Review Lecture 14

Principal Component Analysis

1. $$\Sigma = \text{cov}\left(X\right)$$

2. $$W = \text{eigs}\left(\Sigma, K\right)$$

3. $$Y = X - \mu \times W$$

When to use PCA

- Great when data is truly low dimensional (on a hyperplane (linear))
- Or **approximately** low dimensional (almost lie on plane Eg. very flat ellipsoid)
    - Eg. Dimensionality reduction for face images, for multiple biometric applications as preprocessing…

When to use PCA

- Warning: Scale matters (if some feature in in centimeters and others is meters, PCA direction can change)

- Problem 1 of Homework 1

Problem 1 of Homework 1

- Part 1: draw points on a 45 degree angle line
  - Eg. all coordinates are identical and values drawn for uniform distribution
- Part 2: (Scale matters)
  - Say first coordinate had a variance many magnitudes larger than other coordinates.
  - The coordinates are all independently drawn

When to use PCA

- Warning: direction of importance need not always be the one with good enough spread
  - Problem 2 of Homework 1

CCA Algorithm

1. $X = \left(n \begin{matrix} X_1 \\ d_1 \end{matrix}, \begin{matrix} X_2 \\ d_2 \end{matrix} \right)$

2. $\sum = \begin{matrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{matrix} = \mathrm{cov}\left(X \right)$

3. $W_1 = \mathrm{eigs}\left(\Sigma_{11}^{-1}\Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21}, K \right)$

4. $Y_1 = X_1 - \mu_1 \times W_1$

When to use CCA?

- CCA applies for problems where data can be split into 2 views X = [X1,X2]

- CCA picks directions of projection (in each view) where data is maximally correlated

- Maximizes correlation coefficient and not just covariance so is **scale free**

When to use CCA

- Scenario 1: You have two feature extraction techniques.
    - One provides excellent features for dogs Vs cats and noise on other classes
    - Other method provides excellent features for cars Vs bikes and noise for other classes
- What do we do?

    A. Use CCA to find one common representation

    B. Concatenate the two features extracted

When to use CCA

- Scenario 2: You have two cameras capturing images of the same objects from different angles.

- You have a feature extraction technique that provides feature vectors from each camera.

- You want to extract good features for recognizing the object from the two cameras

- What do we do?

    A. Use CCA to find one common representation

    B. Concatenate features provides excellent features for

Pick a Random W

$$Y = X \times \begin{bmatrix} +1 & \dots & -1 \\ -1 & \dots & +1 \\ +1 & \dots & -1 \\ & \vdots & \\ +1 & \dots & -1 \end{bmatrix}_K \Big/ \sqrt{d}$$

When to use RP?

- When data is huge and very large dimensional
- For PCA, CCA typically you think of K (no. of dimensions we reduce to) in double digits
- For RP think of K typically in 3-4 digit numbers
- RP guarantees preservation of inter-point distances.
    - RP unlike PCA and CCA does not project using unit vectors. (What does this mean?)
        - Problem 1 of Homework 2

How do we choose K?

- For PCA?
- For CCA?
- For Random Projection?

Kernel PCA

1. $\tilde{K}_{n} \; n = \text{compute}\left(k, X \right)$

2. $\left[A, \lambda \right] = \text{eigs}\left(\tilde{K}, K \right)$

3. $P = \left[\frac{A_1}{\sqrt{\lambda_1}} \cdots \frac{A_K}{\sqrt{\lambda_K}} \right]$

Kernel PCA

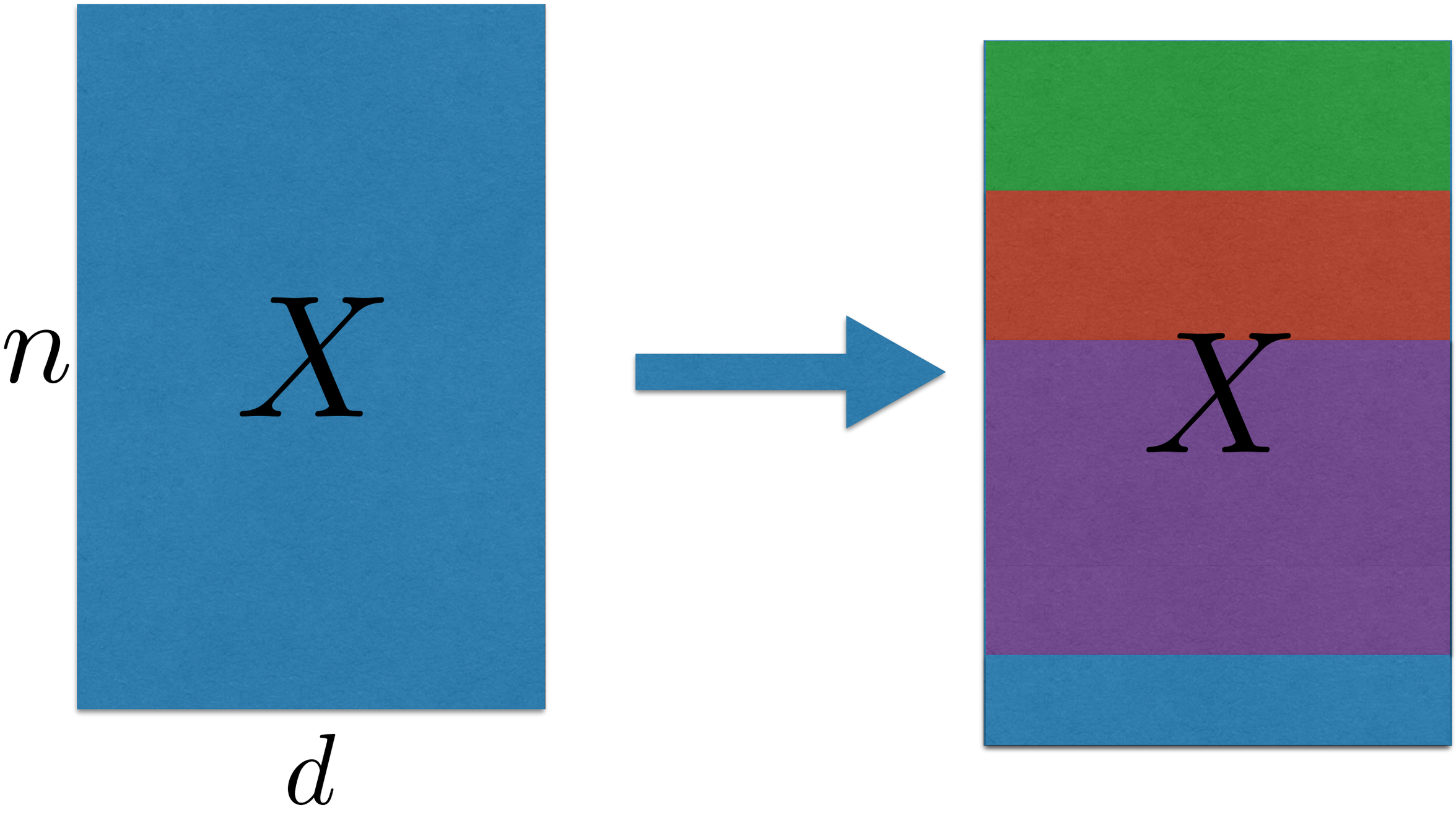
4. $Y = \tilde{K} \times P$

When to use Kernel PCA

- When data lies on some non-linear, low dimensional subspace

- Kernel function matters. (Eg. RBF kernel, only points close to a given point have non-negligible kernel evaluation)

Clustering



K-means

- K-means algorithm: (wishful thinking, EM)
  - Fix parameters (the k means) and compute new cluster assignments (or probabilities) for every point
  - Fix cluster assignment for all data points and re-evaluate parameters (the k-means)

Single-Link Clustering

- Start with all points being their own clusters

- Until we get K-clusters, merge the closest two clusters

When to Use Single Link

- When we have dense sampling of points within each cluster

- When not to use: when we might have outliers

When to use K-means

- When we have nice spherical round equal size clusters or cluster masses are far apart

- Handles outliers better

Homework 3

Spectral Clustering

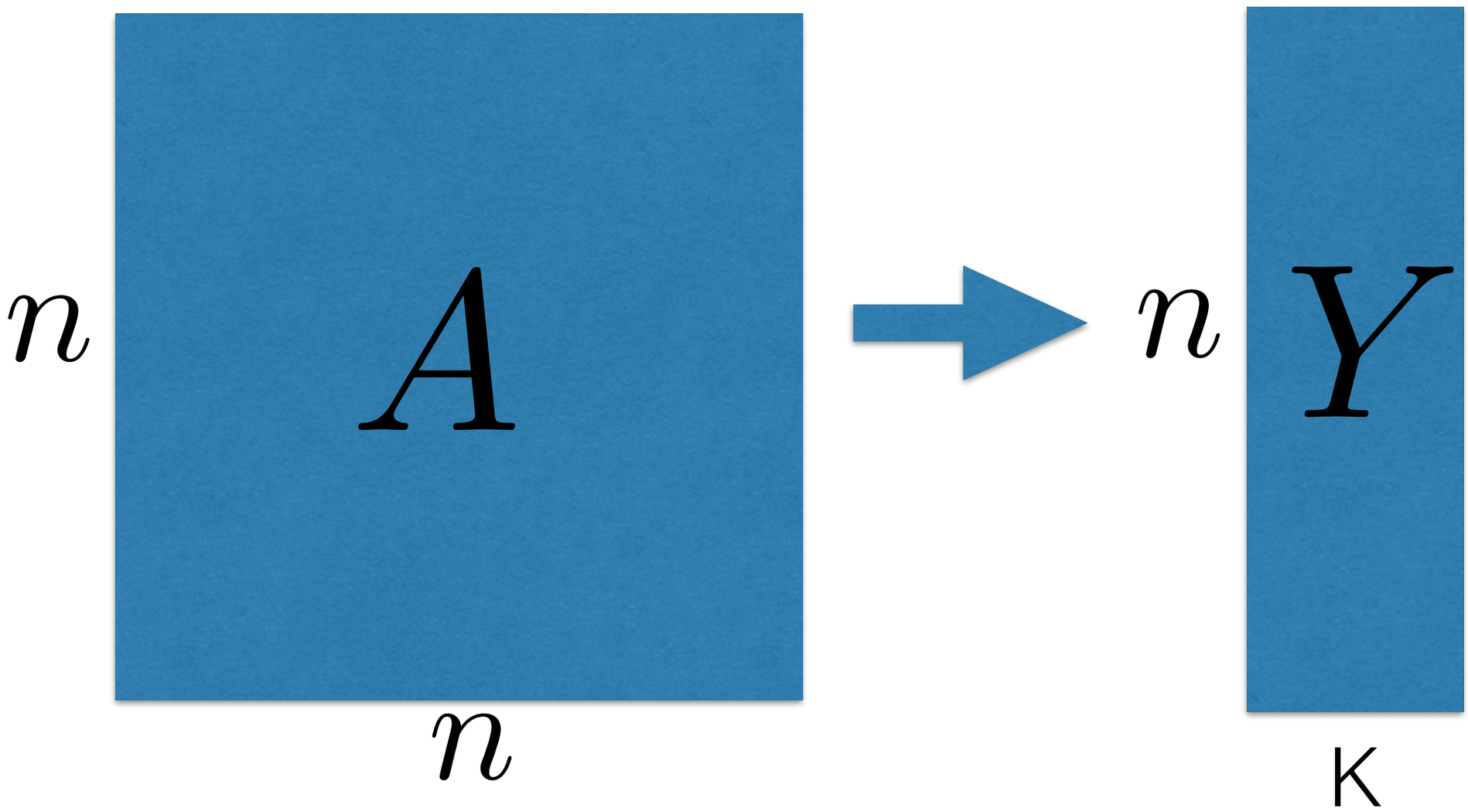
- You want to cluster nodes of a graph into groups based on connectivity

- Unnormalized spectral clustering: divide into groups where as few edges between groups are cut

Spectral Clustering Algorithm (Unnormalized)

1. Given matrix A calculate diagonal matrix D s.t. $D_{i,i} = \sum_{j=1}^{n} A_{i,j}$
2. Calculate the Laplacian matrix $L = D - A$
3. Find eigen vectors $\mathbf{v}_1, \ldots, \mathbf{v}_n$ of L (ascending order of eigenvalues)
4. Pick the K eigenvectors with smallest eigenvalues to get $\mathbf{y}_1, \ldots, \mathbf{y}_n \in \mathbb{R}^K$
5. Use K-means clustering algorithm on $\mathbf{y}_1, \ldots, \mathbf{y}_n$

Spectral Embedding



Use K-means on Y

Normalized Spectral Clustering

- Unnormalized spectral embedding encourages loner nodes to be pushed far away from rest

- This is indeed the minute solution to cut off loners

- Instead form clusters that minimize ratio of edges cut to number of edges each cluster has

  - (busy groups tend to form clusters)

- Algorithm, replace Laplacian matrix by normalized one

Spectral Clustering Algorithm (Normalized)

1. Given matrix A calculate diagonal matrix D s.t. $D_{i,i} = \sum_{j=1}^{n} A_{i,j}$
2. Calculate the normalized Laplacian matrix $\tilde{L} = I - D^{-1/2} A D^{-1/2}$
3. Find eigen vectors $\mathbf{v}_1, \ldots, \mathbf{v}_n$ of L (ascending order of eigenvalues)
4. Pick the K eigenvectors with smallest eigenvalues to get $\mathbf{y}_1, \ldots, \mathbf{y}_n \in \mathbb{R}^K$
5. Use K-means clustering algorithm on $\mathbf{y}_1, \ldots, \mathbf{y}_n$

When to use Spectral Clustering

- First, even works with weighted graph, where weight of edge represents similarity

- When knowledge about how clusters should be formed is solely decided by similarity between points, there is no underlying prior knowledge

Probabilistic Models

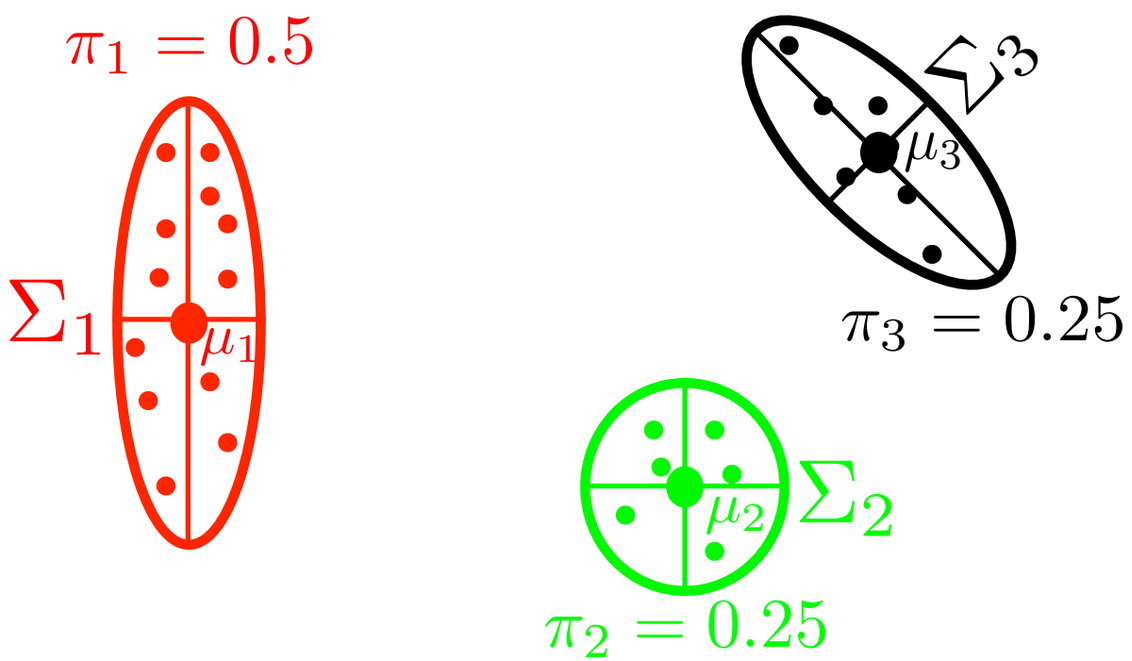
- Θ consists of set of possible parameters
- We have a distribution P_θ over the data induced by each $\theta \in \Theta$
- Data is generated by one of the $\theta \in \Theta$
- Learning: Estimate value or distribution for $\theta^* \in \Theta$ given data

Gaussian Mixture Models

Each $\theta \in \Theta$ is a model.

- Gaussian Mixture Model
  - Each θ consists of mixture distribution $\pi = (\pi_1, \ldots, \pi_K)$, means $\mu_1, \ldots, \mu_K \in \mathbb{R}^d$ and covariance matrices $\Sigma_1, \ldots, \Sigma_K$
  - For each t, independently:

$$c_t \sim \pi, \qquad x_t \sim N(\mu_{c_t}, \Sigma_{c_t})$$



GMM: Power of Wishful Thinking

1. Initialize model parameters $\pi^{(0)}$, $\mu_1^{(0)}, \ldots, \mu_K^{(0)}$ and $\Sigma_1^{(0)}, \ldots, \Sigma_K^{(0)}$
2. For i = 1 until convergence or bored
    1. Under current model parameters $\theta^{(i-1)}$, compute probability $Q_t^{(i)}(k)$ of each point $\mathbf{x}_t$ belonging to cluster k
    2. Given probabilities of each point belonging to the various clusters, compute optimal parameters $\theta^{(i)}$
3. End For

Mixture Models

- π is mixture distribution over the K-types
- $\gamma_1, \ldots, \gamma_K$ are parameters for K distributions
- Generative process:
  - Draw type $c_t \sim \pi$
  - Next given c_t, draw $x_t \sim \text{Distribtuion}(\gamma_{c_t})$

EM Algorithm for Mixture Models

For $i = 1$ to convergence

(E step) For every t, define distribution Q_t over the latent variable c_t as:

$$Q_t^{(i)}(c_t) \propto \text{PDF}(x_t; \gamma_{c_t}^{(i-1)}) \cdot \pi^{(i-1)}[c_t]$$

(M step) For every $k \in \{1, \ldots, K\}$

$$\pi_k^{(i)} = \frac{\sum_{t=1}^{n} Q_t^{(i)}[k]}{n}, \quad \gamma_k^{(i)} = \operatorname*{argmin}_{\gamma} \sum_{t=1}^{n} Q_t[k] \log(\text{PDF}(x_t; \gamma))$$

- x_t observation, c_t latent variable.