Probabilistic Linear Regression

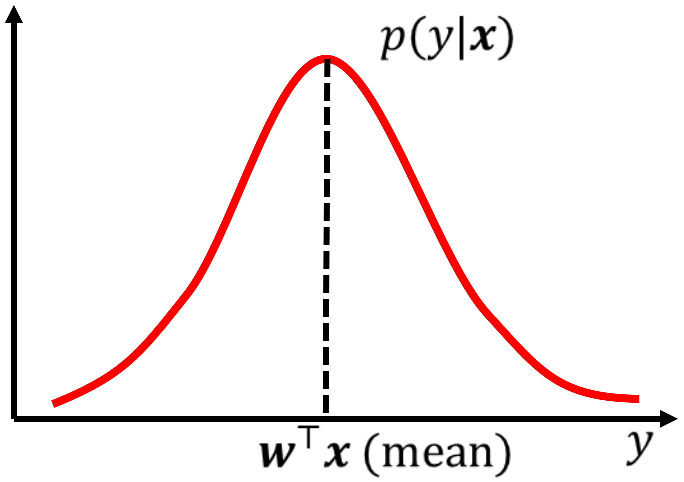
CS771: Introduction to Machine Learning

Nisheeth

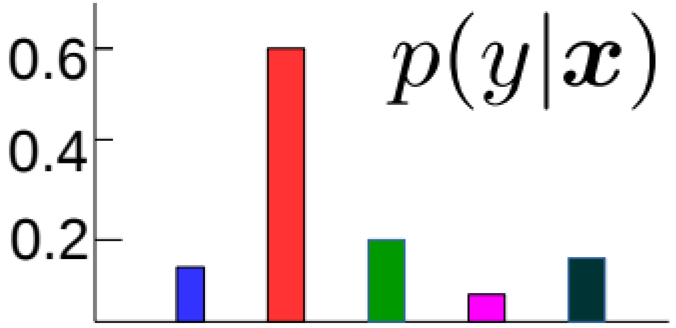
Probabilistic Models for Supervised Learning

- Goal: Learn the conditional distribution of output given input, i.e., $p(y|\boldsymbol{x})$

Probabilistic Linear Regression



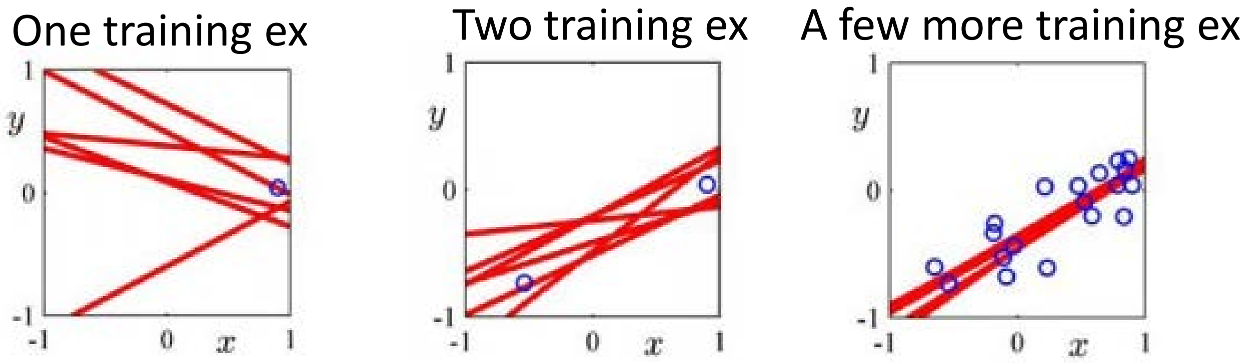
Probabilistic Classification



- $p(y|\boldsymbol{x})$ is more informative than a single prediction y
  - From $p(y|\boldsymbol{x})$, can get "expected" or "most likely" output y
  - For classifn, "soft" predictions (e.g., rather than yes/no, prob. of "yes")
  - "Uncertainty" in the predicted output y (e.g., by looking at the variance of $p(y|\boldsymbol{x})$)
- Can also learn a distribution over the model params using fully Bayesian inference

Distribution over model parameters

- Recall that linear/ridge regression gave a single "optimal" weight vector
- With a probabilistic model for linear regression, we have two options
  - Use MLE/MAP to get a single "optimal" weight vector
  - Use fully Bayesian inference to learn a distribution over weight vectors (figure below)



Rather than returning just a single "best" solution (a line in this example), the fully Bayesian approach would give us several "probable" lines (consistent with training data) by learning the full posterior distribution over the model parameters (each of which corresponds to a line)

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

Posterior predictive distribution by doing posterior weighted averaging over all possible $\boldsymbol{w}$, not just the most likely one. Thus more robust predictions especially if we are uncertain about the best solution.

Predictive distribution using a single $\boldsymbol{w}$ (plug-in predictive distribution)

How important/like this $\boldsymbol{w}$ is under the posterior distribution (its posterior probability)

Probabilistic Models for Supervised Learning

- Usually two ways to model the conditional distribution $p(y|\boldsymbol{x})$
- Approach 1: Don't model $\boldsymbol{x}$, and model $p(y|\boldsymbol{x})$ directly using a prob. distribution

"discriminative" sup learning

Gaussian distribution

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

Probabilistic linear regression

The "sigmoid" function

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

Probabilistic linear binary classification

We assume the conditional distribution to be some appropriate distribution and treat the weights $\boldsymbol{w}$ as learnable parameters of the model (using MLE/MAP/fully Bayesian inference). Need not be a linear model – can replace $\boldsymbol{w}^\top \boldsymbol{x}$ by a nonlinear function $f(\boldsymbol{x})$

- Approach 2: Model both $\boldsymbol{x}$ and $\boldsymbol{y}$ via their joint distr. and get the conditional as

"generative" sup learning

Called "generative" because we are learning the generative distributions for output as well as inputs

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

Here θ denotes all the model parameters that we need to model the joint distribution of $\boldsymbol{x}$ and $\boldsymbol{y}$ (will see examples later)

Prob. distribution of inputs from class $\boldsymbol{k}$

$$p(y=k|\boldsymbol{x},\theta) = \frac{p(\boldsymbol{x},y=k|\theta)}{p(\boldsymbol{x}|\theta)} = \frac{p(\boldsymbol{x}|y=k,\theta)p(y=k|\theta)}{\sum_{\ell=1}^{K} p(\boldsymbol{x}|y=\ell,\theta)p(y=\ell|\theta)}$$

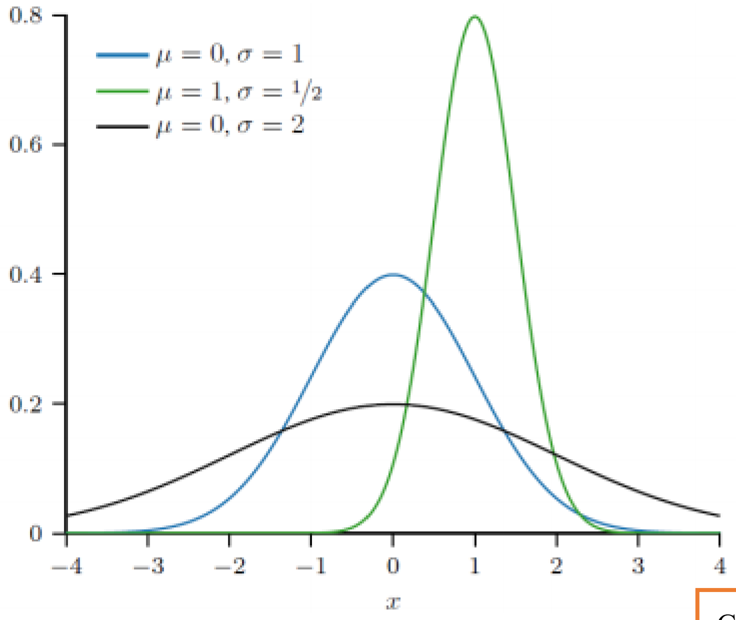
For a multi-class classification model with K classes

Brief Detour (Gaussian Distribution)

Gaussian Distribution (Univariate)

- Distribution over real-valued scalar random variables $x \in \mathbb{R}$
- Defined by a scalar mean μ and a scalar variance σ^2

$$\mathcal{N}(x|\mu,\sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left[-\frac{(x-\mu)^2}{2\sigma^2}\right]$$



- Mean: $\mathbb{E}[x] = \mu$
- Variance: $\text{var}[x] = \sigma^2$
- Inverse of variance is called precision: $\beta = \frac{1}{\sigma^2}$.

Gaussian PDF in terms of precision

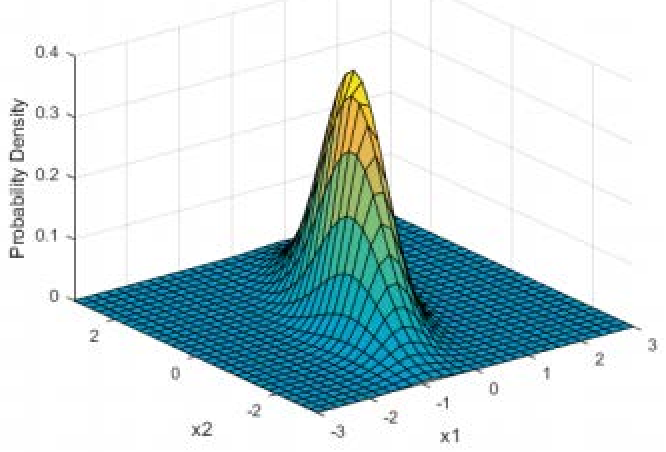
$$\mathcal{N}(x|\mu,\beta) = \sqrt{\frac{\beta}{2\pi}} \exp\left[-\frac{\beta}{2}(x-\mu)^2\right]$$

Gaussian Distribution (Multivariate)

- Distribution over real-valued vector random variables $\boldsymbol{x} \in \mathbb{R}^D$
- Defined by a mean vector $\boldsymbol{\mu} \in \mathbb{R}^D$ and a covariance matrix $\boldsymbol{\Sigma}$

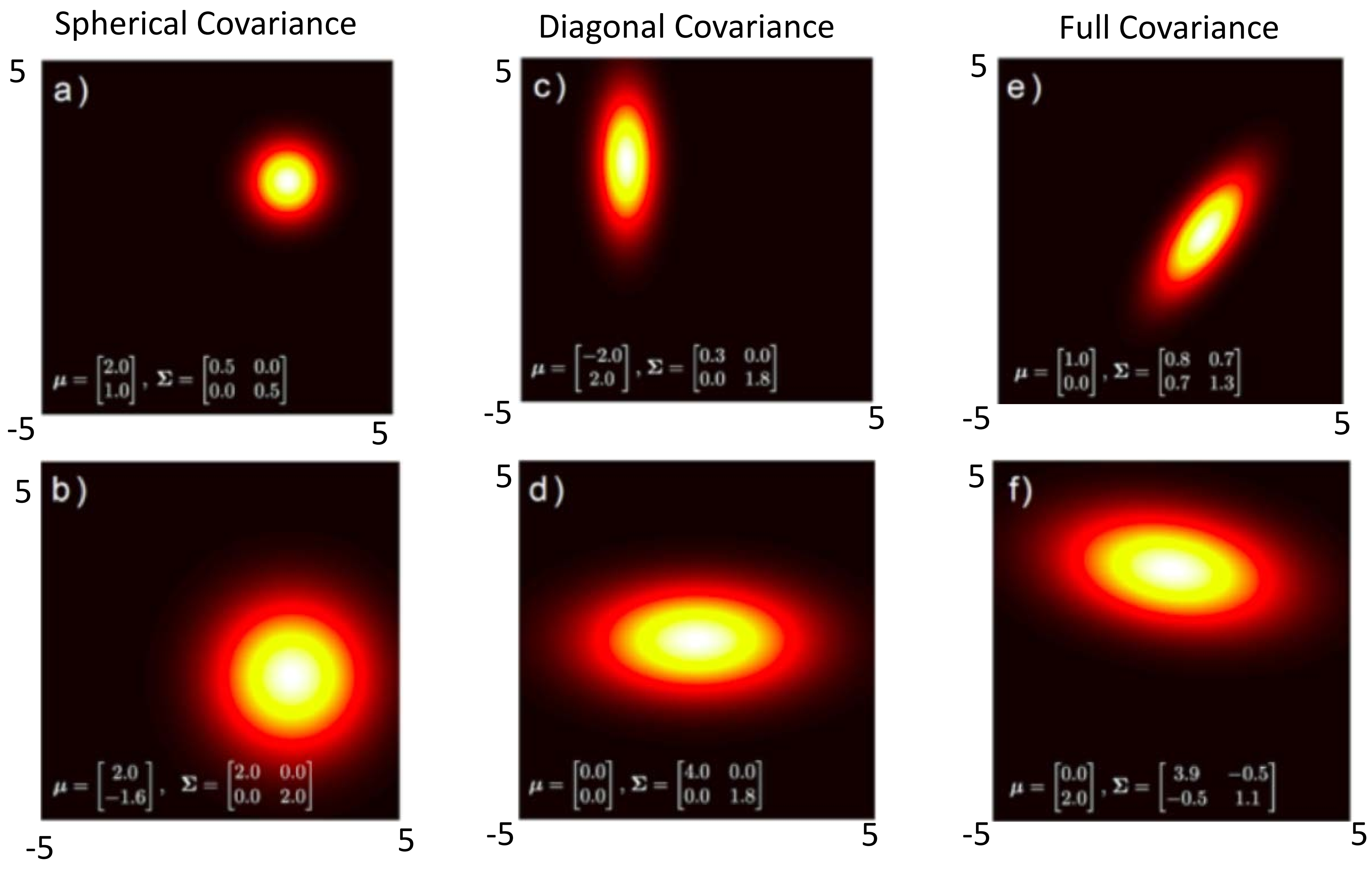
$$\mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu},\boldsymbol{\Sigma}) = \frac{1}{\sqrt{(2\pi)^D|\boldsymbol{\Sigma}|}}\exp[-(\boldsymbol{x}-\boldsymbol{\mu})^\top\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})]$$

A two-dimensional Gaussian



- Note: The cov. matrix $\boldsymbol{\Sigma}$ must be symmetric and PSD
  - All eigenvalues are positive
  - $\mathbf{z}^\top\boldsymbol{\Sigma}\mathbf{z} \geq 0$ for any real vector $\boldsymbol{z}$
- The covariance matrix also controls the shape of the Gaussian

Covariance Matrix for Multivariate Gaussian



Probabilistic Linear Regression

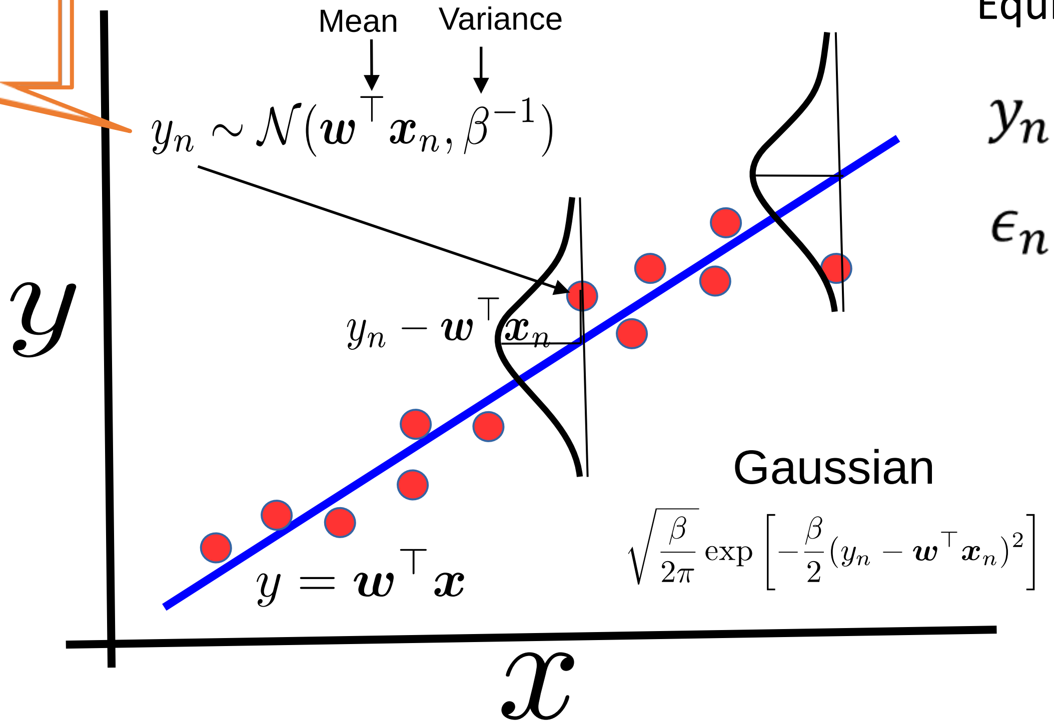
$$p(y|\boldsymbol{x}, \boldsymbol{w}) = \mathcal{N}(y|\boldsymbol{w}^\top \boldsymbol{x}, \beta^{-1})$$

Nice tutorial

Linear Regression: A Probabilistic View

Defines our likelihood model: $p(y_n|\boldsymbol{w}, \boldsymbol{x}_n)$ - Gaussian

Output y_n assumed generated from a Gaussian with mean $\boldsymbol{w}^\top \boldsymbol{x}_n$



Output y_n generated from a linear model and then zero mean Gaussian noise added

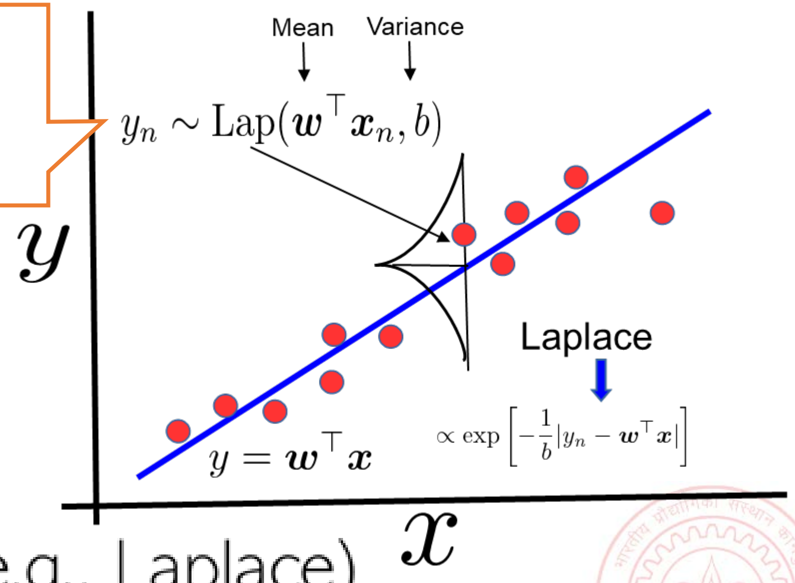
Equivalently:

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

$$\epsilon_n \sim \mathcal{N}(0, \beta^{-1})$$

Note the term in the Gaussian's exponent – just like a squared error we saw for least squares regression

Using a Laplace distribution would correspond to using an absolute loss



- Several variants of this basic model are possible
  - Other distributions to model the additive noise (e.g., Laplace)
  - Different noise variance/precision for each output: $y_n \sim \mathcal{N}(\boldsymbol{w}^\top \boldsymbol{x}_n, \beta_n^{-1})$

MLE for Probabilistic Linear Regression

- Since each likelihood term is a Gaussian, we have

Also note that $\boldsymbol{x}_n$ is fixed here but the likelihood depend on it, so it is being conditioned on

$$p(y_n|\boldsymbol{w},\boldsymbol{x}_n) = \mathcal{N}(y_n|\boldsymbol{w}^\top\boldsymbol{x}_n,\beta^{-1}) = \sqrt{\frac{\beta}{2\pi}}\exp\left[-\frac{\beta}{2}(y_n-\boldsymbol{w}^\top\boldsymbol{x}_n)^2\right]$$

- Thus the overall likelihood (assuming i.i.d. responses) will be

Exercise: Verify that you can also write the overall likelihood as a single $\boldsymbol{N}$ dimensional Gaussian with mean $\boldsymbol{Xw}$ and cov. matrix $\beta^{-1}\boldsymbol{I}_N$

$$p(\boldsymbol{y}|\mathbf{X},\boldsymbol{w}) = \prod_{n=1}^{N} p(y_n|\boldsymbol{x}_n,\boldsymbol{w}) = \left(\frac{\beta}{2\pi}\right)^{N/2}\exp\left[-\frac{\beta}{2}\sum_{n=1}^{N}(y_n-\boldsymbol{w}^\top\boldsymbol{x}_n)^2\right]$$

- Log-likelihood (ignoring constants w.r.t. $\boldsymbol{w}$)

$$\log p(\boldsymbol{y}|\mathbf{X},\boldsymbol{w}) \propto -\frac{\beta}{2}\sum_{n=1}^{N}(y_n-\boldsymbol{w}^\top\boldsymbol{x}_n)^2$$

MLE for probabilistic linear regression with Gaussian noise is equivalent to least squares regression without any regularization (with solution $\hat{\boldsymbol{w}}_{MLE} = (\boldsymbol{X}^\top\boldsymbol{X})^{-1}\boldsymbol{X}^\top\boldsymbol{y}$

- Negative log likelihood (NLL) in this case is similar to squared loss function

MAP Estimation for Prob. Lin. Reg.: The Prior

- For MAP estimation, we need a prior distribution over the parameters $\boldsymbol{w} \in \mathbb{R}^D$
- A reasonable prior for real-valued vectors can be a multivariate Gaussian

$$p(\boldsymbol{w}) = \mathcal{N}(\boldsymbol{w}|\boldsymbol{w}_0, \boldsymbol{\Sigma})$$

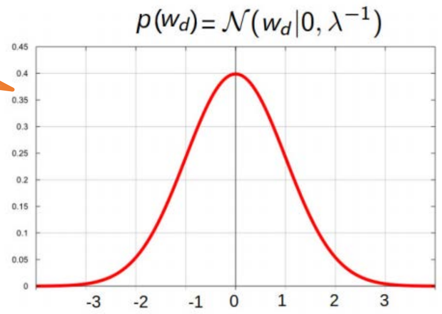
Equivalent to saying that *a priori* we expect the solution to be close to some vector $\boldsymbol{w}_0$ (subject to Σ being such that the variances is not too large

- A specific example of a multivariate Gaussian prior in this problem

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

Omitting λ for brevity

The precision λ of the Gaussian prior controls how aggressively the prior pushes the elements towards mean (0)



This is essentially like a regularizer that pushes elements of $\boldsymbol{w}$ to be small (we will see shortly)

Equivalent to saying that *a priori* we expect each element of the solution to be close to 0 (i.e., "small")

$$\mathcal{N}(w_d|0, \lambda^{-1}) = \sqrt{\frac{\lambda}{2\pi}\exp\left[-\frac{\lambda}{2}w_d^2\right]}$$

$$\mathcal{N}(\boldsymbol{w}|\boldsymbol{0}, \lambda^{-1}\boldsymbol{I}_D) = \left(\frac{\lambda}{2\pi}\right)^{D/2} \exp\left[-\frac{\lambda}{2}\sum_{d=1}^{D} w_d^2\right] = \left(\frac{\lambda}{2\pi}\right)^{D/2} \exp\left[-\frac{\lambda}{2}\boldsymbol{w}^\top \boldsymbol{w}\right]$$

MAP Estimation for Probabilistic Linear Regression

- The MAP objective (log-posterior) will be the log-likelihood + $\log p(\boldsymbol{w})$

$$-\frac{\beta}{2}\sum_{n=1}^{N}(y_n - \boldsymbol{w}^\top \boldsymbol{x}_n)^2 - \frac{\lambda}{2}\boldsymbol{w}^\top \boldsymbol{w}$$

In the likelihood and prior, ignored terms that don't depend on $\boldsymbol{w}$

- Maximizing this is equivalent to minimizing the following w.r.t. $\boldsymbol{w}$

$$\hat{\mathbf{w}}_{MAP} = \arg\min_{\mathbf{w}} \sum_{n=1}^{N}(y_n - \boldsymbol{w}^\top \boldsymbol{x}_n)^2 + \frac{\lambda}{\beta}\boldsymbol{w}^\top \boldsymbol{w}$$

Not surprising since MAP estimation indeed optimizes a regularized loss function!

- This is equivalent to ridge regression with regularization hyperparameter $\frac{\lambda}{\beta}$
- The solution will be $\hat{\boldsymbol{w}}_{MAP} = (\boldsymbol{X}^\top \boldsymbol{X} + \frac{\lambda}{\beta}\boldsymbol{I}_D)^{-1}\boldsymbol{X}^\top \boldsymbol{y}$

Fully Bayesian Inference for Prob. Linear Regression

- Can also compute the full posterior distribution over $\boldsymbol{w}$

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

For brevity, we have not shown the dependence of the various distributions here on the hyperparameters λ and β

- Likelihood and prior are conjugate (both Gaussians) - posterior will be Gaussian

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

$$\boldsymbol{\mu}_N = (\boldsymbol{X}^\top \boldsymbol{X} + \frac{\lambda}{\beta} \boldsymbol{I}_D)^{-1} \boldsymbol{X}^\top \boldsymbol{y}$$

$$\boldsymbol{\Sigma}_N = (\beta \boldsymbol{X}^\top \boldsymbol{X} + \lambda \boldsymbol{I}_D)^{-1}$$

Posterior's mean is the same as the MAP solution since the mean and mode of a Gaussian are the same

Note: λ and β are assumed to be fixed; otherwise, the problem is a bit harder (beyond the scope of CS771)

We now have a distribution over the possible solutions – it has a mean but we can generate other plausible solutions by sampling from this posterior. Each sample will give a weight vector

Prob. Linear Regression: The Predictive Distribution

- Want the predictive distribution $p(y_*|\boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{y})$ of the output y_* for a new input $\boldsymbol{x}_*$
- With MLE/MAP estimate of $\boldsymbol{w}$, we will use the plug-in predictive

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

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

- When doing fully Bayesian inference, can compute the posterior predictive dist.

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

Not true in general for Prob. Lin. Reg. but because the hyperparameters λ and β are treated as fixed

- Requires an integral but has a closed form

Mean prediction

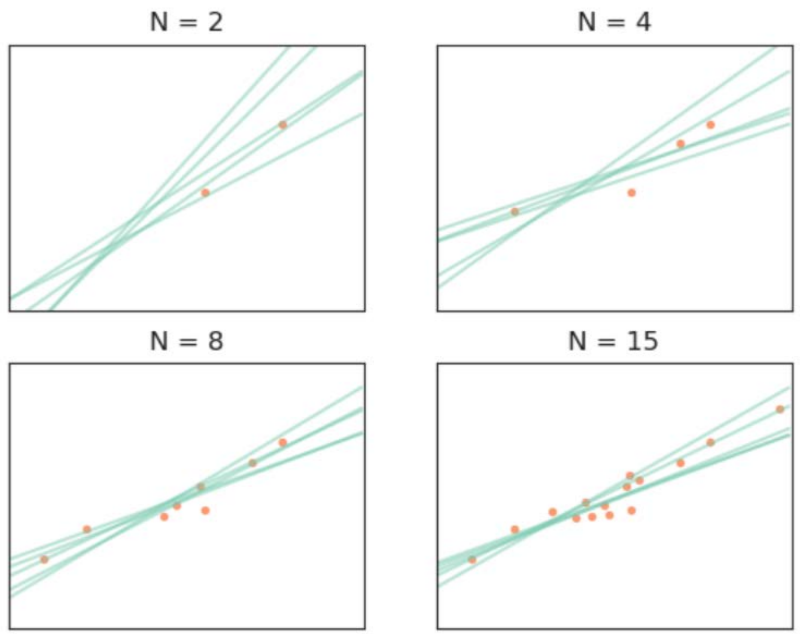
$$p(y_*|\mathbf{x}_*, \mathbf{X}, \mathbf{y}) = \mathcal{N}(\boldsymbol{\mu}_N^\top \mathbf{x}_*, \beta^{-1} + \mathbf{x}_*^\top \boldsymbol{\Sigma}_N \mathbf{x}_*)$$

Input-specific predictive variance unlike the MLE/MAP based predictive where it was β^{-1} (and was same for all test inputs)

- Input-specific predictive uncertainty useful in problems where we want confidence estimates of the predictions made by the model (e.g., Active Learning)

Fully Bayesian Linear Regression – Pictorially

- Each sample from posterior $p(\boldsymbol{w}|\boldsymbol{X},\boldsymbol{y}) = \mathcal{N}(\boldsymbol{\mu}_N, \boldsymbol{\Sigma}_N)$ will give a weight vector $\boldsymbol{w}$
  - In case of lin. reg., each weight vector corresponds to a regression line



The posterior sort of represents an ensemble of solutions (not all are equally good but we can use all of them in an "importance-weighted" fashion to make the prediction using the posterior predictive distribution)

Importance of each solution in this ensemble is its posterior probability $p(\boldsymbol{w}|\boldsymbol{X},\boldsymbol{y})$

- Each weight vector will give a different set of predictions on test data
  - These different predictions will give us a variance (uncertainty) estimate in model's prediction
  - The uncertainty decreases as N increases (we become more sure when we see more training data)

Pic source: https://waterprogramming.wordpress.com/

MLE, MAP/Fully Bayesian Lin. Reg: Summary

- MLE/MAP give point estimate of $\boldsymbol{w}$
  - MLE/MAP based prediction uses that single point estimate of $\boldsymbol{w}$
- Fully Bayesian approach gives the full posterior of $\boldsymbol{w}$
  - Fully Bayesian prediction does posterior averaging (computes posterior predictive distribution)
- Some things to keep in mind:
  - MLE estimation of a parameter leads to unregularized solutions
  - MAP estimation of a parameter leads to regularized solutions
  - A Gaussian likelihood model corresponds to using squared loss
  - A Gaussian prior on parameters acts as an ℓ_2 regularizer
  - Other likelihoods/priors can be chosen (result in other loss functions and regularizers)

E.g., using Laplace distribution for likelihood is equivalent to absolute loss, using it as a prior is equivalent to ℓ_1 regularization

Evaluation Measures for Regression Models

- Plotting the prediction $\hat{y}_n$ vs truth y_n for the validation/test set
- Residual Sum of Squares (RSS) on the validation/test set

$$RSS(\boldsymbol{w}) = \sum_{n=1}^{N} (y_n - \hat{y}_n)^2$$

- RMSE (Root Mean Squared Error) $\triangleq \sqrt{\frac{1}{N} RSS(\boldsymbol{w})}$
- Coefficient of determination or R^2

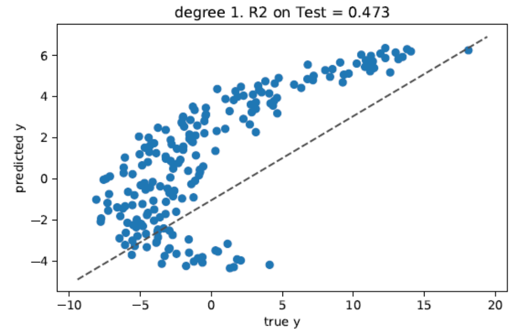
$$R^2 = 1 - \frac{\sum_{n=1}^{N} (y_n - \hat{y}_n)^2}{\sum_{n=1}^{N} (y_n - \bar{y})^2}$$

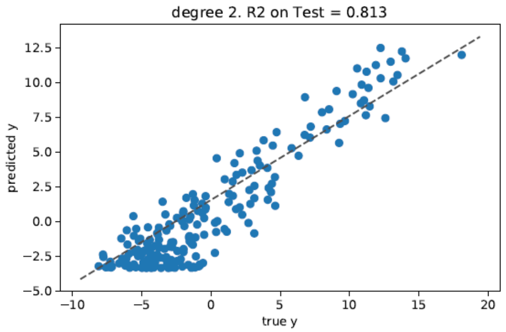
Unlike RSS and RMSE, it is always between 0 and 1 and hence interpretable

"relative" error w.r.t. a model that makes a constant prediction $\bar{y}$ for all inputs

$\bar{y}$ is empirical mean of true responses, i.e., $\frac{1}{N}\sum_{n=1}^{N} y_n$

Plots of true vs predicted outputs and R^2 for two regression models





Pic from MLAPP (Murphy)