

# Probabilistic Supervised Learning (contd)

CS772A: Probabilistic Machine Learning

Piyush Rai

# Plan Today

- Probabilistic linear regression (contd)
- Logistic and Softmax/Multinoulli Regression
- Generalized Linear Models



# Probabilistic Linear Regression

A discriminative model for regression problems

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

Unknown to be estimated

Each weight assumed real-valued

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

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

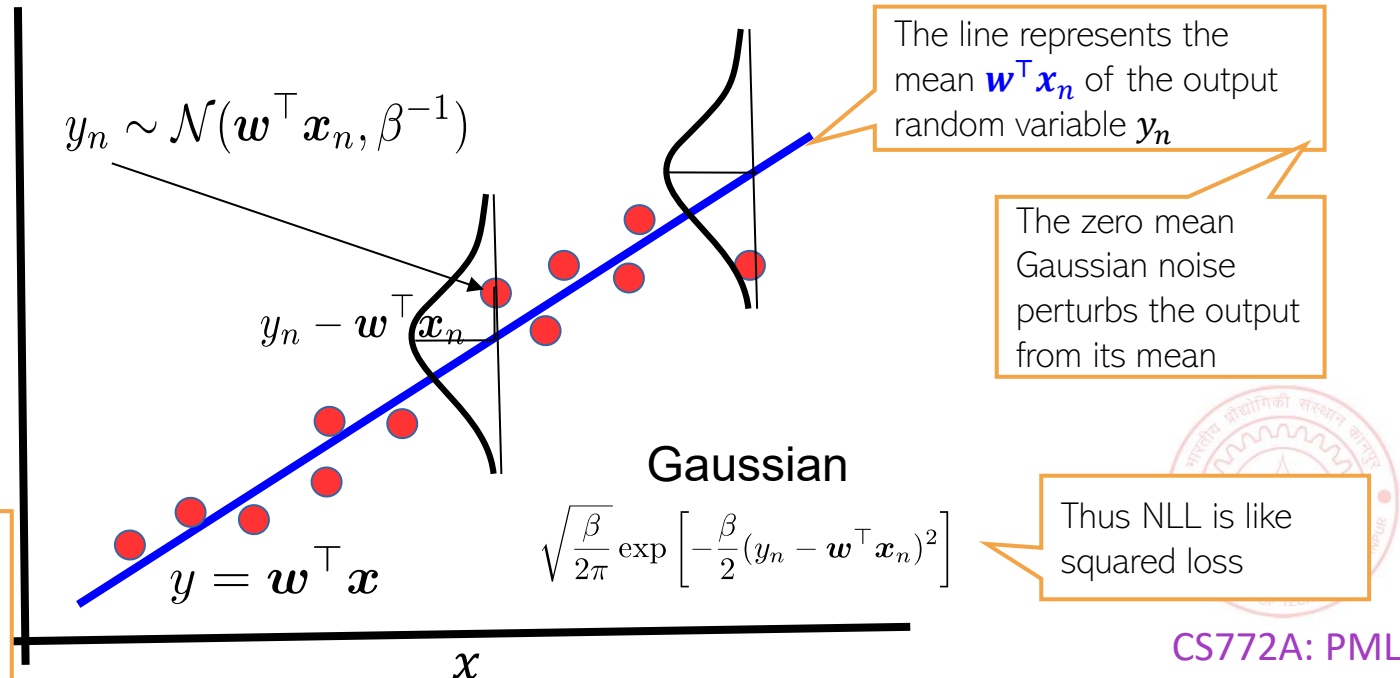
- Notation alert:  $\beta$  is the precision of Gaussian noise (and  $\beta^{-1}$  the variance)

## Likelihood model

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

Input  $\mathbf{x}_n$  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



# Prior on weights

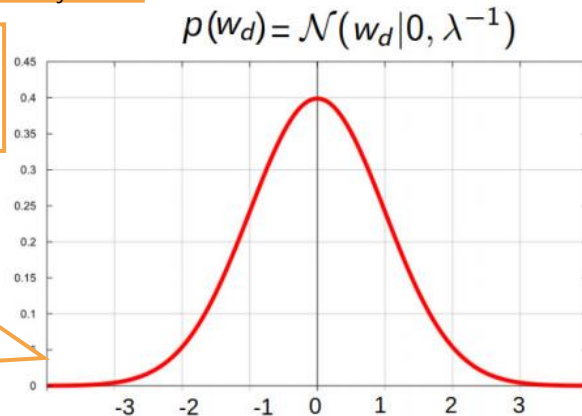
- Assume a **zero-mean Gaussian prior** on  $\mathbf{w}$

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

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

Large  $\lambda$  means more aggressive push towards zero

The precision  $\lambda$  controls how aggressively the prior pushes  $w_d$  towards mean (0)



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

$\lambda$  controls the uncertainty around our prior belief about value of  $w_d$

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

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

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

- Zero-mean Gaussian prior corresponds to  $\ell_2$  regularizer



# The Posterior

MLE/MAP left  
as an exercise



- The posterior over  $\mathbf{w}$  (for now, assume hyperparams  $\beta$  and  $\lambda$  to be known)

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

Must be a Gaussian  
due to conjugacy

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

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

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

$$p(\mathbf{w}|\mathbf{y}, \mathbf{X}, \beta, \lambda) = \mathcal{N}(\mu_N, \Sigma_N)$$

Note that  $\lambda$  and  $\beta$  can be  
learned under the  
probabilistic set-up (though  
assumed fixed as of now)

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

The form is also similar to the solution to **ridge regression**  
 $\arg\min_{\mathbf{w}} \|\mathbf{y} - \mathbf{X}\mathbf{w}\|^2 + \lambda \mathbf{w}^\top \mathbf{w} = (\mathbf{X}^\top \mathbf{X} + \lambda \mathbf{I})^{-1} \mathbf{X}^\top \mathbf{y}$

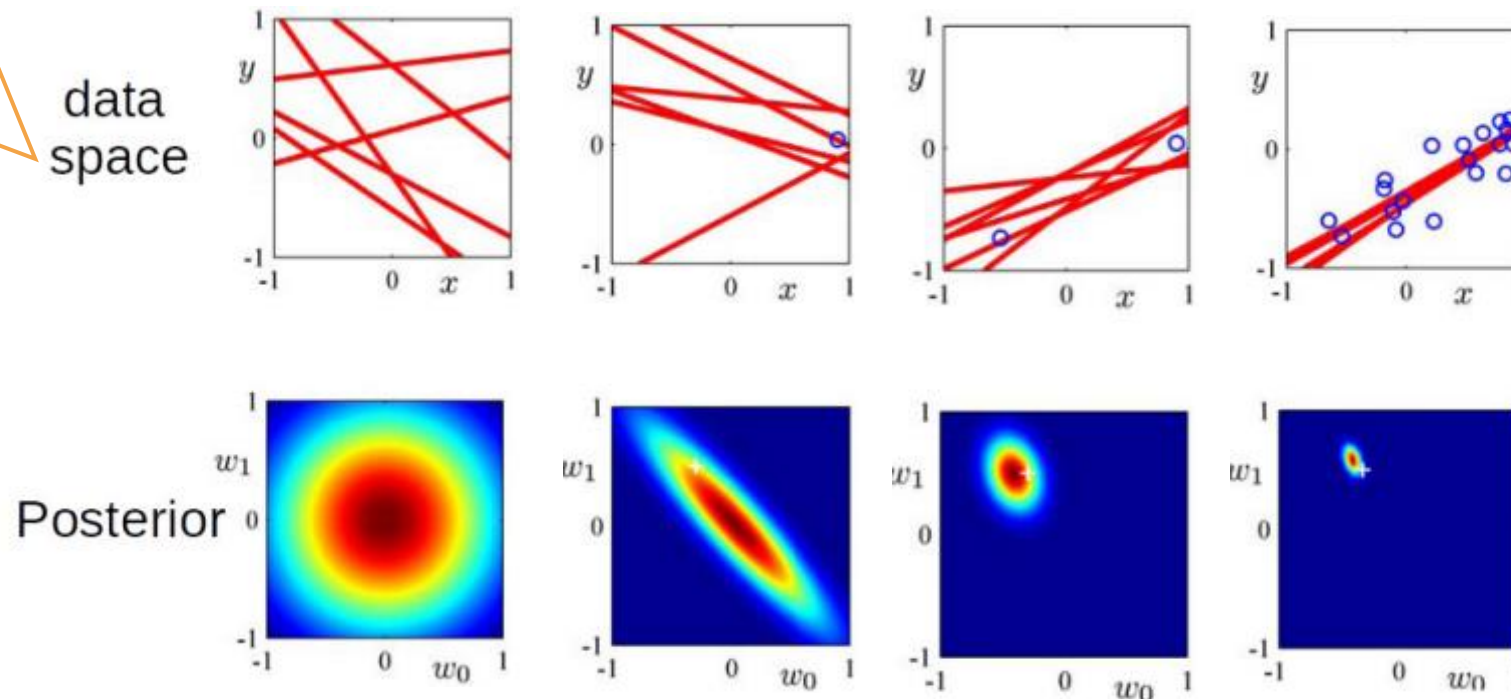
**MAP solution** turns out to be exactly  
the same (reason: Gaussian's mean  
and mode are the same)

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

# The Posterior: A Visualization

- Assume a lin. reg. problem with true  $\mathbf{w} = [w_0, w_1]$ ,  $w_0 = -0.3, w_1 = 0.5$
- Assume data generated by a linear regression model  $y = w_0 + w_1x + \text{"noise"}$ 
  - Note: It's actually 1-D regression ( $w_0$  is just a bias term), or 2-D reg. with feature  $[1, x]$
- Figures below show the “data space” and posterior of  $\mathbf{w}$  for different number of observations (note: with no observations, the posterior = prior)

Each red line represents the “data” generated for a randomly drawn  $\mathbf{w}$  from the current posterior



# Posterior Predictive Distribution

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

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

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

$\mathcal{N}(\mathbf{y}_* | \mathbf{w}^\top \mathbf{x}_*, \beta^{-1})$

$\mathcal{N}(\mathbf{w} | \boldsymbol{\mu}_N, \boldsymbol{\Sigma}_N)$

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

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

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

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

$$\begin{aligned} p(\mathbf{y}_* | \mathbf{x}_*, \mathbf{w}_{MLE}) &= \mathcal{N}(\mathbf{w}_{MLE}^\top \mathbf{x}_*, \beta^{-1}) && \text{- MLE prediction} \\ p(\mathbf{y}_* | \mathbf{x}_*, \mathbf{w}_{MAP}) &= \mathcal{N}(\mathbf{w}_{MAP}^\top \mathbf{x}_*, \beta^{-1}) && \text{- MAP prediction} \end{aligned}$$

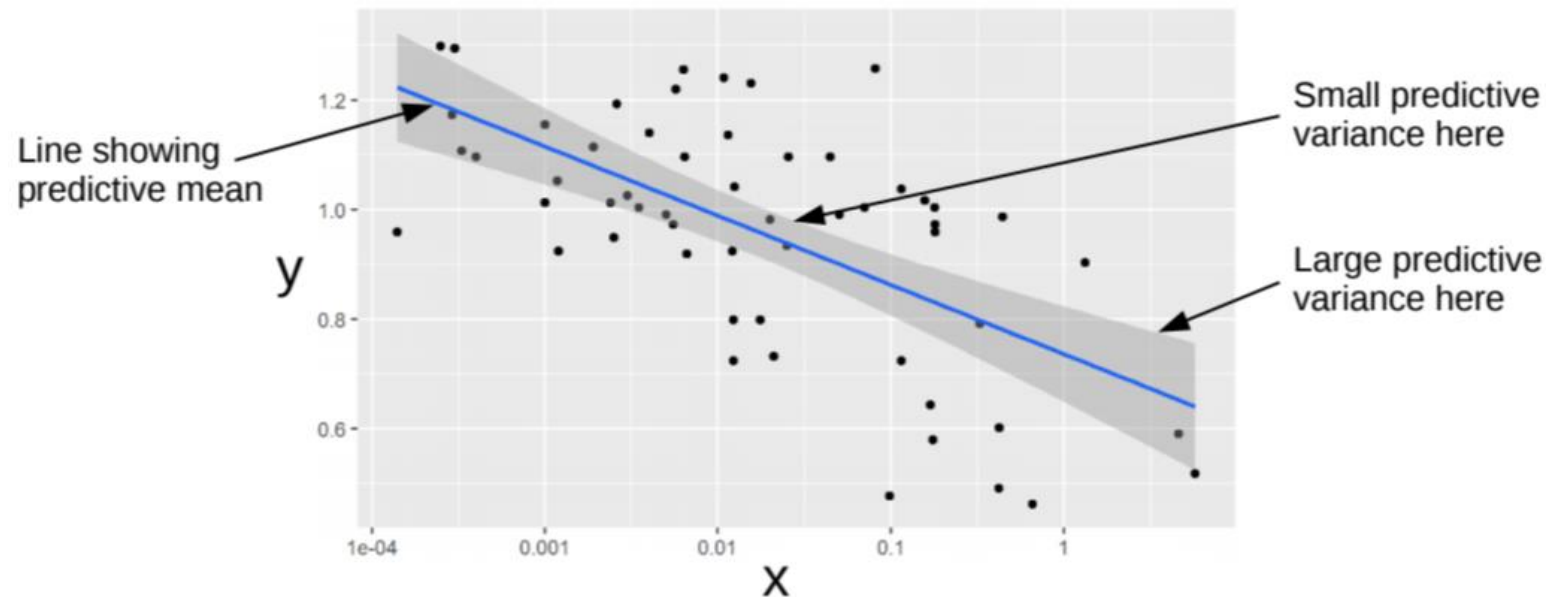
Since PPD also takes into account the uncertainty in  $\mathbf{w}$ , the predictive variance is larger

- Unlike MLE/MAP, variance of  $\mathbf{y}_*$  also depends on the input  $\mathbf{x}_*$  (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

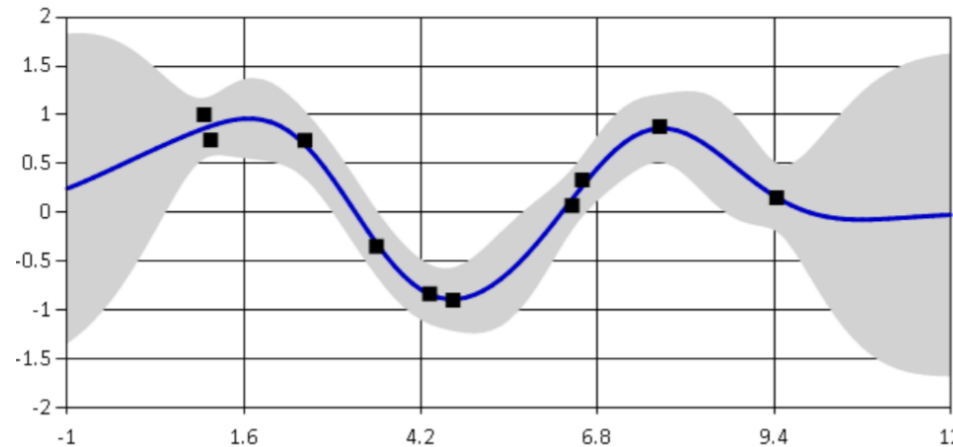


- Width of the shaded region at any  $x$  denotes the predictive uncertainty at that  $x$  ( $\pm$  one std-dev)
- Regions with more training examples have smaller predictive variance





# Nonlinear Regression



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

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

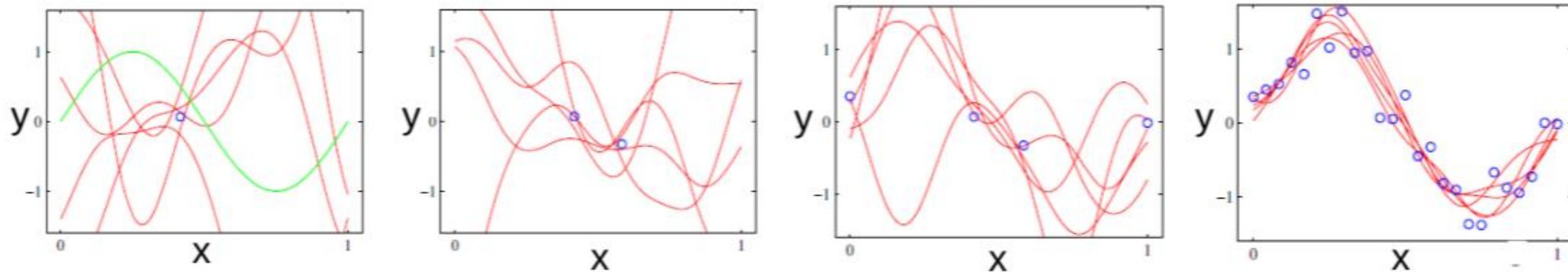
Can be pre-defined (e.g., replace a scalar  $x$  by polynomial mapping  $[1, x, x^2]$ ) or extracted by a pretrained deep neural net

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

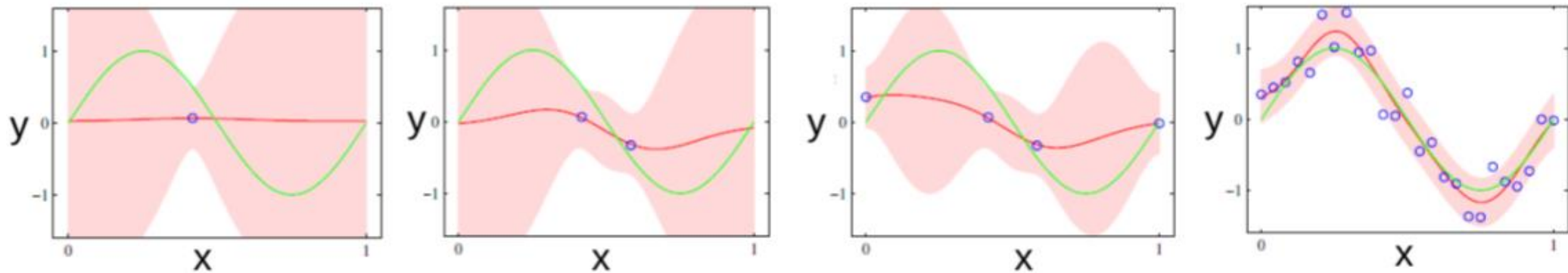


# More on Visualization of Uncertainty

- 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

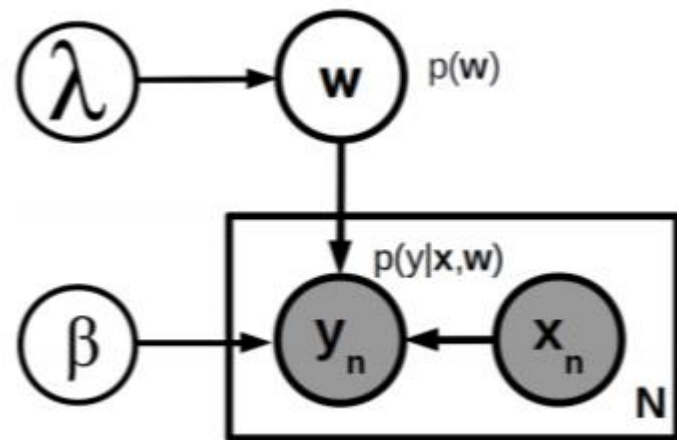


- PPD: Red curve is predictive mean, shaded region denotes predictive uncertainty



# Estimating Hyperparameters via MLE-II

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



Need posterior over all the 3 unknowns

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

PPD would require integrating out all 3 unknowns

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

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

- Posterior and PPD computation is intractable.

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

- If we just want point estimates for  $(\beta, \lambda)$  then MLE-II is an option

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

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

For regression with Gaussian likelihood and Gaussian prior on  $\mathbf{w}$ , 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  $p(y_n | \mathbf{x}_n, \mathbf{w})$  and/or prior distribution  $p(\mathbf{w})$

- Laplace distribution for the likelihood

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

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

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

Can even assume  $\beta_n$  to depend on input  $\mathbf{x}_n$

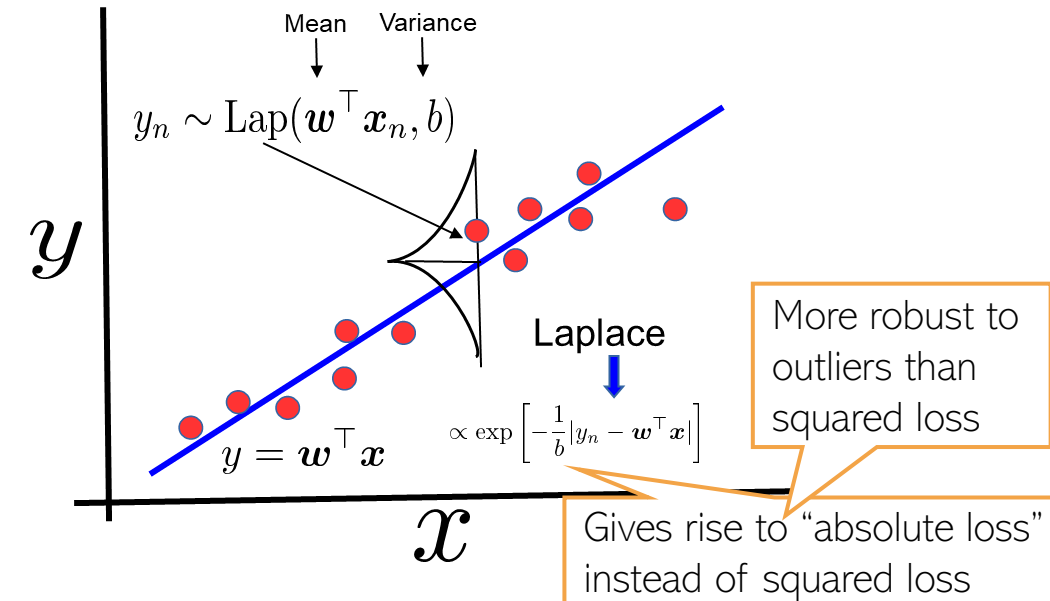
Different noise distribution  $\mathcal{N}(0, \beta_n^{-1})$  for each  $y_n$

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

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

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  $\mathcal{N}(\mathbf{w} | \mathbf{0}, \lambda^{-1} \mathbf{I})$  prior with spherical covariance



Diagonal precision/covariance matrix with  $\lambda_d$ 's along the columns of  $\Lambda$



# Logistic Regression

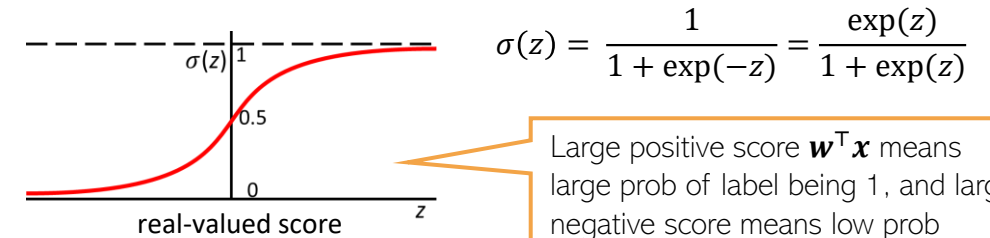
There are other ways too that can convert the score into a probability, such as a CDF:  $p(y = 1|\mathbf{x}, \mathbf{w}) = \mu = \Phi(\mathbf{w}^\top \mathbf{x})$  where  $\Phi$  is the CDF of  $\mathcal{N}(0,1)$ . This model is known as "Probit Regression".



- A discriminative model for binary classification ( $y \in \{0,1\}$ )
- A linear model with parameters  $\mathbf{w} \in \mathbb{R}^D$  computes a **score**  $\mathbf{w}^\top \mathbf{x}$  for input  $\mathbf{x}$
- 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

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



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

Likelihood

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

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



# Logistic Regression: MAP and Posterior

14

- The posterior will be

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

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

$$\begin{aligned}\mathbf{g} &= -\sum_{n=1}^N (y_n - \mu_n) \mathbf{x}_n + \lambda \mathbf{w} = \mathbf{X}^\top (\boldsymbol{\mu} - \mathbf{y}) + \lambda \mathbf{w} \quad (\text{a } D \times 1 \text{ vector}) \\ \mathbf{H} &= \sum_{n=1}^N \mu_n (1 - \mu_n) \mathbf{x}_n \mathbf{x}_n^\top + \lambda \mathbf{I} = \mathbf{X}^\top \mathbf{S} \mathbf{X} + \lambda \mathbf{I} \quad (\text{a } D \times D \text{ matrix}) \\ \mu_n &= \sigma(\mathbf{w}^\top \mathbf{x}_n)\end{aligned}$$

- 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)



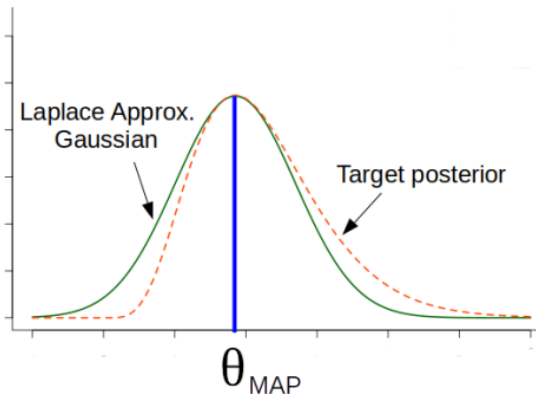


# Laplace's (or Gaussian) Approximation

- 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



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

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

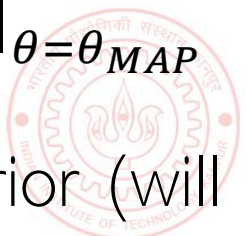
$$\Lambda = -\nabla_{\theta}^2 \log p(\theta|\mathcal{D}) \Big|_{\theta=\theta_{MAP}} = -\nabla_{\theta}^2 \log p(\mathcal{D}, \theta) \Big|_{\theta=\theta_{MAP}}$$

Tells us about the space (curvature) of the true posterior around  $\theta_{MAP}$

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

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

- 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

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

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  $\mathbf{w}_1, \mathbf{w}_2, \dots, \mathbf{w}_M$ , from the approx. of posterior
  - Approximate the PPD as follows

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

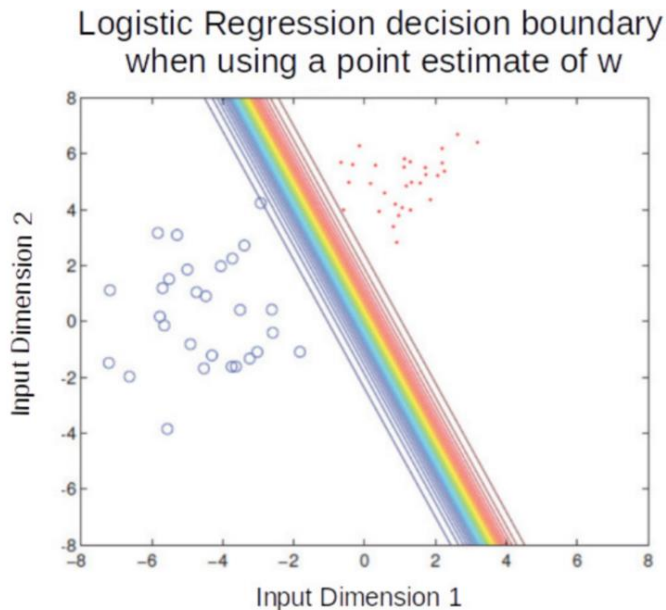
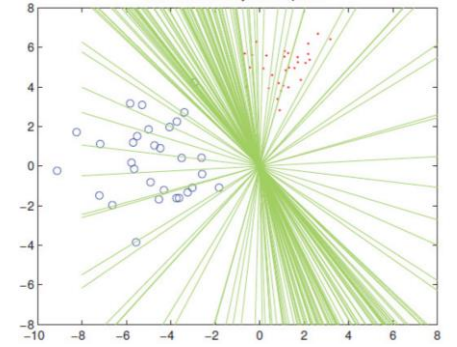
- In contrast, when using MLE/MAP solution  $\hat{\mathbf{w}}_{opt}$ , the plug-in pred. distribution

$$\begin{aligned} p(y_* = 1 | \mathbf{x}_*, \mathbf{X}, \mathbf{y}) &= \int p(y_* = 1 | \mathbf{w}, \mathbf{x}_*) p(\mathbf{w} | \mathbf{X}, \mathbf{y}) d\mathbf{w} \\ &\approx p(y_* = 1 | \hat{\mathbf{w}}_{opt}, \mathbf{x}_*) = \sigma(\hat{\mathbf{w}}_{opt}^\top \mathbf{x}_*) \end{aligned}$$

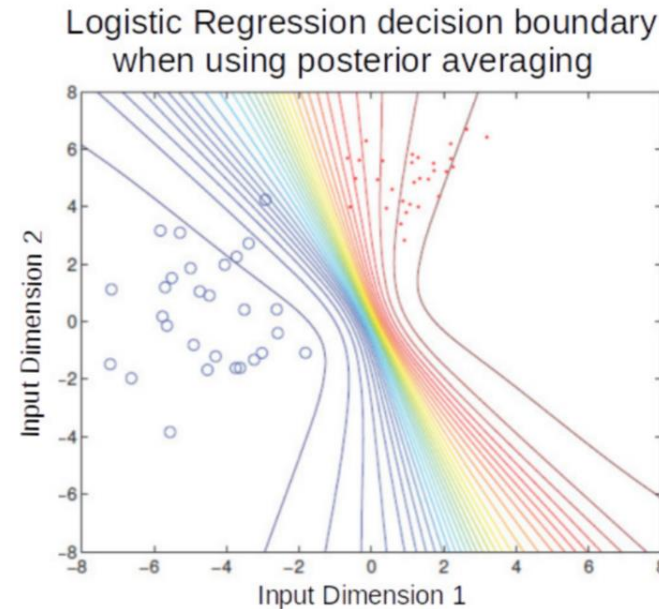


# LR: Plug-in Prediction vs Bayesian Averaging

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

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

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



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

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

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

Softmax function

Real-valued scores  $\mathbf{w}_k^\top \mathbf{x}_n$  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  $\mathbf{x}_n$



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

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

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

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



# Generalized Linear Models

- (Probabilistic) Linear Regression: when response  $y$  is real-valued

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

- Logistic Regression: when response  $y$  is binary (0/1)

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

- Both are examples of a Generalized Linear Model (GLM)

- The model depends on the inputs  $\mathbf{x}$  via a linear model  $\mathbf{w}^\top \mathbf{x}$

- GLM is defined using an exponential family distribution

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

MLE/MAP of  $\mathbf{w}$  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

