Machine Learning
4771
Instructor: Tony Jebara
Topic 10

- Classification with Gaussians
- Regression with Gaussians
- Principal Components Analysis
Classification with Gaussians

• Have two classes, each with their own Gaussian:

\[
\{(x_1, y_1), \ldots, (x_N, y_N)\} \quad x \in \mathbb{R}^D \quad y \in \{0, 1\}
\]

• Given parameters \( \theta = \{\alpha, \mu_0, \Sigma_0, \mu_1, \Sigma_1\} \) we can generate iid data from

\[
p(x, y | \theta) = p(y | \theta) p(x | y, \theta)
\]

by:

1) flipping a coin to get \( y \) via Bernoulli \( \quad p(y | \theta) = \alpha^y (1 - \alpha)^{1-y} \)

2) sampling an \( x \) from \( y \)'th Gaussian \( \quad p(x | y, \theta) = \mathcal{N}(x | \mu_y, \Sigma_y) \)

• Or, recover parameters from data using maximum likelihood

\[
l(\theta) = \log p(\text{data} | \theta) = \sum_{i=1}^{N} \log p(x_i, y_i | \theta)
\]

\[
= \sum_{i=1}^{N} \log p(y_i | \theta) + \sum_{i=1}^{N} \log p(x_i | y_i, \theta)
\]

\[
= \sum_{i=1}^{N} \log p(y_i | \alpha) + \sum_{y_i \in 0} \log p(x_i | \mu_0, \Sigma_0) + \sum_{y_i \in 1} \log p(x_i | \mu_1, \Sigma_1)
\]
Classification with Gaussians

• Max Likelihood can be done separately for the 3 terms
  \[ l = \sum_{i=1}^{N} \log p(y_i | \alpha) + \sum_{y_i \in 0} \log p(x_i | \mu_0, \Sigma_0) + \sum_{y_i \in 1} \log p(x_i | \mu_1, \Sigma_1) \]

• Count # of pos & neg examples (class prior): \( \alpha = \frac{N_1}{N_0 + N_1} \)
• Get mean & cov of negatives and mean & cov of positives:
  \[ \mu_0 = \frac{1}{N_0} \sum_{y_i \in 0} x_i \quad \Sigma_0 = \frac{1}{N_0} \sum_{y_i \in 0} (x_i - \mu_0)(x_i - \mu_0)^T \]
  \[ \mu_1 = \frac{1}{N_1} \sum_{y_i \in 1} x_i \quad \Sigma_1 = \frac{1}{N_1} \sum_{y_i \in 1} (x_i - \mu_1)(x_i - \mu_1)^T \]
• Given (x,y) pair, can now compute likelihood \( p(x, y) \)
• To make classification, a bit of Decision Theory
• Without x, can compute prior guess for y \( p(y) \)
• Give me x, want y, I need posterior \( p(y | x) \)
• Bayes Optimal Decision: \( \hat{y} = \arg \max_{y=\{0,1\}} p(y | x) \)
• Optimal iff we have true probability
Posterior gives Logistic

• Bayes Optimal Decision: \( \hat{y} = \arg \max_{y=\{0,1\}} p(y \mid x) \)

• To get conditional:

\[
p(y \mid x) = \frac{p(x, y)}{p(x)} = \frac{p(x, y)}{\sum_y p(x, y)} = \frac{p(x, y)}{p(x, y = 0) + p(x, y = 1)}
\]

• Check which is greater:

\[
p(y = 0 \mid x) \geq ? \leq p(y = 1 \mid x)
\]

• Or check if this is > 0.5

\[
p(y = 1 \mid x) = \frac{p(x, y = 1)}{p(x, y = 0) + p(x, y = 1)}
\]

\[
= \frac{1}{\frac{p(x, y = 0)}{p(x, y = 1)} + 1}
\]

\[
= \frac{1}{\exp \left( - \log \frac{p(x, y = 1)}{p(x, y = 0)} \right) + 1}
\]

\[
= \text{sigmoid} \left( \log \frac{p(x, y = 1)}{p(x, y = 0)} \right)
\]

• Get logistic squashing function of log-ratio of probability models
Linear or Quadratic Decisions

- Example cases, plotting decision boundary when $\alpha = 0.5$

\[
p(y = 1 \mid x) = \frac{p(x, y = 1)}{p(x, y = 0) + p(x, y = 1)}
= \frac{\alpha N(x \mid \mu_1, \Sigma_1)}{(1-\alpha) N(x \mid \mu_0, \Sigma_0) + \alpha N(x \mid \mu_1, \Sigma_1)}
\]

- If covariances are equal: linear decision
- If covariances are different: quadratic decision
Regression with Gaussians

• Have input and output, each Gaussian:
  \[ \{(x_1, y_1), \ldots, (x_N, y_N)\} \quad x \in \mathbb{R}^D, y \in \mathbb{R}^{D_y} \]

  concatenate \( z_i = \begin{bmatrix} x_i \\ y_i \end{bmatrix} \)

  \[
p(z \mid \mu, \Sigma) = \frac{1}{(2\pi)^{D/2} \sqrt{\det \Sigma}} \exp \left( -\frac{1}{2} (z - \mu)^T \Sigma^{-1} (z - \mu) \right)
  \]

• Maximum Likelihood is as usual for a multivariate Gaussian
  \[
  \mu = \frac{1}{N} \sum_{i=1}^{N} z_i \quad \Sigma = \frac{1}{N} \sum_{i=1}^{N} (z_i - \mu)(z_i - \mu)^T
  \]

• Bayes optimal decision:
  \[
  \hat{y} = \arg \max_{y \in \mathbb{R}} p(y \mid x)
  \]

• Or we can use:
  \[
  \hat{y} = E_{p(y \mid x)} \left\{ y \right\}
  \]

• Have joint, need conditional:
  \[
  p(y \mid x) = \frac{p(x, y)}{p(x)} = \frac{p(x, y)}{\int_y p(x, y)}
  \]
Gaussian Marginals/Conditionals

- Conditional & marginal from joint: \[ p(y \mid x) = \frac{p(x, y)}{p(x)} = \frac{\int_y p(x, y)}{p(x)} \]

- Conditioning the Gaussian:
  \[ p(z \mid \mu, \Sigma) = \frac{1}{(2\pi)^{D/2} \sqrt{\det(\Sigma)}} \exp\left(-\frac{1}{2}(z - \mu)^T \Sigma^{-1}(z - \mu)\right) \]
  \[ p(x, y) = \frac{1}{(2\pi)^{D/2} \sqrt{\det(\Sigma_{xy})}} \exp\left(-\frac{1}{2}\begin{bmatrix} x \\ y \end{bmatrix} - \begin{bmatrix} \mu_x \\ \mu_y \end{bmatrix}^T \begin{bmatrix} \Sigma_{xx} & \Sigma_{xy} \\ \Sigma_{yx} & \Sigma_{yy} \end{bmatrix}^{-1} \begin{bmatrix} x \\ y \end{bmatrix} - \begin{bmatrix} \mu_x \\ \mu_y \end{bmatrix} \right) \]
  \[ p(x) = \frac{1}{(2\pi)^{D/2} \sqrt{\det(\Sigma_{xx})}} \exp\left(-\frac{1}{2}(x - \mu_x)^T \Sigma_{xx}^{-1}(x - \mu_x)\right) \]
  \[ = N(x \mid \mu_x, \Sigma_{xx}) \]
  \[ p(y \mid x) = N(y \mid \mu_y + \Sigma_{yx} \Sigma_{xx}^{-1}(x - \mu_x), \Sigma_{yy} - \Sigma_{yx} \Sigma_{xx}^{-1} \Sigma_{xy}) \]

- Here argmax is expectation which is conditional mean: \[ \hat{y} = \mu_y + \Sigma_{yx} \Sigma_{xx}^{-1}(x - \mu_x) \]
Principal Components Analysis

- Gaussians: for Classification, Regression... & Compression!
- Data can be constant in some directions, changes in others
- Use Gaussian to find directions of high/low variance
Principal Components Analysis

• Idea: instead of writing data in all its dimensions, only write it as mean + steps along one direction

\[
\begin{bmatrix} x_i \\ y_i \end{bmatrix} \approx \begin{bmatrix} \mu_x \\ \mu_y \end{bmatrix} + c_i \begin{bmatrix} v_x \\ v_y \end{bmatrix}
\]

• More generally, keep a subset of dimensions C from D (i.e. 2 of 3)

\[
\tilde{x}_i \approx \tilde{\mu} + \sum_{j=1}^{C} c_{ij} \tilde{v}_j
\]

• Compression method: \( \tilde{x}_i \gg \tilde{c}_i \)

• Optimal directions: along eigenvectors of covariance

• Which directions to keep: highest eigenvalues (variances)
Principal Components Analysis

• If we have eigenvectors, mean and coefficients:
  \[ \tilde{x}_i \approx \bar{\mu} + \sum_{j=1}^{C} c_{ij} \tilde{v}_j \]

• Get eigenvectors (use eig() in Matlab):
  \[ \Sigma = V \Lambda V^T \]

• Eigenvectors are orthonormal:
  \[ \tilde{v}_i^T \tilde{v}_j = \delta_{ij} \]

• In coordinates of \( v \), Gaussian is diagonal, cov = \( \Lambda \)

• All eigenvalues are non-negative \( \lambda_i \geq 0 \)

• Higher eigenvalues are higher variance, use the top \( C \) ones
  \[ \lambda_1 \geq \lambda_2 \geq \lambda_3 \geq \lambda_4 \geq \ldots \]

• To compute the coefficients:
  \[ c_{ij} = (\tilde{x}_i - \bar{\mu})^T \tilde{v}_j \]
Eigenfaces

\[ \{ x_1, \ldots, x_N \} \]

\[
\hat{x}_1 = \mu + \sum_{j=1}^{C} c_{1j} \vec{v}_j, \ldots, \hat{x}_N = \mu + \sum_{j=1}^{C} c_{Nj} \vec{v}_j
\]

\[
c_{ij} = (\vec{x}_i - \bar{\mu})^T \vec{v}_j
\]

Encode

Decode