Overview

- We’ve covered both parametric and nonparametric models for regression and classification.
  - Parametric models summarize the data into a finite-sized model. E.g., linear regression, logistic regression, neural nets, (linear) SVM, Naïve Bayes, GDA
  - Nonparametric models refer back to the data to make predictions. E.g., KNN
- The next two lectures are about Bayesian approaches to regression.
  - This lecture: Bayesian linear regression, a parametric model
  - Next lecture: Gaussian processes, a nonparametric model
Overview

- We’re going to be Bayesian about the parameters of the model.
  - This is in contrast with naïve Bayes and GDA: in those cases, we used Bayes’ rule to infer the class, but used point estimates of the parameters.
  - By inferring a posterior distribution over the parameters, the model can know what it doesn’t know.

- How can uncertainty in the predictions help us?
  - Smooth out the predictions by averaging over lots of plausible explanations (just like ensembles!)
  - Assign confidences to predictions
  - Make more robust decisions
  - Guide exploration (focus on areas you’re uncertain about)
    - E.g., Bayesian optimization (see next tutorial)
Recap: Linear Regression

- Given a training set of inputs and targets \[ \{(x^{(i)}, t^{(i)})\}_{i=1}^N \]
- Linear model:
  \[ y = \mathbf{w}^\top \psi(x) \]
- Squared error loss:
  \[ \mathcal{L}(y, t) = \frac{1}{2}(t - y)^2 \]
- \( L_2 \) regularization:
  \[ \mathcal{R}(\mathbf{w}) = \frac{\lambda}{2}\|\mathbf{w}\|^2 \]

Solution 1: solve analytically by setting the gradient to 0

\[ \mathbf{w} = (\psi^\top \psi + \lambda \mathbf{I})^{-1} \psi^\top \mathbf{t} \]

Solution 2: solve approximately using gradient descent

\[ \mathbf{w} \leftarrow (1 - \alpha \lambda)\mathbf{w} - \alpha \psi^\top (\mathbf{y} - \mathbf{t}) \]
We can give linear regression a probabilistic interpretation by assuming a Gaussian noise model:

\[
\mathbf{t} \mid \mathbf{x} \sim \mathcal{N}(\mathbf{w}^\top \psi(\mathbf{x}), \sigma^2)
\]

Linear regression is just maximum likelihood under this model:

\[
\frac{1}{N} \sum_{i=1}^{N} \log p(t^{(i)} \mid x^{(i)}; \mathbf{w}, b) = \frac{1}{N} \sum_{i=1}^{N} \log \mathcal{N}(t^{(i)}; \mathbf{w}^\top \psi(\mathbf{x}), \sigma^2)
\]

\[
= \frac{1}{N} \sum_{i=1}^{N} \log \left[ \frac{1}{\sqrt{2\pi}\sigma} \exp \left( -\frac{(t^{(i)} - \mathbf{w}^\top \psi(\mathbf{x}))^2}{2\sigma^2} \right) \right]
\]

\[
= \text{const} - \frac{1}{2N\sigma^2} \sum_{i=1}^{N} (t^{(i)} - \mathbf{w}^\top \psi(\mathbf{x}))^2
\]
Recap: Linear Regression

- We can view an $L_2$ regularizer as MAP inference with a Gaussian prior.
- Recall MAP inference:
  \[
  \arg \max_w \log p(w \mid D) = \arg \max_w \left[ \log p(w) + \log p(D \mid w) \right]
  \]
- We just derived the likelihood term $\log p(D \mid w)$:
  \[
  \log p(D \mid w) = -\frac{1}{2N\sigma^2} \sum_{i=1}^{N} \left( t^{(i)} - w^\top x - b \right)^2 + \text{const}
  \]
- Assume a Gaussian prior, $w \sim \mathcal{N}(m, S)$:
  \[
  \log p(w) = \log \mathcal{N}(w; m, S)
  = \log \left[ \frac{1}{(2\pi)^{D/2} |S|^{1/2}} \exp \left( -\frac{1}{2} (w - m)^\top S^{-1} (w - m) \right) \right]
  = -\frac{1}{2} (w - m)^\top S^{-1} (w - m) + \text{const}
  \]
- Commonly, $m = 0$ and $S = \eta I$, so
  \[
  \log p(w) = -\frac{1}{2\eta} \|w\|^2 + \text{const}.
  \]

This is just $L_2$ regularization!
Recap: Full Bayesian Inference

- Recall: full Bayesian inference makes predictions by averaging over all likely explanations under the posterior distribution.
- Compute posterior using Bayes’ Rule:
  \[
  p(w | D) \propto p(w) p(D | w)
  \]
- Make predictions using the posterior predictive distribution:
  \[
  p(t | x, D) = \int p(w | D) p(t | x, w) \, dw
  \]
- Doing this lets us quantify our uncertainty.
Bayesian linear regression considers various plausible explanations for how the data were generated.

It makes predictions using all possible regression weights, weighted by their posterior probability.

Prior distribution: $w \sim \mathcal{N}(0, S)$

Likelihood: $t \mid x, w \sim \mathcal{N}(w^\top \psi(x), \sigma^2)$

Assuming fixed/known $S$ and $\sigma^2$ is a big assumption. More on this later.
Bayesian Linear Regression: Posterior

- Deriving the posterior distribution:

\[
\log p(w | D) = \log p(w) + \log p(D | w) + \text{const}
\]

\[
= -\frac{1}{2}w^\top S^{-1}w - \frac{1}{2\sigma^2}||\Psi w - t||^2 + \text{const}
\]

\[
= -\frac{1}{2}w^\top S^{-1}w - \frac{1}{2\sigma^2} \left( w^\top \Psi^\top \Psi w - 2t^\top \Psi w + t^\top t \right) + \text{const}
\]

\[
= -\frac{1}{2}(w - \mu)^\top \Sigma^{-1}(w - \mu) + \text{const} \quad \text{(complete the square!)}
\]

where

\[
\mu = \sigma^{-2} \Sigma \Psi^\top t
\]

\[
\Sigma^{-1} = \sigma^{-2} \Psi^\top \Psi + S^{-1}
\]

- This is a multivariate Gaussian distribution, i.e.

\[
w | D \sim \mathcal{N}(\mu, \Sigma)
\]
Bayesian Linear Regression: Posterior

- Just showed:

\[
\begin{align*}
  w \mid D &\sim \mathcal{N}(\mu, \Sigma) \\
  \mu &= \sigma^{-2} \Sigma \Psi^\top t \\
  \Sigma^{-1} &= \sigma^{-2} \Psi^\top \Psi + S^{-1}
\end{align*}
\]

- Since a Gaussian prior leads to a Gaussian posterior, this means the Gaussian distribution is the conjugate prior for linear regression!

- Compare \( \mu \) the closed-form solution for linear regression:

\[
  w = (\Psi^\top \Psi + \lambda I)^{-1} \Psi^\top t
\]
Bayesian Linear Regression

— Bishop, Pattern Recognition and Machine Learning
Bayesian Linear Regression

- Example with radial basis function (RBF) features

\[ \psi_j(x) = \exp \left( -\frac{(x - \mu_j)^2}{2s^2} \right) \]

— Bishop, Pattern Recognition and Machine Learning
Bayesian Linear Regression

Functions sampled from the posterior:

— Bishop, Pattern Recognition and Machine Learning
Posterior predictive distribution:

\[ p(t \mid x, D) = \int \underbrace{p(t \mid x, w)}_{\mathcal{N}(t; w^\top \psi(x), \sigma)} \underbrace{p(w \mid D)}_{\mathcal{N}(w; \mu, \Sigma)} \, dw \]

Another interpretation: \( t = w^\top \psi(x) + \varepsilon \), where \( \varepsilon \sim \mathcal{N}(0, \sigma) \) is independent of \( w \).

By the linear combination rules for Gaussian random variables, \( t \) is a Gaussian distribution with parameters

\[
\begin{align*}
\mu_{\text{pred}} &= \mu^\top \psi(x) \\
\sigma^2_{\text{pred}} &= \psi(x)^\top \Sigma \psi(x) + \sigma^2
\end{align*}
\]

Hence, the posterior predictive distribution is \( \mathcal{N}(t; \mu_{\text{pred}}, \sigma^2_{\text{pred}}) \).
Bayesian Linear Regression

Here we visualize confidence intervals based on the posterior predictive mean and variance at each point:

— Bishop, Pattern Recognition and Machine Learning
Bayesian Decision Theory

- What do we actually do with the posterior predictive distribution $p(t \mid x, \mathcal{D})$?
- Often, we want to make a decision. We can formulate this as minimizing the expected loss under the posterior distribution. This is known as decision theory.
- Simple example: want to choose a single prediction $y$ to minimize the expected squared error loss.

$$\arg \min_y \mathbb{E}_{p(t \mid x, \mathcal{D})}[(y - t)^2] = \mathbb{E}_{p(t \mid x, \mathcal{D})}[t]$$

- Same derivation as bias/variance from Lecture 4
- Similarly, you can show that under absolute value loss, you should pick the median.
Now for a more interesting use of Bayesian decision theory...
Bayesian Optimization

- **Black-box optimization**: we want to minimize a function, but we only get to query function values (i.e. no gradients!)
  - Each query is expensive, so we want to do as few as possible
  - Canonical example: minimize the validation error of an ML algorithm with respect to its hyperparameters

- **Bayesian Optimization**: approximate the function with a simpler function (e.g. linear in a feature representation), called the surrogate function.

- After we’ve queried a certain number of points, we can condition on these to infer the posterior over the surrogate function using Bayesian linear regression.
Bayesian Optimization

To choose the next point to query, we must define an acquisition function, which tells us how promising a candidate it is.

What’s wrong with the following acquisition functions:
- posterior mean: \(-\mathbb{E}[f(\theta)]\)
- posterior variance: \(\text{Var}(f(\theta))\)

Desiderata:
- high for points we expect to be good
- high for points we’re uncertain about
- low for points we’ve already tried

Candidate 1: probability of improvement (PI)

\[
\text{PI} = \text{Pr}(f(\theta) < \gamma - \epsilon),
\]

where \(\gamma\) is the best value so far, and \(\epsilon\) is small.
Bayesian Optimization

Examples:

- Plots show the posterior predictive distribution for $f(\theta)$.
Bayesian Optimization

- The problem with Probability of Improvement (PI): it queries points it is highly confident will have a small improvement
  - Usually these are right next to ones we’ve already evaluated
- A better choice: Expected Improvement (EI)

\[ EI = \mathbb{E}[\max(\gamma - f(\theta), 0)] \]

- The idea: if the new value is much better, we win by a lot; if it’s much worse, we haven’t lost anything.
- There is an explicit formula for this if the posterior predictive distribution is Gaussian.
Bayesian Optimization

Examples:

- Best value so far

El = 0.199

El = 0.004

El = 0.396

El = 0.15
True Function with Three Observations
Bayesian nonlinear regression predictive distributions

- 95%
- 90%
- 80%
How do the predictions compare to the current best?

- 95%
- 90%
- 80%
How do the predictions compare to the current best?

- 95%
- 90%
- 80%

Expected Improvement
Bayesian Optimization

- I showed one-dimensional visualizations, but the higher-dimensional case is conceptually no different.
  - Maximize the acquisition function using gradient descent
  - Use lots of random restarts, since it is riddled with local maxima
  - BayesOpt can be used to optimize tens of hyperparameters.
- I’ve described BayesOpt in terms of Bayesian linear regression with basis functions learned by a neural net.
  - In practice, it’s typically done with Gaussian processes, which are the topic of next lecture.
  - But Bayesian linear regression is actually useful, since it scales better to large numbers of queries.
- One variation: some configurations can be much more expensive than others
  - Use another Bayesian regression model to estimate the computational cost, and query the point that maximizes expected improvement per second
Bayesian Optimization

- BayesOpt can often beat hand-tuned configurations in a relatively small number of steps.
- Results on optimizing hyperparameters (layer-specific learning rates, weight decay, and a few other parameters) for a CIFAR-10 conv net:

Each function evaluation takes about an hour

Human expert = Alex Krizhevsky, the creator of AlexNet
Optional material
Occam’s Razor: “Entities should not be multiplied beyond necessity.”
- Named after the 14th century British theologian William of Occam

Huge number of attempts to formalize mathematically
- See Domingos, 1999, “The role of Occam’s Razor in knowledge discovery” for a skeptical overview.

Common misinterpretation: your prior should favor simple explanations
Occam’s Razor (optional)

- Suppose you have a finite set of models, or hypotheses \( \{ \mathcal{H}_i \}_{i=1}^M \) (e.g. polynomials of different degrees)
- Posterior inference over models (Bayes’ Rule):
  \[
p(\mathcal{H}_i | \mathcal{D}) \propto p(\mathcal{H}_i) p(\mathcal{D} | \mathcal{H}_i)
  \]

  
  
  prior       evidence

- Which of these terms do you think is more important?
- The evidence is also called marginal likelihood since it requires marginalizing out the parameters:

  \[
p(\mathcal{D} | \mathcal{H}_i) = \int p(\mathbf{w} | \mathcal{H}_i) p(\mathcal{D} | \mathbf{w}, \mathcal{H}_i) \, d\mathbf{w}
  \]

- If we’re comparing a handful of hypotheses, \( p(\mathcal{H}_i) \) isn’t very important, so we can compare them based on marginal likelihood.
Occam’s Razor (optional)

- Suppose $M_1$, $M_2$, and $M_3$ denote a linear, quadratic, and cubic model.
- $M_3$ is capable of explaining more datasets than $M_1$.
- But its distribution over $\mathcal{D}$ must integrate to 1, so it must assign lower probability to ones it can explain.

— Bishop, Pattern Recognition and Machine Learning
Occam’s Razor (optional)

- How does the evidence (or marginal likelihood) penalize complex models?

\[
p(D | \mathcal{H}_i) = \int p(D | w, \mathcal{H}_i) p(w | \mathcal{H}_i) \]

\[
\approx p(D | \mathbf{w}_{\text{MAP}}, \mathcal{H}_i) p(\mathbf{w}_{\text{MAP}} | \mathcal{H}_i) \Delta w
\]

- Approximating the integral:
Occam’s Razor (optional)

- Multivariate case:

\[ p(D | H_i) \sim p(D | w_{MAP}, H_i) \cdot p(w_{MAP} | H_i) \cdot |A|^{-1/2}, \]

best-fit likelihood \quad \text{Occam factor}

where \( A = \nabla^2_w \log p(D | w, H_i) \)

- The determinant appears because we’re taking the volume.

- The more parameters in the model, the higher dimensional the parameter space, and the faster the volume decays.

— Bishop, Pattern Recognition and Machine Learning
Analyzing the asymptotic behavior:

\[
\mathbf{A} = \nabla^2_w \log p(\mathcal{D} \mid \mathbf{w}, \mathcal{H}_i) \\
= \sum_{j=1}^{N} \nabla^2_w \log p(y_i \mid x_i, \mathbf{w}, \mathcal{H}_i) \triangleq A_i \\
\approx N \mathbb{E}[A_i]
\]

\[
\log \text{Occam factor} = \log p(\mathbf{w}_{\text{MAP}} \mid \mathcal{H}_i) + \log |\mathbf{A}|^{-1/2} \\
\approx \log p(\mathbf{w}_{\text{MAP}} \mid \mathcal{H}_i) + \log |N \mathbb{E}[A_i]|^{-1/2} \\
= \log p(\mathbf{w}_{\text{MAP}} \mid \mathcal{H}_i) - \frac{1}{2} \log |\mathbb{E}[A_i]| - \frac{D \log N}{2} \\
= \text{const} - \frac{D \log N}{2}
\]

**Bayesian Information Criterion (BIC):** penalize the complexity of your model by \(\frac{1}{2} D \log N\).
Occam’s Razor (optional)

- Summary

\[ p(\mathcal{H}_i \mid \mathcal{D}) \propto p(\mathcal{H}_i) p(\mathcal{D} \mid \mathcal{H}_i) \]

\[ p(\mathcal{D} \mid \mathcal{H}_i) \approx p(\mathcal{D} \mid \mathbf{w}_{\text{MAP}}, \mathcal{H}_i) p(\mathbf{w}_{\text{MAP}} \mid \mathcal{H}_i) |\mathbf{A}|^{-1/2} \]

Asymptotically, with lots of data, this behaves like

\[ \log p(\mathcal{D} \mid \mathcal{H}_i) = \log p(\mathcal{D} \mid \mathbf{w}_{\text{MAP}}, \mathcal{H}_i) - \frac{1}{2} D \log N. \]

- Occam’s Razor is about integration, not priors (over hypotheses).