Lecture Notes for

Machine Learning in Python

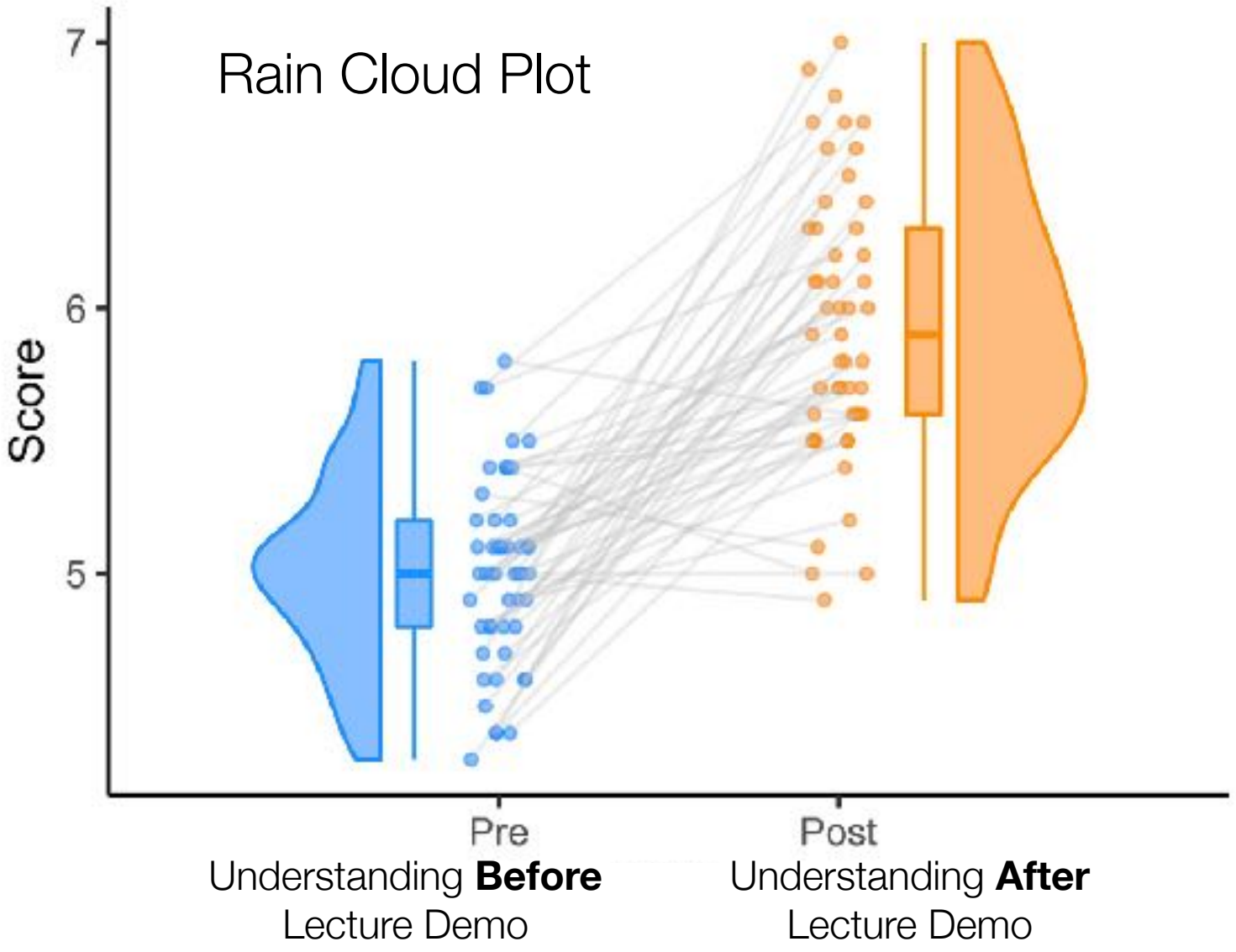
Professor Eric Larson

Visualization and Dimensionality Reduction

Class Logistics and Agenda

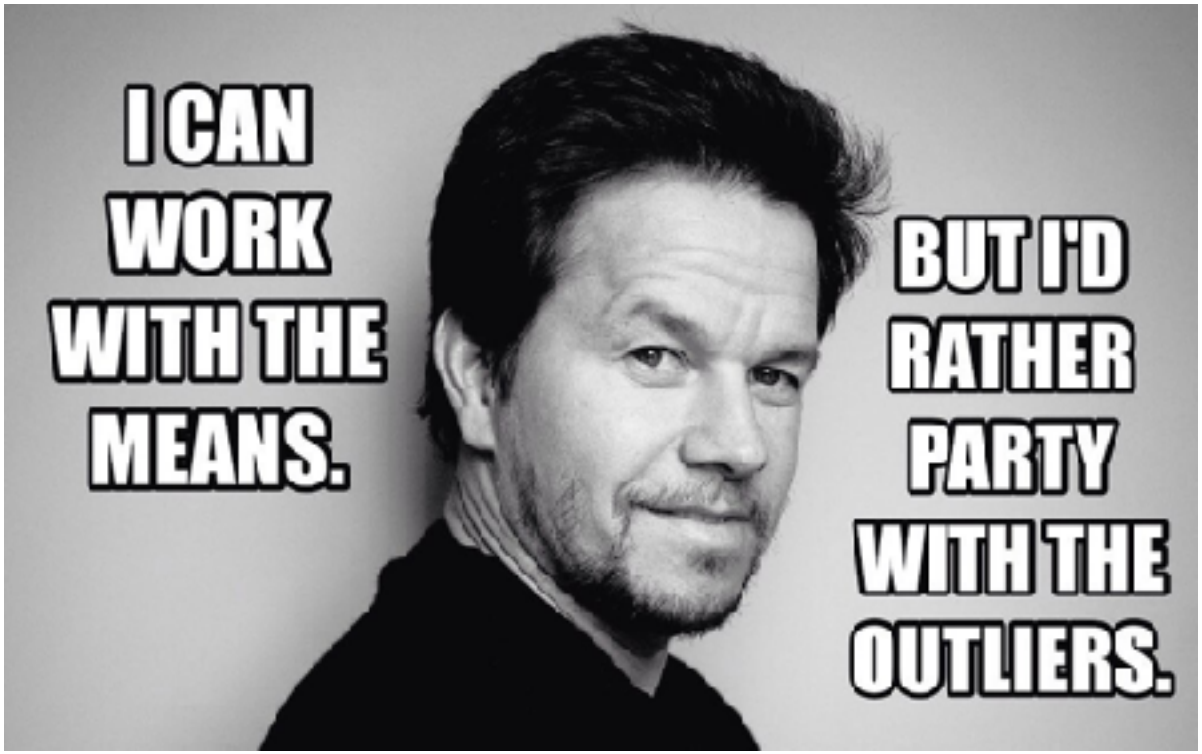
- Dimensionality Reduction
  - PCA
  - Randomized PCA
  - Images Representation with PCA

Last time: visualization



Lab One: Town Hall

Dimensionality Reduction: PCA



Curse of Dimensionality

- When dimensionality increases, data becomes increasingly sparse in the space that it occupies

- Definitions of density and distance between points, which is critical for clustering and outlier detection, become less meaningful

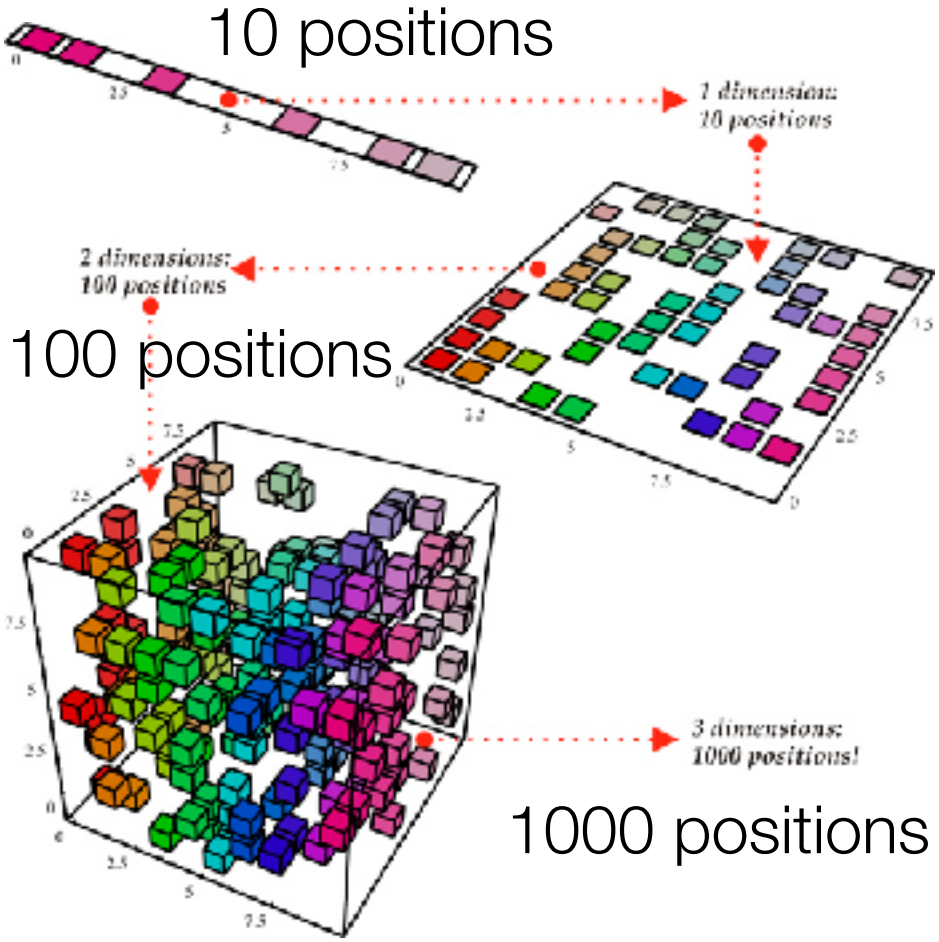


image source: http://www.iro.umontreal.ca/~bengioy/yoshua_en/research_files/CurseDimensionality.jpg

Dimensionality Reduction

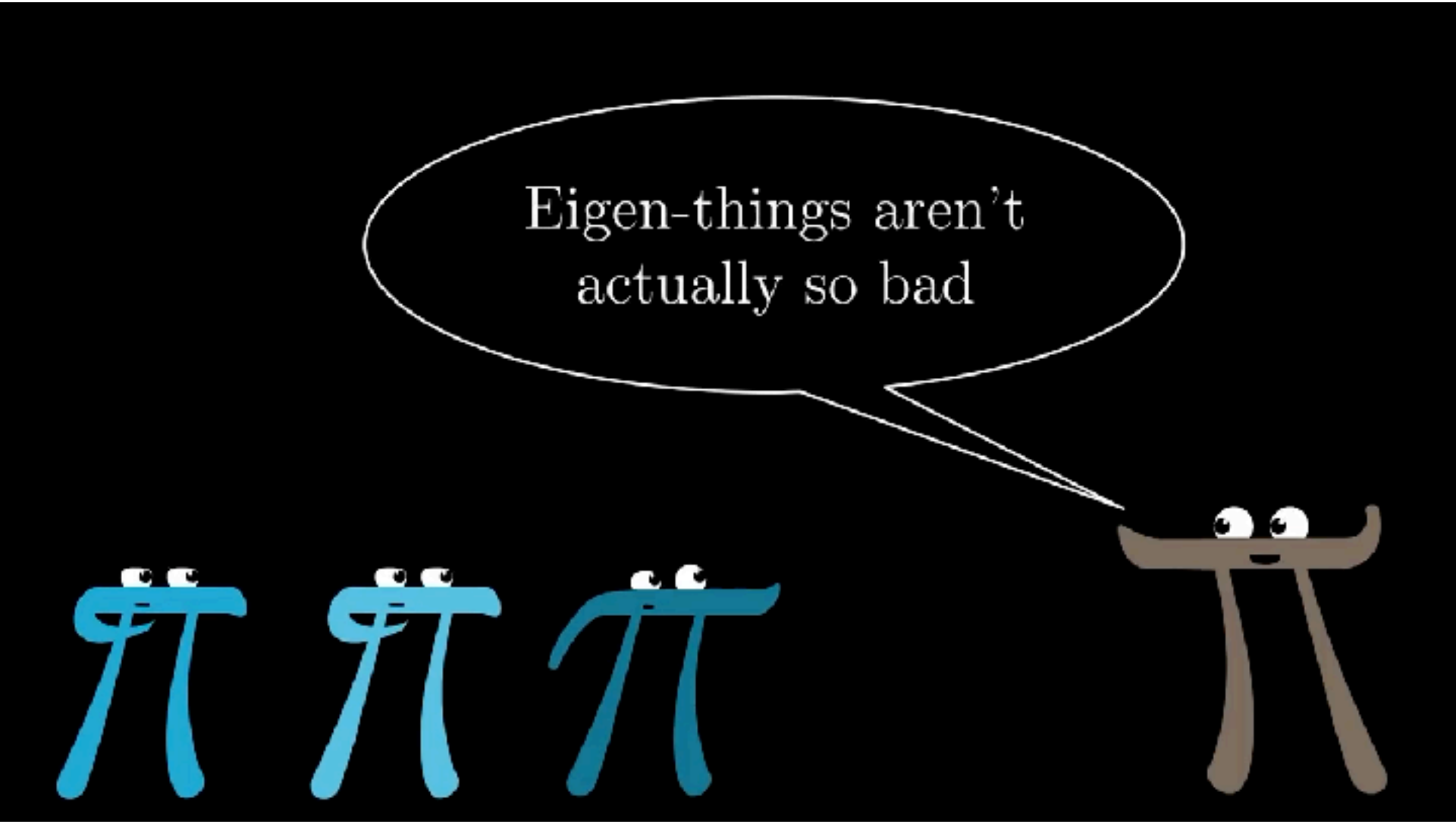
- Purpose:
  - Avoid curse of dimensionality
  - Select subsets of independent features
  - Reduce amount of time and memory required by data mining algorithms
  - Allow data to be more easily visualized
  - May help to eliminate irrelevant features or reduce noise
- Techniques
  - Principle Component Analysis
  - Non-linear PCA
  - Stochastic Neighbor Embedding

I invented PCA…
and *Social Darwinism*

Aside: Eigen Vectors are your friend!

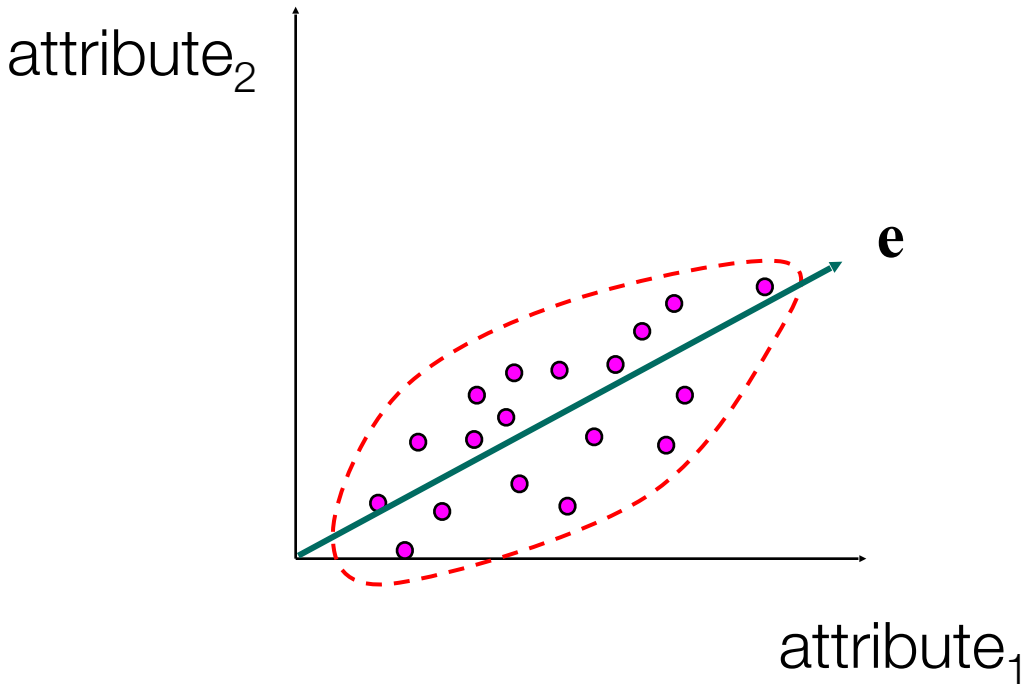
Three Blue One Brown:

https://www.youtube.com/watch?v=PFDu9oVAE-g



Dimensionality Reduction: PCA

- Goal is to find a projection that captures the largest amount of variation in data

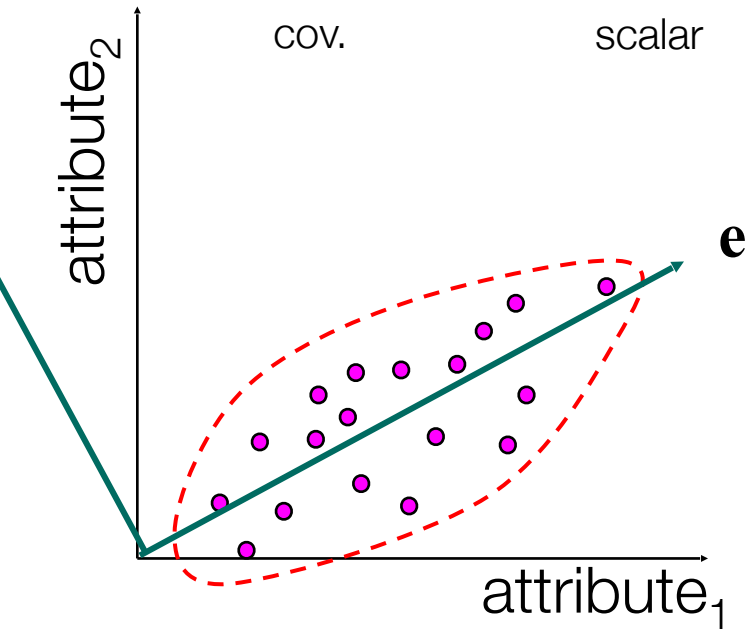


Dimensionality Reduction: PCA

- Find the **eigenvectors** of the **covariance** matrix
- keep the "k" **largest** eigenvectors

$$\mathbf{C} \cdot \mathbf{e} = \lambda \mathbf{e}$$

cov. scalar



E1	E2
0.749	0.662
0.662	-0.749
λ=268.3	λ=1.57

covariance

151.5	132.4
132.4	118.3

	A1	A2
1	14	12.6
2	26	26.6
3	36.3	33.3
4	2.5	3.6
5	15	17.4
6	8	11

	A1	A2
1	-2.96	-4.82
2	9.03	9.18
3	19.33	15.88
4	-14.46	-13.82
5	-1.96	-0.02
6	-8.96	-6.42

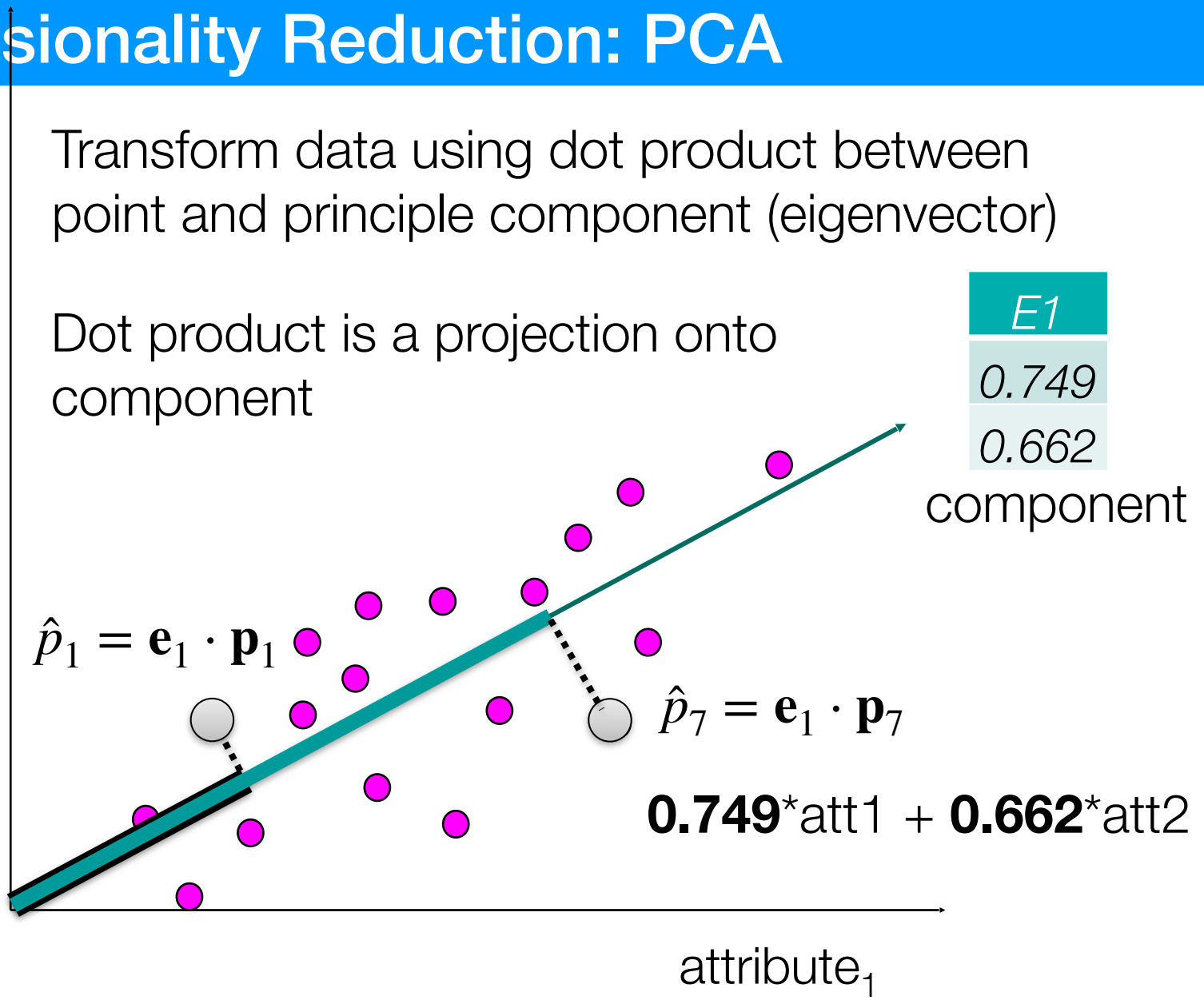
normalize: zero mean

optional: unit std

Dimensionality Reduction: PCA

Transform data using dot product between point and principle component (eigenvector)

Dot product is a projection onto component



Dimensionality Reduction: PCA

	A1	A2
P1	-2.96	-4.82
P2	9.03	9.18
P3	19.33	15.88
P4	-14.46	-13.82
P5	-1.96	-0.02
P6	-8.96	-6.42

zero mean

0.749*att1 + **0.662***att2

	A_{new}
$\hat{P}1$	-5.4
$\hat{P}2$	12.8
$\hat{P}3$	25.0
$\hat{P}4$	-19.9
$\hat{P}5$	-1.4
$\hat{P}6$	-10.9

Transformed Features

This projection is called a **Transform**
known as the **Karhunen-Loève Transform (KLT)**

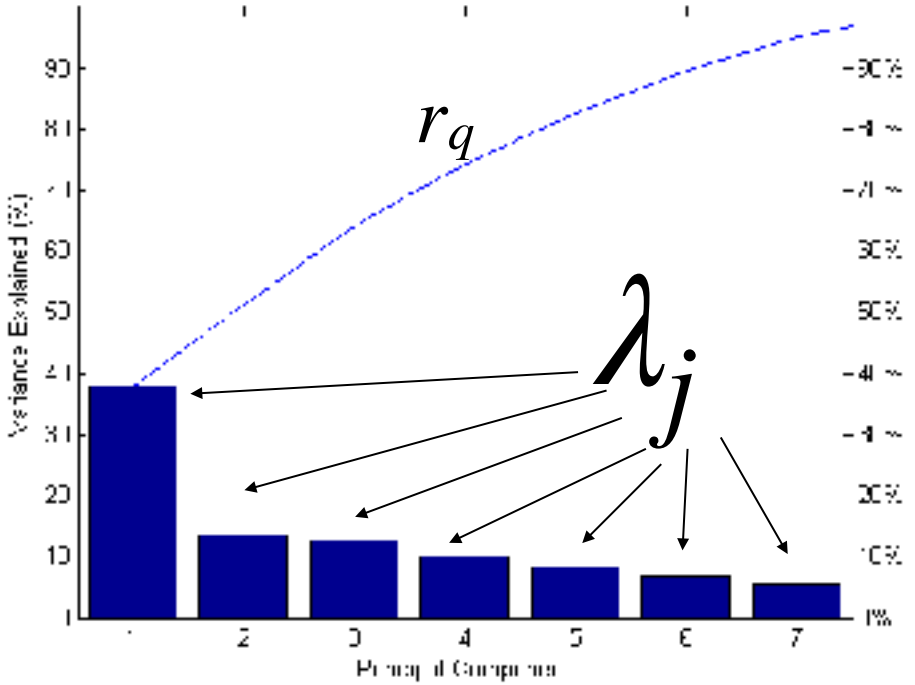
Explained Variance

- Each principle component **explains** a certain **amount of variation** in the data.
- This explained variation is **encoded** in the **eigenvalues** of each **eigenvector**

sum of q largest eigenvalues

$$r_q = \frac{\sum_{j=1}^{q} \lambda_j}{\sum_{\forall i} \lambda_i}$$

sum of all eigenvalues



Dimensionality Reduction: PCA

Genetic profiles distilled to 2 components

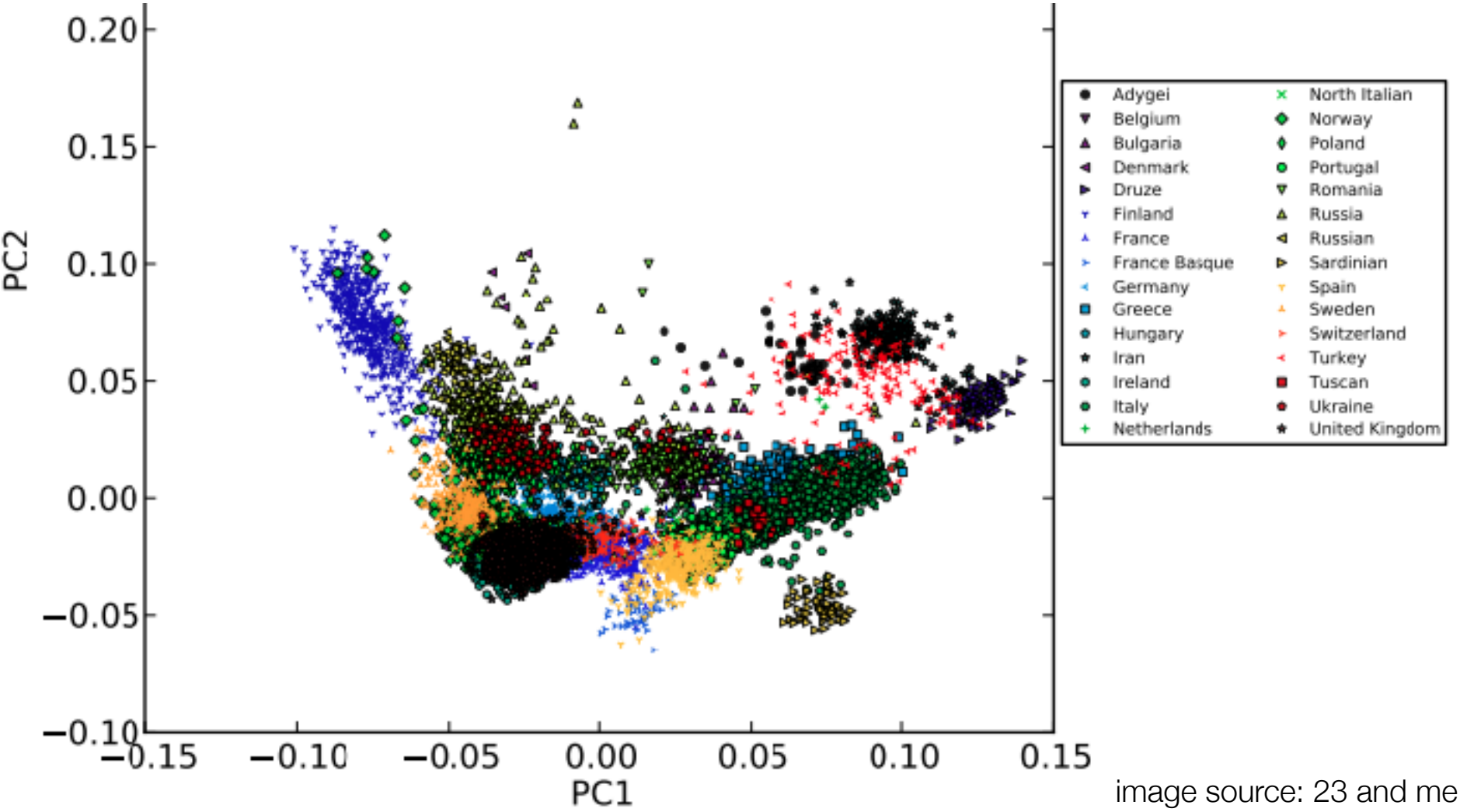
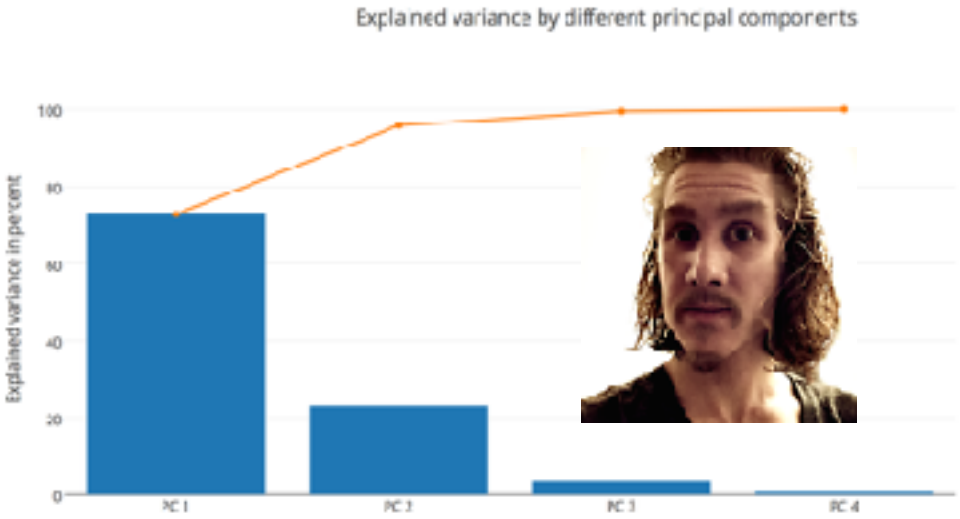


image source: 23 and me

Dimensionality Reduction: PCA

- Need more help with the PCA algorithm (and python)?
  - check out Sebastian Raschka's notebooks:
- http://nbviewer.ipython.org/github/rasbt/pattern_classification/blob/master/dimensionality_reduction/projection/principal_component_analysis.ipynb

Or check out PCA for dummies: https://georgemdallas.wordpress.com/2013/10/30/principal-component-analysis-4-dummies-eigenvectors-eigenvalues-and-dimension-reduction/



Dimension Reduction

Demo

`04.Dimension Reduction and Images.ipynb`

PCA

biplots

Other Tutorials:

http://scikit-learn.org/stable/auto_examples/decomposition/plot_pca_vs_lda.html#example-decomposition-plot-pca-vs-lda-py

http://nbviewer.ipython.org/github/ogrisel/notebooks/blob/master/Labeled%20Faces%20in%20the%20Wild%20recognition.ipynb

Self Test ML2b.1

Principal Components Analysis works well for categorical data by design.

A. True

B. False

C. It doesn't but people do it anyway

Mutual Correspondence Analysis

	Eye Color
1	*Blue*
2	*Brown*
3	*Blue*
4	*Hazel*
5	*Brown*
6	*Brown*

OHE

	Eye Color		
1	*1*	*0*	*0*
2	*0*	*1*	*0*
3	*1*	*0*	*0*
4	*0*	*0*	*1*
5	*0*	*1*	*0*
6	*0*	*1*	*0*

~PCA

	A1	*A2*
1	*0.79*	*-0.30*
2	*-0.60*	*-0.13*
3	*0.79*	*-0.30*
4	*0.24*	*0.99*
5	*-0.60*	*-0.13*
6	*-0.60*	*-0.13*

Dimensionality Reduction: Randomized PCA

- **Problem**: PCA on all that data can take a while to compute
  - What if the number of instances is gigantic?
  - What if the number of dimensions is gigantic?
- What if we partially construct the covariance matrix with a lower rank matrix?
  - By **transforming** our table data, A, with another orthogonal matrix, Q, we can **approximate** the **covariance matrix**, but with **lower rank**
  - Gives a matrix with typically good enough precision of actual eigenvectors, like using SVD. $\boldsymbol{QQ^{\mathrm{T}}A}$ is surrogate

Example Objective

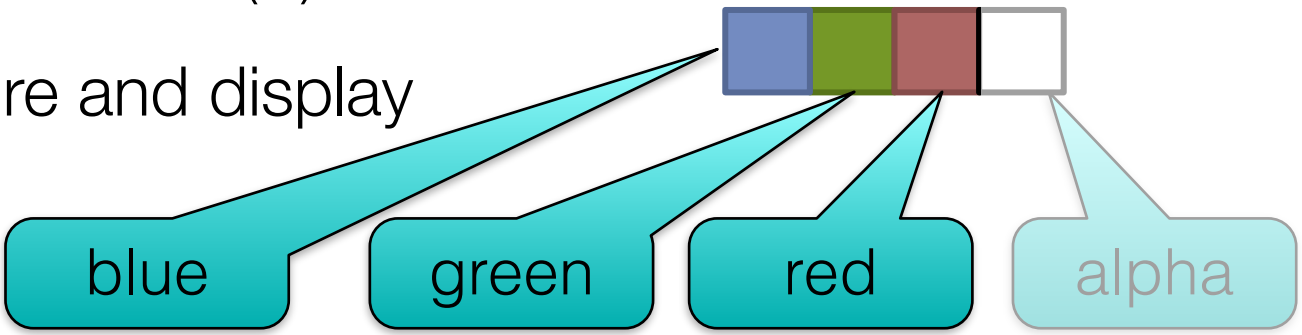
$$\|\boldsymbol{A} - \boldsymbol{QQ}^* \boldsymbol{A}\| \leq \left[1 + 11\sqrt{k+p} \cdot \sqrt{\min\{m,n\}}\right] \sigma_{k+1}$$

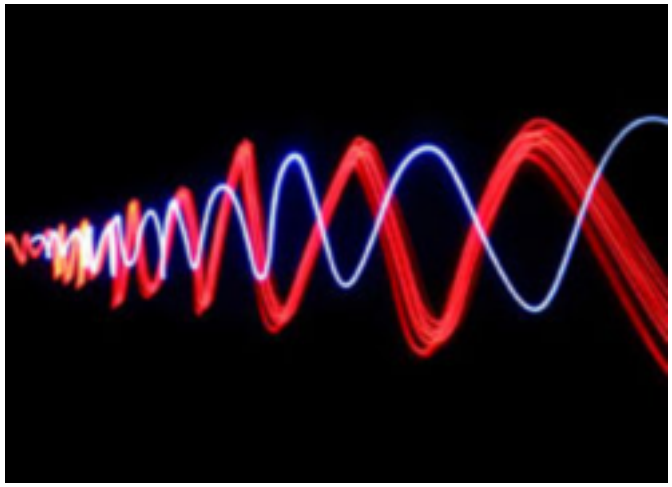
Halko, et al., (2009) Finding structure with randomness: Stochastic algorithms for constructing approximate matrix decompositions. https://arxiv.org/pdf/0909.4061.pdf

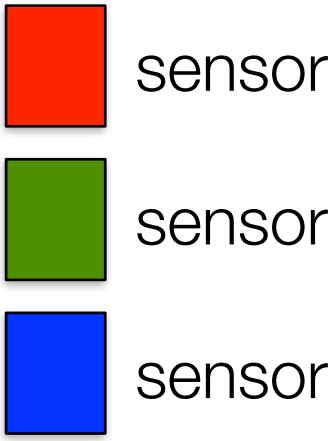
Image Processing and Representation

Images as data

- an image can be represented in many ways
- most common format is a matrix of pixels
  - each “pixel” is BGR(A)
- used for capture and display







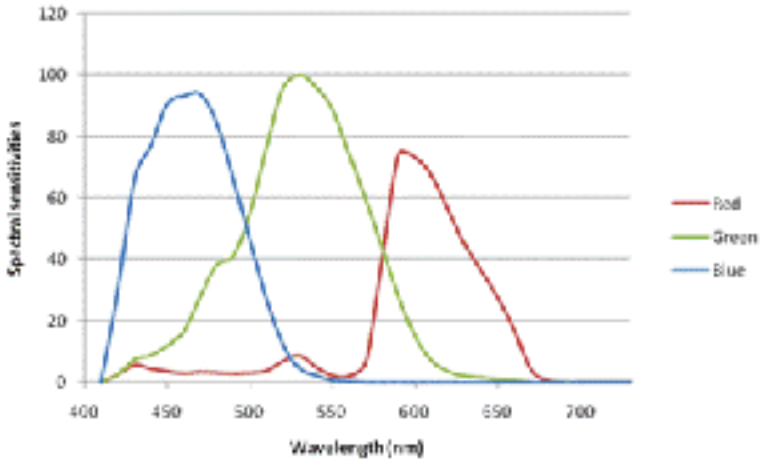


Image Representation

- need a compact representation

- **grayscale**

  0.3**R*+0.59**G*+0.11**B*,

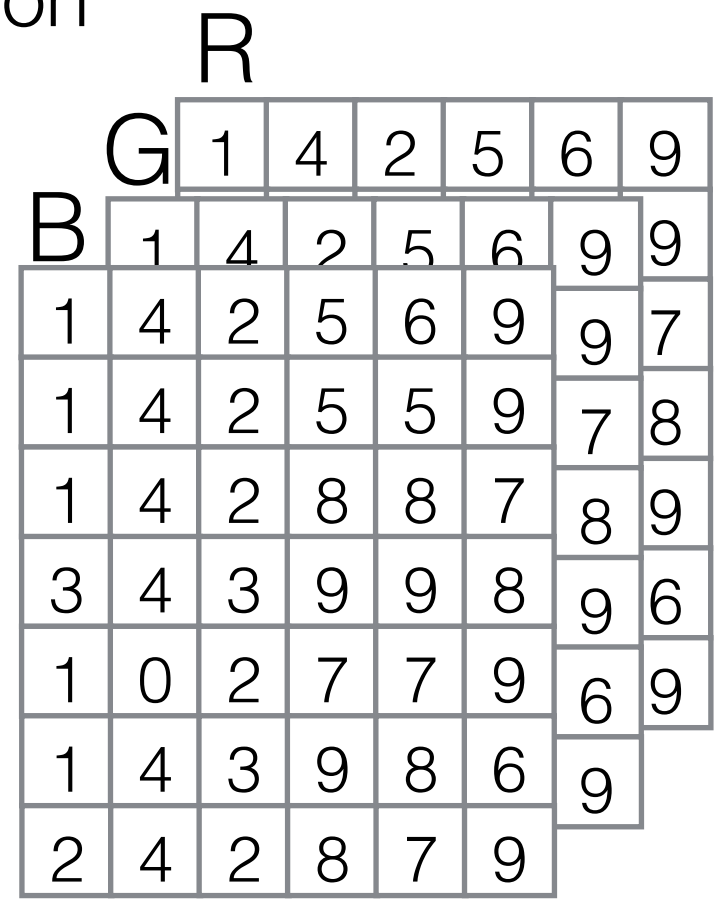
  "luminance"

gray

1	4	2	5	6	9
1	4	2	5	5	9
1	4	2	8	8	7
3	4	3	9	9	8
1	0	2	7	7	9
1	4	3	9	8	6
2	4	2	8	7	9

Numpy Matrix

image[rows, cols]



Numpy Matrix

image[rows, cols, channels]

Image Representation, Features

Problem: need to represent image as table data

- need a compact representation

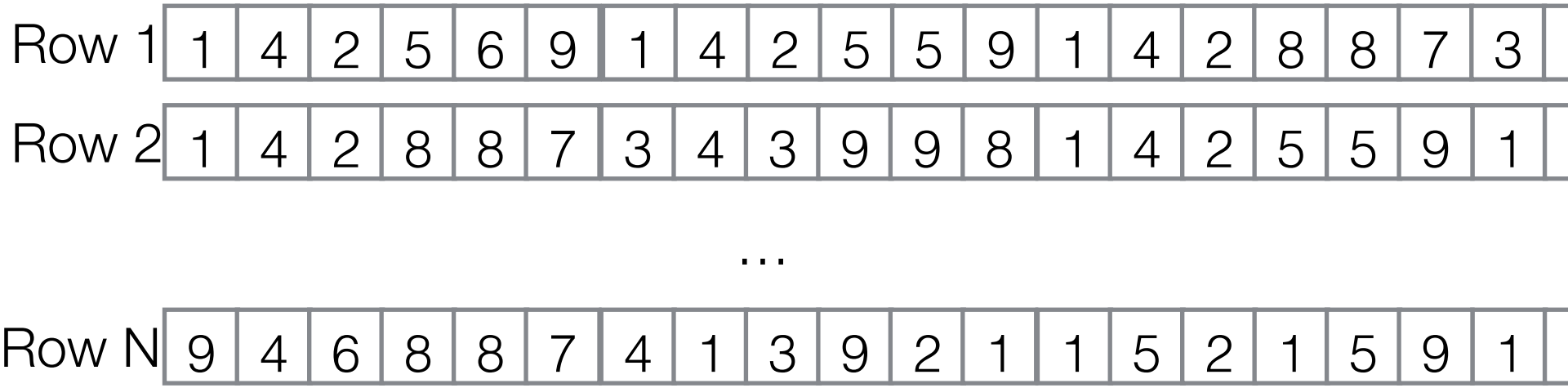
1	4	2	5	6	9
1	4	2	5	5	9
1	4	2	8	8	7
3	4	3	9	9	8
1	0	2	7	7	9
1	4	3	9	8	6
2	4	2	8	7	9

Image Representation, Features

Problem: need to represent image as table data

- need a compact representation

Solution: row concatenation (also, vectorizing)



Self test: 3a-1

- When vectorizing images into table data, each "feature column" corresponds to:
  - a. the value (color) of pixel
  - b. the spatial location of a pixel in the image
  - c. the size of the image
  - d. the spatial location and color channel of a pixel in an image

Dimension Reduction with Images

Demo

Images Representation
in PCA and
Randomized PCA

`04.Dimension Reduction and Images.ipynb`