GLMs (wrap-up) and Generative Models for Supervised Learning

CS772A: Probabilistic Machine Learning
Piyush Rai
Plan for today and announcements

- Wrap up GLM
- Testing conditional independence in directed graphical models
- Generative models for supervised learning
- Also, Quiz 1 on Monday (Sept 5) at 7pm
- Rescheduled class on Saturday (Sept 3) at 6pm
- Mid-sem exam on Sept 19 in L17 (18:00-20:00), ERES seating arrangement
GLM with Canonical Response Function

- For GLM with Canon Resp Func (a.k.a., canonical GLM)
  \[ p(y|\eta) = h(y) \exp(\eta y - A(\eta)) = h(y) \exp(y w^T x - A(\eta)) \]

- Consider doing MLE (assuming \( N \) i.i.d. responses). The log likelihood
  \[ L(\eta) = \log p(Y|\eta) = \log \prod_{n=1}^{N} h(y_n) \exp(y_n w^T x_n - A(\eta_n)) = \sum_{n=1}^{N} \log h(y_n) + w^T \sum_{n=1}^{N} y_n x_n - \sum_{n=1}^{N} A(\eta_n) \]

- Convexity of \( A(\eta) \) guarantees a global optima. Gradient of log-lik w.r.t. \( w \)
  \[ g = \sum_{n=1}^{N} \left( y_n x_n - A'(\eta_n) \frac{d\eta_n}{dw} \right) = \sum_{n=1}^{N} (y_n x_n - \mu_n x_n) = \sum_{n=1}^{N} (y_n - \mu_n) x_n \]

- Note \( \mu_n = f(\xi_n) = f(w^T x_n) \) and \( f = \psi^{-1} \) (“inverse link”) depends on the model
  - Real-valued \( y \) (linear regression): \( f \) is identity, i.e., \( \mu_n = w^T x_n \)
  - Binary \( y \) (logistic regression): \( f \) is sigmoid function, i.e., \( \mu_n = \frac{\exp(w^T x_n)}{1+\exp(w^T x_n)} \)
  - Count-valued \( y \) (Poisson regression): \( f \) is exp, i.e., \( \mu_n = \exp(w^T x_n) \)
  - Non-negative \( y \) (gamma regression): \( f \) is inverse negative i.e., \( \mu_n = -1/(w^T x_n) \)

The simple form of canonical GLM (nat. param just a linear function \( w^T x \)) makes parameter estimation via MLE/MAP easy since gradient and Hessian have simple expressions (though the Hessian may be expensive to compute/invert)
Fully Bayesian Inference for GLMs

- Most GLMs, except linear regression with Gaussian lik. and Gaussian prior, do not have conjugate pairs of likelihood and priors (recall logistic regression)

- Posterior over the weight vector $\mathbf{w}$ is intractable

- Approximate inference methods needed, e.g.,
  - Laplace approximation (have already seen): Easily applicable since derivatives (first and second) can be easily computed (note that we need $\mathbf{w}_{MAP}$ and Hessian)
  - MCMC or variational inference (will see later)
## Various Types of GLMs

<table>
<thead>
<tr>
<th>Type of response</th>
<th>Type of GLM</th>
<th>Link Function $\Psi$</th>
<th>Response Function $f$ (Inv Link Func if canon. GLM) (Operates on $\xi = w^T x$)</th>
<th>Activation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Real</td>
<td>Gaussian</td>
<td>Identity</td>
<td>Identity</td>
<td>Linear</td>
</tr>
<tr>
<td>Binary</td>
<td>Logistic</td>
<td>Log-odds: $\log \frac{\mu}{1-\mu}$</td>
<td>Sigmoid</td>
<td>Sigmoid</td>
</tr>
<tr>
<td>Binary</td>
<td>Probit</td>
<td>Inv CDF: $\Phi^{-1}(\mu)$</td>
<td>$\Phi$ (CDF of N(0,1))</td>
<td>Probit</td>
</tr>
<tr>
<td>Categorical</td>
<td>Multinoulli</td>
<td>Log-odds: $\log \frac{\mu_k}{1-\mu_k}$</td>
<td>Softmax</td>
<td>Softmax</td>
</tr>
<tr>
<td>Count</td>
<td>Poisson</td>
<td>$\log \mu$</td>
<td>exp</td>
<td></td>
</tr>
<tr>
<td>Non-negative real</td>
<td>gamma</td>
<td>Negative of inverse</td>
<td>Negative of inverse</td>
<td></td>
</tr>
<tr>
<td>Binary</td>
<td>Gumbel</td>
<td>Gumbel Inv CDF: $\log(-\log())$</td>
<td>Gumbel CDF: $\exp(-\exp(-))$</td>
<td></td>
</tr>
</tbody>
</table>

.. and several others (exponential, inverse Gaussian, Binomial, Tweedie, etc)
(Directed) Graphical Models

- Most models we study can be represented via directed graphical models (DGMs)
- A DGM is a graph with nodes denoting random variables and edges their dependences

\[
p(x_1, x_2, x_3, x_4, x_5, x_6, x_7) = p(x_1)p(x_2)p(x_3)p(x_4|x_2, x_3)p(x_5|x_1, x_3)p(x_6|x_4)p(x_7|x_4, x_5)
\]

- Plate notation is a compact way of representing DGMs, e.g.,

\[
x = (x_1, x_2, \ldots, x_K)
\]

\[
p(x) = \prod_{i=1}^{K} p(x_i|\text{pa}(x_i))
\]

Also remember: shaded = observed, unshaded = unknown
Goal: Test if two nodes $a$ and $b$ are independent in presence of a third node $c$

$p(a, b, c) = p(a|c)p(b|c)p(c)$

$p(a, b) = \sum_c p(a|c)p(b|c)p(c) = p(a)p(b)$

$p(a|c)p(b|c)p(c) \neq p(a)p(b)$

Thus $a$ and $b$ not marginally independent

$p(a, b, c) = p(a)p(b, c|a)$

$p(a, b) = p(a) \sum_c p(b, c|a) = p(a)p(b|a)$

Thus $a$ and $b$ not marginally independent

$p(a, b, c) = p(a)p(b)p(c|a, b)$

$p(a, b) = p(a)p(b) \sum_c p(c|a, b) = p(a)p(b)$

Thus $a$ and $b$ are independent

$p(a, b|c) = p(a, b|c)\frac{p(c)}{p(c)} = p(a|c)p(b|c)$

Thus $a$ and $b$ are conditionally independent given $c$

$p(a, b, c) = p(a)p(b)p(c|a, b)$

$p(a, b|c) = \frac{p(a)p(b)p(c|a, b)}{p(c)} = p(a)p(b|c)$

Conditionally NOT independent given $c$
DGMs and Independence

- A node in a DGM is independent of all other nodes given its Markov Blanket.
- Markov Blanket (MB) consists of a node's parents, children, and co-parents.

Whenever we write the conditional/posterior distribution of a node, any node that is not in MB would not appear in the conditioned side.
Generative Supervised Learning

- Want to learn the conditional distribution \( p(y|x) \) for supervised learning (reg/class.)
- Generative approach assumes a model for the joint distribution \( p(x, y) \)
- Assuming a classification setting, we can write

\[
p(y|x) = \frac{p(x, y)}{p(x)} = \frac{p(y)p(x|y)}{p(x)} = \frac{p(y)p(x|y)}{\sum_y p(y)p(x|y)}
\]

- Note that the inputs of each class have a specific distribution \( p(x|y) \)
  - Thus the inputs are also assumed to be “generated” via a process defined by \( p(x|y) \)
- We learn \( p(y) \) and \( p(x|y) \) using training data \( (X, Y) = \{(x_n, y_n)\}_{n=1}^{N} \)
  - To estimate \( p(y = k|\theta) \) or \( p(y|\theta) \), the data is \( \{y_n\}_{n=1}^{N} \) : Bernoulli \( (K = 2) \) or multinoulli \( (K > 2) \)
  - To estimate \( p(x|y = k, \theta) \), the data is all the inputs from class \( k \), i.e., \( X_k = \{x_n : y_n = k\} \)
- These distributions can be estimated using point estimation or using fully Bayesian inference

Have already seen the discriminative approach to learn \( p(y|x) \) (prob linear regression, logistic regression, and also GLM)

Class-prior distribution (a discrete distribution since \( y \) is discrete)

Class-conditional distribution of inputs from class \( y \)

All these distributions here, \( p(y), p(x|y) \) (and consequently \( p(y|x) \) too) depend on some params/hyperparams (not shown for brevity)

Can compute it for each possible value of \( y \in \{1, 2, ..., K\} \)

Note that the inputs of each class have a specific distribution \( p(x|y) \)
Generative Supervised Learning: Classification

- Training: Fit an appropriate probability distribution for data from each class

- Test time: Evaluate under which class, the test input has largest probability $p(x|y = k)$ or largest class posterior probability $p(y = k|x) \propto p(y = k)p(x|y = k)$
Generative Classification: A Generative Story

- Assuming binary labels, can define a “generative story” for each example \((x_i, y_i)\)
  - First draw (“generate”) a binary label \(y_i \in \{0,1\}\)
  - \(y_i \sim \text{Bernoulli}(\pi)\)
  
  - Now draw (“generate”) the input \(x_i\) from the distribution of class \(y_i \in \{0,1\}\)
  
  \[x_i | y_i \sim p(x | \theta_{y_i})\]

- Writing \(\theta = (\theta_0, \theta_1)\), the above generative model shown in “plate notation”

Note that in this generative process, we assume \(y\) is generated first since the generation of \(x\) depends on what \(y\) is.

For multi-class problems, we will have a multinoulli instead.

Most generative models (supervised as well as unsupervised/semi-supervised) can be expressed via such a story.

Order of generation in this story depends on what part of the data/parameters depend on what data/params.

Often we also show the generation of parameters/unknowns as well (via their respective distributions).

A discriminative model (no model for \(x_i\)’s)

Compare this with the plate notation diagram of a discriminative model such as prob linear regression or logistic regression.

Note that in this generative process, we assume \(y\) is generated first since the generation of \(x\) depends on what \(y\) is.
Generative Classification: Learning Procedure

- Recall the generative classification model

\[
p(y = k | x) = \frac{p(y = k)p(x|y = k)}{\sum_k p(y = k)p(x|y = k)}
\]

- We need to learn \( p(y) \) and \( p(x|y) \) here given training data \((X, y) = \{(x_n, y_n)\}_{n=1}^N\)

- Class prior distribution \( p(y) \) will always be a discrete distribution, e.g.,
  - For \( y \in \{0,1\} \), \( p(y) = p(y|\pi) = \text{Bernoulli}(y|\pi) \) with \( \pi \in (0,1) \)
  - For \( y \in \{1,2,\ldots,K\} \), \( p(y) = p(y|\pi) = \text{multinoulli}(y|\pi) \) where \( \pi = [\pi_1, \ldots, \pi_K] \)

- Class conditional distribution \( p(x|y) \) will depend on the nature of inputs, e.g.,
  - For \( x \in \mathbb{R}^D \), \( p(x|y = k) \) can be a multivariate Gaussian (one per class)

\[
p(x|y = k) = p(x|\theta_k) = \mathcal{N}(x|\mu_k, \Sigma_k)
\]

- Can estimate \( \pi \) and \( \{\theta_k\}_{k=1}^K \) using \((X, y)\) via point est. or fully Bayesian infer.
Generative Classification: Making Predictions

- Once $\pi$ and $\{\theta_k\}_{k=1}^K$ are learned, we are ready to make predictions for any test input $x_*$. 
- Two ways to make the prediction
- Approach 1: If we have point estimates for $\pi$ and $\{\theta_k\}_{k=1}^K$, say $\hat{\pi}$ and $\{\hat{\theta}_k\}_{k=1}^K$. Then
  \[
p(y_* = k|x_*) = \frac{p(y_* = k|\hat{\pi})p(x_*|\hat{\theta}_k)}{\sum_k p(y = k|\hat{\pi})p(x|\hat{\theta}_k)} \propto \hat{\pi}_k p(x_*|\hat{\theta}_k)
  \]
- Approach 2: If we have the full posterior for $\pi$ and $\{\theta_k\}_{k=1}^K$. Then
  - Instead of using $p(y_* = k|\hat{\pi})$, we will use $p(y_* = k|\pi) = \int p(y_* = k|\pi)p(\pi|y)d\pi$
  - Instead of using $p(x_*|\hat{\theta}_k)$, we will use $p(x_*|X_k) = \int p(x_*|\theta_k)p(\theta_k|X_k)d\theta_k$
  - Using these quantities, the prediction will be made as
  \[
p(y_* = k|x_*, X, y) = \frac{p(y_* = k|y)p(x_*|X_k)}{\sum_k p(y_* = k|y)p(x_*|X_k)} \propto p(y_* = k|y)p(x_*|X_k)
  \]
Generative Sup. Learning: Some Comments

- A very flexible approach for classification
  - Incorporate info about how frequent each class is in the training data ("class prior")
  - Incorporate info about the shape of each class
  - Consequently, can naturally learn nonlinear boundaries, too (without using kernel methods or deep learning)

\[
p(y^* = k | x^*) = \frac{p(y^* = k)p(x^* | y^* = k)}{\sum_k p(y^* = k)p(x^* | y^* = k)}
\]

- Can handle missing labels and missing features
  - These can be treated as latent variables as estimated using methods such as EM

- Ability to handle missing labels makes it suitable for semi-supervised learning

- The choice of the class-conditional and proper estimation is important
  - Can leverage advances in deep generative models to learn very flexible forms for \( p(x | y) \)

- Can also use it for regression (define \( p(x, y) \) via some distr. and obtain \( p(y | x) \) from it)

- Can also combine generative and discriminative approaches for supervised learning

Will discuss this later
Hybrids of Discriminative and Generative Models

- Both discriminative and generative models have their strengths/shortcomings

  - Some aspects about discriminative models for sup. learning
    - Discriminative models have usually fewer parameters (e.g., just a weight vector)
    - Given “plenty” of training data, disc. models can usually outperform generative models

  - Some aspects about generative models for sup. learning
    - Can be more flexible (we have seen the reasons already)
    - Usually have more parameters to be learned
    - Modeling the inputs (learning $p(x|y)$) can be difficult for high-dim inputs

- Some prior work on combining discriminative and generative models. Examples:
  \[
  \alpha \log p(y|x; \theta) + \beta \log p(x; \theta) \quad \quad \quad p(x, y, \theta_d, \theta_g) = p_{\theta_d}(y|x)p_{\theta_g}(x)p(\theta_d, \theta_g)
  \]
  \[
  p(x, y, z) = p(y|x, z) \cdot p(x, z)
  \]

  - Approach 1 (McCullum et al, 2006) – modeling the joint $p(x, y|\theta)$ using a multi-conditional likelihood
  - Approach 2 (Lasserre et al, 2006) – Coupled parameters between discriminative and generative models
  - Approach 3 (Kuleshov and Ermon, 2017) – Coupling discriminative and generative models via a latent variable $z$ (see "Deep Hybrid Models: Bridging Discriminative and Generative Approaches", UAI 2017)