6.867 Machine learning and neural networks

Tommi Jaakkola
MIT AI Lab
tommi@ai.mit.edu

Lecture 5: classification, regularization

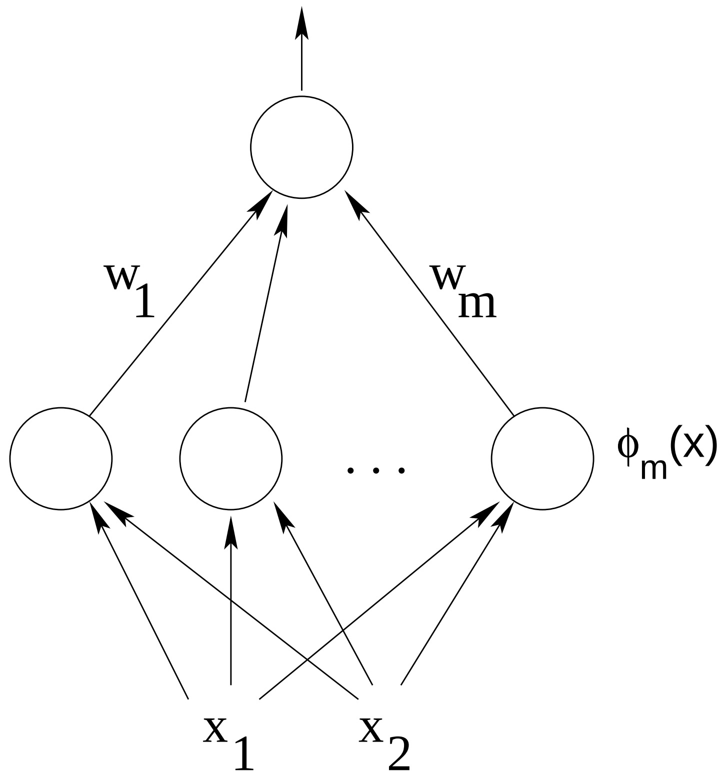
Topics

- Classification cont'd
  - Additive logistic regression
  - Neural networks
- Regularization
  - empirical loss, expected loss
  - effective number of parameters
  - prior probabilities

Additive models and classification

- Similarly to linear regression models, we can extend logistic regression models through additive models

$$P(y = 1|\mathbf{x}, \mathbf{w}) = g\,(\,w_0 + w_1\phi_1(\mathbf{x}) + \ldots w_m\phi_m(\mathbf{x})\,)$$

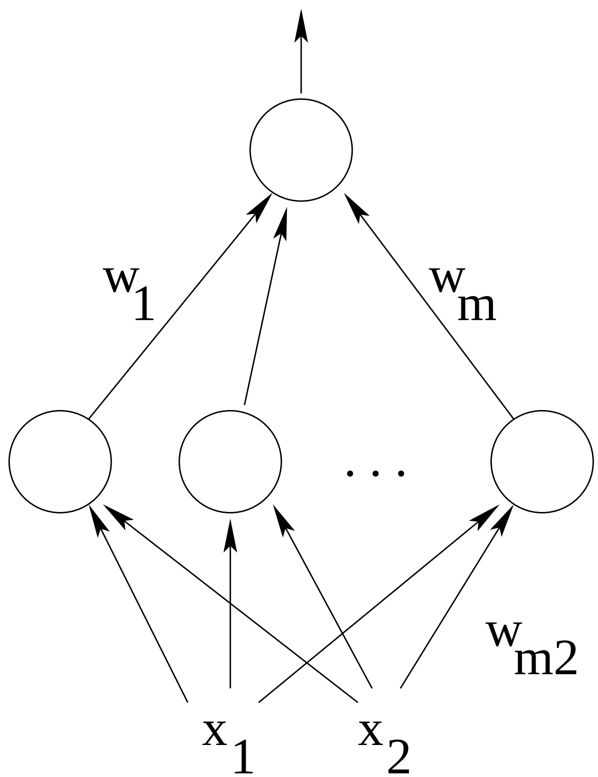


- How should we then choose the basis functions $\phi_i(\mathbf{x})$?
- One approach is to make them adjustable...

Two layer neural network model

- In a neural network model, the basis functions themselves are adjustable (e.g., squashed linear regression models)

$$P(y = 1|\mathbf{x}, \mathbf{w}) \;\; = \;\; g\,(\, w_0 + w_1\phi_1(\mathbf{x}) + \ldots w_m\phi_m(\mathbf{x})\,)$$



$$\phi_m(x) \;\; = \;\; g(\, w_{m0} + w_{m1}x_1 + w_{m2}x_2\,)$$

- We can adjust the model parameters, e.g., via stochastic gradient ascent

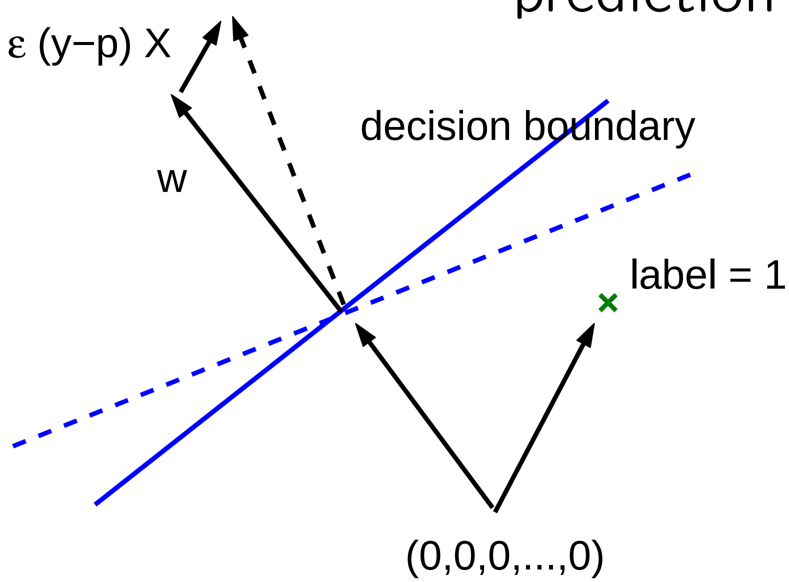
Review: stochastic gradient ascent

- For a logistic regression model with fixed basis functions

$$P(y = 1|\mathbf{x}, \mathbf{w}) = g\,(\,w_0 + w_1\phi_1(\mathbf{x}) + \ldots + w_m\phi_m(\mathbf{x})\,)$$

we get simple on-line parameter updates

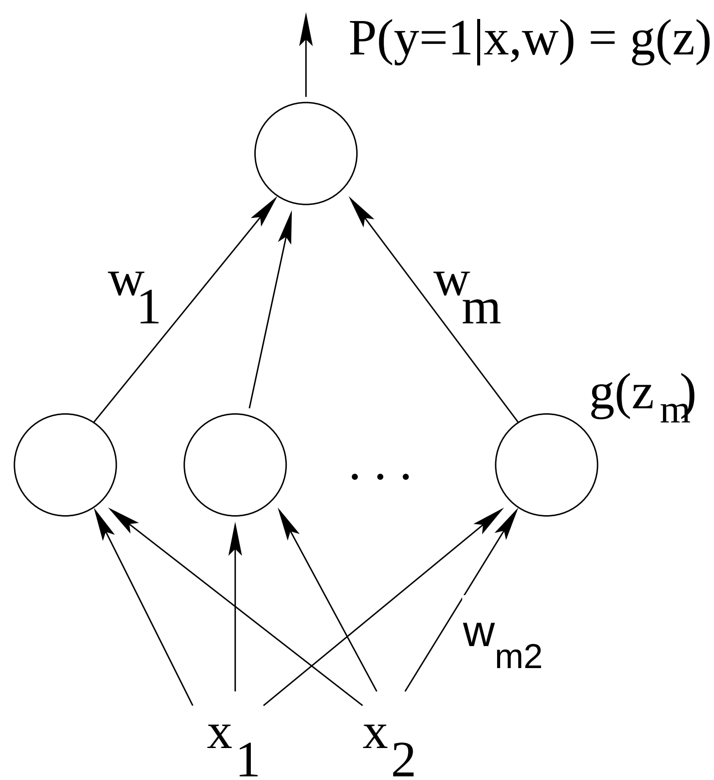
$$\begin{aligned} \mathbf{w} &\leftarrow \mathbf{w} + \epsilon \frac{\partial}{\partial \mathbf{w}} \log P(y_i|\mathbf{x}_i, \mathbf{w}) \\ &= \mathbf{w} + \epsilon \underbrace{\Big(y_i - P(y_i = 1|\mathbf{x}_i, \mathbf{w})\Big)}_{\text{prediction error}} \begin{bmatrix} 1 \\ \phi_1(\mathbf{x}) \\ \cdots \\ \phi_m(\mathbf{x}) \end{bmatrix} \end{aligned}$$



Computing the gradient: back-propagation

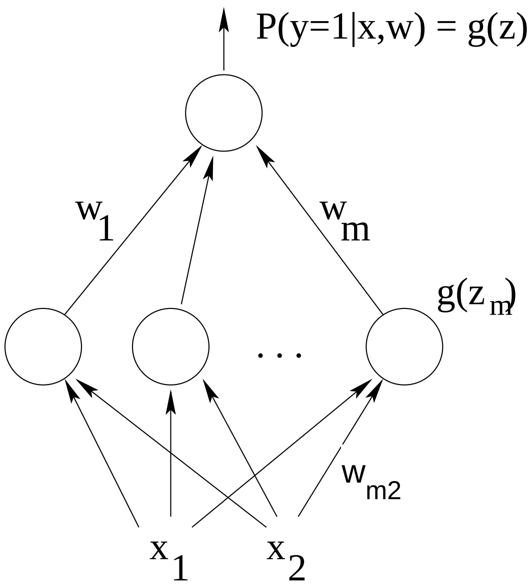
Let z, $z_i, i = 1, \ldots, m$ be the total "input" to each "neuron" computed in response to a training example $\mathbf{x}$

$$
\begin{aligned}
z &= w_0 + w_1 g(z_1) + \ldots + w_m g(z_m) \\
z_i &= w_{i0} + w_{i1} x_1 + w_{i2} x_2, \;\; i = 1, \ldots, m
\end{aligned}
$$



Back-propagation cont'd

- We can propagate the derivatives with respect to the *inputs* z



$$
\begin{aligned}
\delta &= \frac{\partial}{\partial z} \log P(y|\mathbf{x}, \mathbf{w}) \\
\delta_i &= \frac{\partial}{\partial z_i} \log P(y|\mathbf{x}, \mathbf{w}) \\
&= \frac{\partial g(z_i)}{\partial z_i} \times \frac{\partial z}{\partial g(z_i)} \times \frac{\partial}{\partial z} \log P(y|\mathbf{x}, \mathbf{w}) \\
&= g'(z_i) \times w_i \times \delta
\end{aligned}
$$

- The derivatives with respect to the weights w_{ij} are obtained from δ's

$$
\begin{aligned}
\frac{\partial}{\partial w_{ij}} \log P(y|\mathbf{x}, \mathbf{w}) &= \frac{\partial z_i}{\partial w_{ij}} \times \frac{\partial}{\partial z_i} \log P(y|\mathbf{x}, \mathbf{w}) \\
&= x_j \times \delta_i
\end{aligned}
$$

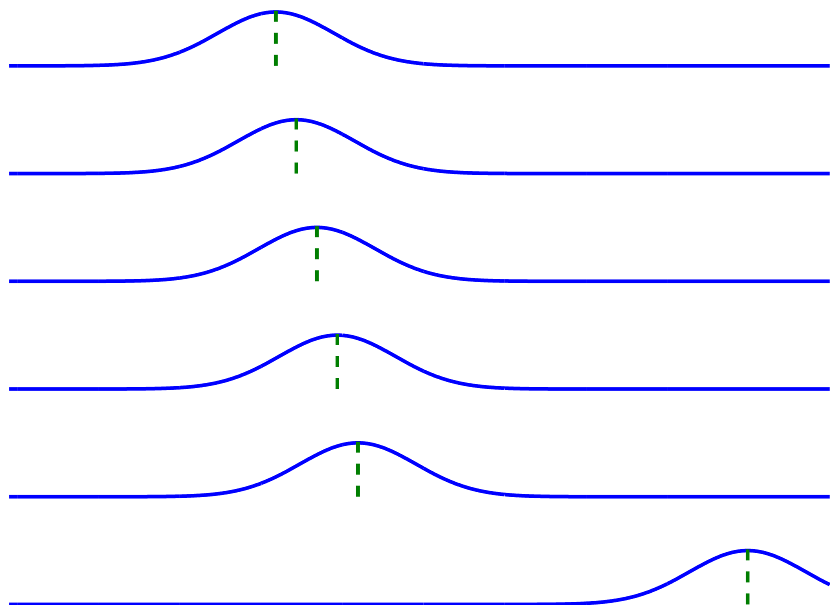
Topics

- Regularization
  - empirical loss, expected loss
  - effective number of parameters
  - prior probabilities

Empirical/expected loss

- Simple example: m parameter choices, n training examples

$$
\begin{aligned}
L_n(\mathbf{w}_1) &= \frac{1}{n}\sum_{i=1}^{n} \mathrm{Loss}(y_i, f(\mathbf{x}_i, \mathbf{w}_1)) \\
&\dots \\
L_n(\mathbf{w}_m) &= \frac{1}{n}\sum_{i=1}^{n} \mathrm{Loss}(y_i, f(\mathbf{x}_i, \mathbf{w}_m))
\end{aligned}
$$



- The empirical loss corresponding to each parameter choice is distributed around the expected loss.

Empirical/expected loss

- We'd like the empirical loss of our parameter estimate $\hat{\mathbf{w}}$ to be close to its expected value

$$\begin{aligned} L_n(\mathbf{w}_k) &= \frac{1}{n}\sum_{i=1}^{n} \mathsf{Loss}(y_i, f(\mathbf{x}_i, \mathbf{w}_k)), \quad k = 1, \ldots, m \\ L_n(\hat{\mathbf{w}}) &= \min_i \{ L_n(\mathbf{w}_i) \} \end{aligned}$$

This is a bit problematic...

Empirical/expected loss

- We'd like the empirical loss of our parameter estimate $\hat{\mathbf{w}}$ to be close to its expected value

$$L_n(\mathbf{w}_k) = \frac{1}{n}\sum_{i=1}^{n} \mathsf{Loss}(y_i, f(\mathbf{x}_i, \mathbf{w}_k)), \quad k = 1, \ldots, m$$
$$L_n(\hat{\mathbf{w}}) = \min_i \{ L_n(\mathbf{w}_i) \}$$

  This is a bit problematic...

  Suppose for simplicity that all the empirical losses corresponding to the different parameter choices are independent (in general they are not).

  Suppose further that they all have a simple Gaussian distribution around their expected losses and that the expected losses are all identical

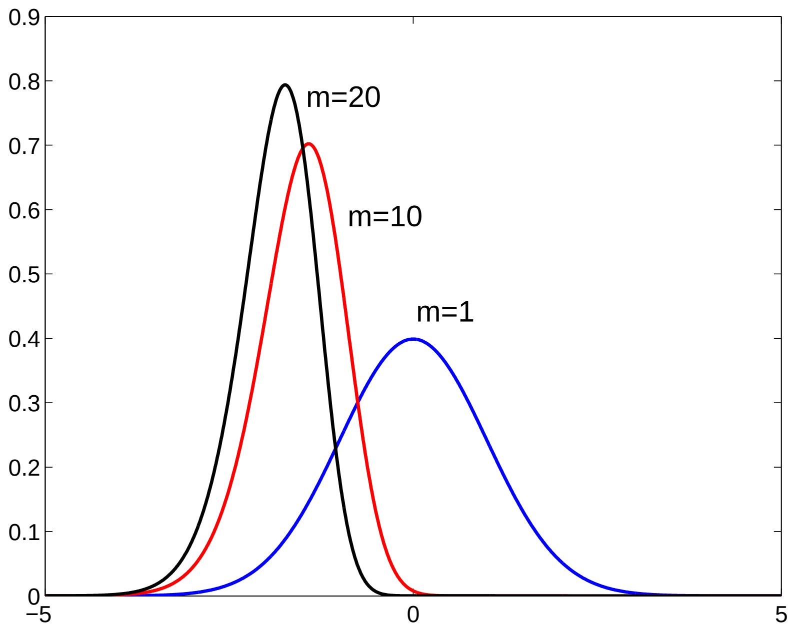
- How is $L_n(\hat{\mathbf{w}}) = \min_i \{ L_n(\mathbf{w}_i) \}$ distributed in this case?

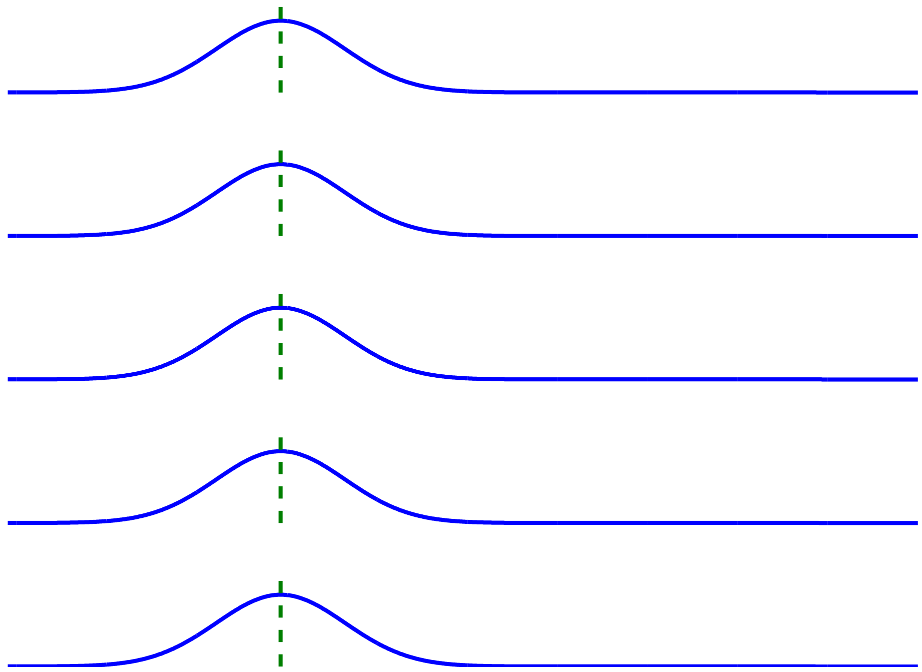
Empirical/expected loss cont'd

- How is $\min_i \{ L_n(\mathbf{w}_i) \}$ distributed in the simple case where each

$$L_n(\mathbf{w}_k) \;=\; \frac{1}{n}\sum_{i=1}^{n} \mathsf{Loss}(y_i, f(\mathbf{x}_i, \mathbf{w}_k)),$$

is a zero mean Gaussian?



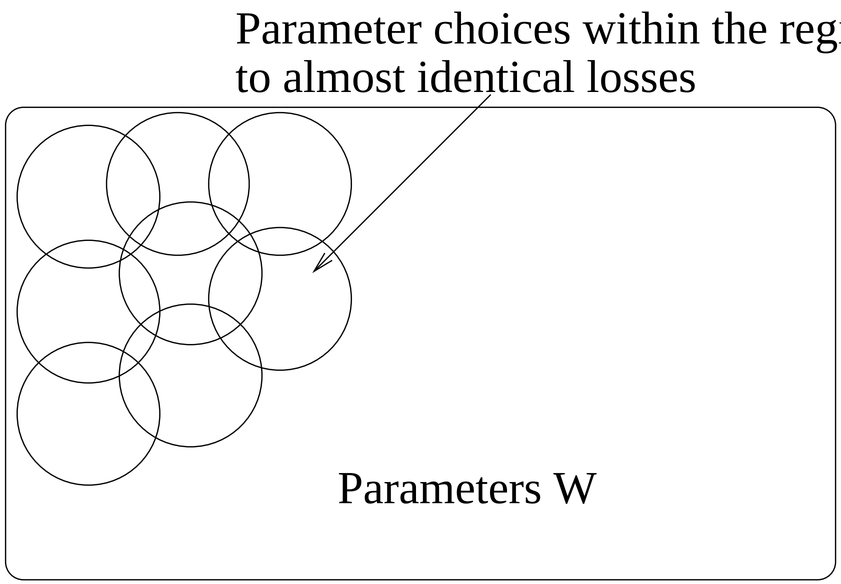


$$p_{min}(z) \;\propto\; p(L_n(\mathbf{w}_i) = z) \prod_{j \neq i} P(L_n(\mathbf{w}_j) > z)$$

Empirical/expected loss cont'd

- The parameters $\mathbf{w}$ are often continuous valued... what is m?

$$\begin{aligned} L_n(\mathbf{w}_1) &= \frac{1}{n}\sum_{i=1}^{n} \mathsf{Loss}(y_i, f(\mathbf{x}_i, \mathbf{w}_1)) \\ &\cdots \\ L_n(\mathbf{w}_m) &= \frac{1}{n}\sum_{i=1}^{n} \mathsf{Loss}(y_i, f(\mathbf{x}_i, \mathbf{w}_m)) \end{aligned}$$



- Effectively we only have a discrete number of parameter choices

Regularization

- The purpose of regularization is to improve generalization
  1. Regularization limits the effective number of parameter choices
     $\Rightarrow$ empirical loss of $\hat{\mathbf{w}}$ close to the expected loss
  2. We can also use regularization to incorporate prior knowledge
- Regularization comes in many flavors:
  1. Keep parameter values small (avoid overly strong predictions)
  2. Complexity penalties (e.g., for linear/quadratic)
  3. Feature/component/subset selection

     etc.

Regularization: example

- Logistic regression model again

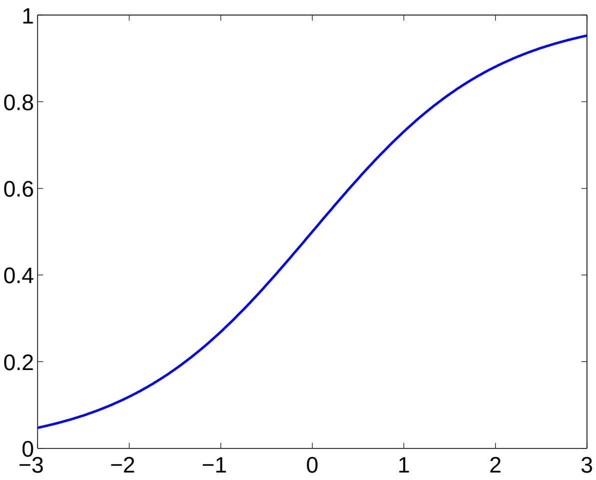
$$P(y = 1|\mathbf{x}, \mathbf{w}) = g(\, w_0 + w_1 x_1 + \ldots + w_d x_d \,)$$

- Maximum penalized likelihood (i.e., with regularization):

$$J_n(\mathbf{w}; C) = \sum_{i=1}^{n} \log P(y_i|\mathbf{x}_i, \mathbf{w}) - \frac{C}{2}\|\mathbf{w}\|^2$$

where larger values of C impose stronger regularization.

- How are we limiting our choices here?



- How can we set C?

Regularization and prior probability

- Let's assign a simple Gaussian prior probability over the parameters $\mathbf{w}$ in the logistic regression model

  $$P(\mathbf{w}) = N(\mathbf{w}; \mu, \sigma^2 I)$$

  and maximize the log-probability of the observed data **and** the parameters

  $$\begin{aligned} J_n(\mathbf{w}; C) &= \sum_{i=1}^{n} \log P(y_i|\mathbf{x}_i, \mathbf{w}) + \log P(\mathbf{w}) \\ &= \sum_{i=1}^{n} \log P(y_i|\mathbf{x}_i, \mathbf{w}) - \frac{1}{2\sigma^2} \|\mathbf{w}\|^2 + \text{const} \end{aligned}$$

  This is the same as before so long as we define $C = 1/\sigma^2$

Modified stochastic gradient ascent

- Overall objective:

$$
\begin{aligned}
J_n(\mathbf{w};C) &= \sum_{i=1}^{n} \log P(y_i|\mathbf{x}_i,\mathbf{w}) - \frac{C}{2}\|\mathbf{w}\|^2 \\
&= \sum_{i=1}^{n} \left[\log P(y_i|\mathbf{x}_i,\mathbf{w}) - \frac{C}{2n}\|\mathbf{w}\|^2\right]
\end{aligned}
$$

- For a regularized logistic regression model we still get simple on-line parameter updates

$$
\begin{aligned}
\mathbf{w} &\leftarrow \mathbf{w} + \epsilon \frac{\partial}{\partial \mathbf{w}} \left[\log P(y_i|\mathbf{x}_i,\mathbf{w}) - \frac{C}{2n}\|\mathbf{w}\|^2\right] \\
&= (\,1 - \frac{\epsilon C}{n}\,)\,\mathbf{w} + \epsilon\, \underbrace{\Big(y_i - P(y_i = 1|\mathbf{x}_i,\mathbf{w})\Big)}_{\text{prediction error}} \begin{bmatrix} 1 \\ \mathbf{x}_i \end{bmatrix}
\end{aligned}
$$