Case Study 1: Estimating Click Probabilities

Intro
Logistic Regression
Gradient Descent + SGD

Machine Learning for Big Data
CSE547/STAT548, University of Washington
Sham Kakade
April 4, 2017

Announcements:

These Two WKs

- Project Proposals: due this Friday!
  – One page
- HW1 posted ~~today~~.
- (starting NEXT week) TA office hours MW 3:30 - 4:30 in CSE 406
- Readings: please do them.

- Today:
  – Review: logistic regression, GD, SGD
  – Hashing and Sketching

"in the field"

Machine Learning for ~~Big Data~~
(CSE 547 / STAT 548)

(…what is "big data" anyways?)

Ad Placement Strategies

banner

- Companies bid on ad prices

  $c_1 = \$10$ $\quad c_3 = \$100$

  $c_2 = \$20$

- Which ad wins? (many simplifications here)
  - Naively:

    $c_3 = \$100$

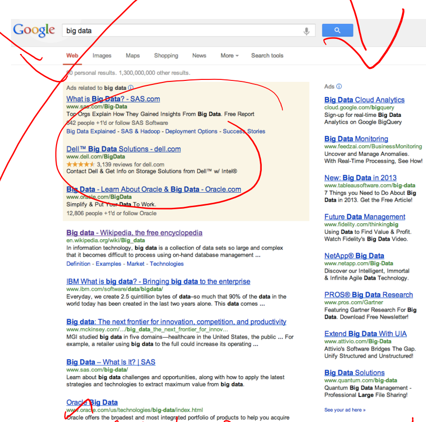
  - But:

    paid on clicks

  - Instead:

    $\hat{Pr}(\text{click} \mid c_3) = 0.01$

    $\hat{Pr}(\text{click} \mid c_2) = 0.1$



$\hat{E}[\$ \mid c_1] = 0.01 \times 100 = \$1$

$\hat{E}[\$ \mid c_2] = 0.1 \times 20 = \$2$

Learning Problem for Click Prediction

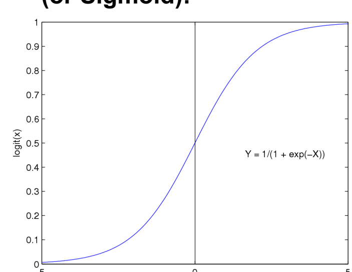
- Prediction task: $Y \in \{0,1\}$ $\quad Pr(Y=1|X)$
  Y is a click.
- Features:
  $X =$ (features of ad, features of person, keyword, person index, other context)
- Data: $\{(x^i, y^i)\}$
  - Batch:
    fixed dataset $(x^1, y^1), \ldots, (x^n, y^n)$
  - Online:
    data as a stream when user arrives at time t
    → show ad, observe Y.
- Many approaches (e.g., logistic regression, SVMs, naïve Bayes, decision trees, boosting,…)
  - Focus on logistic regression; captures main concepts, ideas generalize to other approaches

Logistic Regression

- Learn P(Y|**X**) directly
  - Assume a particular functional form
  - Sigmoid applied to a linear function of the data:

$$P(Y=0|X,W) = \frac{1}{1+exp(\underbrace{w_0 + \sum_i w_i X_i}_{z})}$$

Logistic function (or Sigmoid): $\frac{1}{1+exp(-z)}$



Features can be discrete or continuous!

Maximizing Conditional Log Likelihood

concave

not

$$l(\mathbf{w}) \equiv \ln \prod_j P(y^j|\mathbf{x}^j, \mathbf{w})$$

$$= \sum_j y^j (w_0 + \sum_{i=1}^d w_i x_i^j) - \ln \left(1 + \exp(w_0 + \sum_{i=1}^d w_i x_i^j)\right)$$

iterative methods

Good news: $l(\mathbf{w})$ is concave function of $\mathbf{w}$, no local optima problems

Bad news: no closed-form solution to maximize $l(\mathbf{w})$

Good news: concave functions easy to optimize

Gradient Ascent for LR

Gradient ascent algorithm: iterate until change < ε

$$w_0^{(t+1)} \leftarrow w_0^{(t)} + \eta \sum_j [y^j - \hat{P}(Y^j = 1 \mid \mathbf{x}^j, \mathbf{w}^{(t)})]$$

For i = 1,…,d,

$$w_i^{(t+1)} \leftarrow w_i^{(t)} + \eta \sum_j x_i^j [y^j - \hat{P}(Y^j = 1 \mid \mathbf{x}^j, \mathbf{w}^{(t)})]$$

$$\vec{w} \leftarrow \vec{w} + \eta \sum_j \vec{x}^j (y^j - \hat{P}^j)$$

repeat

Regularized Conditional Log Likelihood

- If data are linearly separable, weights go to infinity
- Leads to overfitting → Penalize large weights

$\sum_{j=1}^{d} w_j^2$

- Add regularization penalty, e.g., L_2:

$$\ell(\mathbf{w}) = \ln \prod_j P(y^j | \mathbf{x}^j, \mathbf{w})) - \frac{\lambda}{2}||\mathbf{w}||_2^2$$

- Practical note about w_0: don't regularize offset

Standard v. Regularized Updates

- Maximum conditional likelihood estimate

$$\mathbf{w}^* = \arg\max_{\mathbf{w}} \ln \left[\prod_{j=1}^{N} P(y^j \mid \mathbf{x}^j, \mathbf{w}) \right]$$

$$\boxed{w_i^{(t+1)} \leftarrow w_i^{(t)} + \eta \sum_j x_i^j [y^j - \hat{P}(Y^j = 1 \mid \mathbf{x}^j, \mathbf{w}^{(t)})]}$$

- Regularized maximum conditional likelihood estimate

$$\mathbf{w}^* = \arg\max_{\mathbf{w}} \ln \left[\prod_j P(y^j | \mathbf{x}^j, \mathbf{w})) \right] - \frac{\lambda}{2} \sum_{i>0} w_i^2$$

$$\boxed{w_i^{(t+1)} \leftarrow w_i^{(t)} + \eta \left\{ -\lambda w_i^{(t)} + \sum_j x_i^j [y^j - \hat{P}(Y^j = 1 \mid \mathbf{x}^j, \mathbf{w}^{(t)})] \right\}}$$

Stopping criterion

$\swarrow \gamma \geq \lambda$

$$\ell(\mathbf{w}) = \ln \prod_j P(y^j|\mathbf{x}^j, \mathbf{w})) - \frac{\lambda}{2}||\mathbf{w}||_2^2$$

- Regularized logistic regression is strongly concave γ-
  - Negative second derivative bounded away from zero:

$f(x)$ strongly concave (and f'' exists) $\longleftrightarrow$ $-f''(x) \geq \gamma$ for $\gamma > 0$

- Strong concavity (convexity) is super helpful!!
- For example, for strongly concave $l(\mathbf{w})$:

$$\ell(\mathbf{w}^*) - \ell(\mathbf{w}) \leq \frac{1}{2\lambda}||\nabla \ell(\mathbf{w})||_2^2$$

for λ-strongly concave

Convergence rates for gradient descent/ascent

- Number of iterations to get to accuracy

$$\ell(\mathbf{w}^*) - \ell(\mathbf{w}) \leq \epsilon$$

- If func l(w) Lipschitz: $O(1/\epsilon^2)$

$|\ell(w) - \ell(w')| \leq L|w - w'|$

- If gradient of func Lipschitz: $O(1/\epsilon)$

smooth $|\nabla\ell(w) - \nabla\ell(w')| \leq L|w - w'|$

- If func is strongly convex: $O(\ln(1/\epsilon))$

+ smooth

Challenge 1: Complexity of computing gradients

$O(d)$

- What's the cost of a gradient update step for LR???

$$w_i^{(t+1)} \leftarrow w_i^{(t)} + \eta \left\{ -\lambda w_i^{(t)} + \sum_{j=1}^{N} x_i^j [y^j - \hat{P}(Y^j = 1 \mid \mathbf{x}^j, \mathbf{w}^{(t)})] \right\}$$

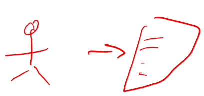
update one coordinate $O(Nd)$ comp.

Naively, complexity is $O(Nd^2)$

with "caching" $O(Nd)$

Challenge 2: Data is streaming

- Assumption thus far: **Batch data**
- But, click prediction is a streaming data task:
  - User enters query, and ad must be selected:
    - Observe $\mathbf{x}^j$, and must predict y^j



$\rightarrow$ x^j $\rightarrow$ predict $\hat{y}^j$ $\rightarrow$ observe y^j

  - User either clicks or doesn't click on ad:
    - Label y^j is revealed afterwards
      - Google gets a reward if user clicks on ad
  - Weights must be updated for next time:

Learning Problems as Expectations

- Minimizing loss in training data:
  - Given dataset:
    - Sampled iid from some distribution p(**x**) on features:
  - Loss function, e.g., hinge loss, logistic loss,…
  - We often minimize loss in training data:

$$\underbrace{\ell_{\mathcal{D}}(\mathbf{w}) = \frac{1}{N}\sum_{j=1}^{N}\ell(\mathbf{w},\mathbf{x}^j)}$$

some loss function

- However, we should really minimize expected loss on all data:

$$\ell(\mathbf{w}) = E_{\mathbf{x}}\left[\ell(\mathbf{w},\mathbf{x})\right] = \int p(\mathbf{x})\ell(\mathbf{w},\mathbf{x})d\mathbf{x}$$

- So, we are approximating the integral by the average on the training data

Gradient Ascent in Terms of Expectations

- "True" objective function:

$$\ell(\mathbf{w}) = E_{\mathbf{x}}\left[\ell(\mathbf{w},\mathbf{x})\right] = \int p(\mathbf{x})\ell(\mathbf{w},\mathbf{x})d\mathbf{x}$$

- Taking the gradient:

$$\nabla \ell(w) = E_x\left[\nabla \ell(w,x)\right]$$

- "True" gradient ascent rule:

$$w \leftarrow w + \eta \nabla \ell(w)$$

- How do we estimate expected gradient?

SGD: Stochastic Gradient Ascent (or Descent)

- "True" gradient: $\nabla \ell(\mathbf{w}) = E_{\mathbf{x}} \left[\nabla \ell(\mathbf{w}, \mathbf{x}) \right]$
- Sample based approximation:

$x \sim Dist$

$$\widehat{\nabla \ell(w)} = \nabla \ell(w, x)$$

or $\widehat{\nabla \ell(w)} = \frac{1}{n} \sum_{i=1}^{n} \nabla \ell(w, x_i)$

- What if we estimate gradient with just one sample???
  - Unbiased estimate of gradient
  - Very noisy!
  - Called stochastic gradient ascent (or descent)
    - Among many other names
  - VERY useful in practice!!!

Stochastic Gradient Ascent: General Case

- Given a stochastic function of parameters:
  - Want to find maximum

can be done in streaming model ✓

- Start from $\mathbf{w}^{(0)}$
- Repeat until convergence:
  - Get a sample data point $\mathbf{x}^t$

$O(d)$

  - Update parameters:

$$w^{t+1} \leftarrow w^t + \eta^t \nabla \ell(w^t, x^t)$$

- Works in the online learning setting!
- Complexity of each gradient step is constant in number of examples!
- In general, step size changes with iterations

Stochastic Gradient Ascent for Logistic Regression

- Logistic loss as a stochastic function:

$$E_{\mathbf{x}}\left[\ell(\mathbf{w},\mathbf{x})\right] = E_{\mathbf{x}}\left[\ln P(y|\mathbf{x},\mathbf{w}) - \frac{\lambda}{2}||\mathbf{w}||_2^2\right]$$

- Batch gradient ascent updates:

$$\boxed{w_i^{(t+1)} \leftarrow w_i^{(t)} + \eta\left\{-\lambda w_i^{(t)} + \frac{1}{N}\sum_{j=1}^{N} x_i^{(j)}[y^{(j)} - P(Y=1|\mathbf{x}^{(j)},\mathbf{w}^{(t)})]\right\}}$$

- Stochastic gradient ascent updates:
  - Online setting: have

to turn η down

$$\boxed{w_i^{(t+1)} \leftarrow w_i^{(t)} + \eta_t\left\{-\lambda w_i^{(t)} + x_i^{(t)}[y^{(t)} - P(Y=1|\mathbf{x}^{(t)},\mathbf{w}^{(t)})]\right\}}$$

over time

Convergence Rate of SGD

- **Theorem**:
  - (see CSE546 notes and readings)
  - Let *f* be a strongly convex stochastic function
  - Assume ~~gradient of~~ ~~f~~ ℓ is Lipschitz continuous

$$|\ell(w) - \ell(w')| \le L|w - w'|$$

  - Then, for step sizes:

$$\eta_t = O\left(\frac{1}{\sqrt{t}}\right) \qquad \left(\text{so } O\left(\frac{1}{\epsilon^2}\right) \text{ rate}\right)$$

  - The expected loss decreases as O(1/t^0.5):

$$E[\ell(w_t) - \ell(w^*)] \le \frac{||w_0 - w^*|| \, L}{\sqrt{t}}$$

Convergence Rates for Gradient Descent/Ascent vs. SGD

- Number of Iterations to get to accuracy

$$\ell(\mathbf{w}^*) - \ell(\mathbf{w}) \leq \epsilon$$

- Gradient descent:
  - If func is strongly convex: O(ln(1/ϵ)) iterations
- Stochastic gradient descent:
  - If func is strongly convex: O(1/ϵ) iterations
- Seems exponentially worse, but much more subtle:
  - Total running time, e.g., for logistic regression:
    - Gradient descent:
    - SGD:
    - SGD can win when we have a lot of data
  - See readings for more details

What you should know about Logistic Regression (LR) and Click Prediction

- Click prediction problem:
  - Estimate probability of clicking
  - Can be modeled as logistic regression
- Logistic regression model: Linear model
- Gradient ascent to optimize conditional likelihood
- Overfitting + regularization
- Regularized optimization
  - Convergence rates and stopping criterion
- Stochastic gradient ascent for large/streaming data
  - Convergence rates of SGD