Warm-up as you walk in

1. https://www.sporcle.com/games/MrChewypoo/minimalist_disney

2. https://www.sporcle.com/games/Stanford0008/minimalist-cartoons-slideshow

3. https://www.sporcle.com/games/MrChewypoo/minimalist

Plan

Last time

- Generative Models

Today

- Wrap-up Generative Models
  - Naïve Bayes
  - Combining MAP and Generative
- Dimensionality Reduction
  - Autoencoders
  - Principal Component Analysis

Wrap-up Generative Models

Previous lecture slides

10-315
Introduction to ML

Deminsionality Reduction: PCA, Autoencoders, and Feature Learning

Instructor: Pat Virtue

Learning Paradigms

Paradigm	**Data**
Supervised	$\mathcal{D} = \{\mathbf{x}^{(i)}, y^{(i)}\}_{i=1}^{N}$ $\quad \mathbf{x} \sim p^*(\cdot)$ and $y = c^*(\cdot)$
$\hookrightarrow$ Regression	$y^{(i)} \in \mathbb{R}$
$\hookrightarrow$ Classification	$y^{(i)} \in \{1, \ldots, K\}$
$\hookrightarrow$ Binary classification	$y^{(i)} \in \{+1, -1\}$
$\hookrightarrow$ Structured Prediction	$\mathbf{y}^{(i)}$ is a vector
Unsupervised	$\mathcal{D} = \{\mathbf{x}^{(i)}\}_{i=1}^{N}$ $\quad \mathbf{x} \sim p^*(\cdot)$
Semi-supervised	$\mathcal{D} = \{\mathbf{x}^{(i)}, y^{(i)}\}_{i=1}^{N_1} \cup \{\mathbf{x}^{(j)}\}_{j=1}^{N_2}$
Online	$\mathcal{D} = \{(\mathbf{x}^{(1)}, y^{(1)}), (\mathbf{x}^{(2)}, y^{(2)}), (\mathbf{x}^{(3)}, y^{(3)}), \ldots\}$
Active Learning	$\mathcal{D} = \{\mathbf{x}^{(i)}\}_{i=1}^{N}$ and can query $y^{(i)} = c^*(\cdot)$ at a cost
Imitation Learning	$\mathcal{D} = \{(s^{(1)}, a^{(1)}), (s^{(2)}, a^{(2)}), \ldots\}$
Reinforcement Learning	$\mathcal{D} = \{(s^{(1)}, a^{(1)}, r^{(1)}), (s^{(2)}, a^{(2)}, r^{(2)}), \ldots\}$

y missing

Slide credit: CMU MLD, Matt Gormley

Outline

Dimensionality Reduction

- High-dimensional data
- Low dimensional representations

Autoencoders

Feature Learning

Principal Component Analysis (PCA)

- Examples: 2D and 3D
- PCA algorithm
- PCA, eigenvectors, and eigenvalues
- PCA objective and optimization

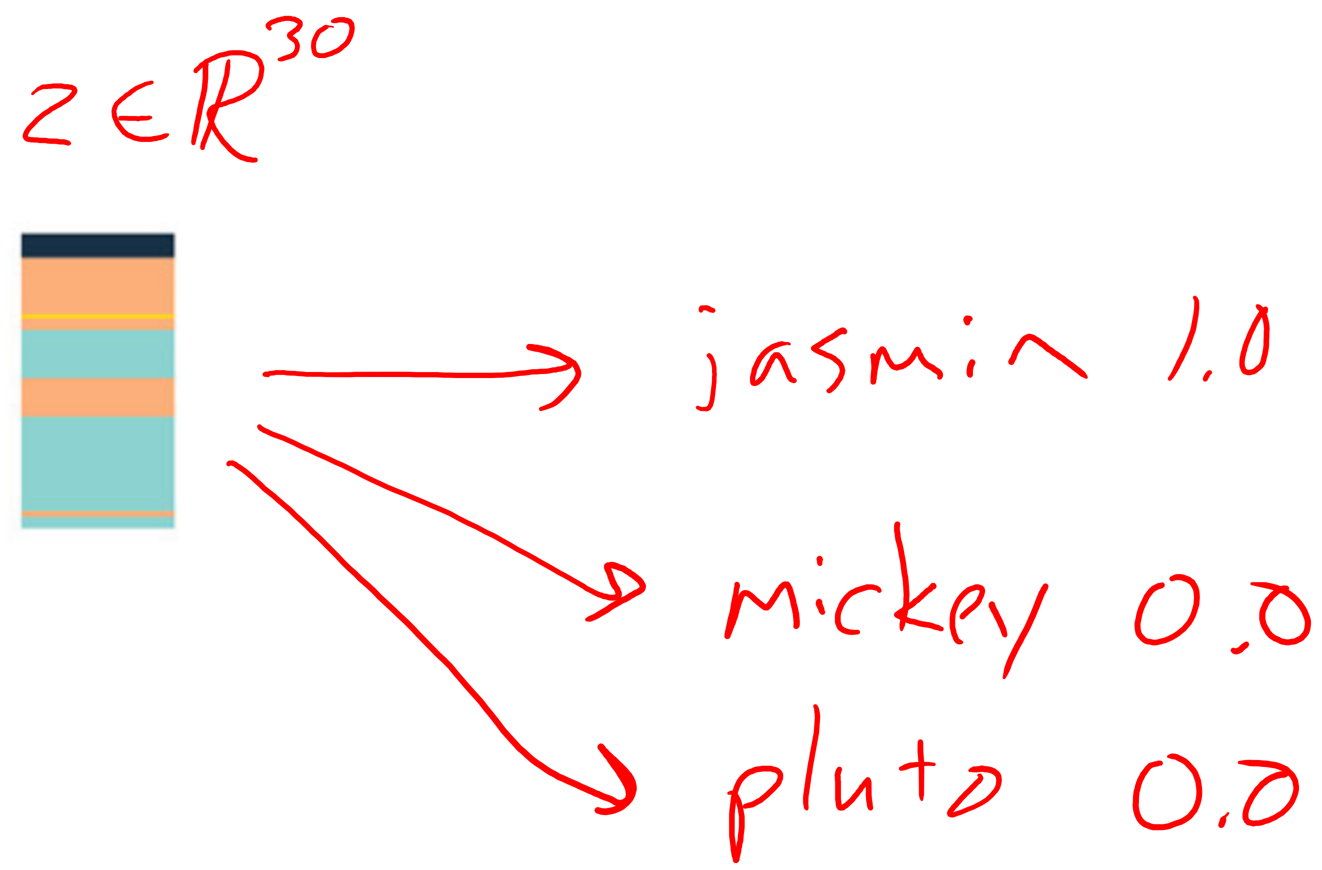
Warm-up as you log in

1. https://www.sporcle.com/games/MrChewypoo/minimalist_disney

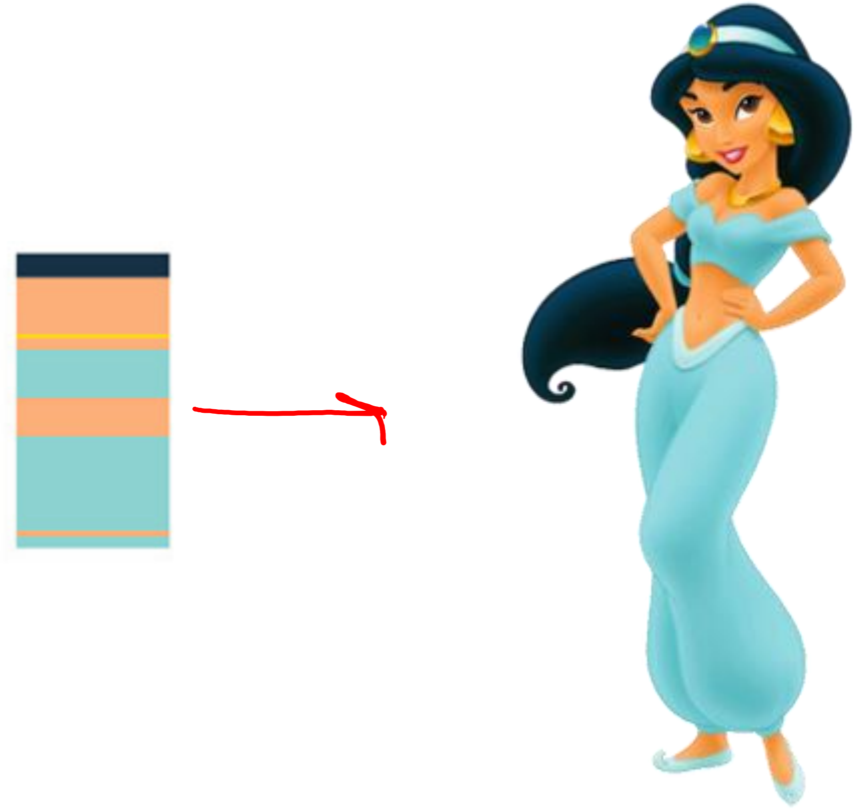
2. https://www.sporcle.com/games/Stanford0008/minimalist-cartoons-slideshow

3. https://www.sporcle.com/games/MrChewypoo/minimalist

Dimensionality Reduction



Dimensionality Reduction



Dimensionality Reduction

$x \in \mathbb{R}^{1000000}$

$z \in \mathbb{R}^{30}$

$x' \in \mathbb{R}^{1000\,000}$

Dimensionality Reduction

x

$\vec{z} \in \mathbb{R}^{30}$

x'

$x' = g(f(x))$

$\longrightarrow \| \vec{x}^{(i)} - \vec{x}'^{(i)} \|_2^2 \longleftarrow$

Dimensionality Reduction

Dimensionality Reduction

For each $\vec{x}^{(i)} \in \mathbb{R}^M$ find representation $\vec{z}^{(i)} \in \mathbb{R}^K$ where $K \ll M$

High Dimension Data

Examples of high dimensional data:

- High resolution images (millions of pixels)

Dimensionality Reduction

http://timbaumann.info/svd-image-compression-demo/

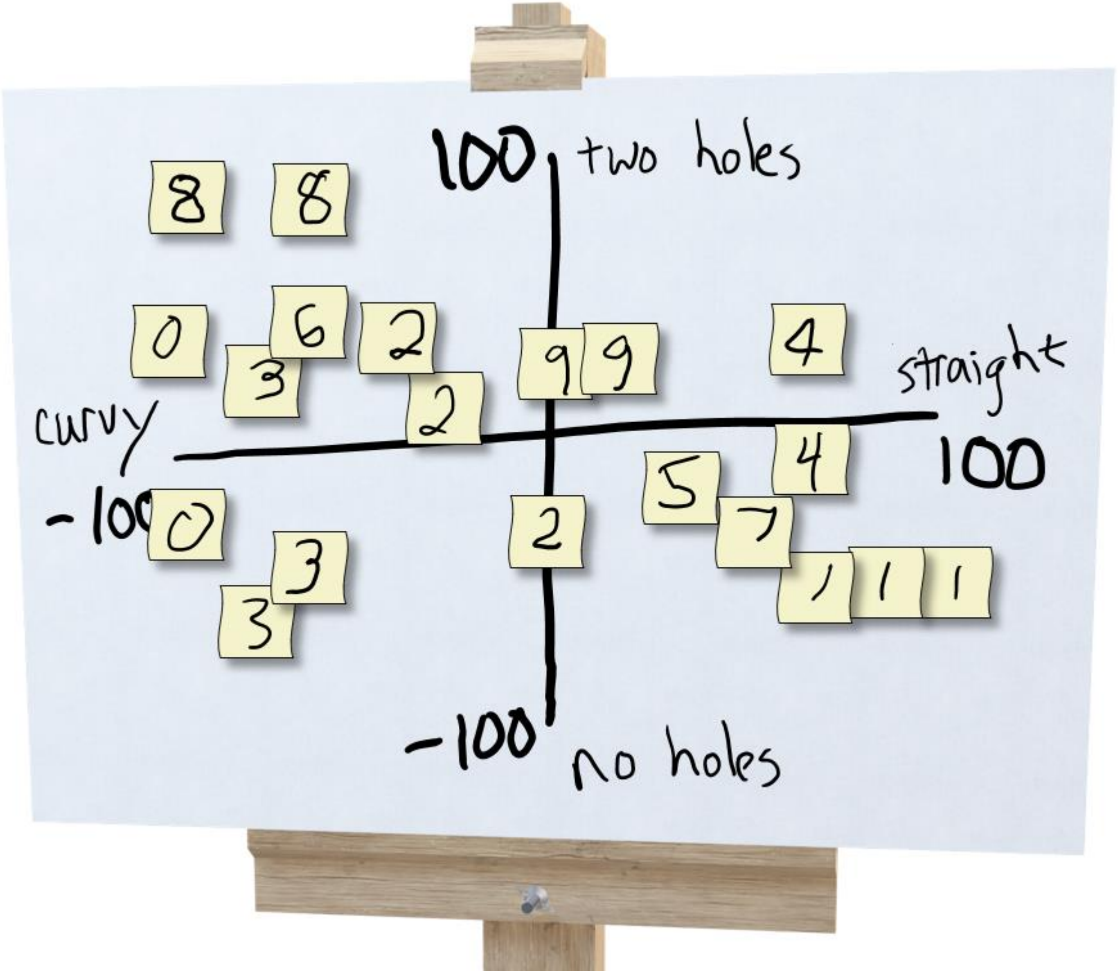
https://cs.stanford.edu/people/karpathy/convnetjs/demo/autoencoder.html

Autoencoders

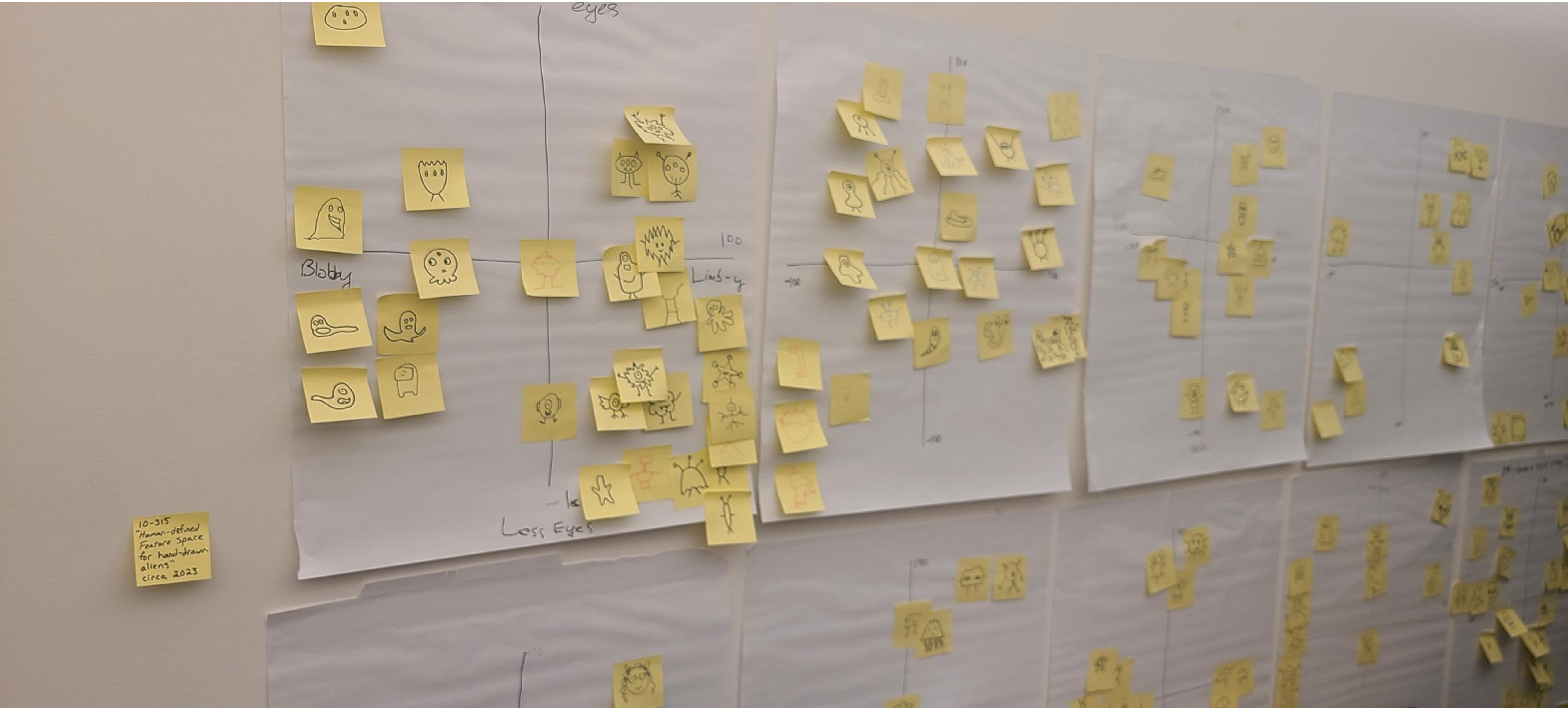
Exercise: Human-defined Feature Space

Step 4: Creation!

1. Select three students: A,B,C
2. Student A draws a new digit and hands it to student B
3. Student B thinks about where to plot it and comes up with a 2-D coordinate, (x, y)
4. Student C looks at the coordinate and the plot (but not the drawing from A) and draws a new digit



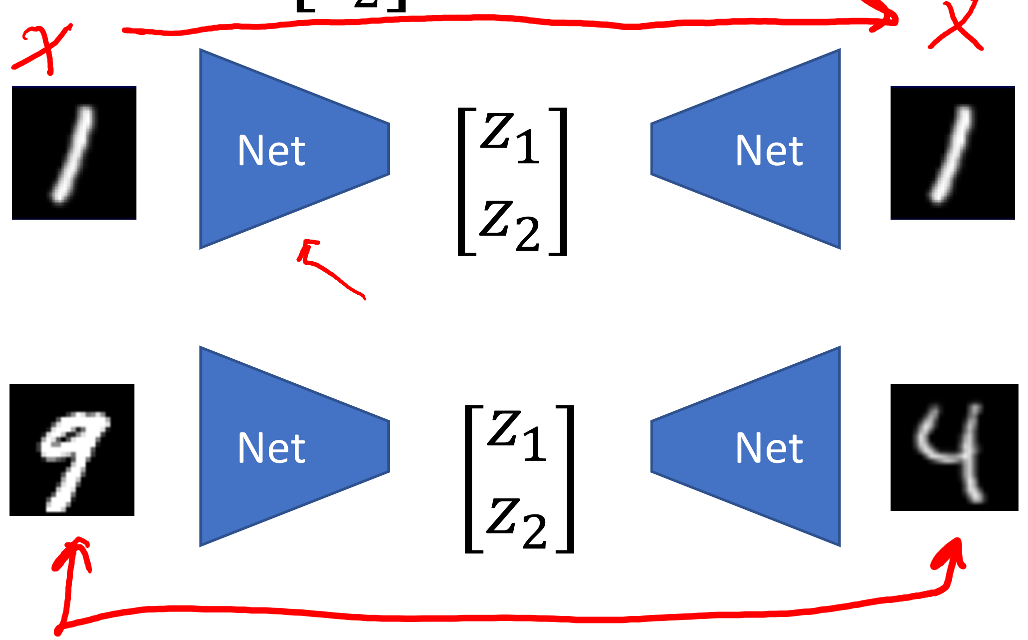
Exercise: Human-defined Feature Space

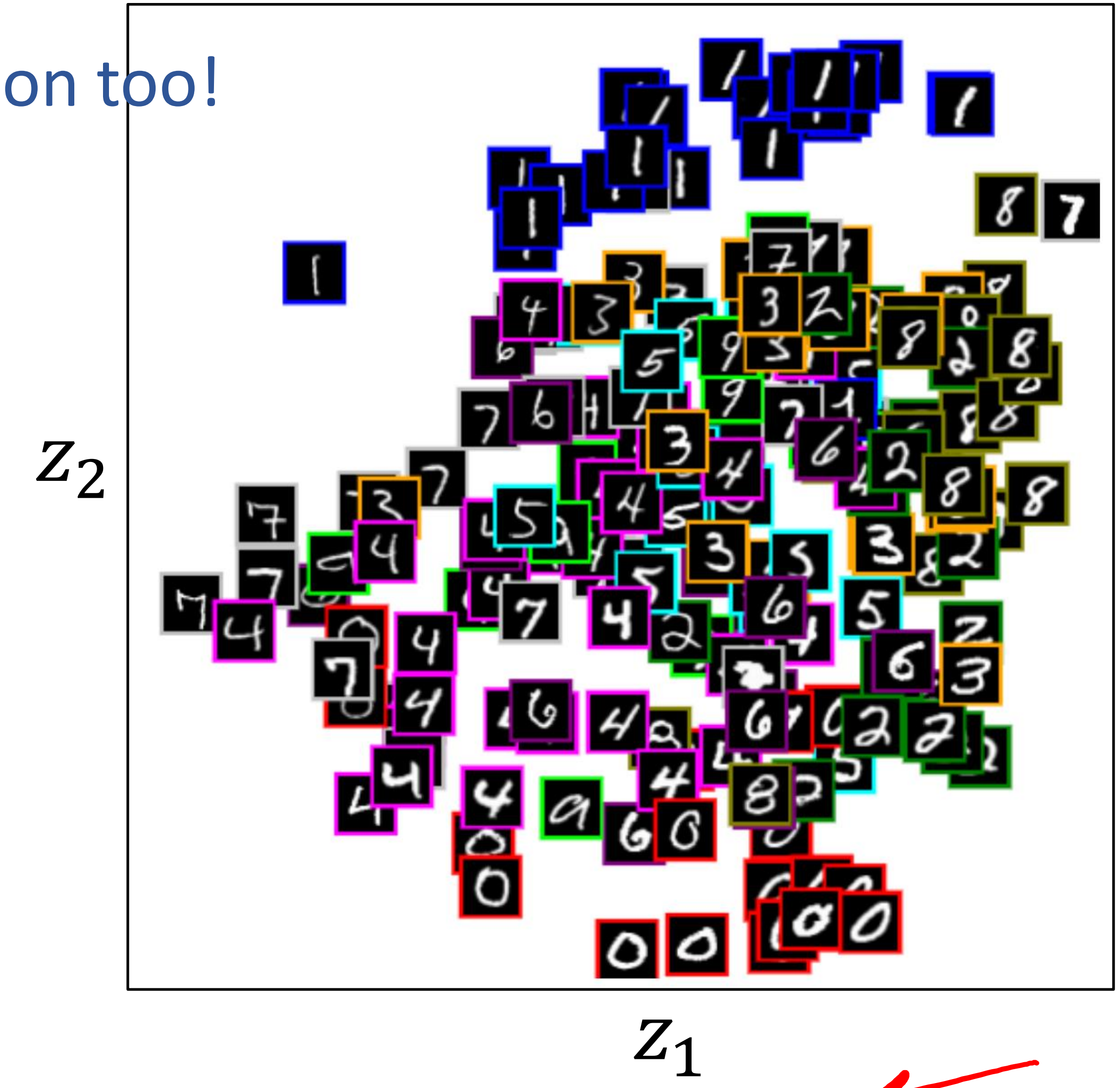


Learning to Organize Data

Neural networks can learn to organization too!

Image → $\begin{bmatrix} z_1 \\ z_2 \end{bmatrix}$ → Image



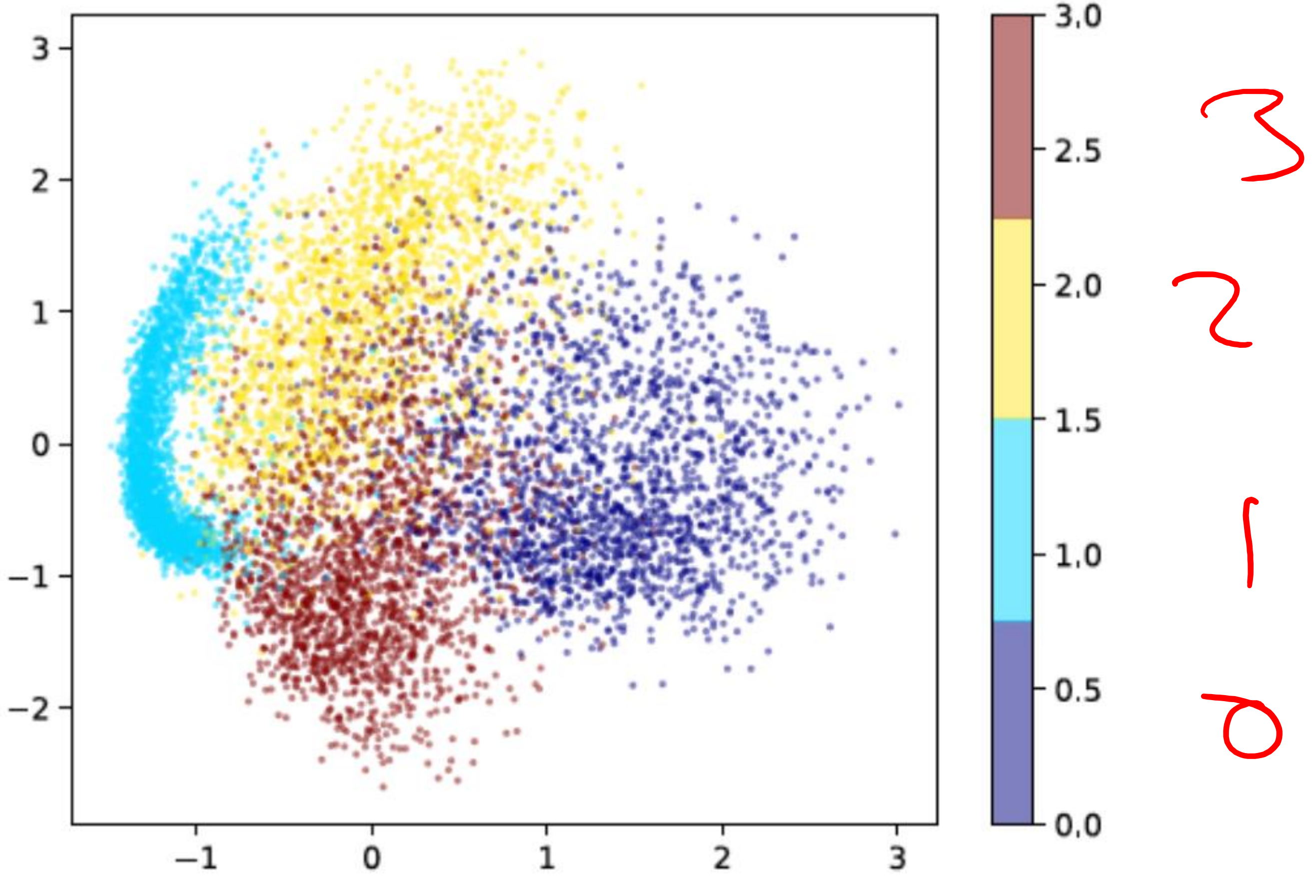


https://cs.stanford.edu/people/karpathy/convnetjs/demo/autoencoder.html

Projecting MNIST digits

Task Setting:

1. Take 28x28 images of digits and project them down to 2 components
2. Plot the 2 dimensional points



Dimensionality Reduction

http://timbaumann.info/svd-image-compression-demo/

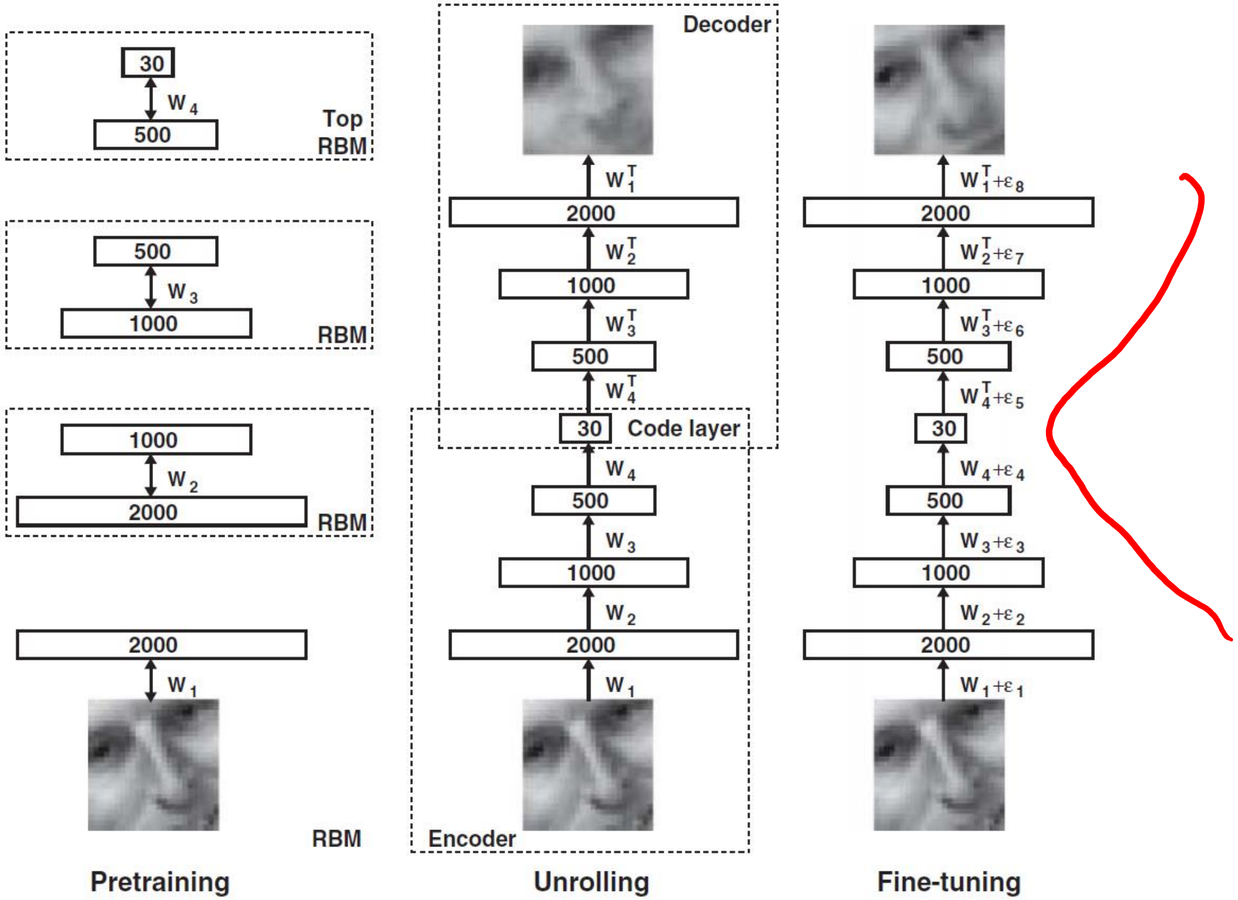
https://cs.stanford.edu/people/karpathy/convnetjs/demo/autoencoder.html

Dimensionality Reduction with Deep Learning

Hinton, Geoffrey E., and Ruslan R. Salakhutdinov.

"Reducing the dimensionality of data with neural networks."

Science 313.5786 (2006): 504-507.



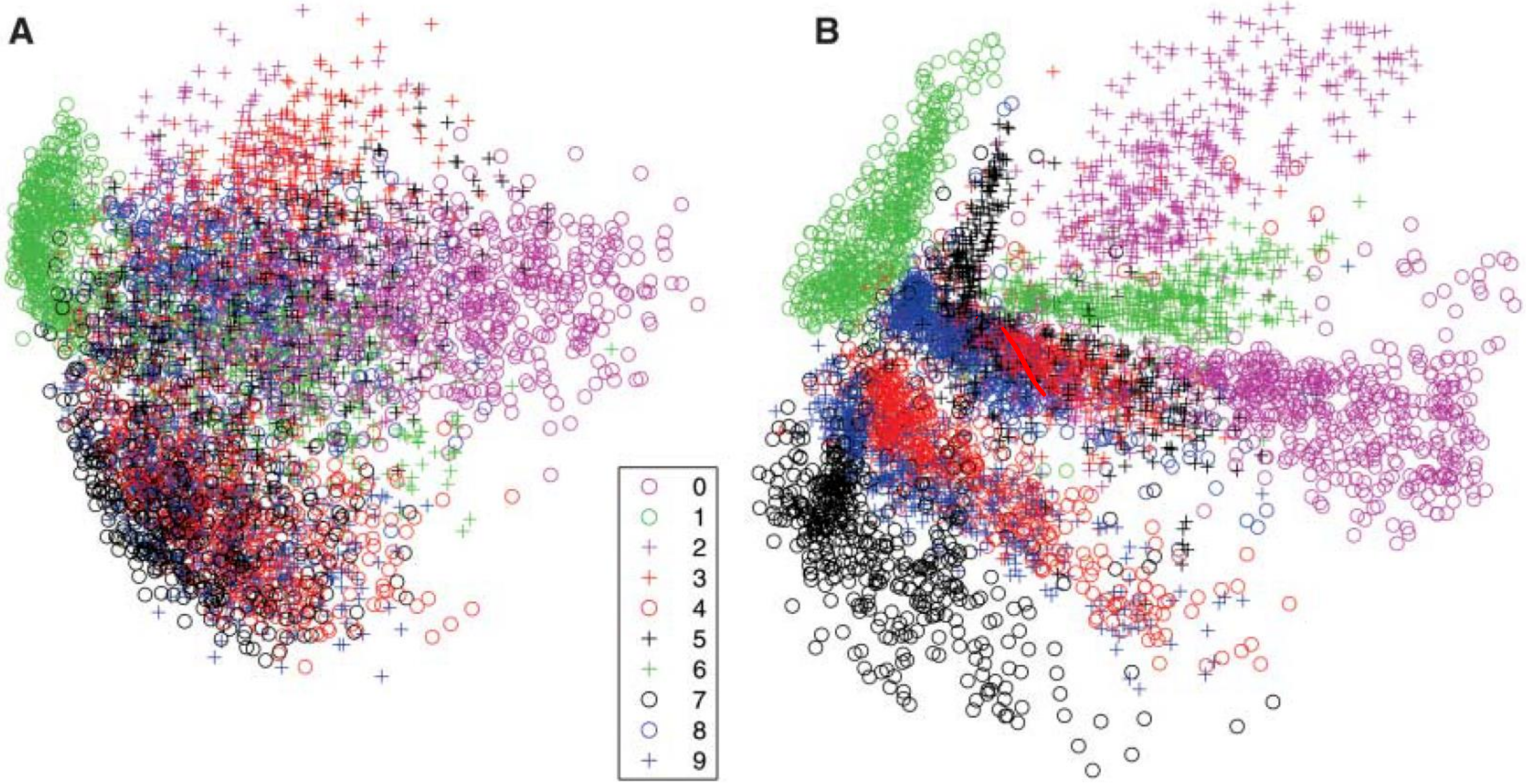
Dimensionality Reduction with Deep Learning

Hinton, Geoffrey E., and Ruslan R. Salakhutdinov.

"Reducing the dimensionality of data with neural networks."

Science 313.5786 (2006): 504-507.

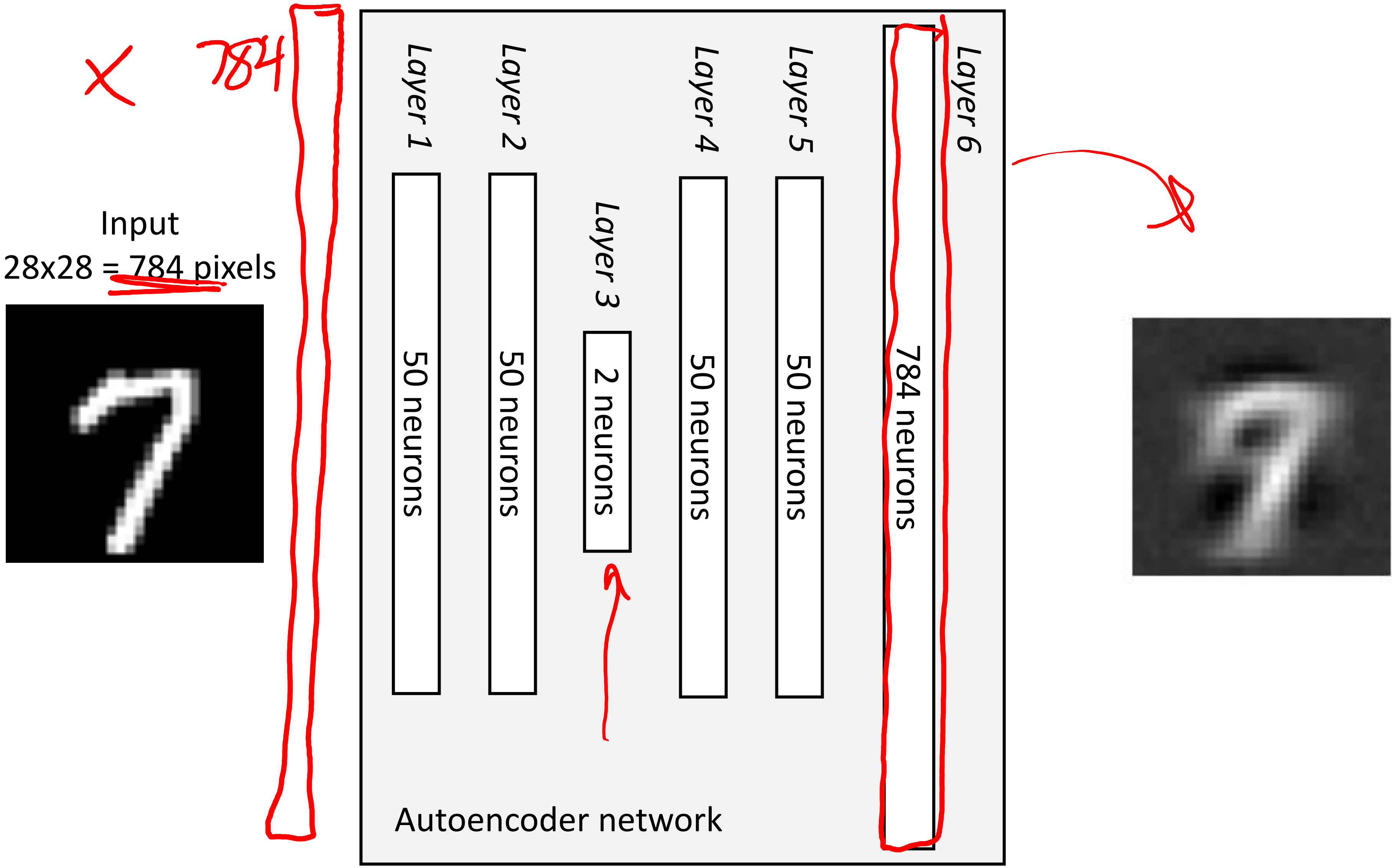
PCA



Neural Network

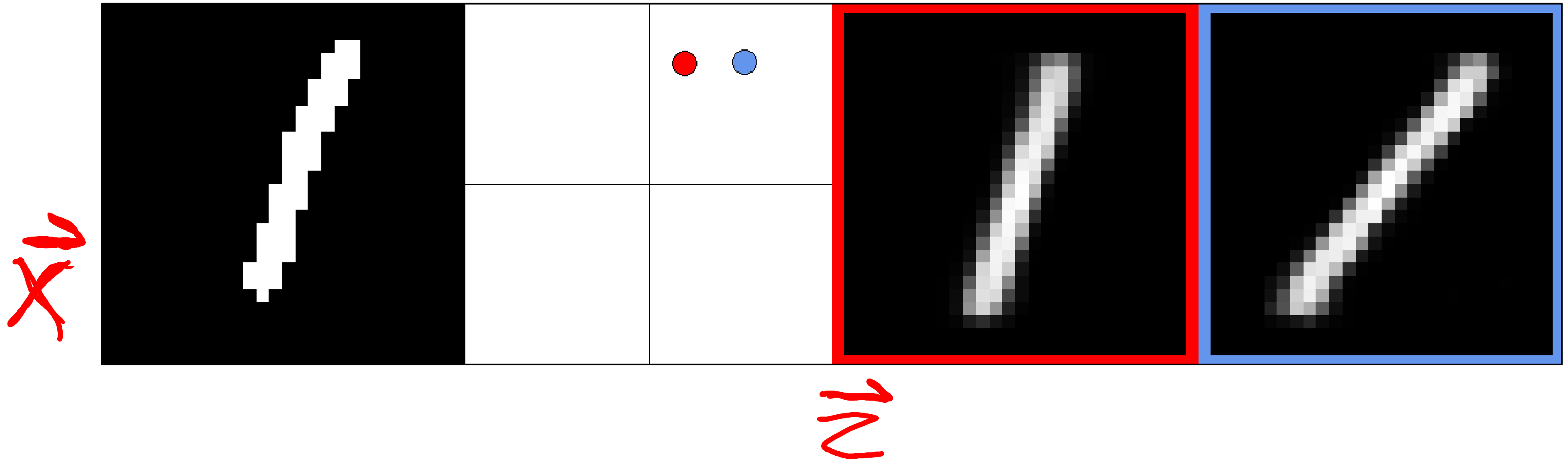
Digit Autoencoder

https://cs.stanford.edu/people/karpathy/convnetjs/demo/autoencoder.html



Digit Autoencoder

Demo: Using a learned feature space

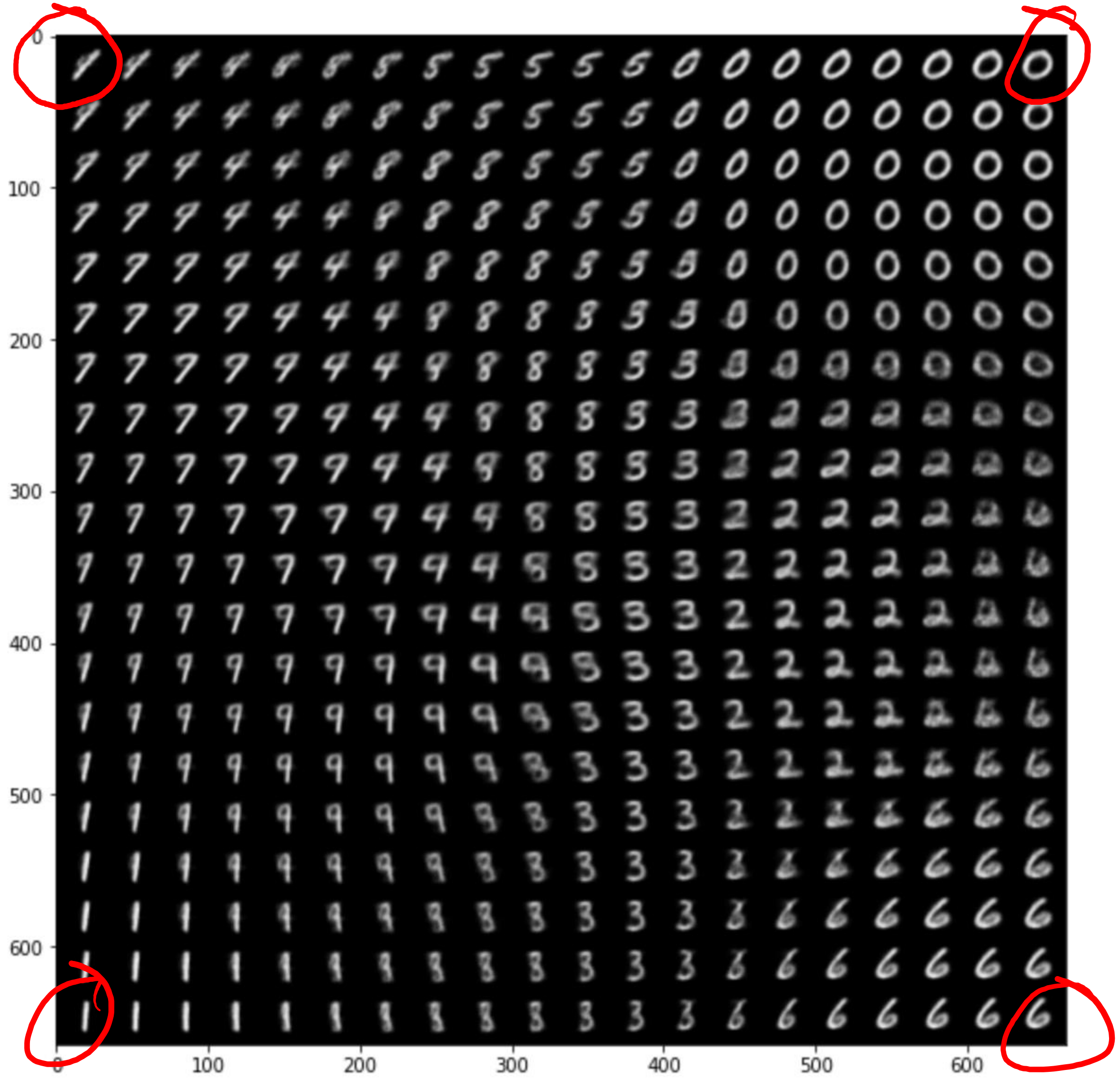


$$\text{encoder}(\vec{x}) \rightarrow \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} \rightarrow \text{decode}(\vec{z}) \rightarrow \hat{\vec{x}}$$

Autoencoder Demo

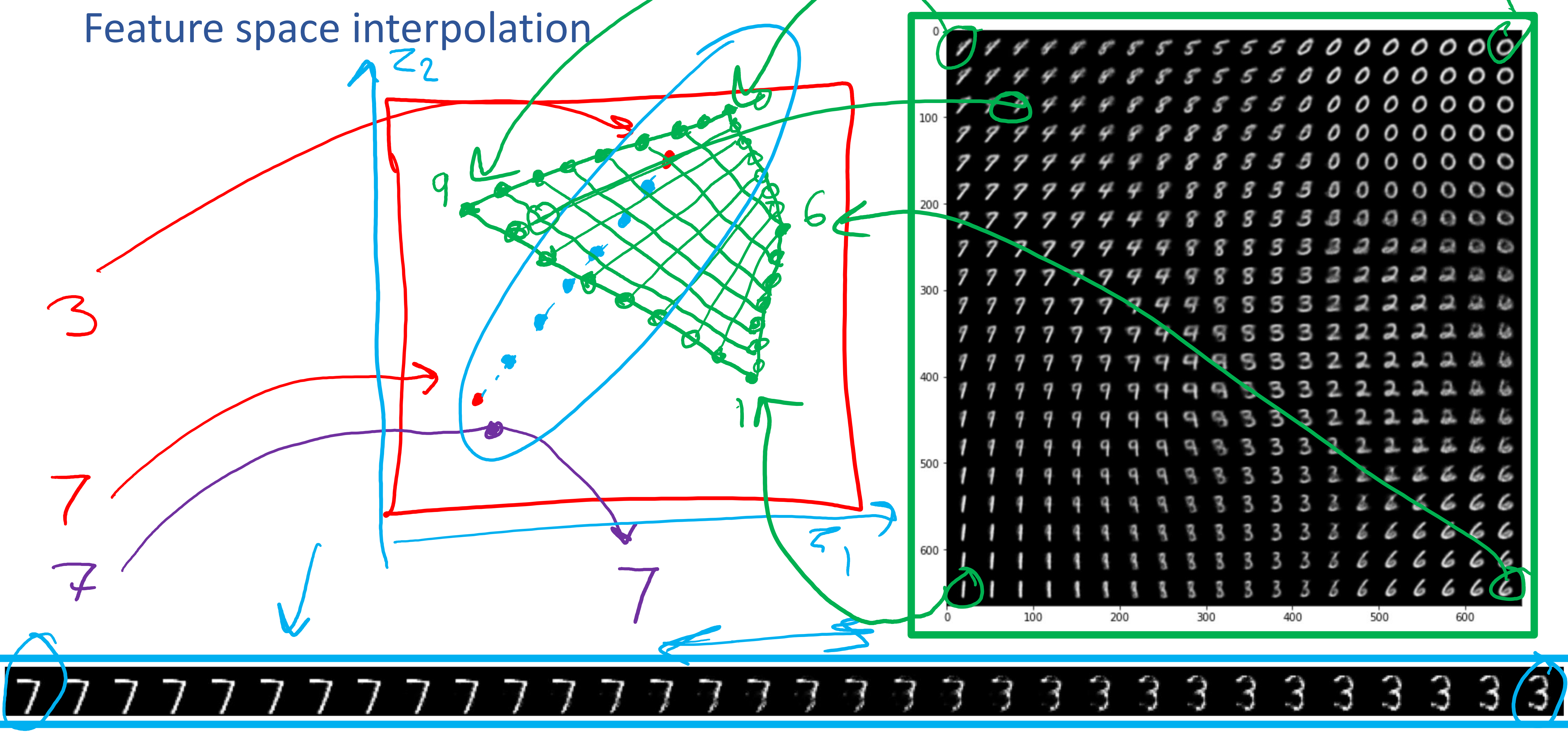
Zhuoyue Lyu, Safinah Ali, and Cynthia Breazeal. EAAI 2022.

https://colab.research.google.com/gist/ZhuoyueLyu/5046225a9ae3675cf633c1df5f63be06/digits-interpolation-notebook-eaai.ipynb



Autoencoder Demo

Feature space interpolation



Feature Learning

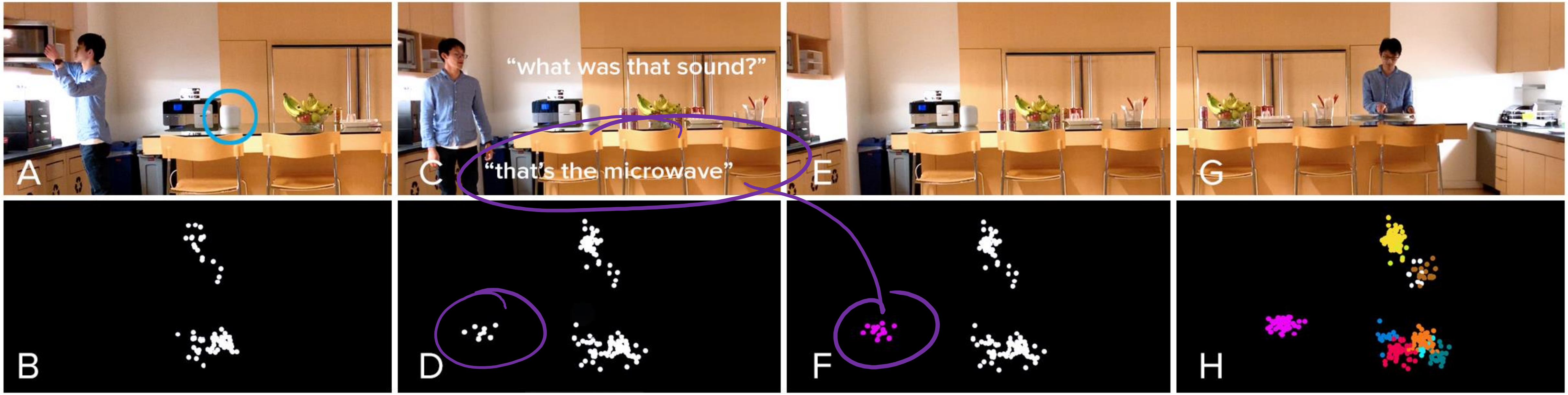
Learning a lower dimensional representation of our data rather than doing feature engineering to represent the data

Also called feature embedding

(embedding data in lower dimensional space)

Feature Learning

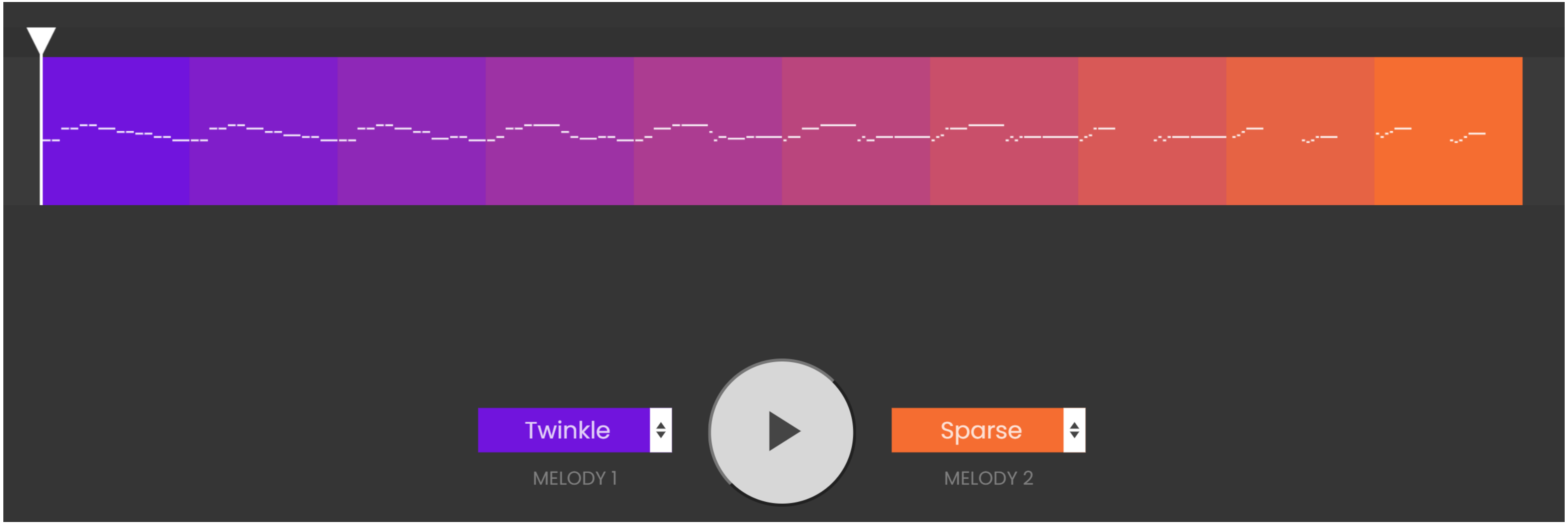
Listen Learner



https://chrisharrison.net/index.php/Research/ListenLearner

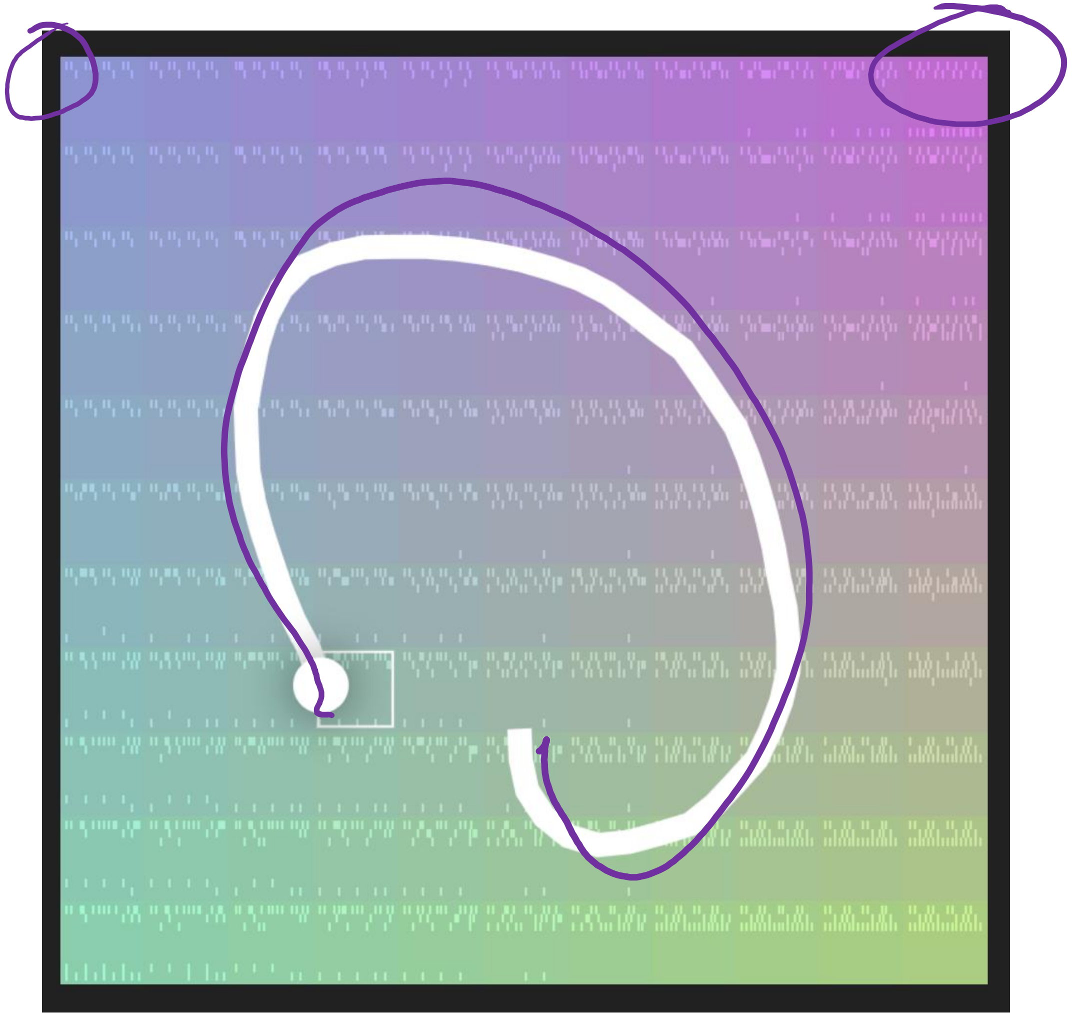
Exploring Feature Space

https://experiments.withgoogle.com/ai/melody-mixer/view/



Exploring Feature Space

https://experiments.withgoogle.com/ai/beat-blender/view/



Feature Learning

Word embedding with word2vec

Training data:

"The king sat on the throne"

"the queen sat on the throne"

"the banana is yellow"

"they sat on the yellow bus"

- king
- sat
- throne
- queen
- banana
- yellow
- they
- bus

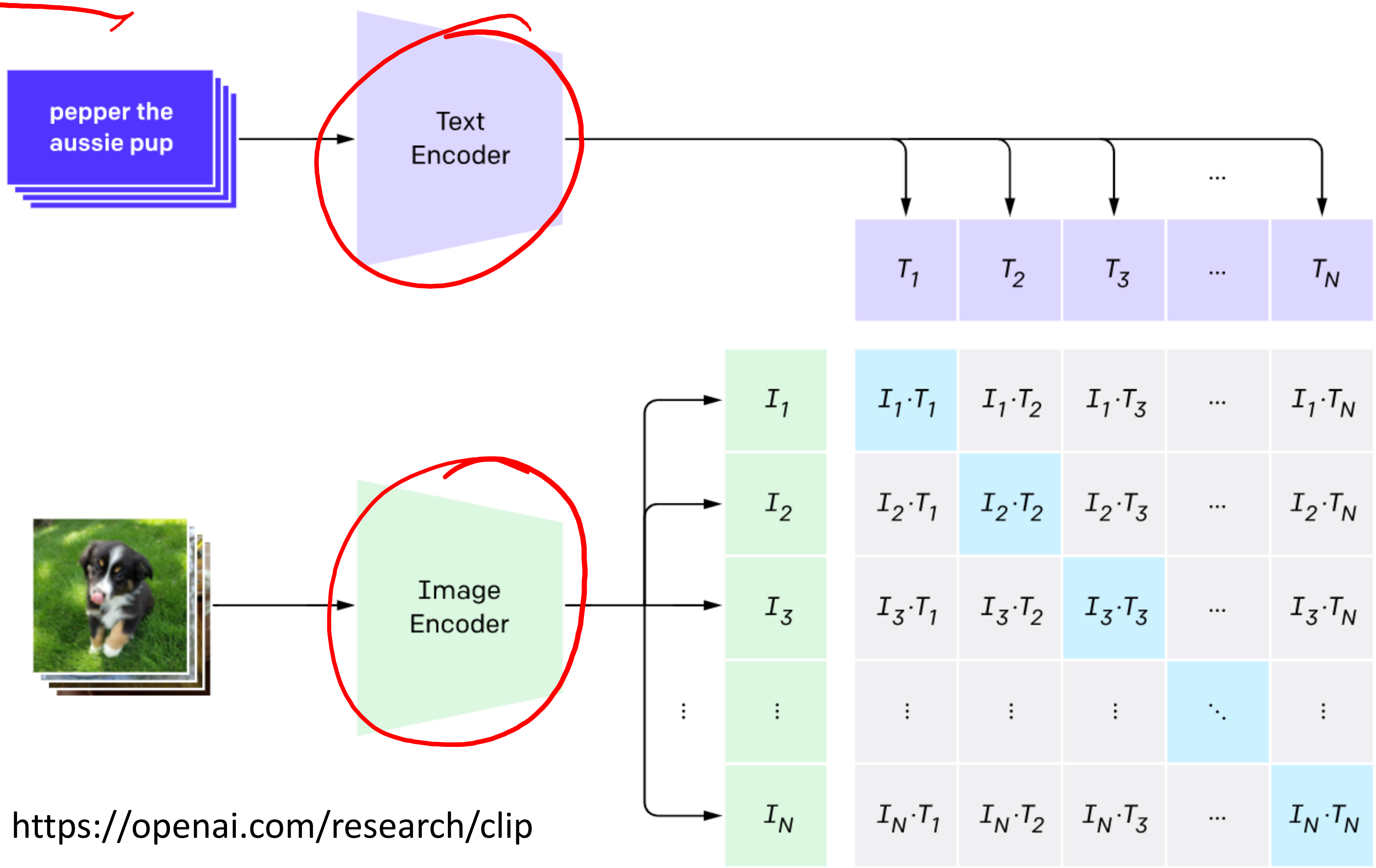
- king
- sat
- throne
- queen
- banana
- yellow
- they
- bus

Skip-gram

score(word, <other words around it>)

Feature Learning

CLIP: Connecting text and images

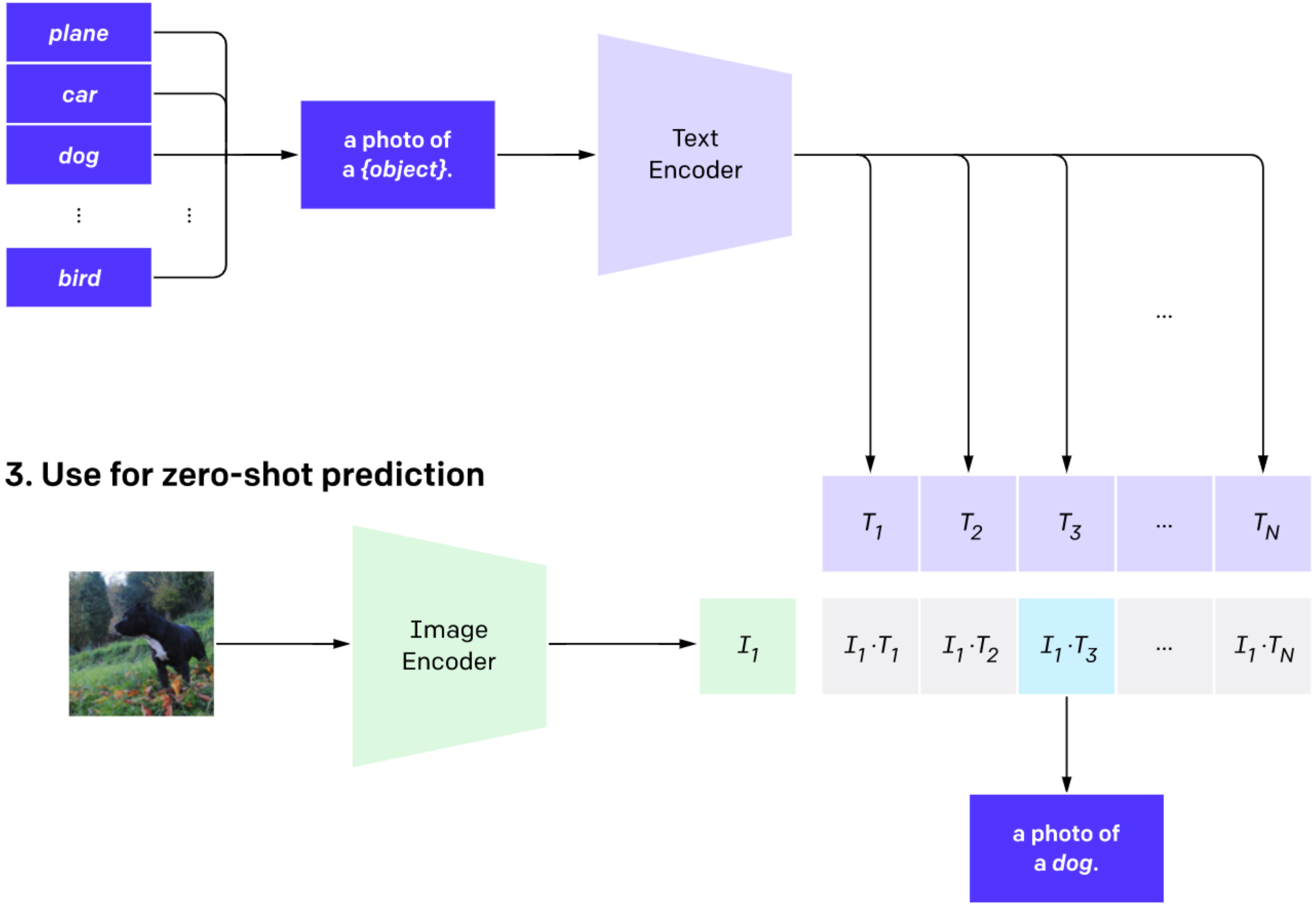


https://openai.com/research/clip

k-dim feat. space.

Feature Learning

CLIP: Connecting text and images



https://openai.com/research/clip

Principal Component Analysis (PCA)

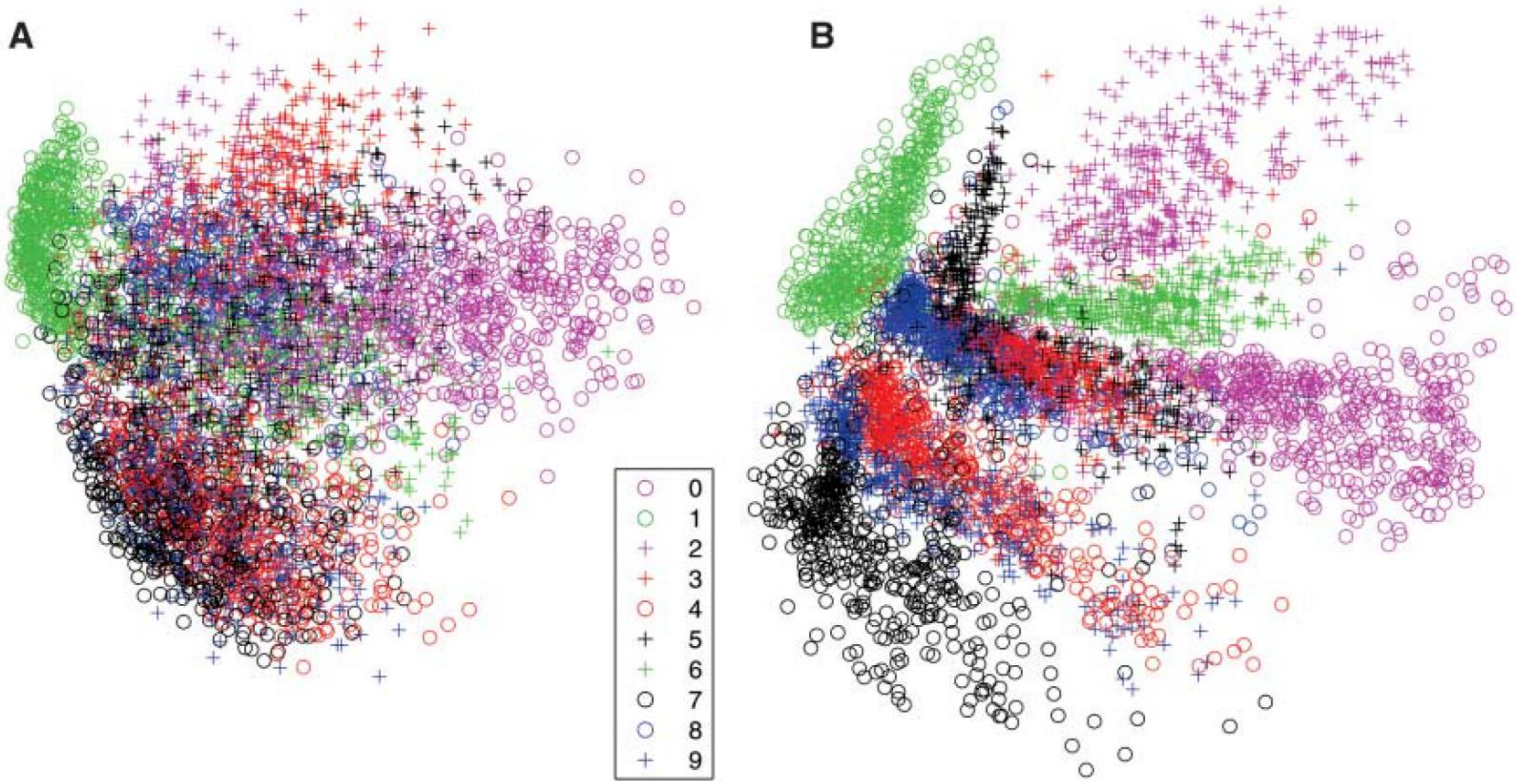
Dimensionality Reduction with Deep Learning

Hinton, Geoffrey E., and Ruslan R. Salakhutdinov.

"Reducing the dimensionality of data with neural networks."

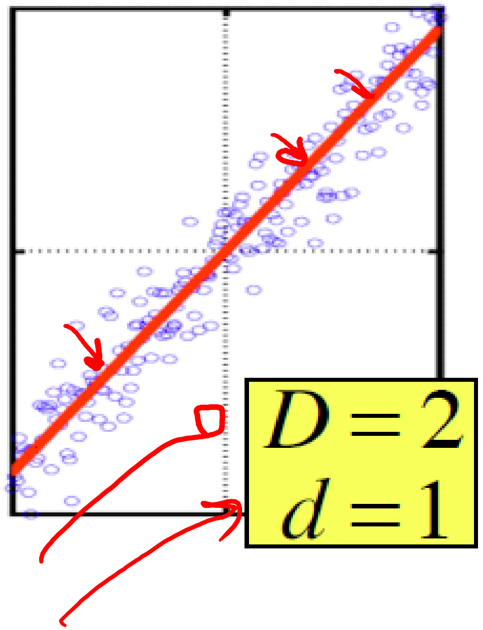
Science 313.5786 (2006): 504-507.

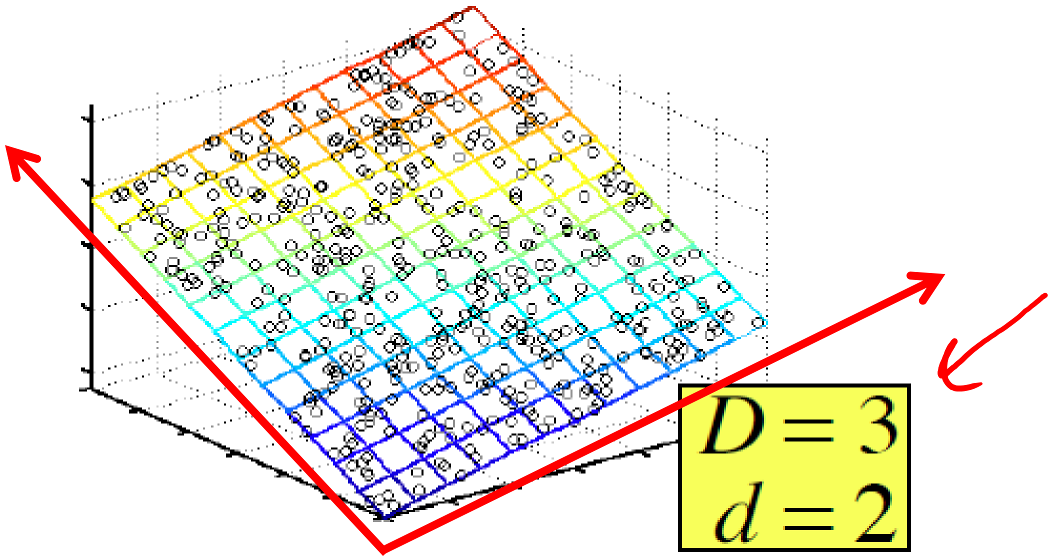
PCA



Neural Network

Principal Component Analysis (PCA)



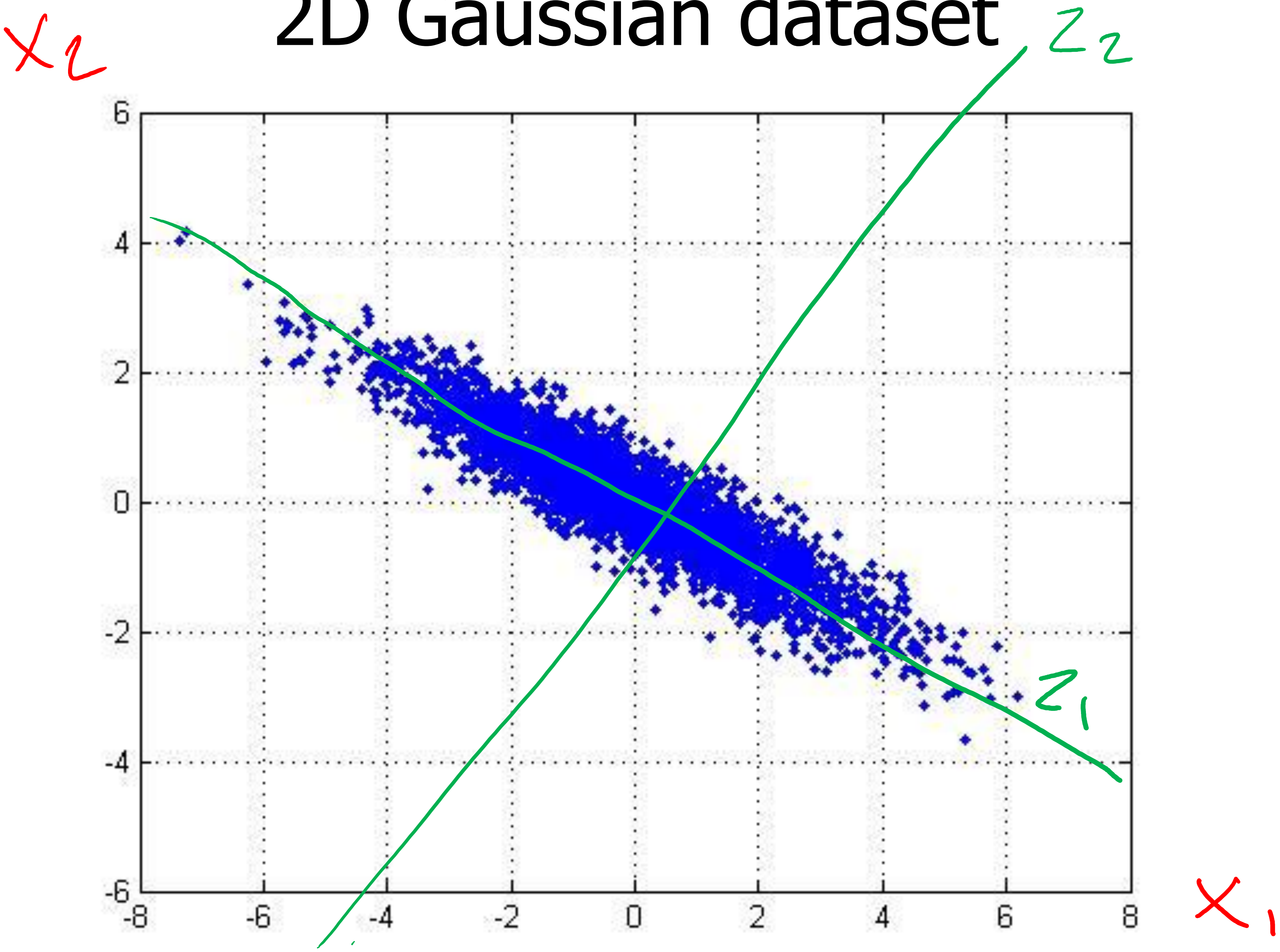


In case where data lies on or near a low d-dimensional linear subspace, axes of this subspace are an effective representation of the data.

Identifying the axes is known as Principal Components Analysis, and can be obtained by using classic matrix computation tools (Eigen or Singular Value Decomposition).

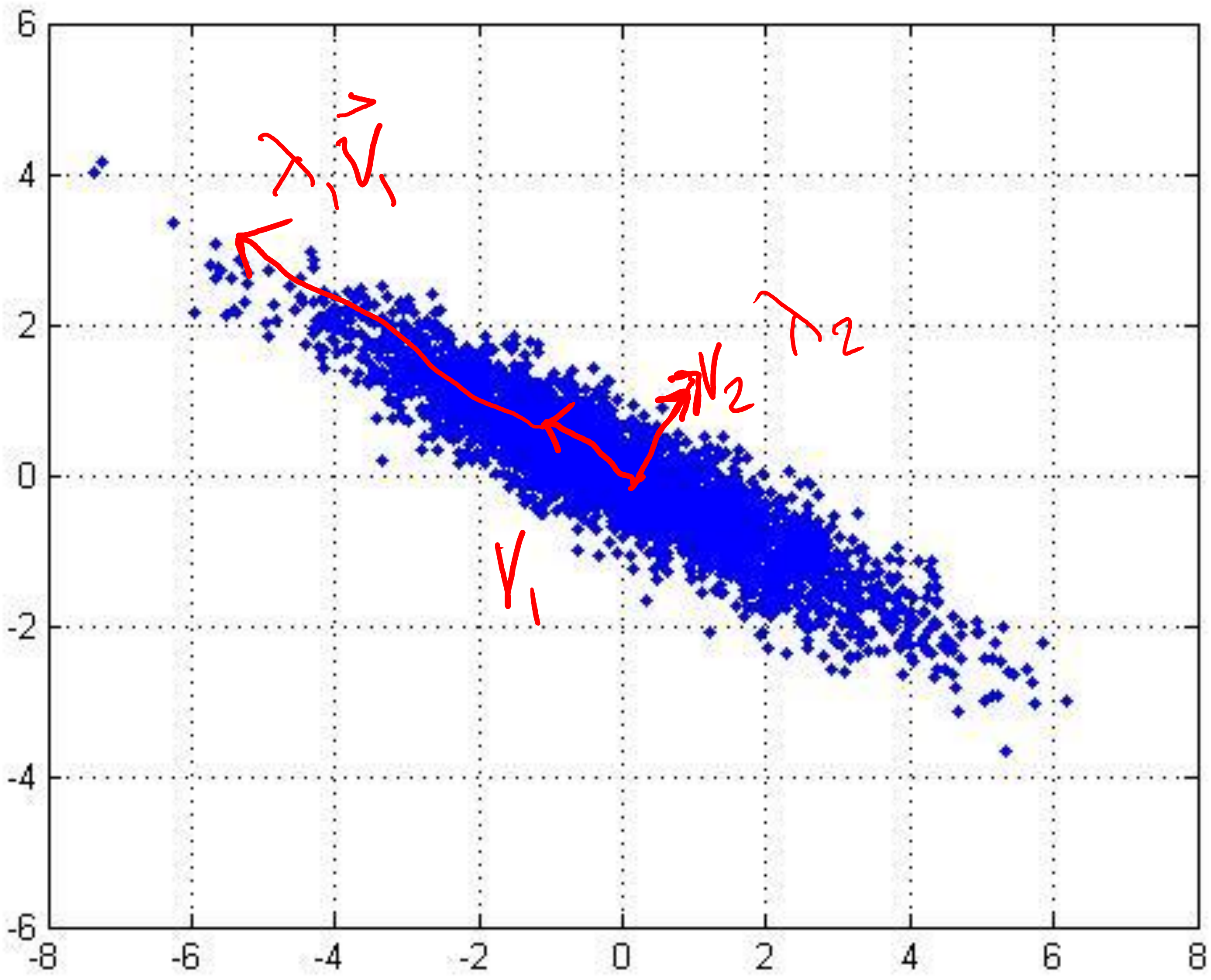
Slide from Nina Balcan

2D Gaussian dataset



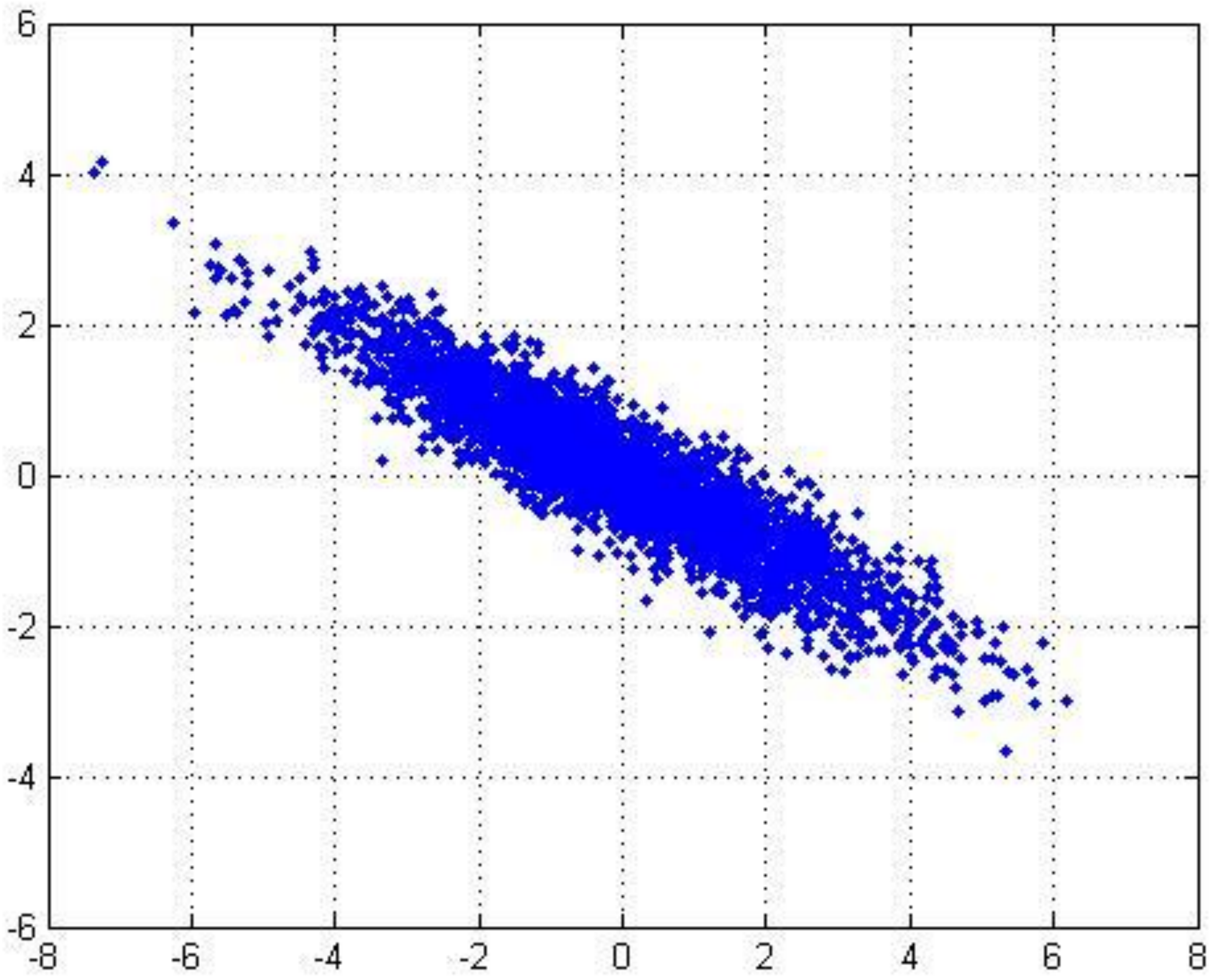
Slide from Barnabas Poczos

1st PCA axis



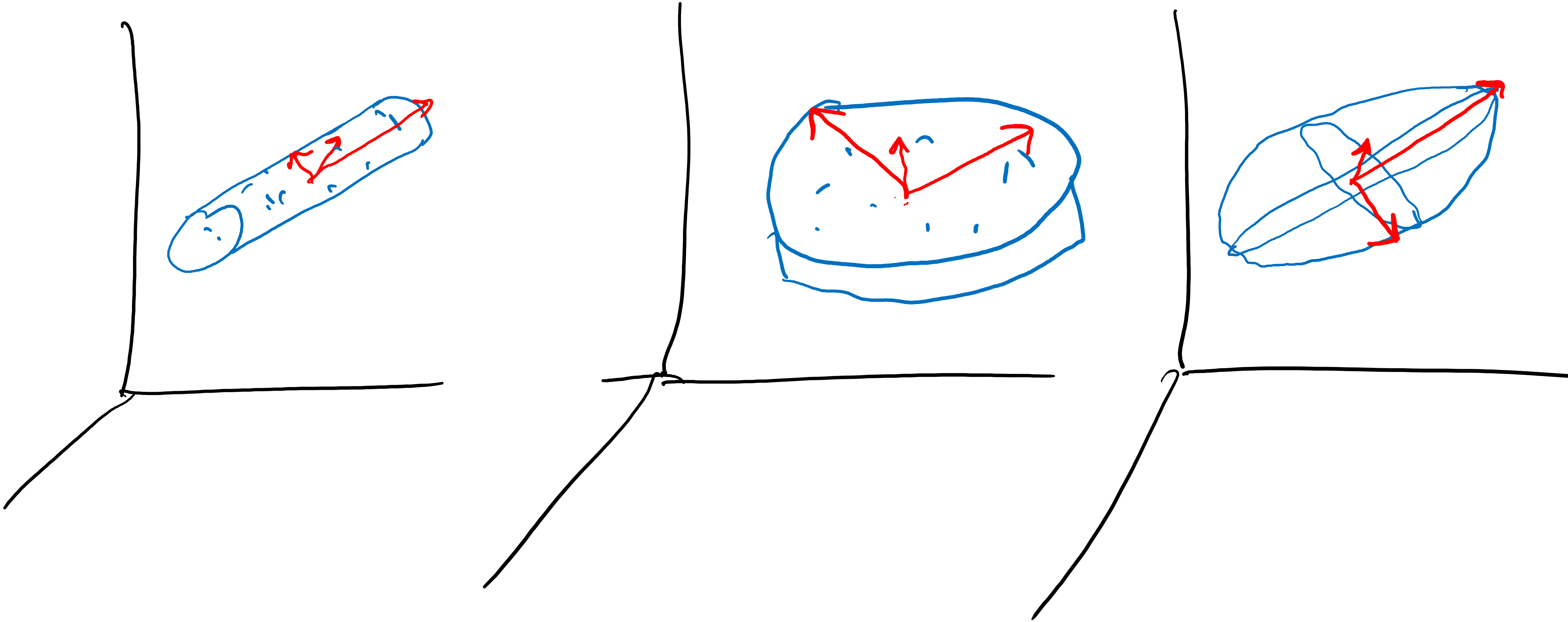
Slide from Barnabas Poczos

2nd PCA axis

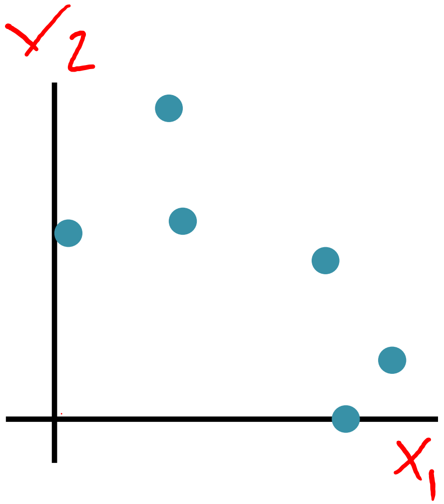


Slide from Barnabas Poczos

PCA Axes



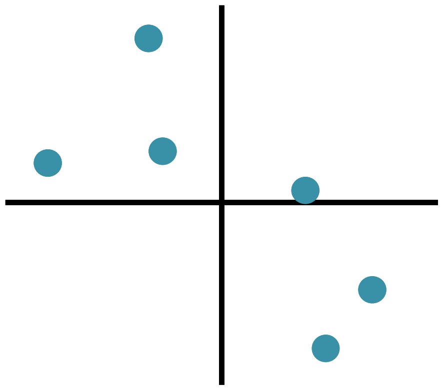
Data for PCA



$$\mathcal{D} = \{\mathbf{x}^{(i)}\}_{i=1}^{N}$$

$$\mathbf{X} = \begin{bmatrix} (\mathbf{x}^{(1)})^T \\ (\mathbf{x}^{(2)})^T \\ \vdots \\ (\mathbf{x}^{(N)})^T \end{bmatrix}$$

x_1 x_2
x_1 x_2
x_1 x_2



We assume the data is **centered**

$$\mu = \frac{1}{N} \sum_{i=1}^{N} \mathbf{x}^{(i)} = \mathbf{0}$$

Q: What if your data is **not** centered?

A: Subtract off the sample mean

Slide from Matt Gormley

Sample Covariance Matrix

N point

M dim

The sample covariance matrix is given by:

$$\Sigma_{jk} = \frac{1}{N} \sum_{i=1}^{N} (x_j^{(i)} - \mu_j)(x_k^{(i)} - \mu_k)$$

Since the data matrix is centered, we rewrite as:

$$\mathbf{\Sigma} = \frac{1}{N} \mathbf{X}^T \mathbf{X}$$

M x M

$$\mathbf{X} = \begin{bmatrix} (\mathbf{x}^{(1)})^T \\ (\mathbf{x}^{(2)})^T \\ \vdots \\ (\mathbf{x}^{(N)})^T \end{bmatrix}$$

Slide from Matt Gormley

PCA Algorithm

Input: $\boldsymbol{X}, \boldsymbol{X}_{test}, K$

1. Center data (and scale each axis) based on training data → $\boldsymbol{X}, \boldsymbol{X}_{test}$
2. $\boldsymbol{V}$ = eigenvectors($\boldsymbol{X}^T\boldsymbol{X}$)
3. Keep only the top K eigenvectors: $\boldsymbol{V}_K$
4. $\mathbf{Z}_{\text{test}} = \boldsymbol{X}_{test}\boldsymbol{V}_K$

Optionally, use $\boldsymbol{V}_K^T$ to rotate $\mathbf{Z}_{\text{test}}$ back to original subspace $\mathbf{X}'_{\text{test}}$ and uncenter

train

$N \times M$

of M

$K \times M$

$$X' = (X_{test} V_K) V_K^T$$

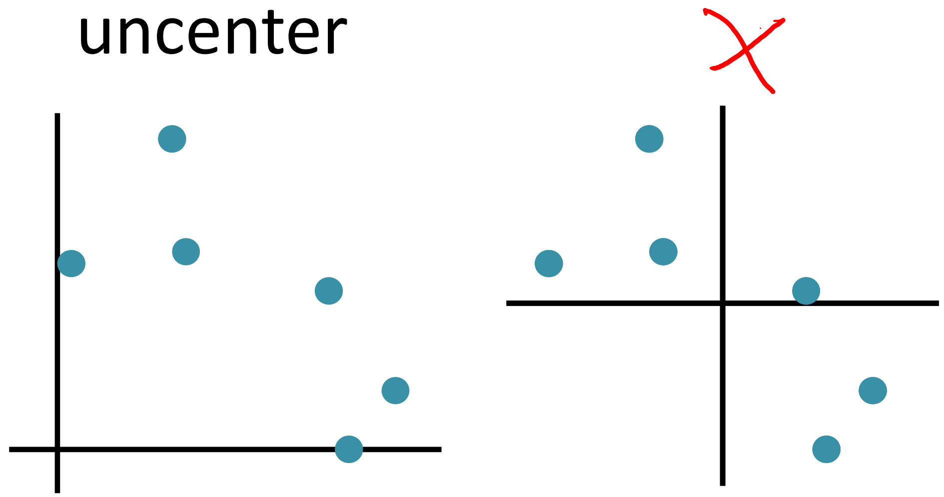
$N \times M$ $N \times M$ $M \times K$ $K \times M$

PCA Algorithm

Input: $\boldsymbol{X}, \boldsymbol{X}_{test}, K$

1. Center data (and scale each axis) based on training data → $\boldsymbol{X}, \boldsymbol{X}_{\boldsymbol{test}}$
2. $\boldsymbol{V}$ = eigenvectors($\boldsymbol{X}^T\boldsymbol{X}$)
3. Keep only the top K eigenvectors: $\boldsymbol{V}_K$
4. $\mathbf{Z}_{\text{test}} = \boldsymbol{X}_{test}\boldsymbol{V}_K$

Optionally, use $\boldsymbol{V}_K^T$ to rotate $\mathbf{Z}_{\text{test}}$ back to original subspace $\mathbf{X}'_{\text{test}}$ and uncenter

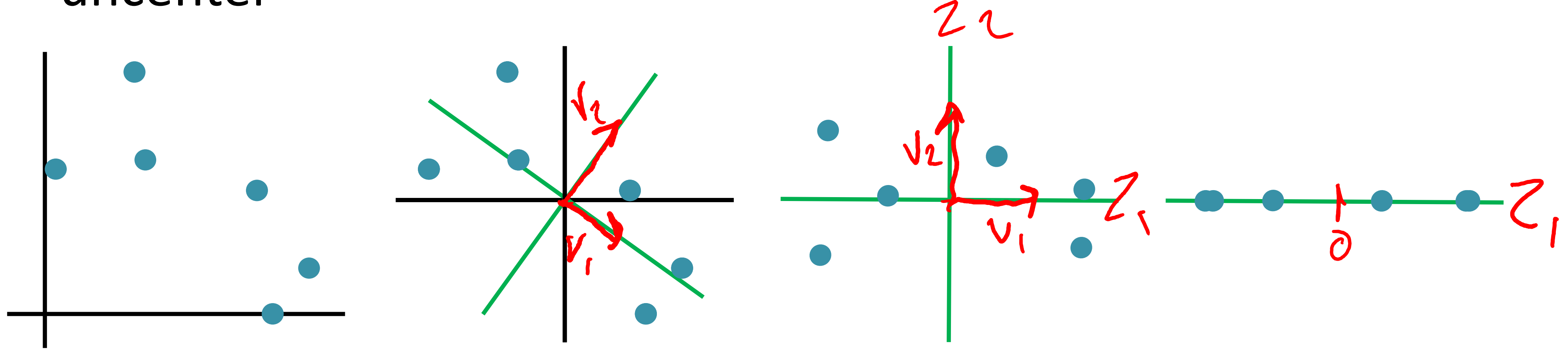


PCA Algorithm

Input: $\boldsymbol{X}, \boldsymbol{X}_{test}, K$

1. Center data (and scale each axis) based on training data → $\boldsymbol{X}, \boldsymbol{X}_{\boldsymbol{test}}$
2. $\boldsymbol{V}$ = eigenvectors($\boldsymbol{X}^T\boldsymbol{X}$)
3. Keep only the top K eigenvectors: $\boldsymbol{V}_K$
4. $\mathbf{Z}_{\text{test}} = \boldsymbol{X}_{test}\boldsymbol{V}_K$

Optionally, use $\boldsymbol{V}_K^T$ to rotate $\mathbf{Z}_{\text{test}}$ back to original subspace $\mathbf{X}'_{\text{test}}$ and uncenter

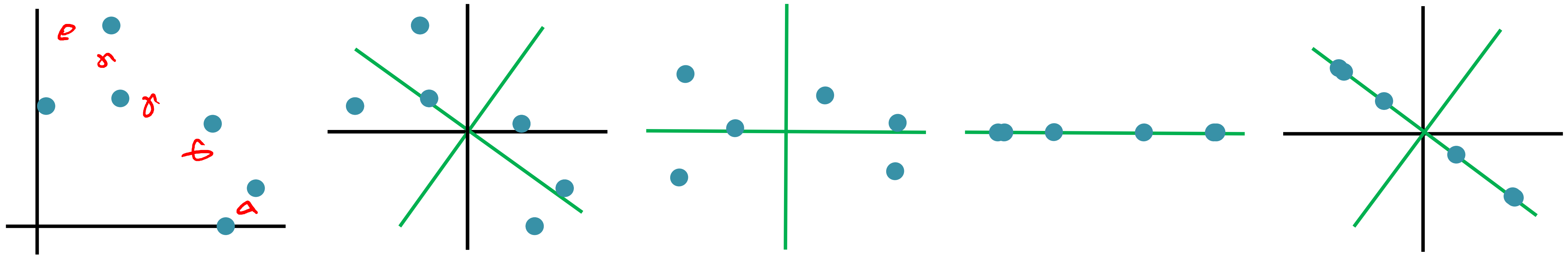


PCA Algorithm

Input: $\boldsymbol{X}, \boldsymbol{X}_{test}, K$

1. Center data (and scale each axis) based on training data → $\boldsymbol{X}, \boldsymbol{X}_{test}$
2. $\boldsymbol{V}$ = eigenvectors($\boldsymbol{X}^T\boldsymbol{X}$)
3. Keep only the top K eigenvectors: $\boldsymbol{V}_K$
4. $\mathbf{Z}_{\text{test}} = \boldsymbol{X}_{test}\boldsymbol{V}_K$

Optionally, use $\boldsymbol{V}_K^T$ to rotate $\mathbf{Z}_{\text{test}}$ back to original subspace $\mathbf{X}'_{\text{test}}$ and uncenter



PCA EXAMPLES

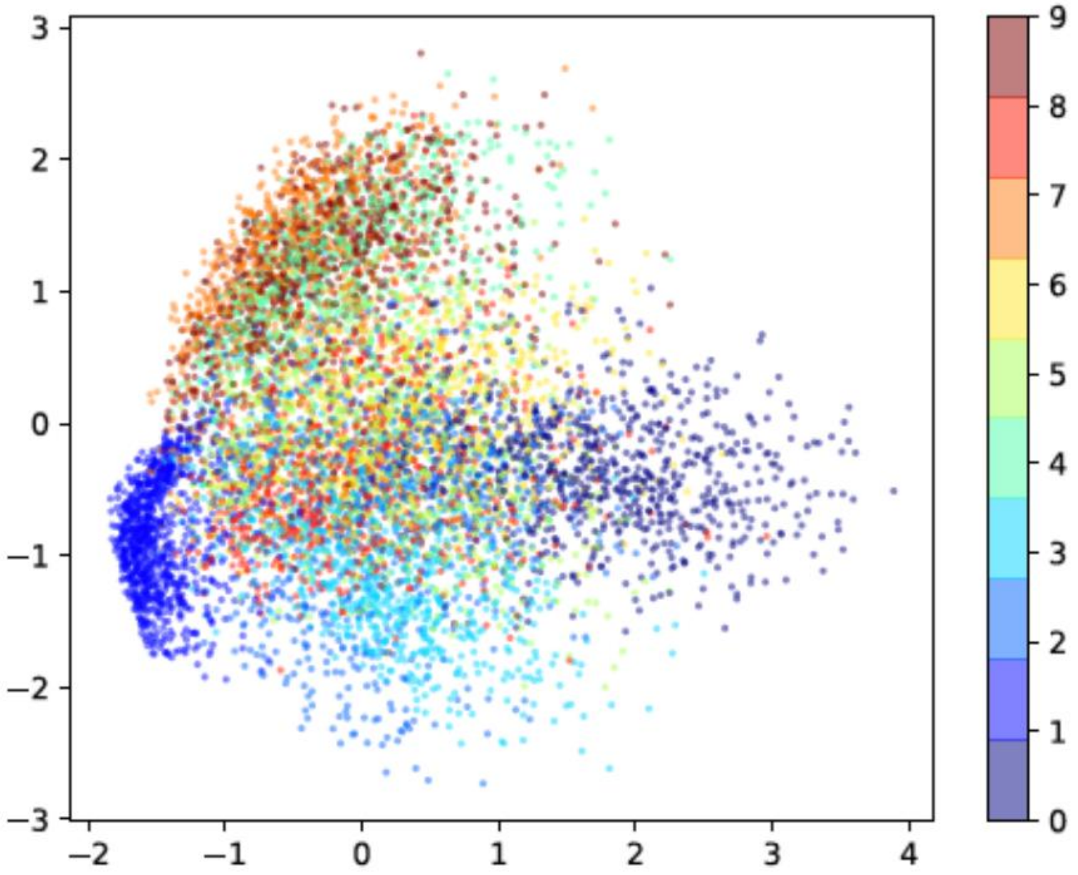
Projecting MNIST digits

Task Setting:

1. Take 28x28 images of digits and project them down to K components
2. Report percent of variance explained for K components
3. Then project back up to 28x28 image to visualize how much information was preserved

Projecting MNIST digits

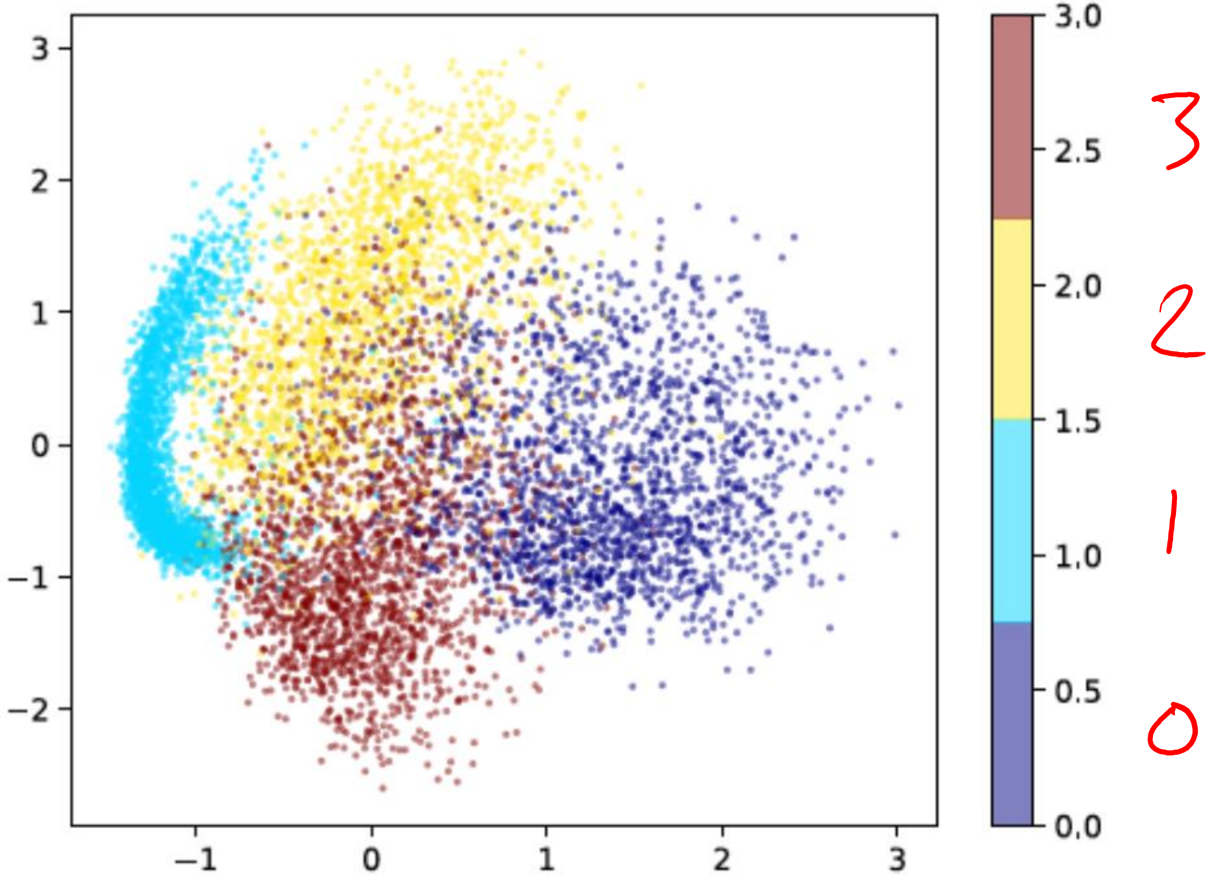
Task Setting:
1. Take 28x28 images of digits and project them down to 2 components
2. Plot the 2 dimensional points



Projecting MNIST digits

Task Setting:

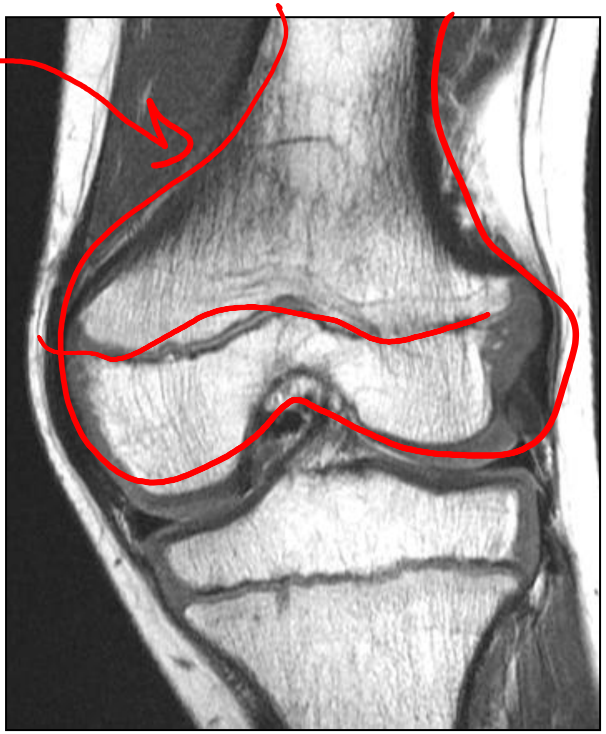
1. Take 28x28 images of digits and project them down to 2 components
2. Plot the 2 dimensional points



Growth Plate Imaging

Growth Plate Disruption and Limb Length Discrepancy

8 year-old boy with previous fracture and
4cm leg length discrepancy



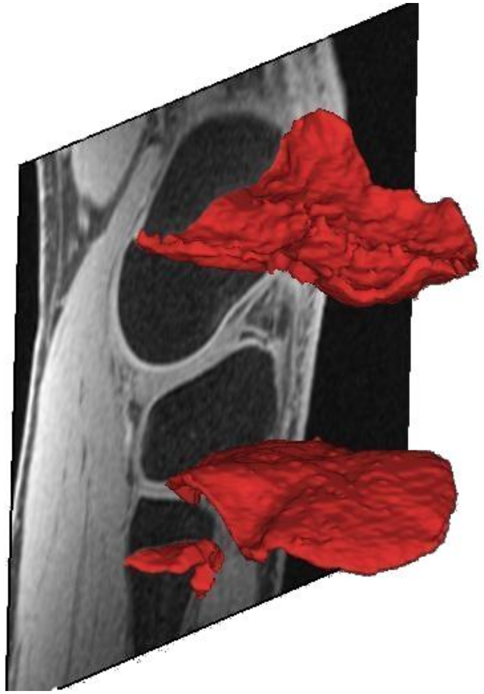
Images Courtesy
H. Potter, H.S.S.

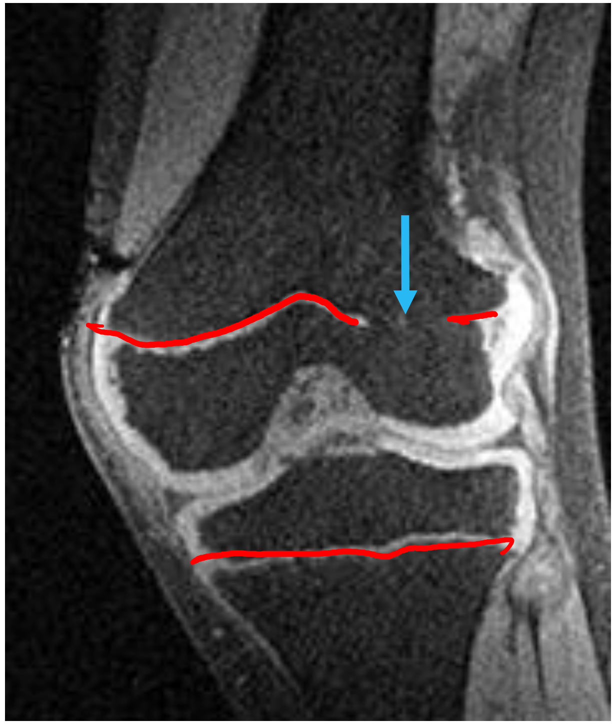
imagination at work

Growth Plate Imaging

Growth Plate Disruption and Limb Length Discrepancy

8 year-old boy with previous fracture and
4cm leg length discrepancy



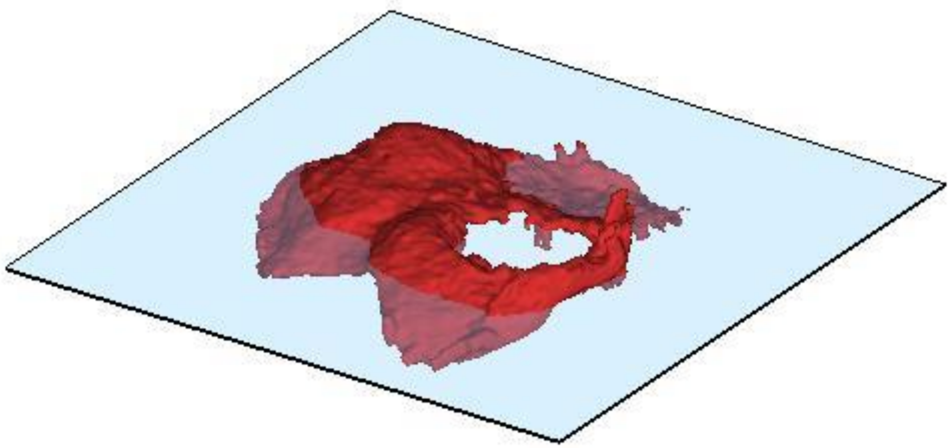


Images Courtesy
H. Potter, H.S.S.

imagination at work

Growth Plate Imaging

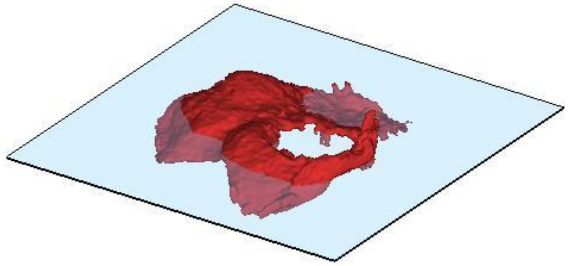
Area Measurement

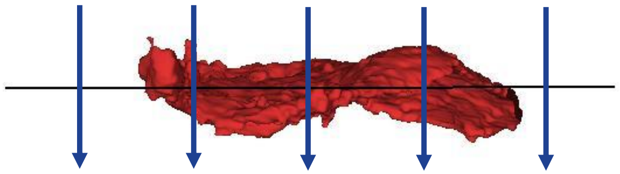


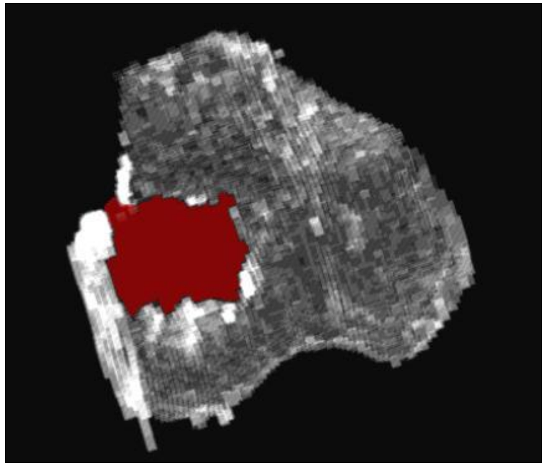
imagination at work

Growth Plate Imaging

Area Measurement







Flatten Growth Plate to Enable 2D Area Measurement

Outline

Dimensionality Reduction

- High-dimensional data
- Low dimensional representations

$$\|x - x'\|_2^2$$

Autoencoders

Feature Learning

Principal Component Analysis (PCA)

- Examples: 2D and 3D
- PCA algorithm
- PCA, eigenvectors, and eigenvalues
- PCA objective and optimization

Poll 1

What is the projection of point $\boldsymbol{x}$ onto vector $\boldsymbol{v}$, assuming that $\|\boldsymbol{v}\|_2 = 1$?

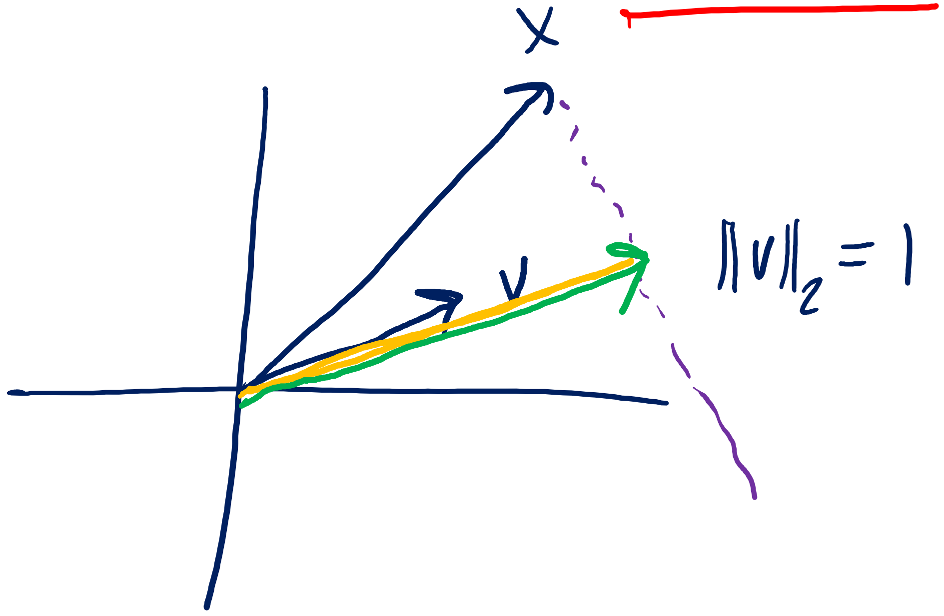
A. $\mathbf{v}\mathbf{x}$

B.

C. $\mathbf{v}^T\mathbf{x}$

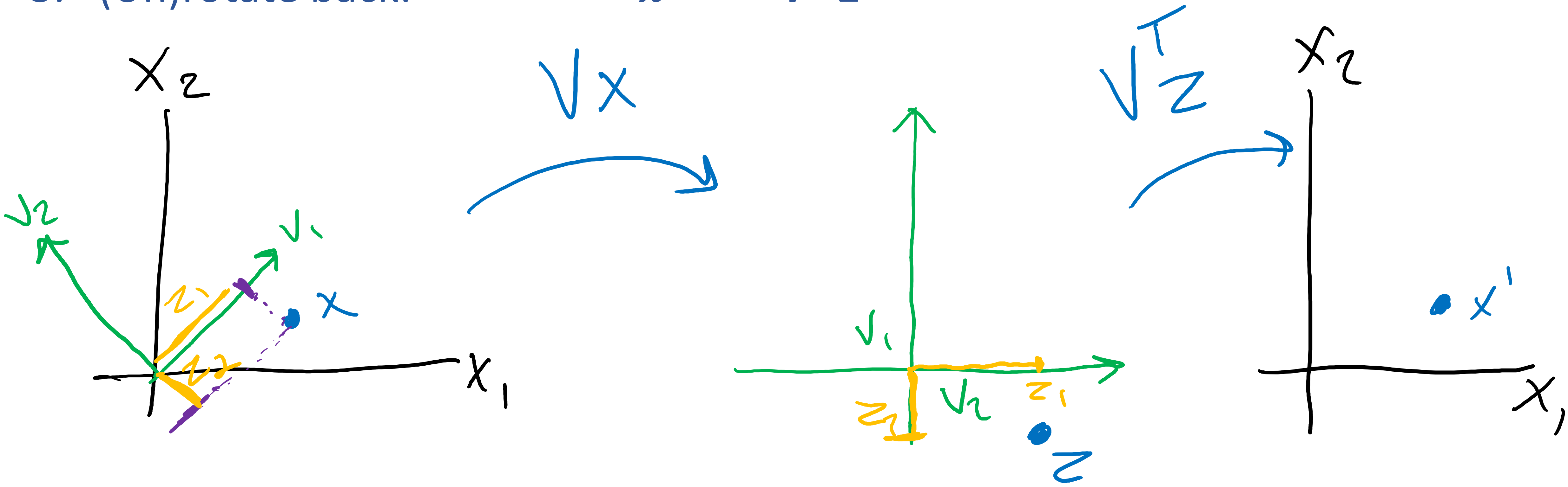
D. $(\mathbf{v}^T\mathbf{x})\mathbf{v}$

E. $\mathbf{v}^T\mathbf{x}\,\mathbf{x}^T\mathbf{v}$



Rotation of Data (and back)

1. For any orthogonal matrix $\boldsymbol{V} \in \mathbb{R}^{M \times M}$
2. Rotate to new space: $\boldsymbol{z}^{(i)} = \boldsymbol{V}\boldsymbol{x}^{(i)} \quad \forall i$
3. (Un)rotate back: $\boldsymbol{x}'^{(i)} = \boldsymbol{V}^T \boldsymbol{z}^{(i)}$

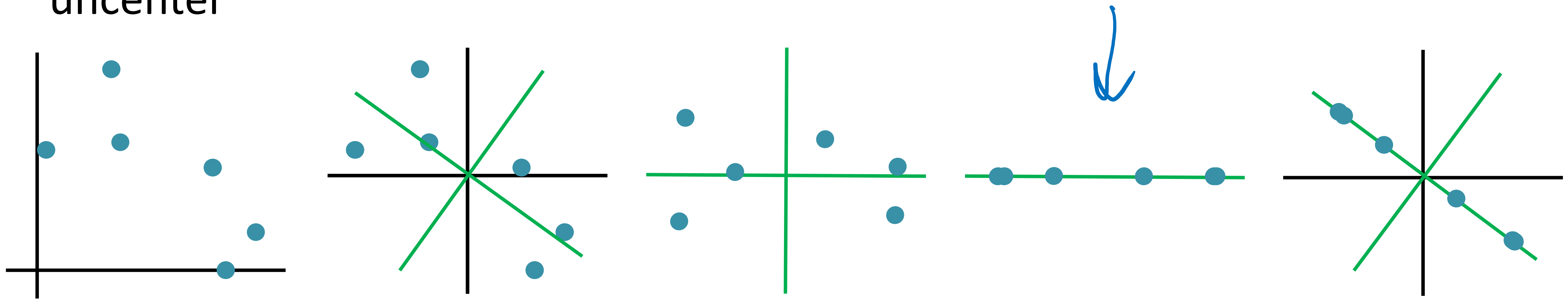


PCA Algorithm

Input: $\boldsymbol{X}, \boldsymbol{X}_{test}, K$

1. Center data (and scale each axis) based on training data → $\boldsymbol{X}, \boldsymbol{X}_{test}$
2. $\boldsymbol{V}$ = eigenvectors($\boldsymbol{X}^T\boldsymbol{X}$)
3. Keep only the top K eigenvectors: $\boldsymbol{V}_K$
4. $\mathbf{Z}_{\text{test}} = \boldsymbol{X}_{test}\boldsymbol{V}_K$

Optionally, use $\boldsymbol{V}_K^T$ to rotate $\mathbf{Z}_{\text{test}}$ back to original subspace $\mathbf{X}'_{\text{test}}$ and uncenter



Sketch of PCA

1. Select "best" $\boldsymbol{V} \in \mathbb{R}^{K \times M}$
2. Project down: $\boldsymbol{z}^{(i)} = \boldsymbol{V}\boldsymbol{x}^{(i)} \quad \forall i$
3. Reconstruct up: $\boldsymbol{x}'^{(i)} = \boldsymbol{V}^T \boldsymbol{z}^{(i)}$

Sketch of PCA

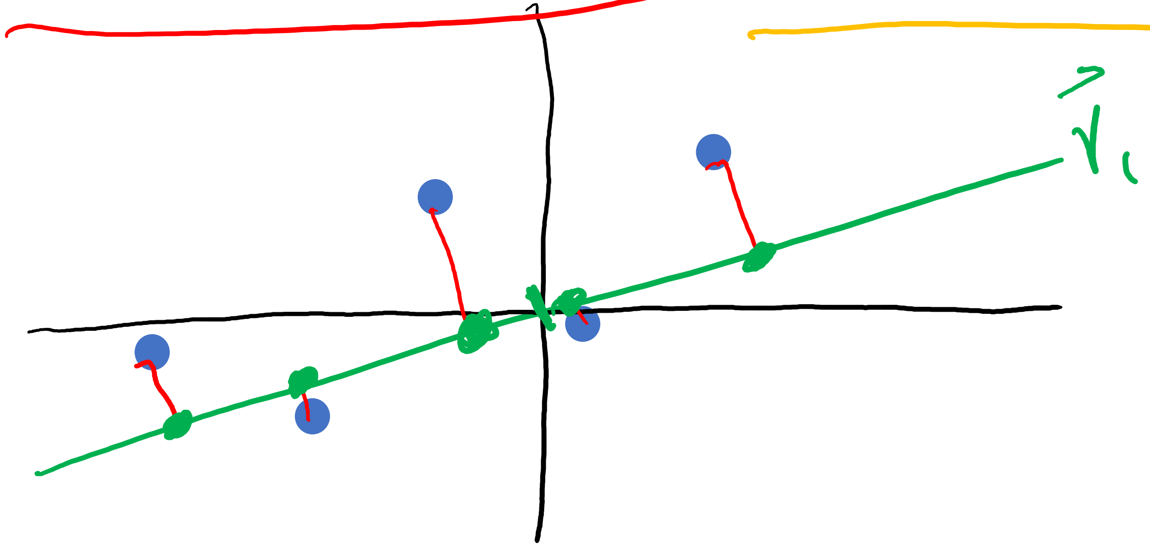
1. Select "best" $\boldsymbol{V} \in \mathbb{R}^{K \times M}$
2. Project down: $\boldsymbol{z}^{(i)} = \boldsymbol{V}\boldsymbol{x}^{(i)} \quad \forall i$
3. Reconstruct up: $\boldsymbol{x}'^{(i)} = \boldsymbol{V}^T \boldsymbol{z}^{(i)}$

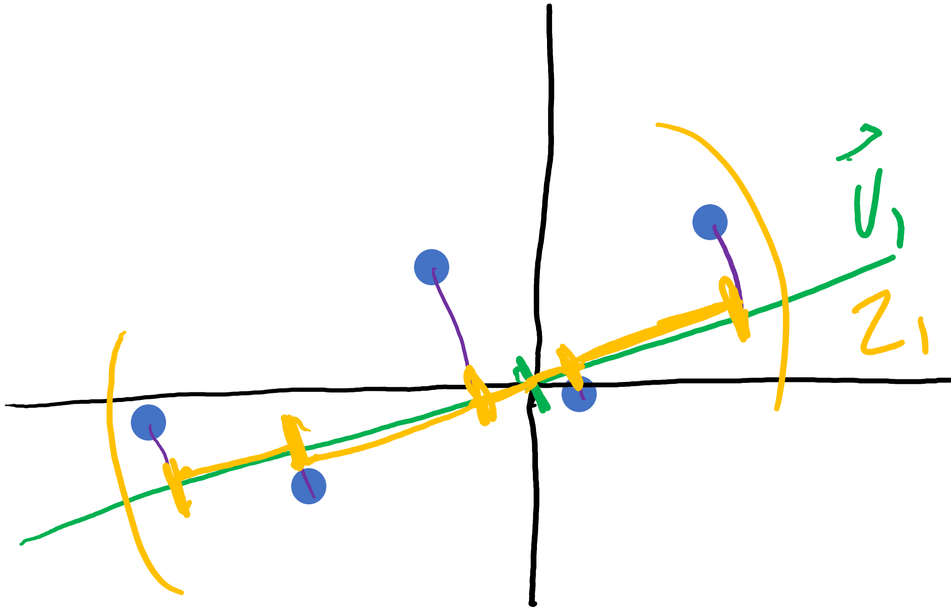
Definition of PCA

1. Select v_1 that best explains data
2. Select next v_j that
   i. Is orthogonal to $v_1, \ldots, v_{j-1}$
   ii. Best explains remaining data
3. Repeat 2 until desired amount of data is explained

Select "Best" Vector

Reconstruction Error vs Variance of Projection





$$\min \sum \| x^{(i)} - \hat{x}^{(i)} \|_2^2$$

$$\hat{x}^{(i)} = ((x^T v) v)$$

$$\max \sum (x^{iT} v)^2$$

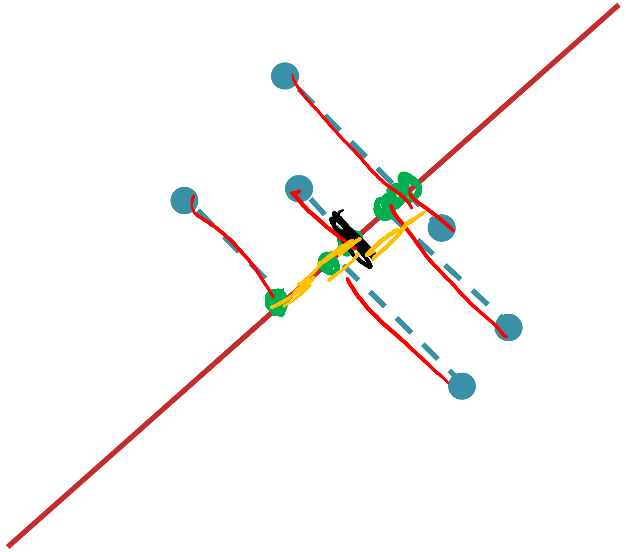
Poll 2 & Poll 3

Consider the two projections below

Poll 2: Which maximizes the variance?

Poll 3: Which minimizes the reconstruction error?

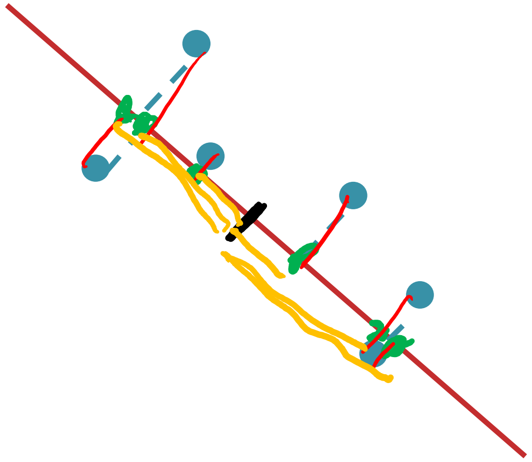
Option B



Option C

3 ✓

2 ✓

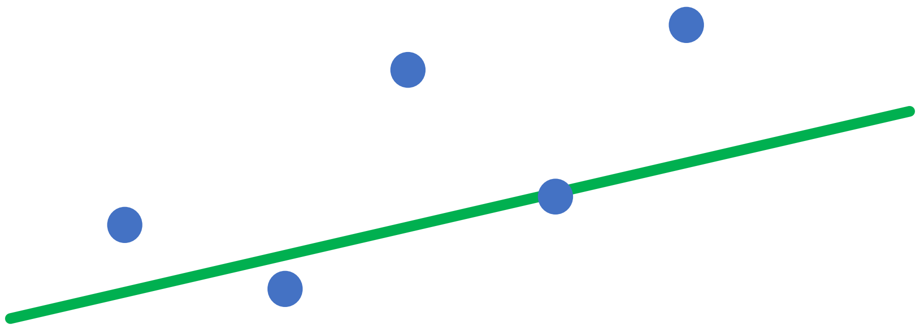


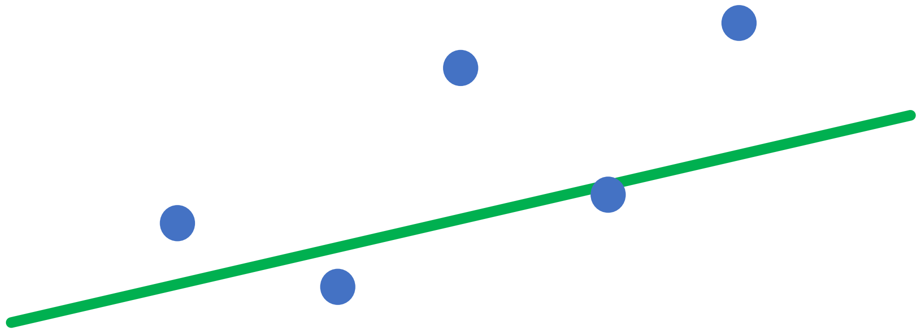
Option A

Calamity

Select “Best” Vector

Reconstruction Error vs Variance of Projection





Reconstruction Error

$$\left\|\mathbf{x}^{(i)} - \mathbf{x}'^{(i)}\right\|_2^2$$

$$\mathbf{v}^* = \underset{\substack{\mathbf{v} \\ s.t.\|\mathbf{v}\|_2=1}}{\operatorname{argmin}} \sum_{i=1}^{N} \left\|\mathbf{x}^{(i)} - \left(\mathbf{v}^T\mathbf{x}^{(i)}\right)\mathbf{v}\right\|_2^2$$

Variance of Projection

$$\mathbf{v}^* = \underset{\substack{\mathbf{v} \\ s.t.\|\mathbf{v}\|_2=1}}{\operatorname{argmax}} \sum_{i=1}^{N} \left(\mathbf{v}^T\mathbf{x}^{(i)}\right)^2$$

PCA

Equivalence of Maximizing Variance and Minimizing Reconstruction Error

Claim: Minimizing the reconstruction error is equivalent to maximizing the variance.

Proof: First, note that:

$$||\mathbf{x}^{(i)} - (\mathbf{v}^T\mathbf{x}^{(i)})\mathbf{v}||^2 = ||\mathbf{x}^{(i)}||^2 - (\mathbf{v}^T\mathbf{x}^{(i)})^2 \quad (1)$$

since $\mathbf{v}^T\mathbf{v} = ||\mathbf{v}||^2 = 1$.

Substituting into the minimization problem, and removing the extraneous terms, we obtain the maximization problem.

$$\mathbf{v}^* = \underset{\mathbf{v}:||\mathbf{v}||^2=1}{\operatorname{argmin}} \frac{1}{N}\sum_{i=1}^{N} ||\mathbf{x}^{(i)} - (\mathbf{v}^T\mathbf{x}^{(i)})\mathbf{v}||^2 \quad (2)$$

$$= \underset{\mathbf{v}:||\mathbf{v}||^2=1}{\operatorname{argmin}} \frac{1}{N}\sum_{i=1}^{N} ||\mathbf{x}^{(i)}||^2 - (\mathbf{v}^T\mathbf{x}^{(i)})^2 \quad (3)$$

$$= \underset{\mathbf{v}:||\mathbf{v}||^2=1}{\operatorname{argmax}} \frac{1}{N}\sum_{i=1}^{N} (\mathbf{v}^T\mathbf{x}^{(i)})^2 \quad (4)$$

Sketch of PCA

1. Select "best" $\boldsymbol{V} \in \mathbb{R}^{K \times M}$
2. Project down: $\boldsymbol{z}^{(i)} = \boldsymbol{V}\boldsymbol{x}^{(i)} \quad \forall i$
3. Reconstruct up: $\boldsymbol{x}'^{(i)} = \boldsymbol{V}^T \boldsymbol{z}^{(i)}$

Definition of PCA

1. Select v_1 that best explains data
2. Select next v_j that
   i. Is orthogonal to $v_1, \ldots, v_{j-1}$
   ii. Best explains remaining data
3. Repeat 2 until desired amount of data is explained

PCA: The First Principal Component

Use method of Lagrange multipliers

PCA: the First Principal Component

To find the first principal component, we wish to solve the following constrained optimization problem (variance maximization).

$$\mathbf{v}_1 = \underset{\mathbf{v}:||\mathbf{v}||^2=1}{\text{argmax}}\ \mathbf{v}^T \mathbf{\Sigma} \mathbf{v} \tag{1}$$

So we turn to the method of Lagrange multipliers. The Lagrangian is:

$$\mathcal{L}(\mathbf{v}, \lambda) = \mathbf{v}^T \mathbf{\Sigma} \mathbf{v} - \lambda(\mathbf{v}^T \mathbf{v} - 1) \tag{2}$$

Taking the derivative of the Lagrangian and setting to zero gives:

$$\frac{d}{d\mathbf{v}} \left(\mathbf{v}^T \mathbf{\Sigma} \mathbf{v} - \lambda(\mathbf{v}^T \mathbf{v} - 1) \right) = 0 \tag{3}$$

$$\mathbf{\Sigma} \mathbf{v} - \lambda \mathbf{v} = 0 \tag{4}$$

$$\mathbf{\Sigma} \mathbf{v} = \lambda \mathbf{v} \tag{5}$$

Recall: For a square matrix $\mathbf{A}$, the vector $\mathbf{v}$ is an **eigenvector** iff there exists **eigenvalue** λ such that:

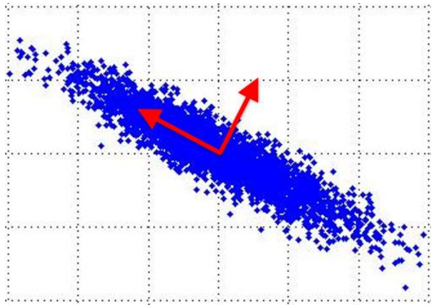
$$\mathbf{A} \mathbf{v} = \lambda \mathbf{v} \tag{6}$$

PCA: The Next Principal Component

Compute the next principal component from the residuals

Principal Component Analysis (PCA)

$(X^T X)v = \lambda v$, so v (the first PC) is the eigenvector of sample covariance matrix $X^T X$



Sample variance of projection $v^T X^T X\, v = \lambda v^T v = \lambda$

Thus, the eigenvalue λ denotes the amount of variability captured along that dimension (aka amount of energy along that dimension).

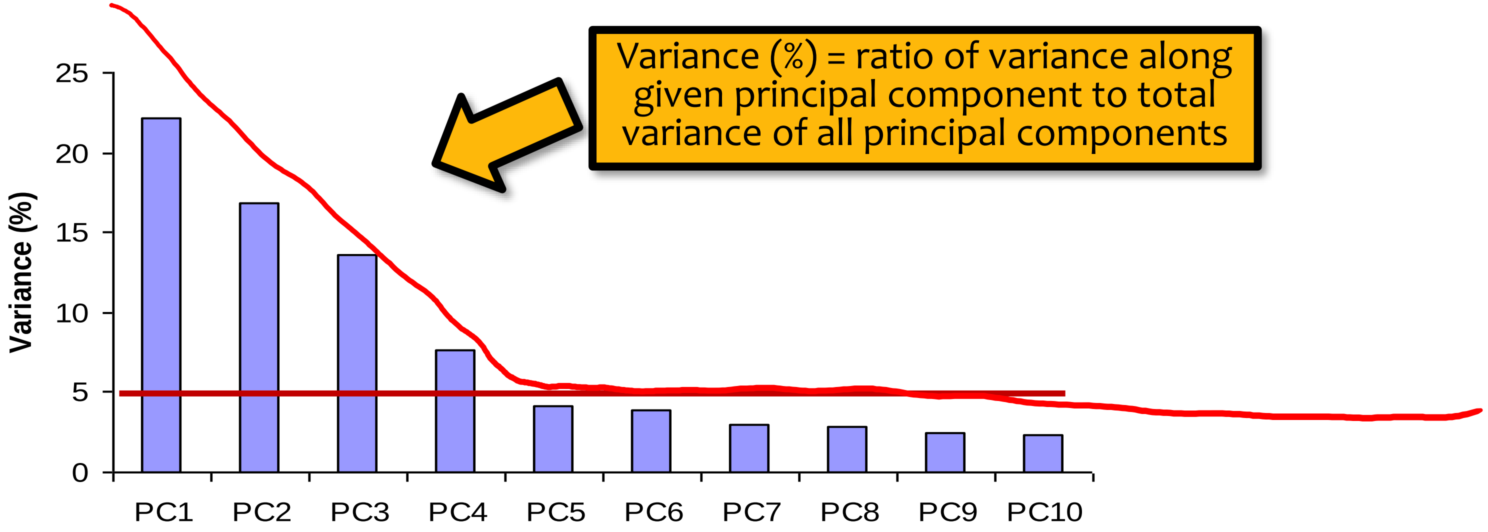
Eigenvalues $\lambda_1 \geq \lambda_2 \geq \lambda_3 \geq \cdots$

- The 1st PC v_1 is the eigenvector of the sample covariance matrix $X^T X$ associated with the largest eigenvalue
- The 2nd PC v_2 is the eigenvector of the sample covariance matrix $X^T X$ associated with the second largest eigenvalue
- And so on …

Slide from Nina Balcan

How Many PCs?

- For M original dimensions, sample covariance matrix is MxM, and has up to M eigenvectors. So M PCs.
- Where does dimensionality reduction come from?
  Can *ignore* the components of lesser significance.



- You do lose some information, but if the eigenvalues are small, you don't lose much
  - M dimensions in original data
  - calculate M eigenvectors and eigenvalues
  - choose only the first D eigenvectors, based on their eigenvalues
  - final data set has only D dimensions

SVD for PCA

SVD matrix factorization

$\boldsymbol{X} = \boldsymbol{U}\boldsymbol{S}\boldsymbol{V}^T, \; A \in \mathbb{R}^{N \times M}$

$\boldsymbol{U}$: $N \times$ N orthogonal matrix

- Columns of $\boldsymbol{U}$ are *left* singular vectors of $\boldsymbol{X}$
- Columns of $\boldsymbol{U}$ are eigenvectors of $\boldsymbol{X}\boldsymbol{X}^T$

$\boldsymbol{V}$: $M \times M$ orthogonal matrix

- Columns of $\boldsymbol{V}$ are *right* singular vectors of $\boldsymbol{X}$
- Columns of $\boldsymbol{V}$ are eigenvectors of $\boldsymbol{X}^T\boldsymbol{X}$

$\boldsymbol{S}$: $N \times M$ diagonal matrix

- Diagonal entries are singular values of $\boldsymbol{X}$, σ_k
- Each σ_k^2 are the eigenvalues of both $\boldsymbol{X}\boldsymbol{X}^T$ and $\boldsymbol{X}^T\boldsymbol{X}$!!

eig(X^TX)

$U, S, V \leftarrow$ svd(X)

SVD for PCA

For any arbitrary matrix $\mathbf{A}$, SVD gives a decomposition:

$$\mathbf{A} = \mathbf{U}\boldsymbol{\Lambda}\mathbf{V}^T \qquad (1)$$

where $\boldsymbol{\Lambda}$ is a diagonal matrix, and $\mathbf{U}$ and $\mathbf{V}$ are orthogonal matrices.

Suppose we obtain an SVD of our data matrix $\mathbf{X}$, so that:

$$\mathbf{X} = \mathbf{U}\boldsymbol{\Lambda}\mathbf{V}^T \qquad (1)$$

Now consider what happens when we rewrite $\boldsymbol{\Sigma} = \frac{1}{N}\mathbf{X}^T\mathbf{X}$ terms of this SVD.

$$\begin{aligned}
\boldsymbol{\Sigma} &= \frac{1}{N}\mathbf{X}^T\mathbf{X} && (2)\\
&= \frac{1}{N}(\mathbf{U}\boldsymbol{\Lambda}\mathbf{V}^T)^T(\mathbf{U}\boldsymbol{\Lambda}\mathbf{V}^T) && (3)\\
&= \frac{1}{N}(\mathbf{V}\boldsymbol{\Lambda}^T\mathbf{U}^T)(\mathbf{U}\boldsymbol{\Lambda}\mathbf{V}^T) && (4)\\
&= \frac{1}{N}\mathbf{V}\boldsymbol{\Lambda}^T\boldsymbol{\Lambda}\mathbf{V}^T && (5)\\
&= \frac{1}{N}\mathbf{V}(\boldsymbol{\Lambda})^2\mathbf{V}^T && (6)
\end{aligned}$$

Above we used the fact that $\mathbf{U}^T\mathbf{U} = \mathbf{I}$ since $\mathbf{U}$ is orthogonal by definition.

We find that $(\boldsymbol{\Lambda})^2$ is a diagonal matrix whose entries are $\Lambda_{ii} = \lambda_i^2$ the squares of the eigenvalues of the SVD of $\mathbf{X}$. Further, both $\mathbf{X}$ and $\mathbf{X}^T\mathbf{X}$ share the same eigenvectors in their SVD.

Thus, we can run SVD on $\mathbf{X}$ without ever instantiating the large $\mathbf{X}^T\mathbf{X}$ to obtain the necessary principal components more efficiently.

PCA Algorithm

Input: $\boldsymbol{X}, \boldsymbol{X}_{test}, K$

1. Center data (and scale each axis) based on training data → $\boldsymbol{X}, \boldsymbol{X}_{\boldsymbol{test}}$
2. $\boldsymbol{V}$ = eigenvectors($\boldsymbol{X}^T\boldsymbol{X}$) ← svd(X)
3. Keep only the top K eigenvectors: $\boldsymbol{V}_K$
4. $\mathbf{Z}_{\text{test}} = \boldsymbol{X}_{test}\boldsymbol{V}_K$

Optionally, use $\boldsymbol{V}_K^T$ to rotate $\mathbf{Z}_{\text{test}}$ back to original subspace $\mathbf{X}'_{\text{test}}$ and uncenter