Dimensionality Reduction (Part 1)
PCA (Principle Component Analysis)

Ivan Corneillet

Outline

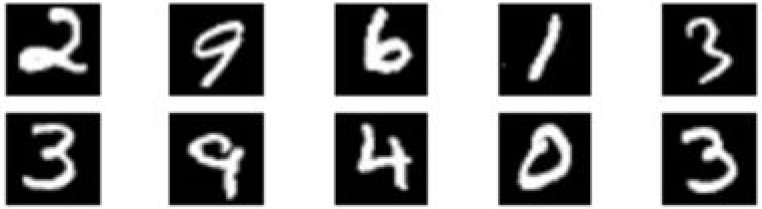
- why reduce dimensions?
- one common technique for reducing dimensionality:
  - PCA (Principle Component Analysis)
- sprint: apply PCA
  - (1) on handwritten digits
  - (2) to remove redundant features

What is the dimensionality of our data?

8 features → dimensionality of 8

	mpg	cylinders	displacement	horsepower	weight	acceleration	model	origin	car_name
0	18	8	307	130.0	3504	12.0	70	1	chevrolet chevelle malibu
1	15	8	350	165.0	3693	11.5	70	1	buick skylark 320
2	18	8	318	150.0	3436	11.0	70	1	plymouth satellite
3	16	8	304	150.0	3433	12.0	70	1	amc rebel sst
4	17	8	302	140.0	3449	10.5	70	1	ford torino

handwritten digits made of images of 28 × 28 pixels (horizontally × vertically)

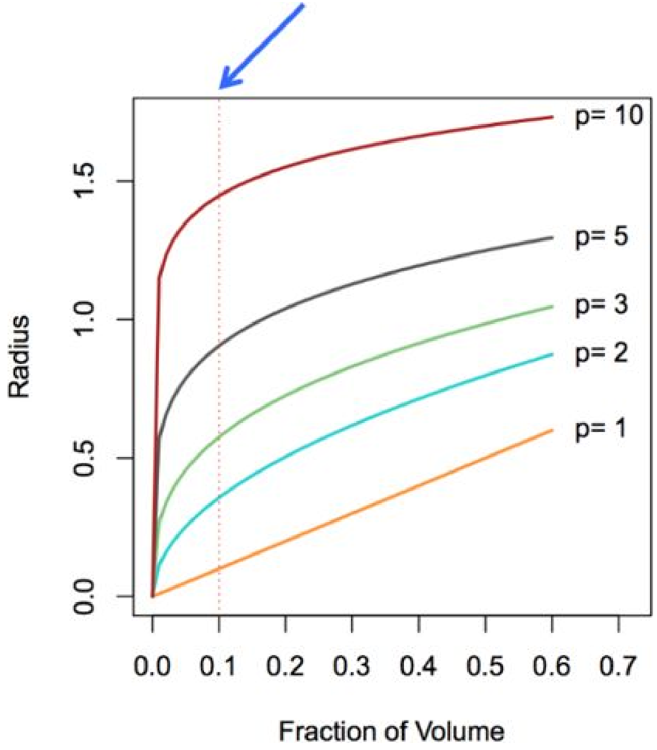


28 × 28 = 784 pixels are used to represent a handwritten digit → 784 features → dimensionality of 784

“dimensionality” = “number of dimensions” = “number of features/predictors”

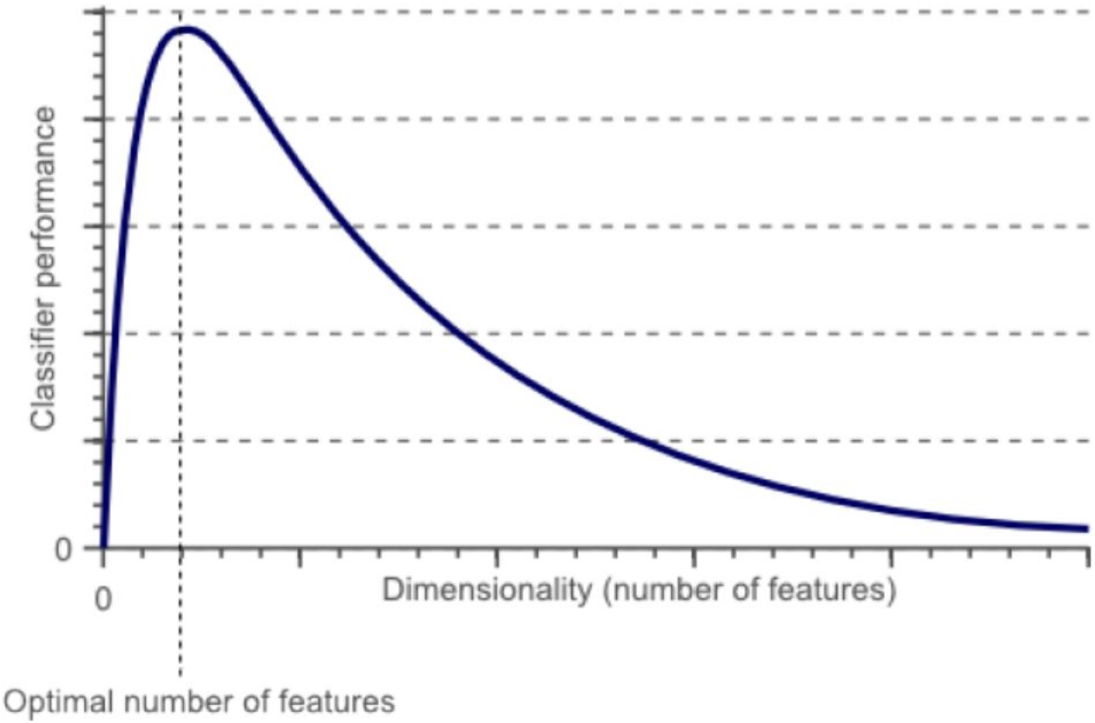
The Curse of Dimensionality: Sparsity of Data Points

- as dimensionality increases, the (average) distance between data points increases
    - the higher dimensional spaces become sparser (assuming the number of data points remain constant)



The Curse of Dimensionality → Models' Performance Decrease

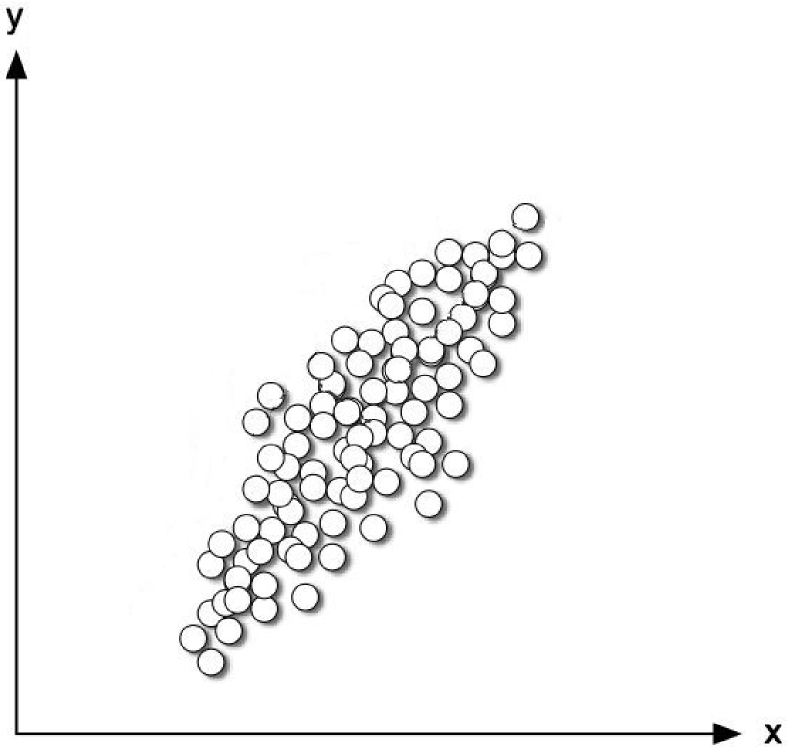
- if you have too few features (dimensions), the classifier is missing important information (underfitting)
- however, past an optimal number of dimensions, the information being fitted is mostly noise (overfitting) (see k-NN/Decision Trees) and the performance of the model decrease (assuming the number of data points remain constant)



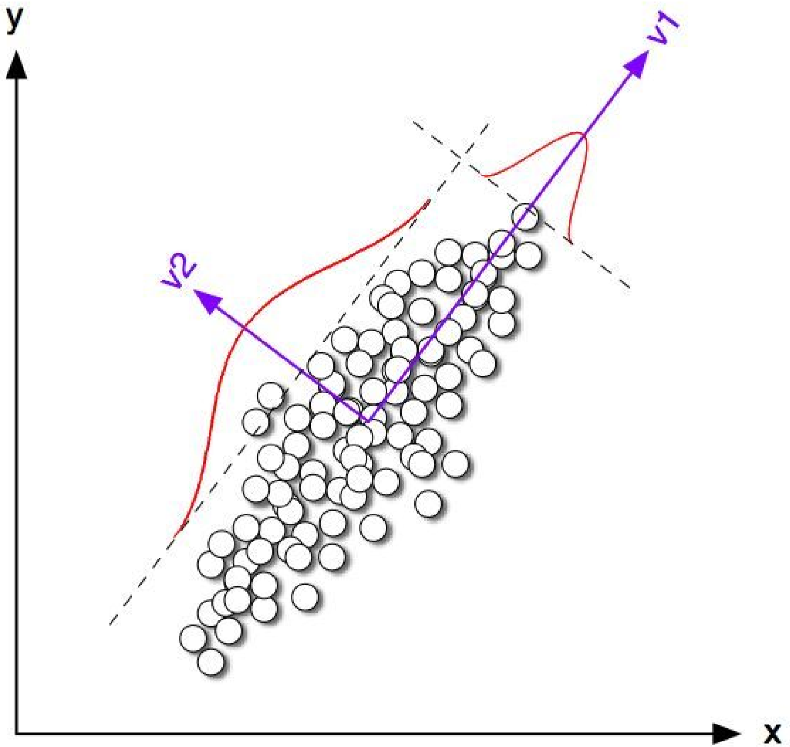
PCA (Principle Component Analysis)

- with many dimensions, you can bet that many features will be correlated (e.g., neighboring pixels for images)
- if the data is highly correlated, there is redundant information
- what if we could reduce the amount of redundant information by decorrelating the input vectors?

PCA graphically: a new set of axis



PCA graphically: a new set of axis (cont.)



let’s derive PCA on the board

PCA: whiteboard summary

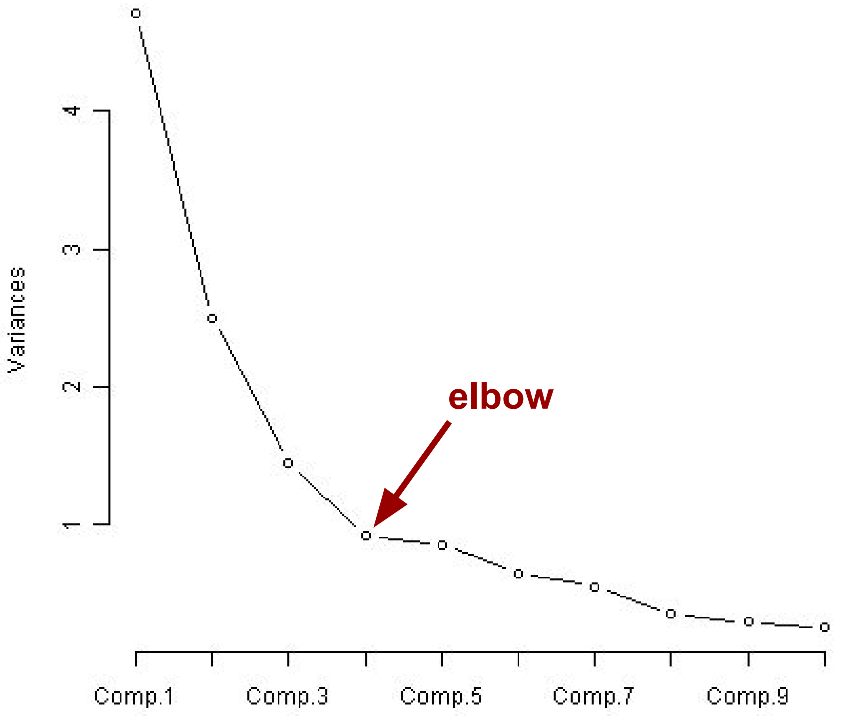
- create the centered design matrix X (n rows/observations × p columns/features)
  - (meaning that each column vector is centered around its mean)
- calculate the covariance matrix X^TX (a p × p square matrix)
- the principal components are the eigenvectors of the covariance matrix; the principal components' variance (σ^2) is
  - ordering the principal components/eigenvectors by decreasing variance/eigenvalue, you get an orthogonal basis capturing the directions of the most-to-least variance of your data

PCA: how many dimensions should we retain?

- the fraction of total variance captured by the first r principal component is

$$f(r) = \frac{\lambda_1 + \cdots + \lambda_r}{\lambda_1 + \cdots + \lambda_p}$$

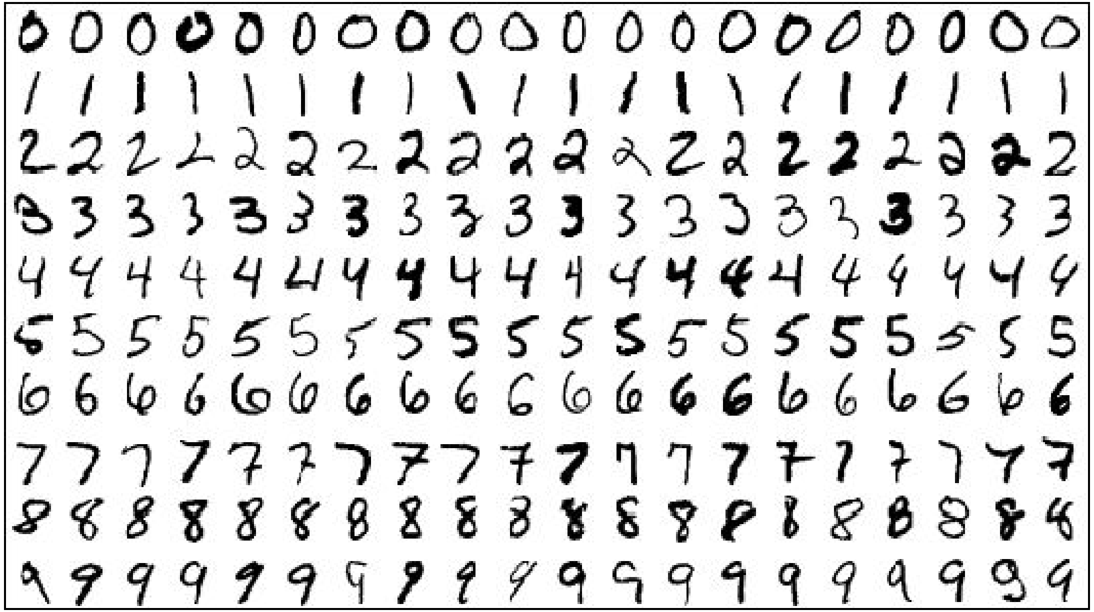
- a scree plot graphs eigenvalues against principal components
  - it is a useful visual aid for determining an appropriate number of principal components: to determine the appropriate number of components, we look for an "elbow" in the scree plot



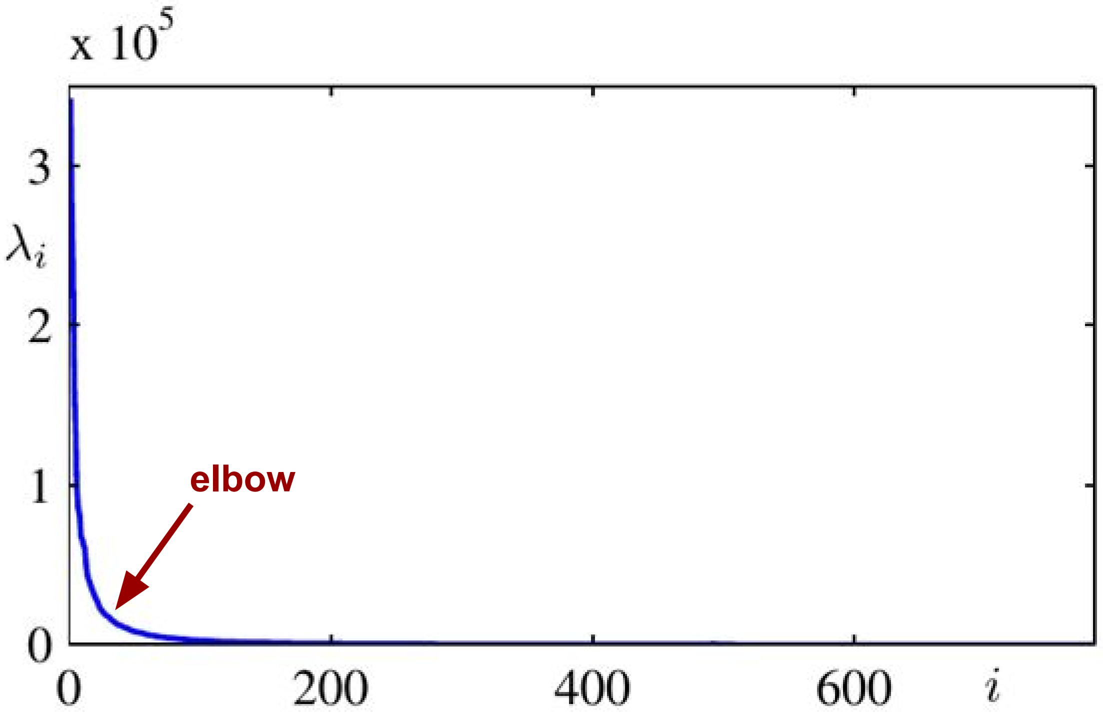
Application of PCA: MNIST Dataset

dataset of handwritten images digits

- 28 × 28 pixels: 784 dimensions
- grayscale color: 0 (black) to 255 (white)
- 10 classes (digits from 0 to 9)
- 6,000 training examples



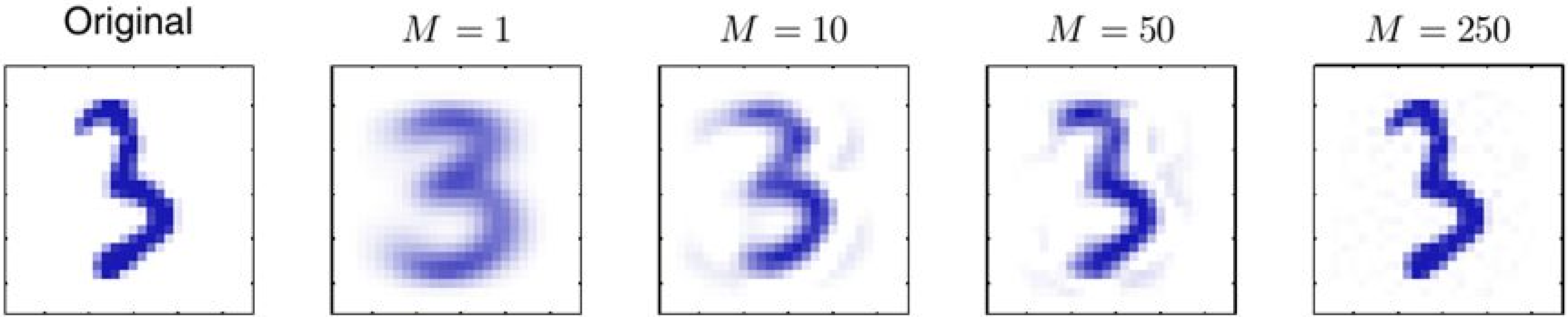
PCA on MNIST: Scree Plot



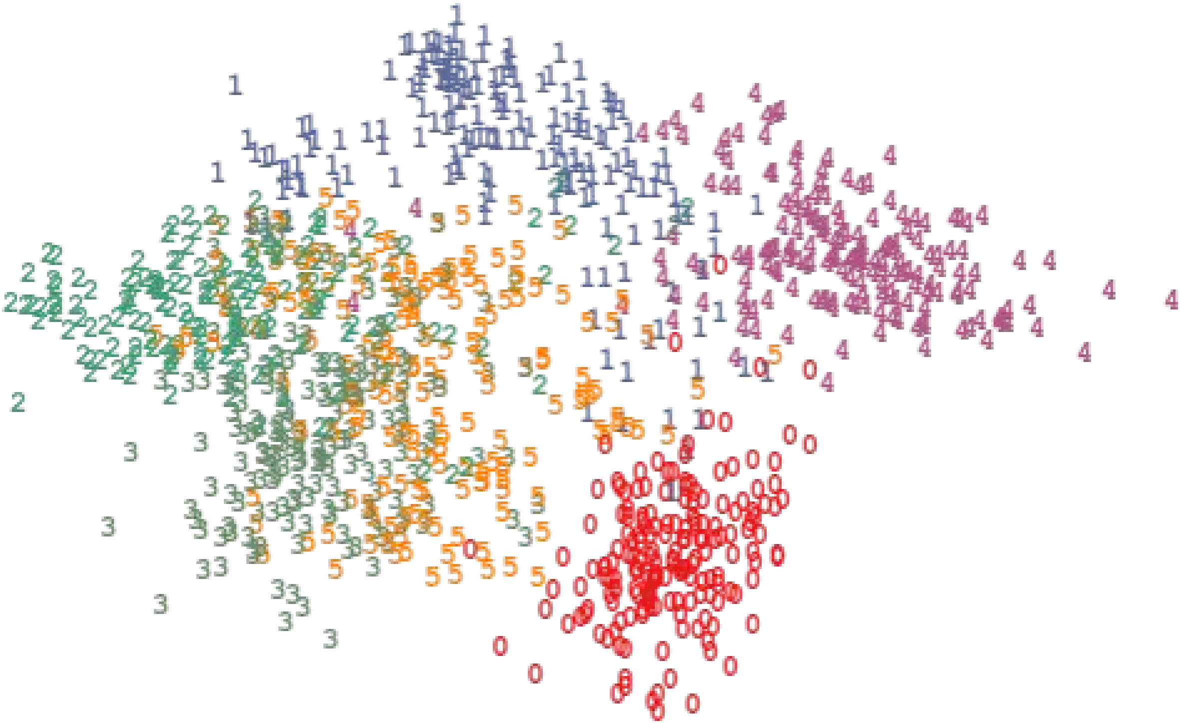
- look how fast the variance captured by the principal components drop
- what the implication?

PCA on MNIST

implication: you can reconstruct most images with the top principal components



k-NN after PCA on MNIST to classify digits



we can use the top principal components resulting from PCA (up to 4, remember?) as features to train a k-NN classifier to classify the handwritten digits

(that’s all for this morning)

Dimensionality Reduction (Part 2)
SVD (Singular Value Decomposition)

Ivan Corneillet

Outline

- SVD (Singular Value Decomposition) to perform PCA
- SVD for Topic Analysis
- sprint:
  - (1) SVD to perform PCA
  - (2) SVD for Topic Analysis

let’s derive SVD on the board

SVD for Topic Analysis

	Matrix	Alien	Serenity	Casablanca	Amelie
Alice	1	2	2	0	0
Bob	3	5	5	0	0
Cindy	4	4	4	0	0
Dan	5	5	5	0	0
Emily	0	2	0	4	4
Frank	0	0	0	5	5
Greg	0	1	0	2	2

SVD for Topic Analysis (cont.)

```
U =
         0    1    2    3
Alice -0.2  0.0  0.3 -0.3
Bob   -0.5  0.1  0.5 -0.5
Cindy -0.5  0.1 -0.3  0.2
Dan   -0.6  0.1 -0.4  0.2
Emily -0.1 -0.6  0.4  0.5
Frank  0.0 -0.7 -0.4 -0.5
Greg  -0.1 -0.3  0.2  0.3
(7, 4)

S =
[[ 13.8   0.    0.    0. ]
 [  0.    9.5   0.    0. ]
 [  0.    0.    1.7   0. ]
 [  0.    0.    0.    1. ]]
(4L, 4L)

V_t =
   Matrix  Alien  Serenity  Casablanca  Amelie
0    -0.5   -0.6      -0.6        -0.1    -0.1
1     0.1    0.0       0.1        -0.7    -0.7
2    -0.8    0.6       0.0        -0.1    -0.1
3     0.4    0.5      -0.8        -0.1    -0.1
(4, 5)
```

(that’s all for today)