Basics of Probabilistic ML: Intro to Parameter Estimation

CS772A: Probabilistic Machine Learning

Piyush Rai

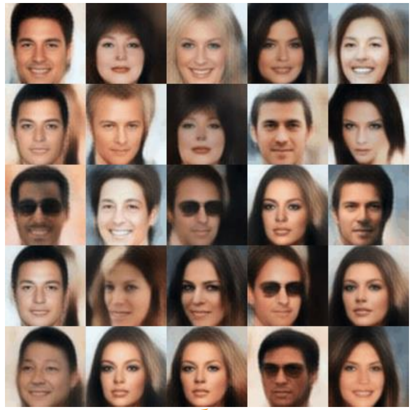
Plan Today

- Some other benefits of probabilistic machine learning
- Some basic ideas
  - Likelihood, prior, posterior, marginal likelihood
  - Parameter estimation via MLE, MAP, and fully Bayesian inference

Use Probabilistic ML also because..

Can Learn Data Distribution and Generate Data

- Often wish to learn the underlying probability distribution $p(\boldsymbol{x})$ of the data from inputs $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N$
- The task is commonly known as generative modeling
- Usually an unsupervised learning problem
- Useful for many tasks, e.g.,
  - Can sample from this distribution to generate new "artificial" but realistic-looking data
  - Outlier/novelty detection: Outliers will have low probability under $p(\boldsymbol{x})$



Several models, such as generative adversarial networks (GAN), variational auto-encoders (VAE), denoising diffusion models, etc can generate realistic looking data (we will study some of these)

The probabilistic perspective of thinking about supervised learning

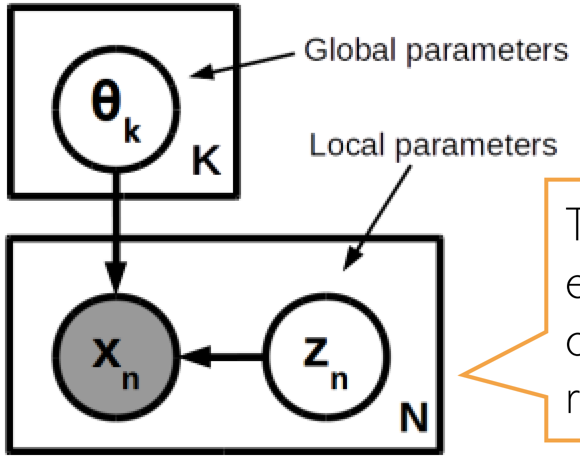
- Note: Even supervised learning problems can be thought of as generative modeling of $p(y|\boldsymbol{x})$ (or if we also wish to model the inputs $\boldsymbol{x}$ then of $p(\boldsymbol{x}, y)$ using which we can get $p(y|\boldsymbol{x})$ via Bayes rule)

Pic credit: https://medium.com/analytics-vidhya/an-introduction-to-generative-deep-learning-792e93d1c6d4

Learning Latent Structures within Data

- Can endow generative models of data with latent variables. For example:

Each data point $\boldsymbol{x}_n$ is associated with a latent variable $\boldsymbol{z}_n$



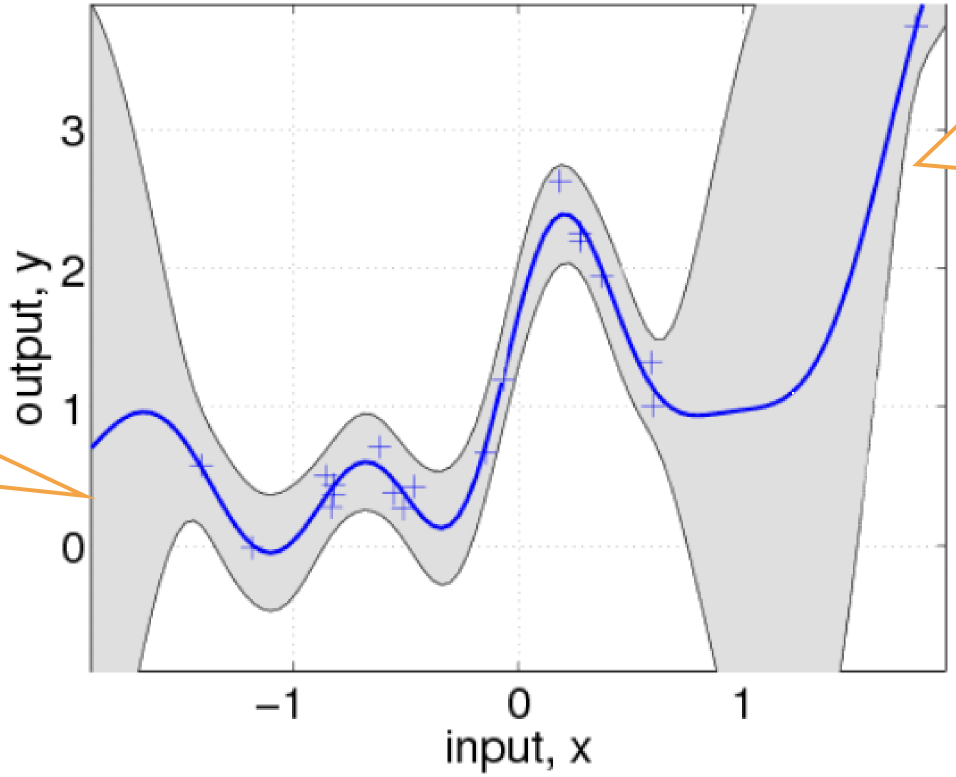
The latent variable $\boldsymbol{z}_n$ can be used to encode some property of $\boldsymbol{x}_n$ (e.g., its cluster membership, or its low-dim representation, or missing parts of $\boldsymbol{x}_n$)

- Such models are used in many problems, especially unsupervised learning: Gaussian mixture model, probabilistic PCA, topic models, deep generative models, etc.

- We will look at several of these in this course and way to learn such models

Helps in Sequential Decision-Making Problems

- Sequential decision-making: Information about uncertainty can "guide" us, e.g.,



Blue curve is the mean of the function (learned so far using the available data), shaded region denotes the current predictive uncertainty

Given our current estimate of the regression function, which training input(s) should we add next to improve its estimate the most?

Uncertainty can help here: Acquire training inputs from regions where the function is most uncertain about its current predictions

- Applications in active learning, reinforcement learning, Bayesian optimization, etc

Can Better Handle OOD Data

For classification, "confidence" refers to the probability of the class predicted to be the most likely

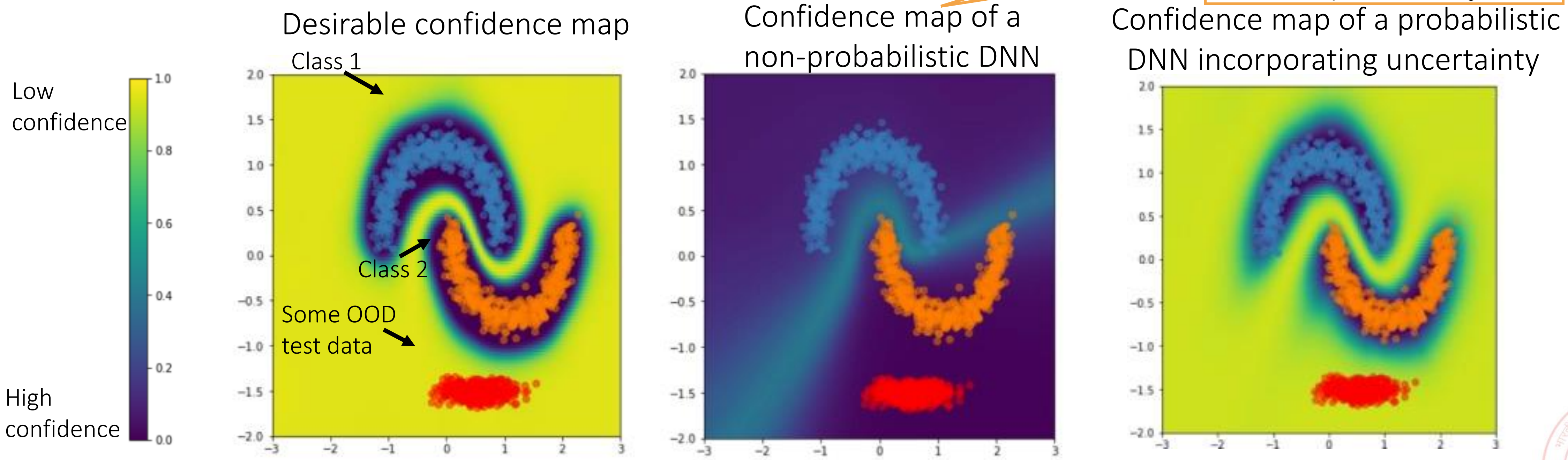
- Many modern deep neural networks (DNN) tend to be overconfident

One of the reasons is that they don't incorporate uncertainty

- Especially true if test data is "out-of-distribution (OOD)"

Overconfident model

Model has high confidence for predictions on even inputs that are far away from training data



- Prob. deep models often provide better uncertainty estimates to flag OOD data

Image source: "Simple and Principled Uncertainty Estimation with Deterministic Deep Learning via Distance Awareness" (Liu et al, 2020)

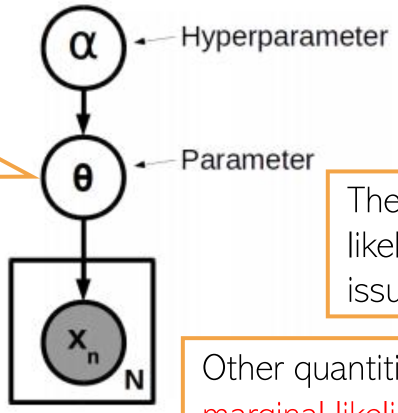
Hyperparameter Estimation

- ML models invariably have hyperparams, e.g., regularization/kernel h.p. in a linear/kernel regression, h.p.'s of a deep neural network, etc.
- Can specify the h.p.'s as additional unknowns and estimate them as well

Pretty much the same way we estimate other unknowns

This doesn't require a separate validation set unlike cross-validation

Assuming θ (its prior distribution) depends on α



Marginal likelihood (more on this later) is like an "averaged" likelihood (averaged over all parameters drawn from the prior)

A way to find the point estimate of the hyperparameters by maximizing the marginal likelihood

The approach of using marginal likelihood for doing such thing has some issues (e.g., dependence on the prior)

Other quantities can be used such as "conditional" marginal likelihood* $\sum_{i=k}^{N} \log p(x_i|\mathbf{X}_{<i}, \alpha)$ for $k >= 1$ (more on this later)

$$\hat{\alpha} = \arg\max_{\alpha} \log p(\mathbf{X}|\alpha)$$
$$= \arg\max_{\alpha} \log \int p(\mathbf{X}|\theta)p(\theta|\alpha)\theta$$

- Can then estimate them, e.g., using a point estimate or a posterior distribution
  - To find point estimate of h.p.'s, we can maximize $p(\mathbf{X}|\alpha)$ w.r.t. the h.p.'s (details later)
  - Posterior on h.p.'s can also be estimated using a prior on them (details later)

*Bayesian Model Selection, the Marginal Likelihood, and Generalization (Lotfi et al, 2022)

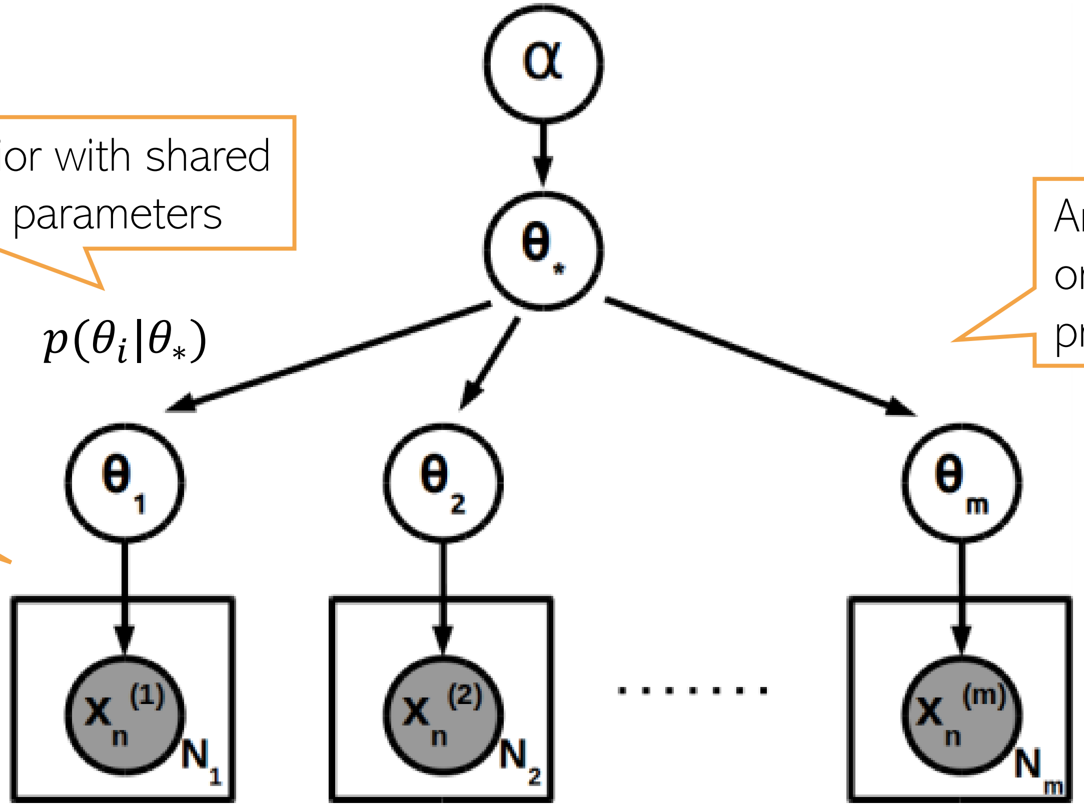
Hierarchical Modeling

- Can design models that can jointly learn from multiple datasets and share information across multiple datasets using shared parameters with a prior distribution

Example: Estimating the means of m datasets, assuming the means are somewhat related. Can do this jointly rather than estimating independently

Prior with shared θ_* parameters

$p(\theta_i|\theta_*)$



An example of transfer learning or multitask learning using a probabilistic approach

Helps especially if the amount of training data per task is small

Easy to do it using a probabilistic approach with shared parameters (will see details later)

Non-probabilistic ML Methods?

- Some non-probabilistic ML methods can give probabilistic answers via heuristics

- Doesn't mean these methods are not useful/used but they don't follow the PML paradigm, so we won't study them in this course

Or methods like Platt Scaling used to get class probabilities for SVMs

- Some examples which you may have seen
  - Converting distances from hyperplane (in hyperplane classifiers) to compute class probabilities
  - Using class-frequencies in nearest neighbors to compute class probabilities

  - Using class-frequencies at leaves of a Decision Tree to compute class probabilities
  - Soft k-means clustering to compute probabilistic cluster memberships

Basics of Probabilistic ML

Probabilistic Modeling

- Assume data $\mathbf{X} = \{\boldsymbol{x}_n\}_{n=1}^N$ generated from a prob distribution with params $\boldsymbol{\theta}$

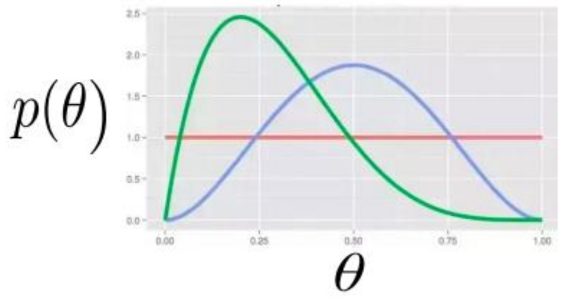
$$\boldsymbol{x}_n \sim p(\boldsymbol{x}|\theta, m) \qquad\qquad n = 1,2,\dots,N$$

- $p(\boldsymbol{x}|\theta, m)$ is also known as the likelihood (a function of the parameters $\boldsymbol{\theta}$)
- Assume a prior distribution $p(\boldsymbol{\theta}|\boldsymbol{m})$ on the parameters $\boldsymbol{\theta}$
- Note: Here $\boldsymbol{m}$ collectively denotes "all other stuff" about the model, e.g.,
  - An "index" for the type of model being considered (e.g., the type of distribution for $\boldsymbol{x}$)
  - Any other (hyper)parameters of the likelihood/prior
- Note: Usually we will omit the explicit use of $\boldsymbol{m}$ in the notation
  - In some situations (e.g., when doing model comparison/selection), we will use it explicitly
- Note: For some models, the likelihood is not defined explicitly using a probability distribution but implicitly† via a probabilistic simulation process

†Hierarchical Implicit Models and Likelihood-Free Variational Inference (Tran et al (NIPS 2017))

Probabilistic Modeling

- The prior $p(\theta|m)$ plays an important role in probabilistic/Bayesian modeling
  - Reflects our prior beliefs about possible parameter values before seeing the data

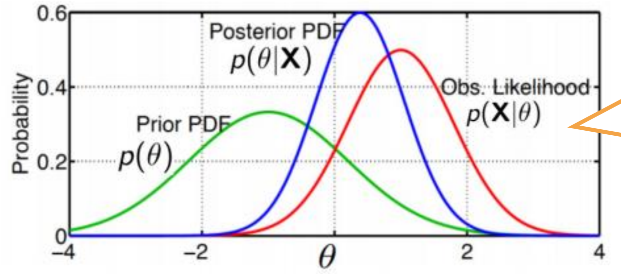


  - Can be "subjective" or "objective" (also a topic of debate, which we won't get into)
  - Subjective: Prior (our beliefs) derived from past experiments
  - Objective: Prior represents "neutral knowledge" (e.g.. uniform, vague prior)
  - Can also be seen as a regularizer (connection with non-probabilistic view)
- The goal of probabilistic modeling is usually one or more of the following
  - Infer the unknowns/parameters $\boldsymbol{\theta}$ given data $\mathbf{X}$ (to summarize/understand the data)
  - Use the inferred quantities to make predictions

Parameter Estimation/Inference

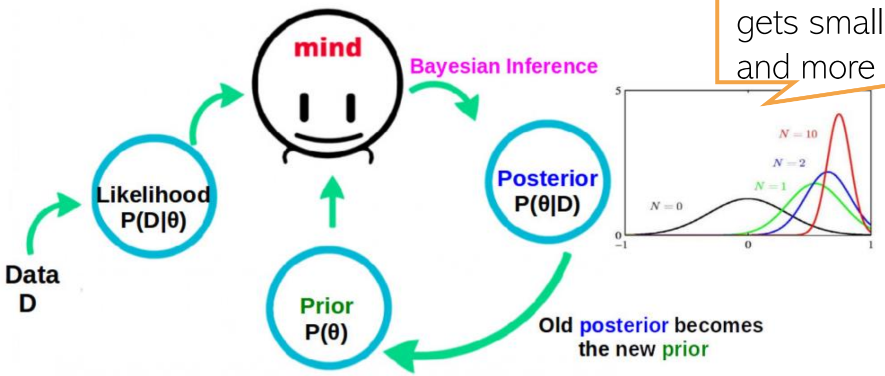
- Can infer params by computing posterior distribution (fully Bayesian inference)

$$p(\theta|\mathbf{X},m) = \frac{p(\mathbf{X}|\theta,m)p(\theta|m)}{p(\mathbf{X}|m)} = \frac{p(\mathbf{X}|\theta,m)p(\theta|m)}{\int p(\mathbf{X}|\theta,m)p(\theta|m)d\theta} = \frac{\text{likelihood} \times \text{prior}}{\text{marginal likelihood}}$$



Note: Prior and posterior are distributions over θ. Likelihood is just a function of θ

Posterior's spread/variance gets smaller as we use more and more data to infer it



- Marginal likelihood is an important quantity (used for hyperparam est. or model sel.)
  - It's the probability of data after integrating out some/all of the unknowns from the likelihood $p(\mathbf{X}|\theta,m)$
  - $p(\mathbf{X}|m)$ above is the likelihood obtained after integrating out θ from the likelihood $p(\mathbf{X}|\theta,m)$
  - Not always available in closed form (the key reason why full posterior is often hard to compute)

Point Estimation of Parameters

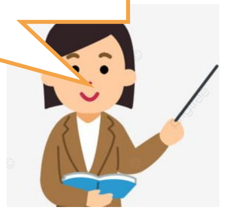
- Recall that the posterior is

Intractable mainly because the marginal likelihood (the denominator on the RHS is intractable in general)

Intractable to compute except for some very simple models or if the likelihood and prior are conjugate (discussed later) to each other

$$p(\theta|\mathbf{X},m) = \frac{p(\mathbf{X}|\theta,m)p(\theta|m)}{p(\mathbf{X}|m)}$$

In some problems as we will see, hybrid inference is also possible/desirable – infer full posterior for some parameters and point estimate for others



- Point estimation is a cheaper alternative to computing the full posterior
  - Maximum likelihood (ML) estimation: Find $\boldsymbol{\theta}$ for which <u>observed data has largest probability</u>

Negative Log likelihood (equivalent to a loss function)

$$\hat{\theta}_{ML} = \arg\max_{\theta} \log p(\mathbf{X}|\theta) = \arg\min_{\theta} -\log p(\mathbf{X}|\theta) = \arg\min_{\theta} NLL(\theta)$$

  - Maximum a posteriori (MAP) estimation: Find $\boldsymbol{\theta}$ that has the <u>largest posterior probability</u>

$$\hat{\theta}_{MAP} = \arg\max_{\theta} \log p(\theta|\mathbf{X}) = \arg\max_{\theta} [\log p(\mathbf{X}|\theta) + \log p(\theta)]$$

$$= \arg\min_{\theta} [NLL(\theta) - \log p(\theta)]$$

Like MLE with info from prior added

Akin to a regularizer added to the loss

Note: The regularizer hyperparameter is part of the prior

Making Predictions: Predictive Distribution

- Posterior can be used to compute the posterior predictive distribution (PPD)
- PPD is essentially our test time prediction using the learned model
- The PPD of a new observation $\boldsymbol{x}_*$ given previous observations $\mathbf{X}$ (m assumed fixed)

New (test) data

Past (training) data

$$p(\boldsymbol{x}_*|\mathbf{X}, m) = \int p(\boldsymbol{x}_*, \theta|\mathbf{X}, m)\, d\theta \;= \int p(\boldsymbol{x}_*|\theta, \mathbf{X}, m) p(\theta|\mathbf{X}, m)\, d\theta$$

$$= \int p(\boldsymbol{x}_*|\theta, m) p(\theta|\mathbf{X}, m)\, d\theta$$

Just a simple example. The actual form of PPD (e.g., what we are predicting and what we condition on, etc) will depend on the problem, e.g., $p(y_*|\boldsymbol{x}_*, \mathbf{X}, \boldsymbol{y})$ in supervised learning

Assuming observations are i.i.d. given θ

This integral is only rarely tractable

Prediction by averaging over the posterior distribution of the unknowns parameters

- Computing PPD requires doing a posterior-weighted averaging over all values of θ
- A crude approximation: Instead of PPD, just use a plug-in predictive distribution

$$p(\boldsymbol{x}_*|\mathbf{X}, m) \approx p(\boldsymbol{x}_*|\hat{\theta}, m)$$

Here $\hat{\theta}$ is the ML or MAP estimate of the parameters

However, this ignores all the uncertainty about θ

- Plug-in pred. is the same as PPD with $p(\theta|\mathbf{X}, m)$ approximated by a point mass at $\hat{\theta}$
  - If we are using plug-in predictive, we are not really being Bayesian!

Model Selection and Model Averaging

- Can use Bayes rule to find the best model from a set of models $m = 1,2,\ldots,M$

Posterior probability of model m

Marginal likelihood of model m

Prior probability of choosing model m

Will discuss later how to compute marginal likelihood

$$p(m|\mathbf{X}) = \frac{p(\mathbf{X}|m)p(m)}{p(\mathbf{X})} = \frac{p(\mathbf{X}|m)p(m)}{\sum_{m=1}^{M} p(\mathbf{X}|m)p(m)}$$

Marginal likelihood over all models

In general, intractable to compute exactly

$$p(\mathbf{X}|m) = \int p(\mathbf{X}|\theta, m)p(\theta|m)d\theta$$

Integrating out all unknown parameters of the model

Best model

$$\hat{m} = \arg\max_m p(m|\mathbf{X}) = \arg\max_m p(\mathbf{X}|m)p(m)$$

- If all models equally likely a priori then $\hat{m} = \arg\max_m p(\mathbf{X}|m)$

- For PPD, can use either the best model $\hat{m}$ or can average over all models

Test data

Training data

$$p(x_*|\mathbf{X}) \approx p(x_*|\mathbf{X}, \hat{m}) \quad \underline{\text{OR}} \quad p(x_*|\mathbf{X}) = \sum_{m=1}^{M} p(x_*|\mathbf{X}, m)p(m|\mathbf{X})$$

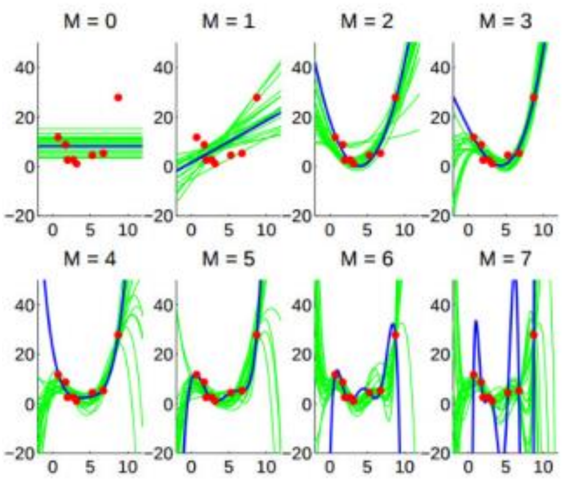
Marginal Likelihood: An Illustration

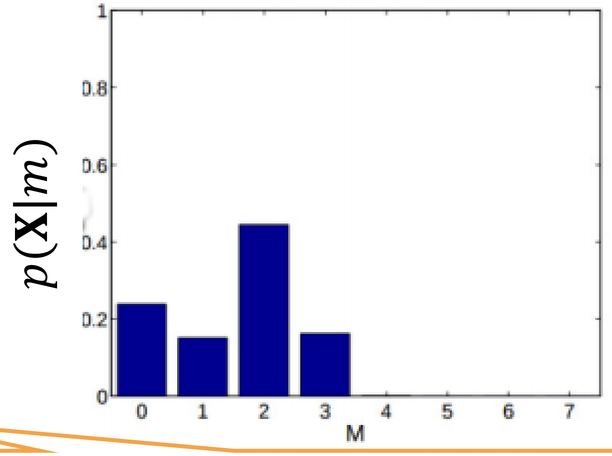
- Marginal likelihood is a hard-to-compute but an important quantity
  - $p(\mathbf{X}|\alpha)$ where α is a hyperparameter can be used to find the best hyperparameter
  - $p(\mathbf{X}|m)$ where m is a model index can be used to find the best model
- Recall that marg. lik. is akin to "averaged" likelihood: $p(\mathbf{X}|m) = \int p(\mathbf{X}|\theta, m)p(\theta|m)d\theta$

Fitting regression models with polynomial degree m

Green lines/curves in each plot are parameters drawn from the prior $p(\theta|m)$

For a good model, most parameters from the prior will fit the expected trend reasonably (thus their averaged likelihood will be large). For a bad model, only a few params will fit well and others won't (e.g., $m = 4 - 7$ in right fig)





No validation data needed

Note that we can get these plots (and compute marginal likelihood) before doing parameter estimation for each model