CS-667 Advanced Machine Learning

Nazar Khan

PUCIT

Conditional Mixture Models

Combining Models

- So far, we have seen how to model complicated machine learning problems by combining simpler models.
  - Boosting for learning a strong classifier by sequentially learning weak classifiers.
  - Gaussian mixture models for modelling (unconditional) density $p(\mathbf{x})$.
- In this lecture, we will *model conditional density $p(t|\mathbf{x})$ by combining simpler models*
  - For continuous t, we obtain a *mixture of linear regression models*.
  - For discrete t, we obtain a *mixture of logistic regression models*.
- As before, parameter learning for mixture models will be achieved by the EM algorithm.

Linear Regression Recap

- We have already covered the probabilistic perspective of linear regression in our polynomial fitting example.
- We assumed that target t was given by a deterministic function $y(\mathbf{x}, \mathbf{w})$ with additive Gaussian noise. That is

$$t = y(\mathbf{x}, \mathbf{w}) + \epsilon$$

  where $\epsilon \sim \mathcal{N}(0, \beta^{-1})$.
- Therefore, we wrote the conditional density of the target as

$$p(t|\mathbf{x}, \mathbf{w}, \beta) = \mathcal{N}(t|y(\mathbf{x}, \mathbf{w}), \beta^{-1})$$

Linear Regression Recap

- Likelihood for i.i.d data $\{(\mathbf{x}_1, t_1), \ldots, (\mathbf{x}_N, t_N)\}$ can be written as

$$\prod_{n=1}^{N} \mathcal{N}(t_n | \mathbf{w}^T \phi(\mathbf{x}_n), \beta^{-1})$$

- Log-likelihood becomes

$$\frac{N}{2} \ln \beta - \frac{N}{2} \ln(2\pi) - \beta \underbrace{\frac{1}{2} \sum_{n=1}^{N} \{t_n - \mathbf{w}^T \phi(\mathbf{x_n})\}^2}_{\text{SSE}}$$

- Therefore, maximisation of log-likelihood with respect to $\mathbf{w}$ is equivalent to minimisation of SSE function.

Linear Regression Recap

- Gradient with respect to $\mathbf{w}$ is $\sum_{n=1}^{N}\{t_n - \mathbf{w}^T\phi(\mathbf{x_n})\}\phi(\mathbf{x_n})^T$.
- Equating gradient to the $\mathbf{0}$ vector

$$\mathbf{0} = \sum_{n=1}^{N} t_n \phi(\mathbf{x_n})^T - \mathbf{w}_{\mathrm{ML}}^T \left(\sum_{n=1}^{N} \phi(\mathbf{x_n})\phi(\mathbf{x_n})^T \right)$$

- By converting to a pure matrix-vector notation, we found the maximum-likelihood solution for linear regression as

$$\mathbf{w}_{\mathrm{ML}} = \underbrace{(\mathbf{\Phi}^T\mathbf{\Phi})^{-1}\mathbf{\Phi}^T}_{\mathbf{\Phi}^\dagger}\mathbf{t}$$

where $\mathbf{\Phi}$ denoted the design matrix and $\mathbf{t}$ denoted the vector of all N target values.

EM Algorithm Recap

Goal is to maximise likelihood $p(\mathbf{X}|\boldsymbol{\theta})$ with respect to $\boldsymbol{\theta}$ by introducing joint distribution $p(\mathbf{X}, \mathbf{Z}|\boldsymbol{\theta})$ involving latent variables $\mathbf{Z}$.

1. Choose initial $\boldsymbol{\theta}^{\text{old}}$
2. **E-step**: Evaluate $p(\mathbf{Z}|\mathbf{X}, \boldsymbol{\theta}^{\text{old}})$
3. **M-step**: Obtain new estimate $\boldsymbol{\theta}^{\text{new}}$ by maximising the expectation $\mathcal{Q}(\boldsymbol{\theta}, \boldsymbol{\theta}^{\text{old}})$

$$\boldsymbol{\theta}^{\text{new}} = \arg\max_{\boldsymbol{\theta}} \mathcal{Q}(\boldsymbol{\theta}, \boldsymbol{\theta}^{\text{old}})$$

   where $\mathcal{Q}(\boldsymbol{\theta}, \boldsymbol{\theta}^{\text{old}}) = \sum_{\mathbf{Z}} p(\mathbf{Z}|\mathbf{X}, \boldsymbol{\theta}^{\text{old}}) \ln p(\mathbf{X}, \mathbf{Z}|\boldsymbol{\theta})$.
4. Check for convergence of either log-likelihood or parameters. If not converged, then

$$\boldsymbol{\theta}^{\text{old}} \leftarrow \boldsymbol{\theta}^{\text{new}}$$

   and return to step 2.

Mixture of linear regression models

- Fitting a single linear function to data being generated from multiple sources gives very poor results.
- Instead of a single Gaussian, we now model the conditional density of targets t_n by a mixture of K Gaussians

$$p(t_n|\mathbf{x}_n, \boldsymbol{\theta}) = \sum_{k=1}^{K} \pi_k \mathcal{N}_k(t_n|y(\mathbf{x}_n, \mathbf{w}_k), \beta_k^{-1})$$

  where $\mathbf{w}_k$ are the parameters of the linear function representing the mean of the k-th Gaussian component and β_k^{-1} is the precision of that component.
- Notice that this is a mixture of *conditional* Gaussians $p(t|\mathbf{x})$ and therefore different from the standard Gaussian Mixture Model which uses marginal densities $p(x)$.

Mixture of linear regression models

- For i.i.d data $(\mathbf{X}, \mathbf{t})$, log-likelihood can therefore be written as

$$\begin{aligned}\ln p(\mathbf{t}|\mathbf{X}, \boldsymbol{\theta}) &= \ln \left(\prod_{n=1}^{N} p(t_n|\mathbf{x}_n, \boldsymbol{\theta}) \right) \\ &= \sum_{n=1}^{N} \ln \left(\sum_{k=1}^{K} \pi_k \mathcal{N}_k(t_n|y(\mathbf{x}_n, \mathbf{w}_k), \beta_k^{-1}) \right)\end{aligned}$$

  where the summation over k 'blocks'the natural logarithm from acting on the exponential function.
- As in the case of GMMs, we maximise log-likelihood by the EM algorithm.
- For that we need to view the problem in terms of latent variables.

Mixture of linear regression models

- Let $z_{nk} = 1$ imply that training data point n was generated by the k-th source (model component).
- Then for every observed t_n there is a corresponding K-dimensional vector $\mathbf{z}_n$ with 1-of-K coding.
- Since we do not know how the data was generated, the $\mathbf{z}_n$ are unobserved/hidden/latent variables.

Mixture of linear regression models

- Log-likelihood for *complete data* (observed + unobserved) can be written as

$$\begin{aligned}\ln p(\mathbf{t}, \mathbf{Z}|\mathbf{X}, \boldsymbol{\theta}) &= \ln\left(\prod_{n=1}^{N} p(t_n, \mathbf{z}_n|\mathbf{x}_n, \boldsymbol{\theta})\right)\\ &= \ln\left(\prod_{n=1}^{N}\prod_{k=1}^{K}\{\pi_k \mathcal{N}_k(t_n|y(\mathbf{x}_n, \mathbf{w}_k), \beta_k^{-1})\}^{z_{nk}}\right)\\ &= \sum_{n=1}^{N}\sum_{k=1}^{K} z_{nk} \ln\left(\pi_k \mathcal{N}_k(t_n|y(\mathbf{x}_n, \mathbf{w}_k), \beta_k^{-1})\right)\end{aligned}$$

which is not computable since we do not know the values of the latent variables $\mathbf{Z}$.

Mixture of linear regression models

- However, *expected* log-likelihood for complete data is computable *if* model parameters $\boldsymbol{\theta}$ are known. This is the *E-step*.

$$
\begin{aligned}
\mathcal{Q}(\boldsymbol{\theta}, \boldsymbol{\theta}^{\text{old}}) &= \mathbb{E}_{\mathbf{Z}|\mathbf{t}}[\ln p(\mathbf{t}, \mathbf{Z}|\mathbf{X}, \boldsymbol{\theta})] \\
&= \sum_{n=1}^{N} \sum_{\mathbf{z}_n} \sum_{k=1}^{K} p(z_{nk}|t_n) z_{nk} \ln \left(\pi_k \mathcal{N}_k(t_n|y(\mathbf{x}_n, \mathbf{w}_k), \beta_k^{-1})\right) \\
&= \sum_{n=1}^{N} \sum_{k=1}^{K} \underbrace{p(z_{nk} = 1|t_n)}_{r_{nk}} \ln \left(\pi_k \mathcal{N}_k(t_n|y(\mathbf{x}_n, \mathbf{w}_k), \beta_k^{-1})\right)
\end{aligned}
$$

where responsibilities r_{nk} are computed using Bayes' theorem

$$
r_{nk} = \frac{p(z_{nk} = 1)p(t_n|z_{nk} = 1)}{\sum_{j=1}^{K} p(z_{nj} = 1)p(t_n|z_{nj} = 1)} = \frac{\pi_k \mathcal{N}_k(t_n|y(\mathbf{x}_n, \mathbf{w}_k), \beta_k^{-1})}{\sum_{j=1}^{K} \pi_j \mathcal{N}_j(t_n|y(\mathbf{x}_n, \mathbf{w}_j), \beta_j^{-1})}
$$

Mixture of linear regression models

- In the *M-step*, we fix the responsibilities and update the parameters $\boldsymbol{\theta} = \{\pi_k, \mathbf{w}_k, \beta_k\}$.
- Since the mixing coefficients π_k represent probabilities, they are optimised for via Lagrange multipliers to yield

$$\pi_k^* = \frac{N_k}{N} = \frac{\sum_{n=1}^{N} r_{nk}}{N}$$

- Optimal regression weights $\mathbf{w}_k$ are obtained as the solution to a weighted least-squares problem

$$\mathbf{w}_k^* = \left(\boldsymbol{\Phi}^T \mathbf{R}_k \boldsymbol{\Phi}\right)^{-1} \boldsymbol{\Phi}^T \mathbf{R}_k \mathbf{t}$$

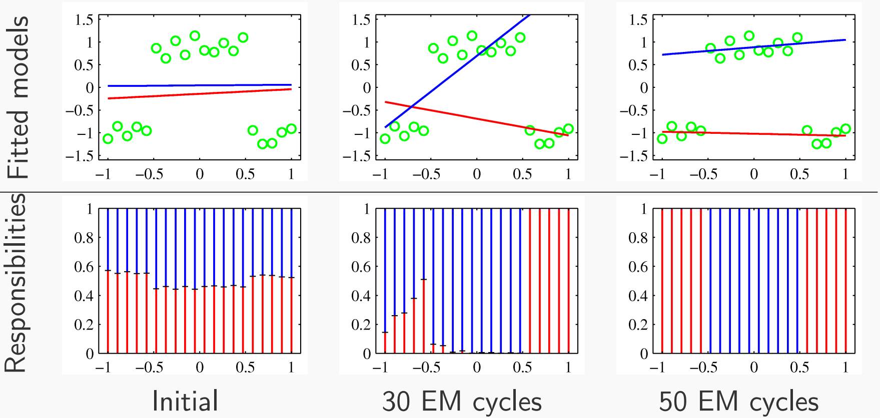
  where $\mathbf{R}_k = \mathrm{diag}(r_{nk})$ is an $N \times N$ diagonal matrix of weights that is recomputed at each E-step.

Mixture of linear regression models

- Finally, optimal precision β_k is obtained as

$$\frac{1}{\beta_k^*} = \frac{\sum_{n=1}^{N} r_{nk}(t_n - \mathbf{w}_k^{*T}\phi_n)^2}{\sum_{n=1}^{N} r_{nk}} = \frac{1}{N_k}\sum_{n=1}^{N} r_{nk}(t_n - \mathbf{w}_k^{*T}\phi_n)^2$$

Mixture of linear regression models



EM Algorithm for Mixutre of Linear Regression Models

Data: Data points $\{(\mathbf{x}_1, t_1), \dots, (\mathbf{x}_N, t_N)\}$, integer $K > 1$.
Result: Component parameters $\{\mathbf{w}_k, \beta_k\}$, mixing coefficients $\{\pi_k\}$

1. Choose some initial values for $\mathbf{w}_k, \beta_k, \pi_k$
2. Fix parameters, update responsibilities
   $r_{nk} = \frac{\pi_k \mathcal{N}_k(t_n | y(\mathbf{x}_n, \mathbf{w}_k), \beta_k^{-1})}{\sum_{j=1}^{K} \pi_j \mathcal{N}_j(t_n | y(\mathbf{x}_n, \mathbf{w}_j), \beta_j^{-1})}$
3. Fix responsibilities, update parameters

$$\pi_k = \frac{N_k}{N}$$
$$\mathbf{w}_k = \left(\boldsymbol{\Phi}^T \mathbf{R}_k \boldsymbol{\Phi}\right)^{-1} \boldsymbol{\Phi}^T \mathbf{R}_k \mathbf{t}$$
$$\beta_k = \frac{N_k}{\sum_{n=1}^{N} r_{nk}(t_n - \mathbf{w}_k^T \phi_n)^2}$$

where $N_k = \sum_{n=1}^{N} r_{nk}$ and $\mathbf{R}_k = \mathrm{diag}(r_{nk})$.

EM Algorithm for Mixutre of Linear Regression Models

4. Evaluate log-likelihood

$$\ln p(\mathbf{t}|\mathbf{X}, \boldsymbol{\theta}) = \sum_{n=1}^{N} \ln \left(\sum_{k=1}^{K} \pi_k \mathcal{N}_k(t_n | y(\mathbf{x}_n, \mathbf{w}_k), \beta_k^{-1}) \right)$$

and check for convergence of either log-likelihood or parameters. If not converged, return to step 2.

Mixture of logistic regression models

- For binary classification problems, we studied logistic regression which outputs posterior probabilities $p(t|\mathbf{x})$ for $t = \{0, 1\}$.
- This allows us to use logistic regression as a component of more complicated probabilistic models.
- A mixture of K logistic regression models can be constructed as

$$p(t|\phi, \boldsymbol{\theta}) = \sum_{k=1}^{K} \pi_k y_k^t (1 - y_k)^{1-t}$$

where $y_k = \sigma(\mathbf{w}_k^T \phi)$ is the output of component k and the adjustable parameters are $\boldsymbol{\theta} = \{\pi_k, \mathbf{w}_k\}$.

- Can be extended to multiclass problems as *mixture of softmax models*.

Mixture of logistic regression models

- Given i.i.d. data $\{\phi_n, t_n\}$, *incomplete data log-likelihood* can be written as

$$p(\mathbf{t}|\boldsymbol{\theta}) = \prod_{n=1}^{N} \sum_{k=1}^{K} \pi_k y_{nk}^{t_n} (1 - y_{nk})^{1-t_n}$$

where $y_{nk} = \sigma(\mathbf{w}_k^T \phi_n)$.

- By employing latent variables z_{nk} with 1-of-K coding, we can write the *complete data likelihood* as

$$p(\mathbf{t}|\boldsymbol{\theta}) = \prod_{n=1}^{N} \prod_{k=1}^{K} \{\pi_k y_{nk}^{t_n} (1 - y_{nk})^{1-t_n}\}^{z_{nk}}$$

and then use EM for parameter learning.

Mixture of logistic regression models

- E-step: Compute responsibilities

$$
\begin{aligned}
r_{nk} &= p(z_{nk} = 1|t_n) \\
&= \frac{p(z_{nk} = 1)p(t_n|z_{nk} = 1)}{\sum_{j=1}^{K} p(z_{nj} = 1)p(t_n|z_{nj} = 1)} = \frac{\pi_k y_{nk}^{t_n}(1 - y_{nk})^{1-t_n}}{\sum_{j=1}^{K} \pi_j y_{nj}^{t_n}(1 - y_{nj})^{1-t_n}}
\end{aligned}
$$

- Allows us to write the expected complete data log-likelihood

$$
\begin{aligned}
\mathcal{Q}(\boldsymbol{\theta}, \boldsymbol{\theta}^{\text{old}}) &= \mathbb{E}_{\mathbf{Z}|\mathbf{t}}[\ln p(\mathbf{t}, \mathbf{Z}|\mathbf{X}, \boldsymbol{\theta})] \\
&= \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} \{\ln \pi_k + t_n \ln y_{nk} + (1 - t_n) \ln(1 - y_{nk})\}
\end{aligned}
$$

Mixture of logistic regression models

- M-step: Maximise $\mathcal{Q}(\boldsymbol{\theta}, \boldsymbol{\theta}^{\text{old}})$ with respect to π_k via Lagrange multipliers to obtain

$$\pi_k^* = \frac{N_k}{N} = \frac{\sum_{n=1}^N r_{nk}}{N}$$

- Find optimal classifier weights $\mathbf{w}_k^*$ via IRLS which requires computation of the gradient vector

$$\nabla_{\mathbf{w}_k} \mathcal{Q} = \sum_{n=1}^N r_{nk}(t_n - y_{nk})\phi_n$$

and the Hessian matrix

$$\nabla_{\mathbf{w}_k} \nabla_{\mathbf{w}_k} \mathcal{Q} = -\sum_{n=1}^N r_{nk} y_{nk}(1 - y_{nk})\phi_n \phi_n^T$$

Mixture of Experts

- By allowing the mixing coefficients to depend on the input, we can obtain an even more powerful class of mixture models.

$$p(t|\mathbf{x},\boldsymbol{\theta}) = \sum_{k=1}^{K} \pi_k(\mathbf{x}) p_k(t|\mathbf{x},\boldsymbol{\theta})$$

- The input-dependent mixing coefficients $\pi_k(\mathbf{x})$ are known as *gating functions*.
- The individual component densities $p_k(t|\mathbf{x},\boldsymbol{\theta})$ are known as the *experts*.
- Gating functions $\pi_k(\mathbf{x})$ determine **which model is how much of an expert in which region of input space**.

Mixture of Experts

- One choice of gating functions is the linear softmax

$$\pi_k(\mathbf{x}) = \frac{e^{\mathbf{v}_k^T \mathbf{x}}}{\sum_{j=1}^{K} e^{\mathbf{v}_j^T \mathbf{x}}}$$

- Parameters $\boldsymbol{\theta}$ now include the linear softmax weights $\{\mathbf{v}_k\}$.
- If the experts are also linear, learning can be performed using the EM algorithm.