Probabilistic Models for Supervised Learning(2): Logistic and Softmax Regression

CS771: Introduction to Machine Learning

Piyush Rai

Logistic Regression (LR)

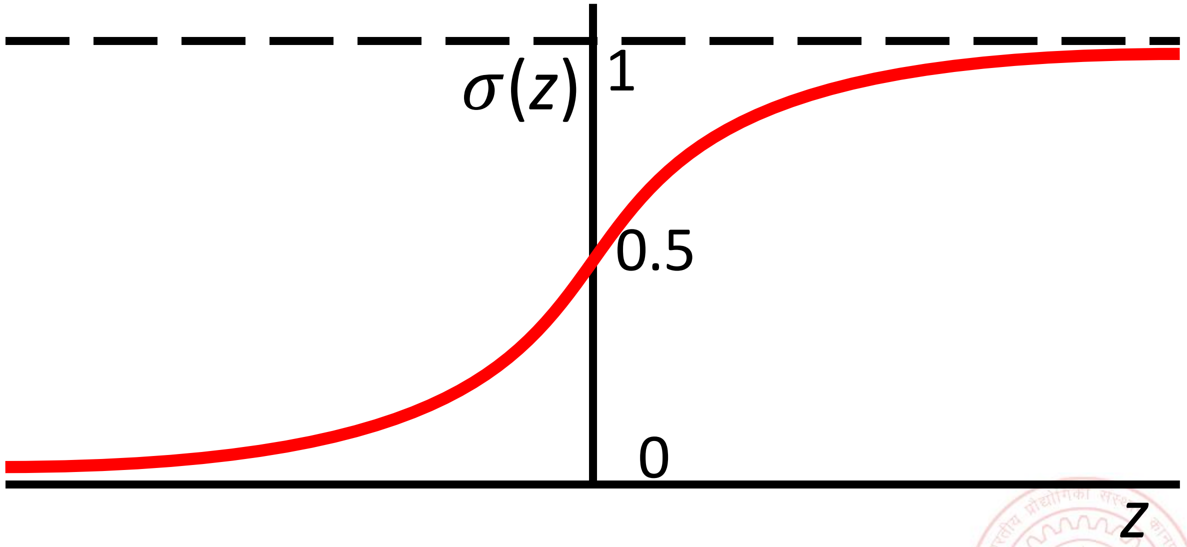
Both very widely used

Multi-class extension known as "softmax regression"

The word "regression" is a misnomer. Both are classification models

- A probabilistic model for binary classification
- Learns the PMF of the output label given the input, i.e., $p(y|\boldsymbol{x})$
- A discriminative model: Does not model inputs $\boldsymbol{x}$ (only relationship b/w $\boldsymbol{x}$ and y)
- Uses the sigmoid function to define the conditional probability of y being 1

$$\mu_x = p(y = 1|\boldsymbol{w}, \boldsymbol{x}) = \sigma(\boldsymbol{w}^\top \boldsymbol{x})$$
$$= \frac{1}{1 + \exp(-\boldsymbol{w}^\top \boldsymbol{x})}$$
$$= \frac{\exp(\boldsymbol{w}^\top \boldsymbol{x})}{1 + \exp(\boldsymbol{w}^\top \boldsymbol{x})}$$



A linear model

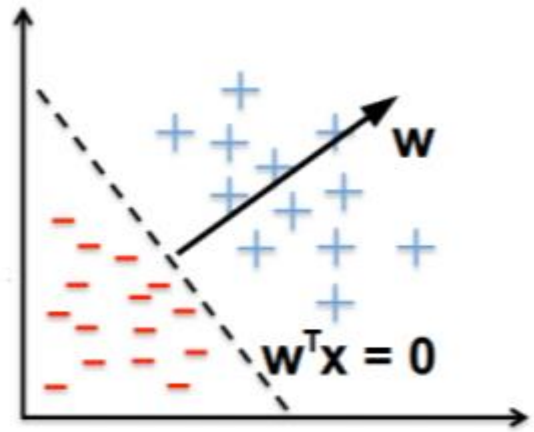
- Here $\boldsymbol{w}^\top \boldsymbol{x}$ is the score for input $\boldsymbol{x}$. The sigmoid turns it into a probability

LR: Decision Boundary

- At the decision boundary where both classes are equiprobable

$$
\begin{aligned}
p(y=1|\mathbf{x},\mathbf{w}) &= p(y=0|\mathbf{x},\mathbf{w}) \\
\frac{\exp(\mathbf{w}^\top \mathbf{x})}{1+\exp(\mathbf{w}^\top \mathbf{x})} &= \frac{1}{1+\exp(\mathbf{w}^\top \mathbf{x})} \\
\exp(\mathbf{w}^\top \mathbf{x}) &= 1 \\
\mathbf{w}^\top \mathbf{x} &= 0
\end{aligned}
$$

A linear hyperplane



- Very large positive $\boldsymbol{w}^\top\boldsymbol{x}$ means $p(y=1|\boldsymbol{w},\boldsymbol{x})$ close to 1
- Very large negative $\boldsymbol{w}^\top\boldsymbol{x}$ means $p(y=0|\boldsymbol{w},\boldsymbol{x})$ close to 1
- At decision boundary, $\boldsymbol{w}^\top\boldsymbol{x}=0$ implies $p(y=1|\boldsymbol{w},\boldsymbol{x}) = p(y=0|\boldsymbol{w},\boldsymbol{x}) = 0.5$

MLE for Logistic Regression

Assumed 0/1, not -1/+1

- Likelihood (PMF of each input's label) is Bernoulli with prob $\mu_n = \frac{\exp(\boldsymbol{w}^\top \boldsymbol{x}_n)}{1+\exp(\boldsymbol{w}^\top \boldsymbol{x}_n)}$

$$p(y_n|\boldsymbol{w}, \boldsymbol{x}_n) = \text{Bernoulli}(\mu_n) = \mu_n^{y_n}(1-\mu_n)^{1-y_n}$$

- Overall likelihood, assuming i.i.d. observations

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

"cross-entropy" loss (a popular loss function for classification)

- The negative log-likelihood $NLL(\boldsymbol{w}) = -\log p(\boldsymbol{y}|\boldsymbol{w}, \boldsymbol{X})$ simplifies to

Loss function

$$NLL(\boldsymbol{w}) = \sum_{n=1}^{N} -[y_n \log \mu_n + (1-y_n)\log(1-\mu_n)]$$

Very large loss if y_n close to 1 and μ_n close to 0, or vice-versa

- Plugging in $\mu_n = \frac{\exp(\boldsymbol{w}^\top \boldsymbol{x}_n)}{1+\exp(\boldsymbol{w}^\top \boldsymbol{x}_n)}$ and simplifying

Good news: For LR, NLL is convex

$$\boxed{NLL(\boldsymbol{w}) = -\sum_{n=1}^{N} [y_n \boldsymbol{w}^\top \boldsymbol{x}_n - \log(1+\exp(\boldsymbol{w}^\top \boldsymbol{x}_n))]}$$

No closed-form expression for $\hat{\boldsymbol{w}}_{MLE} = \arg\min_{\boldsymbol{w}} NLL(\boldsymbol{w})$

Iterative opt needed (gradient or Hessian based). **Exercise:** Try working out the gradient of NLL and notice the expression's form

An Alternate Notation

- If we assume the label $\boldsymbol{y_n}$ as -1/+1 (not 0/1), the likelihood can be written as

$$p(y_n|\boldsymbol{w},\boldsymbol{x}_n) = \frac{1}{1+\exp(-y_n\boldsymbol{w}^\top\boldsymbol{x}_n)} = \sigma(y_n\boldsymbol{w}^\top\boldsymbol{x}_n)$$

- Slightly more convenient notation: A single expression gives the probabilities of both possible label values

- In this case, the total negative log-likelihood will be

$$NLL(\boldsymbol{w}) = \sum_{n=1}^{N} -\log p(y_n|\boldsymbol{w},\boldsymbol{x}_n) = \sum_{n=1}^{N} \log\left(1+\exp(-y_n\boldsymbol{w}^\top\boldsymbol{x}_n)\right)$$

MAP Estimation for Logistic Regression

- Need a prior on the weight vector $\boldsymbol{w} \in \mathbb{R}^D$
- Just like probabilistic linear regression, can use a zero-mean Gaussian prior

$$p(\boldsymbol{w}) = \mathcal{N}(\boldsymbol{w}|\boldsymbol{0}, \lambda^{-1}\boldsymbol{I}_D) \propto \exp\left(-\frac{\lambda}{2}\boldsymbol{w}^\top\boldsymbol{w}\right)$$

Or NLL – log of prior

- The MAP objective (log of posterior) will be log-likelihood + log of prior
- Therefore the MAP solution (ignoring terms that don't depend on $\boldsymbol{w}$) will be

$$\hat{\boldsymbol{w}}_{MAP} = \arg\min_{\boldsymbol{w}} NLL(\boldsymbol{w}) + \frac{\lambda}{2}\boldsymbol{w}^\top\boldsymbol{w}$$

Good news: convex objective

- Just like MLE case, no closed form solution. Iterative opt methods needed
  - Highly efficient solvers (both first and second order) exist for MLE/MAP estimation for LR

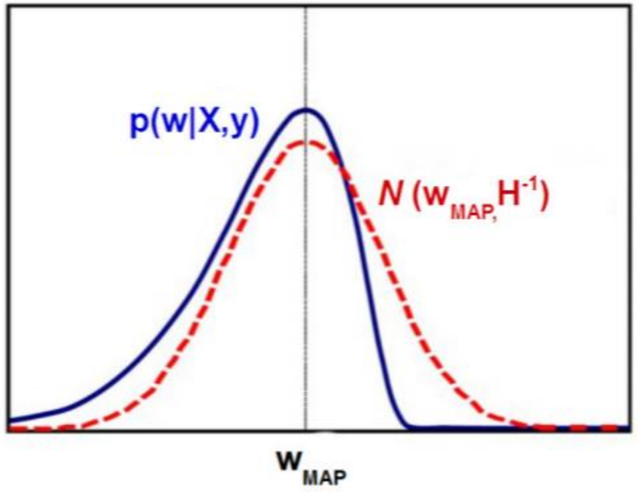
Fully Bayesian Inference for Logistic Regression

- Doing fully Bayesian inference would require computing the posterior

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

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

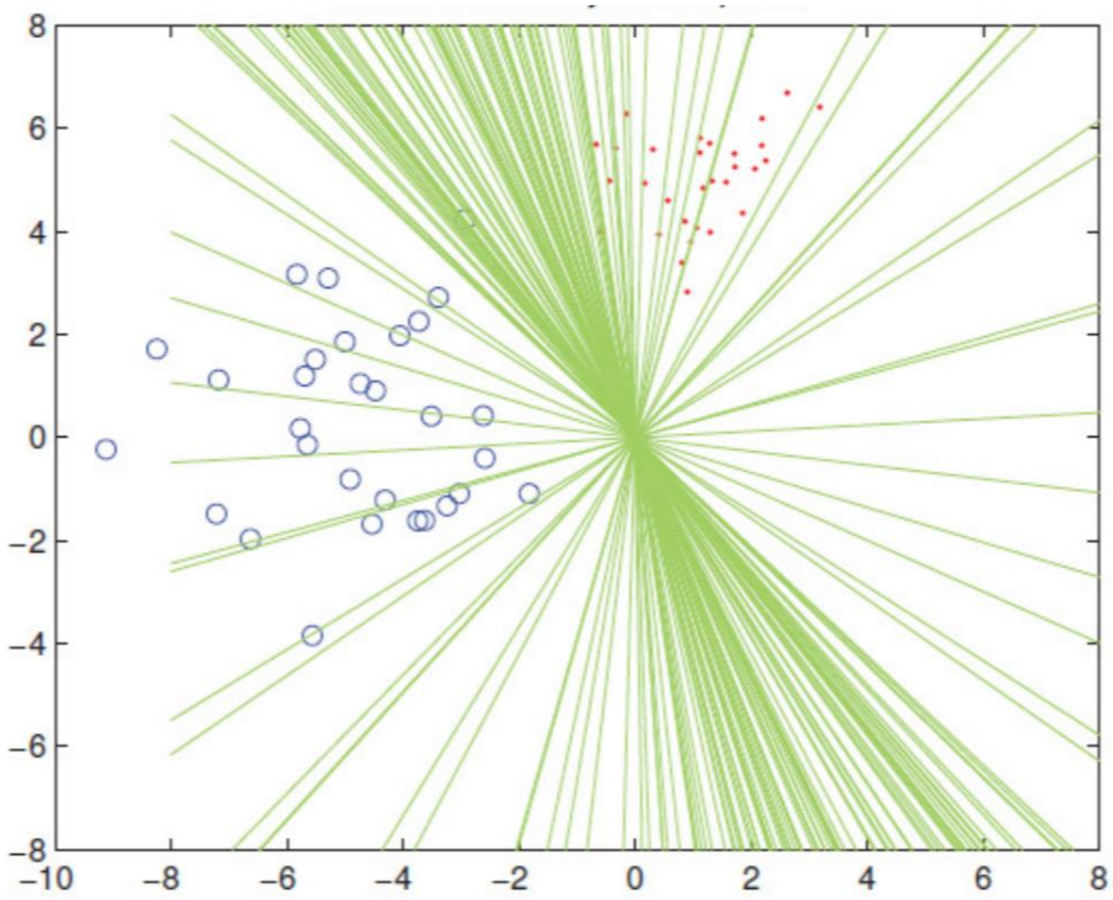


Approximates the posterior of $\boldsymbol{w}$ by a Gaussian whose mean is the MAP solution $\hat{\boldsymbol{w}}_{MAP}$ and covariance matrix is the inverse of the Hessian (Hessian: second derivative of the negative log-posterior of the LR model)

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

Posterior for LR: An Illustration

- Can sample from the posterior of the LR model
- 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

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

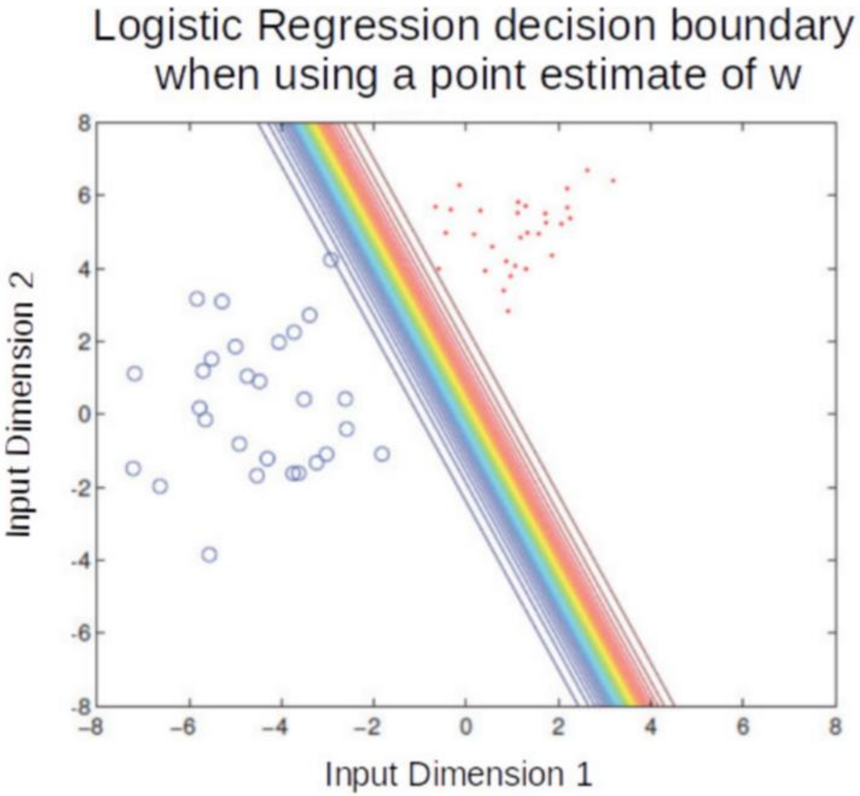
sigmoid

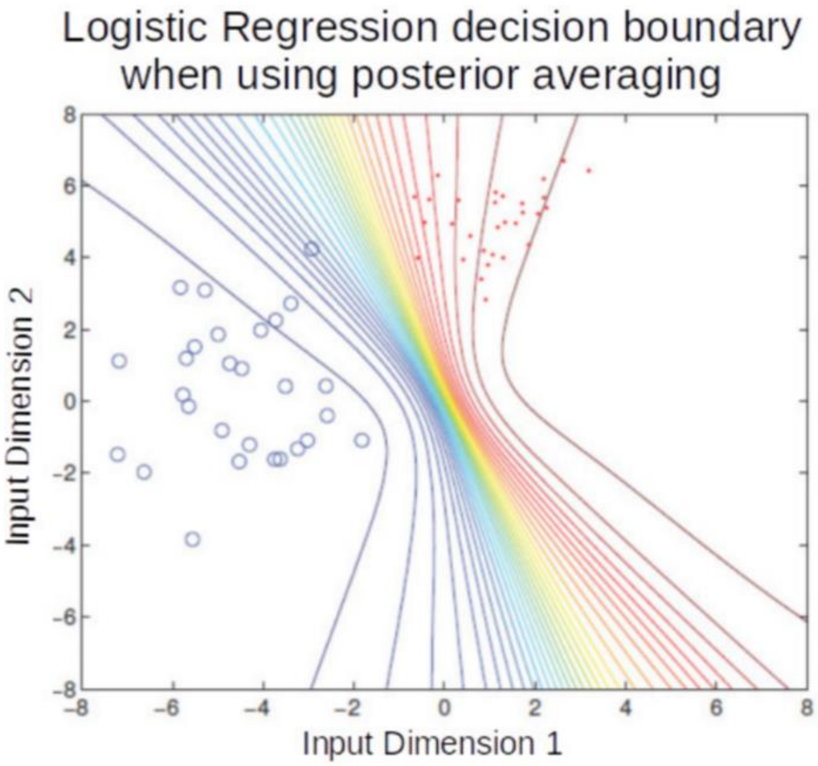
Gaussian (if using Laplace approx.)

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

LR: Plug-in Prediction vs Postrerior Averaging





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

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

Softmax function

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

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

- K weight vecs $\boldsymbol{w}_1, \boldsymbol{w}_2, \dots, \boldsymbol{w}_K$ (one per class), each D-dim, and $\boldsymbol{W} = [\boldsymbol{w}_1, \boldsymbol{w}_2, \dots, \boldsymbol{w}_K]$
- 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

Coming up next

- Generative models for supervised learning