Gaussian Processes for Learning Nonlinear Functions

Piyush Rai

Topics in Probabilistic Modeling and Inference (CS698X)

Jan 28, 2019

Announcements

- Discussion session on project topics/ideas: Tomorrow 7pm-8pm (KD-101)

Announcements

- Discussion session on project topics/ideas: Tomorrow 7pm-8pm (KD-101)
- Project proposals due on Feb 1

Announcements

- Discussion session on project topics/ideas: Tomorrow 7pm-8pm (KD-101)
- Project proposals due on Feb 1
- HW1 out now. Due on Feb 8, 11:59pm. Please start early.

Announcements

- Discussion session on project topics/ideas: Tomorrow 7pm-8pm (KD-101)
- Project proposals due on Feb 1
- HW1 out now. Due on Feb 8, 11:59pm. Please start early.
- Quiz 1 on Jan 31, 7pm-8pm (RM-101)

Recap: Bayesian Generative Classification

- Recall generative classification $p(y=k|x) = \frac{p(y=k)p(x|y=k)}{\sum_{k=1}^{K} p(y=k)p(x|y=k)}$

Recap: Bayesian Generative Classification

- Recall generative classification $p(y = k|x) = \frac{p(y=k)p(x|y=k)}{\sum_{k=1}^{K} p(y=k)p(x|y=k)}$. Prediction rule for a test input $\boldsymbol{x}_*$

$$p(y_* = k|\boldsymbol{x}_*, \mathbf{X}, \boldsymbol{y})$$

Recap: Bayesian Generative Classification

- Recall generative classification $p(y=k|x) = \frac{p(y=k)p(x|y=k)}{\sum_{k=1}^{K} p(y=k)p(x|y=k)}$. Prediction rule for a test input $\boldsymbol{x}_*$

$$p(y_* = k|\boldsymbol{x}_*, \mathbf{X}, \boldsymbol{y}) = \frac{p(y_* = k|\mathbf{X}, \boldsymbol{y})p(\boldsymbol{x}_*|y_* = k, \mathbf{X}, \boldsymbol{y})}{\sum_{k=1}^{K} p(y_* = k|\mathbf{X}, \boldsymbol{y})p(\boldsymbol{x}_*|y_* = k, \mathbf{X}, \boldsymbol{y})}$$

Recap: Bayesian Generative Classification

- Recall generative classification $p(y = k|x) = \frac{p(y=k)p(x|y=k)}{\sum_{k=1}^{K} p(y=k)p(x|y=k)}$. Prediction rule for a test input $\boldsymbol{x}_*$

$$\begin{aligned} p(y_* = k|\boldsymbol{x}_*, \mathbf{X}, \boldsymbol{y}) &= \frac{p(y_* = k|\mathbf{X}, \boldsymbol{y})p(\boldsymbol{x}_*|y_* = k, \mathbf{X}, \boldsymbol{y})}{\sum_{k=1}^{K} p(y_* = k|\mathbf{X}, \boldsymbol{y})p(\boldsymbol{x}_*|y_* = k, \mathbf{X}, \boldsymbol{y})} \\ &= \frac{p(y_* = k|\boldsymbol{y})p(\boldsymbol{x}_*|\mathbf{X}^{(k)})}{\sum_{k=1}^{K} p(y_* = k|\boldsymbol{y})p(\boldsymbol{x}_*|\mathbf{X}^{(k)})} \end{aligned} \quad (1)$$

Recap: Bayesian Generative Classification

- Recall generative classification $p(y = k|x) = \frac{p(y=k)p(x|y=k)}{\sum_{k=1}^{K} p(y=k)p(x|y=k)}$. Prediction rule for a test input $\boldsymbol{x}_*$

$$
\begin{aligned}
p(y_* = k|\boldsymbol{x}_*, \mathbf{X}, \boldsymbol{y}) &= \frac{p(y_* = k|\mathbf{X}, \boldsymbol{y})p(\boldsymbol{x}_*|y_* = k, \mathbf{X}, \boldsymbol{y})}{\sum_{k=1}^{K} p(y_* = k|\mathbf{X}, \boldsymbol{y})p(\boldsymbol{x}_*|y_* = k, \mathbf{X}, \boldsymbol{y})} \\
&= \frac{p(y_* = k|\boldsymbol{y})p(\boldsymbol{x}_*|\mathbf{X}^{(k)})}{\sum_{k=1}^{K} p(y_* = k|\boldsymbol{y})p(\boldsymbol{x}_*|\mathbf{X}^{(k)})} \qquad (1)
\end{aligned}
$$

- Note: $\mathbf{X}^{(k)}$ denotes training inputs with $y = k$

Recap: Bayesian Generative Classification

- Recall generative classification $p(y = k|x) = \frac{p(y=k)p(x|y=k)}{\sum_{k=1}^{K} p(y=k)p(x|y=k)}$. Prediction rule for a test input $\boldsymbol{x}_*$

$$
\begin{aligned}
p(y_* = k|\boldsymbol{x}_*, \mathbf{X}, \boldsymbol{y}) &= \frac{p(y_* = k|\mathbf{X}, \boldsymbol{y})p(\boldsymbol{x}_*|y_* = k, \mathbf{X}, \boldsymbol{y})}{\sum_{k=1}^{K} p(y_* = k|\mathbf{X}, \boldsymbol{y})p(\boldsymbol{x}_*|y_* = k, \mathbf{X}, \boldsymbol{y})} \\
&= \frac{p(y_* = k|\boldsymbol{y})p(\boldsymbol{x}_*|\mathbf{X}^{(k)})}{\sum_{k=1}^{K} p(y_* = k|\boldsymbol{y})p(\boldsymbol{x}_*|\mathbf{X}^{(k)})} \qquad (1)
\end{aligned}
$$

- Note: $\mathbf{X}^{(k)}$ denotes training inputs with $y = k$
- Here $p(y_* = k|\boldsymbol{y}) = \int p(y_*|\pi)p(\pi|\boldsymbol{y})d\pi$ (we did this; recall dice roll example)

Recap: Bayesian Generative Classification

- Recall generative classification $p(y=k|x) = \frac{p(y=k)p(x|y=k)}{\sum_{k=1}^{K} p(y=k)p(x|y=k)}$. Prediction rule for a test input $\boldsymbol{x}_*$

$$
\begin{aligned}
p(y_* = k|\boldsymbol{x}_*, \mathbf{X}, \boldsymbol{y}) &= \frac{p(y_* = k|\mathbf{X}, \boldsymbol{y})p(\boldsymbol{x}_*|y_* = k, \mathbf{X}, \boldsymbol{y})}{\sum_{k=1}^{K} p(y_* = k|\mathbf{X}, \boldsymbol{y})p(\boldsymbol{x}_*|y_* = k, \mathbf{X}, \boldsymbol{y})} \\
&= \frac{p(y_* = k|\boldsymbol{y})p(\boldsymbol{x}_*|\mathbf{X}^{(k)})}{\sum_{k=1}^{K} p(y_* = k|\boldsymbol{y})p(\boldsymbol{x}_*|\mathbf{X}^{(k)})} \qquad (1)
\end{aligned}
$$

- Note: $\mathbf{X}^{(k)}$ denotes training inputs with $y = k$
- Here $p(y_* = k|\boldsymbol{y}) = \int p(y_*|\pi)p(\pi|\boldsymbol{y})d\pi$ (we did this; recall dice roll example)
- Here $p(\boldsymbol{x}_*|\mathbf{X}^{(k)}) = \int p(\boldsymbol{x}_*|\theta_k)p(\theta_k|\mathbf{X}^{(k)})d\theta_k$ (post. predictive dist. of input $\boldsymbol{x}_*$ under class k)

Recap: Bayesian Generative Classification

- Recall generative classification $p(y=k|x) = \frac{p(y=k)p(x|y=k)}{\sum_{k=1}^{K} p(y=k)p(x|y=k)}$. Prediction rule for a test input $\boldsymbol{x}_*$

$$
\begin{aligned}
p(y_* = k|\boldsymbol{x}_*, \mathbf{X}, \boldsymbol{y}) &= \frac{p(y_* = k|\mathbf{X}, \boldsymbol{y})p(\boldsymbol{x}_*|y_* = k, \mathbf{X}, \boldsymbol{y})}{\sum_{k=1}^{K} p(y_* = k|\mathbf{X}, \boldsymbol{y})p(\boldsymbol{x}_*|y_* = k, \mathbf{X}, \boldsymbol{y})} \\
&= \frac{p(y_* = k|\boldsymbol{y})p(\boldsymbol{x}_*|\mathbf{X}^{(k)})}{\sum_{k=1}^{K} p(y_* = k|\boldsymbol{y})p(\boldsymbol{x}_*|\mathbf{X}^{(k)})} \qquad (1)
\end{aligned}
$$

- Note: $\mathbf{X}^{(k)}$ denotes training inputs with $y = k$
- Here $p(y_* = k|\boldsymbol{y}) = \int p(y_*|\pi)p(\pi|\boldsymbol{y})d\pi$ (we did this; recall dice roll example)
- Here $p(\boldsymbol{x}_*|\mathbf{X}^{(k)}) = \int p(\boldsymbol{x}_*|\theta_k)p(\theta_k|\mathbf{X}^{(k)})d\theta_k$ (post. predictive dist. of input $\boldsymbol{x}_*$ under class k)
- Eq (1) is the posterior predictive distribution of test output y_* given input $\boldsymbol{x}_*$

Recap: Bayesian Generative Classification

- Recall generative classification $p(y=k|x) = \frac{p(y=k)p(x|y=k)}{\sum_{k=1}^{K} p(y=k)p(x|y=k)}$. Prediction rule for a test input $\boldsymbol{x}_*$

$$
\begin{aligned}
p(y_* = k|\boldsymbol{x}_*, \mathbf{X}, \boldsymbol{y}) &= \frac{p(y_* = k|\mathbf{X}, \boldsymbol{y})p(\boldsymbol{x}_*|y_* = k, \mathbf{X}, \boldsymbol{y})}{\sum_{k=1}^{K} p(y_* = k|\mathbf{X}, \boldsymbol{y})p(\boldsymbol{x}_*|y_* = k, \mathbf{X}, \boldsymbol{y})} \\
&= \frac{p(y_* = k|\boldsymbol{y})p(\boldsymbol{x}_*|\mathbf{X}^{(k)})}{\sum_{k=1}^{K} p(y_* = k|\boldsymbol{y})p(\boldsymbol{x}_*|\mathbf{X}^{(k)})} \qquad (1)
\end{aligned}
$$

- Note: $\mathbf{X}^{(k)}$ denotes training inputs with $y = k$
- Here $p(y_* = k|\boldsymbol{y}) = \int p(y_*|\pi)p(\pi|\boldsymbol{y})d\pi$ (we did this; recall dice roll example)
- Here $p(\boldsymbol{x}_*|\mathbf{X}^{(k)}) = \int p(\boldsymbol{x}_*|\theta_k)p(\theta_k|\mathbf{X}^{(k)})d\theta_k$ (post. predictive dist. of input $\boldsymbol{x}_*$ under class k)
- Eq (1) is the posterior predictive distribution of test output y_* given input $\boldsymbol{x}_*$
  - Note that we have done posterior averaging for all the parameters

Recap: Bayesian Generative Classification

- Recall generative classification $p(y=k|x) = \frac{p(y=k)p(x|y=k)}{\sum_{k=1}^K p(y=k)p(x|y=k)}$. Prediction rule for a test input $\boldsymbol{x}_*$

$$\begin{aligned} p(y_* = k|\boldsymbol{x}_*, \mathbf{X}, \boldsymbol{y}) &= \frac{p(y_* = k|\mathbf{X}, \boldsymbol{y})p(\boldsymbol{x}_*|y_* = k, \mathbf{X}, \boldsymbol{y})}{\sum_{k=1}^K p(y_* = k|\mathbf{X}, \boldsymbol{y})p(\boldsymbol{x}_*|y_* = k, \mathbf{X}, \boldsymbol{y})} \\ &= \frac{p(y_* = k|\boldsymbol{y})p(\boldsymbol{x}_*|\mathbf{X}^{(k)})}{\sum_{k=1}^K p(y_* = k|\boldsymbol{y})p(\boldsymbol{x}_*|\mathbf{X}^{(k)})} \end{aligned} \tag{1}$$

- Note: $\mathbf{X}^{(k)}$ denotes training inputs with $y = k$
- Here $p(y_* = k|\boldsymbol{y}) = \int p(y_*|\pi)p(\pi|\boldsymbol{y})d\pi$ (we did this; recall dice roll example)
- Here $p(\boldsymbol{x}_*|\mathbf{X}^{(k)}) = \int p(\boldsymbol{x}_*|\theta_k)p(\theta_k|\mathbf{X}^{(k)})d\theta_k$ (post. predictive dist. of input $\boldsymbol{x}_*$ under class k)
- Eq (1) is the posterior predictive distribution of test output y_* given input $\boldsymbol{x}_*$
  - Note that we have done posterior averaging for all the parameters

Recap: Bayesian Generative Classification

- Recall generative classification $p(y=k|x) = \frac{p(y=k)p(x|y=k)}{\sum_{k=1}^{K} p(y=k)p(x|y=k)}$. Prediction rule for a test input $\boldsymbol{x}_*$

$$
\begin{aligned}
p(y_* = k|\boldsymbol{x}_*, \mathbf{X}, \boldsymbol{y}) &= \frac{p(y_* = k|\mathbf{X}, \boldsymbol{y})p(\boldsymbol{x}_*|y_* = k, \mathbf{X}, \boldsymbol{y})}{\sum_{k=1}^{K} p(y_* = k|\mathbf{X}, \boldsymbol{y})p(\boldsymbol{x}_*|y_* = k, \mathbf{X}, \boldsymbol{y})} \\
&= \frac{p(y_* = k|\boldsymbol{y})p(\boldsymbol{x}_*|\mathbf{X}^{(k)})}{\sum_{k=1}^{K} p(y_* = k|\boldsymbol{y})p(\boldsymbol{x}_*|\mathbf{X}^{(k)})} \qquad (1)
\end{aligned}
$$

- Note: $\mathbf{X}^{(k)}$ denotes training inputs with $y = k$
- Here $p(y_* = k|\boldsymbol{y}) = \int p(y_*|\pi)p(\pi|\boldsymbol{y})d\pi$ (we did this; recall dice roll example)
- Here $p(\boldsymbol{x}_*|\mathbf{X}^{(k)}) = \int p(\boldsymbol{x}_*|\theta_k)p(\theta_k|\mathbf{X}^{(k)})d\theta_k$ (post. predictive dist. of input $\boldsymbol{x}_*$ under class k)
- Eq (1) is the posterior predictive distribution of test output y_* given input $\boldsymbol{x}_*$
  - Note that we have done posterior averaging for all the parameters
- In contrast, for gen. class with MLE/MAP, $p(y_* = k|\boldsymbol{y}) \approx \pi_k$ and $p(\boldsymbol{x}_*|\mathbf{X}^{(k)}) \approx p(\boldsymbol{x}_*|\theta_k)$

Gaussian Processes (GP)

(GP = Bayesian Modeling + Kernel Methods)

(Goal: learning **nonlinear** discriminative models $p(y|\boldsymbol{x})$)

Linear Models

- Consider the problem of learning to map an input $\boldsymbol{x} \in \mathbb{R}^D$ to an output y

Linear Models

- Consider the problem of learning to map an input $\boldsymbol{x} \in \mathbb{R}^D$ to an output y
- Linear models use a weighted combination of input features (i.e., $\boldsymbol{w}^\top \boldsymbol{x}$) to generate y

Linear Models

- Consider the problem of learning to map an input $\boldsymbol{x} \in \mathbb{R}^D$ to an output y
- Linear models use a weighted combination of input features (i.e., $\boldsymbol{w}^\top \boldsymbol{x}$) to generate y

$$p(y|\boldsymbol{w}, \boldsymbol{x}) = \mathcal{N}(y|\boldsymbol{w}^\top \boldsymbol{x}, \beta^{-1}) \qquad \text{(Linear Regression)}$$

Linear Models

- Consider the problem of learning to map an input $\boldsymbol{x} \in \mathbb{R}^D$ to an output y
- Linear models use a weighted combination of input features (i.e., $\boldsymbol{w}^\top \boldsymbol{x}$) to generate y

$$
\begin{aligned}
p(y|\boldsymbol{w}, \boldsymbol{x}) &= \mathcal{N}(y|\boldsymbol{w}^\top \boldsymbol{x}, \beta^{-1}) && \text{(Linear Regression)} \\
p(y|\boldsymbol{w}, \boldsymbol{x}) &= [\sigma(\boldsymbol{w}^\top \boldsymbol{x})]^y [1 - \sigma(\boldsymbol{w}^\top \boldsymbol{x})]^{1-y} && \text{(Logistic Regression)}
\end{aligned}
$$

Linear Models

- Consider the problem of learning to map an input $\boldsymbol{x} \in \mathbb{R}^D$ to an output y
- Linear models use a weighted combination of input features (i.e., $\boldsymbol{w}^\top \boldsymbol{x}$) to generate y

$$\begin{aligned} p(y|\boldsymbol{w}, \boldsymbol{x}) &= \mathcal{N}(y|\boldsymbol{w}^\top \boldsymbol{x}, \beta^{-1}) && \text{(Linear Regression)} \\ p(y|\boldsymbol{w}, \boldsymbol{x}) &= [\sigma(\boldsymbol{w}^\top \boldsymbol{x})]^y [1 - \sigma(\boldsymbol{w}^\top \boldsymbol{x})]^{1-y} && \text{(Logistic Regression)} \\ p(y|\boldsymbol{w}, \boldsymbol{x}) &= \text{ExpFam}(\boldsymbol{w}^\top \boldsymbol{x}) && \text{(Generalized Linear Model)} \end{aligned}$$

Linear Models

- Consider the problem of learning to map an input $\boldsymbol{x} \in \mathbb{R}^D$ to an output y
- Linear models use a weighted combination of input features (i.e., $\boldsymbol{w}^\top \boldsymbol{x}$) to generate y

$$
\begin{aligned}
p(y|\boldsymbol{w}, \boldsymbol{x}) &= \mathcal{N}(y|\boldsymbol{w}^\top \boldsymbol{x}, \beta^{-1}) && \text{(Linear Regression)} \\
p(y|\boldsymbol{w}, \boldsymbol{x}) &= [\sigma(\boldsymbol{w}^\top \boldsymbol{x})]^y [1 - \sigma(\boldsymbol{w}^\top \boldsymbol{x})]^{1-y} && \text{(Logistic Regression)} \\
p(y|\boldsymbol{w}, \boldsymbol{x}) &= \text{ExpFam}(\boldsymbol{w}^\top \boldsymbol{x}) && \text{(Generalized Linear Model)}
\end{aligned}
$$

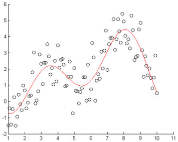
- The weights $\boldsymbol{w}$ can be learned using MLE, MAP, or fully Bayesian inference

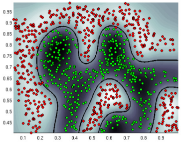
Linear Models

- Consider the problem of learning to map an input $\boldsymbol{x} \in \mathbb{R}^D$ to an output y
- Linear models use a weighted combination of input features (i.e., $\boldsymbol{w}^\top \boldsymbol{x}$) to generate y

$$
\begin{aligned}
p(y|\boldsymbol{w}, \boldsymbol{x}) &= \mathcal{N}(y|\boldsymbol{w}^\top \boldsymbol{x}, \beta^{-1}) && \text{(Linear Regression)} \\
p(y|\boldsymbol{w}, \boldsymbol{x}) &= [\sigma(\boldsymbol{w}^\top \boldsymbol{x})]^y [1 - \sigma(\boldsymbol{w}^\top \boldsymbol{x})]^{1-y} && \text{(Logistic Regression)} \\
p(y|\boldsymbol{w}, \boldsymbol{x}) &= \text{ExpFam}(\boldsymbol{w}^\top \boldsymbol{x}) && \text{(Generalized Linear Model)}
\end{aligned}
$$

- The weights $\boldsymbol{w}$ can be learned using MLE, MAP, or fully Bayesian inference
- However, linear models have limited expressive power. Unable to learn highly nonlinear patterns.


Nonlinear Regression


Nonlinear Classification

Modeling Nonlinear Functions

- Assume the input to output relationship to be modeled by a nonlinear function f

Modeling Nonlinear Functions

- Assume the input to output relationship to be modeled by a nonlinear function f

$$p(y|f, \boldsymbol{x}) = \mathcal{N}(y|f(\boldsymbol{x}), \beta^{-1})$$

Modeling Nonlinear Functions

- Assume the input to output relationship to be modeled by a nonlinear function f

$$
\begin{aligned}
p(y|f, \boldsymbol{x}) &= \mathcal{N}(y|f(\boldsymbol{x}), \beta^{-1}) \\
p(y|f, \boldsymbol{x}) &= [\sigma(f(\boldsymbol{x}))]^{y}[1 - \sigma(f(\boldsymbol{x}))]^{1-y}
\end{aligned}
$$

Modeling Nonlinear Functions

- Assume the input to output relationship to be modeled by a nonlinear function f

$$
\begin{aligned}
p(y|f,\boldsymbol{x}) &= \mathcal{N}(y|f(\boldsymbol{x}),\beta^{-1}) \\
p(y|f,\boldsymbol{x}) &= [\sigma(f(\boldsymbol{x}))]^{y}[1-\sigma(f(\boldsymbol{x}))]^{1-y} \\
p(y|f,\boldsymbol{x}) &= \text{ExpFam}(f(\boldsymbol{x}))
\end{aligned}
$$

Modeling Nonlinear Functions

- Assume the input to output relationship to be modeled by a nonlinear function f

$$\begin{aligned} p(y|f,\boldsymbol{x}) &= \mathcal{N}(y|f(\boldsymbol{x}),\beta^{-1}) \\ p(y|f,\boldsymbol{x}) &= [\sigma(f(\boldsymbol{x}))]^y[1-\sigma(f(\boldsymbol{x}))]^{1-y} \\ p(y|f,\boldsymbol{x}) &= \text{ExpFam}(f(\boldsymbol{x})) \end{aligned}$$

- How can we define such a function nonlinear f?

Modeling Nonlinear Functions

- Assume the input to output relationship to be modeled by a nonlinear function f

$$
\begin{aligned}
p(y|f, \boldsymbol{x}) &= \mathcal{N}(y|f(\boldsymbol{x}), \beta^{-1}) \\
p(y|f, \boldsymbol{x}) &= [\sigma(f(\boldsymbol{x}))]^y [1 - \sigma(f(\boldsymbol{x}))]^{1-y} \\
p(y|f, \boldsymbol{x}) &= \text{ExpFam}(f(\boldsymbol{x}))
\end{aligned}
$$

- How can we define such a function nonlinear f?
- Note: We not only want nonlinearity but also all benefits of probabilistic/Bayesian modeling

Modeling Nonlinear Functions

- Assume the input to output relationship to be modeled by a nonlinear function f

$$
\begin{aligned}
p(y|f,\boldsymbol{x}) &= \mathcal{N}(y|f(\boldsymbol{x}),\beta^{-1}) \\
p(y|f,\boldsymbol{x}) &= [\sigma(f(\boldsymbol{x}))]^{y}[1-\sigma(f(\boldsymbol{x}))]^{1-y} \\
p(y|f,\boldsymbol{x}) &= \text{ExpFam}(f(\boldsymbol{x}))
\end{aligned}
$$

- How can we define such a function nonlinear f?
- Note: We not only want nonlinearity but also all benefits of probabilistic/Bayesian modeling
  - Must be able to get uncertainty estimates in the function and its predictions

Modeling Nonlinear Functions

- Assume the input to output relationship to be modeled by a nonlinear function f

$$
\begin{aligned}
p(y|f,\boldsymbol{x}) &= \mathcal{N}(y|f(\boldsymbol{x}),\beta^{-1}) \\
p(y|f,\boldsymbol{x}) &= [\sigma(f(\boldsymbol{x}))]^y[1-\sigma(f(\boldsymbol{x}))]^{1-y} \\
p(y|f,\boldsymbol{x}) &= \text{ExpFam}(f(\boldsymbol{x}))
\end{aligned}
$$

- How can we define such a function nonlinear f?
- Note: We not only want nonlinearity but also all benefits of probabilistic/Bayesian modeling
  - Must be able to get uncertainty estimates in the function and its predictions
- Usually done in one of the following ways

Modeling Nonlinear Functions

- Assume the input to output relationship to be modeled by a nonlinear function f

$$
\begin{aligned}
p(y|f,\boldsymbol{x}) &= \mathcal{N}(y|f(\boldsymbol{x}),\beta^{-1}) \\
p(y|f,\boldsymbol{x}) &= [\sigma(f(\boldsymbol{x}))]^{y}[1-\sigma(f(\boldsymbol{x}))]^{1-y} \\
p(y|f,\boldsymbol{x}) &= \text{ExpFam}(f(\boldsymbol{x}))
\end{aligned}
$$

- How can we define such a function nonlinear f?
- Note: We not only want nonlinearity but also all benefits of probabilistic/Bayesian modeling
  - Must be able to get uncertainty estimates in the function and its predictions
- Usually done in one of the following ways
  - Ad-hoc: Define nonlinear features $\phi(\boldsymbol{x})$ + train Bayesian linear model ($f(\boldsymbol{x}) = \boldsymbol{w}^\top\phi(\boldsymbol{x})$): HW1

Modeling Nonlinear Functions

- Assume the input to output relationship to be modeled by a nonlinear function f

$$\begin{aligned} p(y|f,\boldsymbol{x}) &= \mathcal{N}(y|f(\boldsymbol{x}),\beta^{-1}) \\ p(y|f,\boldsymbol{x}) &= [\sigma(f(\boldsymbol{x}))]^y[1-\sigma(f(\boldsymbol{x}))]^{1-y} \\ p(y|f,\boldsymbol{x}) &= \text{ExpFam}(f(\boldsymbol{x})) \end{aligned}$$

- How can we define such a function nonlinear f?
- Note: We not only want nonlinearity but also all benefits of probabilistic/Bayesian modeling
  - Must be able to get uncertainty estimates in the function and its predictions
- Usually done in one of the following ways
  - Ad-hoc: Define nonlinear features $\phi(\boldsymbol{x})$ + train Bayesian linear model ($f(\boldsymbol{x}) = \boldsymbol{w}^\top\phi(\boldsymbol{x})$): HW1)
  - Ad-hoc: Train a neural net to extract features $\phi(\boldsymbol{x})$ + train Bayesian linear model

Modeling Nonlinear Functions

- Assume the input to output relationship to be modeled by a nonlinear function f

$$\begin{aligned} p(y|f,\boldsymbol{x}) &= \mathcal{N}(y|f(\boldsymbol{x}),\beta^{-1}) \\ p(y|f,\boldsymbol{x}) &= [\sigma(f(\boldsymbol{x}))]^y[1-\sigma(f(\boldsymbol{x}))]^{1-y} \\ p(y|f,\boldsymbol{x}) &= \text{ExpFam}(f(\boldsymbol{x})) \end{aligned}$$

- How can we define such a function nonlinear f?
- Note: We not only want nonlinearity but also all benefits of probabilistic/Bayesian modeling
  - Must be able to get uncertainty estimates in the function and its predictions
- Usually done in one of the following ways
  - Ad-hoc: Define nonlinear features $\phi(\boldsymbol{x})$ + train Bayesian linear model ($f(\boldsymbol{x}) = \boldsymbol{w}^\top\phi(\boldsymbol{x})$): HW1)
  - Ad-hoc: Train a neural net to extract features $\phi(\boldsymbol{x})$ + train Bayesian linear model
  - Bayesian Neural Networks (later this semester)

Modeling Nonlinear Functions

- Assume the input to output relationship to be modeled by a nonlinear function f

$$
\begin{aligned}
p(y|f,\boldsymbol{x}) &= \mathcal{N}(y|f(\boldsymbol{x}),\beta^{-1}) \\
p(y|f,\boldsymbol{x}) &= [\sigma(f(\boldsymbol{x}))]^y[1-\sigma(f(\boldsymbol{x}))]^{1-y} \\
p(y|f,\boldsymbol{x}) &= \text{ExpFam}(f(\boldsymbol{x}))
\end{aligned}
$$

- How can we define such a function nonlinear f?
- Note: We not only want nonlinearity but also all benefits of probabilistic/Bayesian modeling
  - Must be able to get uncertainty estimates in the function and its predictions
- Usually done in one of the following ways
  - Ad-hoc: Define nonlinear features $\phi(\boldsymbol{x})$ + train Bayesian linear model ($f(\boldsymbol{x}) = \boldsymbol{w}^\top\phi(\boldsymbol{x})$): HW1)
  - Ad-hoc: Train a neural net to extract features $\phi(\boldsymbol{x})$ + train Bayesian linear model
  - Bayesian Neural Networks (later this semester)
  - Gaussian Processes (a Bayesian approach to kernel based nonlinear learning; today)

What is Gaussian Process?

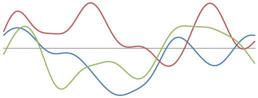
- A Gaussian Process, denoted as $\mathcal{GP}(\mu, \kappa)$, defines a distribution over functions
  - The GP is defined by mean function μ and covariance/kernel function κ

What is Gaussian Process?

- A Gaussian Process, denoted as $\mathcal{GP}(\mu, \kappa)$, defines a distribution over functions
  - The GP is defined by mean function μ and covariance/kernel function κ
- Can use GP as a prior distribution over functions

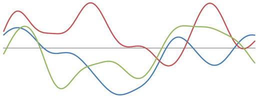
What is Gaussian Process?

- A Gaussian Process, denoted as $\mathcal{GP}(\mu, \kappa)$, defines a distribution over functions
  - The GP is defined by mean function μ and covariance/kernel function κ
- Can use GP as a prior distribution over functions
- Draw from a $\mathcal{GP}(\mu, \kappa)$ will give us a random function f (imagine it as an infinite dim. vector)



What is Gaussian Process?

- A Gaussian Process, denoted as $\mathcal{GP}(\mu, \kappa)$, defines a distribution over functions
  - The GP is defined by mean function μ and covariance/kernel function κ
- Can use GP as a prior distribution over functions
- Draw from a $\mathcal{GP}(\mu, \kappa)$ will give us a random function f (imagine it as an infinite dim. vector)

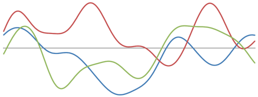


- Mean function μ models the "average" function f from $\mathcal{GP}(\mu, \kappa)$

$$\mu(\boldsymbol{x}) = \mathbb{E}[f(\boldsymbol{x})]$$

What is Gaussian Process?

- A Gaussian Process, denoted as $\mathcal{GP}(\mu, \kappa)$, defines a distribution over functions
  - The GP is defined by mean function μ and covariance/kernel function κ
- Can use GP as a prior distribution over functions
- Draw from a $\mathcal{GP}(\mu, \kappa)$ will give us a random function f (imagine it as an infinite dim. vector)



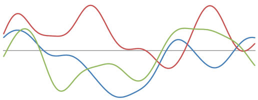
- Mean function μ models the "average" function f from $\mathcal{GP}(\mu, \kappa)$

$$\mu(\boldsymbol{x}) = \mathbb{E}[f(\boldsymbol{x})]$$

- Cov. function κ models "shape/smoothness" of these functions

What is Gaussian Process?

- A Gaussian Process, denoted as $\mathcal{GP}(\mu, \kappa)$, defines a distribution over functions
  - The GP is defined by mean function μ and covariance/kernel function κ
- Can use GP as a prior distribution over functions
- Draw from a $\mathcal{GP}(\mu, \kappa)$ will give us a random function f (imagine it as an infinite dim. vector)



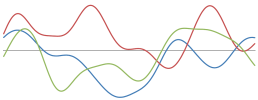
- Mean function μ models the "average" function f from $\mathcal{GP}(\mu, \kappa)$

$$\mu(\boldsymbol{x}) = \mathbb{E}[f(\boldsymbol{x})]$$

- Cov. function κ models "shape/smoothness" of these functions
  - $\kappa(.,.)$ is a function that computes similarity between two inputs (just like a kernel function)

What is Gaussian Process?

- A Gaussian Process, denoted as $\mathcal{GP}(\mu, \kappa)$, defines a distribution over functions
  - The GP is defined by mean function μ and covariance/kernel function κ
- Can use GP as a prior distribution over functions
- Draw from a $\mathcal{GP}(\mu, \kappa)$ will give us a random function f (imagine it as an infinite dim. vector)



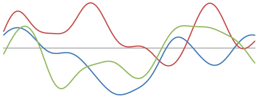
- Mean function μ models the "average" function f from $\mathcal{GP}(\mu, \kappa)$

$$\mu(\boldsymbol{x}) = \mathbb{E}[f(\boldsymbol{x})]$$

- Cov. function κ models "shape/smoothness" of these functions
  - $\kappa(.,.)$ is a function that computes similarity between two inputs (just like a kernel function)
  - Note: $\kappa(.,.)$ needs to be positive definite (just like kernel functions)

What is Gaussian Process?

- A Gaussian Process, denoted as $\mathcal{GP}(\mu, \kappa)$, defines a distribution over functions
  - The GP is defined by mean function μ and covariance/kernel function κ
- Can use GP as a prior distribution over functions
- Draw from a $\mathcal{GP}(\mu, \kappa)$ will give us a random function f (imagine it as an infinite dim. vector)



- Mean function μ models the "average" function f from $\mathcal{GP}(\mu, \kappa)$

$$\mu(\boldsymbol{x}) = \mathbb{E}[f(\boldsymbol{x})]$$

- Cov. function κ models "shape/smoothness" of these functions
  - $\kappa(.,.)$ is a function that computes similarity between two inputs (just like a kernel function)
  - Note: $\kappa(.,.)$ needs to be positive definite (just like kernel functions)
- Can even learn μ and especially κ (makes GP very flexible to model, possibly nonlinear, functions)

Gaussian Process

- f is said to be drawn from a $\mathcal{GP}(\mu, \kappa)$ if its finite dim. version is the following joint Gaussian

$$\begin{bmatrix} f(\boldsymbol{x}_1) \\ f(\boldsymbol{x}_2) \\ \vdots \\ f(\boldsymbol{x}_N) \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \mu(\boldsymbol{x}_1) \\ \mu(\boldsymbol{x}_2) \\ \vdots \\ \mu(\boldsymbol{x}_N) \end{bmatrix}, \begin{bmatrix} \kappa(\boldsymbol{x}_1, \boldsymbol{x}_1) \dots \kappa(\boldsymbol{x}_1, \boldsymbol{x}_N) \\ \kappa(\boldsymbol{x}_2, \boldsymbol{x}_1) \dots \kappa(\boldsymbol{x}_2, \boldsymbol{x}_N) \\ \vdots \quad\quad \ddots \quad\quad \vdots \\ \kappa(\boldsymbol{x}_N, \boldsymbol{x}_1) \dots \kappa(\boldsymbol{x}_N, \boldsymbol{x}_N) \end{bmatrix} \right)$$

Gaussian Process

- f is said to be drawn from a $\mathcal{GP}(\mu, \kappa)$ if its finite dim. version is the following joint Gaussian

$$\begin{bmatrix} f(\boldsymbol{x}_1) \\ f(\boldsymbol{x}_2) \\ \vdots \\ f(\boldsymbol{x}_N) \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \mu(\boldsymbol{x}_1) \\ \mu(\boldsymbol{x}_2) \\ \vdots \\ \mu(\boldsymbol{x}_N) \end{bmatrix}, \begin{bmatrix} \kappa(\boldsymbol{x}_1, \boldsymbol{x}_1) \ldots \kappa(\boldsymbol{x}_1, \boldsymbol{x}_N) \\ \kappa(\boldsymbol{x}_2, \boldsymbol{x}_1) \ldots \kappa(\boldsymbol{x}_2, \boldsymbol{x}_N) \\ \vdots \quad \ddots \quad \vdots \\ \kappa(\boldsymbol{x}_N, \boldsymbol{x}_1) \ldots \kappa(\boldsymbol{x}_N, \boldsymbol{x}_N) \end{bmatrix} \right)$$

- The above means that f's values at any finite set of inputs are jointly Gaussian

Gaussian Process

- f is said to be drawn from a $\mathcal{GP}(\mu, \kappa)$ if its finite dim. version is the following joint Gaussian

$$\begin{bmatrix} f(\boldsymbol{x}_1) \\ f(\boldsymbol{x}_2) \\ \vdots \\ f(\boldsymbol{x}_N) \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \mu(\boldsymbol{x}_1) \\ \mu(\boldsymbol{x}_2) \\ \vdots \\ \mu(\boldsymbol{x}_N) \end{bmatrix}, \begin{bmatrix} \kappa(\boldsymbol{x}_1, \boldsymbol{x}_1) \dots \kappa(\boldsymbol{x}_1, \boldsymbol{x}_N) \\ \kappa(\boldsymbol{x}_2, \boldsymbol{x}_1) \dots \kappa(\boldsymbol{x}_2, \boldsymbol{x}_N) \\ \vdots \quad \ddots \quad \vdots \\ \kappa(\boldsymbol{x}_N, \boldsymbol{x}_1) \dots \kappa(\boldsymbol{x}_N, \boldsymbol{x}_N) \end{bmatrix} \right)$$

- The above means that f's values at any finite set of inputs are jointly Gaussian
- We can also write the above more compactly as $\mathbf{f} \sim \mathcal{N}(\boldsymbol{\mu}, \mathbf{K})$ where

$$\mathbf{f} = \begin{bmatrix} f(\boldsymbol{x}_1) \\ f(\boldsymbol{x}_2) \\ \vdots \\ f(\boldsymbol{x}_N) \end{bmatrix}, \boldsymbol{\mu} = \begin{bmatrix} \mu(\boldsymbol{x}_1) \\ \mu(\boldsymbol{x}_2) \\ \vdots \\ \mu(\boldsymbol{x}_N) \end{bmatrix}, \mathbf{K} = \begin{bmatrix} \kappa(\boldsymbol{x}_1, \boldsymbol{x}_1) \dots \kappa(\boldsymbol{x}_1, \boldsymbol{x}_N) \\ \kappa(\boldsymbol{x}_2, \boldsymbol{x}_1) \dots \kappa(\boldsymbol{x}_2, \boldsymbol{x}_N) \\ \vdots \quad \ddots \quad \vdots \\ \kappa(\boldsymbol{x}_N, \boldsymbol{x}_1) \dots \kappa(\boldsymbol{x}_N, \boldsymbol{x}_N) \end{bmatrix}$$

Gaussian Process

- f is said to be drawn from a $\mathcal{GP}(\mu, \kappa)$ if its finite dim. version is the following joint Gaussian

$$\begin{bmatrix} f(\mathbf{x}_1) \\ f(\mathbf{x}_2) \\ \vdots \\ f(\mathbf{x}_N) \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \mu(\mathbf{x}_1) \\ \mu(\mathbf{x}_2) \\ \vdots \\ \mu(\mathbf{x}_N) \end{bmatrix}, \begin{bmatrix} \kappa(\mathbf{x}_1, \mathbf{x}_1) \dots \kappa(\mathbf{x}_1, \mathbf{x}_N) \\ \kappa(\mathbf{x}_2, \mathbf{x}_1) \dots \kappa(\mathbf{x}_2, \mathbf{x}_N) \\ \vdots \quad \ddots \quad \vdots \\ \kappa(\mathbf{x}_N, \mathbf{x}_1) \dots \kappa(\mathbf{x}_N, \mathbf{x}_N) \end{bmatrix} \right)$$

- The above means that f's values at any finite set of inputs are jointly Gaussian
- We can also write the above more compactly as $\mathbf{f} \sim \mathcal{N}(\boldsymbol{\mu}, \mathbf{K})$ where

$$\mathbf{f} = \begin{bmatrix} f(\mathbf{x}_1) \\ f(\mathbf{x}_2) \\ \vdots \\ f(\mathbf{x}_N) \end{bmatrix}, \boldsymbol{\mu} = \begin{bmatrix} \mu(\mathbf{x}_1) \\ \mu(\mathbf{x}_2) \\ \vdots \\ \mu(\mathbf{x}_N) \end{bmatrix}, \mathbf{K} = \begin{bmatrix} \kappa(\mathbf{x}_1, \mathbf{x}_1) \dots \kappa(\mathbf{x}_1, \mathbf{x}_N) \\ \kappa(\mathbf{x}_2, \mathbf{x}_1) \dots \kappa(\mathbf{x}_2, \mathbf{x}_N) \\ \vdots \quad \ddots \quad \vdots \\ \kappa(\mathbf{x}_N, \mathbf{x}_1) \dots \kappa(\mathbf{x}_N, \mathbf{x}_N) \end{bmatrix}$$

- Note that $p(\mathbf{f}) = \mathcal{N}(\boldsymbol{\mu}, \mathbf{K})$ can be seen as the finite-dimensional version of the GP prior over f

Gaussian Process

- f is said to be drawn from a $\mathcal{GP}(\mu, \kappa)$ if its finite dim. version is the following joint Gaussian

$$\begin{bmatrix} f(\boldsymbol{x}_1) \\ f(\boldsymbol{x}_2) \\ \vdots \\ f(\boldsymbol{x}_N) \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \mu(\boldsymbol{x}_1) \\ \mu(\boldsymbol{x}_2) \\ \vdots \\ \mu(\boldsymbol{x}_N) \end{bmatrix}, \begin{bmatrix} \kappa(\boldsymbol{x}_1, \boldsymbol{x}_1) \dots \kappa(\boldsymbol{x}_1, \boldsymbol{x}_N) \\ \kappa(\boldsymbol{x}_2, \boldsymbol{x}_1) \dots \kappa(\boldsymbol{x}_2, \boldsymbol{x}_N) \\ \vdots \quad \ddots \quad \vdots \\ \kappa(\boldsymbol{x}_N, \boldsymbol{x}_1) \dots \kappa(\boldsymbol{x}_N, \boldsymbol{x}_N) \end{bmatrix} \right)$$

- The above means that f's values at any finite set of inputs are jointly Gaussian
- We can also write the above more compactly as $\mathbf{f} \sim \mathcal{N}(\boldsymbol{\mu}, \mathbf{K})$ where

$$\mathbf{f} = \begin{bmatrix} f(\boldsymbol{x}_1) \\ f(\boldsymbol{x}_2) \\ \vdots \\ f(\boldsymbol{x}_N) \end{bmatrix}, \boldsymbol{\mu} = \begin{bmatrix} \mu(\boldsymbol{x}_1) \\ \mu(\boldsymbol{x}_2) \\ \vdots \\ \mu(\boldsymbol{x}_N) \end{bmatrix}, \mathbf{K} = \begin{bmatrix} \kappa(\boldsymbol{x}_1, \boldsymbol{x}_1) \dots \kappa(\boldsymbol{x}_1, \boldsymbol{x}_N) \\ \kappa(\boldsymbol{x}_2, \boldsymbol{x}_1) \dots \kappa(\boldsymbol{x}_2, \boldsymbol{x}_N) \\ \vdots \quad \ddots \quad \vdots \\ \kappa(\boldsymbol{x}_N, \boldsymbol{x}_1) \dots \kappa(\boldsymbol{x}_N, \boldsymbol{x}_N) \end{bmatrix}$$

- Note that $p(\mathbf{f}) = \mathcal{N}(\boldsymbol{\mu}, \mathbf{K})$ can be seen as the finite-dimensional version of the GP prior over f
- If mean function is zero, we will have $p(\mathbf{f}) = \mathcal{N}(\mathbf{0}, \mathbf{K})$

Gaussian Process

- f is said to be drawn from a $\mathcal{GP}(\mu, \kappa)$ if its finite dim. version is the following joint Gaussian

$$\begin{bmatrix} f(\mathbf{x}_1) \\ f(\mathbf{x}_2) \\ \vdots \\ f(\mathbf{x}_N) \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \mu(\mathbf{x}_1) \\ \mu(\mathbf{x}_2) \\ \vdots \\ \mu(\mathbf{x}_N) \end{bmatrix}, \begin{bmatrix} \kappa(\mathbf{x}_1, \mathbf{x}_1) \dots \kappa(\mathbf{x}_1, \mathbf{x}_N) \\ \kappa(\mathbf{x}_2, \mathbf{x}_1) \dots \kappa(\mathbf{x}_2, \mathbf{x}_N) \\ \vdots \quad \ddots \quad \vdots \\ \kappa(\mathbf{x}_N, \mathbf{x}_1) \dots \kappa(\mathbf{x}_N, \mathbf{x}_N) \end{bmatrix} \right)$$

- The above means that f's values at any finite set of inputs are jointly Gaussian
- We can also write the above more compactly as $\mathbf{f} \sim \mathcal{N}(\boldsymbol{\mu}, \mathbf{K})$ where

$$\mathbf{f} = \begin{bmatrix} f(\mathbf{x}_1) \\ f(\mathbf{x}_2) \\ \vdots \\ f(\mathbf{x}_N) \end{bmatrix}, \boldsymbol{\mu} = \begin{bmatrix} \mu(\mathbf{x}_1) \\ \mu(\mathbf{x}_2) \\ \vdots \\ \mu(\mathbf{x}_N) \end{bmatrix}, \mathbf{K} = \begin{bmatrix} \kappa(\mathbf{x}_1, \mathbf{x}_1) \dots \kappa(\mathbf{x}_1, \mathbf{x}_N) \\ \kappa(\mathbf{x}_2, \mathbf{x}_1) \dots \kappa(\mathbf{x}_2, \mathbf{x}_N) \\ \vdots \quad \ddots \quad \vdots \\ \kappa(\mathbf{x}_N, \mathbf{x}_1) \dots \kappa(\mathbf{x}_N, \mathbf{x}_N) \end{bmatrix}$$

- Note that $p(\mathbf{f}) = \mathcal{N}(\boldsymbol{\mu}, \mathbf{K})$ can be seen as the finite-dimensional version of the GP prior over f
- If mean function is zero, we will have $p(\mathbf{f}) = \mathcal{N}(\mathbf{0}, \mathbf{K})$. **Important:** $p(\boldsymbol{f}_i | \boldsymbol{f}_{-i})$ is also Gaussian (where i denotes any subset of inputs and $-i$ denotes rest of the inputs) due to Gaussian properties

A Perspective from Bayesian Linear Regression

- Let's first consider the (probabilistic) linear regression model

$$
\begin{aligned}
p(\boldsymbol{w}) &= \mathcal{N}(\boldsymbol{w}|\boldsymbol{\mu_0}, \Sigma_0) \qquad &\text{(Prior)} \\
p(\boldsymbol{y}|\mathbf{X}, \boldsymbol{w}) &= \mathcal{N}(\mathbf{X}\boldsymbol{w}, \beta^{-1}\mathbf{I}_N) \qquad &\text{(Likelihood w.r.t. } N \text{ obs.)}
\end{aligned}
$$

A Perspective from Bayesian Linear Regression

- Let's first consider the (probabilistic) linear regression model

$$\begin{aligned} p(\boldsymbol{w}) &= \mathcal{N}(\boldsymbol{w}|\boldsymbol{\mu_0}, \Sigma_0) & \text{(Prior)} \\ p(\boldsymbol{y}|\mathbf{X}, \boldsymbol{w}) &= \mathcal{N}(\mathbf{X}\boldsymbol{w}, \beta^{-1}\mathbf{I}_N) & \text{(Likelihood w.r.t. } N \text{ obs.)} \\ p(\boldsymbol{y}|\mathbf{X}) &= \int p(\boldsymbol{y}|\mathbf{X}, \boldsymbol{w})p(\boldsymbol{w})d\boldsymbol{w} = \mathcal{N}(\mathbf{X}\boldsymbol{\mu_0}, \beta^{-1}\mathbf{I}_N + \mathbf{X}\boldsymbol{\Sigma_0}\mathbf{X}^\top) & \text{(Marginal likelihood)} \end{aligned}$$

A Perspective from Bayesian Linear Regression

- Let's first consider the (probabilistic) linear regression model

$$
\begin{aligned}
p(\boldsymbol{w}) &= \mathcal{N}(\boldsymbol{w}|\boldsymbol{\mu_0}, \Sigma_0) && \text{(Prior)} \\
p(\boldsymbol{y}|\mathbf{X}, \boldsymbol{w}) &= \mathcal{N}(\mathbf{X}\boldsymbol{w}, \beta^{-1}\mathbf{I}_N) && \text{(Likelihood w.r.t. } N \text{ obs.)} \\
p(\boldsymbol{y}|\mathbf{X}) &= \int p(\boldsymbol{y}|\mathbf{X}, \boldsymbol{w})p(\boldsymbol{w})d\boldsymbol{w} = \mathcal{N}(\mathbf{X}\boldsymbol{\mu_0}, \beta^{-1}\mathbf{I}_N + \mathbf{X}\boldsymbol{\Sigma_0}\mathbf{X}^\top) && \text{(Marginal likelihood)} \\
p(\boldsymbol{y}|\mathbf{X}) &= \mathcal{N}(\mathbf{0}, \beta^{-1}\mathbf{I}_N + \mathbf{X}\mathbf{X}^\top) && \text{(if } \boldsymbol{\mu}_0 = 0 \text{ and } \boldsymbol{\Sigma}_0 = \mathbf{I})
\end{aligned}
$$

A Perspective from Bayesian Linear Regression

- Let's first consider the (probabilistic) linear regression model

$$\begin{aligned}
p(\boldsymbol{w}) &= \mathcal{N}(\boldsymbol{w}|\boldsymbol{\mu_0}, \Sigma_0) && \text{(Prior)} \\
p(\boldsymbol{y}|\mathbf{X}, \boldsymbol{w}) &= \mathcal{N}(\mathbf{X}\boldsymbol{w}, \beta^{-1}\mathbf{I}_N) && \text{(Likelihood w.r.t. } N \text{ obs.)} \\
p(\boldsymbol{y}|\mathbf{X}) &= \int p(\boldsymbol{y}|\mathbf{X}, \boldsymbol{w})p(\boldsymbol{w})d\boldsymbol{w} = \mathcal{N}(\mathbf{X}\boldsymbol{\mu_0}, \beta^{-1}\mathbf{I}_N + \mathbf{X}\boldsymbol{\Sigma_0}\mathbf{X}^\top) && \text{(Marginal likelihood)} \\
p(\boldsymbol{y}|\mathbf{X}) &= \mathcal{N}(\mathbf{0}, \beta^{-1}\mathbf{I}_N + \mathbf{X}\mathbf{X}^\top) && \text{(if } \boldsymbol{\mu}_0 = 0 \text{ and } \boldsymbol{\Sigma}_0 = \mathbf{I}) \\
p(\boldsymbol{y}|\mathbf{X}) &= \mathcal{N}(\mathbf{0}, \mathbf{X}\mathbf{X}^\top) && \text{(if } \beta^{-1} = \infty\text{, i.e., zero noise)}
\end{aligned}$$

A Perspective from Bayesian Linear Regression

- Let's first consider the (probabilistic) linear regression model

$$
\begin{aligned}
p(\boldsymbol{w}) &= \mathcal{N}(\boldsymbol{w}|\boldsymbol{\mu_0}, \Sigma_0) && \text{(Prior)} \\
p(\boldsymbol{y}|\mathbf{X}, \boldsymbol{w}) &= \mathcal{N}(\mathbf{X}\boldsymbol{w}, \beta^{-1}\mathbf{I}_N) && \text{(Likelihood w.r.t. } N \text{ obs.)} \\
p(\boldsymbol{y}|\mathbf{X}) &= \int p(\boldsymbol{y}|\mathbf{X}, \boldsymbol{w})p(\boldsymbol{w})d\boldsymbol{w} = \mathcal{N}(\mathbf{X}\boldsymbol{\mu_0}, \beta^{-1}\mathbf{I}_N + \mathbf{X}\boldsymbol{\Sigma_0}\mathbf{X}^\top) && \text{(Marginal likelihood)} \\
p(\boldsymbol{y}|\mathbf{X}) &= \mathcal{N}(\mathbf{0}, \beta^{-1}\mathbf{I}_N + \mathbf{X}\mathbf{X}^\top) && (\text{if } \boldsymbol{\mu}_0 = 0 \text{ and } \boldsymbol{\Sigma}_0 = \mathbf{I}) \\
p(\boldsymbol{y}|\mathbf{X}) &= \mathcal{N}(\mathbf{0}, \mathbf{X}\mathbf{X}^\top) && (\text{if } \beta^{-1} = \infty, \text{ i.e., zero noise})
\end{aligned}
$$

- Thus the joint marginal distr. of $\boldsymbol{y}$ conditioned on $\mathbf{X}$ is the following multivariate Gaussian

$$
\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \begin{bmatrix} \boldsymbol{x}_1^\top \boldsymbol{x}_1 \dots \boldsymbol{x}_1^\top \boldsymbol{x}_N \\ \boldsymbol{x}_2^\top \boldsymbol{x}_1 \dots \boldsymbol{x}_2^\top \boldsymbol{x}_N \\ \vdots \quad \ddots \quad \vdots \\ \boldsymbol{x}_N^\top \boldsymbol{x}_1 \dots \boldsymbol{x}_N^\top \boldsymbol{x}_N \end{bmatrix} \right)
$$

A Perspective from Bayesian Linear Regression

- Let's first consider the (probabilistic) linear regression model

$$
\begin{aligned}
p(\boldsymbol{w}) &= \mathcal{N}(\boldsymbol{w}|\boldsymbol{\mu_0}, \Sigma_0) && \text{(Prior)} \\
p(\boldsymbol{y}|\mathbf{X}, \boldsymbol{w}) &= \mathcal{N}(\mathbf{X}\boldsymbol{w}, \beta^{-1}\mathbf{I}_N) && \text{(Likelihood w.r.t. } N \text{ obs.)} \\
p(\boldsymbol{y}|\mathbf{X}) &= \int p(\boldsymbol{y}|\mathbf{X}, \boldsymbol{w})p(\boldsymbol{w})d\boldsymbol{w} = \mathcal{N}(\mathbf{X}\boldsymbol{\mu_0}, \beta^{-1}\mathbf{I}_N + \mathbf{X\Sigma_0X}^\top) && \text{(Marginal likelihood)} \\
p(\boldsymbol{y}|\mathbf{X}) &= \mathcal{N}(\mathbf{0}, \beta^{-1}\mathbf{I}_N + \mathbf{XX}^\top) && (\text{if } \boldsymbol{\mu}_0 = 0 \text{ and } \boldsymbol{\Sigma}_0 = \mathbf{I}) \\
p(\boldsymbol{y}|\mathbf{X}) &= \mathcal{N}(\mathbf{0}, \mathbf{XX}^\top) && (\text{if } \beta^{-1} = \infty, \text{ i.e., zero noise})
\end{aligned}
$$

- Thus the joint marginal distr. of $\boldsymbol{y}$ conditioned on $\mathbf{X}$ is the following multivariate Gaussian

$$
\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \begin{bmatrix} \boldsymbol{x}_1^\top \boldsymbol{x}_1 \dots \boldsymbol{x}_1^\top \boldsymbol{x}_N \\ \boldsymbol{x}_2^\top \boldsymbol{x}_1 \dots \boldsymbol{x}_2^\top \boldsymbol{x}_N \\ \vdots \quad \ddots \quad \vdots \\ \boldsymbol{x}_N^\top \boldsymbol{x}_1 \dots \boldsymbol{x}_N^\top \boldsymbol{x}_N \end{bmatrix} \right)
$$

- A "function space" view of linear regression as opposed to "weight space" view (both equivalent)

GP for Regression and Classification

(Note that GP only defines the score $f(\boldsymbol{x})$ but $y = f(\boldsymbol{x}) +$ "noise")

("noise" may be Gaussian, sigmoid-Bernoulli, or something else)

GP Regression

- Training data: $\{\boldsymbol{x}_n, y_n\}_{n=1}^N$. $\boldsymbol{x}_n \in \mathbb{R}^D$, $y_n \in \mathbb{R}$

GP Regression

- Training data: $\{\boldsymbol{x}_n, y_n\}_{n=1}^N$. $\boldsymbol{x}_n \in \mathbb{R}^D$, $y_n \in \mathbb{R}$
- Assume the responses to be a noisy function of the inputs

$$y_n = f(\boldsymbol{x}_n) + \epsilon_n = f_n + \epsilon_n$$

GP Regression

- Training data: $\{\boldsymbol{x}_n, y_n\}_{n=1}^N$. $\boldsymbol{x}_n \in \mathbb{R}^D$, $y_n \in \mathbb{R}$
- Assume the responses to be a noisy function of the inputs

$$y_n = f(\boldsymbol{x}_n) + \epsilon_n = f_n + \epsilon_n$$

- Assume a zero-mean Gaussian noise: $\epsilon_n \sim \mathcal{N}(\epsilon_n|0, \sigma^2)$

GP Regression

- Training data: $\{\boldsymbol{x}_n, y_n\}_{n=1}^N$. $\boldsymbol{x}_n \in \mathbb{R}^D$, $y_n \in \mathbb{R}$
- Assume the responses to be a noisy function of the inputs

$$y_n = f(\boldsymbol{x}_n) + \epsilon_n = f_n + \epsilon_n$$

- Assume a zero-mean Gaussian noise: $\epsilon_n \sim \mathcal{N}(\epsilon_n|0, \sigma^2)$
- This implies the following likelihood model: $p(y_n|f_n) = \mathcal{N}(y_n|f_n, \sigma^2)$

GP Regression

- Training data: $\{\boldsymbol{x}_n, y_n\}_{n=1}^N$. $\boldsymbol{x}_n \in \mathbb{R}^D$, $y_n \in \mathbb{R}$
- Assume the responses to be a noisy function of the inputs

$$y_n = f(\boldsymbol{x}_n) + \epsilon_n = f_n + \epsilon_n$$

- Assume a zero-mean Gaussian noise: $\epsilon_n \sim \mathcal{N}(\epsilon_n|0, \sigma^2)$
- This implies the following likelihood model: $p(y_n|f_n) = \mathcal{N}(y_n|f_n, \sigma^2)$
- Denote $\boldsymbol{f} = [f_1, \ldots, f_N]$ and $\boldsymbol{y} = [y_1, \ldots, y_N]$.

GP Regression

- Training data: $\{\boldsymbol{x}_n, y_n\}_{n=1}^N$. $\boldsymbol{x}_n \in \mathbb{R}^D$, $y_n \in \mathbb{R}$
- Assume the responses to be a noisy function of the inputs

$$y_n = f(\boldsymbol{x}_n) + \epsilon_n = f_n + \epsilon_n$$

- Assume a zero-mean Gaussian noise: $\epsilon_n \sim \mathcal{N}(\epsilon_n|0, \sigma^2)$
- This implies the following likelihood model: $p(y_n|f_n) = \mathcal{N}(y_n|f_n, \sigma^2)$
- Denote $\boldsymbol{f} = [f_1, \ldots, f_N]$ and $\boldsymbol{y} = [y_1, \ldots, y_N]$. For i.i.d. responses, the joint likelihood will be

$$p(\boldsymbol{y}|\mathbf{f}) = \mathcal{N}(\boldsymbol{y}|\mathbf{f}, \sigma^2\mathbf{I}_N)$$

GP Regression

- Training data: $\{\boldsymbol{x}_n, y_n\}_{n=1}^N$. $\boldsymbol{x}_n \in \mathbb{R}^D$, $y_n \in \mathbb{R}$
- Assume the responses to be a noisy function of the inputs
$$y_n = f(\boldsymbol{x}_n) + \epsilon_n = f_n + \epsilon_n$$
- Assume a zero-mean Gaussian noise: $\epsilon_n \sim \mathcal{N}(\epsilon_n|0, \sigma^2)$
- This implies the following likelihood model: $p(y_n|f_n) = \mathcal{N}(y_n|f_n, \sigma^2)$
- Denote $\boldsymbol{f} = [f_1, \ldots, f_N]$ and $\boldsymbol{y} = [y_1, \ldots, y_N]$. For i.i.d. responses, the joint likelihood will be
$$p(\boldsymbol{y}|\mathbf{f}) = \mathcal{N}(\boldsymbol{y}|\mathbf{f}, \sigma^2 \mathbf{I}_N)$$
- We now need a prior on the function f that enables us to model a nonlinear f

GP Regression

- Training data: $\{\boldsymbol{x}_n, y_n\}_{n=1}^N$. $\boldsymbol{x}_n \in \mathbb{R}^D$, $y_n \in \mathbb{R}$
- Assume the responses to be a noisy function of the inputs

$$y_n = f(\boldsymbol{x}_n) + \epsilon_n = f_n + \epsilon_n$$

- Assume a zero-mean Gaussian noise: $\epsilon_n \sim \mathcal{N}(\epsilon_n|0, \sigma^2)$
- This implies the following likelihood model: $p(y_n|f_n) = \mathcal{N}(y_n|f_n, \sigma^2)$
- Denote $\boldsymbol{f} = [f_1, \ldots, f_N]$ and $\boldsymbol{y} = [y_1, \ldots, y_N]$. For i.i.d. responses, the joint likelihood will be

$$p(\boldsymbol{y}|\mathbf{f}) = \mathcal{N}(\boldsymbol{y}|\mathbf{f}, \sigma^2\mathbf{I}_N)$$

- We now need a prior on the function f that enables us to model a nonlinear f
- Let's choose zero mean Gaussian Process prior $\mathcal{GP}(0, \kappa)$ on f, which is equivalent to

$$p(\mathbf{f}) = \mathcal{N}(\mathbf{f}|\mathbf{0}, \mathbf{K})$$

where $K_{nm} = \kappa(\boldsymbol{x}_n, \boldsymbol{x}_m)$.

GP Regression

- Training data: $\{\boldsymbol{x}_n, y_n\}_{n=1}^N$. $\boldsymbol{x}_n \in \mathbb{R}^D$, $y_n \in \mathbb{R}$
- Assume the responses to be a noisy function of the inputs

$$y_n = f(\boldsymbol{x}_n) + \epsilon_n = f_n + \epsilon_n$$

- Assume a zero-mean Gaussian noise: $\epsilon_n \sim \mathcal{N}(\epsilon_n|0, \sigma^2)$
- This implies the following likelihood model: $p(y_n|f_n) = \mathcal{N}(y_n|f_n, \sigma^2)$
- Denote $\boldsymbol{f} = [f_1, \ldots, f_N]$ and $\boldsymbol{y} = [y_1, \ldots, y_N]$. For i.i.d. responses, the joint likelihood will be

$$p(\boldsymbol{y}|\mathbf{f}) = \mathcal{N}(\boldsymbol{y}|\mathbf{f}, \sigma^2\mathbf{I}_N)$$

- We now need a prior on the function f that enables us to model a nonlinear f
- Let's choose zero mean Gaussian Process prior $\mathcal{GP}(0, \kappa)$ on f, which is equivalent to

$$p(\mathbf{f}) = \mathcal{N}(\mathbf{f}|\mathbf{0}, \mathbf{K})$$

where $K_{nm} = \kappa(\boldsymbol{x}_n, \boldsymbol{x}_m)$. For now, assume κ is a known function with fixed hyperparameters.

GP Regression

- The likelihood model: $p(\boldsymbol{y}|\mathbf{f}) = \mathcal{N}(\boldsymbol{y}|\mathbf{f}, \sigma^2\mathbf{I}_N)$. The prior distribution: $p(\mathbf{f}) = \mathcal{N}(\mathbf{f}|\mathbf{0}, \mathbf{K})$

GP Regression

- The likelihood model: $p(\boldsymbol{y}|\mathbf{f}) = \mathcal{N}(\boldsymbol{y}|\mathbf{f}, \sigma^2\mathbf{I}_N)$. The prior distribution: $p(\mathbf{f}) = \mathcal{N}(\mathbf{f}|\mathbf{0}, \mathbf{K})$
- The posterior $p(\mathbf{f}|\boldsymbol{y}) \propto p(\mathbf{f})p(\boldsymbol{y}|\mathbf{f})$, which will be another Gaussian (Exercise: Find its expression)

GP Regression

- The likelihood model: $p(\boldsymbol{y}|\mathbf{f}) = \mathcal{N}(\boldsymbol{y}|\mathbf{f}, \sigma^2\mathbf{I}_N)$. The prior distribution: $p(\mathbf{f}) = \mathcal{N}(\mathbf{f}|\mathbf{0}, \mathbf{K})$
- The posterior $p(\mathbf{f}|\boldsymbol{y}) \propto p(\mathbf{f})p(\boldsymbol{y}|\mathbf{f})$, which will be another Gaussian (Exercise: Find its expression)
- What's the posterior predictive $p(y_*|\boldsymbol{x}_*, \boldsymbol{y}, \mathbf{X})$ or $p(y_*|\boldsymbol{y})$ (skipping $\mathbf{X}, \boldsymbol{x}_*$ from the notation)?

GP Regression

- The likelihood model: $p(\boldsymbol{y}|\mathbf{f}) = \mathcal{N}(\boldsymbol{y}|\mathbf{f}, \sigma^2\mathbf{I}_N)$. The prior distribution: $p(\mathbf{f}) = \mathcal{N}(\mathbf{f}|\mathbf{0}, \mathbf{K})$
- The posterior $p(\mathbf{f}|\boldsymbol{y}) \propto p(\mathbf{f})p(\boldsymbol{y}|\mathbf{f})$, which will be another Gaussian (Exercise: Find its expression)
- What's the posterior predictive $p(y_*|\boldsymbol{x}_*, \boldsymbol{y}, \mathbf{X})$ or $p(y_*|\boldsymbol{y})$ (skipping $\mathbf{X}, \boldsymbol{x}_*$ from the notation)?

$$p(y_*|\boldsymbol{y}) = \int p(y_*|f_*)p(f_*|\boldsymbol{y})df_*$$

GP Regression

- The likelihood model: $p(\boldsymbol{y}|\mathbf{f}) = \mathcal{N}(\boldsymbol{y}|\mathbf{f}, \sigma^2\mathbf{I}_N)$. The prior distribution: $p(\mathbf{f}) = \mathcal{N}(\mathbf{f}|\mathbf{0}, \mathbf{K})$
- The posterior $p(\mathbf{f}|\boldsymbol{y}) \propto p(\mathbf{f})p(\boldsymbol{y}|\mathbf{f})$, which will be another Gaussian (Exercise: Find its expression)
- What's the posterior predictive $p(y_*|\boldsymbol{x}_*, \boldsymbol{y}, \mathbf{X})$ or $p(y_*|\boldsymbol{y})$ (skipping $\mathbf{X}, \boldsymbol{x}_*$ from the notation)?

$$p(y_*|\boldsymbol{y}) = \int p(y_*|f_*)p(f_*|\boldsymbol{y})df_*$$

where $p(f_*|\boldsymbol{y}) = \int p(f_*|\boldsymbol{f})p(\boldsymbol{f}|\boldsymbol{y})d\boldsymbol{f}$ and note that $p(f_*|\boldsymbol{f})$ must be Gaussian for GP

GP Regression

- The likelihood model: $p(\boldsymbol{y}|\mathbf{f}) = \mathcal{N}(\boldsymbol{y}|\mathbf{f}, \sigma^2\mathbf{I}_N)$. The prior distribution: $p(\mathbf{f}) = \mathcal{N}(\mathbf{f}|\mathbf{0}, \mathbf{K})$
- The posterior $p(\mathbf{f}|\boldsymbol{y}) \propto p(\mathbf{f})p(\boldsymbol{y}|\mathbf{f})$, which will be another Gaussian (Exercise: Find its expression)
- What's the posterior predictive $p(y_*|\boldsymbol{x}_*, \boldsymbol{y}, \mathbf{X})$ or $p(y_*|\boldsymbol{y})$ (skipping $\mathbf{X}, \boldsymbol{x}_*$ from the notation)?

$$p(y_*|\boldsymbol{y}) = \int p(y_*|f_*)p(f_*|\boldsymbol{y})df_*$$

  where $p(f_*|\boldsymbol{y}) = \int p(f_*|\boldsymbol{f})p(\boldsymbol{f}|\boldsymbol{y})d\boldsymbol{f}$ and note that $p(f_*|\boldsymbol{f})$ must be Gaussian for GP
- For this case (GP regression), we actually don't need to compute $p(y_*|\boldsymbol{y})$ using the above method

GP Regression

- The likelihood model: $p(\mathbf{y}|\mathbf{f}) = \mathcal{N}(\mathbf{y}|\mathbf{f}, \sigma^2\mathbf{I}_N)$. The prior distribution: $p(\mathbf{f}) = \mathcal{N}(\mathbf{f}|\mathbf{0}, \mathbf{K})$
- The posterior $p(\mathbf{f}|\boldsymbol{y}) \propto p(\mathbf{f})p(\boldsymbol{y}|\mathbf{f})$, which will be another Gaussian (Exercise: Find its expression)
- What's the posterior predictive $p(y_*|\boldsymbol{x}_*, \boldsymbol{y}, \mathbf{X})$ or $p(y_*|\boldsymbol{y})$ (skipping $\mathbf{X}, \boldsymbol{x}_*$ from the notation)?

$$p(y_*|\boldsymbol{y}) = \int p(y_*|f_*)p(f_*|\boldsymbol{y})df_*$$

  where $p(f_*|\boldsymbol{y}) = \int p(f_*|\boldsymbol{f})p(\boldsymbol{f}|\boldsymbol{y})d\boldsymbol{f}$ and note that $p(f_*|\boldsymbol{f})$ must be Gaussian for GP
- For this case (GP regression), we actually don't need to compute $p(y_*|\boldsymbol{y})$ using the above method
- Reason: The marginal distribution of the training data responses $\boldsymbol{y}$

$$p(\boldsymbol{y}) = \int p(\boldsymbol{y}|\mathbf{f})p(\mathbf{f})d\mathbf{f} = \mathcal{N}(\boldsymbol{y}|\mathbf{0}, \mathbf{K} + \sigma^2\mathbf{I}_N) = \mathcal{N}(\boldsymbol{y}|\mathbf{0}, \mathbf{C}_N)$$

GP Regression

- The likelihood model: $p(\mathbf{y}|\mathbf{f}) = \mathcal{N}(\mathbf{y}|\mathbf{f}, \sigma^2\mathbf{I}_N)$. The prior distribution: $p(\mathbf{f}) = \mathcal{N}(\mathbf{f}|\mathbf{0}, \mathbf{K})$
- The posterior $p(\mathbf{f}|\boldsymbol{y}) \propto p(\mathbf{f})p(\boldsymbol{y}|\mathbf{f})$, which will be another Gaussian (Exercise: Find its expression)
- What's the posterior predictive $p(y_*|\boldsymbol{x}_*, \boldsymbol{y}, \mathbf{X})$ or $p(y_*|\boldsymbol{y})$ (skipping $\mathbf{X}, \boldsymbol{x}_*$ from the notation)?

$$p(y_*|\boldsymbol{y}) = \int p(y_*|f_*)p(f_*|\boldsymbol{y})df_*$$

where $p(f_*|\boldsymbol{y}) = \int p(f_*|\boldsymbol{f})p(\boldsymbol{f}|\boldsymbol{y})d\boldsymbol{f}$ and note that $p(f_*|\boldsymbol{f})$ must be Gaussian for GP

- For this case (GP regression), we actually don't need to compute $p(y_*|\boldsymbol{y})$ using the above method
- Reason: The marginal distribution of the training data responses $\boldsymbol{y}$

$$p(\boldsymbol{y}) = \int p(\boldsymbol{y}|\mathbf{f})p(\mathbf{f})d\mathbf{f} = \mathcal{N}(\boldsymbol{y}|\mathbf{0}, \mathbf{K} + \sigma^2\mathbf{I}_N) = \mathcal{N}(\boldsymbol{y}|\mathbf{0}, \mathbf{C}_N)$$

- Using the same result, the marginal distribution $p(y_*) = \mathcal{N}(y_*|0, \kappa(\boldsymbol{x}_*, \boldsymbol{x}_*) + \sigma^2)$

GP Regression: Making Predictions

- Let's consider the joint distr. of N training responses $\boldsymbol{y}$ and test response y_*

$$p\left(\begin{bmatrix} \boldsymbol{y} \\ y_* \end{bmatrix}\right) = \mathcal{N}\left(\begin{bmatrix} \boldsymbol{y} \\ y_* \end{bmatrix} \middle| \begin{bmatrix} \mathbf{0} \\ 0 \end{bmatrix}, \mathbf{C}_{N+1}\right)$$

where the $(N+1) \times (N+1)$ matrix $\mathbf{C}_{N+1}$ is given by

$$\mathbf{C}_{N+1} = \begin{bmatrix} \mathbf{C}_N & \mathbf{k}_* \\ \mathbf{k}_*^\top & c \end{bmatrix}$$

GP Regression: Making Predictions

- Let's consider the joint distr. of N training responses $\boldsymbol{y}$ and test response y_*

$$p\left(\begin{bmatrix} \boldsymbol{y} \\ y_* \end{bmatrix}\right) = \mathcal{N}\left(\begin{bmatrix} \boldsymbol{y} \\ y_* \end{bmatrix} \middle| \begin{bmatrix} \mathbf{0} \\ 0 \end{bmatrix}, \mathbf{C}_{N+1}\right)$$

where the $(N+1) \times (N+1)$ matrix $\mathbf{C}_{N+1}$ is given by

$$\mathbf{C}_{N+1} = \begin{bmatrix} \mathbf{C}_N & \mathbf{k}_* \\ \mathbf{k}_*{}^\top & c \end{bmatrix}$$

and $\mathbf{k}_* = [\kappa(\boldsymbol{x}_*, \boldsymbol{x}_1), \ldots, \kappa(\boldsymbol{x}_*, \boldsymbol{x}_N)]^\top$, $c = \kappa(\boldsymbol{x}_*, \boldsymbol{x}_*) + \sigma^2$

GP Regression: Making Predictions

- Let's consider the joint distr. of N training responses $\boldsymbol{y}$ and test response y_*

$$p\left(\begin{bmatrix} \boldsymbol{y} \\ y_* \end{bmatrix}\right) = \mathcal{N}\left(\begin{bmatrix} \boldsymbol{y} \\ y_* \end{bmatrix} \middle| \begin{bmatrix} \mathbf{0} \\ 0 \end{bmatrix}, \mathbf{C}_{N+1}\right)$$

where the $(N+1) \times (N+1)$ matrix $\mathbf{C}_{N+1}$ is given by

$$\mathbf{C}_{N+1} = \begin{bmatrix} \mathbf{C}_N & \mathbf{k}_* \\ \mathbf{k}_*{}^\top & c \end{bmatrix}$$

and $\mathbf{k}_* = [\kappa(\boldsymbol{x}_*, \boldsymbol{x}_1), \ldots, \kappa(\boldsymbol{x}_*, \boldsymbol{x}_N)]^\top$, $c = \kappa(\boldsymbol{x}_*, \boldsymbol{x}_*) + \sigma^2$

- The desired predictive posterior will be (using conditional from joint property of Gaussian)

$$p(y_*|\boldsymbol{y}) \quad = \quad \mathcal{N}(y_*|\mu_*, \sigma_*^2)$$

GP Regression: Making Predictions

- Let's consider the joint distr. of N training responses $\boldsymbol{y}$ and test response y_*

$$p\left(\begin{bmatrix} \boldsymbol{y} \\ y_* \end{bmatrix}\right) = \mathcal{N}\left(\begin{bmatrix} \boldsymbol{y} \\ y_* \end{bmatrix} \middle| \begin{bmatrix} \mathbf{0} \\ 0 \end{bmatrix}, \mathbf{C}_{N+1}\right)$$

where the $(N+1) \times (N+1)$ matrix $\mathbf{C}_{N+1}$ is given by

$$\mathbf{C}_{N+1} = \begin{bmatrix} \mathbf{C}_N & \mathbf{k}_* \\ \mathbf{k}_*{}^\top & c \end{bmatrix}$$

and $\mathbf{k}_* = [\kappa(\boldsymbol{x}_*, \boldsymbol{x}_1), \ldots, \kappa(\boldsymbol{x}_*, \boldsymbol{x}_N)]^\top$, $c = \kappa(\boldsymbol{x}_*, \boldsymbol{x}_*) + \sigma^2$

- The desired predictive posterior will be (using conditional from joint property of Gaussian)

$$\begin{aligned} p(y_*|\boldsymbol{y}) &= \mathcal{N}(y_*|\mu_*, \sigma_*^2) \\ \mu_* &= \mathbf{k}_*{}^\top \mathbf{C}_N^{-1} \boldsymbol{y} \end{aligned}$$

GP Regression: Making Predictions

- Let's consider the joint distr. of N training responses $\boldsymbol{y}$ and test response y_*

$$p\left(\begin{bmatrix} \boldsymbol{y} \\ y_* \end{bmatrix}\right) = \mathcal{N}\left(\begin{bmatrix} \boldsymbol{y} \\ y_* \end{bmatrix} \middle| \begin{bmatrix} \mathbf{0} \\ 0 \end{bmatrix}, \mathbf{C}_{N+1}\right)$$

  where the $(N+1) \times (N+1)$ matrix $\mathbf{C}_{N+1}$ is given by

$$\mathbf{C}_{N+1} = \begin{bmatrix} \mathbf{C}_N & \mathbf{k}_* \\ \mathbf{k}_*^\top & c \end{bmatrix}$$

  and $\mathbf{k}_* = [\kappa(\boldsymbol{x}_*, \boldsymbol{x}_1), \ldots, \kappa(\boldsymbol{x}_*, \boldsymbol{x}_N)]^\top$, $c = \kappa(\boldsymbol{x}_*, \boldsymbol{x}_*) + \sigma^2$

- The desired predictive posterior will be (using conditional from joint property of Gaussian)

$$\begin{aligned} p(y_*|\boldsymbol{y}) &= \mathcal{N}(y_*|\mu_*, \sigma_*^2) \\ \mu_* &= \mathbf{k}_*^\top \mathbf{C}_N^{-1} \boldsymbol{y} \\ \sigma_*^2 &= \kappa(\boldsymbol{x}_*, \boldsymbol{x}_*) + \sigma^2 - \mathbf{k}_*^\top \mathbf{C}_N^{-1} \mathbf{k}_* \end{aligned}$$

GP Regression: Interpreting GP Predictions

- Let's look at the predictions made by GP regression

$$
\begin{aligned}
p(y_*|\boldsymbol{y}) &= \mathcal{N}(y_*|\mu_*, \sigma_*^2) \\
\mu_* &= \mathbf{k}_*{}^\top \mathbf{C}_N^{-1}\boldsymbol{y} \\
\sigma_*^2 &= k(\boldsymbol{x}_*, \boldsymbol{x}_*) + \sigma^2 - \mathbf{k}_*{}^\top \mathbf{C}_N^{-1}\mathbf{k}_*
\end{aligned}
$$

GP Regression: Interpreting GP Predictions

- Let's look at the predictions made by GP regression

$$
\begin{aligned}
p(y_*|\boldsymbol{y}) &= \mathcal{N}(y_*|\mu_*, \sigma_*^2) \\
\mu_* &= \mathbf{k}_*^\top \mathbf{C}_N^{-1} \boldsymbol{y} \\
\sigma_*^2 &= k(\boldsymbol{x}_*, \boldsymbol{x}_*) + \sigma^2 - \mathbf{k}_*^\top \mathbf{C}_N^{-1} \mathbf{k}_*
\end{aligned}
$$

- Two interpretations for the mean prediction μ_*

GP Regression: Interpreting GP Predictions

- Let's look at the predictions made by GP regression

$$
\begin{aligned}
p(y_*|\boldsymbol{y}) &= \mathcal{N}(y_*|\mu_*, \sigma_*^2) \\
\mu_* &= \mathbf{k}_*{}^\top \mathbf{C}_N^{-1} \boldsymbol{y} \\
\sigma_*^2 &= k(\boldsymbol{x}_*, \boldsymbol{x}_*) + \sigma^2 - \mathbf{k}_*{}^\top \mathbf{C}_N^{-1} \mathbf{k}_*
\end{aligned}
$$

- Two interpretations for the mean prediction μ_*
  - A kernel SVM like interpretation

$$
\mu_* = \mathbf{k}_*{}^\top \mathbf{C}_N^{-1} \boldsymbol{y} = \mathbf{k}_*{}^\top \boldsymbol{\alpha} = \sum_{n=1}^{N} k(\boldsymbol{x}_*, \boldsymbol{x}_n) \alpha_n
$$

where $\boldsymbol{\alpha}$ is akin to the weights of support vectors

GP Regression: Interpreting GP Predictions

- Let's look at the predictions made by GP regression

$$
\begin{aligned}
p(y_*|\boldsymbol{y}) &= \mathcal{N}(y_*|\mu_*, \sigma_*^2) \\
\mu_* &= \mathbf{k}_*{}^\top \mathbf{C}_N^{-1} \boldsymbol{y} \\
\sigma_*^2 &= k(\boldsymbol{x}_*, \boldsymbol{x}_*) + \sigma^2 - \mathbf{k}_*{}^\top \mathbf{C}_N^{-1} \mathbf{k}_*
\end{aligned}
$$

- Two interpretations for the mean prediction μ_*
  - A kernel SVM like interpretation

$$
\mu_* = \mathbf{k}_*{}^\top \mathbf{C}_N^{-1} \boldsymbol{y} = \mathbf{k}_*{}^\top \boldsymbol{\alpha} = \sum_{n=1}^{N} k(\boldsymbol{x}_*, \boldsymbol{x}_n)\alpha_n
$$

    where $\boldsymbol{\alpha}$ is akin to the weights of support vectors

  - A nearest neighbors interpretation

$$
\mu_* = \mathbf{k}_*{}^\top \mathbf{C}_N^{-1} \boldsymbol{y} = \boldsymbol{w}^\top \boldsymbol{y} = \sum_{n=1}^{N} w_n y_n
$$

    where $\boldsymbol{w}$ is akin to the weights of the neighbors

Next Class

- Properties of GP based models, choice of kernels, etc
- Learning hyperparameters in GP based models
- GP for classification and GLMs
- Making GP models scalable
- Some recent advances in GP