Hewlett Packard Enterprise Data Science Institute
UNIVERSITY OF HOUSTON

311 Introduction to Machine Learning

Summer 2024

Instructor: Ioannis Konstantinidis

Quick review

Model evaluation
- Training Cross-validation (k-fold)
- Testing Metrics and Scoring

Supervised models for classification
- k-Nearest Neighbors
- Logistic regression
- Support Vector Machines
- Decision Trees
    - Random Forests
    - Gradient Boosting

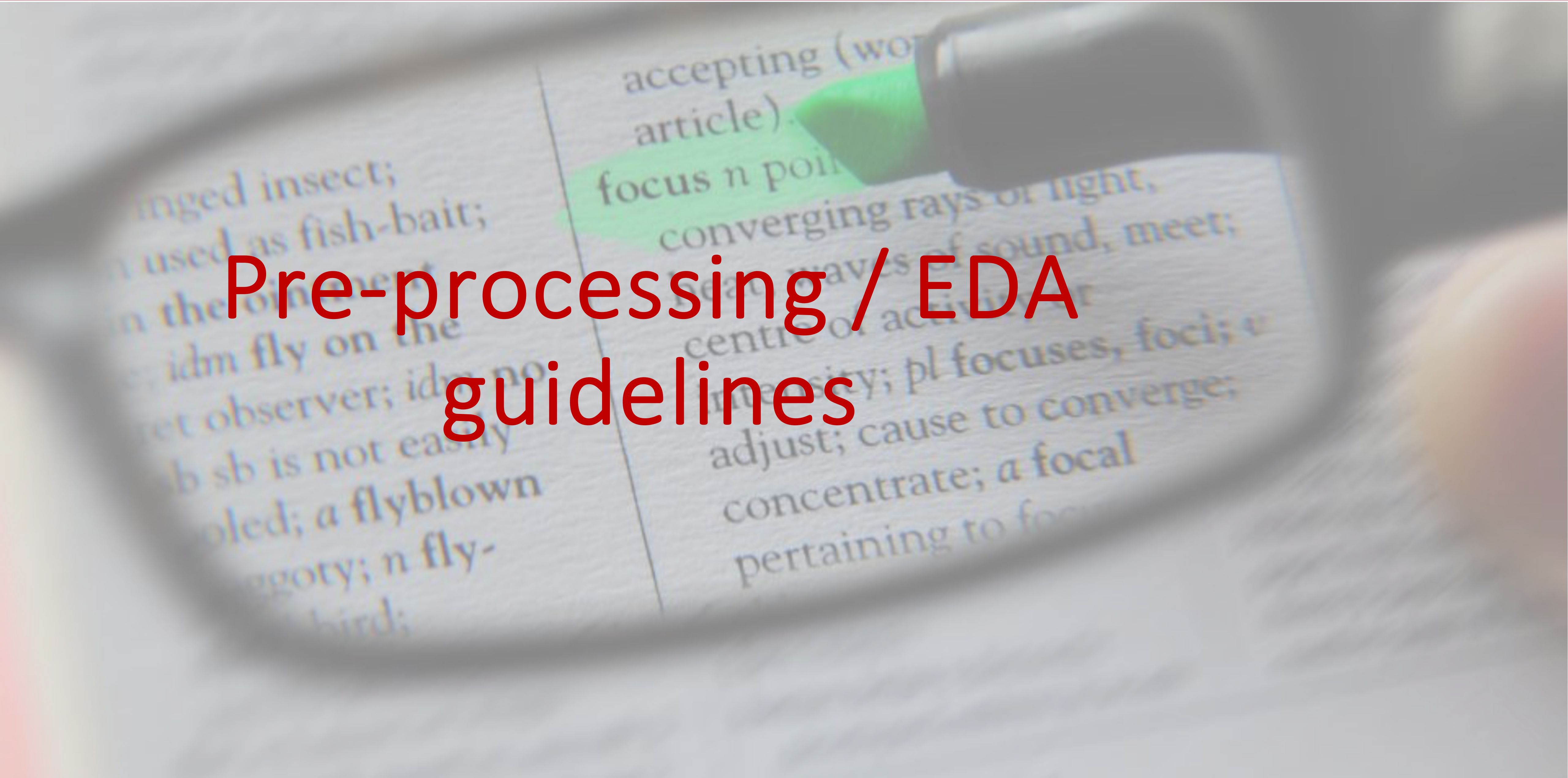
Pit stop: organizational guidelines

Data pre-processing and exploratory data analysis (EDA)

- Data cleaning and tidying up
- Numerical summaries
- Graphical summaries

Data transformations

- Scaling (standard/MinMax)
- Feature Extraction (PCA / dimension reduction)

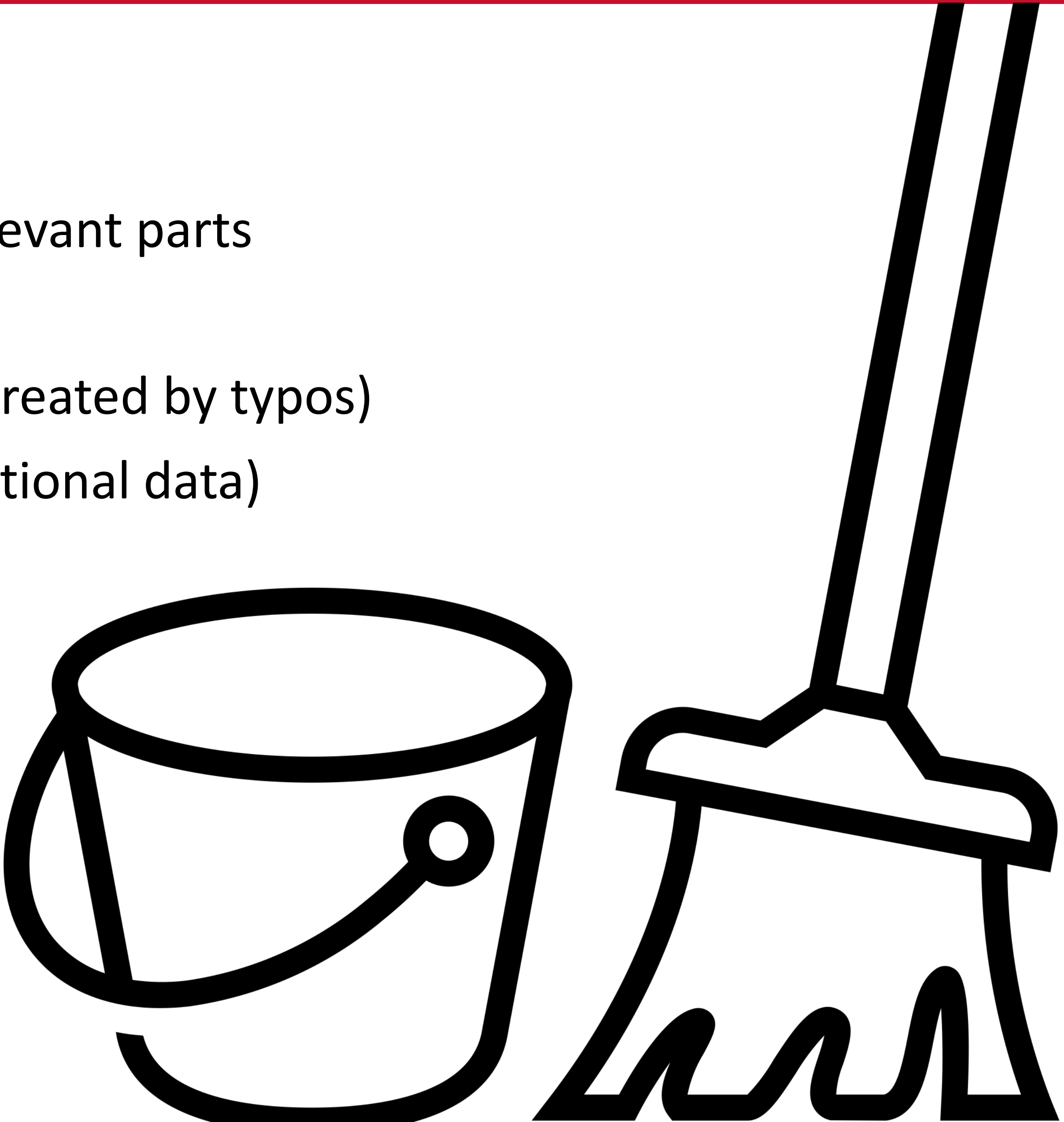


Pre-processing / EDA guidelines

Cleaning data

No incomplete, incorrect, inaccurate, or irrelevant parts

- identifying missing values
- matching similar but not identical values (created by typos)
- correcting character encodings (for international data)
- filling in structural missing values
- parsing dates and numbers
- …



EDA checklist

Sanity checks:

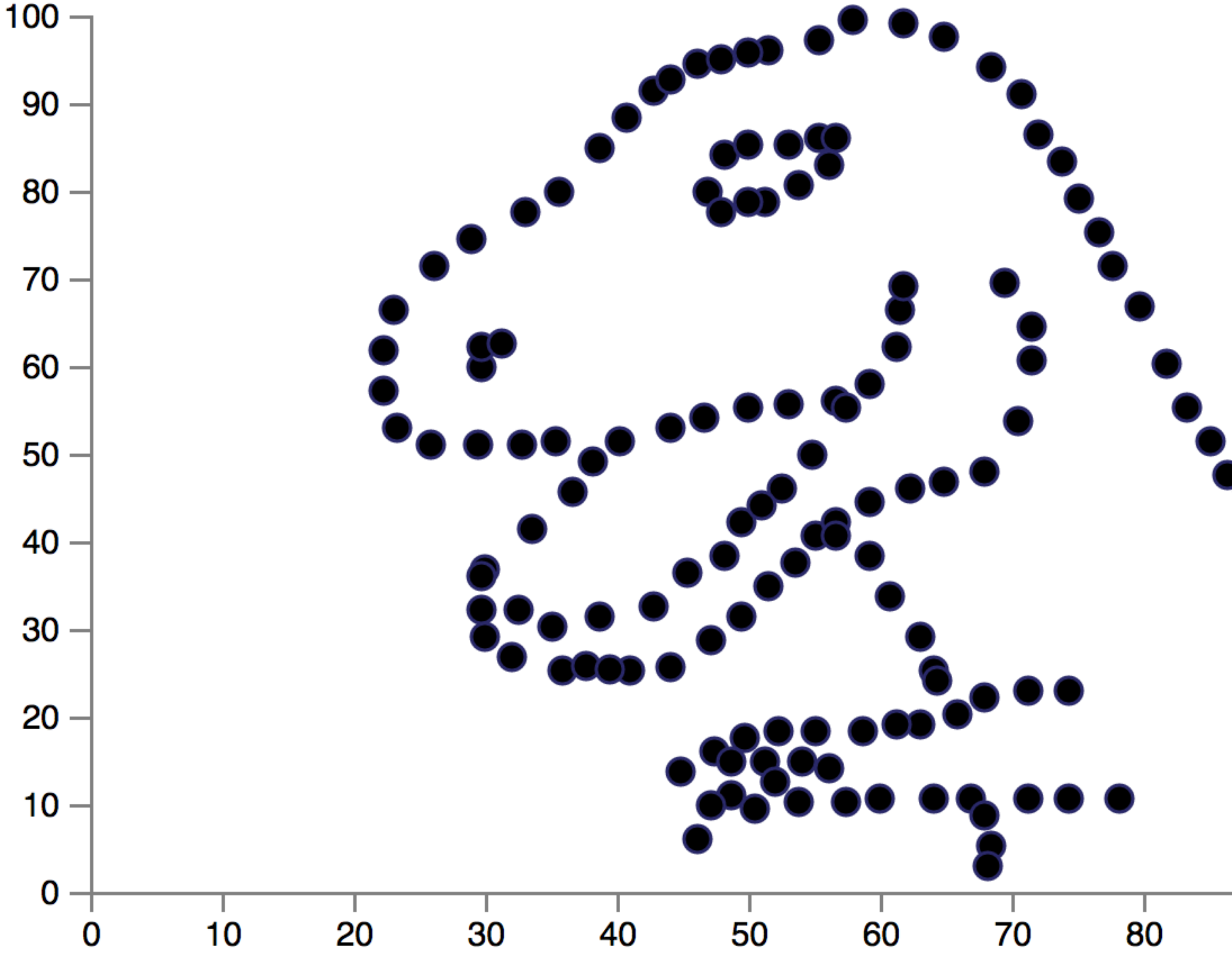
- Look at the top and the bottom of your data tables
- Check your univariate statistics
  - Numerical Summaries
- Check your bivariate plots
  - Graphical Summaries

Reasons for making plots

- Setting expectations for what the data should look like
- Checking deviations from what you might expect
- Numerical summaries don’t give the whole picture

Datasaurus

https://blog.revolutionanalytics.com/2017/05/the-datasaurus-dozen.html

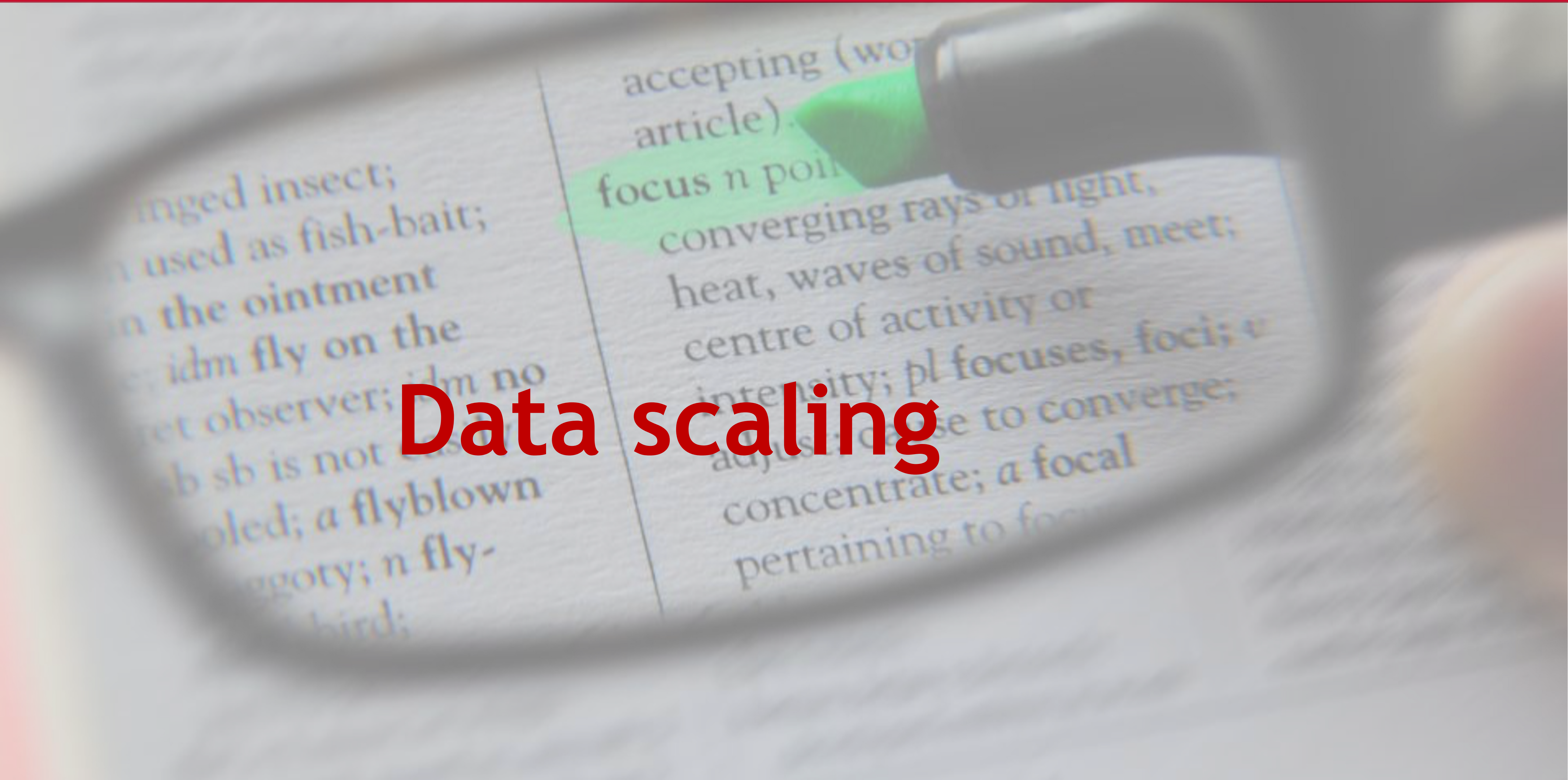


Data transformations are data processing tasks

- They come after pre-processing the data
- They come after EDA

EDA and data pre-processing tasks are done to make the learning process possible

Feature organization transformations are done to improve the learning process



Data scaling

Why mess with the data values?

Some algorithms rely on distance/similarity

- kNN and SVM rely on computing distance / similarity to the nearest neighbors or the support vectors, respectively.

- Tree-based algorithms on the other hand (e.g., decision trees, random forests) do not rely on finding distance / similarity to any specific points (they use comparisons to fixed thresholds instead).

Distance measurements are affected by scaling

Temp and humidity:

- F value range is about 0 - 100
- % value range is about 0 - 100

A change of one unit in temperature value

counts the same as

a change of one unit in humidity values

Distance measurements are affected by scaling

Temp and humidity:

- F value range is about 0 - 100
- % value range is about 0 - 100

A change of one unit in temperature value

counts the same as

a change of one unit in humidity values

But this is an accident, due to the arbitrary choice of units

Distance measurements are affected by scaling

Temp and humidity:

- C value range is about 0 - 40
- decimal value range is about 0 - 1

A change of one unit in temperature value

counts as a LOT less than

a change of one unit in humidity values

Distance measurements are affected by scaling

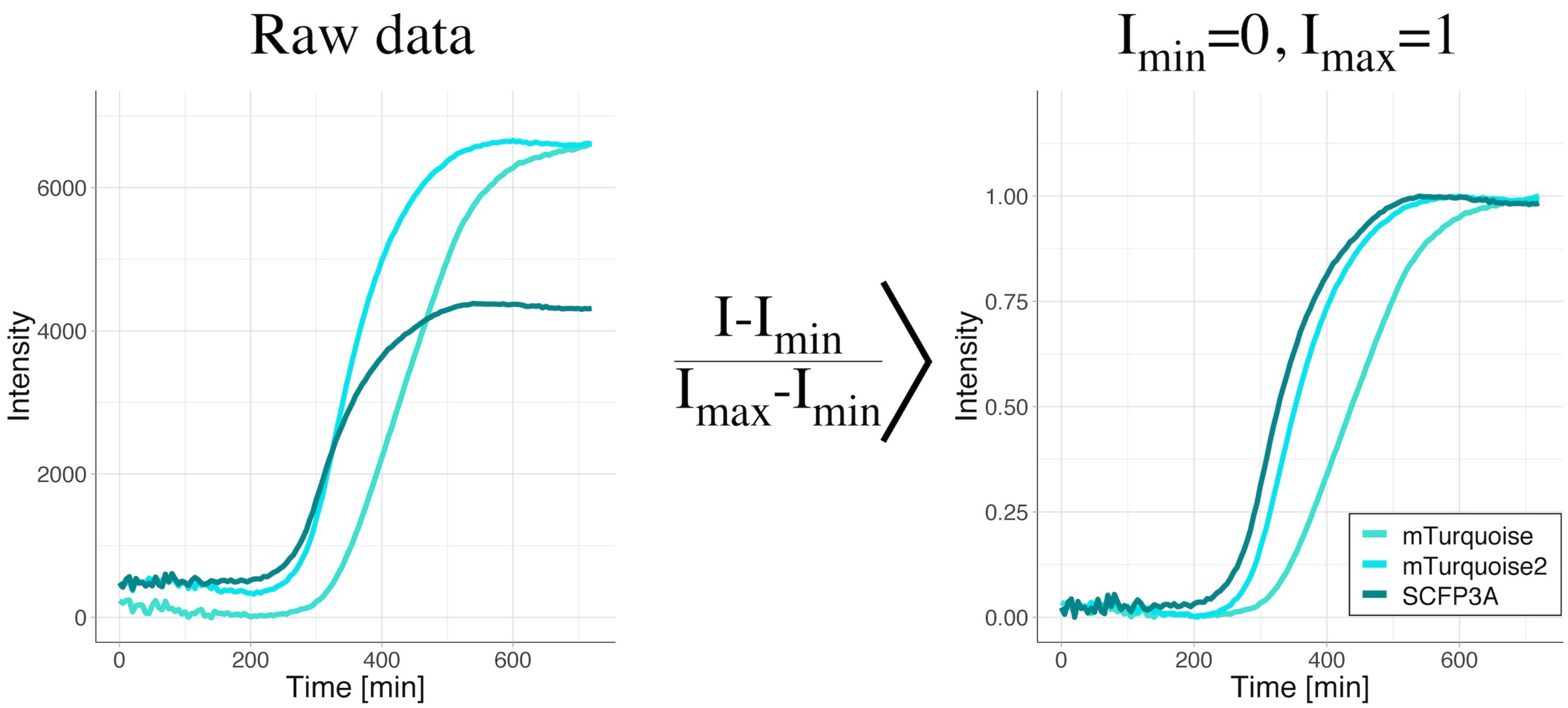
The relative influence of a variable on the total similarity/distance should not depend on an arbitrary choice of units

-> Need to scale the data

How to scale the data?

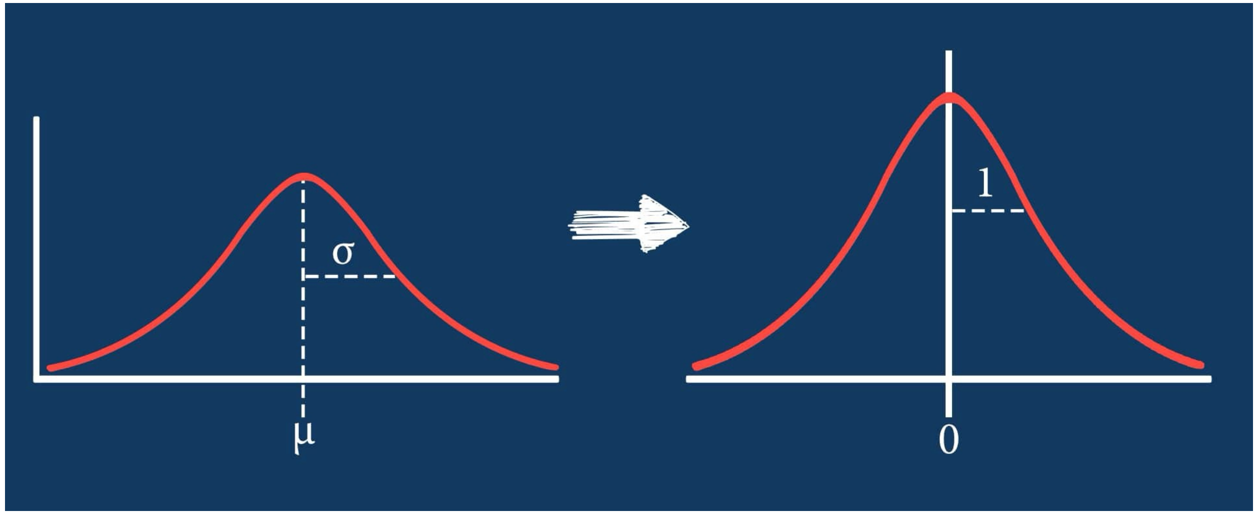
MinMaxScaler: uniform range of 0 - 1

$$x_{\text{norm}} = \frac{x - \min(x)}{\max(x) - \min(x)}$$



StandardScaler: centered at 0

$$x_{\text{stand}} = \frac{x - \text{mean}(x)}{\text{standard deviation }(x)}$$



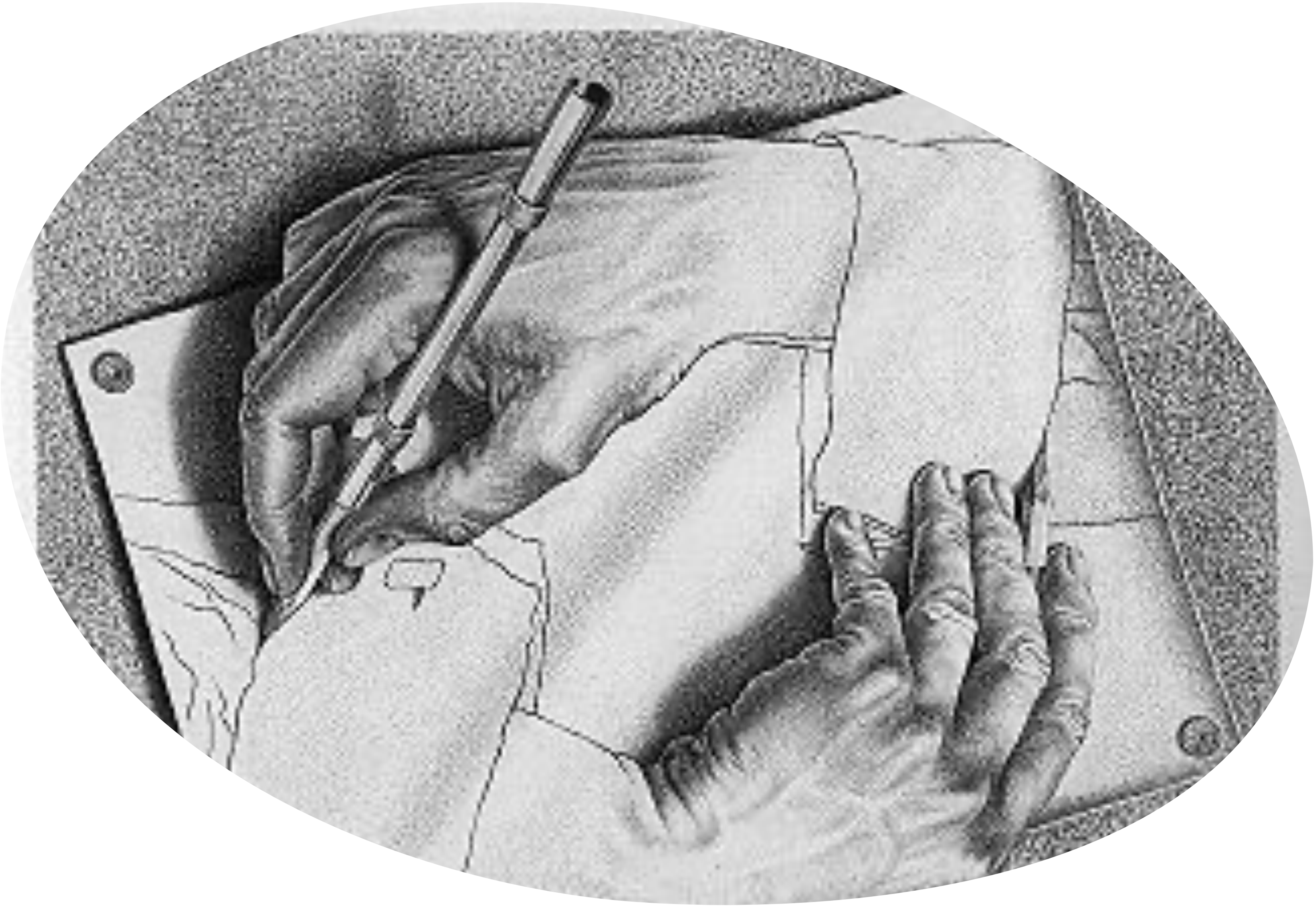
Are the data normally distributed?

Fit once, and then reuse

Fit the scaler using available training data. For normalization, this means the training data will be used to estimate the minimum and maximum observable values. This is done by calling the fit() function.

Apply the scale to training data. This means you can use the normalized data to train your model. This is done by calling the transform() function.

Apply the scale to data going forward. This means you can prepare new data in the future on which you want to make predictions.

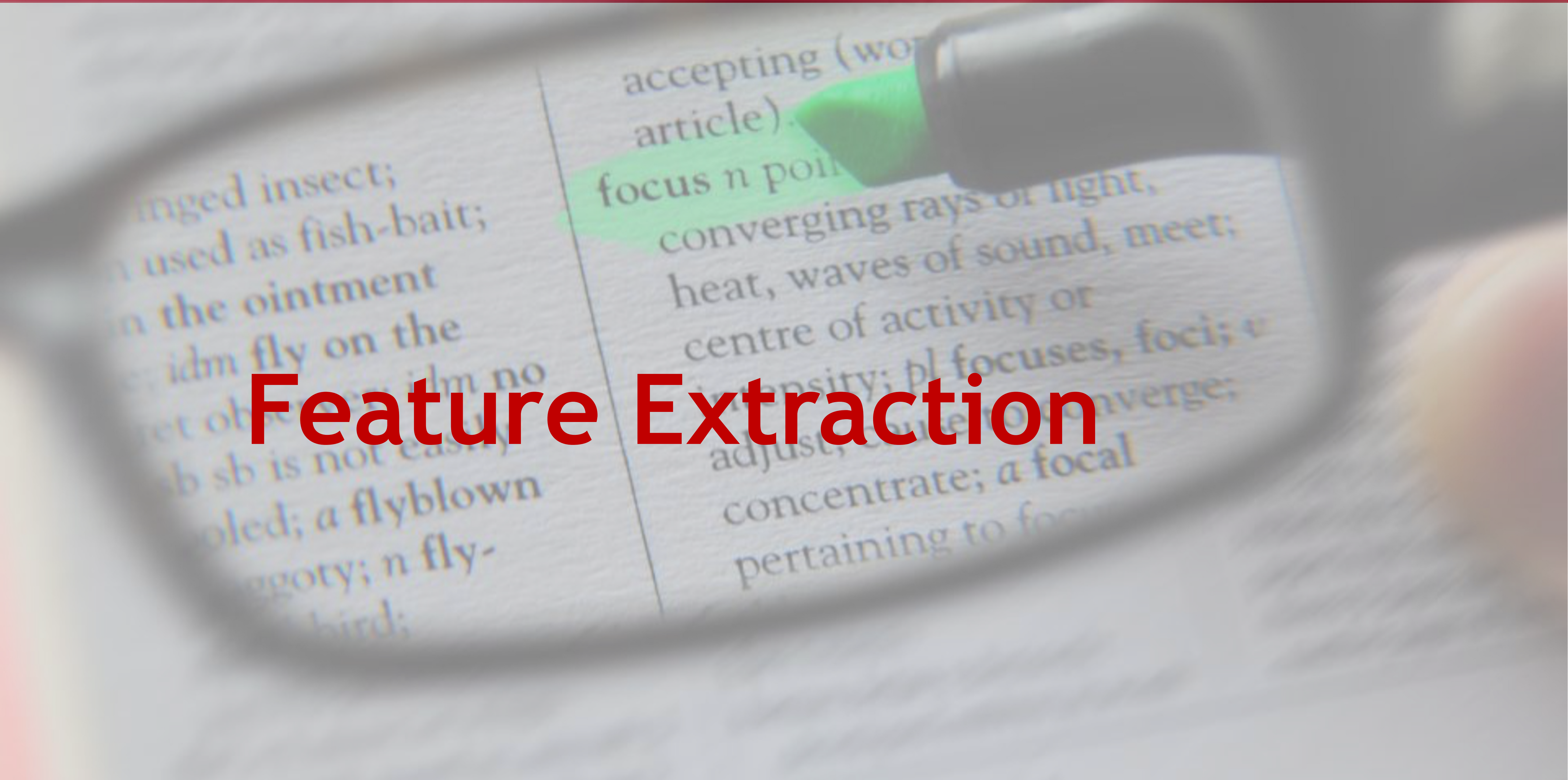


Hands-on Example:

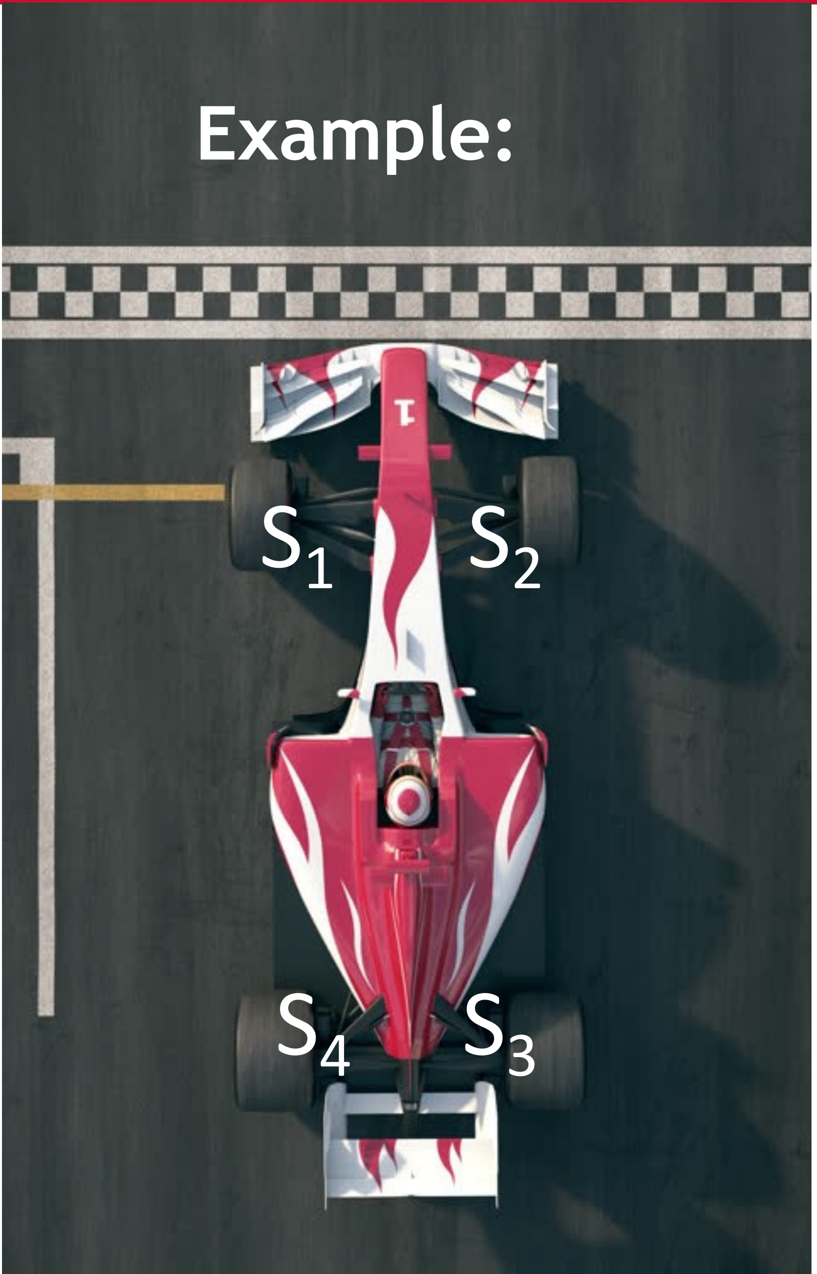
Scaling

Hewlett Packard Enterprise
Data Science Institute
UNIVERSITY OF HOUSTON

Feature Extraction



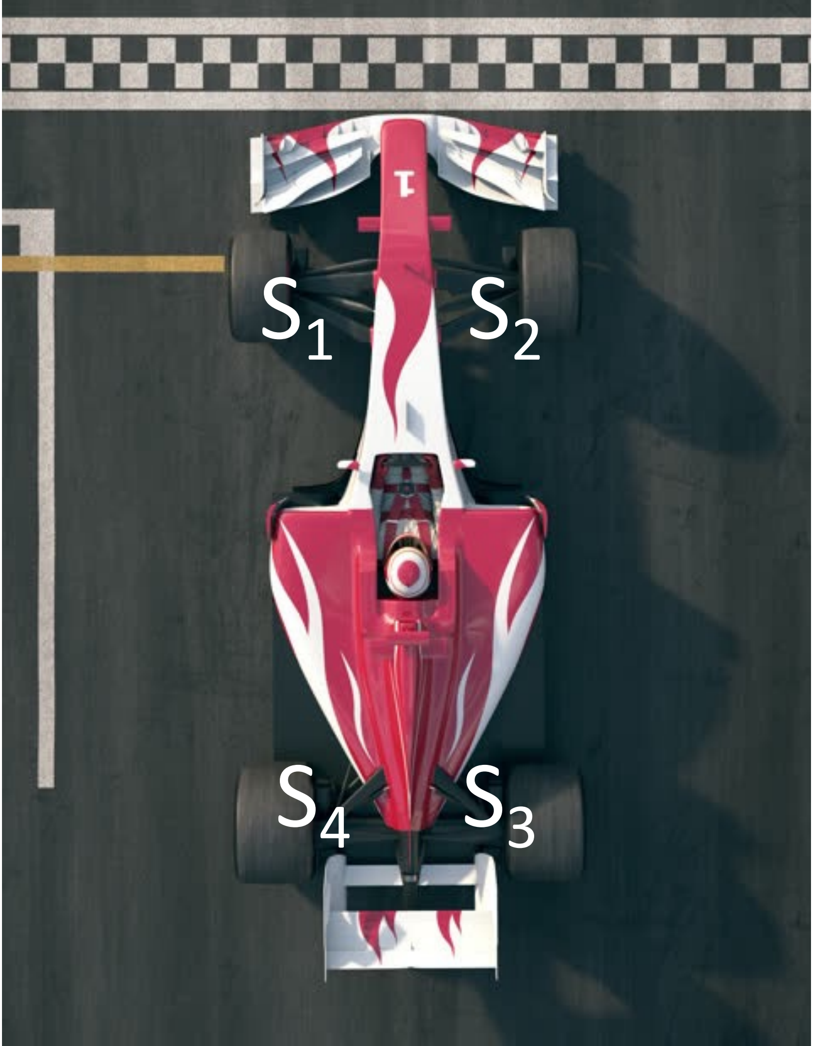
Why mess with the variables / columns?



Original variables

Four sensors measuring rotation speed (spin) at each wheel: S_1, S_2, S_3, S_4

Example:



Features

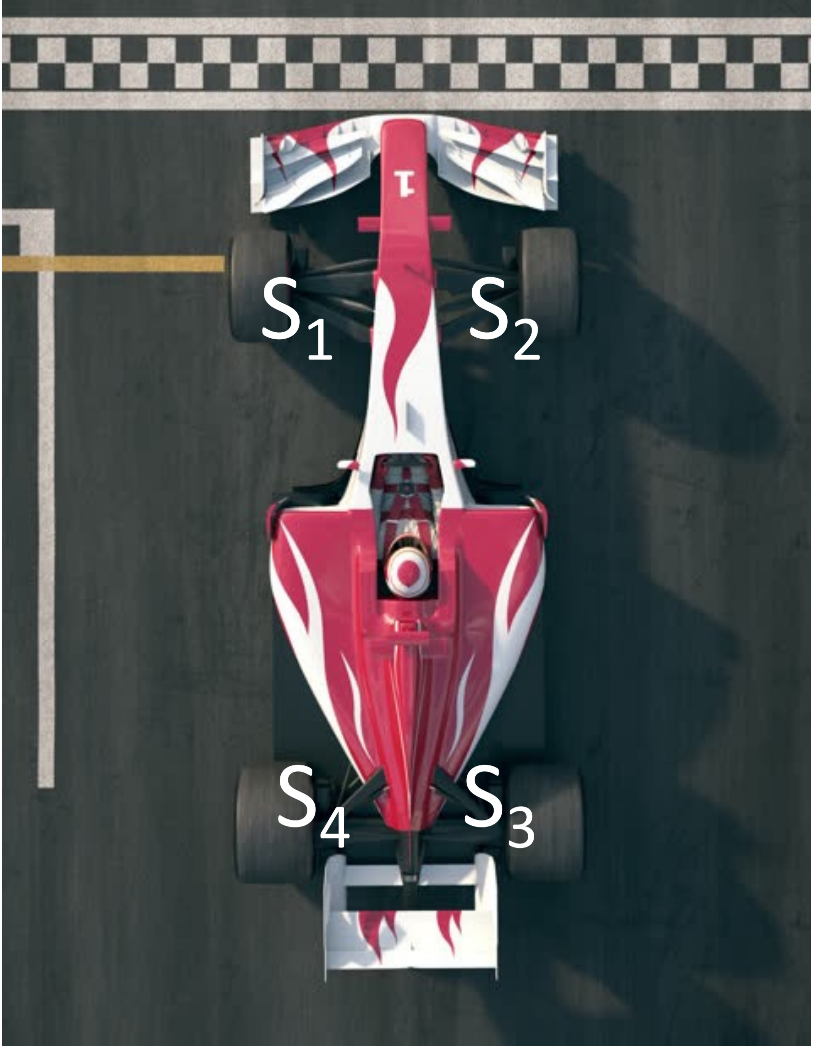
Four sensors measuring rotation speed (spin) at each wheel: S_1, S_2, S_3, S_4

New **composite** measure (feature):

$T_1 = (S_1 + S_2 + S_3 + S_4) / 4 = \frac{1}{4}S_1 + \frac{1}{4}S_2 + \frac{1}{4}S_3 + \frac{1}{4}S_4$

This is a more reliable indicator of car speed

Example:



Features

Four sensors measuring rotation speed (spin) at each wheel: S_1, S_2, S_3, S_4

New **composite** measure (feature):

$T_1 = (S_1 + S_2 + S_3 + S_4) / 4 = \frac{1}{4}S_1 + \frac{1}{4}S_2 + \frac{1}{4}S_3 + \frac{1}{4}S_4$

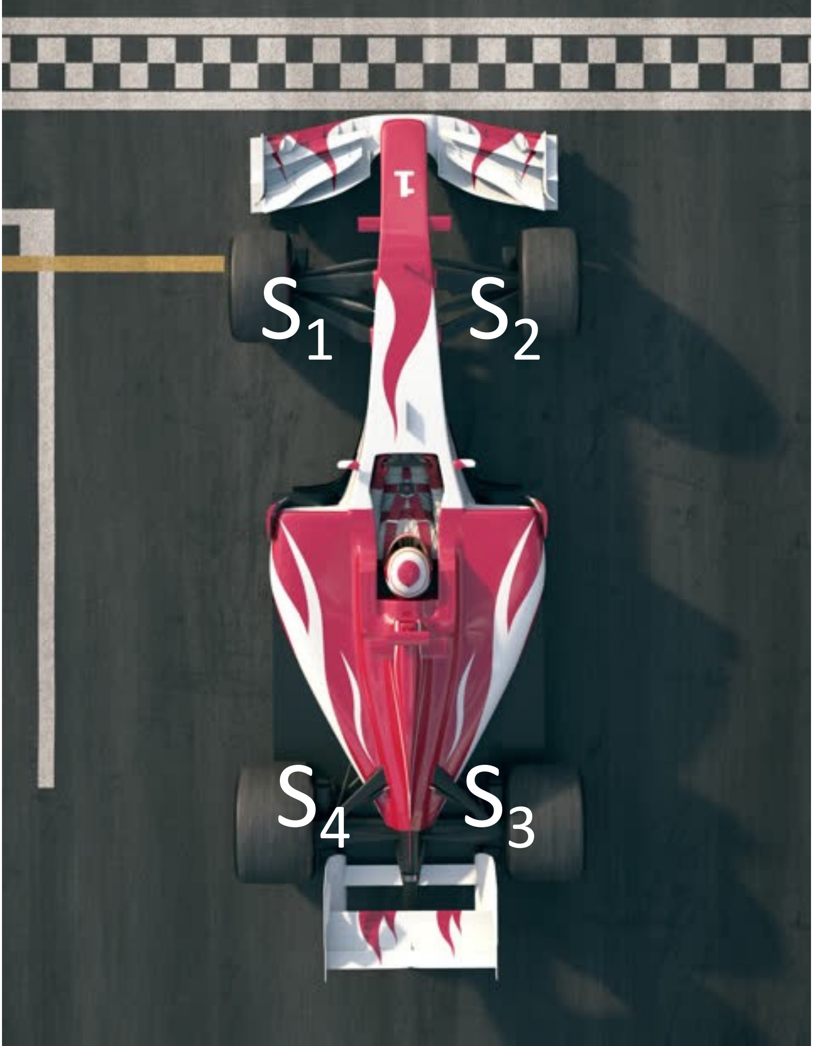
This is a more reliable indicator of car speed

New **composite** measure (feature):

$T_2 = 0.5 \{ (\frac{S_1 + S_3 + S_4}{3}) - S_2\} = \frac{1}{6}S_1 - \frac{1}{2}S_2 + \frac{1}{6}S_3 + \frac{1}{6}S_4$

If this starts to veer away from zero, then tire #2 is spinning faster than the others (possible flat)

Example:



Features

Four sensors measuring rotation speed (spin) at each wheel: S_1, S_2, S_3, S_4

New **composite** measure (feature):

$T_1 = (S_1 + S_2 + S_3 + S_4) / 4 = \frac{1}{4}S_1 + \frac{1}{4}S_2 + \frac{1}{4}S_3 + \frac{1}{4}S_4$

This is a more reliable indicator of car speed

New **composite** measure (feature):

$T_2 = 0.5 \{ (\frac{S_1 + S_3 + S_4}{3}) - S_2 \} = \frac{1}{6}S_1 - \frac{1}{2}S_2 + \frac{1}{6}S_3 + \frac{1}{6}S_4$

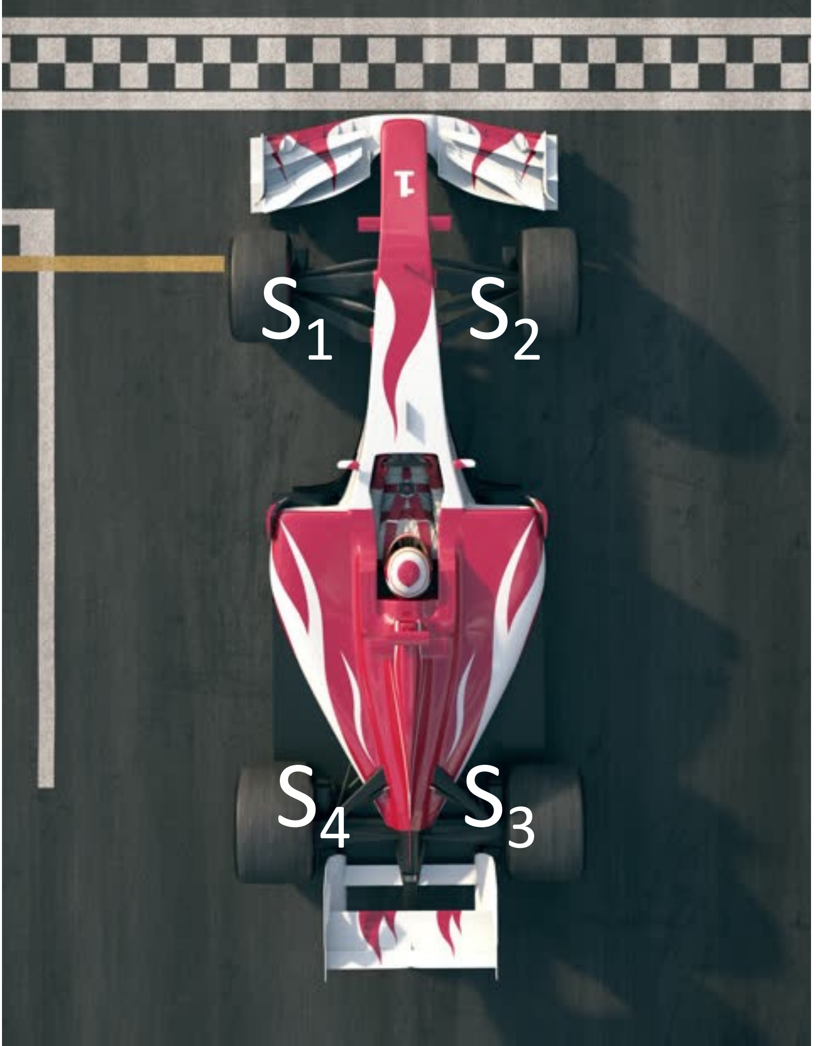
If this starts to veer away from zero, then tire #2 is spinning faster than the others (possible flat)

Similarly,

$T_3 = 0.5 \{ (\frac{S_1 + S_2 + S_4}{3}) - S_3 \}$

$T_4 = 0.5 \{ (\frac{S_1 + S_2 + S_3}{3}) - S_4 \}$

Example:



Features

Original measures (variables):
S_1, S_2, S_3, S_4

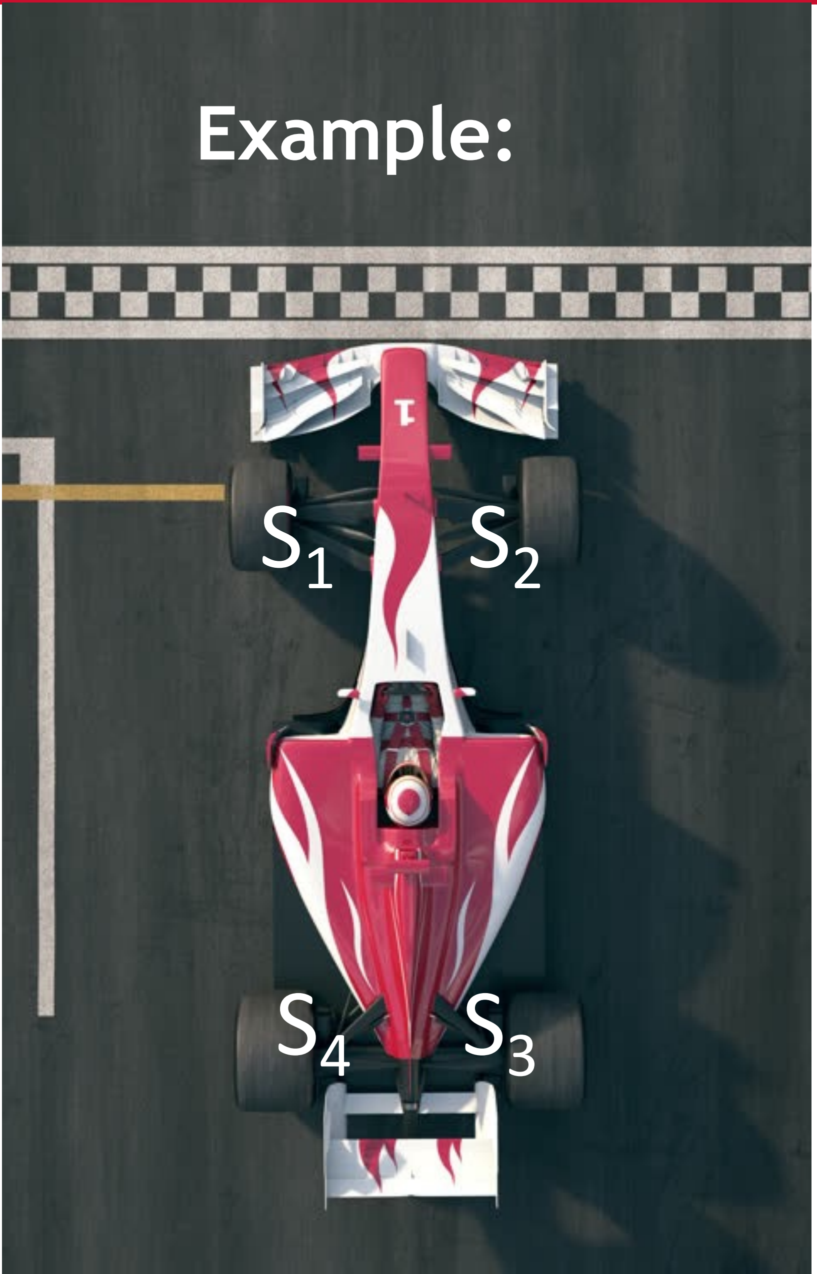
New composite measures (features):

$$T_1 = \frac{1}{4}S_1 + \frac{1}{4}S_2 + \frac{1}{4}S_3 + \frac{1}{4}S_4$$

$$T_2 = \frac{1}{6}S_1 - \frac{1}{2}S_2 + \frac{1}{6}S_3 + \frac{1}{6}S_4$$

$$T_3 = \frac{1}{6}S_1 + \frac{1}{6}S_2 - \frac{1}{2}S_3 + \frac{1}{6}S_4$$

$$T_4 = \frac{1}{6}S_1 + \frac{1}{6}S_2 + \frac{1}{6}S_3 - \frac{1}{2}S_4$$



Features

Original measures (variables):

S_1, S_2, S_3, S_4

New composite measures (features):

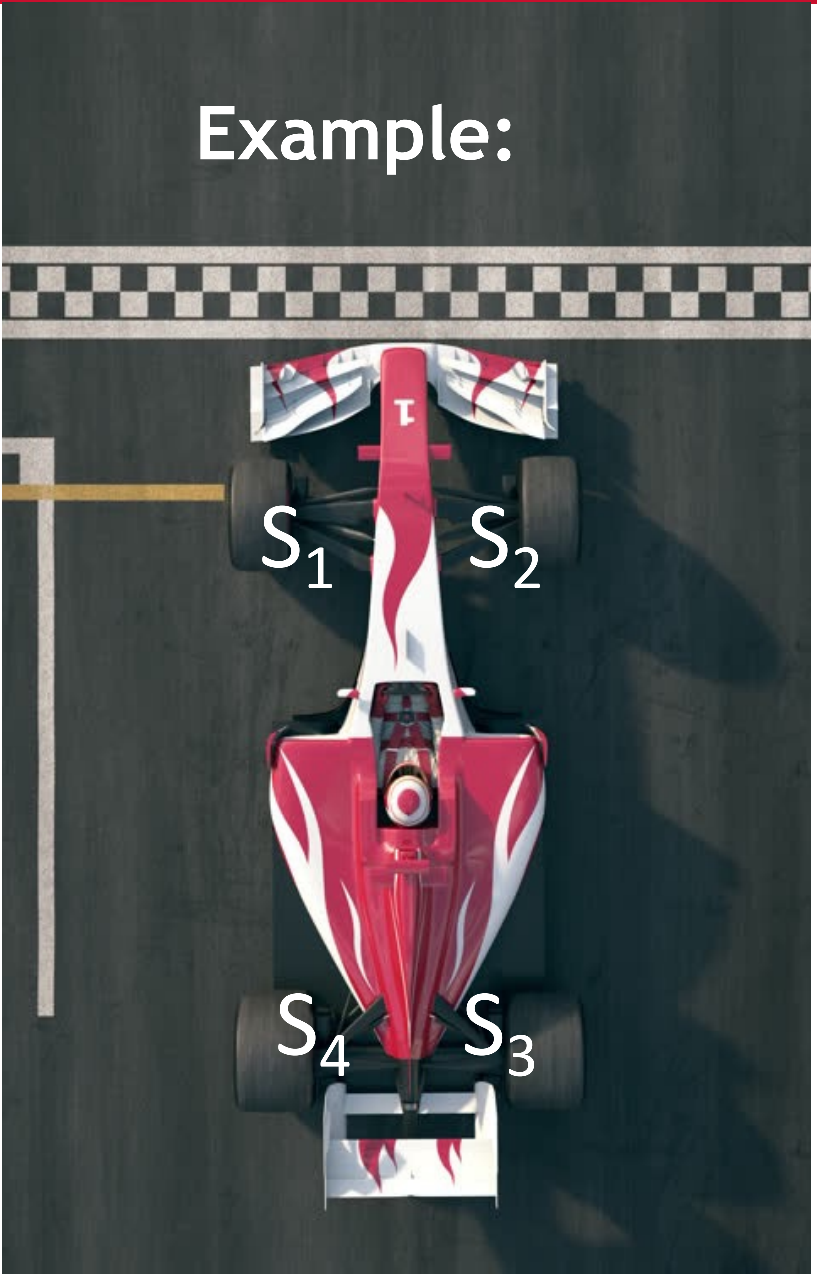
$$T_1 = +\frac{1}{4}S_1 + \frac{1}{4}S_2 + \frac{1}{4}S_3 + \frac{1}{4}S_4$$

$$T_2 = +\frac{1}{6}S_1 - \frac{1}{2}S_2 + \frac{1}{6}S_3 + \frac{1}{6}S_4$$

$$T_3 = +\frac{1}{6}S_1 + \frac{1}{6}S_2 - \frac{1}{2}S_3 + \frac{1}{6}S_4$$

$$T_4 = +\frac{1}{6}S_1 + \frac{1}{6}S_2 + \frac{1}{6}S_3 - \frac{1}{2}S_4$$

Example:



Features

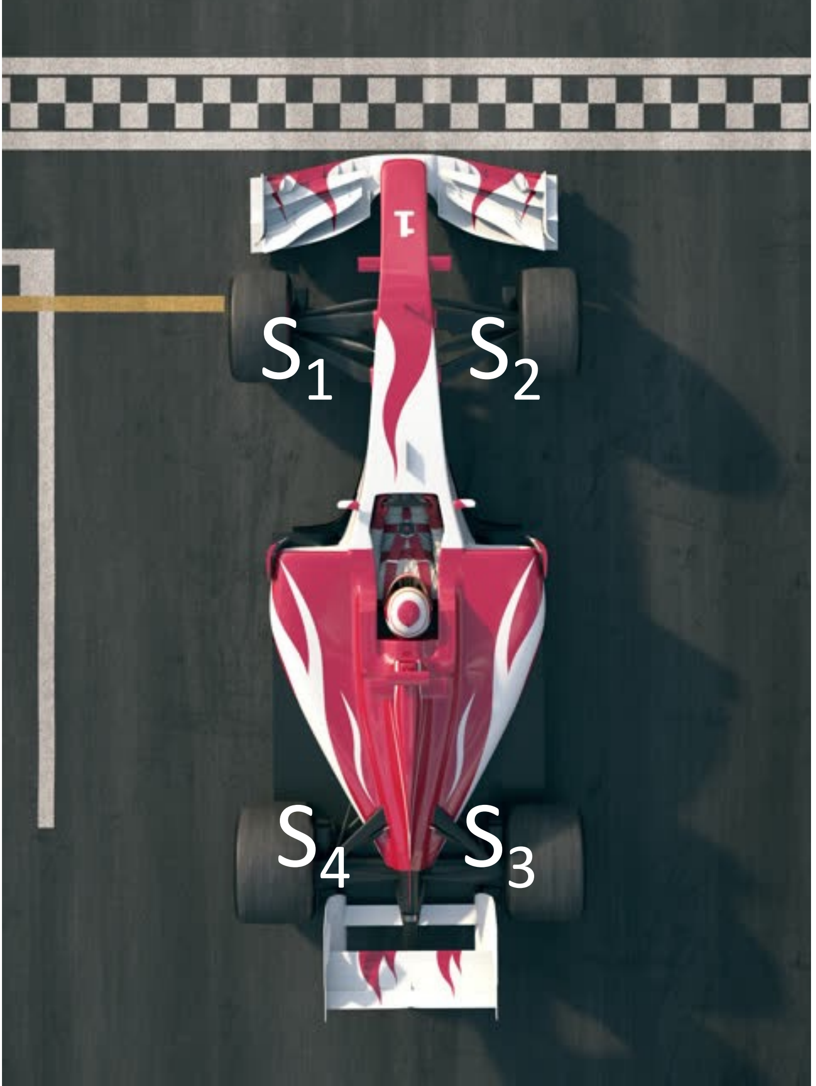
Original measures (variables):

S_1, S_2, S_3, S_4

New composite measures (features):

$$\begin{bmatrix} T_1 \\ T_2 \\ T_3 \\ T_4 \end{bmatrix} = \begin{bmatrix} +\frac{1}{4} & +\frac{1}{4} & +\frac{1}{4} & +\frac{1}{4} \\ +\frac{1}{6} & -\frac{1}{2} & +\frac{1}{6} & +\frac{1}{6} \\ +\frac{1}{6} & +\frac{1}{6} & -\frac{1}{2} & +\frac{1}{6} \\ +\frac{1}{6} & +\frac{1}{6} & +\frac{1}{6} & -\frac{1}{2} \end{bmatrix} \begin{bmatrix} S_1 \\ S_2 \\ S_3 \\ S_4 \end{bmatrix}$$

Example:



Features

Original measures (variables):
S_1, S_2, S_3, S_4

New composite measures (features):

$$T = W^T S, \qquad W^T = \begin{bmatrix} +\frac{1}{4} & +\frac{1}{4} & +\frac{1}{4} & +\frac{1}{4} \\ +\frac{1}{6} & -\frac{1}{2} & +\frac{1}{6} & +\frac{1}{6} \\ +\frac{1}{6} & +\frac{1}{6} & -\frac{1}{2} & +\frac{1}{6} \\ +\frac{1}{6} & +\frac{1}{6} & +\frac{1}{6} & -\frac{1}{2} \end{bmatrix}$$

Principal Component Analysis (PCA)

PCA is a method for computing new features from existing variables according to a generic principle.

PCA will compute the weight matrix W for the new composite measures T_i (which are called principal components) so the data are now measured according to these new composite measures

NOTES: The original variables must be centered (i.e., have mean zero)

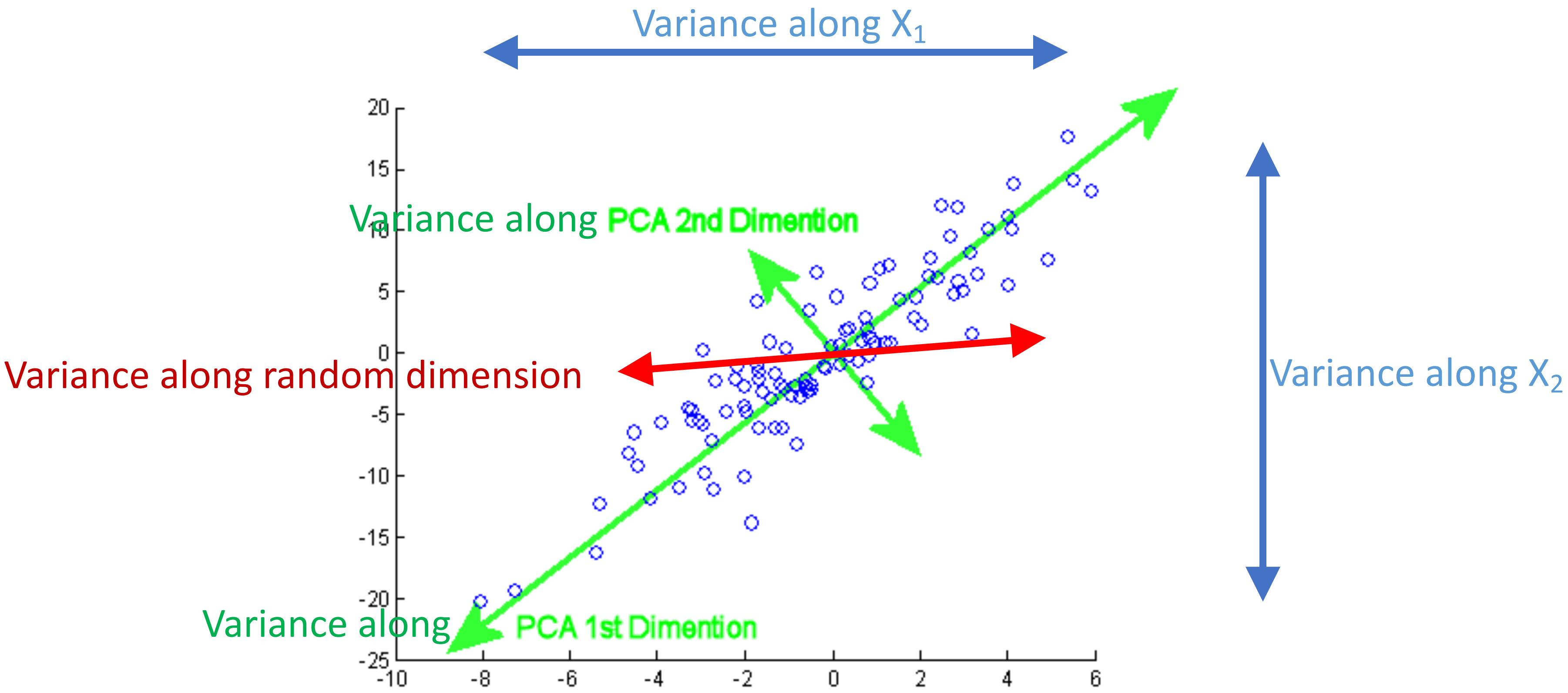
Original X (variables):

Observation ID	S_1	S_2	S_3	S_4
1				
…				
N				

Transformed X (features/components):

Observation ID	T_1	T_2	T_3	T_4
1				
…				
N				

PCA principle: PC_1 is the direction of maximum variance



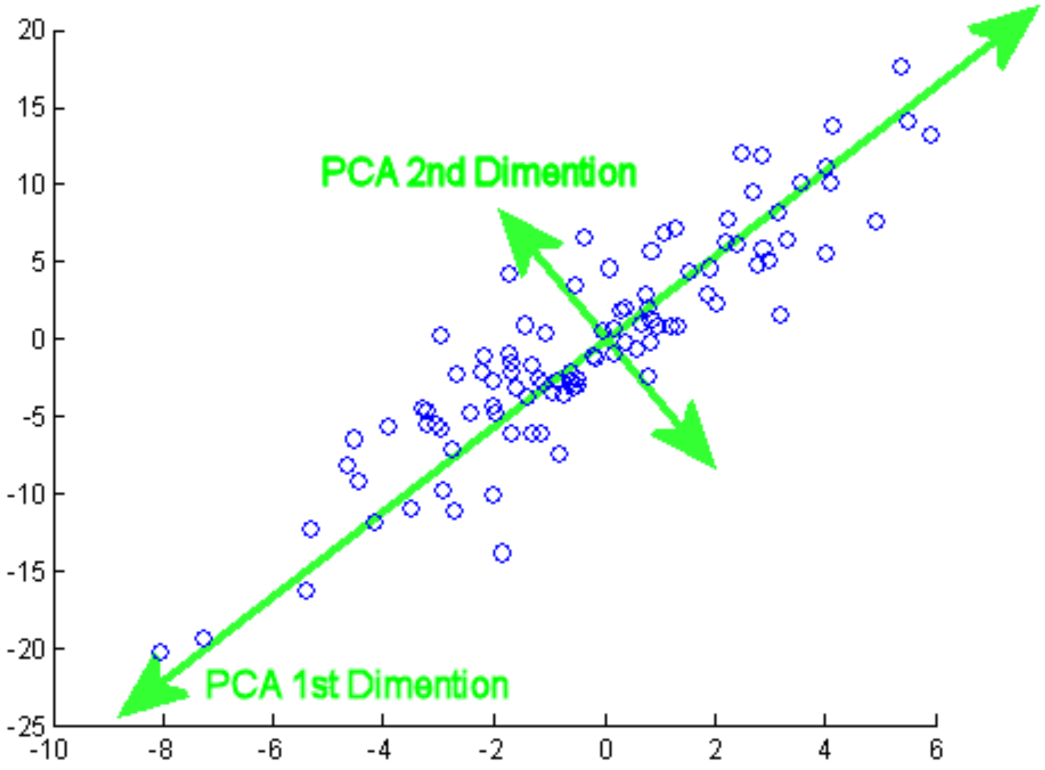
$$Var(X_1) + Var(X_2) = Var(PC_1) + Var(PC_2)$$

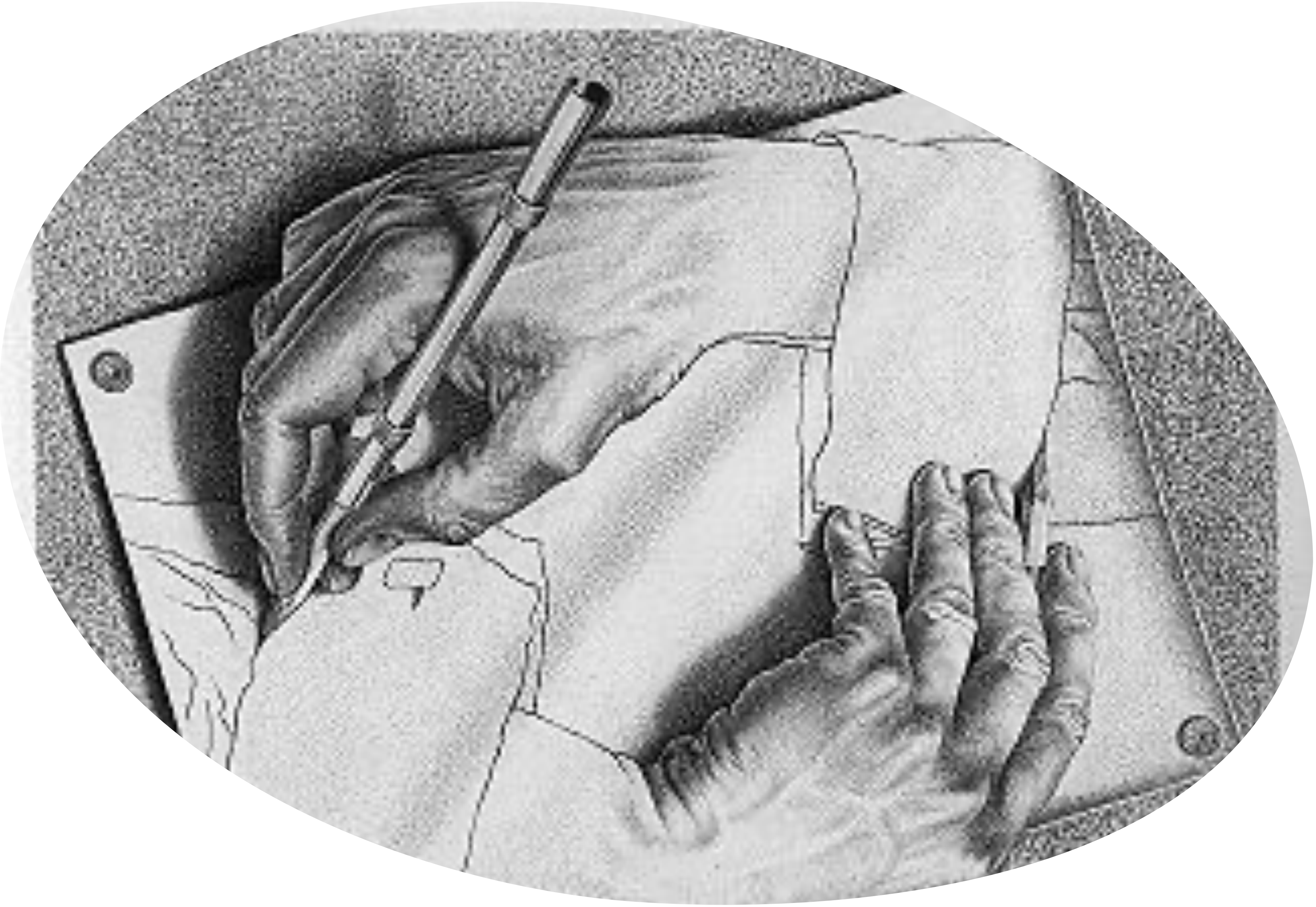
PCA principles, continued

Principal Components are orthogonal to each other

Principal Components are ordered

- every principal component captures less variance than the ones before, i.e., $\mathrm{Var}(PC_1) \geq \mathrm{Var}(PC_2) \geq \ldots$





Hands-on Example: PCA

https://jakevdp.github.io/PythonDataScienceHandbook/05.09-principal-component-analysis.html

What is the curse of dimensionality?

More dimensions, more problems

1-D

If I dropped my keys somewhere along the path between my car and my house, it would take only a few minutes to walk the straight path and find them.

2-D

If I dropped my keys somewhere in my yard while mowing my lawn, it could take me hours to search the whole yard to find them.

3-D

If a dropped my keys in one of the offices in PGH while going door-to-door delivering girl scout cookies, it would take days to search all the building floors to find them.

Dimension Reduction with PCA

Work with only the top few principal components, since they capture most of the variance

Homework Assignment #2
Due Monday, June 24, 11:59 pm (Central)

Ready to move on

Supervised classification using deep learning models

- Perceptron / Neural Nets

Unsupervised clustering using generative models

- Hierarchical clustering
- K-means clustering