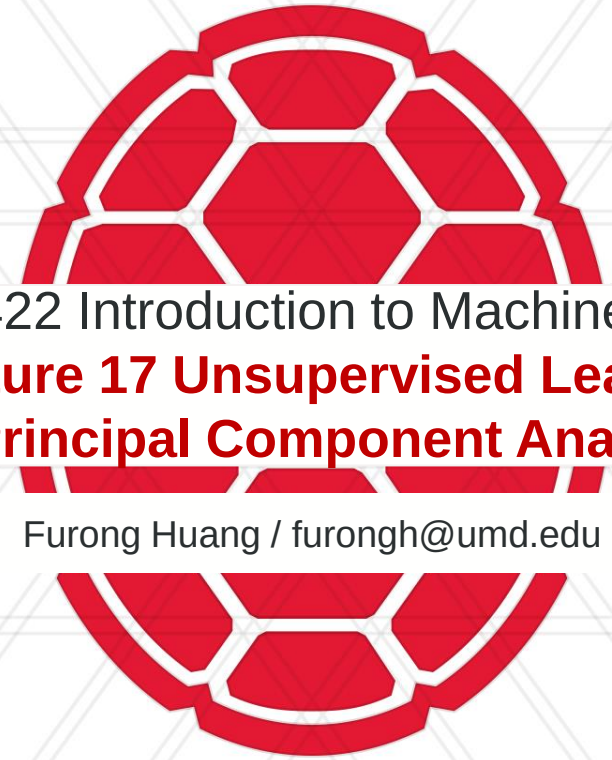
Slides adapted from Prof Carpuat and Duraiswami



CMSC 422 Introduction to Machine Learning

Lecture 17 Unsupervised Learning – Principal Component Analysis

Furong Huang / furongh@umd.edu

UNIVERSITY OF MARYLAND

Naïve Bayes classifier

$$\hat{y} = argmax_y\, P(Y = y \,|X = x)$$
$$= argmax_y P(Y = y)P(X = x \,|Y = y)$$
$$= argmax_y P(Y = y) \prod_{i=1}^{d} P(X_i = x_i \,|Y = y)$$

Bayes rule

\+ Conditional independence assumption

How many parameters do we need to learn?

(Suppose all random variables are Boolean)

To describe P(Y)?

- Bernoulli (biased coin): 1 parameter

To describe $P(X = < X_1, X_2, \ldots X_d > | Y)$

Without conditional independence assumption?

- For $P(X_1 \ldots . X_d | Y = 0)$: $2^d - 1$ parameters
- For $P(X_1 \ldots . X_d | Y = 1)$: $2^d - 1$ parameters
- Total = $2(2^d - 1)$ parameters

With conditional independence assumption?

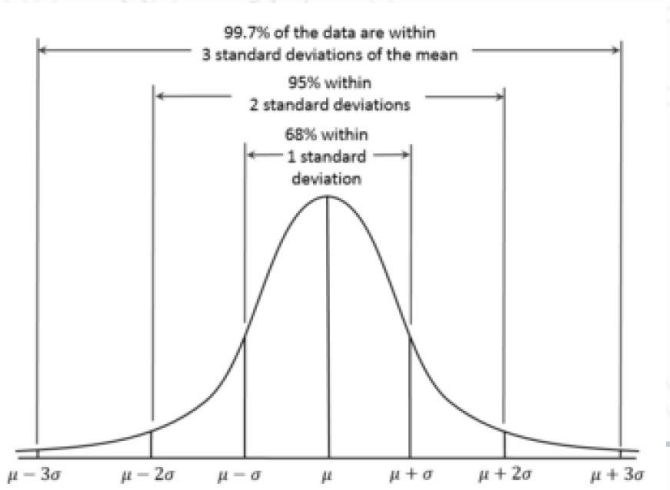
- P(Xi|Y=0): 1 parameter
- P(Xi|Y=1): 1 parameter
- Total per X_i: 2 parameters
- Total = 2d parameters

Naïve Bayes Properties

Naïve Bayes is a linear classifier

- See CIML for example of computation of Log Likelihood Ratio

Choice of probability distribution is a form of inductive bias



Generative Stories

Probabilistic models tell a fictional story explaining how our training data was created

Example of a generative story for a multiclass classification task with continuous features

For each example $n = 1 \dots N$:

(a) Choose a label $y_n \sim \mathcal{D}isc(\boldsymbol{\theta})$

(b) For each feature $d = 1 \dots D$:

i. Choose feature value $x_{n,d} \sim \mathcal{N}or(\mu_{y_n,d}, \sigma^2_{y_n,d})$

From the Generative Story to the Likelihood Function

For each example $n = 1 \dots N$:

(a) Choose a label $y_n \sim \mathcal{D}isc(\boldsymbol{\theta})$

(b) For each feature $d = 1 \dots D$:

i. Choose feature value $x_{n,d} \sim \mathcal{N}or(\mu_{y_n,d}, \sigma^2_{y_n,d})$

$$p(D) = \overbrace{\prod_n \underbrace{\theta_{y_n}}_{\text{choose label}} \underbrace{\prod_d \frac{1}{\sqrt{2\pi\sigma^2_{y_n,d}}} \exp\left[-\frac{1}{2\sigma^2_{y_n,d}}(x_{n,d} - \mu_{y_n,d})^2\right]}_{\text{choose feature value}}}^{\text{for each example}}$$

for each feature

What you should know

The Naïve Bayes classifier

- Conditional independence assumption
- How to train it?
- How to make predictions?
- How does it relate to other classifiers we know?

Fundamental Machine Learning concepts

- iid assumption
- Bayes optimal classifier
- Maximum Likelihood estimation
- Generative story

Unsupervised Learning

Discovering hidden structure in data

What algorithms do we know for unsupervised learning?

K-Means Clustering

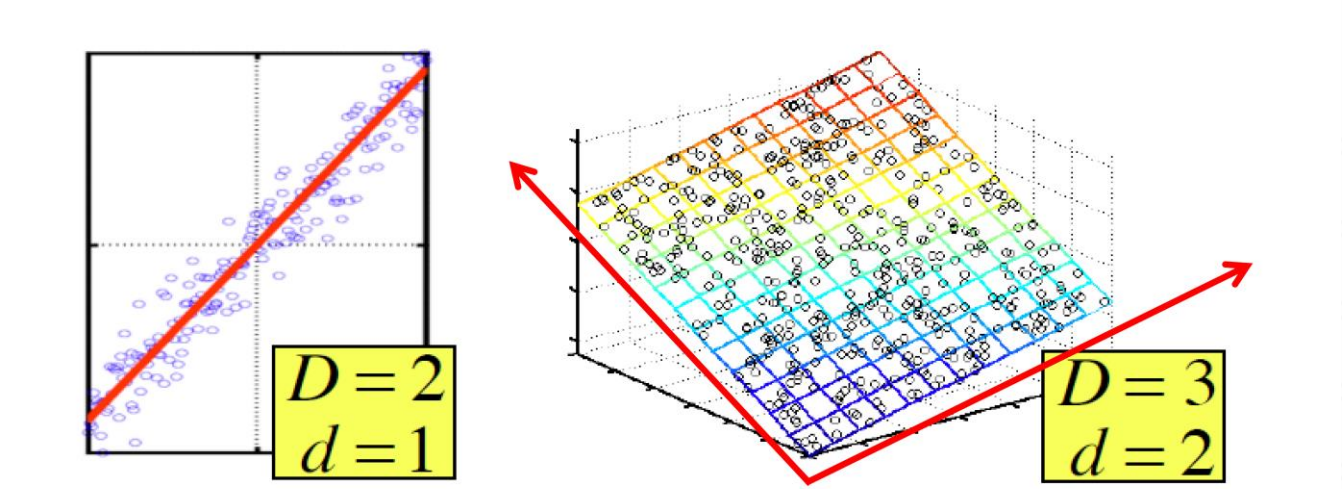
Today: how can we learn better representations of our data points?

Dimensionality Reduction

Goal: extract hidden lower-dimensional structure from high dimensional datasets

Why?

- To visualize data more easily
- To remove noise in data
- To lower resource requirements for storing/processing data
- To improve classification/clustering



Examples of data points in D dimensional space that can be effectively represented in a d-dimensional subspace (d < D)

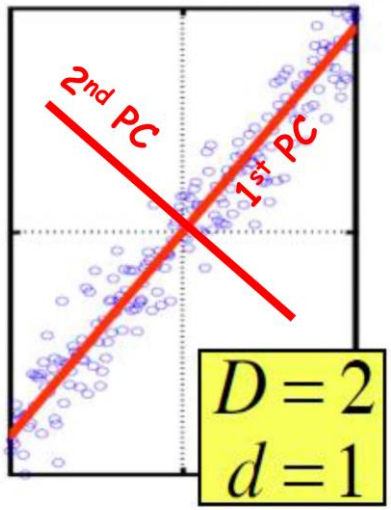
Principal Component Analysis

Goal: Find a **projection** of the data onto directions that **maximize variance** of the original data set

- Intuition: those are directions in which most information is encoded

Definition: **Principal Components** are orthogonal directions that capture most of the variance in the data

PCA: finding principal components



1st PC

Projection of data points along 1st PC discriminates data most along any one direction

2nd PC

next orthogonal direction of greatest variability

And so on…

PCA: notation

Data points

- Represented by matrix X of size NxD
- X_i is the i-th row, i.e., the i-th example
- X_{ij} is the value of j-th feature for example i
- Let's assume data is centered, i.e., $\sum_i X_i = \vec{0}$

Principal components are d vectors: $v_1, v_2, \ldots v_d$

$v_i^\top v_j = 0, i \neq j$ and $v_i^\top v_i = 1, \forall\, i$

The sample variance data projected on vector v is

$$\sum_{i=1}^{n} (x_i{}^T v)^2 = (Xv)^T (Xv)$$

PCA formally

Finding vector that maximizes sample variance of projected data:

$$argmax_v\, v^T X^T Xv \text{ such that } v^T v = 1$$

A constrained optimization problem(method of Lagrange multipliers)

- Lagrangian folds constraint into objective: $argmax_v\, v^T X^T Xv - \lambda(v^T v - 1)$
- Solutions are vectors v such that $X^T Xv = \lambda v$
  - i.e. eigenvectors of $X^T X$(sample covariance matrix)

Lagrange Multiplier

- A strategy for finding the local maxima and minima of a function subject to equality constraints
- Consider the optimization problem

$$\max_{v} f(v)$$

$$\text{Subject to } g(v) = 0$$

  - Introduce a new variable λ called a Lagrange multiplier
  - Study the Lagrange (Lagrangian) function

$$\mathcal{L}(v, \lambda) = f(v) - \lambda g(v)$$

Relationship between PCA and Eigen Value Decomposition

$$argmax_v \, v^T X^T Xv - \lambda(v^T v - 1) \qquad (*)$$

where X is the N by D data matrix

- For this optimization problem, taking derivative with respect to v, set the gradient to 0 to solve for the stationary point

$$X^T Xv - \lambda v = 0$$

- Therefore the eigenvector of covariance matrix $X^T X$ is a stationary point of the optimization problem (*).

PCA formally

- The eigenvalue λ denotes the amount of variability captured along dimension v

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

If we rank eigenvalues from large to small

The 1st PC is the eigenvector of $X^T X$ associated with largest eigenvalue

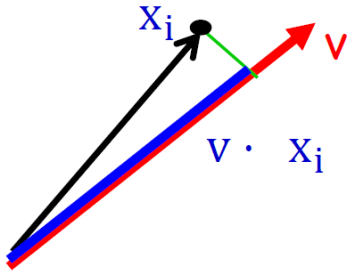
The 2nd PC is the eigenvector of $X^T X$ associated with 2nd largest eigenvalue

…

Alternative interpretation of PCA

PCA finds vectors v such that projection on to these vectors minimizes reconstruction error

$$\frac{1}{n}\sum_{i=1}^{n} \|\mathbf{x}_i - (\mathbf{v}^T\mathbf{x}_i)\mathbf{v}\|^2$$



Resulting PCA algorithm

Algorithm 36 PCA(X, K)

1: $\boldsymbol{\mu} \leftarrow \textsc{mean}(\mathbf{X})$ // compute data mean for centering

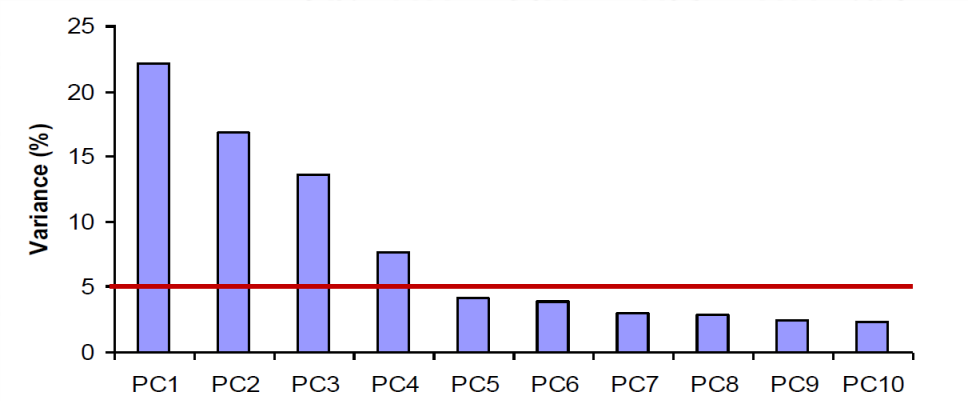
2: $\mathbf{D} \leftarrow \left(\mathbf{X} - \boldsymbol{\mu}\mathbf{1}^\top\right)^\top \left(\mathbf{X} - \boldsymbol{\mu}\mathbf{1}^\top\right)$ // compute covariance, $\mathbf{1}$ is a vector of ones

3: $\{\lambda_k, \mathbf{u}_k\} \leftarrow$ top K eigenvalues/eigenvectors of $\mathbf{D}$

4: **return** $(\mathbf{X} - \boldsymbol{\mu}\mathbf{1})\,\mathbf{U}$ // project data using $\mathbf{U}$

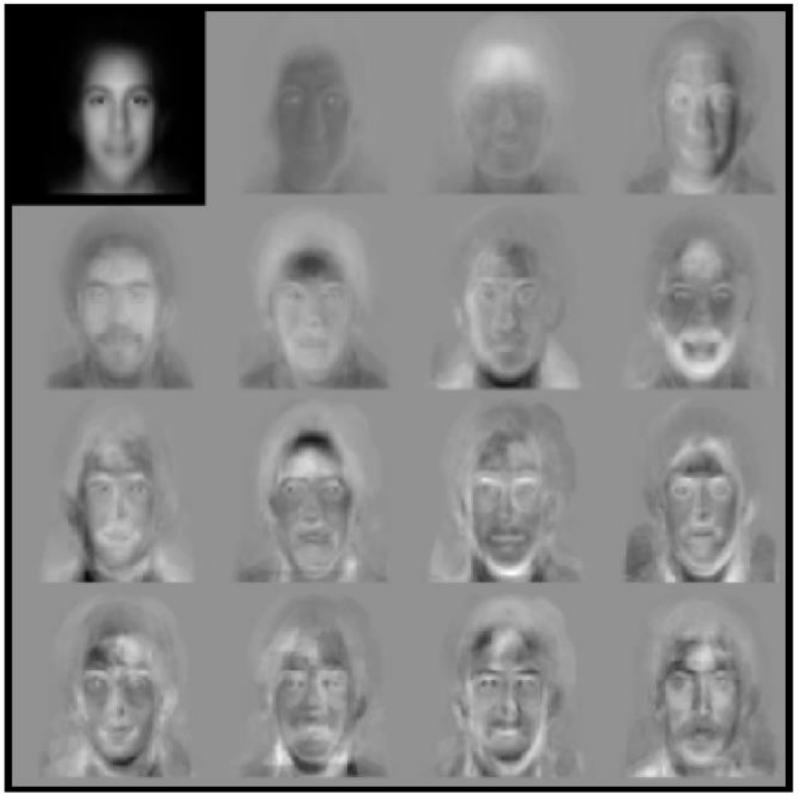
How to choose the hyperparameter K?

i.e. the number of dimensions



We can ignore the components of smaller significance

An example: Eigenfaces



Eigenfaces from 7562 images:

top left image is linear combination of rest.

Sirovich & Kirby (1987)
Turk & Pentland (1991)

PCA pros and cons

Pros

- Eigenvector method
- No tuning of the parameters
- No local optima

Cons

- Only based on covariance (2nd order statistics)
- Limited to linear projections

What you should know

Principal Components Analysis

Goal: Find a **projection** of the data onto directions that **maximize variance** of the original data set

PCA **optimization objectives** and resulting **algorithm**

Why this is useful!

UNIVERSITY OF
MARYLAND

Furong Huang
4124 IRB, College Park, MD 20740
301.405.8010 / furongh@umd.edu