Probabilistic Modeling of Data (contd)

CS771: Introduction to Machine Learning

Piyush Rai

Recap

- MLE and MAP used to estimate parameters of probabilistic models
- Both provide a "point estimate" of parameters by solving an optimization problem

Maxima of the likelihood function

Assuming i.i.d. observations

$$\theta_{MLE} = \arg\max_{\theta} \log p(\boldsymbol{y}|\theta) = \arg\max_{\theta} \sum_{n=1}^{N} \log p(y_n|\theta) = \arg\min_{\theta} \sum_{n=1}^{N} -\log p(y_n|\theta) = \arg\min_{\theta} NLL(\theta)$$

Maxima of the posterior distribution

Can be found even without computing the posterior

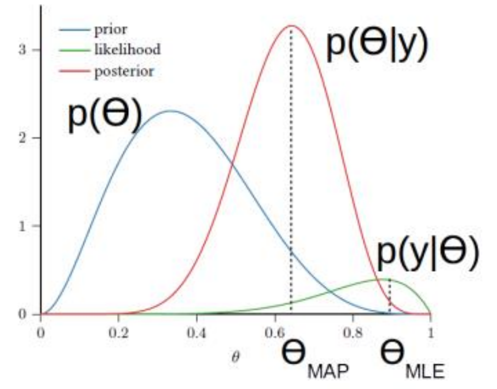
Expresses our prior knowledge/belief about the various values of θ can take

$$\theta_{MAP} = \arg\max_{\theta} \log p(\theta|\boldsymbol{y}) = \arg\max_{\theta} [\log p(\boldsymbol{y}|\theta) + \log p(\theta) = \arg\min_{\theta} [NLL(\theta) - \log p(\theta)]$$

- MAP is akin to doing a regularized MLE (prior also acts as a regularizer for $\boldsymbol{\theta}$)
- If we want more than just a point estimate, we can compute the full posterior

Once we compute this, we can get mean, mode (same as MAP solution), median, quantiles, variance, etc, of $\boldsymbol{\theta}$ (more holistic information)

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



Computing the full posterior distribution is in general much harder but if prior and likelihood are conjugate to each other, it is easy

Conjugacy

- Many pairs of distributions are conjugate to each other
  - Bernoulli (likelihood) + Beta (prior) ⇒ Beta posterior
  - Binomial (likelihood) + Beta (prior) ⇒ Beta posterior
  - Multinomial (likelihood) + Dirichlet (prior) ⇒ Dirichlet posterior
  - Poisson (likelihood) + Gamma (prior) ⇒ Gamma posterior
  - Gaussian (likelihood) + Gaussian (prior) ⇒ Gaussian posterior
  - and many other such pairs ..

Not in the general case but in special cases, e.g., when the variance of the Gaussian likelihood is held fixed and we want to estimate the mean of the Gaussian likelihood using a Gaussian prior on the mean

- Tip: If two distr are conjugate to each other, their functional forms are similar
  - Example: Bernoulli and Beta have the forms

$$\text{Bernoulli}(y|\theta) = \theta^{y}\,(1-\theta)^{1-y}$$

$$\text{Beta}(\theta|\alpha,\beta) = \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)}\,\theta^{\alpha-1}(1-\theta)^{\beta-1}$$

This is why, when we multiply them while computing the posterior, the exponents get added and we get the same form for the posterior as the prior but with just updated hyperparameter. Also, we can identify the posterior and its hyperparameters simply by inspection

Probabilistic Models: Making Predictions

- Having estimated $\boldsymbol{\theta}$, we can now use it to make predictions
- Prediction entails computing posterior predictive distribution of a new observation $\boldsymbol{y}_*$

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

Marginalizing (summing/integrating) over the unknown random variable $\boldsymbol{\theta}$

Conditional distribution of the new observation, given past observations

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

Decomposing the joint using chain rule

In some simple cases, this can be computed exactly but, in general, it can't be and needs to be approximated

$$= \int p(y_*|\theta) p(\theta|\boldsymbol{y}) d\theta$$

This step assumes i.i.d. data, i.e., given $\boldsymbol{\theta}$, $\boldsymbol{y}_*$ does not depend on $\boldsymbol{y}$

$$= \mathbb{E}_{p(\theta|\boldsymbol{y})}[p(y_*|\theta)]$$

This computes the predictive distribution by averaging over the full posterior – basically calculates $\boldsymbol{p}(\boldsymbol{y}_*|\boldsymbol{\theta})$ for each possible $\boldsymbol{\theta}$, weighs it by the probability of $\boldsymbol{\theta}$ under the posterior $\boldsymbol{p}(\boldsymbol{\theta}|\boldsymbol{y})$, and sums all such posterior weighted predictions. Note that not each value of $\boldsymbol{\theta}$ is given equal importance here in the averaging

- When doing MLE/MAP, we approximate the posterior $\boldsymbol{p}(\boldsymbol{\theta}|\boldsymbol{y})$ by a single point $\boldsymbol{\theta}_{opt}$

$$p(y_*|\boldsymbol{y}) = \int p(y_*|\theta) p(\theta|\boldsymbol{y}) d\theta \approx p(y_*|\theta_{opt})$$

A "plug-in predictive distribution" - simply plugged in the single best estimate (MLE/MAP) that we have

- Plug-in prediction which uses MLE/MAP of $\boldsymbol{\theta}$ is cheaper since no integral involved

Making Predictions: An Example

- For coin-toss example, prediction means computing $p(y_{N+1} = 1|\boldsymbol{y})$
- This can be done using the MLE/MAP estimate, or using the full posterior

$$\theta_{MLE} = \frac{N_1}{N} \qquad \theta_{MAP} = \frac{N_1 + \alpha - 1}{N + \alpha + \beta - 2} \qquad p(\theta|\boldsymbol{y}) = \text{Beta}(\theta|\alpha + N_1, \beta + N_0)$$

- Thus for this example (where observations are assumed iid from a Bernoulli)

MLE prediction: $p(y_{N+1} = 1|\mathbf{y}) = \int p(y_{N+1} = 1|\theta)p(\theta|\mathbf{y})d\theta \approx p(y_{N+1} = 1|\theta_{MLE}) = \theta_{MLE} = \frac{N_1}{N}$

MAP prediction: $p(y_{N+1} = 1|\mathbf{y}) = \int p(y_{N+1} = 1|\theta)p(\theta|\mathbf{y})d\theta \approx p(y_{N+1} = 1|\theta_{MAP}) = \theta_{MAP} = \frac{N_1 + \alpha - 1}{N + \alpha + \beta - 2}$

Fully Bayesian: $p(y_{N+1} = 1|\mathbf{y}) = \int p(y_{N+1} = 1|\theta)p(\theta|\mathbf{y})d\theta = \int \theta p(\theta|\mathbf{y})d\theta = \int \theta \text{Beta}(\theta|\alpha + N_1, \beta + N_0)d\theta = \frac{N_1 + \alpha}{N + \alpha + \beta}$

Again, keep in mind that the posterior weighted averaged prediction used in the fully Bayesian case would usually not be as simple to compute as it was in this case. We will look at some hard cases later

Expectation of $\boldsymbol{\theta}$ under the Beta posterior that we computed using fully Bayesian inference

Predictive Distribution: About notation

- We used the following notation for the predictive distribution

$$p(y_*|\boldsymbol{y}) = \int p(y_*|\theta)p(\theta|\boldsymbol{y})d\theta$$

Posterior predictive distribution (PPD)

$$p(y_*|\boldsymbol{y}) \approx p(y_*|\theta_{opt})$$

Plug-in predictive distribution

An approximation of the PPD using a single best value $\boldsymbol{\theta_{opt}}$

- In some case, as we will see, it may additionally depend on other given quantities

$$p(y_*|\boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{y})$$

The PPD for a supervised learning model: Depends on the test input $\boldsymbol{x}_*$ and the training data $\boldsymbol{X}, \boldsymbol{y}$

$$p(y_*|\boldsymbol{x}_*, \theta_{opt})$$

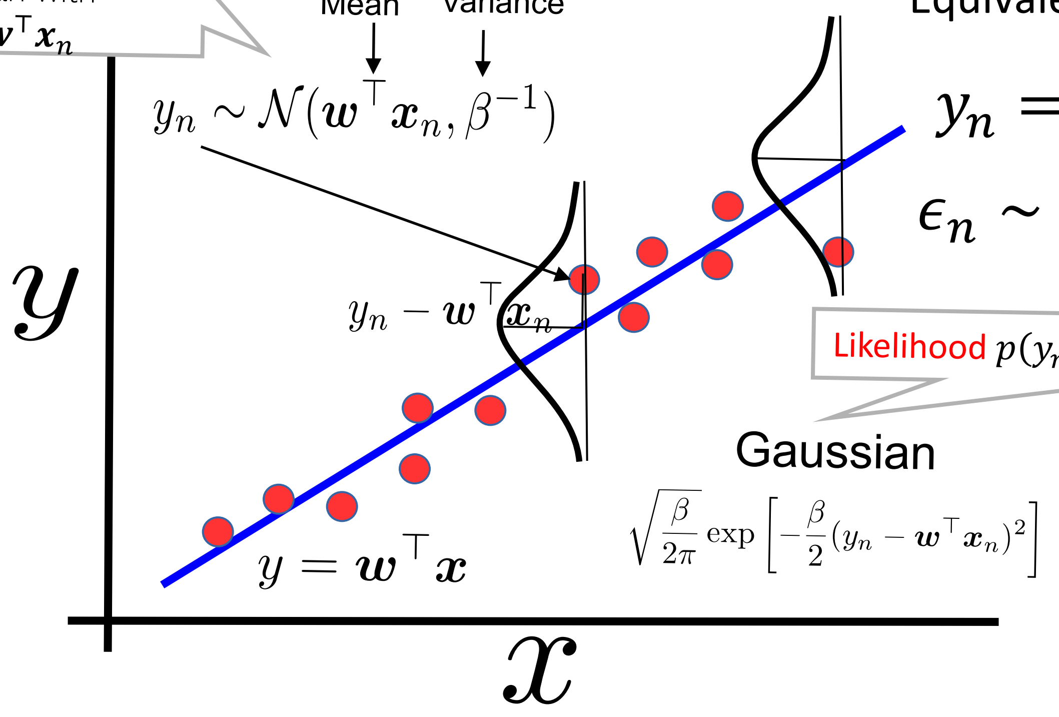
The plug-in predictive for a supervised learning model: depends on the test input $\boldsymbol{x}_*$ and the point estimate of $\boldsymbol{\theta}$

Probabilistic Models for Supervised Learning

Why Probabilistic Models for Supervised Learning?

- We can use a probability distribution to model the input-output relationship

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



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

Equivalently:

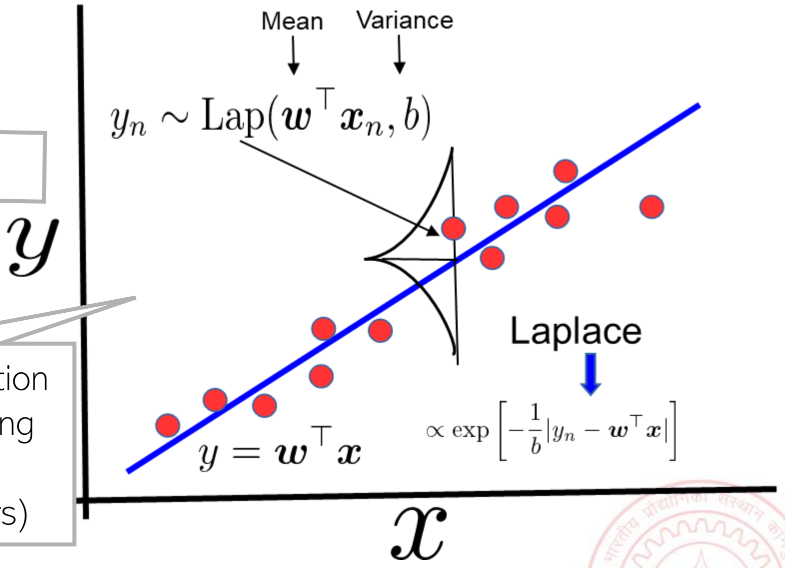
Output $\boldsymbol{y_n}$ assumed generated from a linear model and then zero mean Gaussian noise added

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

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

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

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



Mean
Variance
$y_n \sim \mathrm{Lap}(\boldsymbol{w}^\top \boldsymbol{x}_n, b)$
y
Laplace
$\propto \exp\left[-\frac{1}{b}|y_n - \boldsymbol{w}^\top \boldsymbol{x}|\right]$
$y = \boldsymbol{w}^\top \boldsymbol{x}$
x

Using a Laplace distribution would correspond to using an absolute loss (better model if data has outliers)

- Depending on the nature of the data, we can choose a suitable likelihood model (and it naturally corresponds to a loss function, e.g., squared, cross-entropy, etc)

Why Probabilistic Models for Supervised Learning?

- We can use suitable priors for the model parameters $\boldsymbol{w}$ (and these priors naturally correspond to regularizers) and also compute the posterior of $\boldsymbol{w}$

In form of $p(y_*|\boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{y})$ or $p(y_*|\boldsymbol{x}_*, \boldsymbol{w}_{opt})$

- For test input $\boldsymbol{x}_*$, we can compute distribution over the predicted label
  - This gives us not just a single "mean" prediction y_* but also its uncertainty/variance
  - Variance in the prediction of y_* can be used as a measure of confidence
  - Variance/confidence in predictions is useful in many domains such as healthcare, and in active learning (using a learned model to to collect more training data to improve it)
- There are various other benefits (will see later)
  - Handling missing data (missing features, missing labels, etc)
  - Hyperparameter estimation

Prob. Models for Supervised Learning

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

- Goal: Learn the conditional distribution $p(y|\boldsymbol{x})$. Broadly, two approaches

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 $\boldsymbol{x}$)

- Both approaches have their pros and cons (discussed later)

The Discriminative Approach

- This approach models $\boldsymbol{p(y|x)}$ directly using a suitable prob. distribution, e.g.,

Regression model likelihood

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

Binary classification model likelihood

$\mu_i = \sigma(\boldsymbol{w}^\top\boldsymbol{x}_i)$

$$p(y_i|\boldsymbol{x}_i,\boldsymbol{w}) = \text{Bernoulli}(y_i|\sigma(\boldsymbol{w}^\top\boldsymbol{x}_i)) = \mu_i^{y_i}(1-\mu_i)^{1-y_i}$$

- The negative log-likelihood (assuming i.i.d. outputs) for the above two models

$$NLL(\boldsymbol{w}) = \sum_{i=1}^{N} -\log p(y_i|\boldsymbol{x}_i,\boldsymbol{w}) = \sum_{i=1}^{N} -\log\sqrt{\frac{\beta}{2\pi}}\exp\left[-\frac{\beta}{2}(y_i-\boldsymbol{w}^\top\boldsymbol{x}_i)^2\right]$$

$NLL(\boldsymbol{w}) = \frac{\beta}{2}\sum_{n=1}^{N}(y_i-\boldsymbol{w}^\top\boldsymbol{x}_i)^2$ (same as squared loss ☺)

$$NLL(\boldsymbol{w}) = \sum_{i=1}^{N} -\log p(y_i|\boldsymbol{x}_i,\boldsymbol{w}) = \sum_{i=1}^{N} -\log \mu_i^{y_i}(1-\mu_i)^{1-y_i}$$

$NLL(\boldsymbol{w}) = -\sum_{n=1}^{N}[y_i\log\mu_i + (1-y_i)\log(1-\mu_i)]$ (same as cross-entropy loss ☺)

Thus minimization of NLL (i.e., doing MLE) is equivalent to minimization of the training loss

Lacks regularization (and thus can overfit) but we can use a prior on $\boldsymbol{w}$ to do MAP estimation

A prior over $\boldsymbol{w}$

- For MAP estimation (and to compute full posterior), we need a prior $\boldsymbol{w} \in \mathbb{R}^D$
- 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 $\boldsymbol{w}_0$

Σ specifies how strong our belief is that $\boldsymbol{w}$ to close to $\boldsymbol{w}_0$

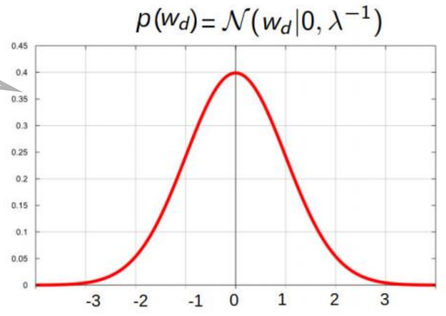
- A specific example of a multivariate Gaussian prior in this problem

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

Omitting λ for brevity

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

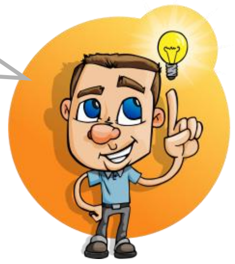
The precision λ of the Gaussian prior controls how aggressively the prior pushes the elements towards mean (0)



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

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

Aha! This $\boldsymbol{w}^\top \boldsymbol{w}$ term reminds me of the ℓ_2 regularizer ☺



That's indeed the case ☺

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

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

- The MAP objective (log-posterior) will be the log-likelihood + $\log p(\boldsymbol{w})$
- Ignoring terms that don't depend on $\boldsymbol{w}$ the log-posterior will be

$$NLL(\boldsymbol{w}) - \log\exp\left[-\frac{\lambda}{2}\boldsymbol{w}^\top\boldsymbol{w}\right] = NLL(\boldsymbol{w}) + \frac{\lambda}{2}\boldsymbol{w}^\top\boldsymbol{w}$$

- Exercise: For regression with likelihood $p(y_i|\boldsymbol{x}_i, \boldsymbol{w}) = \mathcal{N}(y_i|\boldsymbol{w}^\top\boldsymbol{x}_i, \beta^{-1})$

$$\widehat{w}_{MAP} = \underset{\boldsymbol{w}}{\operatorname{argmin}} \frac{\beta}{2}\sum_{i=1}^{N}(y_i - \boldsymbol{w}^\top\boldsymbol{x}_i)^2 + \frac{\lambda}{2}\boldsymbol{w}^\top\boldsymbol{w} = (\boldsymbol{X}^\top\boldsymbol{X} + \frac{\lambda}{\beta}\boldsymbol{I}_D)^{-1}\ \boldsymbol{X}^\top\boldsymbol{y}$$

- Classification with likelihood $p(y_i|\boldsymbol{x}_i, \boldsymbol{w}) = \text{Bernoulli}(y_i|\sigma(\boldsymbol{w}^\top\boldsymbol{x}_i))$ is the same as logistic regression. When also using the above prior, the MAP solution will be the same as cross-entropy loss minimization + L2 regularization on the weights

Prob. Linear Regression: The Full Posterior

- If we want the full posterior distribution over $\boldsymbol{w}$, it can be computed as

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

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

- For regression, with Gaussian likelihood and zero mean Gaussian prior are conjugate (both Gaussians), and the posterior will be Gaussian

Will provide the proof separately but it is straightforward when using properties of 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)

- For binary classification with Bernoulli likelihood and zero mean Gaussian prior, we don't have conjugacy, and the posterior can't be computed in closed form

Logistic Regression: The Full Posterior

- The posterior distribution for the logistic regression model

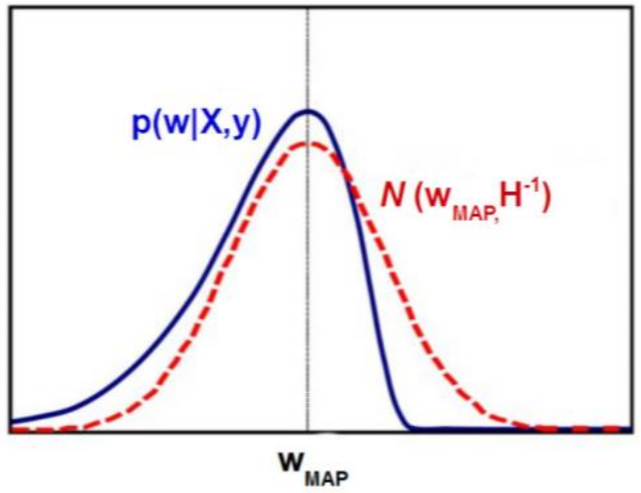
Gaussian

Bernoulli

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

Unfortunately, Gaussian and Bernoulli are not conjugate with each other, so analytic expression for the posterior can't be obtained unlike prob. linear regression

- Need to approximate the posterior in this case
- We will use a simple approximation called Laplace approximation

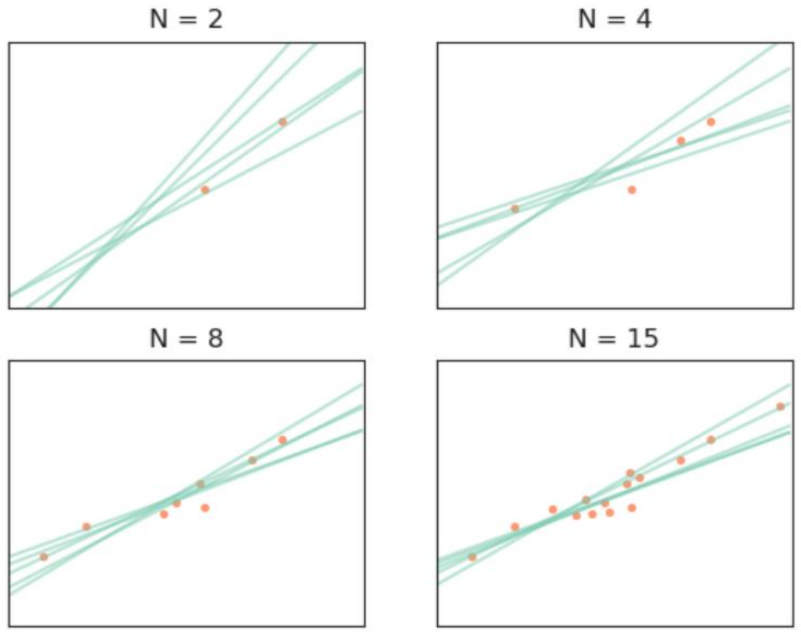


Approximates the posterior of $\boldsymbol{w}$ by a Gaussian whose mean is the MAP solution $\hat{\boldsymbol{w}}_{MAP}$ and covariance matrix is the inverse of the Hessian (Hessian: second derivative of the negative log-posterior of the LR model)

Can also employ more advanced posterior approximation methods, like MCMC and variational inference (beyond the scope of CS771)

What does posterior of a linear model look like ?

- 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)

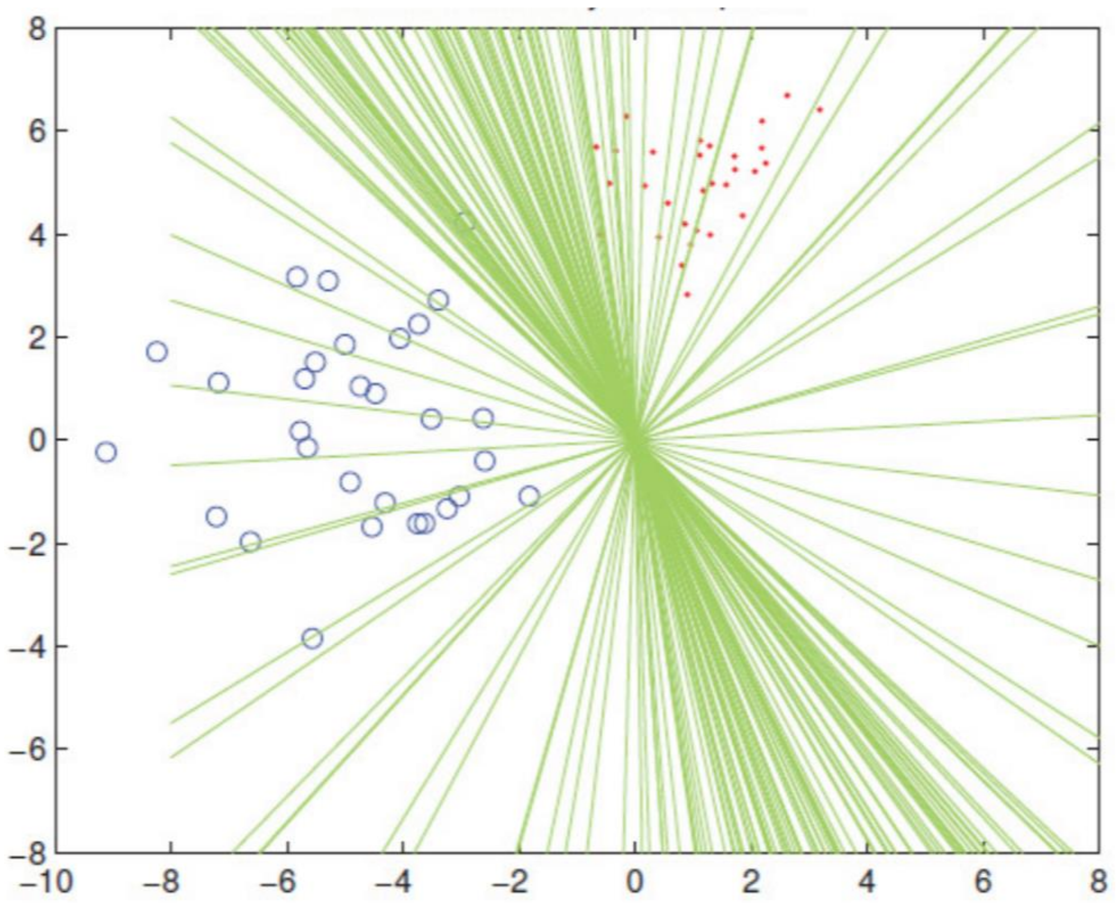
Importance of each solution in this ensemble is its posterior probability $p(\boldsymbol{w}|\boldsymbol{X},\boldsymbol{y})$

- 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/

What does posterior of a linear model look like ?

- Can also sample from the Laplace approximate posterior of logistic regression
- Each sample will give a weight vec defining a hyperplane separator



Not all separators are equally good; their goodness depends on their posterior probabilities

When making predictions, we can still use all of them but weighted by their importance based on their posterior probabilities

That's exactly what we do when computing the predictive distribution

Prob. Linear Regression: 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

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

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

- If we have the full posterior over $\boldsymbol{w}$, can compute the posterior predictive dist.

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

Assuming the hyperparameters λ and β are known (otherwise, the PPD can't be computed exactly)

- Requires an integral but has a closed form

Mean prediction

Will provide the proof separately but it is straightforward when using properties of Gaussian

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

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)

Logistic Regression: Predictive Distribution

- When using MLE/MAP solution $\hat{\boldsymbol{w}}_{opt}$, can use the plug-in predictive distribution

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

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

- When using fully Bayesian inference, we must compute the posterior predictive

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

Integral not tractable and must be approximated

sigmoid

Gaussian (if using Laplace approx.)

Monte-Carlo approximation of this integral is one possible way

Generate M samples $\boldsymbol{w}_1, \boldsymbol{w}_2, \dots, \boldsymbol{w}_M$, from the Gaussian approx. of posterior and use $p(y_* = 1|\boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{y}) \approx \frac{1}{M}\sum_{m=1}^{M} p(y_* = 1|\boldsymbol{w}_\text{m}, \boldsymbol{x}_*) = \frac{1}{M}\sum_{m=1}^{M} \sigma(\boldsymbol{w}_m^\top \boldsymbol{x}_n)$

Coming up..

- The generative approach to supervised learning