Bayesian Logistic Regression, Bayesian Generative Classification

Piyush Rai

Topics in Probabilistic Modeling and Inference (CS698X)

Jan 23, 2019
Bayesian Logistic Regression

Recall that the **likelihood model** for logistic regression is Bernoulli (since $y \in \{0, 1\}$)

$$p(y|x, w) = \text{Bernoulli}(\sigma(w^\top x)) = \left[ \frac{\exp(w^\top x)}{1 + \exp(w^\top x)} \right]^y \left[ \frac{1}{1 + \exp(w^\top x)} \right]^{(1-y)} = \mu^y (1 - \mu)^{1-y}$$
Bayesian Logistic Regression

Recall that the likelihood model for logistic regression is Bernoulli (since \( y \in \{0, 1\} \))

\[
p(y|x, w) = \text{Bernoulli}(\sigma(w^\top x)) = \left[ \frac{\exp(w^\top x)}{1 + \exp(w^\top x)} \right]^y \left[ \frac{1}{1 + \exp(w^\top x)} \right]^{1-y} = \mu^y (1 - \mu)^{1-y}
\]

Just like the Bayesian linear regression case, let’s use a Gaussian prior on \( w \)

\[
p(w) = \mathcal{N}(0, \lambda^{-1} I_D) \propto \exp\left(-\frac{\lambda}{2} w^\top w\right)
\]
Bayesian Logistic Regression

- Recall that the likelihood model for logistic regression is Bernoulli (since \( y \in \{0, 1\} \))

\[
p(y|x, w) = \text{Bernoulli}(\sigma(w^\top x)) = \left[ \frac{\exp(w^\top x)}{1 + \exp(w^\top x)} \right]^y \left[ \frac{1}{1 + \exp(w^\top x)} \right]^{(1-y)} = \mu^y (1 - \mu)^{1-y}
\]

- Just like the Bayesian linear regression case, let’s use a Gaussian prior on \( w \)

\[
p(w) = \mathcal{N}(0, \lambda^{-1}I_D) \propto \exp(-\frac{\lambda}{2} w^\top w)
\]

- Given \( N \) observations \((X, y) = \{x_n, y_n\}_{n=1}^N\), where \( X \) is \( N \times D \) and \( y \) is \( N \times 1 \), the posterior over \( w \)

\[
p(w|X, y) = \frac{p(y|X, w)p(w)}{\int p(y|X, w)p(w)dw}
\]
Bayesian Logistic Regression

- Recall that the likelihood model for logistic regression is Bernoulli (since $y \in \{0, 1\}$)

$$p(y|x, w) = \text{Bernoulli}(\sigma(w^\top x)) = \left[ \frac{\exp(w^\top x)}{1 + \exp(w^\top x)} \right]^y \left[ \frac{1}{1 + \exp(w^\top x)} \right]^{(1-y)} = \mu^y (1 - \mu)^{1-y}$$

- Just like the Bayesian linear regression case, let's use a Gaussian prior on $w$

$$p(w) = \mathcal{N}(0, \lambda^{-1}I_D) \propto \exp(-\frac{\lambda}{2}w^\top w)$$

- Given $N$ observations $(X, y) = \{x_n, y_n\}_{n=1}^N$, where $X$ is $N \times D$ and $y$ is $N \times 1$, the posterior over $w$

$$p(w|X, y) = \frac{p(y|X, w)p(w)}{\int p(y|X, w)p(w)dw} = \frac{\prod_{n=1}^N p(y_n|x_n, w)p(w)}{\int \prod_{n=1}^N p(y_n|x_n, w)p(w)dw}$$
Bayesian Logistic Regression

- Recall that the **likelihood model** for logistic regression is Bernoulli (since $y \in \{0, 1\}$)

$$p(y|x, w) = \text{Bernoulli}(\sigma(w^T x)) = \left[ \frac{\exp(w^T x)}{1 + \exp(w^T x)} \right]^y \left[ \frac{1}{1 + \exp(w^T x)} \right]^{1-y} = \mu^y (1 - \mu)^{1-y}$$

- Just like the Bayesian linear regression case, let's use a Gaussian **prior** on $w$

$$p(w) = \mathcal{N}(0, \lambda^{-1} I_D) \propto \exp\left(-\frac{\lambda}{2} w^T w\right)$$

- Given $N$ observations $(X, y) = \{x_n, y_n\}_{n=1}^N$, where $X$ is $N \times D$ and $y$ is $N \times 1$, the posterior over $w$

$$p(w|X, y) = \frac{p(y|X, w) p(w)}{\int p(y|X, w) p(w) d w} = \frac{\prod_{n=1}^N p(y_n|x_n, w) p(w)}{\int \prod_{n=1}^N p(y_n|x_n, w) p(w) d w}$$

- The denominator is intractable in general (logistic-Bernoulli and Gaussian are not conjugate)
Bayesian Logistic Regression

- Recall that the likelihood model for logistic regression is Bernoulli (since \( y \in \{0, 1\} \))

\[
p(y|x, w) = \text{Bernoulli}(\sigma(w^\top x)) = \left[ \frac{\exp(w^\top x)}{1 + \exp(w^\top x)} \right]^y \left[ \frac{1}{1 + \exp(w^\top x)} \right]^{(1-y)} = \mu^y (1 - \mu)^{1-y}
\]

- Just like the Bayesian linear regression case, let’s use a Gaussian prior on \( w \)

\[
p(w) = \mathcal{N}(0, \lambda^{-1} \mathbf{I}_D) \propto \exp(-\frac{\lambda}{2} w^\top w)
\]

- Given \( N \) observations \((X, y) = \{x_n, y_n\}_{n=1}^N\), where \( X \) is \( N \times D \) and \( y \) is \( N \times 1 \), the posterior over \( w \)

\[
p(w|X, y) = \frac{p(y|X, w)p(w)}{\int p(y|X, w)p(w)dw} = \frac{\prod_{n=1}^N p(y_n|x_n, w)p(w)}{\int \prod_{n=1}^N p(y_n|x_n, w)p(w)dw}
\]

- The denominator is intractable in general (logistic-Bernoulli and Gaussian are not conjugate)

- Can’t get a closed form expression for \( p(w|X, y) \). Must approximate it!
Bayesian Logistic Regression

- Recall that the likelihood model for logistic regression is Bernoulli (since $y \in \{0, 1\}$)

$$p(y|x, w) = \text{Bernoulli}(\sigma(w^T x)) = \left[ \frac{\exp(w^T x)}{1 + \exp(w^T x)} \right]^y \left[ \frac{1}{1 + \exp(w^T x)} \right]^{(1-y)} = \mu^y (1 - \mu)^{1-y}$$

- Just like the Bayesian linear regression case, let’s use a Gaussian prior on $w$

$$p(w) = \mathcal{N}(0, \lambda^{-1} I_D) \propto \exp\left(-\frac{\lambda}{2} w^T w\right)$$

- Given $N$ observations $(X, y) = \{x_n, y_n\}_{n=1}^N$, where $X$ is $N \times D$ and $y$ is $N \times 1$, the posterior over $w$

$$p(w|X, y) = \frac{p(y|X, w)p(w)}{\int p(y|X, w)p(w)dw} = \frac{\prod_{n=1}^N p(y_n|x_n, w)p(w)}{\int \prod_{n=1}^N p(y_n|x_n, w)p(w)dw}$$

- The denominator is intractable in general (logistic-Bernoulli and Gaussian are not conjugate)
  - Can’t get a closed form expression for $p(w|X, y)$. Must approximate it!
  - Several ways to do it, e.g., MCMC, variational inference, Laplace approximation (today)
Laplace Approximation of Posterior Distribution

- Approximate the posterior distribution \( p(\theta | D) = \frac{p(D|\theta)p(\theta)}{p(D)} = \frac{p(D,\theta)}{p(D)} \) by the following Gaussian

\[
p(\theta | D) \approx \mathcal{N}(\theta_{MAP}, H^{-1})
\]
Laplace Approximation of Posterior Distribution

- Approximate the posterior distribution
  \[ p(\theta | D) = \frac{p(D|\theta)p(\theta)}{p(D)} = \frac{p(D,\theta)}{p(D)} \]
  by the following Gaussian
  \[ p(\theta | D) \approx \mathcal{N}(\theta_{MAP}, H^{-1}) \]

- Note: \( \theta_{MAP} \) is the maximum-a-posteriori (MAP) estimate of \( \theta \), i.e.,
  \[ \theta_{MAP} = \arg \max_{\theta} p(\theta | D) = \arg \max_{\theta} p(D, \theta) \]
Laplace Approximation of Posterior Distribution

Approximate the posterior distribution $p(\theta|D) = \frac{p(D|\theta)p(\theta)}{p(D)} = \frac{p(D,\theta)}{p(D)}$ by the following Gaussian

$$p(\theta|D) \approx \mathcal{N}(\theta_{MAP}, H^{-1})$$

Note: $\theta_{MAP}$ is the maximum-a-posteriori (MAP) estimate of $\theta$, i.e.,

$$\theta_{MAP} = \arg \max_\theta p(\theta|D) = \arg \max_\theta p(D, \theta) = \arg \max_\theta p(D|\theta)p(\theta)$$
Laplace Approximation of Posterior Distribution

Approximate the posterior distribution \( p(\theta|D) = \frac{p(D|\theta)p(\theta)}{p(D)} = \frac{p(D, \theta)}{p(D)} \) by the following Gaussian

\[
p(\theta|D) \approx \mathcal{N}(\theta_{MAP}, H^{-1})
\]

Note: \( \theta_{MAP} \) is the maximum-a-posteriori (MAP) estimate of \( \theta \), i.e.,

\[
\theta_{MAP} = \arg \max_\theta p(\theta|D) = \arg \max_\theta p(D, \theta) = \arg \max_\theta p(D|\theta)p(\theta) = \arg \max_\theta [\log p(D|\theta) + \log p(\theta)]
\]
Laplace Approximation of Posterior Distribution

- Approximate the posterior distribution \( p(\theta|D) = \frac{p(D|\theta)p(\theta)}{p(D)} = \frac{p(D,\theta)}{p(D)} \) by the following Gaussian

\[
p(\theta|D) \approx \mathcal{N}(\theta_{MAP}, H^{-1})
\]

Note: \( \theta_{MAP} \) is the maximum-a-posteriori (MAP) estimate of \( \theta \), i.e.,

\[
\theta_{MAP} = \arg\max_{\theta} p(\theta|D) = \arg\max_{\theta} p(D, \theta) = \arg\max_{\theta} p(D|\theta)p(\theta) = \arg\max_{\theta} \left[ \log p(D|\theta) + \log p(\theta) \right]
\]

- Usually \( \theta_{MAP} \) can be easily solved for (e.g., using first/second order iterative methods)
Laplace Approximation of Posterior Distribution

Approximate the posterior distribution

\[ p(\theta|D) = \frac{p(D|\theta)p(\theta)}{p(D)} = \frac{p(D,\theta)}{p(D)} \]

by the following Gaussian

\[ p(\theta|D) \approx \mathcal{N}(\theta_{\text{MAP}}, H^{-1}) \]

Note: \( \theta_{\text{MAP}} \) is the maximum-a-posteriori (MAP) estimate of \( \theta \), i.e.,

\[ \theta_{\text{MAP}} = \arg\max_{\theta} p(\theta|D) = \arg\max_{\theta} p(D,\theta) = \arg\max_{\theta} p(D|\theta)p(\theta) = \arg\max_{\theta}[\log p(D|\theta) + \log p(\theta)] \]

Usually \( \theta_{\text{MAP}} \) can be easily solved for (e.g., using first/second order iterative methods)

- \( H \) is the Hessian matrix of the negative log-posterior (or negative log-joint-prob) at \( \theta_{\text{MAP}} \)

\[ H = -\nabla^2 \log p(\theta|D)|_{\theta=\theta_{\text{MAP}}} \]
Laplace Approximation of Posterior Distribution

- Approximate the posterior distribution \( p(\theta|D) = \frac{p(D|\theta)p(\theta)}{p(D)} = \frac{p(D,\theta)}{p(D)} \) by the following Gaussian

\[
p(\theta|D) \approx N(\theta_{MAP}, H^{-1})
\]

- Note: \( \theta_{MAP} \) is the maximum-a-posteriori (MAP) estimate of \( \theta \), i.e.,

\[
\theta_{MAP} = \arg \max_{\theta} p(\theta|D) = \arg \max_{\theta} p(D, \theta) = \arg \max_{\theta} p(D|\theta)p(\theta) = \arg \max_{\theta} \left[ \log p(D|\theta) + \log p(\theta) \right]
\]

- Usually \( \theta_{MAP} \) can be easily solved for (e.g., using first/second order iterative methods)

- \( H \) is the Hessian matrix of the negative log-posterior (or negative log-joint-prob) at \( \theta_{MAP} \)

\[
H = -\nabla^2 \log p(\theta|D)|_{\theta=\theta_{MAP}} = -\nabla^2 \log p(D, \theta)|_{\theta=\theta_{MAP}}
\]
Laplace Approximation of Posterior Distribution

- Approximate the posterior distribution
  \[ p(\theta|D) = \frac{p(D|\theta)p(\theta)}{p(D)} = \frac{p(D,\theta)}{p(D)} \]
  by the following Gaussian
  \[ p(\theta|D) \approx \mathcal{N}(\theta_{\text{MAP}}, H^{-1}) \]

- Note: \( \theta_{\text{MAP}} \) is the maximum-a-posteriori (MAP) estimate of \( \theta \), i.e.,
  \[ \theta_{\text{MAP}} = \arg\max_{\theta} p(\theta|D) = \arg\max_{\theta} p(D, \theta) = \arg\max_{\theta} p(D|\theta)p(\theta) = \arg\max_{\theta} [\log p(D|\theta) + \log p(\theta)] \]

- Usually \( \theta_{\text{MAP}} \) can be easily solved for (e.g., using first/second order iterative methods)

- \( H \) is the Hessian matrix of the negative log-posterior (or negative log-joint-prob) at \( \theta_{\text{MAP}} \)
  \[ H = -\nabla^2 \log p(\theta|D)|_{\theta=\theta_{\text{MAP}}} = -\nabla^2 \log p(D, \theta)|_{\theta=\theta_{\text{MAP}}} = -\nabla^2 [\log p(D|\theta) + \log p(\theta)]|_{\theta=\theta_{\text{MAP}}} \]
Derivation of the Laplace Approximation

Let's write the Bayes rule as

\[ p(\theta|\mathcal{D}) = \frac{p(\mathcal{D}, \theta)}{p(\mathcal{D})} \]

Suppose \( \log p(\mathcal{D}, \theta) = f(\theta) \). Let's approximate \( f(\theta) \) using its 2nd order Taylor expansion

\[ f(\theta) \approx f(\theta_0) + (\theta - \theta_0)^\top \nabla f(\theta_0) + \frac{1}{2} (\theta - \theta_0)^\top \nabla^2 f(\theta_0)(\theta - \theta_0) \]

where \( \theta_0 \) is some arbitrarily chosen point in the domain of \( f \). Let's choose \( \theta_0 = \theta_{MAP} \).

Note that \( \nabla f(\theta_{MAP}) = \nabla \log p(\mathcal{D}, \theta_{MAP}) = 0 \).

Therefore

\[ \log p(\mathcal{D}, \theta) \approx \log p(\mathcal{D}, \theta_{MAP}) + \frac{1}{2} (\theta - \theta_{MAP})^\top \nabla^2 \log p(\mathcal{D}, \theta_{MAP})(\theta - \theta_{MAP}) \]
Derivation of the Laplace Approximation

Let’s write the Bayes rule as

\[ p(\theta | \mathcal{D}) = \frac{p(\mathcal{D}, \theta)}{p(\mathcal{D})} = \frac{p(\mathcal{D}, \theta)}{\int p(\mathcal{D}, \theta) d\theta} \]

Suppose \( \log p(\mathcal{D}, \theta) = f(\theta) \). Let’s approximate \( f(\theta) \) using its 2nd order Taylor expansion

\[ f(\theta) \approx f(\theta_0) + (\theta - \theta_0)^\top \nabla f(\theta_0) + \frac{1}{2} (\theta - \theta_0)^\top \nabla^2 f(\theta_0) (\theta - \theta_0) \]

where \( \theta_0 \) is some arbitrarily chosen point in the domain of \( f \).

Let’s choose \( \theta_0 = \theta_{\text{MAP}} \).

Note that \( \nabla f(\theta_{\text{MAP}}) = \nabla \log p(\mathcal{D}, \theta_{\text{MAP}}) = 0 \).

Therefore

\[ \log p(\mathcal{D}, \theta) \approx \log p(\mathcal{D}, \theta_{\text{MAP}}) + \frac{1}{2} (\theta - \theta_{\text{MAP}})^\top \nabla^2 \log p(\mathcal{D}, \theta_{\text{MAP}}) (\theta - \theta_{\text{MAP}}) \]
Derivation of the Laplace Approximation

Let’s write the Bayes rule as

\[
p(\theta|\mathcal{D}) = \frac{p(\mathcal{D}, \theta)}{p(\mathcal{D})} = \frac{p(\mathcal{D}, \theta)}{\int p(\mathcal{D}, \theta) d\theta} = \frac{e^{\log p(\mathcal{D}, \theta)}}{\int e^{\log p(\mathcal{D}, \theta)} d\theta}
\]
Derivation of the Laplace Approximation

Let’s write the Bayes rule as

\[
p(\theta|D) = \frac{p(D, \theta)}{p(D)} = \frac{p(D, \theta)}{\int p(D, \theta) \, d\theta} = \frac{e^{\log p(D, \theta)}}{\int e^{\log p(D, \theta)} \, d\theta}
\]

Suppose \( \log p(D, \theta) = f(\theta) \). Let’s approximate \( f(\theta) \) using its 2nd order Taylor expansion

\[
f(\theta) \approx f(\theta_0) + (\theta - \theta_0)^T \nabla f(\theta_0) + \frac{1}{2} (\theta - \theta_0)^T \nabla^2 f(\theta_0)(\theta - \theta_0)
\]

where \( \theta_0 \) is some arbitrarily chosen point in the domain of \( f \).
Derivation of the Laplace Approximation

- Let’s write the Bayes rule as

\[
p(\theta|\mathcal{D}) = \frac{p(\mathcal{D}, \theta)}{p(\mathcal{D})} = \frac{p(\mathcal{D}, \theta)}{\int p(\mathcal{D}, \theta) d\theta} = \frac{e^{\log p(\mathcal{D}, \theta)}}{\int e^{\log p(\mathcal{D}, \theta)} d\theta}
\]

- Suppose \( \log p(\mathcal{D}, \theta) = f(\theta) \). Let’s approximate \( f(\theta) \) using its 2nd order Taylor expansion

\[
f(\theta) \approx f(\theta_0) + (\theta - \theta_0)^\top \nabla f(\theta_0) + \frac{1}{2} (\theta - \theta_0)^\top \nabla^2 f(\theta_0)(\theta - \theta_0)
\]

where \( \theta_0 \) is some arbitrarily chosen point in the domain of \( f \)

- Let’s choose \( \theta_0 = \theta_{MAP} \).
Derivation of the Laplace Approximation

Let’s write the Bayes rule as

$$p(\theta|\mathcal{D}) = \frac{p(\mathcal{D}, \theta)}{p(\mathcal{D})} = \frac{p(\mathcal{D}, \theta)}{\int p(\mathcal{D}, \theta) d\theta} = \frac{e^{\log p(\mathcal{D}, \theta)}}{\int e^{\log p(\mathcal{D}, \theta)} d\theta}$$

Suppose $\log p(\mathcal{D}, \theta) = f(\theta)$. Let’s approximate $f(\theta)$ using its 2nd order Taylor expansion

$$f(\theta) \approx f(\theta_0) + (\theta - \theta_0)^\top \nabla f(\theta_0) + \frac{1}{2} (\theta - \theta_0)^\top \nabla^2 f(\theta_0)(\theta - \theta_0)$$

where $\theta_0$ is some arbitrarily chosen point in the domain of $f$

Let’s choose $\theta_0 = \theta_{MAP}$. Note that $\nabla f(\theta_{MAP}) = \nabla \log p(\mathcal{D}, \theta_{MAP}) = 0$. 
Derivation of the Laplace Approximation

Let’s write the Bayes rule as

\[ p(\theta|\mathcal{D}) = \frac{p(\mathcal{D}, \theta)}{p(\mathcal{D})} = \frac{p(\mathcal{D}, \theta)}{\int p(\mathcal{D}, \theta) d\theta} = \frac{e^{\log p(\mathcal{D}, \theta)}}{\int e^{\log p(\mathcal{D}, \theta)} d\theta} \]

Suppose \( \log p(\mathcal{D}, \theta) = f(\theta) \). Let’s approximate \( f(\theta) \) using its 2nd order Taylor expansion

\[ f(\theta) \approx f(\theta_0) + (\theta - \theta_0)^\top \nabla f(\theta_0) + \frac{1}{2} (\theta - \theta_0)^\top \nabla^2 f(\theta_0) (\theta - \theta_0) \]

where \( \theta_0 \) is some arbitrarily chosen point in the domain of \( f \)

Let’s choose \( \theta_0 = \theta_{MAP} \). Note that \( \nabla f(\theta_{MAP}) = \nabla \log p(\mathcal{D}, \theta_{MAP}) = 0 \). Therefore

\[ \log p(\mathcal{D}, \theta) \approx \log p(\mathcal{D}, \theta_{MAP}) + \frac{1}{2} (\theta - \theta_{MAP})^\top \nabla^2 \log p(\mathcal{D}, \theta_{MAP}) (\theta - \theta_{MAP}) \]
Derivation of the Laplace Approximation

- Plugging in this 2nd order Taylor approximation for \( \log p(D, \theta) \), we have

\[
p(\theta | D) = \frac{e^{\log p(D, \theta)}}{\int e^{\log p(D, \theta)} d\theta}
\]

Further simplifying, we have

\[
p(\theta | D) \approx e^{-\frac{1}{2} (\theta - \theta_{MAP})^\top \{ -\nabla^2 \log p(D, \theta_{MAP}) \} (\theta - \theta_{MAP})}
\]

Therefore the Laplace approximation of the posterior \( p(\theta | D) \) is a Gaussian and is given by

\[
p(\theta | D) \approx N(\theta | \theta_{MAP}, H^{-1})
\]
Derivation of the Laplace Approximation

Plugging in this 2nd order Taylor approximation for \( \log p(D, \theta) \), we have

\[
p(\theta|D) = \frac{e^{\log p(D, \theta)}}{\int e^{\log p(D, \theta)} d\theta} \approx \frac{e^{\log p(D, \theta_{\text{MAP}}) + \frac{1}{2}(\theta - \theta_{\text{MAP}})^\top \nabla^2 \log p(D, \theta_{\text{MAP}})(\theta - \theta_{\text{MAP}})}}{\int e^{\log p(D, \theta_{\text{MAP}}) + \frac{1}{2}(\theta - \theta_{\text{MAP}})^\top \nabla^2 \log p(D, \theta_{\text{MAP}})(\theta - \theta_{\text{MAP}})} d\theta}
\]

Therefore the Laplace approximation of the posterior \( p(\theta|D) \) is a Gaussian and is given by

\[
p(\theta|D) \approx N(\theta|\theta_{\text{MAP}}, H^{-1})
\]
Derivation of the Laplace Approximation

Plugging in this 2nd order Taylor approximation for $\log p(D, \theta)$, we have

$$p(\theta|D) = \frac{e^{\log p(D, \theta)}}{\int e^{\log p(D, \theta)} d\theta} \approx \frac{e^{\log p(D, \theta_{MAP}) + \frac{1}{2} (\theta - \theta_{MAP})^\top \nabla^2 \log p(D, \theta_{MAP})(\theta - \theta_{MAP})}}{\int e^{\log p(D, \theta_{MAP}) + \frac{1}{2} (\theta - \theta_{MAP})^\top \nabla^2 \log p(D, \theta_{MAP})(\theta - \theta_{MAP})} d\theta}$$

Further simplifying, we have

$$p(\theta|D) \approx \frac{e^{-\frac{1}{2} (\theta - \theta_{MAP})^\top \{-\nabla^2 \log p(D, \theta_{MAP})\}(\theta - \theta_{MAP})}}{\int e^{-\frac{1}{2} (\theta - \theta_{MAP})^\top \{-\nabla^2 \log p(D, \theta_{MAP})\}(\theta - \theta_{MAP})} d\theta}$$
Derivation of the Laplace Approximation

- Plugging in this 2nd order Taylor approximation for \( \log p(D, \theta) \), we have

\[
p(\theta|D) = \frac{e^{\log p(D, \theta)}}{\int e^{\log p(D, \theta)} d\theta} \approx \frac{e^{\log p(D, \theta_{MAP}) + \frac{1}{2} (\theta - \theta_{MAP})^\top \nabla^2 \log p(D, \theta_{MAP}) (\theta - \theta_{MAP})}}{\int e^{\log p(D, \theta_{MAP}) + \frac{1}{2} (\theta - \theta_{MAP})^\top \nabla^2 \log p(D, \theta_{MAP}) (\theta - \theta_{MAP})} d\theta}
\]

- Further simplifying, we have

\[
p(\theta|D) \approx \frac{e^{-\frac{1}{2} (\theta - \theta_{MAP})^\top \{- \nabla^2 \log p(D, \theta_{MAP})\} (\theta - \theta_{MAP})}}{\int e^{-\frac{1}{2} (\theta - \theta_{MAP})^\top \{- \nabla^2 \log p(D, \theta_{MAP})\} (\theta - \theta_{MAP})} d\theta}
\]

- Therefore the Laplace approximation of the posterior \( p(\theta|D) \) is a Gaussian and is given by

\[
p(\theta|D) \approx \mathcal{N}(\theta|\theta_{MAP}, H^{-1}) \quad \text{where} \quad H = -\nabla^2 \log p(D, \theta_{MAP})
\]
Derivation of the Laplace Approximation

- Plugging in this 2nd order Taylor approximation for $\log p(D, \theta)$, we have

$$p(\theta|D) = \frac{e^{\log p(D, \theta)}}{\int e^{\log p(D, \theta)} d\theta} \approx \frac{e^{\log p(D, \theta_{MAP}) + \frac{1}{2}(\theta - \theta_{MAP})^\top \nabla^2 \log p(D, \theta_{MAP})(\theta - \theta_{MAP})}}{\int e^{\log p(D, \theta_{MAP}) + \frac{1}{2}(\theta - \theta_{MAP})^\top \nabla^2 \log p(D, \theta_{MAP})(\theta - \theta_{MAP})} d\theta}$$

- Further simplifying, we have

$$p(\theta|D) \approx \frac{e^{-\frac{1}{2}(\theta - \theta_{MAP})^\top \{-\nabla^2 \log p(D, \theta_{MAP})\}(\theta - \theta_{MAP})}}{\int e^{-\frac{1}{2}(\theta - \theta_{MAP})^\top \{-\nabla^2 \log p(D, \theta_{MAP})\}(\theta - \theta_{MAP})} d\theta}$$

- Therefore the Laplace approximation of the posterior $p(\theta|D)$ is a Gaussian and is given by

$$p(\theta|D) \approx \mathcal{N}(\theta|\theta_{MAP}, H^{-1})$$

where $H = -\nabla^2 \log p(D, \theta_{MAP})$
Properties of Laplace Approximation

- Usually straightforward if derivatives (first and second) can be computed easily
- Expensive if the number of parameters is very large (due to Hessian computation and inversion)
- Can do badly if the (true) posterior is multimodal
- Can actually apply it when working with any regularized loss function (not just probabilistic models) to get a Gaussian posterior distribution over the parameters
- negative log-likelihood (NLL) = loss function, negative log-prior = regularizer

Easy exercise: Try doing this for $\ell_2$ regularized least squares regression (will get the same posterior as in Bayesian linear regression)
Properties of Laplace Approximation

- Usually straightforward if derivatives (first and second) can be computed easily.
- Expensive if the number of parameters is very large (due to Hessian computation and inversion).

Can do badly if the (true) posterior is multimodal.

Can actually apply it when working with any regularized loss function (not just probabilistic models) to get a Gaussian posterior distribution over the parameters.

Negative log-likelihood (NLL) = loss function, negative log-prior = regularizer.

Easy exercise: Try doing this for $\ell_2$ regularized least squares regression (will get the same posterior as in Bayesian linear regression).
Properties of Laplace Approximation

- Usually straightforward if derivatives (first and second) can be computed easily
- Expensive if the number of parameters is very large (due to Hessian computation and inversion)
- Can do badly if the (true) posterior is multimodal

Can actually apply it when working with any regularized loss function (not just probabilistic models) to get a Gaussian posterior distribution over the parameters

easy exercise: Try doing this for $\ell_2$ regularized least squares regression (will get the same posterior as in Bayesian linear regression)
Properties of Laplace Approximation

- Usually straightforward if derivatives (first and second) can be computed easily.
- Expensive if the number of parameters is very large (due to Hessian computation and inversion).
- Can do badly if the (true) posterior is multimodal.

Can actually apply it when working with any regularized loss function (not just probabilistic models) to get a Gaussian posterior distribution over the parameters.
Properties of Laplace Approximation

- Usually straightforward if derivatives (first and second) can be computed easily
- Expensive if the number of parameters is very large (due to Hessian computation and inversion)
- Can do badly if the (true) posterior is multimodal

Can actually apply it when working with **any regularized loss function** (not just probabilistic models) to get a Gaussian posterior distribution over the parameters

- negative log-likelihood (NLL) = loss function, negative log-prior = regularizer
Properties of Laplace Approximation

- Usually straightforward if derivatives (first and second) can be computed easily.
- Expensive if the number of parameters is very large (due to Hessian computation and inversion).
- Can do badly if the (true) posterior is multimodal.

Can actually apply it when working with any regularized loss function (not just probabilistic models) to get a Gaussian posterior distribution over the parameters.

- negative log-likelihood (NLL) = loss function, negative log-prior = regularizer.
- Easy exercise: Try doing this for $l_2$ regularized least squares regression (will get the same posterior as in Bayesian linear regression).
Laplace Approximation for Bayesian Logistic Regression

Data $\mathcal{D} = (X, y)$ and parameter $\theta = w$. The Laplace approximation of posterior will be

$$p(w | X, y) \approx \mathcal{N}(w_{MAP}, H^{-1})$$
Laplace Approximation for Bayesian Logistic Regression

- Data $\mathcal{D} = (X, y)$ and parameter $\theta = w$. The Laplace approximation of posterior will be

$$p(w|X, y) \approx \mathcal{N}(w_{MAP}, H^{-1})$$

- The required quantities are defined as

$$w_{MAP} = \arg\max_w \log p(w|y, X) = \arg\max_w \log p(y, w|X) = \arg\min_w [-\log p(y, w|X)]$$

$$H = \nabla^2 [\log p(y, w|X)] \bigg|_{w = w_{MAP}}$$

We can compute $w_{MAP}$ using iterative methods (gradient descent):

First-order (gradient) methods:

$$w_{t+1} = w_t - \eta g_t$$

Requires gradient $g$ of $\log p(y, w|X)$

$$g = \nabla [\log p(y, w|X)]$$

Second-order methods:

$$w_{t+1} = w_t - H^{-1} t g_t$$

Requires both gradient and Hessian (defined above)

Note: When using second order methods for estimating $w_{MAP}$, we anyway get the Hessian needed for the Laplace approximation of the posterior.
Laplace Approximation for Bayesian Logistic Regression

- Data $\mathcal{D} = (X, y)$ and parameter $\theta = w$. The Laplace approximation of posterior will be

$$p(w|X, y) \approx \mathcal{N}(w_{\text{MAP}}, H^{-1})$$

- The required quantities are defined as

$$w_{\text{MAP}} = \arg \max_w \log p(w|y, X) = \arg \max_w \log p(y, w|X) = \arg \min_w [-\log p(y, w|X)]$$

$$H = \nabla^2 [-\log p(y, w|X)]|_{w=w_{\text{MAP}}}$$
Laplace Approximation for Bayesian Logistic Regression

- Data $\mathcal{D} = (\mathbf{X}, \mathbf{y})$ and parameter $\theta = \mathbf{w}$. The Laplace approximation of posterior will be

$$p(\mathbf{w}|\mathbf{X}, \mathbf{y}) \approx \mathcal{N}(\mathbf{w}_{\text{MAP}}, \mathbf{H}^{-1})$$

- The required quantities are defined as

$$\mathbf{w}_{\text{MAP}} = \arg \max_{\mathbf{w}} \log p(\mathbf{w}|\mathbf{y}, \mathbf{X}) = \arg \max_{\mathbf{w}} \log p(\mathbf{y}, \mathbf{w}|\mathbf{X}) = \arg \min_{\mathbf{w}} [-\log p(\mathbf{y}, \mathbf{w}|\mathbf{X})]$$

$$\mathbf{H} = \nabla^2 [-\log p(\mathbf{y}, \mathbf{w}|\mathbf{X})]|_{\mathbf{w}=\mathbf{w}_{\text{MAP}}}$$

- We can compute $\mathbf{w}_{\text{MAP}}$ using iterative methods (gradient descent):

  - First-order (gradient) methods:

    $$\mathbf{w}_{t+1} = \mathbf{w}_t - \eta \mathbf{g}_t$$

    Requires gradient $\mathbf{g}$ of $-\log p(\mathbf{y}, \mathbf{w}|\mathbf{X})$

  - Second-order methods:

    $$\mathbf{w}_{t+1} = \mathbf{w}_t - \mathbf{H}^{-1}_t \mathbf{g}_t$$

    Requires both gradient and Hessian (defined above)

Note: When using second order methods for estimating $\mathbf{w}_{\text{MAP}}$, we anyway get the Hessian needed for the Laplace approximation of the posterior.
Laplace Approximation for Bayesian Logistic Regression

- Data $D = (X, y)$ and parameter $\theta = w$. The Laplace approximation of posterior will be
  
  $$p(w|X, y) \approx \mathcal{N}(w_{MAP}, H^{-1})$$

- The required quantities are defined as
  
  $$w_{MAP} = \arg\max_w \log p(w|y, X) = \arg\max_w \log p(y, w|X) = \arg\min_w [-\log p(y, w|X)]$$
  
  $$H = \nabla^2 [−\log p(y, w|X)]|_{w=w_{MAP}}$$

- We can compute $w_{MAP}$ using iterative methods (gradient descent):
  
  - First-order (gradient) methods: $w_{t+1} = w_t - \eta g_t$. Requires gradient $g$ of $−\log p(y, w|X)$
    
    $$g = \nabla [−\log p(y, w|X)]$$
Laplace Approximation for Bayesian Logistic Regression

- Data $\mathcal{D} = (\mathbf{X}, \mathbf{y})$ and parameter $\theta = \mathbf{w}$. The Laplace approximation of posterior will be

$$p(\mathbf{w}|\mathbf{X}, \mathbf{y}) \approx \mathcal{N}(\mathbf{w}_{\text{MAP}}, \mathbf{H}^{-1})$$

- The required quantities are defined as

$$\mathbf{w}_{\text{MAP}} = \arg \max_{\mathbf{w}} \log p(\mathbf{w}|\mathbf{y}, \mathbf{X}) = \arg \max_{\mathbf{w}} \log p(\mathbf{y}, \mathbf{w}|\mathbf{X}) = \arg \min_{\mathbf{w}} [-\log p(\mathbf{y}, \mathbf{w}|\mathbf{X})]$$

$$\mathbf{H} = \nabla^2 [-\log p(\mathbf{y}, \mathbf{w}|\mathbf{X})]|_{\mathbf{w}=\mathbf{w}_{\text{MAP}}}$$

- We can compute $\mathbf{w}_{\text{MAP}}$ using iterative methods (gradient descent):

  - First-order (gradient) methods: $\mathbf{w}_{t+1} = \mathbf{w}_t - \eta g_t$. Requires gradient $g$ of $-\log p(\mathbf{y}, \mathbf{w}|\mathbf{X})$

    $$g = \nabla [-\log p(\mathbf{y}, \mathbf{w}|\mathbf{X})]$$

  - Second-order methods. $\mathbf{w}_{t+1} = \mathbf{w}_t - \mathbf{H}_t^{-1} g_t$. Requires both gradient and Hessian (defined above)
Laplace Approximation for Bayesian Logistic Regression

- Data $\mathcal{D} = (\mathbf{X}, \mathbf{y})$ and parameter $\theta = \mathbf{w}$. The Laplace approximation of posterior will be
  
  $p(\mathbf{w}|\mathbf{X}, \mathbf{y}) \approx \mathcal{N}(\mathbf{w}_{\text{MAP}}, \mathbf{H}^{-1})$

- The required quantities are defined as
  
  $\mathbf{w}_{\text{MAP}} = \arg \max_{\mathbf{w}} \log p(\mathbf{w}|\mathbf{y}, \mathbf{X}) = \arg \max_{\mathbf{w}} \log p(\mathbf{y}, \mathbf{w}|\mathbf{X}) = \arg \min_{\mathbf{w}} [−\log p(\mathbf{y}, \mathbf{w}|\mathbf{X})]$

  $\mathbf{H} = \nabla^2 [−\log p(\mathbf{y}, \mathbf{w}|\mathbf{X})]|_{\mathbf{w}=\mathbf{w}_{\text{MAP}}}$

- We can compute $\mathbf{w}_{\text{MAP}}$ using iterative methods (gradient descent):
  
  - First-order (gradient) methods: $\mathbf{w}_{t+1} = \mathbf{w}_t − \eta \mathbf{g}_t$. Requires gradient $\mathbf{g}$ of $−\log p(\mathbf{y}, \mathbf{w}|\mathbf{X})$
    
    $\mathbf{g} = \nabla [−\log p(\mathbf{y}, \mathbf{w}|\mathbf{X})]$

  - Second-order methods. $\mathbf{w}_{t+1} = \mathbf{w}_t − \mathbf{H}_t^{-1} \mathbf{g}_t$. Requires both gradient and Hessian (defined above)

- Note: When using second order methods for estimating $\mathbf{w}_{\text{MAP}}$, we anyway get the Hessian needed for the Laplace approximation of the posterior
The LR objective function $- \log p(y, w|X) = - \log p(y|X, w) - \log p(w)$ can be written as

$$- \log \prod_{n=1}^{N} p(y_n|x_n, w) - \log p(w)$$
The LR objective function $- \log p(y, w|X) = - \log p(y|X, w) - \log p(w)$ can be written as

$$- \log \prod_{n=1}^{N} p(y_n|x_n, w) - \log p(w) = - \sum_{n=1}^{N} \log p(y_n|x_n, w) - \log p(w)$$
An Aside: Gradient and Hessian for Logistic Regression

The LR objective function \( - \log p(y, w|X) = - \log p(y|X, w) - \log p(w) \) can be written as

\[
- \log \prod_{n=1}^{N} p(y_n|x_n, w) - \log p(w) = - \sum_{n=1}^{N} \log p(y_n|x_n, w) - \log p(w)
\]

For the logistic regression model, \( p(y_n|x_n, w) = \mu_n^{y_n} (1 - \mu_n)^{1-y_n} \) where \( \mu_n = \frac{\exp(w^\top x_n)}{1 + \exp(w^\top x_n)} \)
The LR objective function \(- \log p(y, w|X) = - \log p(y|X, w) - \log p(w)\) can be written as

\[ - \log \prod_{n=1}^{N} p(y_n|x_n, w) - \log p(w) = - \sum_{n=1}^{N} \log p(y_n|x_n, w) - \log p(w) \]

For the logistic regression model, \(p(y_n|x_n, w) = \mu_n^{y_n}(1 - \mu_n)^{1-y_n}\) where \(\mu_n = \frac{\exp(w^\top x_n)}{1 + \exp(w^\top x_n)}\)

With a Gaussian prior \(p(w) = \mathcal{N}(w|0, \lambda^{-1}I) \propto \exp(-\lambda w^\top w)\), the gradient and Hessian will be
An Aside: Gradient and Hessian for Logistic Regression

- The LR objective function $- \log p(y, w|X) = - \log p(y|X, w) - \log p(w)$ can be written as

$$- \log \prod_{n=1}^{N} p(y_n|x_n, w) - \log p(w) = - \sum_{n=1}^{N} \log p(y_n|x_n, w) - \log p(w)$$

- For the logistic regression model, $p(y_n|x_n, w) = \mu_n^{y_n}(1 - \mu_n)^{1-y_n}$ where $\mu_n = \frac{\exp(w^\top x_n)}{1 + \exp(w^\top x_n)}$

- With a Gaussian prior $p(w) = \mathcal{N}(w|0, \lambda^{-1}I) \propto \exp(-\lambda w^\top w)$, the gradient and Hessian will be

$$g = - \sum_{n=1}^{N} (y_n - \mu_n)x_n + \lambda I w$$
The LR objective function \(- \log p(y, w | X) = - \log p(y | X, w) - \log p(w)\) can be written as

\[- \log \prod_{n=1}^{N} p(y_n | x_n, w) - \log p(w) = - \sum_{n=1}^{N} \log p(y_n | x_n, w) - \log p(w)\]

For the logistic regression model, \(p(y_n | x_n, w) = \mu_n^{y_n} (1 - \mu_n)^{1-y_n}\) where \(\mu_n = \frac{\exp(w^\top x_n)}{1+\exp(w^\top x_n)}\)

With a Gaussian prior \(p(w) = \mathcal{N}(w|0, \lambda^{-1}I) \propto \exp(-\lambda w^\top w)\), the gradient and Hessian will be

\[g = - \sum_{n=1}^{N} (y_n - \mu_n)x_n + \lambda w = X^\top (\mu - y) + \lambda w\]
An Aside: Gradient and Hessian for Logistic Regression

- The LR objective function $- \log p(y, w|X) = - \log p(y|X, w) - \log p(w)$ can be written as
  
  $$- \log \prod_{n=1}^{N} p(y_n|x_n, w) - \log p(w) = - \sum_{n=1}^{N} \log p(y_n|x_n, w) - \log p(w)$$

- For the logistic regression model, $p(y_n|x_n, w) = \mu_n^{y_n}(1 - \mu_n)^{1-y_n}$ where $\mu_n = \frac{\exp(w^\top x_n)}{1 + \exp(w^\top x_n)}$

- With a Gaussian prior $p(w) = \mathcal{N}(w|0, \lambda^{-1}I) \propto \exp(-\lambda w^\top w)$, the gradient and Hessian will be

$$g = - \sum_{n=1}^{N} (y_n - \mu_n)x_n + \lambda w = X^\top(\mu - y) + \lambda w \quad (a \ D \times 1 \ vector)$$
The LR objective function \(- \log p(y, w|X) = - \log p(y|X, w) - \log p(w)\) can be written as

\[- \log \prod_{n=1}^{N} p(y_n|x_n, w) - \log p(w) = - \sum_{n=1}^{N} \log p(y_n|x_n, w) - \log p(w)\]

For the logistic regression model, \(p(y_n|x_n, w) = \mu_n^y_n(1 - \mu_n)^{1-y_n}\) where \(\mu_n = \frac{\exp(w^\top x_n)}{1 + \exp(w^\top x_n)}\)

With a Gaussian prior \(p(w) = \mathcal{N}(w|0, \lambda^{-1}I) \propto \exp(-\lambda w^\top w)\), the gradient and Hessian will be

\[g = - \sum_{n=1}^{N} (y_n - \mu_n)x_n + \lambda w = X^\top (\mu - y) + \lambda w \quad (\text{a } D \times 1 \text{ vector})\]

\[H = \sum_{n=1}^{N} \mu_n(1 - \mu_n)x_n x_n^\top + \lambda I\]
An Aside: Gradient and Hessian for Logistic Regression

The LR objective function $- \log p(y, w|X) = - \log p(y|X, w) - \log p(w)$ can be written as

$$- \log \prod_{n=1}^{N} p(y_n|x_n, w) - \log p(w) = - \sum_{n=1}^{N} \log p(y_n|x_n, w) - \log p(w)$$

For the logistic regression model, $p(y_n|x_n, w) = \mu_n^{y_n}(1 - \mu_n)^{1-y_n}$ where $\mu_n = \frac{\exp(w^\top x_n)}{1+\exp(w^\top x_n)}$

With a Gaussian prior $p(w) = \mathcal{N}(w|0, \lambda^{-1}I) \propto \exp(-\lambda w^\top w)$, the gradient and Hessian will be

$$g = - \sum_{n=1}^{N} (y_n - \mu_n)x_n + \lambda w = X^\top (\mu - y) + \lambda w \quad \text{(a } D \times 1 \text{ vector)}$$

$$H = \sum_{n=1}^{N} \mu_n(1 - \mu_n)x_n x_n^\top + \lambda I = X^\top S X + \lambda I$$
An Aside: Gradient and Hessian for Logistic Regression

- The LR objective function $- \log p(y, w | X) = - \log p(y | X, w) - \log p(w)$ can be written as

  $$- \log \prod_{n=1}^{N} p(y_n | x_n, w) - \log p(w) = - \sum_{n=1}^{N} \log p(y_n | x_n, w) - \log p(w)$$

- For the logistic regression model, $p(y_n | x_n, w) = \mu_n^{y_n} (1 - \mu_n)^{1-y_n}$ where $\mu_n = \frac{\exp(w^\top x_n)}{1 + \exp(w^\top x_n)}$

- With a Gaussian prior $p(w) = \mathcal{N}(w | 0, \lambda^{-1} I) \propto \exp(-\lambda w^\top w)$, the gradient and Hessian will be

  $$g = - \sum_{n=1}^{N} (y_n - \mu_n) x_n + \lambda w = X^\top (\mu - y) + \lambda w \quad (a \ D \times 1 \ vector)$$

  $$H = \sum_{n=1}^{N} \mu_n (1 - \mu_n) x_n x_n^\top + \lambda I = X^\top S X + \lambda I \quad (a \ D \times D \ matrix)$$
An Aside: Gradient and Hessian for Logistic Regression

- The LR objective function \(- \log p(y, w|X) = - \log p(y|X, w) - \log p(w)\) can be written as
  \[- \log \prod_{n=1}^{N} p(y_n|x_n, w) - \log p(w) = - \sum_{n=1}^{N} \log p(y_n|x_n, w) - \log p(w)\]

- For the logistic regression model, 
  \[p(y_n|x_n, w) = \mu_n^{y_n}(1 - \mu_n)^{1-y_n}\]
  where \(\mu_n = \frac{\exp(w^\top x_n)}{1+\exp(w^\top x_n)}\)

- With a Gaussian prior \(p(w) = \mathcal{N}(w|0, \lambda^{-1}I) \propto \exp(-\lambda w^\top w)\), the gradient and Hessian will be

  \[g = - \sum_{n=1}^{N} (y_n - \mu_n)x_n + \lambda w = X^\top (\mu - y) + \lambda w \quad \text{(a } D \times 1 \text{ vector)}\]

  \[H = \sum_{n=1}^{N} \mu_n(1 - \mu_n)x_nx_n^\top + \lambda I = X^\top S X + \lambda I \quad \text{(a } D \times D \text{ matrix)}\]

- \(\mu = [\mu_1, \ldots, \mu_N]^\top\) is \(N \times 1\) and \(S\) is a \(N \times N\) diagonal matrix with \(S_{nn} = \mu_n(1 - \mu_n)\)
Logistic Regression: Predictive Distributions

- When using MLE, the predictive distribution will be

\[ p(y_\star = 1|x_\star, w_{MLE}) = \sigma(w_{MLE}^\top x_\star) \]
Logistic Regression: Predictive Distributions

When using MLE, the predictive distribution will be

\[
p(y_* = 1|x_*, w_{MLE}) = \sigma(w_{MLE}^T x_*)
\]

\[
p(y_*|x_*, w_{MLE}) = \text{Bernoulli}(\sigma(w_{MLE}^T x_*))
\]
Logistic Regression: Predictive Distributions

- When using MLE, the predictive distribution will be

\[
p(y_* = 1 | x_*, w_{MLE}) = \sigma(w_{MLE}^T x_*) \\
p(y_* | x_*, w_{MLE}) = \text{Bernoulli}(\sigma(w_{MLE}^T x_*))
\]

- When using MAP, the predictive distribution will be

\[
p(y_* = 1 | x_*, w_{MAP}) = \sigma(w_{MAP}^T x_*) \\
p(y_* | x_*, w_{MAP}) = \text{Bernoulli}(\sigma(w_{MAP}^T x_*))
\]
Logistic Regression: Predictive Distributions

- When using MLE, the predictive distribution will be
  
  \[ p(y_* = 1|x_*, w_{MLE}) = \sigma(w_{MLE}^T x_*) \]
  
  \[ p(y_* | x_*, w_{MLE}) = \text{Bernoulli}(\sigma(w_{MLE}^T x_*)) \]

- When using MAP, the predictive distribution will be
  
  \[ p(y_* = 1|x_*, w_{MAP}) = \sigma(w_{MAP}^T x_*) \]
  
  \[ p(y_* | x_*, w_{MAP}) = \text{Bernoulli}(\sigma(w_{MAP}^T x_*)) \]

- When using Bayesian inference, the posterior predictive distribution, based on posterior averaging
  
  \[ p(y_* = 1|x_*, X, y) = \int p(y_* = 1|x_*, w)p(w|X, y)dw \]
Logistic Regression: Predictive Distributions

- When using MLE, the predictive distribution will be
  
  \[ p(y_* = 1|x_*, w_{MLE}) = \sigma(w_{MLE}^T x_*) \]
  
  \[ p(y_*|x_*, w_{MLE}) = \text{Bernoulli}(\sigma(w_{MLE}^T x_*)) \]

- When using MAP, the predictive distribution will be
  
  \[ p(y_* = 1|x_*, w_{MAP}) = \sigma(w_{MAP}^T x_*) \]
  
  \[ p(y_*|x_*, w_{MAP}) = \text{Bernoulli}(\sigma(w_{MAP}^T x_*)) \]

- When using Bayesian inference, the posterior predictive distribution, based on posterior averaging
  
  \[ p(y_* = 1|x_*, X, y) = \int p(y_* = 1|x_*, w) p(w|X, y) dw = \int \sigma(w^T x_*) p(w|X, y) dw \]
Logistic Regression: Predictive Distributions

- When using MLE, the predictive distribution will be
  
  \[ p(y_\ast = 1|x_\ast, w_{\text{MLE}}) = \sigma(w_{\text{MLE}}^\top x_\ast) \]
  
  \[ p(y_\ast | x_\ast, w_{\text{MLE}}) = \text{Bernoulli}(\sigma(w_{\text{MLE}}^\top x_\ast)) \]

- When using MAP, the predictive distribution will be
  
  \[ p(y_\ast = 1|x_\ast, w_{\text{MAP}}) = \sigma(w_{\text{MAP}}^\top x_\ast) \]
  
  \[ p(y_\ast | x_\ast, w_{\text{MAP}}) = \text{Bernoulli}(\sigma(w_{\text{MAP}}^\top x_\ast)) \]

- When using Bayesian inference, the posterior predictive distribution, based on posterior averaging
  
  \[ p(y_\ast = 1|x_\ast, X, y) = \int p(y_\ast = 1|x_\ast, w)p(w|X, y)dw = \int \sigma(w^\top x_\ast)p(w|X, y)dw \]

  Above is hard in general. :-(

  If using the Laplace approximation for \( p(w|X, y) \), it will be
Logistic Regression: Predictive Distributions

- When using MLE, the predictive distribution will be
  \[ p(y_* = 1 | x_*, w_{MLE}) = \sigma(w_{MLE}^T x_*) \]
  \[ p(y_* | x_*, w_{MLE}) = \text{Bernoulli}(\sigma(w_{MLE}^T x_*)) \]

- When using MAP, the predictive distribution will be
  \[ p(y_* = 1 | x_*, w_{MAP}) = \sigma(w_{MAP}^T x_*) \]
  \[ p(y_* | x_*, w_{MAP}) = \text{Bernoulli}(\sigma(w_{MAP}^T x_*)) \]

- When using Bayesian inference, the posterior predictive distribution, based on posterior averaging
  \[ p(y_* = 1 | x_*, X, y) = \int p(y_* = 1 | x_*, w) p(w | X, y) dw = \int \sigma(w^T x_*) p(w | X, y) dw \]

Above is hard in general. :-(( If using the Laplace approximation for \( p(w | X, y) \), it will be

\[ p(y_* = 1 | x_*, X, y) \approx \int \sigma(w^T x_*) \mathcal{N}(w | w_{MAP}, H^{-1}) dw \]
Logistic Regression: Predictive Distributions

- When using MLE, the predictive distribution will be
  \[ p(y_\ast = 1|x_\ast, w_{\text{MLE}}) = \sigma(w_{\text{MLE}}^\top x_\ast) \]
  \[ p(y_\ast|x_\ast, w_{\text{MLE}}) = \text{Bernoulli}(\sigma(w_{\text{MLE}}^\top x_\ast)) \]

- When using MAP, the predictive distribution will be
  \[ p(y_\ast = 1|x_\ast, w_{\text{MAP}}) = \sigma(w_{\text{MAP}}^\top x_\ast) \]
  \[ p(y_\ast|x_\ast, w_{\text{MAP}}) = \text{Bernoulli}(\sigma(w_{\text{MAP}}^\top x_\ast)) \]

- When using Bayesian inference, the posterior predictive distribution, based on posterior averaging
  \[ p(y_\ast = 1|x_\ast, X, y) = \int p(y_\ast = 1|x_\ast, w)p(w|X, y)dw = \int \sigma(w^\top x_\ast)p(w|X, y)dw \]

Above is hard in general. :-( If using the Laplace approximation for \( p(w|X, y) \), it will be
\[ p(y_\ast = 1|x_\ast, X, y) \approx \int \sigma(w^\top x_\ast)\mathcal{N}(w|w_{\text{MAP}}, H^{-1})dw \]

Even after Laplace approximation for \( p(w|X, y) \), the above integral to compute posterior predictive is intractable. So we will need to also approximate the predictive posterior. :-(
Posterior Predictive via Monte-Carlo Sampling

- The posterior predictive is given by the following integral

\[ p(y_\ast = 1|x_\ast, X, y) = \int \sigma(w^\top x_\ast) \mathcal{N}(w|w_{MAP}, H^{-1}) dw \]
Posterior Predictive via Monte-Carlo Sampling

- The posterior predictive is given by the following integral

\[ p(y_* = 1 | x_*, X, y) = \int \sigma(w^\top x_*) \mathcal{N}(w | \mathbf{w}_{MAP}, H^{-1}) dw \]

- **Monte-Carlo approximation**: Draw several samples of \( w \) from \( \mathcal{N}(w | \mathbf{w}_{MAP}, H^{-1}) \) and replace the above integral by an empirical average of \( \sigma(w^\top x_*) \) computed using each of those samples

\[ p(y_* = 1 | x_*, X, y) \approx \frac{1}{S} \sum_{s=1}^{S} \sigma(w_s^\top x_*) \]

where \( w_s \sim \mathcal{N}(w | \mathbf{w}_{MAP}, H^{-1}), s = 1, \ldots, S \)
The posterior predictive is given by the following integral

\[ p(y_* = 1 | x_*, X, y) = \int \sigma(w^\top x_*) \mathcal{N}(w | \mathbf{w}_{\text{MAP}}, H^{-1}) dw \]

**Monte-Carlo approximation**: Draw several samples of \( w \) from \( \mathcal{N}(w | \mathbf{w}_{\text{MAP}}, H^{-1}) \) and replace the above integral by an empirical average of \( \sigma(w^\top x_*) \) computed using each of those samples

\[ p(y_* = 1 | x_*, X, y) \approx \frac{1}{S} \sum_{s=1}^{S} \sigma(w_s^\top x_*) \]

where \( w_s \sim \mathcal{N}(w | \mathbf{w}_{\text{MAP}}, H^{-1}) \), \( s = 1, \ldots, S \)

More on Monte-Carlo methods when we discuss MCMC sampling
The posterior predictive we wanted to compute was

\[ p(y_\star = 1|x_\star, X, y) \approx \int \sigma(w^\top x_\star) N(w|w_{MAP}, H^{-1}) \, dw \]
The posterior predictive we wanted to compute was

\[ p(y_\ast = 1|x_\ast, X, y) \approx \int \sigma(w^\top x_\ast) \mathcal{N}(w|w_{MAP}, H^{-1}) \, dw \]

In the above, let’s replace the sigmoid \( \sigma(w^\top x_\ast) \) by \( \Phi(w^\top x_\ast) \), i.e., CDF of standard normal

\[ \Phi(z) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{z} e^{-t^2} \, dt \quad \text{(Note: } z \text{ is a scalar and } 0 \leq \Phi(z) \leq 1) \]
The posterior predictive we wanted to compute was

\[ p(y^* = 1| x^*, X, y) \approx \int \sigma(w^T x^*) N(w|w_{MAP}, H^{-1}) dw \]

In the above, let’s replace the sigmoid \( \sigma(w^T x^*) \) by \( \Phi(w^T x^*) \), i.e., CDF of standard normal

\[ \Phi(z) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{z} e^{-t^2} dt \quad \text{(Note: \( z \) is a scalar and \( 0 \leq \Phi(z) \leq 1 \))} \]

Note: \( \Phi(z) \) is also called the **probit function**
The posterior predictive we wanted to compute was

\[ p(y_\star = 1|x_\star, X, y) \approx \int \sigma(w^\top x_\star)N(w|w_{MAP}, H^{-1}) \, dw \]

In the above, let’s replace the sigmoid \( \sigma(w^\top x_\star) \) by \( \Phi(w^\top x_\star) \), i.e., CDF of standard normal

\[ \Phi(z) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{z} e^{-t^2} \, dt \quad \text{(Note: } z \text{ is a scalar and } 0 \leq \Phi(z) \leq 1) \]

Note: \( \Phi(z) \) is also called the probit function
The posterior predictive we wanted to compute was

\[ p(y_\ast = 1|x_\ast, X, y) \approx \int \sigma(w^\top x_\ast)N(w|\omega_{\text{MAP}}, H^{-1})dw \]

In the above, let’s replace the sigmoid \( \sigma(w^\top x_\ast) \) by \( \Phi(w^\top x_\ast) \), i.e., CDF of standard normal

\[ \Phi(z) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{z} e^{-\frac{t^2}{2}} dt \quad \text{(Note: } z \text{ is a scalar and } 0 \leq \Phi(z) \leq 1) \]

Note: \( \Phi(z) \) is also called the **probit function**

This approach relies on numerical approximation (as we will see)
Predictive Posterior via Probit Approximation

- With this approximation, the predictive posterior will be

\[
p(y^*_r = 1|x^*_r, X, y) = \int \Phi(w^T x^*_r) N(w|w_{MAP}, H^{-1}) dw \quad \text{(an expectation)}
\]
Predictive Posterior via Probit Approximation

With this approximation, the predictive posterior will be

\[
p(y_\star = 1|x_\star, X, y) = \int \Phi(w^\top x_\star) N(w|w_{\text{MAP}}, H^{-1}) dw \quad \text{(an expectation)}
\]

\[
= \int_{-\infty}^{\infty} \Phi(a)p(a|\mu_a, \sigma_a^2) da \quad \text{(an equivalent expectation)}
\]

Since \(a = w^\top x_\star = x_\star^\top w\), and \(w\) is normally distributed, \(p(a|\mu_a, \sigma_a^2) = N(a|\mu_a, \sigma_a^2)\), with \(\mu_a = w^\top w_{\text{MAP}}x_\star\) and \(\sigma_a^2 = x_\star^\top H^{-1} x_\star\) (follows from the linear trans. property of random vars).

Given \(\mu_a = w^\top w_{\text{MAP}}x_\star\) and \(\sigma_a^2 = x_\star^\top H^{-1} x_\star\), the predictive posterior will be

\[
p(y_\star = 1|x_\star, X, y) \approx \int_{-\infty}^{\infty} \Phi(a)p(a|\mu_a, \sigma_a^2) da = \Phi(\mu_a \sqrt{1 + \sigma_a^2})
\]

Note that the variance \(\sigma_a^2\) also "moderates" the probability of \(y_\star\) being 1 (MAP would give \(\Phi(\mu_a)\)).

Since logistic and probit aren't exactly identical, we usually scale \(a\) by a scalar \(t\) s.t. \(t^2 = \pi/8\).
Predictive Posterior via Probit Approximation

With this approximation, the predictive posterior will be

\[
p(y_\ast = 1 | x_\ast, X, y) = \int \Phi(w^\top x_\ast) \mathcal{N}(w | w_{MAP}, H^{-1}) dw \quad \text{(an expectation)}
\]

\[
= \int_{-\infty}^{\infty} \Phi(a) p(a | \mu_a, \sigma_a^2) da \quad \text{(an equivalent expectation)}
\]

Since \( a = w^\top x_\ast = x_\ast^\top w \), and \( w \) is normally distributed, \( p(a | \mu_a, \sigma_a^2) = \mathcal{N}(a | \mu_a, \sigma_a^2) \), with \( \mu_a = w_{MAP}^\top x_\ast \) and \( \sigma_a^2 = x_\ast^\top H^{-1} x_\ast \) (follows from the linear trans. property of random vars)
Predictive Posterior via Probit Approximation

With this approximation, the predictive posterior will be

\[ p(y_\star = 1|x_\star, X, y) = \int \Phi(w^T x_\star) N(w|w_{MAP}, H^{-1}) dw \]  

(AN expectation)

\[ = \int_{-\infty}^{\infty} \Phi(a)p(a|\mu_a, \sigma_a^2) da \]  

(AN equivalent expectation)

Since \( a = w^T x_\star = x_\star^T w \), and \( w \) is normally distributed, \( p(a|\mu_a, \sigma_a^2) = N(a|\mu_a, \sigma_a^2) \), with \( \mu_a = w_{MAP}^T x_\star \) and \( \sigma_a^2 = x_\star^T H^{-1} x_\star \) (follows from the linear trans. property of random vars)

Given \( \mu_a = w_{MAP}^T x_\star \) and \( \sigma_a^2 = x_\star^T H^{-1} x_\star \), the predictive posterior will be

\[ p(y_\star = 1|x_\star, X, y) \approx \int_{-\infty}^{\infty} \Phi(a)N(a|\mu_a, \sigma_a^2) da = \Phi \left( \frac{\mu_a}{\sqrt{1 + \sigma_a^2}} \right) \]
Predictive Posterior via Probit Approximation

With this approximation, the predictive posterior will be

\[ p(y_* = 1|x_*, X, y) = \int \Phi(w^T x_*)N(w|w_{MAP}, H^{-1})dw \] (an expectation)

\[ = \int_{-\infty}^{\infty} \Phi(a)p(a|\mu_a, \sigma^2_a)da \] (an equivalent expectation)

Since \( a = w^T x_* = x_*^T w \), and \( w \) is normally distributed, \( p(a|\mu_a, \sigma^2_a) = N(a|\mu_a, \sigma^2_a) \), with \( \mu_a = w_{MAP}^T x_* \) and \( \sigma^2_a = x_*^T H^{-1} x_* \) (follows from the linear trans. property of random vars)

Given \( \mu_a = w_{MAP}^T x_* \) and \( \sigma^2_a = x_*^T H^{-1} x_* \), the predictive posterior will be

\[ p(y_* = 1|x_*, X, y) \approx \int_{-\infty}^{\infty} \Phi(a)N(a|\mu_a, \sigma^2_a)da = \Phi\left( \frac{\mu_a}{\sqrt{1 + \sigma^2_a}} \right) \]

Note that the variance \( \sigma^2_a \) also “moderates” the probability of \( y_n \) being 1 (MAP would give \( \Phi(\mu_a) \))
Predictive Posterior via Probit Approximation

With this approximation, the predictive posterior will be

\[ p(y^* = 1|x^*_*, X, y) = \int \Phi(w^T x^*_*) N(w|w_{MAP}, H^{-1}) dw \]  
  (an expectation)  

\[ = \int_{-\infty}^{\infty} \Phi(a) p(a|\mu_a, \sigma^2_a) da \]  
  (an equivalent expectation)  

Since \( a = w^T x^*_* = x^*_* \) \( w \), and \( w \) is normally distributed, \( p(a|\mu_a, \sigma^2_a) = N(a|\mu_a, \sigma^2_a) \), with \( \mu_a = w_{MAP}^T x^*_* \) and \( \sigma^2_a = x^*_* H^{-1} x^*_* \) (follows from the linear trans. property of random vars)  

Given \( \mu_a = w_{MAP}^T x^*_* \) and \( \sigma^2_a = x^*_* H^{-1} x^*_* \), the predictive posterior will be

\[ p(y^* = 1|x^*_*, X, y) \approx \int_{-\infty}^{\infty} \Phi(a) N(a|\mu_a, \sigma^2_a) da = \Phi\left( \frac{\mu_a}{\sqrt{1 + \sigma^2_a}} \right) \]  

Note that the variance \( \sigma^2_a \) also “moderates” the probability of \( y_n \) being 1 (MAP would give \( \Phi(\mu_a) \))  

Since logistic and probit aren’t exactly identical, we usually scale \( a \) by a scalar \( t \) s.t. \( t^2 = \pi/8 \)

\[ p(y^* = 1|x^*_*, X, y) = \int_{-\infty}^{\infty} \Phi(ta) N(a|\mu_a, \sigma^2_a) da = \Phi\left( \frac{\mu_a}{\sqrt{t^2 + \sigma^2_a}} \right) \]
Bayesian Logistic Regression: Posterior over Linear Classifiers!

Figure courtesy: MLAPP (Murphy)
Logistic Regression: Plug-in Prediction vs Bayesian Averaging

- (Left) Predictive distribution when using a point estimate uses only a single linear hyperplane $w$
- (Right) Posterior predictive distribution averages over many linear hyperplanes $w$
Some Comments

- We saw basic logistic regression model and some ways to perform Bayesian inference for this model.
Some Comments

- We saw basic logistic regression model and some ways to perform Bayesian inference for this model
  - We assumed the hyperparameters (e.g., precision/variance of $p(w) = \mathcal{N}(0, \lambda^{-1}I)$) to be fixed. However, these can also be learned if desired
Some Comments

- We saw basic logistic regression model and some ways to perform Bayesian inference for this model
  - We assumed the hyperparameters (e.g., precision/variance of $p(w) = \mathcal{N}(0, \lambda^{-1}I)$) to be fixed. However, these can also be learned if desired
  - LR is a linear classification model. Can be extended to nonlinear classification (more on this later)
Some Comments

- We saw basic logistic regression model and some ways to perform Bayesian inference for this model
  - We assumed the hyperparameters (e.g., precision/variance of $p(w) = \mathcal{N}(0, \lambda^{-1}I)$) to be fixed. However, these can also be learned if desired
  - LR is a linear classification model. Can be extended to nonlinear classification (more on this later)

- Logistic Regression (and its Bayesian version) is widely used in probabilistic classification
We saw basic logistic regression model and some ways to perform Bayesian inference for this model.

- We assumed the hyperparameters (e.g., precision/variance of $p(w) = \mathcal{N}(0, \lambda^{-1}I)$) to be fixed. However, these can also be learned if desired.
- LR is a linear classification model. Can be extended to nonlinear classification (more on this later).

Logistic Regression (and its Bayesian version) is widely used in probabilistic classification.

- Its multiclass extension is softmax regression (which again can be treated in a Bayesian manner).
Some Comments

- We saw basic logistic regression model and some ways to perform Bayesian inference for this model.
  - We assumed the hyperparameters (e.g., precision/variance of $p(w) = \mathcal{N}(0, \lambda^{-1}I)$) to be fixed. However, these can also be learned if desired.
  - LR is a linear classification model. Can be extended to nonlinear classification (more on this later).

- Logistic Regression (and its Bayesian version) is widely used in probabilistic classification.

- Its multiclass extension is softmax regression (which again can be treated in a Bayesian manner).

- LR and softmax some of the simplest models for discriminative classification but non-conjugate.
Some Comments

- We saw basic logistic regression model and some ways to perform Bayesian inference for this model.
  - We assumed the hyperparameters (e.g., precision/variance of $p(\mathbf{w}) = \mathcal{N}(0, \lambda^{-1}\mathbf{I})$) to be fixed. However, these can also be learned if desired.
  - LR is a linear classification model. Can be extended to nonlinear classification (more on this later).

- Logistic Regression (and its Bayesian version) is widely used in probabilistic classification.
- Its multiclass extension is softmax regression (which again can be treated in a Bayesian manner).
- LR and softmax some of the simplest models for discriminative classification but non-conjugate.
- The Laplace approximation is one of the simplest approximations to handle non-conjugacy.
Some Comments

- We saw basic logistic regression model and some ways to perform Bayesian inference for this model.
  - We assumed the hyperparameters (e.g., precision/variance of \( p(w) = \mathcal{N}(0, \lambda^{-1}I) \)) to be fixed. However, these can also be learned if desired.
  - LR is a linear classification model. Can be extended to nonlinear classification (more on this later).

- Logistic Regression (and its Bayesian version) is widely used in probabilistic classification.

- Its multiclass extension is **softmax regression** (which again can be treated in a Bayesian manner).

- LR and softmax some of the simplest models for discriminative classification but non-conjugate.

- The Laplace approximation is one of the simplest approximations to handle non-conjugacy.

- A variety of other approximate inference algorithms exist for these models.
Some Comments

- We saw basic logistic regression model and some ways to perform Bayesian inference for this model

  - We assumed the hyperparameters (e.g., precision/variance of \( p(w) = \mathcal{N}(0, \lambda^{-1}I) \)) to be fixed. However, these can also be learned if desired

  - LR is a linear classification model. Can be extended to nonlinear classification (more on this later)

- Logistic Regression (and its Bayesian version) is widely used in probabilistic classification

- Its multiclass extension is softmax regression (which again can be treated in a Bayesian manner)

- LR and softmax some of the simplest models for discriminative classification but non-conjugate

- The Laplace approximation is one of the simplest approximations to handle non-conjugacy

- A variety of other approximate inference algorithms exist for these models

  - We will revisit LR when discussing such approximate inference methods
Bayesian Generative Classification
Consider $N$ labeled examples $\{(x_i, y_i)\}_{n=1}^{N}$. Assume binary labels, i.e., $y_i \in \{0, 1\}$.
A Generative Model for Classification

- Consider $N$ labeled examples $\{(x_i, y_i)\}_{n=1}^N$. Assume binary labels, i.e., $y_i \in \{0, 1\}$
- Goal: Classify a new example $x$ by assigning a label $y \in \{0, 1\}$ to it

We will assume a Generative Model for both labels $y$ and features $x$.

What it means: We will have (probabilistic) observation models for both $y$ as well as $x$.

In contrast, in Bayesian linear regression model (and Bayesian logistic regression model), we didn’t model $x$ (there, we simply conditioned $y$ on $x$, treating $x$ as “fixed”).

When we don’t model $x$ and simply model $y$ as a function of $x$: Discriminative Model.

Generative classification models have many benefits. E.g.,
- Can also utilize unlabeled examples (semi-supervised learning)
- Can handle missing/corrupted features in $x$
- Can properly handle cases when features in $x$ could be of mixed type (e.g., real, binary, count)
- And many others (more on this later during the semester)
A Generative Model for Classification

- Consider $N$ labeled examples $\{(x_i, y_i)\}_{n=1}^{N}$. Assume binary labels, i.e., $y_i \in \{0, 1\}$
- Goal: Classify a new example $x$ by assigning a label $y \in \{0, 1\}$ to it
- We will assume a Generative Model for both labels $y$ and features $x$
A Generative Model for Classification

Consider \( N \) labeled examples \( \{(x_i, y_i)\}_{n=1}^{N} \). Assume binary labels, i.e., \( y_i \in \{0, 1\}\).

Goal: Classify a new example \( x \) by assigning a label \( y \in \{0, 1\} \) to it.

We will assume a Generative Model for both labels \( y \) and features \( x \).

What it means: We will have (probabilistic) observation models for both \( y \) as well as \( x \).
A Generative Model for Classification

- Consider $N$ labeled examples $\{(x_i, y_i)\}_{n=1}^N$. Assume binary labels, i.e., $y_i \in \{0, 1\}$
- Goal: Classify a new example $x$ by assigning a label $y \in \{0, 1\}$ to it
- We will assume a Generative Model for both labels $y$ and and features $x$
  - What it means: We will have (probabilistic) observation models for both $y$ as well as $x$
  - In contrast, in Bayesian linear regression model (and Bayesian logistic regression model), we didn’t model $x$ (there, we simply conditioned $y$ on $x$, treating $x$ as “fixed”)

Prob. Mod. & Inference - CS698X (Piyush Rai, IITK)
Consider $N$ labeled examples $\{(x_i, y_i)\}_{n=1}^{N}$. Assume binary labels, i.e., $y_i \in \{0, 1\}$.

Goal: Classify a new example $x$ by assigning a label $y \in \{0, 1\}$ to it.

We will assume a Generative Model for both labels $y$ and features $x$.

- What it means: We will have (probabilistic) observation models for both $y$ as well as $x$.
- In contrast, in Bayesian linear regression model (and Bayesian logistic regression model), we didn’t model $x$ (there, we simply conditioned $y$ on $x$, treating $x$ as “fixed”).
- When we don’t model $x$ and simply model $y$ as a function of $x$: Discriminative Model.

Generative classification models have many benefits. E.g.,

- Can also utilize unlabeled examples (semi-supervised learning).
- Can handle missing/corrupted features in $x$.
- Can properly handle cases when features in $x$ could be of mixed type (e.g., real, binary, count).
- And many others (more on this later during the semester).
A Generative Model for Classification

- Consider $N$ labeled examples $\{(x_i, y_i)\}_{n=1}^{N}$. Assume binary labels, i.e., $y_i \in \{0, 1\}$
- Goal: Classify a new example $x$ by assigning a label $y \in \{0, 1\}$ to it
- We will assume a Generative Model for both labels $y$ and and features $x$
  - What it means: We will have (probabilistic) observation models for both $y$ as well as $x$
  - In contrast, in Bayesian linear regression model (and Bayesian logistic regression model), we didn’t model $x$ (there, we simply conditioned $y$ on $x$, treating $x$ as “fixed”)
  - When we don’t model $x$ and simply model $y$ as a function of $x$: Discriminative Model
- Generative classification models have many benefits. E.g.,
A Generative Model for Classification

Consider $N$ labeled examples $\{(x_i, y_i)\}_{n=1}^{N}$. Assume binary labels, i.e., $y_i \in \{0, 1\}$

Goal: Classify a new example $x$ by assigning a label $y \in \{0, 1\}$ to it

We will assume a Generative Model for both labels $y$ and and features $x$

- What it means: We will have (probabilistic) observation models for both $y$ as well as $x$
- In contrast, in Bayesian linear regression model (and Bayesian logistic regression model), we didn’t model $x$ (there, we simply conditioned $y$ on $x$, treating $x$ as “fixed”)
- When we don’t model $x$ and simply model $y$ as a function of $x$: Discriminative Model

Generative classification models have many benefits. E.g.,

- Can also utilize unlabeled examples (semi-supervised learning)
A Generative Model for Classification

- Consider \( N \) labeled examples \( \{(x_i, y_i)\}_{n=1}^{N} \). Assume binary labels, i.e., \( y_i \in \{0, 1\} \)
- Goal: Classify a new example \( x \) by assigning a label \( y \in \{0, 1\} \) to it
- We will assume a Generative Model for both labels \( y \) and and features \( x \)
  - What it means: We will have (probabilistic) observation models for both \( y \) as well as \( x \)
  - In contrast, in Bayesian linear regression model (and Bayesian logistic regression model), we didn’t model \( x \) (there, we simply conditioned \( y \) on \( x \), treating \( x \) as “fixed”)
  - When we don’t model \( x \) and simply model \( y \) as a function of \( x \): Discriminative Model
- Generative classification models have many benefits. E.g.,
  - Can also utilize unlabeled examples (semi-supervised learning)
  - Can handle missing/corrupted features in \( x \)
A Generative Model for Classification

Consider \( N \) labeled examples \( \{(x_i, y_i)\}_{n=1}^{N} \). Assume binary labels, i.e., \( y_i \in \{0, 1\} \)

Goal: Classify a new example \( x \) by assigning a label \( y \in \{0, 1\} \) to it

We will assume a **Generative Model** for both labels \( y \) and and features \( x \)

- What it means: We will have (probabilistic) observation models for both \( y \) as well as \( x \)
- In contrast, in Bayesian linear regression model (and Bayesian logistic regression model), we didn’t model \( x \) (there, we simply conditioned \( y \) on \( x \), treating \( x \) as “fixed”)
- When we don’t model \( x \) and simply model \( y \) as a function of \( x \): **Discriminative Model**

Generative classification models have many benefits. E.g.,

- Can also utilize unlabeled examples (**semi-supervised learning**)  
- Can handle missing/corrupted features in \( x \)
- Can properly handle cases when features in \( x \) could be of mixed type (e.g., real, binary, count)
A Generative Model for Classification

- Consider $N$ labeled examples $\{(x_i; y_i)\}_{n=1}^N$. Assume binary labels, i.e., $y_i \in \{0, 1\}$
- Goal: Classify a new example $x$ by assigning a label $y \in \{0, 1\}$ to it
- We will assume a Generative Model for both labels $y$ and and features $x$
  - What it means: We will have (probabilistic) observation models for both $y$ as well as $x$
  - In contrast, in Bayesian linear regression model (and Bayesian logistic regression model), we didn’t model $x$ (there, we simply conditioned $y$ on $x$, treating $x$ as “fixed”)
  - When we don’t model $x$ and simply model $y$ as a function of $x$: Discriminative Model

- Generative classification models have many benefits. E.g.,
  - Can also utilize unlabeled examples (semi-supervised learning)
  - Can handle missing/corrupted features in $x$
  - Can properly handle cases when features in $x$ could be of mixed type (e.g., real, binary, count)
  - And many others (more on this later during the semester)
Generative Classification: The Generative Story

- Basic idea: Each $x_i$ is assumed generated conditioned on the value of corresponding label $y_i$. 

First draw ("generate") a binary label $y_i \in \{0, 1\}$, $y_i \sim \text{Bernoulli}(\pi)$.

Now draw ("generate") the feature vector $x_i$ from a distribution specific to the value $y_i$ takes, $x_i | y_i \sim p(x | \theta_{y_i})$.
Generative Classification: The Generative Story

- Basic idea: Each $x_i$ is assumed generated conditioned on the value of corresponding label $y_i$.
- The associated generative story is as follows:

  1. Draw a binary label $y_i \in \{0, 1\}$.
  2. $y_i \sim \text{Bernoulli}(\pi)$
  3. Draw the feature vector $x_i$ from a distribution specific to the value $y_i$ takes.
  4. $x_i | y_i \sim p(x | \theta_{y_i})$
Generative Classification: The Generative Story

- Basic idea: Each $x_i$ is assumed generated conditioned on the value of corresponding label $y_i$
- The associated generative story is as follows
  - First draw ("generate") a binary label $y_i \in \{0, 1\}$
    - $y_i \sim \text{Bernoulli}(\pi)$
Generative Classification: The Generative Story

- Basic idea: Each $x_i$ is assumed generated conditioned on the value of corresponding label $y_i$
- The associated generative story is as follows
  - First draw ("generate") a binary label $y_i \in \{0, 1\}$
    $$y_i \sim \text{Bernoulli}(\pi)$$
  - Now draw ("generate") the feature vector $x$ from a distribution specific to the value $y_i$ takes
    $$x_i|y_i \sim p(x|\theta_{y_i})$$
Generative Classification: The Generative Story

- Basic idea: Each $x_i$ is assumed generated conditioned on the value of corresponding label $y_i$
- The associated generative story is as follows
  - First draw ("generate") a binary label $y_i \in \{0, 1\}$
    \[
    y_i \sim \text{Bernoulli}(\pi)
    \]
  - Now draw ("generate") the feature vector $x$ from a distribution specific to the value $y_i$ takes
    \[
    x_i | y_i \sim p(x | \theta_{y_i})
    \]
- The above generative model shown in "plate notation" (shaded = observed)
A Generative Model for Classification

- Our generative model for classification is

  \[ y_i \sim \text{Bernoulli}(\pi), \quad x_i | y_i \sim p(x | \theta_{y_i}) \]

- Note: We have two distributions \( p(x | \theta_0) \) and \( p(x | \theta_1) \) for feature vector \( x \) (depending on its label)

Note: When \( y_i \) for each \( x_i \) is a hidden variable, we can think of it as the cluster id of \( x \). It then becomes a mixture model based data clustering problem (unsupervised learning).
A Generative Model for Classification

- Our generative model for classification is
  \[ y_i \sim \text{Bernoulli}(\pi), \quad x_i|y_i \sim p(x|\theta_{y_i}) \]

- Note: We have two distributions \( p(x|\theta_0) \) and \( p(x|\theta_1) \) for feature vector \( x \) (depending on its label)

- These distributions are also known as "class-conditional distributions"
A Generative Model for Classification

- Our generative model for classification is

\[ y_i \sim \text{Bernoulli}(\pi), \quad x_i|y_i \sim p(x|\theta_{y_i}) \]

- Note: We have two distributions \( p(x|\theta_0) \) and \( p(x|\theta_1) \) for feature vector \( x \) (depending on its label)

- These distributions are also known as "class-conditional distributions"

- For now, we will not assume any specific form for the distributions \( p(x|\theta_0) \) and \( p(x|\theta_1) \)
A Generative Model for Classification

- Our generative model for classification is
  \[ y_i \sim \text{Bernoulli}(\pi), \quad x_i | y_i \sim p(x | \theta_{y_i}) \]

  - Note: We have two distributions \( p(x | \theta_0) \) and \( p(x | \theta_1) \) for feature vector \( x \) (depending on its label)
  - These distributions are also known as "class-conditional distributions"
  - For now, we will not assume any specific form for the distributions \( p(x | \theta_0) \) and \( p(x | \theta_1) \)
    - Depends on nature of \( x \) (real-valued vectors? binary vectors? count vectors?)

Model parameters to be learned here: \((\pi, \theta_0, \theta_1)\)

Note: Can extend to more than 2 classes (e.g., by replacing the Bernoulli on \( y \) by multinoulli)
A Generative Model for Classification

- Our generative model for classification is

  \[ y_i \sim \text{Bernoulli}(\pi), \quad x_i|y_i \sim p(x|\theta_{y_i}) \]

- Note: We have two distributions \( p(x|\theta_0) \) and \( p(x|\theta_1) \) for feature vector \( x \) (depending on its label)

- These distributions are also known as "class-conditional distributions"

- For now, we will not assume any specific form for the distributions \( p(x|\theta_0) \) and \( p(x|\theta_1) \)
  
  - Depends on nature of \( x \) (real-valued vectors? binary vectors? count vectors?)

- Model parameters to be learned here: \((\pi, \theta_0, \theta_1)\)
A Generative Model for Classification

- Our generative model for classification is
  
  \[ y_i \sim \text{Bernoulli}(\pi), \quad x_i|y_i \sim p(x|\theta_{y_i}) \]

- Note: We have two distributions \( p(x|\theta_0) \) and \( p(x|\theta_1) \) for feature vector \( x \) (depending on its label)

- These distributions are also known as "class-conditional distributions"

- For now, we will not assume any specific form for the distributions \( p(x|\theta_0) \) and \( p(x|\theta_1) \)
  - Depends on nature of \( x \) (real-valued vectors? binary vectors? count vectors?)

- Model parameters to be learned here: \( (\pi, \theta_0, \theta_1) \)

- Note: Can extend to more than 2 classes (e.g., by replacing the Bernoulli on \( y \) by multinoulli)
A Generative Model for Classification

- Our generative model for classification is

\[ y_i \sim \text{Bernoulli}(\pi), \quad x_i|y_i \sim p(x|\theta_{y_i}) \]

- Note: We have two distributions \( p(x|\theta_0) \) and \( p(x|\theta_1) \) for feature vector \( x \) (depending on its label)

- These distributions are also known as "class-conditional distributions"

- For now, we will not assume any specific form for the distributions \( p(x|\theta_0) \) and \( p(x|\theta_1) \)
  - Depends on nature of \( x \) (real-valued vectors? binary vectors? count vectors?)

- Model parameters to be learned here: \( (\pi, \theta_0, \theta_1) \)

- Note: Can extend to more than 2 classes (e.g., by replacing the Bernoulli on \( y \) by multinoulli)

- Note: When \( y_i \) for each \( x_i \) is a hidden variable, we can think of it as the cluster id of \( x \)
A Generative Model for Classification

- Our generative model for classification is
  \[ y_i \sim \text{Bernoulli}(\pi), \quad x_i | y_i \sim p(x | \theta_{y_i}) \]

- Note: We have two distributions \( p(x | \theta_0) \) and \( p(x | \theta_1) \) for feature vector \( x \) (depending on its label)

- These distributions are also known as “class-conditional distributions”

- For now, we will not assume any specific form for the distributions \( p(x | \theta_0) \) and \( p(x | \theta_1) \)
  - Depends on nature of \( x \) (real-valued vectors? binary vectors? count vectors?)

- Model parameters to be learned here: \((\pi, \theta_0, \theta_1)\)

- Note: Can extend to more than 2 classes (e.g., by replacing the Bernoulli on \( y \) by multinoulli)

- Note: When \( y_i \) for each \( x_i \) is a hidden variable, we can think of it as the cluster id of \( x \)
  - It then becomes a mixture model based data clustering problem (unsupervised learning)
Predicting Labels in Generative Classification

- Note: The generative model only defines \( p(y|\pi) \) and \( p(x|\theta_y) \). Doesn’t define \( p(y|x) \)
Predicting Labels in Generative Classification

- Note: The generative model only defines $p(y|\pi)$ and $p(x|\theta_y)$. Doesn’t define $p(y|x)$

- We combine these using Bayes rule to get $p(y|x)$

$$p(y|x) = \frac{p(y|\pi)p(x|\theta_y)}{p(x)} = \frac{p(y|\pi)p(x|\theta_y)}{\sum_y p(y|\pi)p(x|\theta_y)}$$
Predicting Labels in Generative Classification

- Note: The generative model only defines $p(y|\pi)$ and $p(x|\theta_y)$. Doesn’t define $p(y|x)$.

- We combine these using Bayes rule to get $p(y|x)$

$$p(y|x) = \frac{p(y|\pi)p(x|\theta_y)}{p(x)} = \frac{p(y|\pi)p(x|\theta_y)}{\sum_y p(y|\pi)p(x|\theta_y)}$$

- Parameters of distributions $p(y|\pi)$ and $p(x|\theta_y)$ are estimated from training data using point estimation methods (MLE or MAP) or using fully Bayesian inference (discussed today).
Predicting Labels in Generative Classification

- Note: The generative model only defines \( p(y|\pi) \) and \( p(x|\theta_y) \). Doesn’t define \( p(y|x) \)

- We combine these using Bayes rule to get \( p(y|x) \)

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

- Parameters of distributions \( p(y|\pi) \) and \( p(x|\theta_y) \) are estimated from training data using point estimation methods (MLE or MAP) or using fully Bayesian inference (discussed today)

- Once these parameters \( \pi \) and \( \theta_y \) are estimated (point estimates, or full posterior if doing Bayesian inference), the above Bayes rule can be applied to a new input \( \hat{x} \) to compute \( p(\hat{y}|\hat{x}) \)
Predicting Labels in Generative Classification

- Note: The generative model only defines $p(y|\pi)$ and $p(x|\theta_y)$. Doesn’t define $p(y|x)$
- We combine these using Bayes rule to get $p(y|x)$

$$p(y|x) = \frac{p(y|\pi)p(x|\theta_y)}{p(x)} = \frac{p(y|\pi)p(x|\theta_y)}{\sum_y p(y|\pi)p(x|\theta_y)}$$

- Parameters of distributions $p(y|\pi)$ and $p(x|\theta_y)$ are estimated from training data using point estimation methods (MLE or MAP) or using **fully Bayesian inference** (discussed today)
- Once these parameters $\pi$ and $\theta_y$ are estimated (point estimates, or full posterior if doing Bayesian inference), the above Bayes rule can be applied to a new input $\hat{x}$ to compute $p(\hat{y}|\hat{x})$
- Let’s now set up the parameter estimation for $\pi$ and $\theta_y$ as a Bayesian inference problem
Predicting Labels in Generative Classification

- Note: The generative model only defines \( p(y|\pi) \) and \( p(x|\theta_y) \). Doesn’t define \( p(y|x) \)

- We combine these using Bayes rule to get \( p(y|x) \)

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

- Parameters of distributions \( p(y|\pi) \) and \( p(x|\theta_y) \) are estimated from training data using point estimation methods (MLE or MAP) or using fully Bayesian inference (discussed today)

- Once these parameters \( \pi \) and \( \theta_y \) are estimated (point estimates, or full posterior if doing Bayesian inference), the above Bayes rule can be applied to a new input \( \hat{x} \) to compute \( p(\hat{y}|\hat{x}) \)

- Let’s now set up the parameter estimation for \( \pi \) and \( \theta_y \) as a Bayesian inference problem

  - Note: As we will see in the end, in this approach, computing \( p(\hat{y}|\hat{x}) \) for a new input \( \hat{x} \) will NOT use a point estimate of the parameters \( \pi, \theta_y \) but would use posterior averaging
The Priors

Let us focus on the supervised, binary classification setting for now.
The Priors

- Let us focus on the supervised, binary classification setting for now.
- In this case, we have three parameters to be learned: $\pi$, $\theta_0$, and $\theta_1$. 

Probability $\pi \in (0, 1)$ of the Bernoulli. Can assume the following Beta prior $\pi \sim \text{Beta}(a, b)$.

Parameters $\theta_0$ and $\theta_1$ of the class-conditional distributions. Will assume the same prior on both $\theta_0, \theta_1 \sim p(\theta)$.

Note: The actual form of $p(\theta)$ will depend on what the class conditional distributions $p(x | \theta_0)$ and $p(x | \theta_1)$ are (e.g., if these are Gaussians and if we want to learn both mean and covariance matrix of these Gaussians, then $p(\theta)$ will be some distribution over mean and covariance matrix, e.g., a Normal-inverse Wishart distribution).

We will jointly denote the prior on $\pi$, $\theta_0$, and $\theta_1$ as $p(\pi, \theta_0, \theta_1) = p(\pi) p(\theta_0) p(\theta_1)$. 

Prob. Mod. & Inference - CS698X (Piyush Rai, IITK)
The Priors

- Let us focus on the supervised, binary classification setting for now.
- In this case, we have three parameters to be learned: $\pi$, $\theta_0$, and $\theta_1$.
  - Probability $\pi \in (0, 1)$ of the Bernoulli. Can assume the following Beta prior:
    $$\pi \sim \text{Beta}(a, b)$$
The Priors

- Let us focus on the supervised, binary classification setting for now
- In this case, we have three parameters to be learned: \( \pi \), \( \theta_0 \), and \( \theta_1 \)
  - Probability \( \pi \in (0, 1) \) of the Bernoulli. Can assume the following Beta prior
    \[
    \pi \sim \text{Beta}(a, b)
    \]
  - Parameters \( \theta_0 \), and \( \theta_1 \) of the class-conditional distributions. Will assume the same prior on both
    \[
    \theta_0, \theta_1 \sim p(\theta)
    \]
The Priors

- Let us focus on the supervised, binary classification setting for now

- In this case, we have three parameters to be learned: $\pi$, $\theta_0$, and $\theta_1$
  - Probability $\pi \in (0, 1)$ of the Bernoulli. Can assume the following Beta prior
    \[ \pi \sim \text{Beta}(a, b) \]
  - Parameters $\theta_0$, and $\theta_1$ of the class-conditional distributions. Will assume the same prior on both
    \[ \theta_0, \theta_1 \sim p(\theta) \]

- Note: The actual form of $p(\theta)$ will depend on what the class conditional distributions $p(x|\theta_0)$ and $p(x|\theta_1)$ are (e.g., if these are Gaussians and if we want to learn both mean and covariance matrix of these Gaussians, then $p(\theta)$ will be some distribution over mean and covariance matrix, e.g., a Normal-inverse Wishart distribution)
Let us focus on the supervised, binary classification setting for now.

In this case, we have three parameters to be learned: $\pi$, $\theta_0$, and $\theta_1$.

- Probability $\pi \in (0, 1)$ of the Bernoulli. Can assume the following Beta prior
  \[ \pi \sim \text{Beta}(a, b) \]

- Parameters $\theta_0$, and $\theta_1$ of the class-conditional distributions. Will assume the same prior on both
  \[ \theta_0, \theta_1 \sim p(\theta) \]

Note: The actual form of $p(\theta)$ will depend on what the class conditional distributions $p(x|\theta_0)$ and $p(x|\theta_1)$ are (e.g., if these are Gaussians and if we want to learn both mean and covariance matrix of these Gaussians, then $p(\theta)$ will be some distribution over mean and covariance matrix, e.g., a Normal-inverse Wishart distribution).

We will jointly denote the prior on $\pi$, $\theta_0$, and $\theta_1$ as $p(\pi, \theta_0, \theta_1) = p(\pi)p(\theta_0)p(\theta_1)$.
Denote the $N \times D$ feature matrix by $X$ and the $N \times 1$ label vector by $y$. 
Denote the $N \times D$ feature matrix by $X$ and the $N \times 1$ label vector by $y$.

Since both $X$ and $y$ are being modeled here, the likelihood function will be
The Likelihood

- Denote the $N \times D$ feature matrix by $X$ and the $N \times 1$ label vector by $y$.
- Since both $X$ and $y$ are being modeled here, the likelihood function will be

$$p(X, y|\pi, \theta_1, \theta_0) = \prod_{i=1}^{N} p(x_i, y_i|\pi, \theta_1, \theta_0)$$

$$= \prod_{i=1}^{N} p(x_i|y_i, \pi, \theta_1, \theta_0)p(y_i|\pi, \theta_1, \theta_0)$$

$$= \prod_{i=1}^{N} p(x_i|\theta_{y_i})p(y_i|\pi)$$
The Posterior

- We need to infer the following posterior distribution

\[
p(\pi, \theta_1, \theta_0 | \tilde{y}, X) = \frac{p(X, \tilde{y} | \pi, \theta_1, \theta_0)p(\pi, \theta_1, \theta_0)}{\int_{\Omega_\theta} \int_{\Omega_\theta} \int_0^1 p(X, \tilde{y} | \pi, \theta_1, \theta_0)p(\pi, \theta_1, \theta_0) d\pi d\theta_1 d\theta_0}
\]

- Note: $\Omega_\theta$ denotes the domain of $\theta$
The Posterior

- We need to infer the following posterior distribution

\[ p(\pi, \theta_1, \theta_0 | y, X) = \frac{p(X, \bar{y} | \pi, \theta_1, \theta_0)p(\pi, \theta_1, \theta_0)}{\int_{\Omega_{\theta}} \int_{\Omega_{\theta}} \int_0^1 p(X, \bar{y} | \pi, \theta_1, \theta_0)p(\pi, \theta_1, \theta_0) d\pi d\theta_1 d\theta_0} \]

- Note: \( \Omega_{\theta} \) denotes the domain of \( \theta \)
- Might look scary at first but it isn’t actually
The Posterior

- We need to infer the following posterior distribution

\[
p(\pi, \theta_1, \theta_0 | \vec{y}, X) = \frac{p(X, \vec{y} | \pi, \theta_1, \theta_0)p(\pi, \theta_1, \theta_0)}{\int_{\Omega_\theta} \int_{\Omega_\theta} \int_0^1 p(X, \vec{y} | \pi, \theta_1, \theta_0)p(\pi, \theta_1, \theta_0)d\pi d\theta_1 d\theta_0}
\]

- Note: \( \Omega_\theta \) denotes the domain of \( \theta \)
- Might look scary at first but it isn’t actually
- Recall the prior \( p(\pi, \theta_0, \theta_1) = p(\pi)p(\theta_0)p(\theta_1) \).
We need to infer the following posterior distribution

\[
p(\pi, \theta_1, \theta_0 | \mathbf{y}, X) = \frac{p(X, \mathbf{y} | \pi, \theta_1, \theta_0) p(\pi, \theta_1, \theta_0)}{\int_{\Omega_\theta} \int_{\Omega_\theta} \int_0^1 p(X, \mathbf{y} | \pi, \theta_1, \theta_0) p(\pi, \theta_1, \theta_0) d\pi d\theta_1 d\theta_0}
\]

Note: \( \Omega_\theta \) denotes the domain of \( \theta \)

Might look scary at first but it isn’t actually

Recall the prior \( p(\pi, \theta_0, \theta_1) = p(\pi)p(\theta_0)p(\theta_1) \). The likelihood also factorized over data points, i.e.,

\[
p(X, \mathbf{y} | \pi, \theta_1, \theta_0) = \prod_{i=1}^N p(x_i | \theta_{y_i}) p(y_i | \pi)
\]
The Posterior

- We need to infer the following posterior distribution

\[
p(\pi, \theta_1, \theta_0 | \vec{y}, X) = \frac{p(X, \vec{y} | \pi, \theta_1, \theta_0) p(\pi, \theta_1, \theta_0)}{\int_{\Omega_\theta} \int_{\Omega_\theta} \int_0^1 p(X, \vec{y} | \pi, \theta_1, \theta_0) p(\pi, \theta_1, \theta_0) \, d\pi \, d\theta_1 \, d\theta_0}
\]

- Note: $\Omega_\theta$ denotes the domain of $\theta$
- Might look scary at first but it isn’t actually
- Recall the prior $p(\pi, \theta_0, \theta_1) = p(\pi) p(\theta_0) p(\theta_1)$. The likelihood also factorized over data points, i.e.,

\[
p(X, \vec{y} | \pi, \theta_1, \theta_0) = \prod_{i=1}^N p(x_i | \theta_{y_i}) p(y_i | \pi)
\]
- Thus, the posterior will be

\[
p(\pi, \theta_1, \theta_0 | \vec{y}, X) \propto \left[ \prod_{i:y_i=1} p(x_i | \theta_1) p(\theta_1) \right] \left[ \prod_{i:y_i=0} p(x_i | \theta_0) p(\theta_0) \right] \left[ \prod_{i=1}^N p(y_i | \pi) p(\pi) \right]
\]
The Posterior

- We need to infer the following posterior distribution

\[
p(\pi, \theta_1, \theta_0 | \vec{y}, X) = \frac{p(X, \vec{y} | \pi, \theta_1, \theta_0)p(\pi, \theta_1, \theta_0)}{\int_{\Omega_{\theta}} \int_{\Omega_{\theta}} \int_{\mathbb{R}^1} p(X, \vec{y} | \pi, \theta_1, \theta_0)p(\pi, \theta_1, \theta_0)d\pi d\theta_1 d\theta_0}
\]

- Note: $\Omega_{\theta}$ denotes the domain of $\theta$
- Might look scary at first but it isn’t actually
- Recall the prior $p(\pi, \theta_0, \theta_1) = p(\pi)p(\theta_0)p(\theta_1)$. The likelihood also factorized over data points, i.e.,

\[
p(X, y | \pi, \theta_1, \theta_0) = \prod_{i=1}^{N} p(x_i | \theta_{y_i})p(y_i | \pi)
\]

- Thus, the posterior will be

\[
p(\pi, \theta_1, \theta_0 | \vec{y}, X) \propto \left[ \prod_{i:y_i=1} p(x_i | \theta_1)p(\theta_1) \right] \left[ \prod_{i:y_i=0} p(x_i | \theta_0)p(\theta_0) \right] \left[ \prod_{i=1}^{N} p(y_i | \pi)p(\pi) \right]
\]

- But what about the normalization constant in the denominator?
The Posterior

- Luckily, in this case, the same factorization structure simplifies the denominator as well

\[
p(\pi, \theta_1, \theta_0 | \bar{y}, X) = \frac{\prod_{i:y_i=1} p(x_i | \theta_1)p(\theta_1)}{\int \prod_{i:y_i=1} p(x_i | \theta_1)p(\theta_1)d\theta_1} \cdot \frac{\prod_{i:y_i=0} p(x_i | \theta_0)p(\theta_0)}{\int \prod_{i:y_i=0} p(x_i | \theta_0)p(\theta_0)d\theta_0} \cdot \frac{\prod_{i=1}^N p(y_i | \pi)p(\pi)}{\int \prod_{i=1}^N p(y_i | \pi)p(\pi)d\pi}
\]
The Posterior

- Luckily, in this case, the same factorization structure simplifies the denominator as well

\[
p(\pi, \theta_1, \theta_0 | \tilde{y}, X) = \frac{\prod_{i:y_i=1} p(x_i | \theta_1) p(\theta_1)}{\int \prod_{i:y_i=1} p(x_i | \theta_1) p(\theta_1) d\theta_1} \cdot \frac{\prod_{i:y_i=0} p(x_i | \theta_0) p(\theta_0)}{\int \prod_{i:y_i=0} p(x_i | \theta_0) p(\theta_0) d\theta_0} \cdot \frac{\prod_{i=1}^{N} p(y_i | \pi) p(\pi)}{\int \prod_{i=1}^{N} p(y_i | \pi) p(\pi) d\pi}
\]

- The above is just a product of three posterior distributions!

\[
p(\pi, \theta_1, \theta_0 | \tilde{y}, X) = p(\theta_1 | \{x_i : y_i = 1\}) p(\theta_0 | \{x_i : y_i = 0\}) p(\pi | \tilde{y})
\]
The Posterior

- Luckily, in this case, the same factorization structure simplifies the denominator as well

\[ p(\pi, \theta_1, \theta_0 | \bar{y}, X) = \frac{\prod_{i:y_i=1} p(x_i|\theta_1)p(\theta_1)}{\int \prod_{i:y_i=1} p(x_i|\theta_1)p(\theta_1)d\theta_1} \cdot \frac{\prod_{i:y_i=0} p(x_i|\theta_0)p(\theta_0)}{\int \prod_{i:y_i=0} p(x_i|\theta_0)p(\theta_0)d\theta_0} \cdot \frac{\prod_{i=1}^N p(y_i|\pi)p(\pi)}{\int \prod_{i=1}^N p(y_i|\pi)p(\pi)d\pi} \]

- The above is just a product of three posterior distributions!

\[ p(\pi, \theta_1, \theta_0 | \bar{y}, X) = p(\theta_1 | \{x_i : y_i = 1\})p(\theta_0 | \{x_i : y_i = 0\})p(\pi | \bar{y}) \]

- We also know what \( p(\pi | y) \) will be (recall the coin-toss example)

\[ p(\pi | \bar{y}) \propto \prod_{i=1}^N p(y_i | \pi)p(\pi) \quad \rightarrow \quad p(\pi | \bar{y}) = \text{Beta}(a + \sum_i y_i, b + N - \sum_i y_i) \]
The Posterior

• Luckily, in this case, the same factorization structure simplifies the denominator as well

\[ p(\pi, \theta_1, \theta_0 | \vec{y}, X) = \frac{\prod_{i:y_i=1} p(x_i | \theta_1)p(\theta_1)}{\int \prod_{i:y_i=1} p(x_i | \theta_1)p(\theta_1) d\theta_1} \cdot \frac{\prod_{i:y_i=0} p(x_i | \theta_0)p(\theta_0)}{\int \prod_{i:y_i=0} p(x_i | \theta_0)p(\theta_0) d\theta_0} \cdot \frac{\prod_{i=1}^{N} p(y_i | \pi)p(\pi)}{\int \prod_{i=1}^{N} p(y_i | \pi)p(\pi) d\pi} \]

• The above is just a product of three posterior distributions!

\[ p(\pi, \theta_1, \theta_0 | \vec{y}, X) = p(\theta_1 | \{x_i : y_i = 1\})p(\theta_0 | \{x_i : y_i = 0\})p(\pi | \vec{y}) \]

• We also know what \( p(\pi | y) \) will be (recall the coin-toss example)

\[ p(\pi | \vec{y}) \propto \prod_{i=1}^{N} p(y_i | \pi)p(\pi) \quad \rightarrow \quad p(\pi | \vec{y}) = \text{Beta}(a + \sum_i y_i, b + N - \sum_i y_i) \]

• Form of posteriors on \( \theta_1 \) and \( \theta_2 \) will depend on \( p(x | \theta_1) \) and \( p(\theta_1) \), and \( p(x | \theta_0) \) and \( p(\theta_0) \), resp.
The Predictive Posterior Distribution

- We have already seen how to compute the parameter posterior \( p(\pi, \theta_1, \theta_0 | y, X) \) for this model.
The Predictive Posterior Distribution

- We have already seen how to compute the parameter posterior $p(\pi, \theta_1, \theta_0 | y, X)$ for this model.
- Original goal is classification. We thus also want the predictive posterior for label of a new input, i.e., $p(\hat{y} | \hat{x})$, for which the more “complete” notation in this Bayesian setting would be $p(\hat{y} | \hat{x}, X, y)$.

\[
p(\hat{y} | \hat{x}, X, y) = \int_{\Omega_\theta} \int_{\Omega_\theta} \int_0^1 p(\hat{y} | \hat{x}, \theta_1, \theta_0, \pi) p(\theta_1, \theta_0, \pi | X, y) d\pi d\theta_1 d\theta_0
\]
The Predictive Posterior Distribution

- We have already seen how to compute the parameter posterior \( p(\pi, \theta_1, \theta_0 | y, X) \) for this model.

- Original goal is classification. We thus also want the predictive posterior for label of a new input, i.e., \( p(\hat{y} | \hat{x}) \), for which the more “complete” notation in this Bayesian setting would be \( p(\hat{y} | \hat{x}, X, y) \).

\[
p(\hat{y} | \hat{x}, X, y) = \int_{\Omega_{\theta}} \int_{\Omega_{\theta}} \int_0^1 p(\hat{y} | \hat{x}, \theta_1, \theta_0, \pi) p(\theta_1, \theta_0, \pi | X, y) d\pi d\theta_1 d\theta_0
\]

- Luckily, in this case, this too has a rather simple form. Using Bayes rule, we have

\[
p(\hat{y} | \hat{x}, X, y) = \frac{p(\hat{x} | \hat{y}, X, y)p(\hat{y} | X, y)}{p(\hat{x} | \hat{y} = 1, X, y)p(\hat{y} = 1 | X, y) + p(\hat{x} | \hat{y} = 0, X, y)p(\hat{y} = 0 | X, y)}
\]
The Predictive Posterior Distribution

- We have already seen how to compute the parameter posterior $p(\pi, \theta_1, \theta_0 | y, X)$ for this model.

- Original goal is classification. We thus also want the predictive posterior for label of a new input, i.e., $p(\hat{y} | \hat{x})$, for which the more “complete” notation in this Bayesian setting would be $p(\hat{y} | \hat{x}, X, y)$.

\[
p(\hat{y} | \hat{x}, X, \bar{y}) = \int_{\Omega_\theta} \int_{\Omega_\theta} \int_0^1 p(\hat{y} | \hat{x}, \theta_1, \theta_0, \pi) p(\theta_1, \theta_0, \pi | X, \bar{y}) d\pi d\theta_1 d\theta_0
\]

- Luckily, in this case, this too has a rather simple form. Using Bayes rule, we have

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

\[
= \frac{p(\hat{x} | \hat{y}, X, \bar{y}) p(\hat{y} | \bar{y})}{p(\hat{x} | \hat{y} = 1, X, \bar{y}) p(\hat{y} = 1 | \bar{y}) + p(\hat{x} | \hat{y} = 0, X, \bar{y}) p(\hat{y} = 0 | \bar{y})}
\]
The Predictive Posterior Distribution

- We have already seen how to compute the parameter posterior \( p(\pi, \theta_1, \theta_0 | y, X) \) for this model.

- Original goal is classification. We thus also want the predictive posterior for label of a new input, i.e., \( p(\hat{y} | \hat{x}) \), for which the more “complete” notation in this Bayesian setting would be \( p(\hat{y} | \hat{x}, X, y) \).

\[
p(\hat{y} | \hat{x}, X, y) = \int_{\Omega_\theta} \int_{\Omega_\pi} \int_0^1 p(\hat{y} | \hat{x}, \theta_1, \theta_0, \pi) p(\theta_1, \theta_0, \pi | X, y) d\pi d\theta_1 d\theta_0
\]

- Luckily, in this case, this too has a rather simple form. Using Bayes rule, we have

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

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

- In order to compute this, we need \( p(\hat{x} | \hat{y}, X, y) \) and \( p(\hat{y} | y) \).
The Predictive Posterior Distribution

- We have already seen how to compute the parameter posterior \( p(\pi, \theta_1, \theta_0 | y, X) \) for this model.

- Original goal is classification. We thus also want the predictive posterior for label of a new input, i.e., \( p(\hat{y}|\hat{x}) \), for which the more “complete” notation in this Bayesian setting would be \( p(\hat{y}|x, y) \).

\[
p(\hat{y}|\hat{x}, X, y) = \int_{\Omega_\theta} \int_{\Omega_\theta} \int_0^1 p(\hat{y}|\hat{x}, \theta_1, \theta_0, \pi) p(\theta_1, \theta_0, \pi | X, y) d\pi d\theta_1 d\theta_0
\]

- Luckily, in this case, this too has a rather simple form. Using Bayes rule, we have

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

- In order to compute this, we need \( p(\hat{x}|\hat{y}, X, y) \) and \( p(\hat{y}|y) \):

  - \( p(\hat{x}|\hat{y}, X, y) \): Marginal class-conditional distribution of the new input vector \( \hat{x} \)
The Predictive Posterior Distribution

- We have already seen how to compute the parameter posterior $p(\pi, \theta_1, \theta_0 | y, X)$ for this model.

- Original goal is classification. We thus also want the predictive posterior for label of a new input, i.e., $p(\hat{y} | \hat{x})$, for which the more “complete” notation in this Bayesian setting would be $p(\hat{y} | \hat{x}, X, y)$.

\[
    p(\hat{y} | \hat{x}, X, y) = \int_{\Omega_\theta} \int_{\Omega_\theta} \int_0^1 p(\hat{y} | \hat{x}, \theta_1, \theta_0, \pi)p(\theta_1, \theta_0, \pi | X, y)d\pi d\theta_1 d\theta_0
\]

- Luckily, in this case, this too has a rather simple form. Using Bayes rule, we have

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

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

- In order to compute this, we need $p(\hat{x} | \hat{y}, X, y)$ and $p(\hat{y} | y)$:
  - $p(\hat{x} | \hat{y}, X, y)$: Marginal class-conditional distribution of the new input vector $\hat{x}$
  - $p(\hat{y} | y)$: Marginal probability of its label $\hat{y}$ given the labels of training data.
The Predictive Posterior Distribution (Contd.)

- Predictive posterior requires computing $p(\hat{x}|\hat{y}, X, y)$ and $p(\hat{y}|y)$
The Predictive Posterior Distribution (Contd.)

- Predictive posterior requires computing $p(\hat{x}|\hat{y}, X, y)$ and $p(\hat{y}|y)$

- The marginal likelihood $p(\hat{x}|\hat{y}, X, y)$ of $\hat{x}$ can be computed as

$$p(\hat{x}|\hat{y}, X, y) = \int_{\Omega_\theta} \int_{\Omega_\theta} p(\hat{x}|\hat{y}, \theta_1, \theta_0)p(\theta_1, \theta_0|X, \bar{y})d\theta_1d\theta_0$$

$$= \int_{\Omega_\theta} p(\hat{x}|\theta_0)p(\theta_0|\{x_i : y_i = \hat{y}\})d\theta_0$$
Predictive posterior requires computing $p(\hat{x}|\hat{y}, X, y)$ and $p(\hat{y}|y)$.

The marginal likelihood $p(\hat{x}|\hat{y}, X, y)$ of $\hat{x}$ can be computed as

$$p(\hat{x}|\hat{y}, X, y) = \int_{\Omega_{\hat{y}}} \int_{\Omega_{\theta}} p(\hat{x}|\hat{y}, \theta_1, \theta_0)p(\theta_1, \theta_0|X, \hat{y})d\theta_1d\theta_0$$

$$= \int_{\Omega_{\hat{y}}} p(\hat{x}|\theta_{\hat{y}})p(\theta_{\hat{y}}|\{x_i : y_i = \hat{y}\})d\theta_{\hat{y}}$$

The above is simply the posterior predictive distribution of class $\hat{y}$. The final expression will depend on the forms of $p(\hat{x}|\theta_{\hat{y}})$ and $p(\theta_{\hat{y}}|.)$. If exp-family, we will have closed form expression!
The Predictive Posterior Distribution (Contd.)

- Predictive posterior requires computing \( p(\hat{x}|\hat{y}, X, y) \) and \( p(\hat{y}|y) \)

- The marginal likelihood \( p(\hat{x}|\hat{y}, X, y) \) of \( \hat{x} \) can be computed as

\[
p(\hat{x}|\hat{y}, X, y) = \int_{\Omega_\theta} \int_{\Omega_{\theta_0}} p(\hat{x}|\hat{y}, \theta_1, \theta_0) p(\theta_1, \theta_0|X, y) d\theta_1 d\theta_0
\]

\[
= \int_{\Omega_{\theta_0}} p(\hat{x}|\theta_{\hat{y}}) p(\theta_{\hat{y}}|\{x_i : y_i = \hat{y}\}) d\theta_{\hat{y}}
\]

The above is simply the posterior predictive distribution of class \( \hat{y} \). The final expression will depend on the forms of \( p(\hat{x}|\theta_{\hat{y}}) \) and \( p(\theta_{\hat{y}}|.) \). If exp-family, we will have closed form expression!

- The marginal likelihood \( p(\hat{y}|y) \) is something we have already seen (recall Bernoulli coin-toss)

\[
p(\hat{y} = 1|y) = \int p(\hat{y} = 1|\pi) p(\pi|y) d\pi = \int \pi p(\pi|y) d\pi = \frac{a + \sum_{i=1}^{N} y_i}{a + b + N}
\]
The Predictive Posterior Distribution (Contd.)

- Predictive posterior requires computing \( p(\hat{x} | \hat{y}, X, y) \) and \( p(\hat{y} | y) \)
- The marginal likelihood \( p(\hat{x} | \hat{y}, X, y) \) of \( \hat{x} \) can be computed as
  \[
  p(\hat{x} | \hat{y}, X, y) = \int_{\Omega_0} \int_{\Omega_0} p(\hat{x} | \hat{y}, \theta_1, \theta_0) p(\theta_1, \theta_0 | X, y) d\theta_1 d\theta_0
  \]
  \[
  = \int_{\Omega_0} p(\hat{x} | \theta_1) p(\theta_1 | \{x_i : y_i = \hat{y}\}) d\theta_1
  \]

  The above is simply the posterior predictive distribution of class \( \hat{y} \). The final expression will depend on the forms of \( p(\hat{x} | \theta_\hat{y}) \) and \( p(\theta_\hat{y} | .) \). If exp-family, we will have closed form expression!

- The marginal likelihood \( p(\hat{y} | y) \) is something we have already seen (recall Bernoulli coin-toss)
  \[
  p(\hat{y} = 1 | y) = \int p(\hat{y} = 1 | \pi) p(\pi | y) d\pi = \int \pi p(\pi | y) d\pi = a + \sum_{i=1}^{N} y_i \overline{a + b + N}
  \]
  \[
  \Rightarrow \quad p(\hat{y} = 0 | y) = 1 - p(\hat{y} = 1 | y) = \frac{b + N - \sum_{i=1}^{N} y_i}{a + b + N}
  \]
A Simple/Special Case: Naïve Bayes Assumption

Usually the most critical choice in generative classification is that of class conditional $p(x|\theta_y)$.
A Simple/Special Case: Naïve Bayes Assumption

- Usually the most critical choice in generative classification is that of class conditional $p(x|\theta_y)$
- Very complex $p(x|\theta_y)$ with lots of parameters may make estimation difficult
A Simple/Special Case: Naïve Bayes Assumption

- Usually the most critical choice in generative classification is that of class conditional $p(x|\theta_y)$
- Very complex $p(x|\theta_y)$ with lots of parameters may make estimation difficult
- Often however we can choose simple forms of $p(x|\theta_y)$ to make estimation easier
A Simple/Special Case: Naïve Bayes Assumption

- Usually the most critical choice in generative classification is that of class conditional $p(x|\theta_y)$
- Very complex $p(x|\theta_y)$ with lots of parameters may make estimation difficult
- Often however we can choose simple forms of $p(x|\theta_y)$ to make estimation easier
- The naïve Bayes assumption: The conditional distribution $p(x|\theta_y)$ factorizes over individual features (or over groups of features)
A Simple/Special Case: Naïve Bayes Assumption

- Usually the most critical choice in generative classification is that of class conditional $p(x|\theta_y)$
- Very complex $p(x|\theta_y)$ with lots of parameters may make estimation difficult
- Often however we can choose simple forms of $p(x|\theta_y)$ to make estimation easier
- The naïve Bayes assumption: The conditional distribution $p(x|\theta_y)$ factorizes over individual features (or over groups of features)
  - Suppose the features of $\hat{x}$ can be partitioned into $v$ groups $\hat{x} = \{\hat{x}(j)\}_{j=1}^{v}$
A Simple/Special Case: Naïve Bayes Assumption

- Usually the most critical choice in generative classification is that of class conditional $p(x|\theta_y)$
- Very complex $p(x|\theta_y)$ with lots of parameters may make estimation difficult
- Often however we can choose simple forms of $p(x|\theta_y)$ to make estimation easier
- The naïve Bayes assumption: The conditional distribution $p(x|\theta_y)$ factorizes over individual features (or over groups of features)
  - Suppose the features of $\hat{x}$ can be partitioned into $v$ groups $\hat{x} = \{\hat{x}(j)\}_{j=1}^v$
  - Can also assume a similar partitioning for the parameters $\theta_{\hat{y}}$
A Simple/Special Case: Naïve Bayes Assumption

- Usually the most critical choice in generative classification is that of class conditional \( p(x|\theta_y) \)
- Very complex \( p(x|\theta_y) \) with lots of parameters may make estimation difficult
- Often however we can choose simple forms of \( p(x|\theta_y) \) to make estimation easier
- The naïve Bayes assumption: The conditional distribution \( p(x|\theta_y) \) factorizes over individual features (or over groups of features)
  - Suppose the features of \( \hat{x} \) can be partitioned into \( v \) groups \( \hat{x} = \{ \hat{x}(j) \}_{j=1}^{v} \)
  - Can also assume a similar partitioning for the parameters \( \theta_{\hat{y}} \)
  - This further simplifies calculation of marginal likelihood \( p(\hat{x}|\hat{y}, X, y) \)

\[
p(\hat{x}|\hat{y}, X, y) = \int_{\Omega_{\theta}} \prod_{j=1}^{v} p(\hat{x}(j)|\theta_{\hat{y}}(j)) p(\theta_{\hat{y}}(j)|\{x_i(j) : y_i = \hat{y}\}) d\theta_{\hat{y}}
\]

\[
= \prod_{j=1}^{v} \int p(\hat{x}(j)|\theta_{\hat{y}}(j)) p(\theta_{\hat{y}}(j)|\{x_i(j) : y_i = \hat{y}\}) d\theta_{\hat{y}}(j)
\]
A Simple/Special Case: Naïve Bayes Assumption

- Usually the most critical choice in generative classification is that of class conditional $p(x|\theta_y)$
- Very complex $p(x|\theta_y)$ with lots of parameters may make estimation difficult
- Often however we can choose simple forms of $p(x|\theta_y)$ to make estimation easier
- The naïve Bayes assumption: The conditional distribution $p(x|\theta_y)$ factorizes over individual features (or over groups of features)
  - Suppose the features of $\hat{x}$ can be partitioned into $v$ groups $\hat{x} = \{\hat{x}(j)\}_{j=1}^{v}$
  - Can also assume a similar partitioning for the parameters $\theta_{\hat{y}}$
  - This further simplifies calculation of marginal likelihood $p(\hat{x}|\hat{y}, X, y)$

\[
p(\hat{x}|\hat{y}, X, \bar{y}) = \int_{\Omega_{\theta}} \prod_{j=1}^{v} p(\hat{x}(j)|\theta_{\hat{y}}(j)) p(\theta_{\hat{y}}(j)|\{x_i(j) : y_i = \hat{y}\}) d\theta_{\hat{y}}
\]
\[
= \prod_{j=1}^{v} \int p(\hat{x}(j)|\theta_{\hat{y}}(j)) p(\theta_{\hat{y}}(j)|\{x_i(j) : y_i = \hat{y}\}) d\theta_{\hat{y}}(j)
\]

- This modeling choice in a Bayesian setting gives rise to a “Bayesian naïve Bayes” model
In the Bayesian naïve Bayes model, we can still choose different types of class conditional $p(x|\theta_y)$
In the Bayesian naïve Bayes model, we can still choose different types of class conditional $p(x|\theta_y)$.

- Gaussian naïve Bayes: if $x$ is modeled using a multivariate Gaussian (assumed factorized as per the naïve Bayes assumption)
A Simple/Special Case: Naïve Bayes Assumption

In the Bayesian naïve Bayes model, we can still choose different types of class conditional \( p(x|\theta_y) \)

- Gaussian naïve Bayes: if \( x \) is modeled using a multivariate Gaussian (assumed factorized as per the naïve Bayes assumption)
- Multivariate Bernoulli naïve Bayes: if \( x \) is modeled using a multivariate Bernoulli (assumed factorized as per the naïve Bayes assumption)
A Simple/Special Case: Naïve Bayes Assumption

- In the Bayesian naïve Bayes model, we can still choose different types of class conditional $p(x|\theta_y)$
  - Gaussian naïve Bayes: if $x$ is modeled using a multivariate Gaussian (assumed factorized as per the naïve Bayes assumption)
  - Multivariate Bernoulli naïve Bayes: if $x$ is modeled using a multivariate Bernoulli (assumed factorized as per the naïve Bayes assumption)
- MLAPP (Murphy) Section 3.5.1.2 and 3.5.5 contains an example of Multivariate Bernoulli case