Linear Classifiers (Part 2)

CS114B Lab 3

Kenneth Lai

February 3, 2023

Training Linear Classifiers

- Naïve Bayes: estimate parameters (log-prior, log-likelihood) directly from training data

Training Linear Classifiers

- Naïve Bayes: estimate parameters (log-prior, log-likelihood) directly from training data
- Logistic regression, perceptron:
  - Define a loss function
  - Update the parameters using gradient descent

Loss Functions

- How wrong is your classifier?
- Logistic regression: cross-entropy loss
  - $L(\hat{y}, y) = -\log P(y|\mathbf{x}) = -[y \log \hat{y} + (1-y)\log(1-\hat{y})]$
    - Minimizing loss = maximizing the (log-)probability of the true y given $\mathbf{x}$

Loss Functions

- How wrong is your classifier?
- Logistic regression: cross-entropy loss
  - $L(\hat{y}, y) = -\log P(y|\mathbf{x}) = -[y \log \hat{y} + (1-y)\log(1-\hat{y})]$
    - Minimizing loss = maximizing the (log-)probability of the true y given $\mathbf{x}$
- Perceptron: perceptron loss
  - $L(\hat{y}, y) = (\hat{y} - y)z$ (for $y \in \{0, 1\}$)
    - (You may also see $L = \max(0, -yz)$, for $y \in \{-1, 1\}$)
    - If $\hat{y} = y$, $L = 0$
    - If $\hat{y} \neq y$, $L > 0$

Derivatives

- The derivative of a function measures the instantaneous rate of change in a function's output with respect to a change in its input
  - "Slope of a function's graph"
- The derivative of f with respect to x is denoted $\frac{df}{dx}$, f', etc.

Derivatives

- Rules of differentiation
  - Constant rule: If $f(x)$ is constant, then $f'(x) = 0$
  - Sum rule: $(f+g)' = f' + g'$
  - Product rule: $(fg)' = f'g + fg'$
  - Power rule: If $f(x) = x^r$, then $f'(x) = rx^{r-1}$
  - Chain rule: If $h(x) = f(g(x))$, then $h'(x) = f'(g(x)) \cdot g'(x)$ (or $\frac{dh}{dx} = \frac{df}{dg} \cdot \frac{dg}{dx}$)

Derivatives

- Rules of differentiation
  - Constant rule: If $f(x)$ is constant, then $f'(x) = 0$
  - Sum rule: $(f + g)' = f' + g'$
  - Product rule: $(fg)' = f'g + fg'$
  - Power rule: If $f(x) = x^r$, then $f'(x) = rx^{r-1}$
  - Chain rule: If $h(x) = f(g(x))$, then $h'(x) = f'(g(x)) \cdot g'(x)$ (or $\frac{dh}{dx} = \frac{df}{dg} \cdot \frac{dg}{dx}$)
- Anything more complicated than this, we will tell you what the derivative is

Partial Derivatives

- The partial derivative of a function of several variables measures the instantaneous rate of change in a function's output with respect to a change in one of its inputs, with the others held constant
- The partial derivative of f with respect to x is denoted $\dfrac{\partial f}{\partial x}$, f_x, etc.

Gradients

- The gradient of a function of several variables is a vector of partial derivatives

$$\nabla F = \begin{bmatrix} \dfrac{\partial F}{\partial x_1} \\ \vdots \\ \dfrac{\partial F}{\partial x_n} \end{bmatrix}$$

Gradient Descent

- Initialize parameters $\theta = \mathbf{w}, b$ (randomly or $\mathbf{0}$)

Gradient Descent

- Initialize parameters $\theta = \mathbf{w}, b$ (randomly or $\mathbf{0}$)
- At each time step t:
  - Compute gradient ∇L
    - $\nabla L = \begin{bmatrix} \frac{\partial L}{\partial w_1} \\ \vdots \\ \frac{\partial L}{\partial w_n} \\ \frac{\partial L}{\partial b} \end{bmatrix}$
    - $\approx$ slope of loss function

Gradient Descent

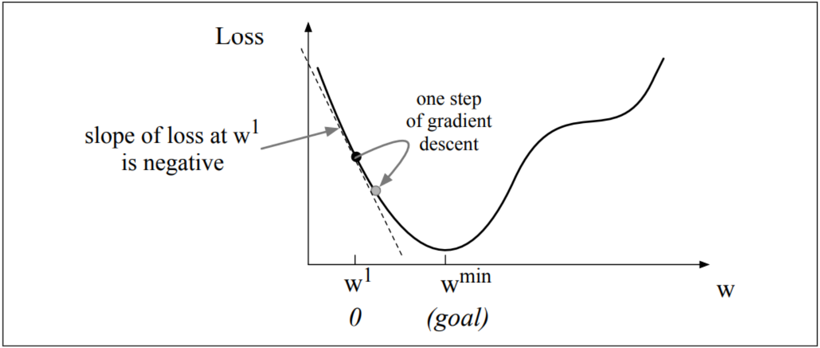

Figure 5.4 The first step in iteratively finding the minimum of this loss function, by moving w in the reverse direction from the slope of the function. Since the slope is negative, we need to move w in a positive direction, to the right. Here superscripts are used for learning steps, so w^1 means the initial value of w (which is 0), w^2 at the second step, and so on.

Gradient Descent

- Initialize parameters $\theta = \mathbf{w}, b$ (randomly or $\mathbf{0}$)
- At each time step t:
  - Compute gradient ∇L
  - Move in direction of negative gradient

Gradient Descent

- Initialize parameters $\theta = \mathbf{w}, b$ (randomly or $\mathbf{0}$)
- At each time step t:
  - Compute gradient ∇L
  - Move in direction of negative gradient
- $\theta_{t+1} = \theta_t - \eta \nabla L$
  - $\eta =$ learning rate
    - "Hyperparameter": parameter set before training
    - Trade-off between speed of convergence and "zig-zag" behavior
    - Often a function of t