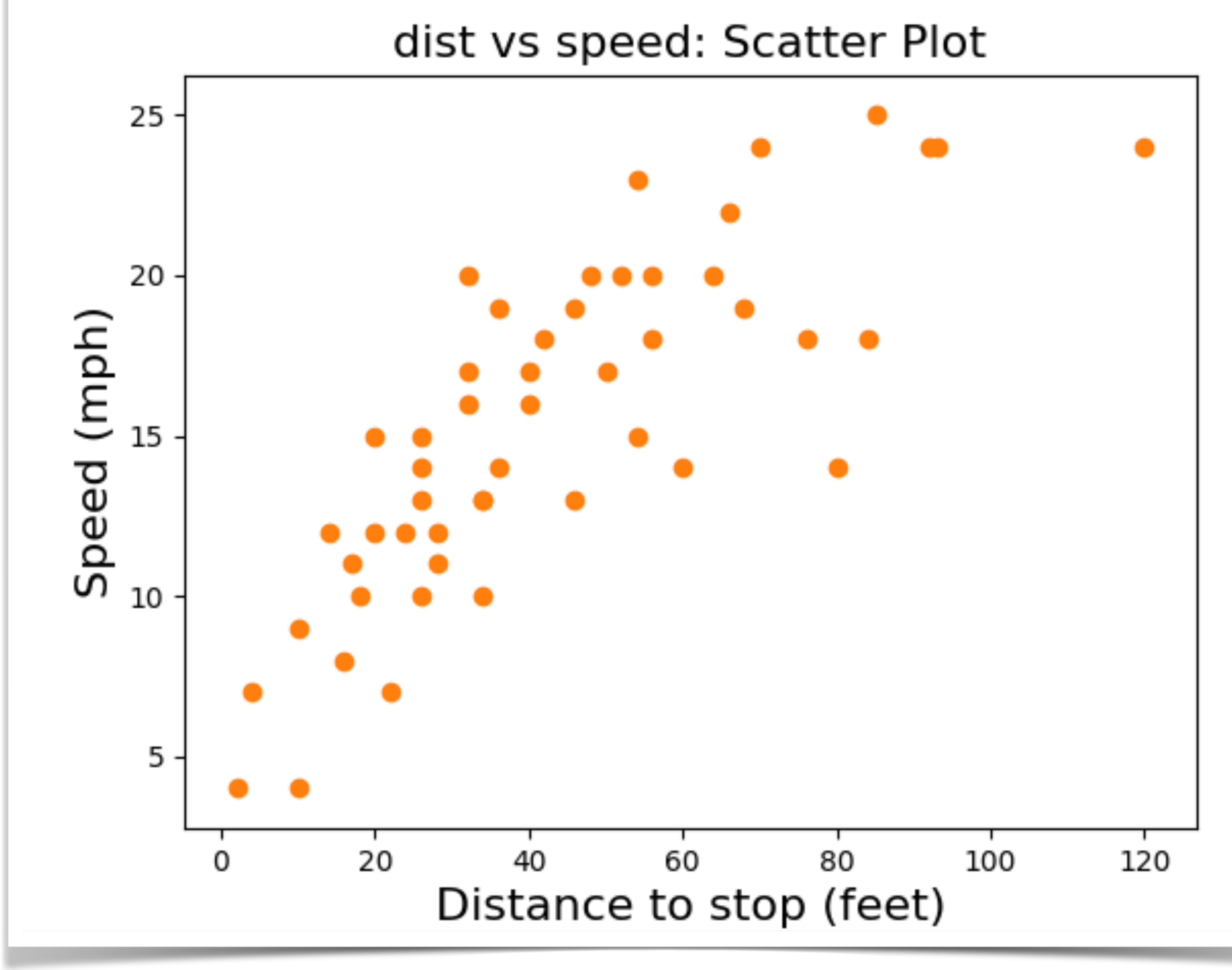
$[e_1, e_2, e_3, \ldots, e_{1900}]$

- Sum of squared errors:
  - $(y_1 - f(x_1))^2 + (y_2 - f(x_2))^2 + (y_1 - f(x_1))^2 + \ldots + (y_{1900} - f(x_{1900}))^2$
    - + e_1 p, e_2, e_{1900}
  - $\boxed{\sum_{i=1}^{n} e_i^2}$ ← $e_1^2 + e_2^2 + e_3^2 \ldots + e_{1900}^2$
  - y1 is the **truth;** f(x1) is the **prediction**
  - Let **e1** = y1 - f(x1)
  - $\boxed{\text{y1 = } \mathbf{m}\text{x1 + } \mathbf{b} \text{ + e1}}$ ←
    - e is the error (can be positive or negative number)

true_val = $f(x_i) + e_i$

y_i

speed (mph)	distance to stop
4	2
4	10
7	4
7	22
8	16



Linear regression

"Least squares"

- y1 = **m**x1 + **b** + e1
  - One point is not worth much; can't find m or b
    - Need two points to draw a line…
  - For two points, error will always be 0
  - So, this is inherently a problem for **system** of **multiple** equations
    - which may be written and solved as a **matrix multiplication** problem

$y_1 = m x_1 + b + e_1$

$y_2 = m x_2 + b + e_2$

$\vdots$

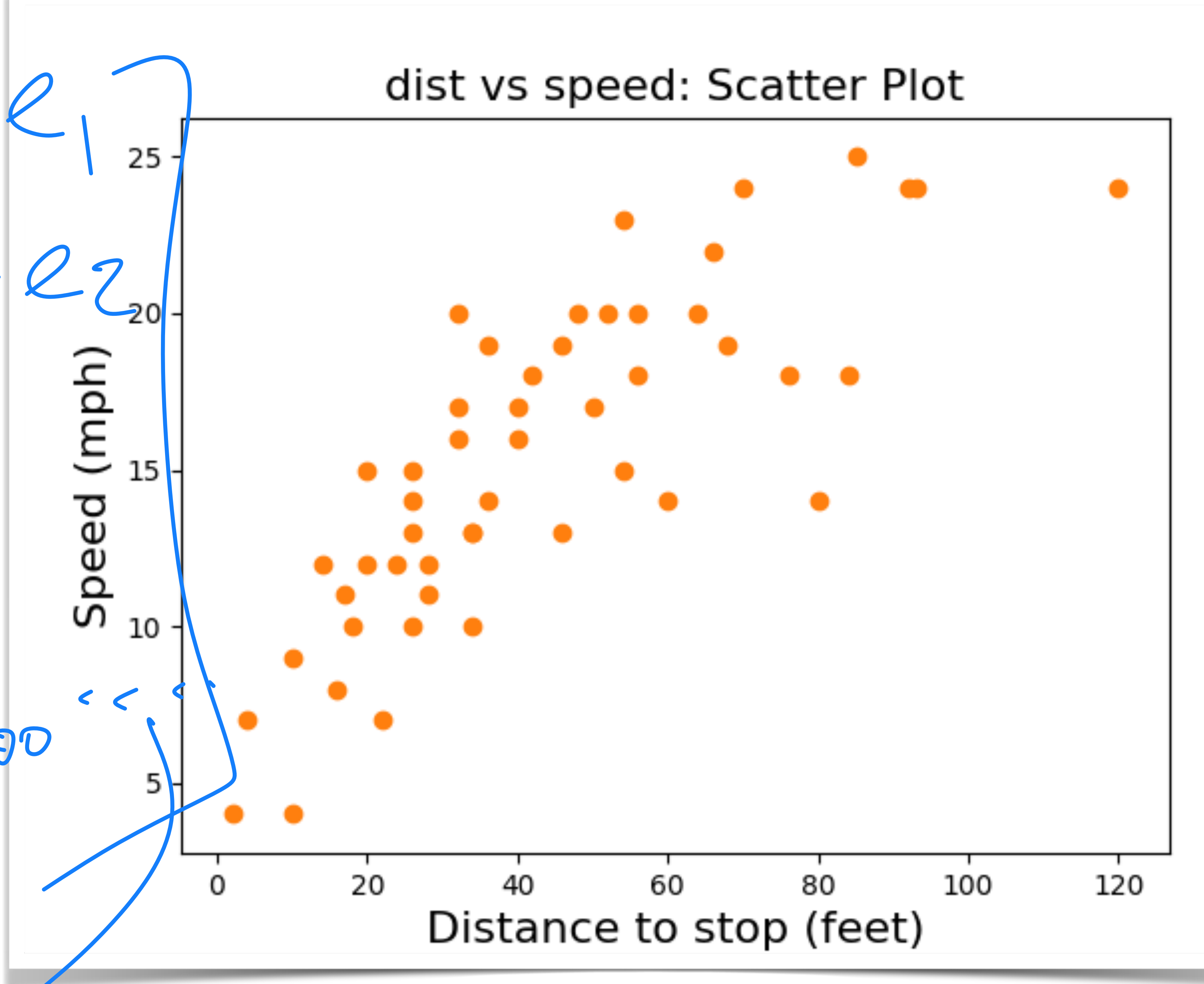
$y_{1900} = m x_{1900}$

speed (mph)	distance to stop
4	2
4	10
7	4
7	22
8	16

2x1

$$X = \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \vdots \\ 1 & x_n \end{bmatrix} \quad Y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} \quad A = \begin{bmatrix} b \\ m \end{bmatrix} \quad E = \begin{bmatrix} e_1 \\ e_2 \\ \vdots \\ e_n \end{bmatrix}$$

params

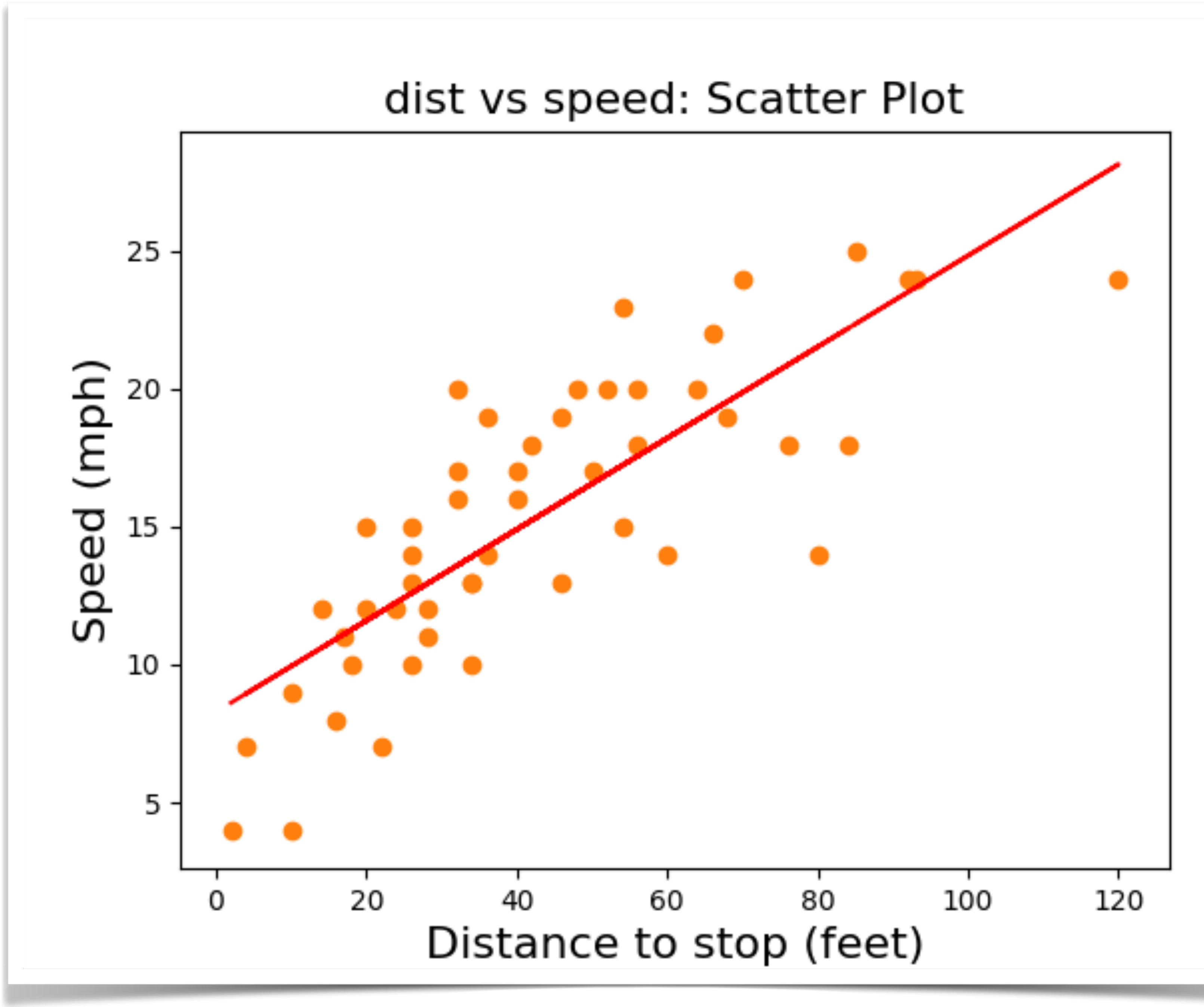


Linear regression

“Least squares”

$$X = \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \vdots \\ 1 & x_n \end{bmatrix} \quad Y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} \quad A = \begin{bmatrix} b \\ m \end{bmatrix} \quad E = \begin{bmatrix} e_1 \\ e_2 \\ \vdots \\ e_n \end{bmatrix}$$

- **Y = XA + E**
  - All things here are **matrices**
  - Y, A, E are just **vectors** (matrices of **width 1**)
    - **vectors are matrices**, too!
    - X needs to have the same **width** as the **length** of A
      - ...to conform to matrix multiplication definition
- Want: **solve for A to minimize** $\sum_{i=1}^{n} e_i^2$



Linear regression

“Least squares”

$X^T \cdot X$

e_i^2

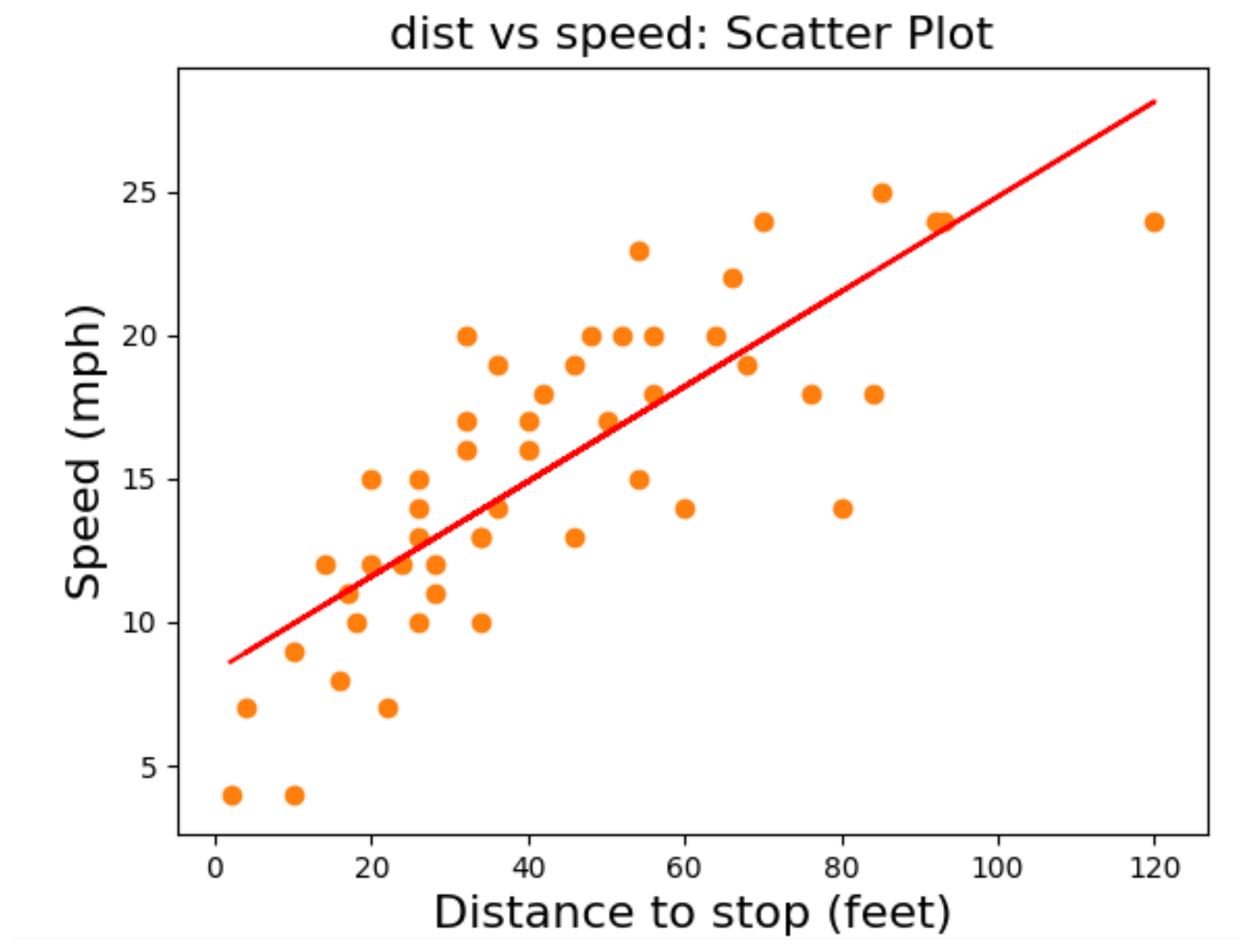
$X^T \cdot X$

$e_i \cdot e_i$

$$X = \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \vdots \\ 1 & x_n \end{bmatrix} \quad Y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} \quad A = \begin{bmatrix} b \\ m \end{bmatrix} \quad E = \begin{bmatrix} e_1 \\ e_2 \\ \vdots \\ e_n \end{bmatrix}$$

- **Y = XA + E** <— want **E** to be **minimal.** Now, you know **Y** and you know **X**! You **don’t** know **A**.
- **If** you’d t**aken a course (or two) in linear algebra**, you would **know** that, to minimize $\sum_{i=1}^{n} e_i^2$:
  - $A = (X^TX)^{-1}X^TY$ <— magic. Just remember that it involves matrix multiplication…
  - I can’t explain **why** in this class :(
    - It has to do with the fact that the sum of squares is related to both dot product (sum) and multiplying a thing by itself (squaring, product)
    - It has to do with derivatives and solving a matrix equation for the derivative set = 0
  - Doesn’t matter! Point is, **need to multiply matrices!**
  - What **matters** for you:
    - Understand that **data are matrices**
    - which **have dimensions** such that multiplication is possible
    - **Also matters:** some matrices will be **transposed** in order to have the right dimensions and to get multiplied

$y = 2x \rightarrow \begin{bmatrix} 0 \\ 2 \end{bmatrix}$



| 3 | 4 | 2 |

Pandas linear regression demo

Matrix multiplication
in machine learning

- The x-s are **observations**
- The y-s/h-s are ~~**predictions**~~ true values
  - **h** is actually a **vector**
  - (if you are lucky, it will be marked as such…)
- The betas/thetas are **coefficients**
  - weights
  - **parameters**
  - **Goal:** solve for parameters such that sum of squared errors is minimized
- Overfitting?!
  - Yes! But let's talk about it next time (briefly!)

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix} \times \begin{bmatrix} 7 & 8 \\ 9 & 10 \\ 11 & 12 \end{bmatrix} = \begin{bmatrix} 58 & 64 \\ 139 & 154 \end{bmatrix} \checkmark$$

https://www.mathsisfun.com/algebra/matrix-multiplying.html

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} = \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \vdots \\ 1 & x_n \end{bmatrix} \begin{bmatrix} \beta_0 \\ \beta_1 \end{bmatrix} + \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_n \end{bmatrix}$$

$$Y = X\beta + \varepsilon$$

$$\vec{h}_\theta(x) = \begin{bmatrix} \theta_0 & \theta_1 & \dots & \theta_n \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_n \end{bmatrix} = \theta^T \vec{x}$$

https://www.programmersought.com/article/1216251344/

Recap
of today's madness

- We looked at some pretty **dense** stuff
- ...which gets at the **core** of **how ML works**
- It is **not possible** to internalize it right away
- ...**especially** if you have not taken linear algebra
- **Our goal** was:
  - To convince ourselves that it is important to be able to **store data in tables** (matrices)
  - ...and that it is important to understand what matrices' **dimensions** are

Lecture survey:
in the chat