LOGISTIC REGRESSION

Master's Deep Learning

Sergey Nikolenko

Harbour Space University, Barcelona, Spain
November 9, 2017

CLASSIFICATION AND LOGISTIC REGRESSION

- For classification problems it is even more clear: we want to classify a vector $\mathbf{x}$ to one of K classes $\mathcal{C}_k$.
- Suppose that class $\mathcal{C}_k$ has density $p(\mathbf{x} \mid \mathcal{C}_k)$, find prior distributions $p(\mathcal{C}_k)$, and then compute $p(\mathcal{C}_k \mid \mathbf{x})$ by Bayes' theorem.
- For two classes:

$$p(\mathcal{C}_1 \mid \mathbf{x}) = \frac{p(\mathbf{x} \mid \mathcal{C}_1)p(\mathcal{C}_1)}{p(\mathbf{x} \mid \mathcal{C}_1)p(\mathcal{C}_1) + p(\mathbf{x} \mid \mathcal{C}_2)p(\mathcal{C}_2)}.$$

- We rewrite:

$$p(\mathcal{C}_1 \mid \mathbf{x}) = \frac{p(\mathbf{x} \mid \mathcal{C}_1)p(\mathcal{C}_1)}{p(\mathbf{x} \mid \mathcal{C}_1)p(\mathcal{C}_1) + p(\mathbf{x} \mid \mathcal{C}_2)p(\mathcal{C}_2)} = \frac{1}{1+e^{-a}} = \sigma(a),$$

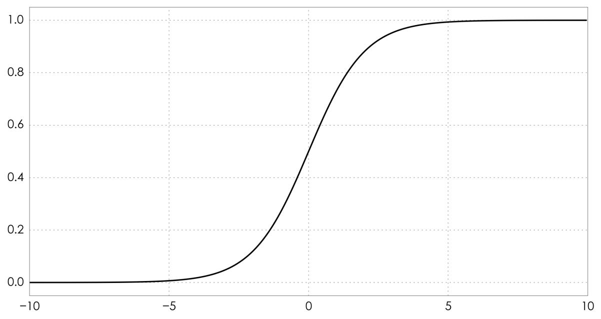
where

$$a = \ln \frac{p(\mathbf{x} \mid \mathcal{C}_1)p(\mathcal{C}_1)}{p(\mathbf{x} \mid \mathcal{C}_2)p(\mathcal{C}_2)}, \qquad \sigma(a) = \frac{1}{1+e^{-a}}.$$

- $\sigma(a)$ is the *logistic sigmoid*:

$$\sigma(a) = \frac{1}{1 + e^{-a}}$$

- $\sigma(-a) = 1 - \sigma(a)$.
- $a = \ln\left(\frac{\sigma}{1-\sigma}\right)$ – *logit function*.



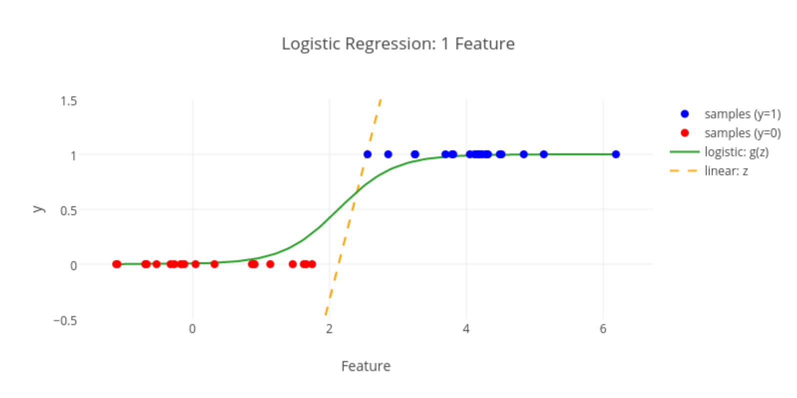
- This, in particular, leads to *logistic regression*: we optimize $\mathbf{w}$ directly.
- For a dataset $\{\phi_n, t_n\}$, $t_n \in \{0, 1\}$, $\phi_n = \phi(\mathbf{x}_n)$:

$$p(\mathbf{t} \mid \mathbf{w}) = \prod_{n=1}^{N} y_n^{t_n} (1 - y_n)^{1-t_n}, \quad y_n = p(\mathcal{C}_1 \mid \phi_n).$$

- We find maximal likelihood parameters, minimizing $-\ln p(\mathbf{t} \mid \mathbf{w})$:

$$E(\mathbf{w}) = -\ln p(\mathbf{t} \mid \mathbf{w}) = -\sum_{n=1}^{N} \left[t_n \ln y_n + (1 - t_n) \ln(1 - y_n) \right].$$

- And we get a sigmoid that optimally separates the data and that even tries to model probabilities:



- Let's go back to classification.
- Two classes, the posterior is the logistic sigmoid of a linear function:

$$p(\mathcal{C}_1 \mid \phi) = y(\phi) = \sigma(\mathbf{w}^\top \phi), \quad p(\mathcal{C}_2 \mid \phi) = 1 - p(\mathcal{C}_1 \mid \phi).$$

- *Logistic regression* is when we optimize $\mathbf{w}$ directly.

TWO CLASSES

- For a dataset $\{\phi_n, t_n\}$, $t_n \in \{0, 1\}$, $\phi_n = \phi(\mathbf{x}_n)$:

$$p(\mathbf{t} \mid \mathbf{w}) = \prod_{n=1}^{N} y_n^{t_n} (1 - y_n)^{1-t_n}, \quad y_n = p(\mathcal{C}_1 \mid \phi_n).$$

- We look for maximal likelihood parameters by minimizing $-\ln p(\mathbf{t} \mid \mathbf{w})$:

$$E(\mathbf{w}) = -\ln p(\mathbf{t} \mid \mathbf{w}) = -\sum_{n=1}^{N} \left[t_n \ln y_n + (1 - t_n) \ln(1 - y_n)\right].$$

- Since $\sigma' = \sigma(1 - \sigma)$, we take the gradient:

$$\nabla E(\mathbf{w}) = \sum_{n=1}^{N} (y_n - t_n) \phi_n.$$

- If we now perform gradient descent, we get the separating surface.
- Note that if the data are actually separable, we could get heavy overfitting: $\|\mathbf{w}\| \to \infty$, and the sigmoid turns into a Heaviside function.
- We have to regularize.

IRLS

- Logistic regression does not yield a closed form solution because of the sigmoid.
- But function $E(\mathbf{w})$ is convex, and we can use Newton–Raphson's method: use local quadratic approximation to the loss function on each step:

$$\mathbf{w}^{\text{new}} = \mathbf{w}^{\text{old}} - \mathbf{H}^{-1}\nabla E(\mathbf{w}),$$

  where $\mathbf{H}$ (Hessian) is the matrix of second derivatives for $E(\mathbf{w})$.

- Aside: let us apply Newton–Raphson's method to regular linear regression with quadratic error:

$$\nabla E(\mathbf{w}) = \sum_{n=1}^{N} \left(\mathbf{w}^\top \phi_n - t_n\right) \phi_n = \Phi^\top \Phi \mathbf{w} - \Phi^\top \mathbf{t},$$

$$\nabla\nabla E(\mathbf{w}) = \sum_{n=1}^{N} \phi_n \phi_n^\top = \Phi^\top \Phi,$$

and the optimization step will be

$$\mathbf{w}^{\text{new}} = \mathbf{w}^{\text{old}} - \left(\Phi^\top \Phi\right)^{-1} \left[\Phi^\top \Phi \mathbf{w}^{\text{old}} - \Phi^\top \mathbf{t}\right] = \\ = \left(\Phi^\top \Phi\right)^{-1} \Phi^\top \mathbf{t},$$

i.e., we get a solution in one step.

- For logistic regression:

$$\nabla E(\mathbf{w}) = \sum_{n=1}^{N} (y_n - t_n)\, \phi_n = \Phi^\top (\mathbf{y} - \mathbf{t})\,,$$

$$\mathbf{H} = \nabla\nabla E(\mathbf{w}) = \sum_{n=1}^{N} y_n(1 - y_n)\phi_n \phi_n^\top = \Phi^\top R \Phi$$

for a diagonal matrix R c $R_{nn} = y_n(1 - y_n)$.

- Optimization step formula:

$$\mathbf{w}^{\text{new}} = \mathbf{w}^{\text{old}} - \left(\Phi^\top R \Phi\right)^{-1} \Phi^\top (\mathbf{y} - \mathbf{t}) = \left(\Phi^\top R \Phi\right)^{-1} \Phi^\top R \mathbf{z},$$

  where $\mathbf{z} = \Phi \mathbf{w}^{\text{old}} - R^{-1} (\mathbf{y} - \mathbf{t})$.
- This is like a weighted least squares optimization problem with matrix of weights R.
- Hence the title: iterative reweighted least squares (IRLS).

SEVERAL CLASSES

- In case of several classes

$$p(\mathcal{C}_k \mid \phi) = y_k(\phi) = \frac{e^{a_k}}{\sum_j e^{a_j}} \text{ for } a_k = \mathbf{w}_k^\top \phi.$$

- Consider the ML estimate again; first,

$$\frac{\partial y_k}{\partial a_j} = y_k \left([k = j] - y_j \right).$$

- Let us now write the likelihood: for a 1-of-K coding scheme we have target vector $\mathbf{t}_n$ and likelihood

$$p(\mathbf{T} \mid \mathbf{w}_1, \dots, \mathbf{w}_K) = \prod_{n=1}^{N} \prod_{k=1}^{K} p(\mathcal{C}_k \mid \phi_n)^{t_{nk}} = \prod_{n=1}^{N} \prod_{k=1}^{K} y_{nk}^{t_{nk}}$$

for $y_{nk} = y_k(\phi_n)$; taking the log, we get

$$E(\mathbf{w}_1, \dots, \mathbf{w}_K) = -\ln p(\mathbf{T} \mid \mathbf{w}_1, \dots, \mathbf{w}_K) = -\sum_{n=1}^{N} \sum_{k=1}^{K} t_{nk} \ln y_{nk}, \text{ и}$$

$$\nabla_{\mathbf{w}_j} E(\mathbf{w}_1, \dots, \mathbf{w}_K) = -\sum_{n=1}^{N} \left(y_{nj} - t_{nj}\right) \phi_n.$$

- Again, we can optimize with Newton–Raphson's method; the Hessian is

$$\nabla_{\mathbf{w}_k} \nabla_{\mathbf{w}_j} E(\mathbf{w}_1, \dots, \mathbf{w}_K) = -\sum_{n=1}^{N} y_{nk} \left([k = j] - y_{nj} \right) \phi_n \phi_n^\top.$$

- Conclusion: for a classification problem it makes sense to minimize the cross-entropy $\sum_{n=1}^{N} [t_n \ln y_n + (1 - t_n) \ln(1 - y_n)]$ and softmax (rather than classification error, which is problematic).
- One question remains: how do we optimize all this?
- For logistic regression, we have IRLS and even better approaches.
- But how do we optimize complicated functions in general?

GRADIENT DESCENT

- Gradient descent: take the gradient w.r.t. weights, move in that direction.
- Formally: for an error function E, targets y, and model f with parameters θ,

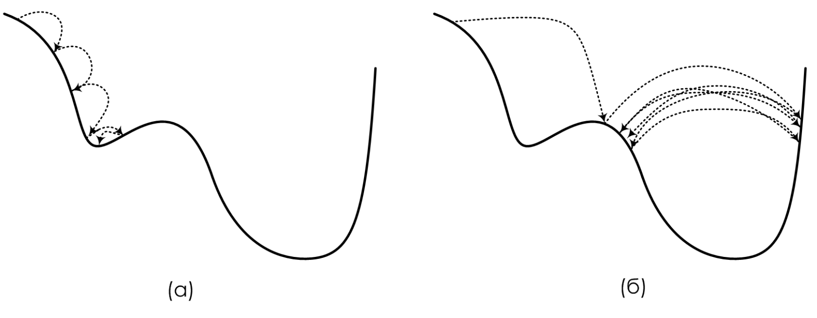
$$E(\theta) = \sum_{(\mathbf{x},y)\in D} E(f(\mathbf{x},\theta),y),$$

$$\theta_t = \theta_{t-1} - \eta\nabla E(\theta_{t-1}) = \theta_{t-1} - \eta \sum_{(\mathbf{x},y)\in D} \nabla E(f(\mathbf{x},\theta_{t-1}),y).$$

- So we need to sum over the entire dataset for every step?!..

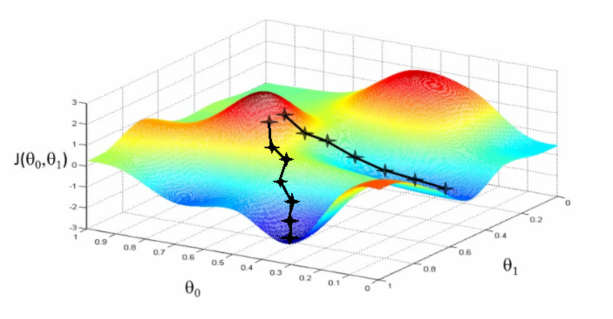
- Hence, *stochastic gradient descent*: after every training sample update
$$\theta_t = \theta_{t-1} - \eta \nabla E(f(\mathbf{x}_t, \theta_{t-1}), y_t),$$
- In practice people usually use *mini-batches*, it's easy to parallelize and smoothes out excessive "stochasticity".
- So far the only parameter is the learning rate η.

- Lots of problems with η:



- We will get to them later, for now let's concentrate on the certainly required step: the derivatives.

- Gradient descent: virtually the only way to optimize complicated non-convex functions.



- Take the gradient $\nabla E(\mathbf{w})$ w.r.t. weights, move in that direction.

- E.g., for logistic regression we can optimize

$$E(\mathbf{w}) = -\ln p(\mathbf{t} \mid \mathbf{w}) = -\sum_{n=1}^{N} \left[t_n \ln y_n + (1 - t_n) \ln(1 - y_n)\right].$$

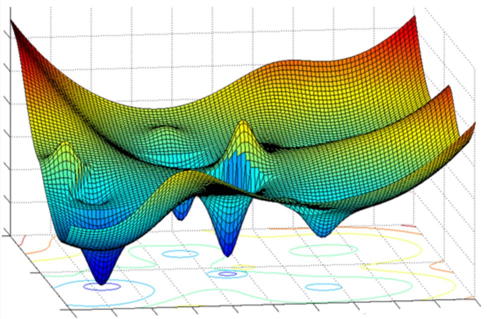
- We use the fact that $\sigma' = \sigma(1 - \sigma)$.
- Take the gradient:

$$\nabla E(\mathbf{w}) = \sum_{n=1}^{N} (y_n - t_n) \phi_n.$$

- And then we can simply use gradient descent (or do even better with IRLS).

GRADIENT DESCENT

- Gradient descent is a local optimization procedure.



- But there are no global ones... we will only talk of local improvements of gradient descent, but there will never be a guarantee with these methods.

THANK YOU!

Thank you for your attention!