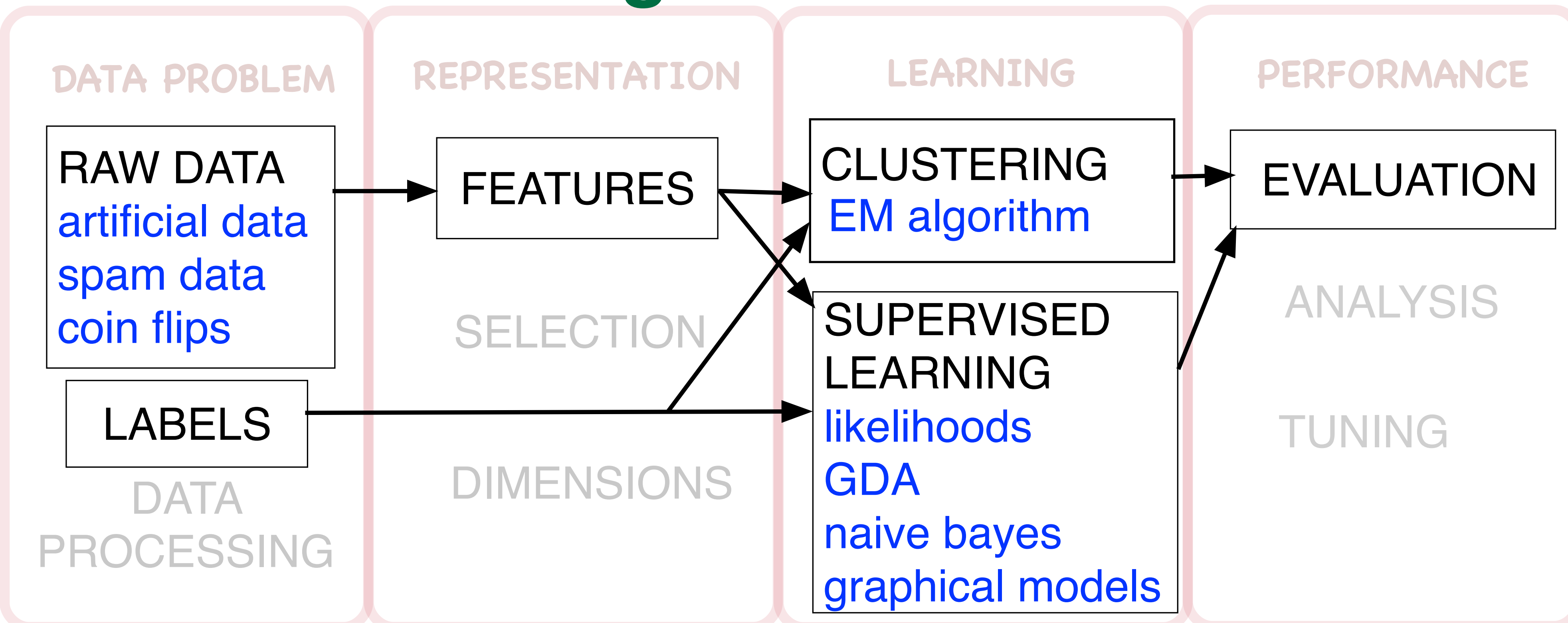
Generative Models

Module 3 Objectives

module 3: generative methods



- Recap Probabilities
  - Distributions, Expectations, Bias, Variance. Conditional/Joint Probabilities
- Naive Bayes
- Gaussian Discriminant Analysis
- Maximum Likelihood Estimation
- Density Estimation for Matrix data
- EM algorithm for mixture models

Generative Models – Density Estimation

- Given a datapoint x, estimate probability P(x)
  - how likely is to see datapoint x?
  - count the observed "x"
- Given a datapoint x and a class/label y, estimate the probability P(y|x)
  - how likely is to see a datapoint like x with label y?
  - count the observed "x" with label y
- Lets assume $x=(x^1,x^2,\ldots,x^d)$ features form
  - then $P(y|x) = P(y|x^1,x^2,\ldots,x^d)$
- we can estimate that as a joint (d+1) dimensional distribution from data
  - typically by using a grid/bucket partitioning of the feature space

Density Estimation Problem

- $P(y|x) = P(y|x^1, x^2, \ldots, x^d)$ joint (d+1)-dim distribution
- … actually we cannot estimate this joint
- if each feature has 10 buckets, and we have 100 features (very reasonable assumptions)
- then the joint distribution has 10^{100} cells - impossible

Bayes Rule

- estimating $P(y|x^1,x^2,\ldots,x^d)$ for classification/ prediction purpose is the same as estimating $P(x^1,x^2,\ldots,x^d|y)$ - due to Bayes Rule:
  - $P(y|x^1,x^2,\ldots,x^d) * P(x^1,x^2,\ldots,x^d) = P(x^1,x^2,\ldots,x^d|y) * P(y)$

Bayes Rule

- estimating $P(y|x^1,x^2,\ldots,x^d)$ for classification/ prediction purpose is the same as estimating $P(x^1,x^2,\ldots,x^d|y)$ - due to Bayes Rule:
  - $P(y|x^1,x^2,\ldots,x^d) * P(x^1,x^2,\ldots,x^d) = P(x^1,x^2,\ldots,x^d|y) * P(y)$

not a factor in ranking $P(y|x)$
(same for all y)

Bayes Rule

- estimating $P(y|x^1,x^2,\ldots,x^d)$ for classification/ prediction purpose is the same as estimating $P(x^1,x^2,\ldots,x^d|y)$ - due to Bayes Rule:
  - $P(y|x^1,x^2,\ldots,x^d) * P(x^1,x^2,\ldots,x^d) = P(x^1,x^2,\ldots,x^d|y) * P(y)$

not a factor in ranking $P(y|x)$
(same for all y)

prior
(estimated from
training counts)

how to get around estimating the joint $P(x^1, x^2, \ldots, x^d | y)$?

- OPTION 1 : assume feature independence
- OPTION 2: model/restrict the joint, instead of estimating any possible such joint distribution
- OPTION 3: mix, bend, tweak options 1 and 2

how to get around estimating the joint $P(x^1,x^2,\ldots,x^d|y)$?

- **OPTION 1** : assume feature independence
  - then $P(x^1,x^2,\ldots,x^d|y) = P(x^1|y)*P(x^2|y)*\ldots P(x^d|y)$
  - estimate each feature density, usually easy
  - the independence assumption rarely holds perfectly, but the model kind-of-works if it approx. holds

- it is called **NAIVE BAYES**
  - very easy to implement
  - smoothing often necessary
  - very popular

how to get around estimating the joint $P(x^1,x^2,\ldots,x^d|y)$?

- **OPTION 2**: model/restrict the joint, instead of estimating any possible such joint distribution
  - typically with a well known parametrized form
  - estimate the parameters of the imposed model

- called **Gaussian Discriminant Analysis**
  - when the model imposed is gaussian

- using **Expectation Maximization** algorithm
  - when the model imposed is a mixture of distributions

how to get around estimating the joint $P(x^1,x^2,\ldots,x^d|y)$?

- OPTION 2: model/restrict the joint, instead of estimating any possible such joint distribution
  - fore example with a well known parametrized form
  - such as multi-dim gaussian distribution
  - estimate the parameters of the imposed model
- called **Gaussian Discriminant Analysis** (when the model imposed is gaussian)
  - easy to implement due to math tools facilitating gaussian parameters estimation (mean, covariance)
  - multidim implies "covariance" matrix instead of simple variance
  - doesnt fit data in many cases

how to get around estimating the joint $P(x^1,x^2,\ldots,x^d|y)$?

- OPTION 3: mix, bend, tweak options 1 and 2
  - don't fully factorize by independence like Naive bayes, instead group dependent features into factors
  - $P(x^1,x^2,\ldots,x^d|y) = P(x^1|y)*P(x^2,x^3|y)*\ldots P(x^4|y)*P(x^3,x^5|y)\ldots$
  - estimate for each factor joint using modeling or bucketing or brute force, depending on the size and nature of the factor
- called **BAYESIAN NETWORK**
  - **also "GRAPHICAL MODEL" or "FACTOR GRAPH"**
  - graph that models only some dependencies as conditional probabilities

Maximum Likelihood Parameter Estimation

- suppose $P(x|y,\theta)$ is modeled by θ parameters
  - for example θ can be mean, variance, covariance, mixture parameters etc - all that defines the probability density function $P(x|y,\theta)$
- data Likelihood and log likelihood
  - how "probable" is to observe the training set given parameters θ ?

$$L = \prod_{i=1}^{m} P(x_i, y_i|\theta) = \prod_{i=1}^{m} P(x_i|y_i, \theta)P(y_i|\theta)$$

$$\log L = \log \prod_{i=1}^{m} P(x_i, y_i|\theta) = \sum_{i=1}^{m} \log P(x_i|y_i, \theta)P(y_i|\theta)$$

- maximize logL as function of θ : solution θ_{ML} is the θ that maximizes the data likelihood

Maximum Likelihood Parameter Estimation

$$L = \prod_{i=1}^{m} P(x_i, y_i|\theta) = \prod_{i=1}^{m} P(x_i|y_i, \theta)P(y_i|\theta)$$

$$\log L = \log \prod_{i=1}^{m} P(x_i, y_i|\theta) = \sum_{i=1}^{m} \log P(x_i|y_i, \theta)P(y_i|\theta)$$

- maximize logL as function of θ : solution θ_{ML} is the θ that maximizes the data likelihood

- if the model used is math-nice, θ_{ML} can be computed in closed form
  - example for Gaussian models

Use model on the test data

- Learned model is encoded by params θ which give $P(x|y, \theta)$

- Equivalently model dictates $P(y|x,\theta)$
  - using Bayes Rule

- On each test datapoint x compute $P(y|x,\theta)$ for all y, and predict the y with highest chance.