CSI 436/536
Introduction to Machine Learning

Logistic regression

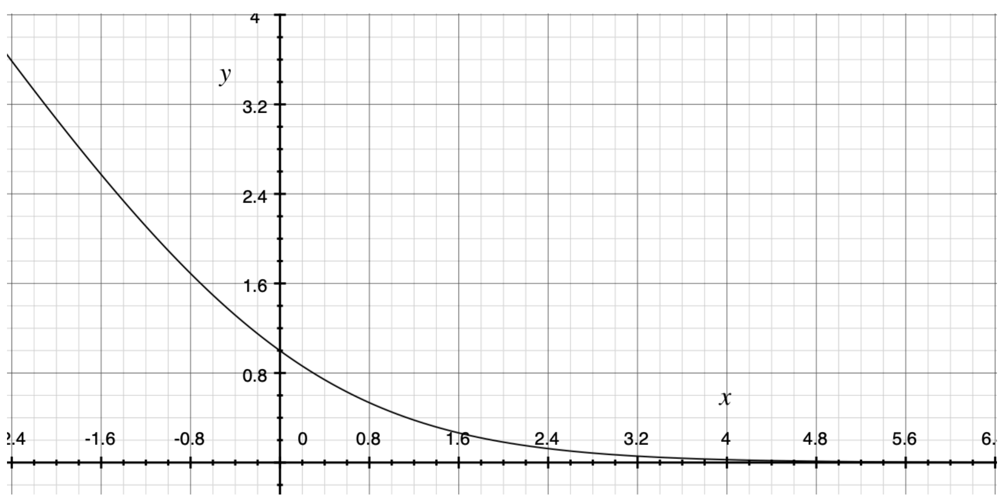
Professor Siwei Lyu
Computer Science
University at Albany, State University of New York

Logistic regression: binary classification

- Given a set of training data $(x_1, y_1), \cdots, (x_n, y_n)$, with $x_i \in \mathscr{R}^d$ and $y_i \in \{-1, +1\}$, we aim to find a ***linear*** classifier with parameter (w, b) in the form of $\hat{y} = \text{sign}(w^\top x + b)$
    - The choice of binary label is arbitrary, for any classifier outputs ± 1, we can convert it to the output of $\{0,1\}$, and vice versa: $\dfrac{y+1}{2} : \{-1, +1\} \mapsto \{0,1\}$, and $2y - 1 : \{0,1\} \mapsto \{-1, +1\}$
    - We usually use homogeneous coordinates to eliminate the constant $x \mapsto (x,1)^\top$, $w \mapsto (w, b)^\top$, and we work with $\hat{y} = \text{sign}(w^\top x)$, and we can check $yw^\top x$

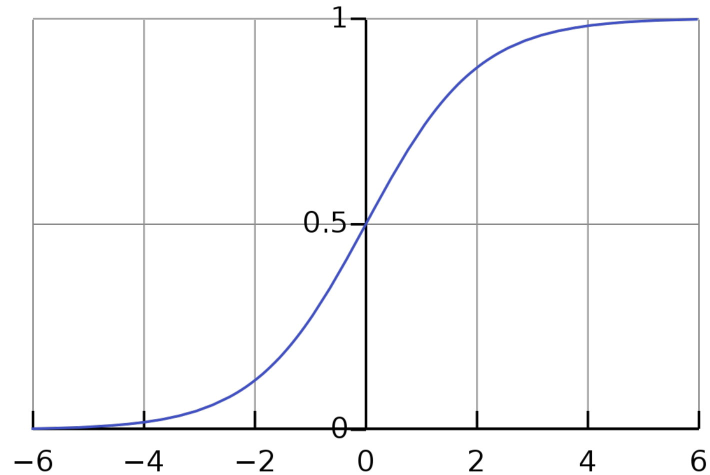
Training logistic regression

- Logistic loss function: $\min_w \sum_{i=1}^{n} \log(1 + e^{-y_i w^T x_i})$
    - Individual loss function $\ell(x, y; w) = \log(1 + e^{-y w^T x})$
        - $y w^\top x > 0$: predicted label and ground truth have the same sign, $\ell(x, y; w) \leq \log 2$
        - $y w^\top x < 0$: predicted label and ground truth have different sign, $\ell(x, y; w) \geq \log 2$
    - Logistic function $h(z) = \log(1 + e^{-z})$



Optimization

- $h(z) = \log(1 + e^{-z})$,
  - $h'(z) = -e^{-z}(1 + e^{-z})^{-1} < 0$, function decreasing
  - Define sigmoid function $\sigma(z) = (1 + e^{-z})^{-1}$



  - therefore, $h'(z) = \sigma(z) - 1$, and also $h''(z) = \sigma'(z) = (1 - \sigma(z))\sigma(z) > 0$, so this function is a convex function

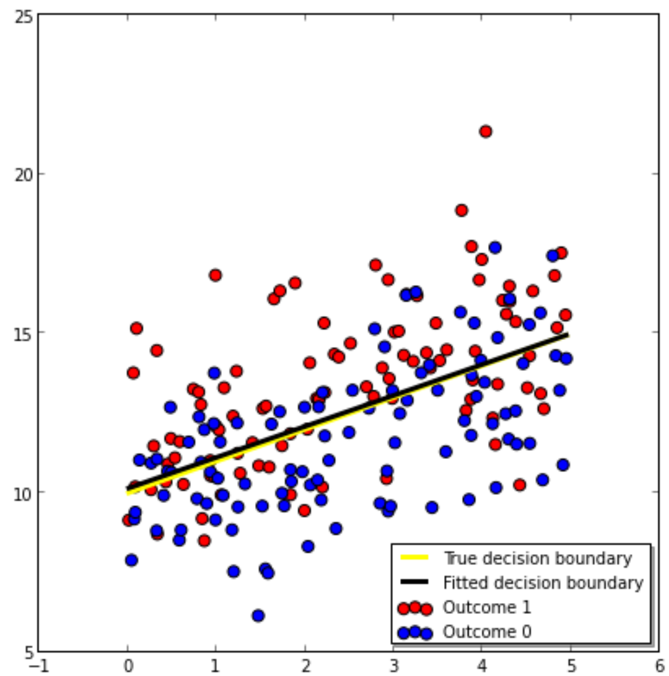
Gradient & Hessian matrix

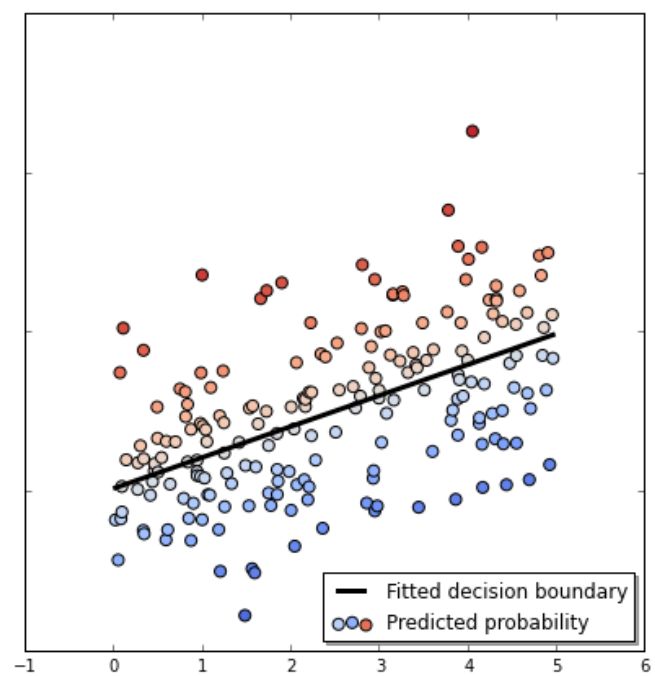
- Objective function $L(w) = \sum_{i=1}^{n} h(y_i w^T x_i)$
  - $\nabla L(w) = \sum_{i=1}^{n} h'(y_i w^T x_i) y_i x_i = \sum_{i=1}^{n} (\sigma(y_i w^T x_i) - 1) y_i x_i$
  - $\nabla^2 L(w) = \sum_{i=1}^{n} h''(y_i w^T x_i) y_i^2 x_i x_i^\top = \sum_{i=1}^{n} h''(y_i w^T x_i) x_i x_i^\top$
    $\nabla^2 L(w) = \sum_{i=1}^{n} \sigma(y_i w^T x_i)(1 - \sigma(y_i w^T x_i)) x_i x_i^\top$
    - Hessian matrix is positive definite, so the objective function is convex and affords a global optimum
- Optimization procedure
  - Gradient descent $w^{(t+1)} \leftarrow w^{(t)} - \eta_t \nabla L(w^{(t)})$
  - Newton's method
    $w^{(t+1)} \leftarrow w^{(t)} - \eta_t (\nabla^2 L(w^{(t)}))^{-1} \nabla L(w^{(t)})$
    - η_t is properly chosen step size (back-tracking)

Interpretation

- $\Pr(y=1\,|\,x)=\sigma(yw^Tx)$, i.e., probability of output label is +1 if input is x and $\Pr(y=-1\,|\,x)=1-\sigma(yw^Tx)$, i.e., probability of output label is -1 if input is x
- Cross-entropy loss

$$-\sum_i\left(\frac{1+y_i}{2}\log\Pr(y_i=1\,|\,x_i)+\frac{1-y_i}{2}\log\Pr(y_i=-1\,|\,x_i)\right)$$





Stochastic gradient method

- Gradient descent method
  - Compute gradient $\nabla L(w) = \sum_{i=1}^{n} h'(y_i w^T x_i) y_i x_i$
  - Update $w^{(t+1)} \leftarrow w^{(t)} - \eta_t \sum_{i=1}^{n} h'(y_i {w^{(t)}}^T x_i) y_i x_i$
    - η_t is properly chosen step size (back-tracking)
- Stochastic gradient method: update one data point a time $w^{(t+1)} \leftarrow w^{(t)} - \eta_t h'(y_i {w^{(t)}}^T x_i) y_i x_i$
  - It applies under the following situations
    - Dataset is too large to hold in memory
    - Streaming data, samples come one at a time

Stochastic gradient method

- Standard model is to assume data sample is selected randomly
- SG is not a descent method,
  - convergence is guaranteed under convex objective function, convergence is very slow
  - Extremely robust