Probabilistic Supervised Learning

CS772A: Probabilistic Machine Learning

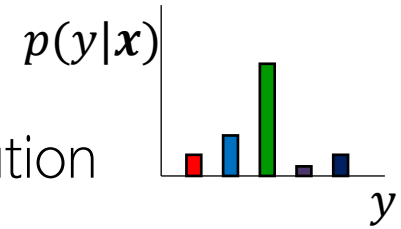
Piyush Rai

Plan Today

- Probabilistic Supervised Learning
- Discriminative vs Generative Supervised Learning
- Discriminative Supervised Learning
  - (Probabilistic) Linear Regression
  - Logistic and Softmax/Multinoulli Regression
  - Generalized Linear Models

Probabilistic Supervised Learning

- Goal: To learn the conditional distribution $p(y|x)$ of output given input
- The form of the distribution $p(y|x)$ depends on output type, e.g.,
  - Real: Model $p(y|x)$ using a Gaussian (or some other suitable real-valued distribution)
  - Binary: Model $p(y|x)$ using a Bernoulli
  - Categorical/multiclass: Model $p(y|x)$ using a multinoulli/categorical distribution



  - Various other types (e.g., count, positive reals, etc) can also be modeled using appropriate distributions (e.g., Poisson for count, gamma for positive reals)
- The distribution $p(y|x)$ can be defined directly or indirectly

"Direct" way without modeling the inputs x_n

Parameters of this distribution are the outputs of function f

$$p(y|x) = p(y|f(x, w))$$

"Indirect" way by modeling the outputs as well as the inputs

"Indirect" way requires first learning the joint distribution of inputs and outputs

$$p(y|x) = \frac{p(y, x)}{p(x)}$$

Discriminative vs Generative Sup. Learning

Non-probabilistic supervised learning approaches (e.g., SVM) are usually considered discriminative since $p(x)$ is never modeled

- Direct way of sup. learning is discriminative, indirect way is generative

Discriminative Approach

$$p(y|\boldsymbol{x}) = p(y|f(\boldsymbol{x}, \boldsymbol{w}))$$

f can be any function which uses inputs and weights $\boldsymbol{w}$ to defines parameters of distr. p

Some examples

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

$$p(y|\boldsymbol{x}) = \text{Bernoulli}(y|\sigma(\boldsymbol{w}^\top \boldsymbol{x}))$$

Generative Approach

$$p(y|\boldsymbol{x}) = \frac{p(y, \boldsymbol{x})}{p(\boldsymbol{x})}$$

Requires estimating the joint distribution of inputs and outputs to get the conditional $p(y|\boldsymbol{x})$ (unlike the discriminative approach which directly estimates the conditional $p(y|\boldsymbol{x})$ and does not model the distribution of $\boldsymbol{x}$)

- Note: Generative approach can also be used for other settings too, such as unsupervised learning and semi-supervised learning (will see later)

Probabilistic Linear Regression

A discriminative model for regression problems

- Assume training data $\{\boldsymbol{x}_n, y_n\}_{n=1}^N$, with features $\boldsymbol{x}_n \in \mathbb{R}^D$ and responses $y_n \in \mathbb{R}$
- Assume y_n generated by a noisy linear model with wts $\boldsymbol{w} = [w_1, \ldots, w_D] \in \mathbb{R}^D$

Unknown to be estimated

Each weight assumed real-valued

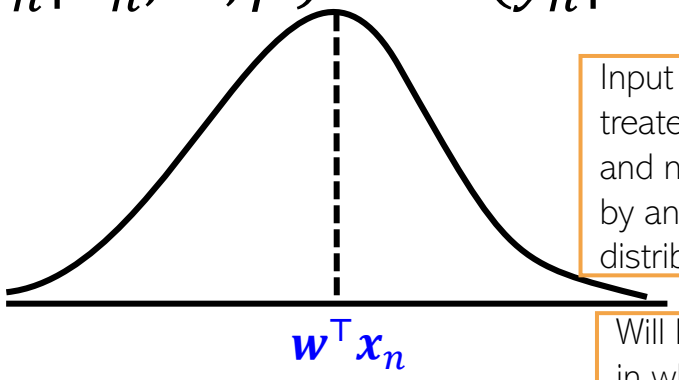
$$y_n = \boldsymbol{w}^\top \boldsymbol{x}_n + \epsilon_n$$

Gaussian noise drawn from $\mathcal{N}(\epsilon_n | 0, \beta^{-1})$

- Notation alert: β is the precision of Gaussian noise (and β^{-1} the variance)

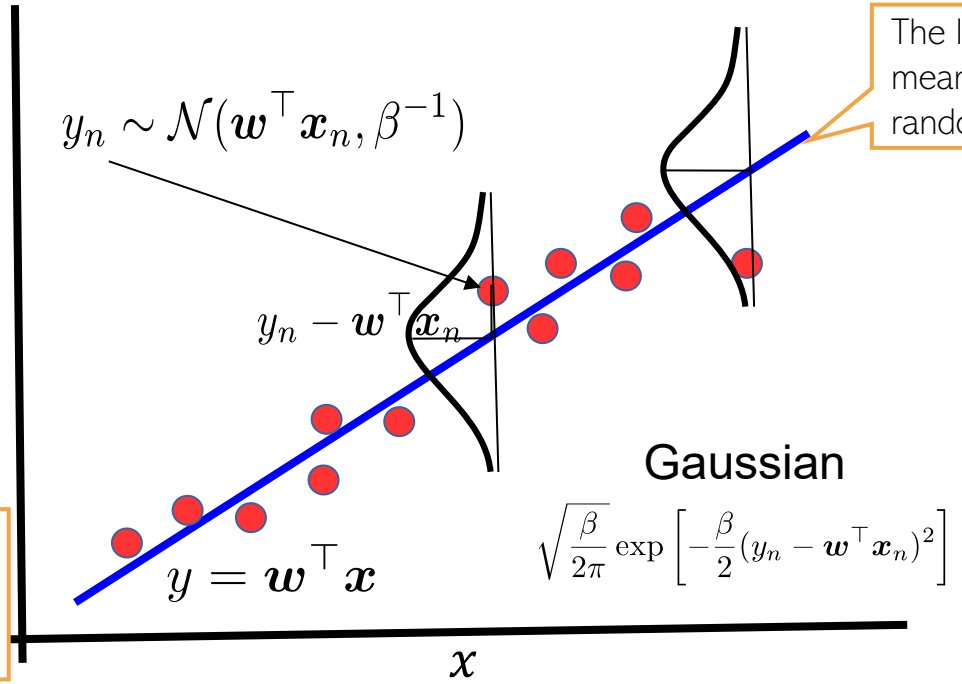
Likelihood model

$$p(y_n | \boldsymbol{x}_n, \boldsymbol{w}, \beta) = \mathcal{N}(y_n | \boldsymbol{w}^\top \boldsymbol{x}_n, \beta^{-1})$$



Input $\boldsymbol{x}_n$ being treated as given and not modeled by any probability distribution

Will later study models in which both input and output are modeled by distributions



The line represents the mean $\boldsymbol{w}^\top \boldsymbol{x}_n$ of the output random variable y_n

The zero mean Gaussian noise perturbs the output from its mean

Thus NLL is like squared loss

Probabilistic Linear Regression

- For all the training data, we can write the above model in matrix-vector notation

$\boldsymbol{y} = [y_1; y_2; \ldots; y_N]$ is the $N \times 1$ response vector

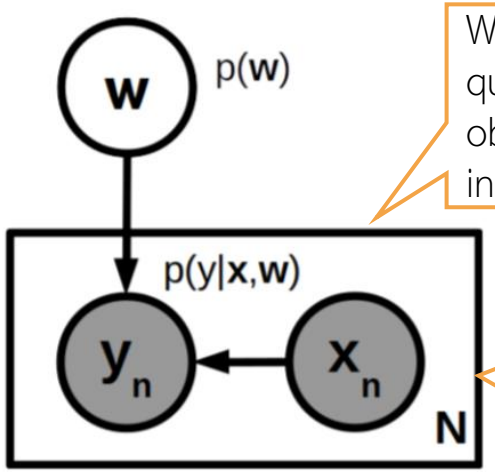
$\boldsymbol{X} = [\boldsymbol{x}_1^\top; \boldsymbol{x}_2^\top; \ldots; \boldsymbol{x}_N^\top]$ is the $N \times D$ input matrix

$\boldsymbol{\epsilon} = [\epsilon_1; \epsilon_2; \ldots; \epsilon_N]$ is the $N \times 1$ noise vector drawn from $\mathcal{N}(\boldsymbol{0}, \beta^{-1}\boldsymbol{I}_N)$

Same as writing $p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{w}, \beta) = \mathcal{N}(\boldsymbol{y}|\boldsymbol{X}\boldsymbol{w}, \beta^{-1}\boldsymbol{I}_N)$

$$\boldsymbol{y} = \boldsymbol{X}\boldsymbol{w} + \boldsymbol{\epsilon}$$

- This is a linear Gaussian model with $\boldsymbol{w}$ being the unknown Gaussian r.v.
- A simple "plate diagram" for this model would look like this (hyperparameters not shown in the diagram)



Direction of arrow show dependency

White nodes denote unknown quantities, grey nodes denote observed quantities (training input-output pairs)

The plate/box with number N shows that we have N such i.i.d. observations

On compact notations..

- When writing the likelihood (assuming y_n's are i.i.d. given $\boldsymbol{w}$ and $\boldsymbol{x}_n$)

$$p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{w}, \beta) = \prod_{n=1}^{N} \mathcal{N}(y_n|\boldsymbol{w}^\top \boldsymbol{x}_n, \beta^{-1})$$

$$= \mathcal{N}(\boldsymbol{y}|\boldsymbol{X}\boldsymbol{w}, \beta^{-1}\boldsymbol{I}_N)$$

- Thus a product of $\boldsymbol{N}$ univariate Gaussians here (not always) is equivalent to an $\boldsymbol{N}$-dim Gaussian over the vector $\boldsymbol{y} = [y_1, y_2, \ldots, y_N]$

- We will prefer to use this equivalence at other places too whenever we have multiple i.i.d. random variables, each having a univariate Gaussian distribution

Prior on weights

- Assume a zero-mean Gaussian prior on $\boldsymbol{w}$

$$p(\boldsymbol{w}|\lambda) = \prod_{d=1}^{D} p(w_d|\lambda) = \prod_{d=1}^{D} \mathcal{N}(w_d|0, \lambda^{-1})$$

May also use a non-zero mean Gaussian prior, e.g., $\mathcal{N}(w_d|\mu, \lambda^{-1})$ if we expect weights to be close to some value μ

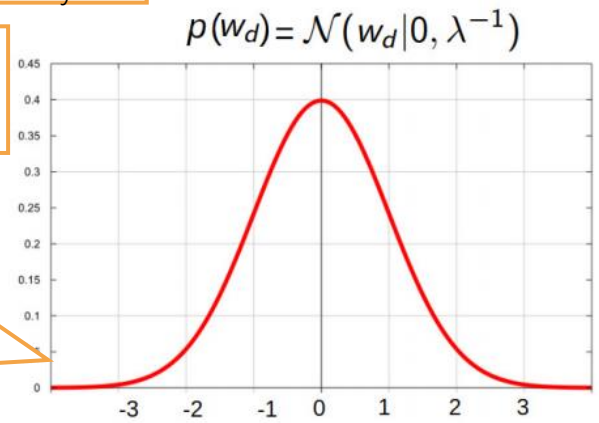
This prior assumes that *a priori* each weight has a small value (close to zero)

λ controls the uncertainty around our prior belief about value of w_d

In zero-mean case, λ sort of denotes each feature's importance. Think why?

Large λ means more aggressive push towards zero

The precision λ controls how aggressively the prior pushes w_d towards mean (0)



$$= \mathcal{N}(\boldsymbol{w}|\boldsymbol{0}, \lambda^{-1}\mathbf{I}_D)$$

$$\propto \left(\frac{\lambda}{2\pi}\right)^{\frac{D}{2}} \exp\left[-\frac{\lambda}{2}\boldsymbol{w}^\top\boldsymbol{w}\right]$$

Can also use a full covariance matrix Λ^{-1} for the prior to impose a priori correlations among different weights

Prior's hyperparameters ($\boldsymbol{\lambda}/\boldsymbol{\Lambda}/\boldsymbol{\mu}$) etc can be learned as well using point estimation (e.g., MLE-II) or fully Bayesian inference

- Zero-mean Gaussian prior corresponds to ℓ_2 regularizer

Reason: The negative log prior $-\log p(\boldsymbol{w}) \propto \frac{\lambda}{2}\boldsymbol{w}^\top\boldsymbol{w}$

The Posterior

MLE/MAP left as an exercise

- The posterior over $\boldsymbol{w}$ (for now, assume hyperparams β and λ to be known)

$$p(\boldsymbol{w}|\boldsymbol{y}, \mathbf{X}, \beta, \lambda) = \frac{p(\boldsymbol{w}|\lambda)p(\boldsymbol{y}|\boldsymbol{w}, \mathbf{X}, \beta)}{p(\boldsymbol{y}|\mathbf{X}, \beta, \lambda)} \propto p(\boldsymbol{w}|\lambda)p(\boldsymbol{y}|\boldsymbol{w}, \mathbf{X}, \beta)$$

Marginal likelihood for this regression model. Note that it is conditioned on $\mathbf{X}$ too which is assumed given and not being modeled

Must be a Gaussian due to conjugacy

$$p(\boldsymbol{w}|\boldsymbol{y}, \mathbf{X}, \beta, \lambda) \propto \mathcal{N}(\boldsymbol{w}|\mathbf{0}, \lambda^{-1}\mathbf{I}_D) \times \mathcal{N}(\boldsymbol{y}|\mathbf{X}\boldsymbol{w}, \beta^{-1}\mathbf{I}_N)$$

- Using the "completing the squares" trick (or linear Gaussian model results)

$$p(\boldsymbol{w}|\boldsymbol{y}, \mathbf{X}, \beta, \lambda) = \mathcal{N}(\boldsymbol{\mu}_N, \boldsymbol{\Sigma}_N)$$

Note that λ and β can be learned under the probabilistic set-up (though assumed fixed as of now)

$$\text{where} \quad \boldsymbol{\Sigma}_N = (\beta \sum_{n=1}^{N} \boldsymbol{x}_n \boldsymbol{x}_n^\top + \lambda \mathbf{I}_D)^{-1} = (\beta \mathbf{X}^\top \mathbf{X} + \lambda \mathbf{I}_D)^{-1} \quad \text{(posterior's covariance matrix)}$$

The form is also similar to the solution to ridge regression $\text{argmin}_w ||y - Xw||^2 + \lambda w^\top w = (X^\top X + \lambda I)^{-1} X^\top y$

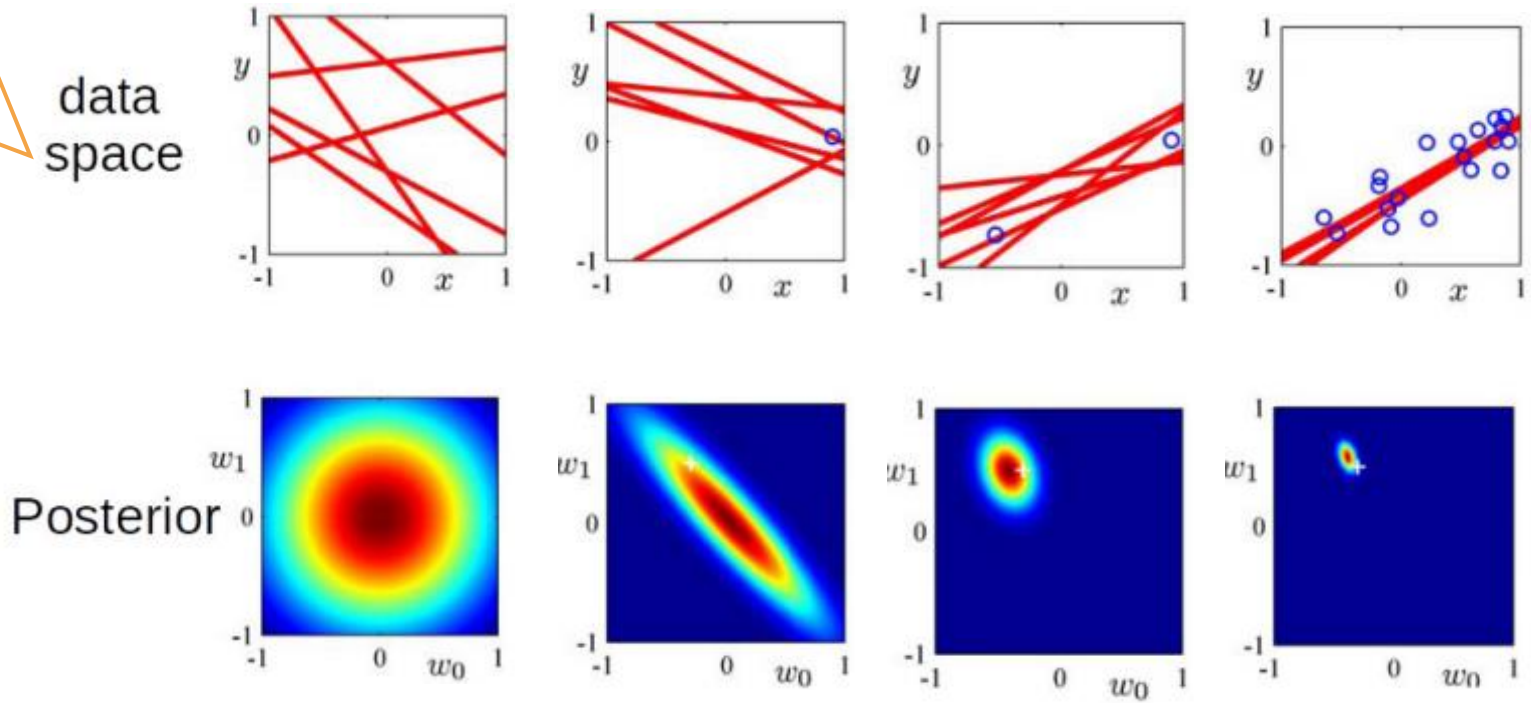
MAP solution turns out to be exactly the same (reason: Gaussian's mean and mode are the same)

$$\boldsymbol{\mu}_N = \boldsymbol{\Sigma}_N \left[\beta \sum_{n=1}^{N} y_n \boldsymbol{x}_n\right] = \boldsymbol{\Sigma}_N \left[\beta \mathbf{X}^\top \boldsymbol{y}\right] = (\mathbf{X}^\top \mathbf{X} + \frac{\lambda}{\beta}\mathbf{I}_D)^{-1} \mathbf{X}^\top \boldsymbol{y} \quad \text{(posterior's mean)}$$

The Posterior: A Visualization

- Assume a lin. reg. problem with true $\boldsymbol{w} = [w_0, w_1], w_0 = -0.3, w_1 = 0.5$
- Assume data generated by a linear regression model $y = w_0 + w_1 x + \text{"noise"}$
  - Note: It's actually 1-D regression ($\boldsymbol{w_0}$ is just a bias term), or 2-D reg. with feature $[1, x]$
- Figures below show the "data space" and posterior of $\boldsymbol{w}$ for different number of observations (note: with no observations, the posterior = prior)

Each red line represents the "data" generated for a randomly drawn $\boldsymbol{w}$ from the current posterior



Posterior Predictive Distribution

- To get the prediction y_* for a new input $\boldsymbol{x}_*$, we can compute its PPD

$$p(y_*|\boldsymbol{x}_*, \mathbf{X}, \boldsymbol{y}, \beta, \lambda) = \int p(y_*|\boldsymbol{x}_*, \boldsymbol{w}, \beta) p(\boldsymbol{w}|\mathbf{X}, \boldsymbol{y}, \beta, \lambda) d\boldsymbol{w}$$

$\mathcal{N}(y_*|\boldsymbol{w}^\top \boldsymbol{x}_*, \beta^{-1})$ $\quad$ $\mathcal{N}(\boldsymbol{w}|\boldsymbol{\mu}_N, \boldsymbol{\Sigma}_N)$

Only $\boldsymbol{w}$ is unknown with a posterior distribution so only $\boldsymbol{w}$ has to be integrated out

- The above is the marginalization of $\boldsymbol{w}$ from $\mathcal{N}(y_*|\boldsymbol{w}^\top \boldsymbol{x}_*, \beta^{-1})$. Using LGM results

$$p(y_*|\boldsymbol{x}_*, \mathbf{X}, \boldsymbol{y}, \beta, \lambda) = \mathcal{N}(\boldsymbol{\mu}_N^\top \boldsymbol{x}_*, \beta^{-1} + \boldsymbol{x}_*^\top \boldsymbol{\Sigma}_N \boldsymbol{x}_*)$$

Can also derive it by writing $y_* = \boldsymbol{w}^\top \boldsymbol{x}_* + \epsilon$ where $\boldsymbol{w} \sim \mathcal{N}(\boldsymbol{\mu}_N, \boldsymbol{\Sigma}_N)$ and $\epsilon \sim \mathcal{N}(0, \beta^{-1})$

- So we have a predictive mean $\boldsymbol{\mu}_N^\top \boldsymbol{x}_*$ as well as an input-specific predictive variance
- In contrast, MLE and MAP make "plug-in" predictions (using the point estimate of $\boldsymbol{w}$)

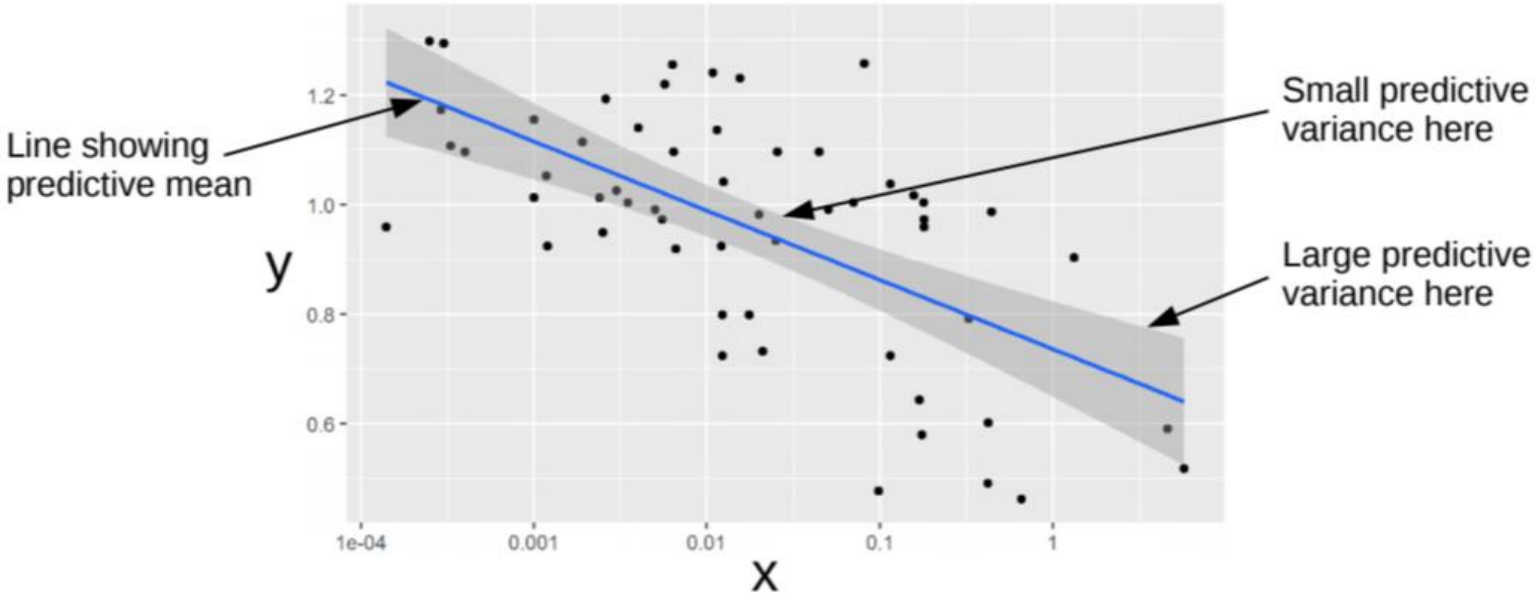
$$p(y_*|\boldsymbol{x}_*, \boldsymbol{w}_{MLE}) = \mathcal{N}(\boldsymbol{w}_{MLE}^\top \boldsymbol{x}_*, \beta^{-1}) \quad \text{- MLE prediction}$$
$$p(y_*|\boldsymbol{x}_*, \boldsymbol{w}_{MAP}) = \mathcal{N}(\boldsymbol{w}_{MAP}^\top \boldsymbol{x}_*, \beta^{-1}) \quad \text{- MAP prediction}$$

Since PPD also takes into account the uncertainty in $\boldsymbol{w}$, the predictive variance is larger

- Unlike MLE/MAP, variance of y_* also depends on the input $\boldsymbol{x}_*$ (this, as we will see later, will be very useful in sequential decision-making problems such as active learning)

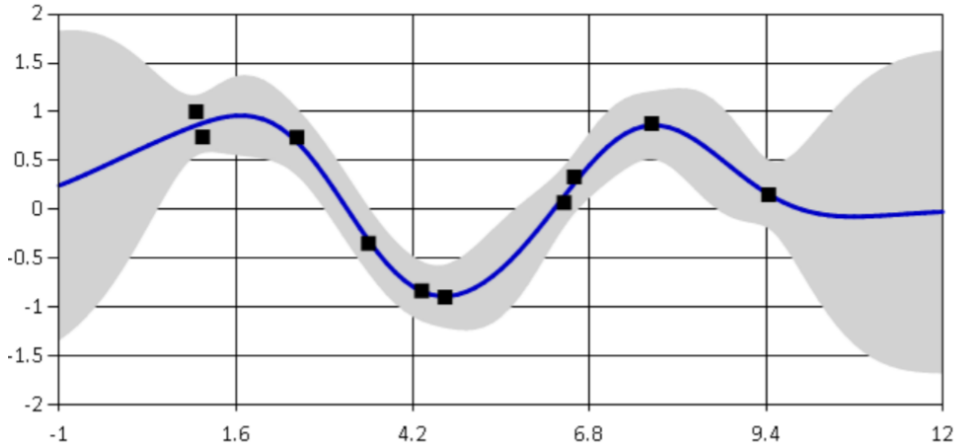
Posterior Predictive Distribution: An Illustration

- Black dots are training examples



- Width of the shaded region at any $\boldsymbol{x}$ denotes the predictive uncertainty at that $\boldsymbol{x}$ (+/- one std-dev)
- Regions with more training examples have smaller predictive variance

Nonlinear Regression



- Can extend the linear regression model to handle nonlinear regression problems
- One way is to replace the feature vectors $\boldsymbol{x}$ by a nonlinear mapping $\boldsymbol{\phi(x)}$

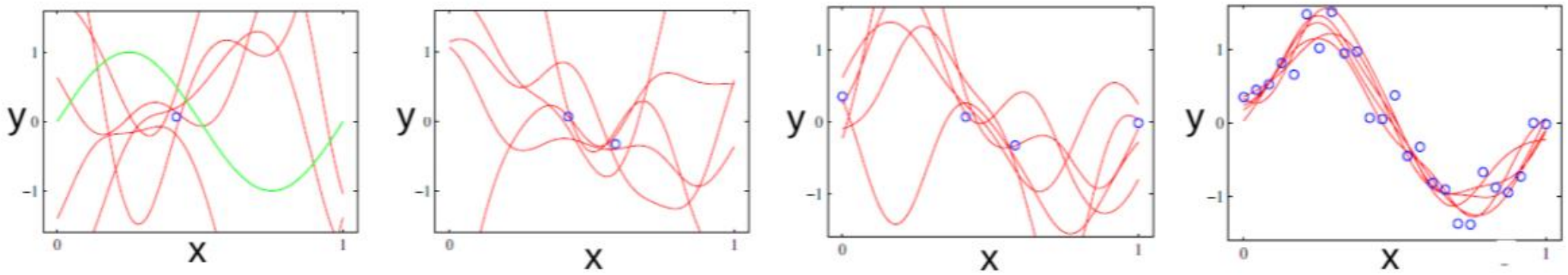
$$p(y|\boldsymbol{x}, \boldsymbol{w}) = \mathcal{N}(\boldsymbol{w}^\top \phi(\boldsymbol{x}), \beta^{-1})$$

Can be pre-defined (e.g., replace a scalar x by polynomial mapping $[1, x, x^2]$) or extracted by a pretrained deep neural net

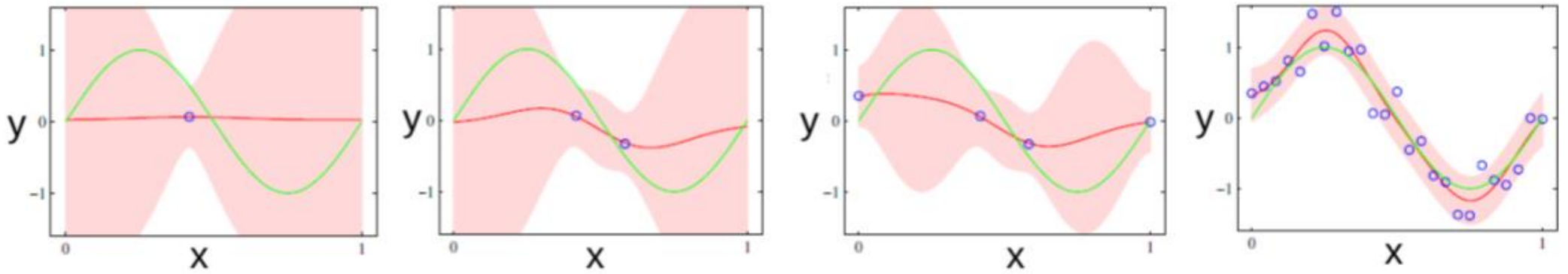
- Alternatively, a kernel function can be used to implicitly define the nonlinear mapping
- More on nonlinear regression when we discuss Gaussian Processes

More on Visualization of Uncertainty

- Figures below: Green curve is the true function and blue circles are observations
- Posterior of the nonlinear regression model: Some curves drawn from the posterior

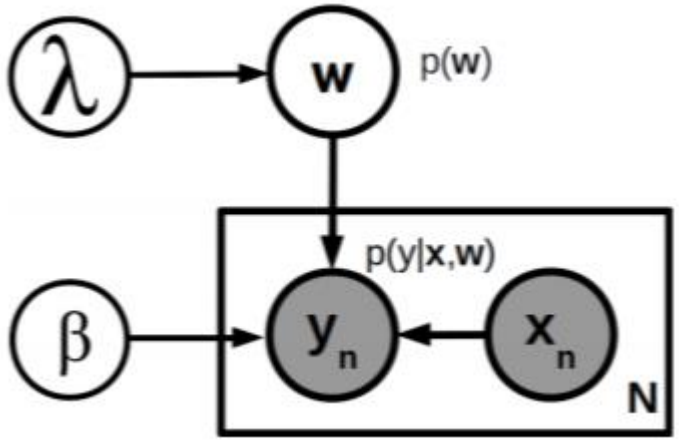


- PPD: Red curve is predictive mean, shaded region denotes predictive uncertainty



Estimating Hyperparameters via MLE-II

- The probabilistic linear reg. model we saw had two hyperparams (β, λ)
  - Thus total three unknowns $(\boldsymbol{w}, \beta, \lambda)$



Need posterior over all the 3 unknowns

$$p(\boldsymbol{w}, \beta, \lambda | \mathbf{X}, \boldsymbol{y}) = \frac{p(\boldsymbol{y}|\mathbf{X}, \boldsymbol{w}, \beta, \lambda) p(\boldsymbol{w}, \lambda, \beta)}{p(\boldsymbol{y}|\mathbf{X})}$$

PPD would require integrating out all 3 unknowns

$$= \frac{p(\boldsymbol{y}|\mathbf{X}, \boldsymbol{w}, \beta, \lambda) p(\boldsymbol{w}|\lambda) p(\beta) p(\lambda)}{\int p(\boldsymbol{y}|\mathbf{X}, \boldsymbol{w}, \beta) p(\boldsymbol{w}|\lambda) p(\beta) p(\lambda) \, d\boldsymbol{w} \, d\lambda d\beta}$$

$$p(y_*|\boldsymbol{x}_*, \mathbf{X}, \boldsymbol{y}) = \int p(y_*|\boldsymbol{x}_*, \boldsymbol{w}, \beta) p(\boldsymbol{w}, \beta, \lambda | \mathbf{X}, \boldsymbol{y}) \, d\boldsymbol{w} \, d\beta \, d\lambda$$

- Posterior and PPD computation is intractable.
- If we just want point estimates for (β, λ) then MLE-II is an option

Called "MLE-II" because we are maximizing marginal likelihood, not the likelihood

$$(\hat{\beta}, \hat{\lambda}) = \text{argmax}_{\beta,\lambda} \log p(\boldsymbol{y}|\boldsymbol{X}, \beta, \lambda)$$

And then compute $p(\boldsymbol{w}|\boldsymbol{X}, \boldsymbol{y}, \hat{\beta}, \hat{\lambda})$ treating $\hat{\beta}, \hat{\lambda}$ as given

For regression with Gaussian likelihood and Gaussian prior on $\boldsymbol{w}$, the marginal likelihood has an exact expression

Will see various other methods like EM, variational inference, MCMC, etc later

Prob. Linear Regression: Some Other Variations

- Can use other likelihoods $p(y_n|\boldsymbol{x}_n, \boldsymbol{w})$ and/or prior distribution $p(\boldsymbol{w})$

- Laplace distribution for the likelihood

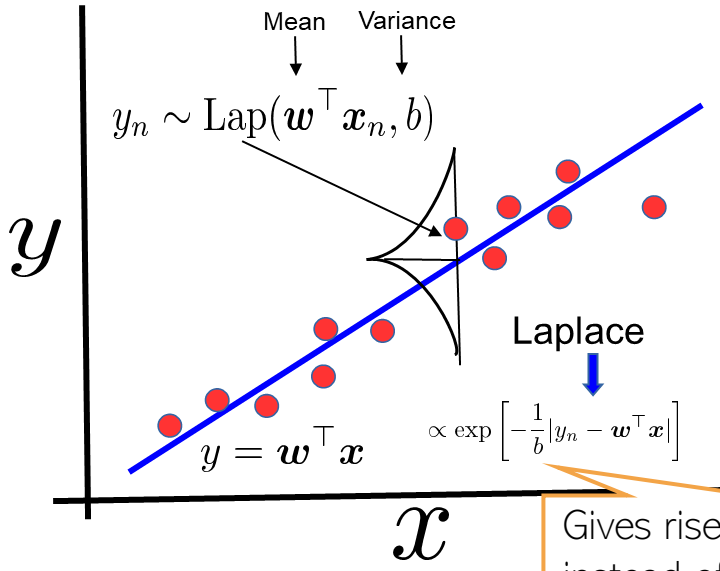
$$p(y_n|\boldsymbol{x}_n, \boldsymbol{w}) = \text{Lap}(y_n|\boldsymbol{w}^\top \boldsymbol{x}_n, b)$$

- Heteroskedastic noise in the likelihood, e.g.,

$$p(y_n|\boldsymbol{x}_n, \boldsymbol{w}) = \mathcal{N}(y_n|\boldsymbol{w}^\top \boldsymbol{x}_n, \beta_n^{-1})$$

Can even assume β_n to depend on input $\boldsymbol{x}_n$

Different noise distribution $\mathcal{N}(0, \beta_n^{-1})$ for each y_n



More robust to outliers than squared loss

Gives rise to "absolute loss" instead of squared loss

- Feature-specific variances in the prior for $\boldsymbol{w}$

$$p(\boldsymbol{w}) = \prod_{d=1}^{D} \mathcal{N}(w_d|0, \lambda_d^{-1}) = \mathcal{N}(\boldsymbol{w}|\boldsymbol{0}, \boldsymbol{\Lambda}^{-1})$$

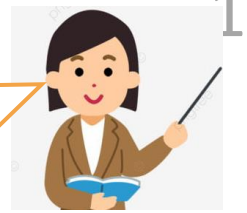
Diagonal precision/covariance matrix with λ_d's along the columns of Λ

This has the effect of having feature-specific regularization

Since we can also learn these precisions (e.g., using MLE-II), using such a prior, we can learn the importance of different features (feature selection) which isn't possible with a $\mathcal{N}(\boldsymbol{w}|\boldsymbol{0}, \lambda^{-1}\mathbf{I})$ prior with spherical covariance

Logistic Regression

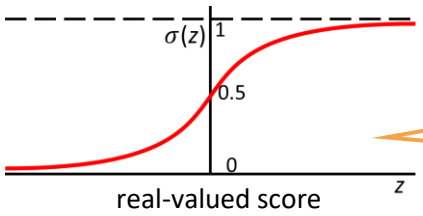
There are other ways too that can convert the score into a probability, such as a CDF: $p(y=1|\boldsymbol{x},\boldsymbol{w}) = \mu = \Phi(\boldsymbol{w}^\top \boldsymbol{x})$ where Φ is the CDF of $\mathcal{N}(0,1)$. This model is known as "Probit Regression".



- A discriminative model for binary classification ($y \in \{0,1\}$)
- A linear model with parameters $\boldsymbol{w} \in \mathbb{R}^D$ computes a score $\boldsymbol{w}^\top \boldsymbol{x}$ for input $\boldsymbol{x}$
- A sigmoid function maps this real-valued score into probability of label being 1

Also used as a nonlinear "activation function" in deep neural networks

$$p(y=1|\boldsymbol{x},\boldsymbol{w}) = \mu = \sigma(\boldsymbol{w}^\top \boldsymbol{x})$$



$$\sigma(z) = \frac{1}{1+\exp(-z)} = \frac{\exp(z)}{1+\exp(z)}$$

Large positive score $\boldsymbol{w}^\top \boldsymbol{x}$ means large prob of label being 1, and large negative score means low prob

- Thus conditional distribution of label $y \in \{0,1\}$ given $\boldsymbol{x}$ is the following Bernoulli

Likelihood

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

- NLL is the binary cross-entropy loss: $-[y_n \log \mu_n + (1-y_n)\log(1-\mu_n)]$
- NLL is convex in $\boldsymbol{w}$. Can also use a prior $p(\boldsymbol{w}|\lambda) = \mathcal{N}(\boldsymbol{w}|\boldsymbol{0}, \lambda^{-1}\boldsymbol{I})$ if interested in MAP or full posterior on $\boldsymbol{w}$

Logistic Regression: MAP and Posterior

- The posterior will be

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

Gaussian

Bernoulli

- MAP estimation is easy. $-\log p(\boldsymbol{w}|\boldsymbol{X},\boldsymbol{y})$ is convex for LR. Unique minima
  - Can use first or second order optimization with gradient and Hessian being

$$\boldsymbol{g} = -\sum_{n=1}^{N}(y_n - \mu_n)\boldsymbol{x}_n + \lambda \mathbf{I}\boldsymbol{w} = \mathbf{X}^\top(\boldsymbol{\mu} - \boldsymbol{y}) + \lambda \boldsymbol{w} \qquad (\text{a } D \times 1 \text{ vector})$$

$$\mathbf{H} = \sum_{n=1}^{N}\mu_n(1-\mu_n)\boldsymbol{x}_n\boldsymbol{x}_n^\top + \lambda \mathbf{I} = \mathbf{X}^\top \mathbf{S}\mathbf{X} + \lambda \mathbf{I} \qquad (\text{a } D \times D \text{ matrix})$$

$$\mu_n = \sigma(\boldsymbol{w}^\top \boldsymbol{x}_n)$$

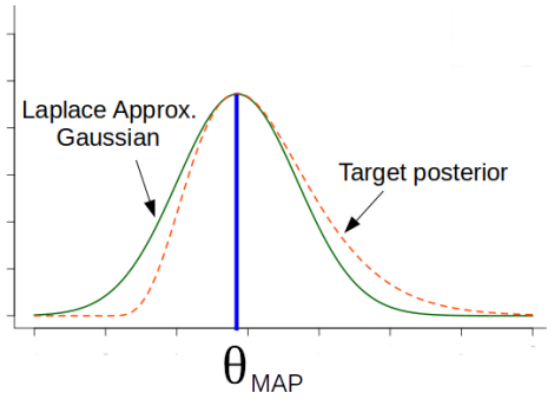
- Full posterior is intractable because of non-conjugacy
  - A popular option is to use the Laplace's approximation (other methods like MCMC and variational inference can also be used; will see them later)

Laplace's (or Gaussian) Approximation

- Consider a posterior distribution that is intractable to compute

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

- Laplace approximation approximates the above using a Gaussian distribution



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

$$\theta_{MAP} = \text{argmax}_\theta \log p(\theta|\mathcal{D})$$

$$\mathbf{\Lambda} = -\nabla^2_\theta \log p(\theta|\mathcal{D})\Big|_{\theta=\theta_{MAP}} = -\nabla^2_\theta \log p(\mathcal{D},\theta)\Big|_{\theta=\theta_{MAP}}$$

Tells us about the space (curvature) of the true posterior around θ_{MAP}

Related to the **Fisher Information Matrix** (FIM); will see shortly

Negative of the Hessian, i.e., the second derivative of the log joint, at θ_{MAP}

- Laplace's approx. is based on a second-order Taylor approx. of the posterior (will see the proof and details later)

LR: Posterior Predictive Distribution

- The posterior predictive distribution can be computed as

$$p(y_* = 1|\boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{y}) = \int p(y_* = 1|\boldsymbol{w}, \boldsymbol{x}_*) p(\boldsymbol{w}|\boldsymbol{X}, \boldsymbol{y}) d\boldsymbol{w}$$

Integral not tractable and must be approximated

sigmoid

Gaussian (if using Laplace approx.)

- Monte-Carlo approximation of this integral is one possible way
  - Draw M samples $\boldsymbol{w}_1, \boldsymbol{w}_2, \ldots, \boldsymbol{w}_M$, from the approx. of posterior
  - Approximate the PPD as follows

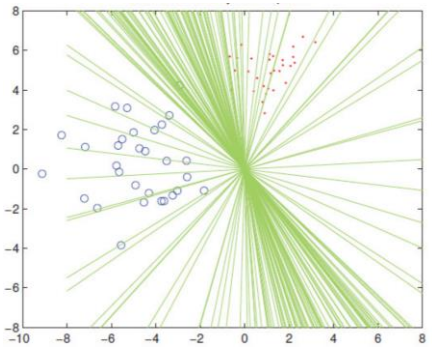
$$p(y_* = 1|\boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{y}) \approx \frac{1}{M} \sum_{m=1}^{M} p(y_* = 1|\boldsymbol{w}_m, \boldsymbol{x}_*) = \frac{1}{M} \sum_{m=1}^{M} \sigma(\boldsymbol{w}_m^\top \boldsymbol{x}_n)$$

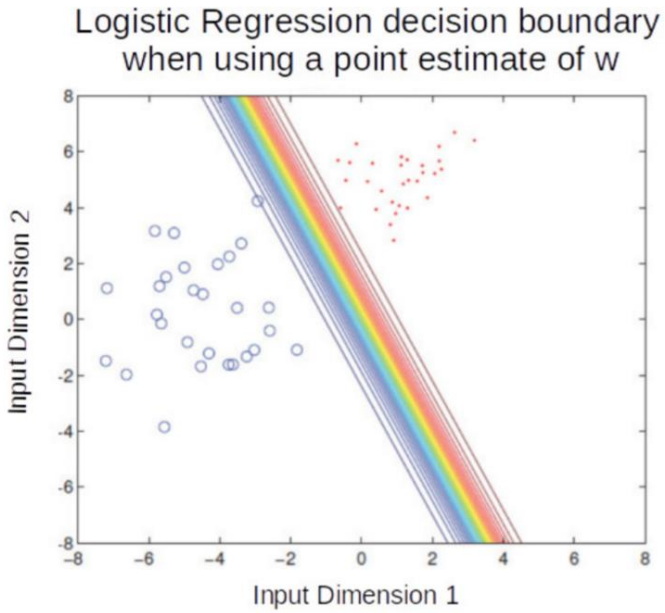
- In contrast, when using MLE/MAP solution $\widehat{\boldsymbol{w}}_{opt}$, the plug-in pred. distribution

$$p(y_* = 1|\boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{y}) = \int p(y_* = 1|\boldsymbol{w}, \boldsymbol{x}_*) p(\boldsymbol{w}|\boldsymbol{X}, \boldsymbol{y}) d\boldsymbol{w}$$
$$\approx p(y_* = 1|\widehat{\boldsymbol{w}}_{opt}, \boldsymbol{x}_*) = \sigma(\widehat{\boldsymbol{w}}_{opt}{}^\top \boldsymbol{x}_n)$$

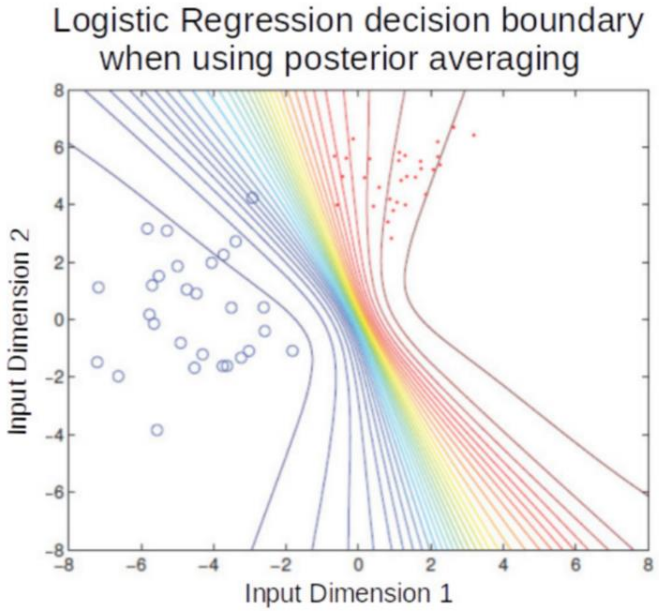
LR: Plug-in Prediction vs Bayesian Averaging

- Plug-in prediction uses a single $\boldsymbol{w}$ (point est) to make prediction
- PPD does an averaging using all possible $\boldsymbol{w}$'s from the posterior





Color transitions (red to blue) in both plots denote how the probability of an input changes from belonging to red class to belonging to blue class. All inputs on a line (or curve on RHS plot) have the same probability of belonging to the red/blue class



Posterior averaging is like using an ensemble of models. In this example, each model is a linear classifier but the ensemble-like effect resulted in nonlinear boundaries

$$p(y_* = 1|\boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{y}) \approx \sigma(\hat{\boldsymbol{w}}_{opt}{}^{\top}\boldsymbol{x}_n)$$

$$p(y_* = 1|\boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{y}) \approx \frac{1}{M}\sum_{m=1}^{M}\sigma(\boldsymbol{w}_m^{\top}\boldsymbol{x}_n)$$

Multiclass Logistic (a.k.a. Softmax) Regression

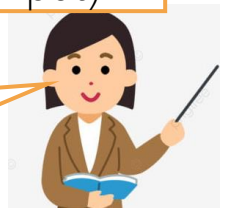
- Also called multinoulli/multinomial regression: Basically, LR for $K > 2$ classes
- In this case, $y_n \in \{1,2,\ldots,K\}$ and label probabilities are defined as

$$p(y_n = k|\boldsymbol{x}_n, \boldsymbol{W}) = \frac{\exp(\boldsymbol{w}_k^\top \boldsymbol{x}_n)}{\sum_{\ell=1}^{K} \exp(\boldsymbol{w}_\ell^\top \boldsymbol{x}_n)} = \mu_{nk}$$

Softmax function

Real-valued scores $\boldsymbol{w}_k^\top \boldsymbol{x}_n$ are also known as "logits" (thus K logits for each input)

Also note that $\sum_{\ell=1}^{K} \mu_{n\ell} = 1$ for any input $\boldsymbol{x}_n$

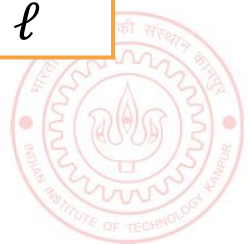


- K weight vecs $\boldsymbol{w}_1, \boldsymbol{w}_2, \ldots, \boldsymbol{w}_K$ (one per class), each D-dim, and $\boldsymbol{W} = [\boldsymbol{w}_1, \boldsymbol{w}_2, \ldots, \boldsymbol{w}_K]$
- Each likelihood $p(y_n|\boldsymbol{x}_n, \boldsymbol{W})$ is a multinoulli distribution. Therefore total likelihood

$$p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{W}) = \prod_{n=1}^{N} \prod_{\ell=1}^{K} \mu_{n\ell}^{y_{n\ell}}$$

Notation: $y_{n\ell} = 1$ if true class of $\boldsymbol{x}_n$ is ℓ and $y_{n\ell'} = 0 \; \forall \, \ell' \neq \ell$

- Can do MLE/MAP/fully Bayesian estimation for $\boldsymbol{W}$ similar to LR model



Generalized Linear Models

- (Probabilistic) Linear Regression: when response $\boldsymbol{y}$ is real-valued

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

- Logistic Regression: when response $\boldsymbol{y}$ is binary (0/1)

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

- Both are examples of a Generalized Linear Model (GLM)
  - The model depends on the inputs $\boldsymbol{x}$ via a linear model $\boldsymbol{w}^\top \boldsymbol{x}$
- GLM is defined using an exponential family distribution

$$p(y|\boldsymbol{x}, \boldsymbol{w}) = \text{ExpFam}[y|f(\boldsymbol{w}^\top \boldsymbol{x})]$$

MLE/MAP of $\boldsymbol{w}$ is easy for GLMs (due to convex objective, thanks to exp-family). Posterior usually requires approximations if likelihood and prior are not conjugate pairs (Laplace approximation or other methods used)

- ExpFam can be any suitable distribution depending on the nature of outputs, e.g.,
  - Gaussian for reals, Bernoulli for binary, Poisson for Count, gamma for positive reals
- ExpFam distributions are more generally useful in other contexts as well