Case Study 1: Estimating Click Probabilities

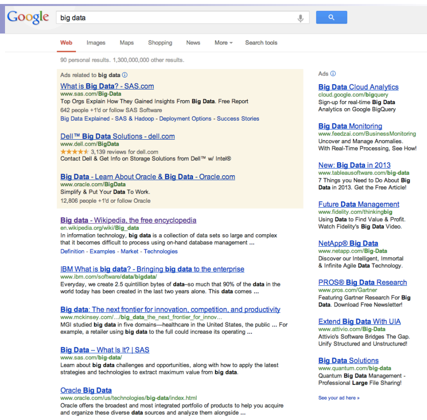
Intro
Logistic Regression
Gradient Descent + SGD
AdaGrad

Machine Learning for Big Data
CSE547/STAT548, University of Washington
Emily Fox
January 7th, 2014

Ad Placement Strategies

- Companies bid on ad prices

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



90 personal results. 1,300,000,000 other results.

Ads related to **big data**

What is **Big Data**? - SAS.com
www.sas.com/**Big-Data**
Top Orgs Explain How They Gained Insights From **Big Data**. Free Report
642 people +1'd or follow SAS Software
Big Data Explained - SAS & Hadoop - Deployment Options - Success Stories

Dell™ **Big Data** Solutions - dell.com
www.dell.com/**BigData**
★★★★★ 3,139 reviews for dell.com
Contact Dell & Get Info on Storage Solutions from Dell™ w/ Intel®

Big Data - Learn About Oracle & **Big Data** - Oracle.com
www.oracle.com/**BigData**
Simplify & Put Your **Data** To Work.
12,806 people +1'd or follow Oracle

Big data - Wikipedia, the free encyclopedia
en.wikipedia.org/wiki/**Big_data**
In information technology, **big data** is a collection of data sets so large and complex that it becomes difficult to process using on-hand database management ...
Definition - Examples - Market - Technologies

IBM What is **big data**? - Bringing **big data** to the enterprise
www.ibm.com/software/**data/bigdata**/
Everyday, we create 2.5 quintillion bytes of **data**–so much that 90% of the **data** in the world today has been created in the last two years alone. This **data** comes ...

Big data: The next frontier for innovation, competition, and productivity
www.mckinsey.com/.../**big_data**_the_next_frontier_for_innov...
MGI studied **big data** in five domains—healthcare in the United States, the public ... For example, a retailer using **big data** to the full could increase its operating ...

Big Data – What Is It? | SAS
www.sas.com/**big-data**/
Learn about **big data** challenges and opportunities, along with how to apply the latest strategies and technologies to extract maximum value from **big data**.

Oracle **Big Data**
www.oracle.com/us/technologies/**big-data**/index.html
Oracle offers the broadest and most integrated portfolio of products to help you acquire and organize these diverse **data** sources and analyze them alongside ...

Ads

Big Data Cloud Analytics
cloud.google.com/**big**query
Sign-up for real-time **Big Data** Analytics on Google BigQuery

Big Data Monitoring
www.feedzai.com/BusinessMonitoring
Uncover and Manage Anomalies. With Real-Time Processing, See How!

New: **Big Data** in 2013
www.tableausoftware.com/**big-data**
7 Things you Need to Do About **Big Data** in 2013. Get the Free Article!

Future **Data** Management
www.fidelity.com/thinking**big**
Using **Data** to Find Value & Profit. Watch Fidelity's **Big Data** Video.

NetApp® **Big Data**
www.netapp.com/**Big-Data**
Discover our Intelligent, Immortal & Infinite Agile **Data** Technology.

PROS® **Big Data** Research
www.pros.com/Gartner
Featuring Gartner Research For **Big Data**. Download Free Newsletter!

Extend **Big Data** With UIA
www.attivio.com/**Big-Data**
Attivio's Software Bridges The Gap. Unify Structured and Unstructured!

Big Data Solutions
www.quantum.com/**big-data**
Quantum **Big Data** Management - Professional **Large** File Sharing!

See your ad here »

Key Task: Estimating Click Probabilities

- What is the probability that user *i* will click on ad *j*

- Not important just for ads:
  - □ Optimize search results
  - □ Suggest news articles
  - □ Recommend products

- Methods much more general, useful for:
  - □ Classification
  - □ Regression
  - □ Density estimation

Learning Problem for Click Prediction

- Prediction task:

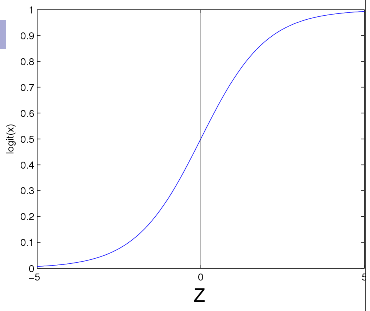
- Features:

- Data:
  - □ Batch:
  - □ Online:

- 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

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



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

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

Features can be discrete or continuous!

Very convenient!

$$P(Y=0\ |X=<X_1,...X_n>) = \frac{1}{1+exp(w_0+\sum_i w_i X_i)}$$

linear classification rule!

implies

$$\ln \frac{P(Y=1\ |X)}{P(Y=0\ |X)} = w_0 + \sum_i w_i X_i$$

Digression: Logistic regression more generally

- Logistic regression in more general case, where *Y in* {y_1,…,y_R}

  for *k<R*

  $$P(Y = y_k|X) = \frac{\exp(w_{k0} + \sum_{i=1}^{n} w_{ki}X_i)}{1 + \sum_{j=1}^{R-1} \exp(w_{j0} + \sum_{i=1}^{n} w_{ji}X_i)}$$

  for *k=R* (normalization, so no weights for this class)

  $$P(Y = y_R|X) = \frac{1}{1 + \sum_{j=1}^{R-1} \exp(w_{j0} + \sum_{i=1}^{n} w_{ji}X_i)}$$

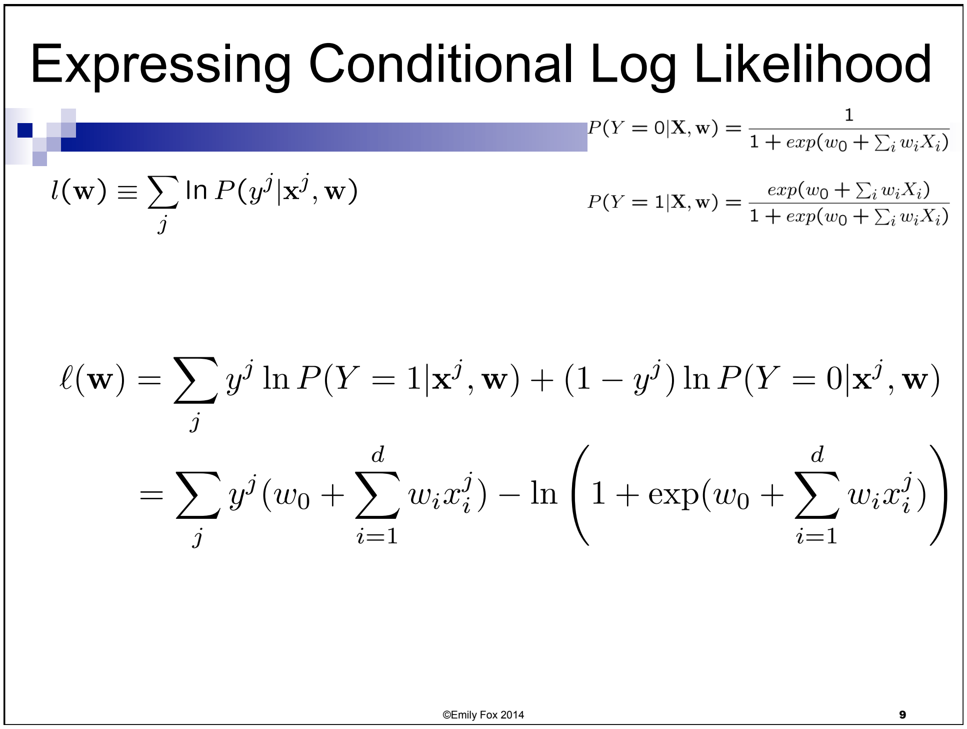
Features can be discrete or continuous!

Loss function: Conditional Likelihood

- Have a bunch of iid data of the form:

- Discriminative (logistic regression) loss function:
  Conditional Data Likelihood

$$\ln P(\mathcal{D}_Y \mid \mathcal{D}_{\mathbf{X}}, \mathbf{w}) = \sum_{j=1}^{N} \ln P(y^j \mid \mathbf{x}^j, \mathbf{w})$$



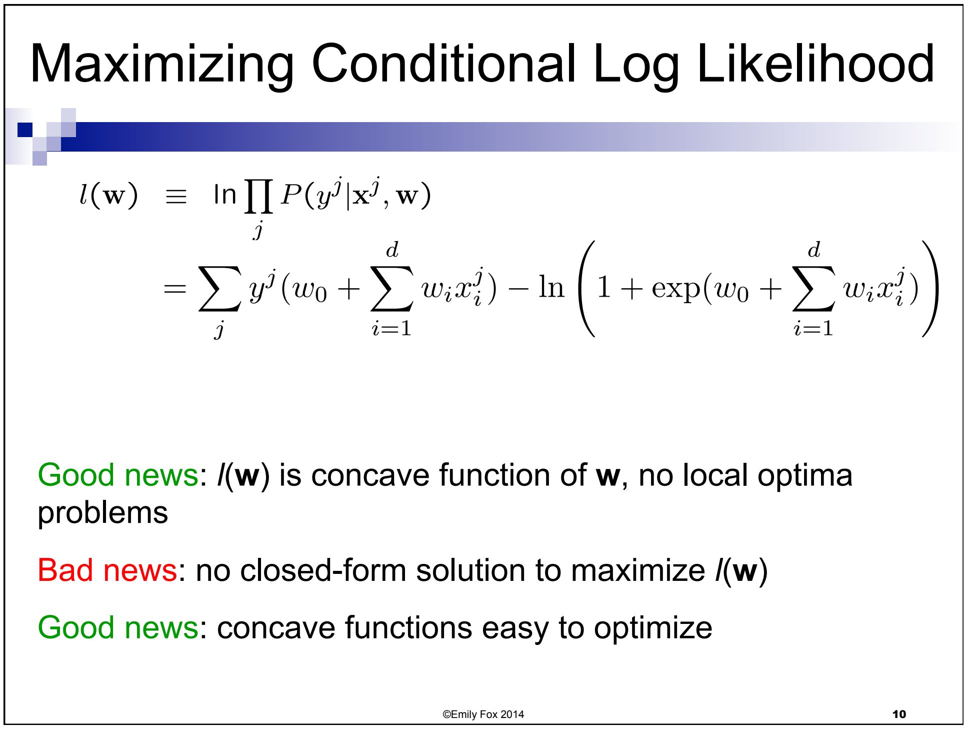
Expressing Conditional Log Likelihood

$$P(Y=0|\mathbf{X},\mathbf{w}) = \frac{1}{1+exp(w_0+\sum_i w_i X_i)}$$

$$P(Y=1|\mathbf{X},\mathbf{w}) = \frac{exp(w_0+\sum_i w_i X_i)}{1+exp(w_0+\sum_i w_i X_i)}$$

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

$$\begin{aligned}\ell(\mathbf{w}) &= \sum_j y^j \ln P(Y=1|\mathbf{x}^j,\mathbf{w}) + (1-y^j)\ln P(Y=0|\mathbf{x}^j,\mathbf{w}) \\ &= \sum_j y^j (w_0 + \sum_{i=1}^d w_i x_i^j) - \ln\left(1+\exp(w_0+\sum_{i=1}^d w_i x_i^j)\right)\end{aligned}$$



Maximizing Conditional Log Likelihood

$$\begin{aligned}l(\mathbf{w}) &\equiv \ln \prod_j P(y^j|\mathbf{x}^j,\mathbf{w}) \\ &= \sum_j y^j (w_0 + \sum_{i=1}^d w_i x_i^j) - \ln\left(1+\exp(w_0+\sum_{i=1}^d w_i x_i^j)\right)\end{aligned}$$

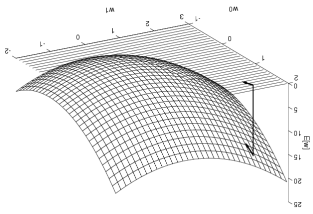
Good news: *l*(**w**) is concave function of **w**, no local optima problems

Bad news: no closed-form solution to maximize *l*(**w**)

Good news: concave functions easy to optimize

Optimizing concave function – Gradient ascent

- Conditional likelihood for Logistic Regression is concave. Find optimum with gradient ascent



Gradient: $\nabla_{\mathbf{w}} l(\mathbf{w}) = [\frac{\partial l(\mathbf{w})}{\partial w_0}, \ldots, \frac{\partial l(\mathbf{w})}{\partial w_n}]'$

Step size, η>0

Update rule: $\Delta \mathbf{w} = \eta \nabla_{\mathbf{w}} l(\mathbf{w})$

$$w_i^{(t+1)} \leftarrow w_i^{(t)} + \eta \frac{\partial l(\mathbf{w})}{\partial w_i}$$

- Gradient ascent is simplest of optimization approaches
  - e.g., Conjugate gradient ascent much better (see reading)

Gradient Ascent for LR

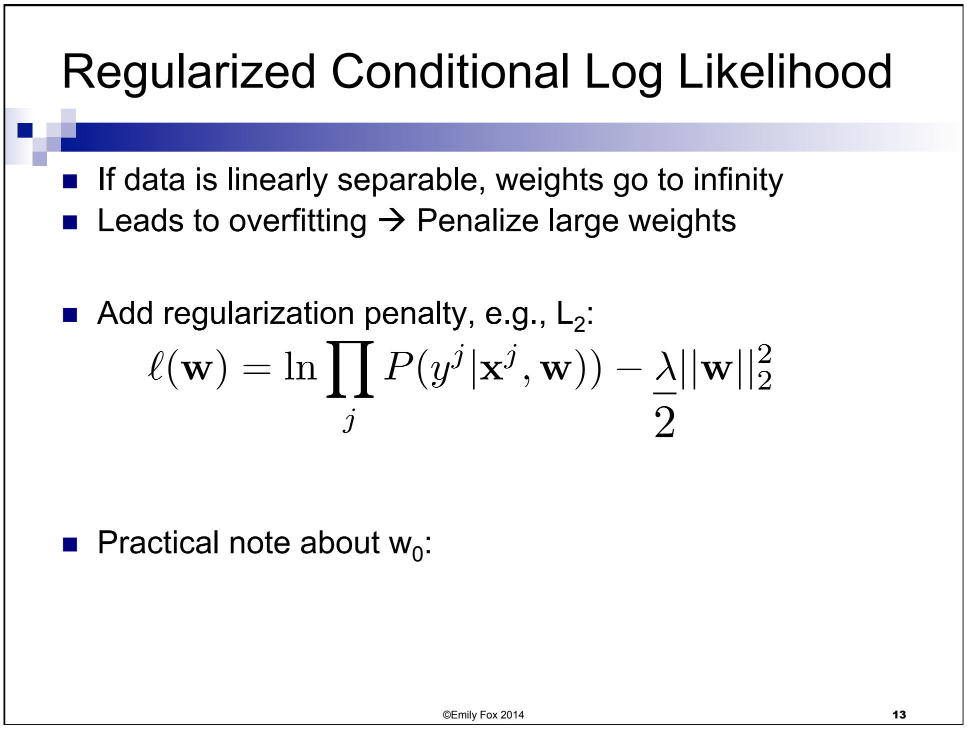
Gradient ascent algorithm: iterate until change < ε

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

For *i* = 1,…,*d*,

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

repeat



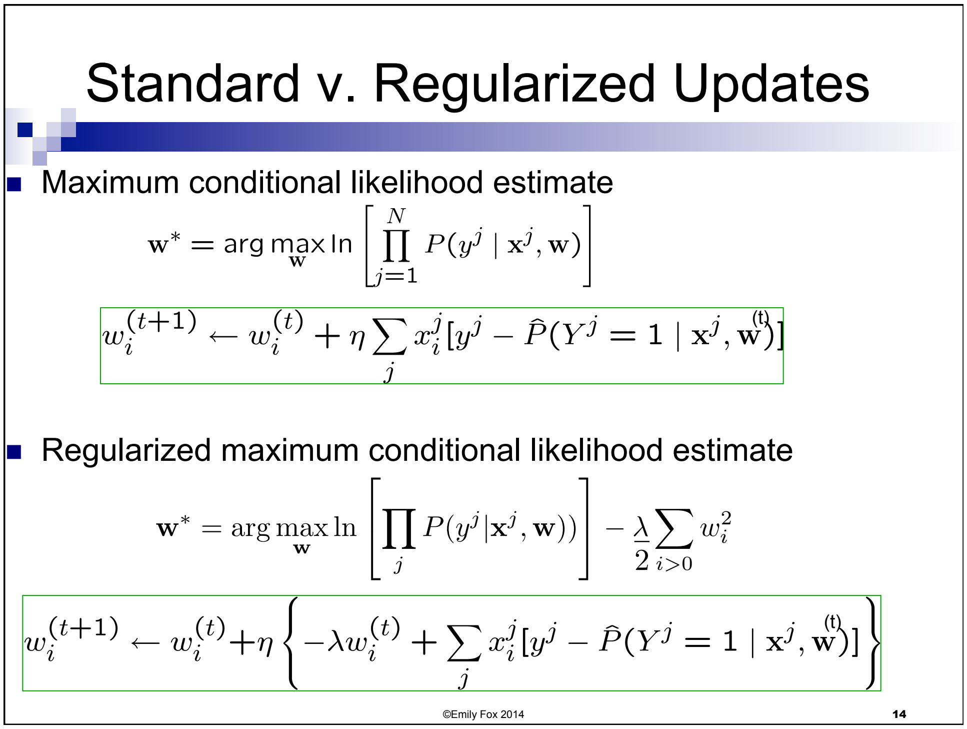
Regularized Conditional Log Likelihood

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

- 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:



Standard v. Regularized Updates

- Maximum conditional likelihood estimate

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

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

- Regularized maximum conditional likelihood estimate

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

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

Stopping criterion

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

- Regularized logistic regression is strongly concave
  - Negative second derivative bounded away from zero:
- Strong concavity (convexity) is super helpful!!
- For example, for strongly concave *l*(**w**):

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

Convergence rates for gradient descent/ascent

- Number of Iterations to get to accuracy

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

- If func Lipschitz: O(1/ϵ^2)
- If gradient of func Lipschitz: O(1/ϵ)
- If func is strongly convex: O(ln(1/ϵ))

Challenge 1: Complexity of computing gradients

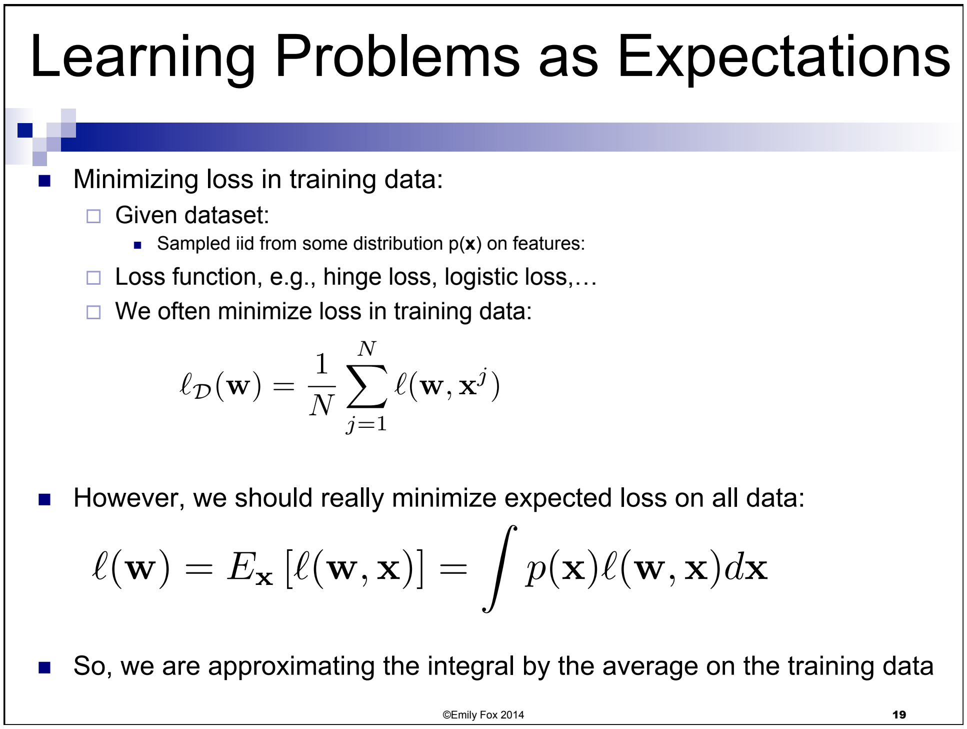
- What's the cost of a gradient update step for LR???

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

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
  - □ 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

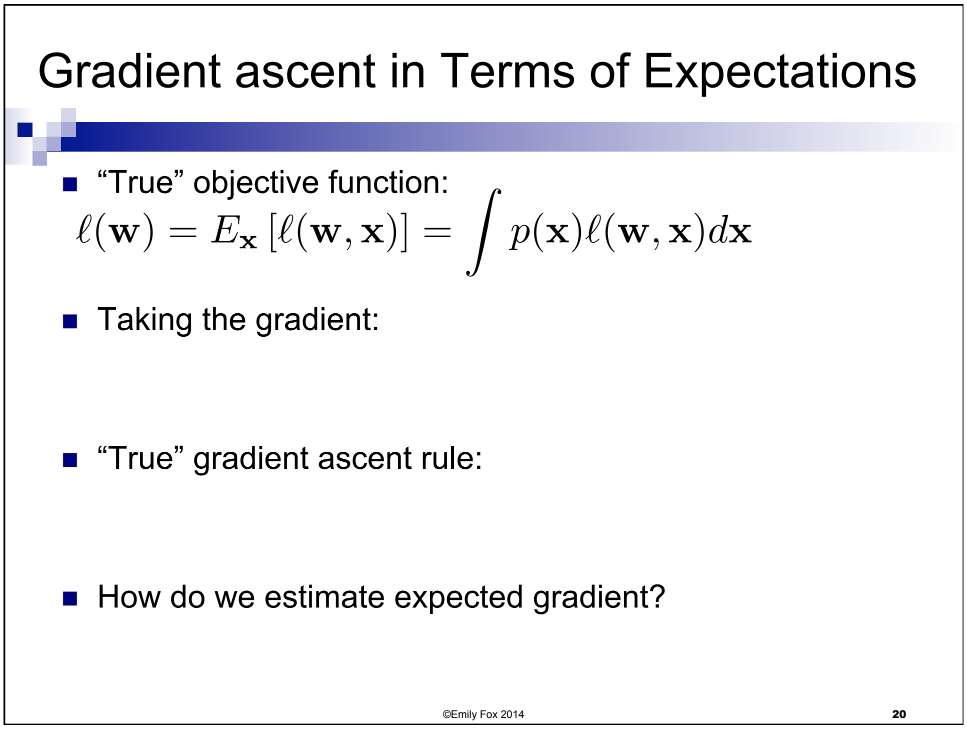
- Minimizing loss in training data:
  - Given dataset:
    - Sampled iid from some distribution p(**x**) on features:
  - Loss function, e.g., hinge loss, logistic loss,…
  - We often minimize loss in training data:

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

- 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

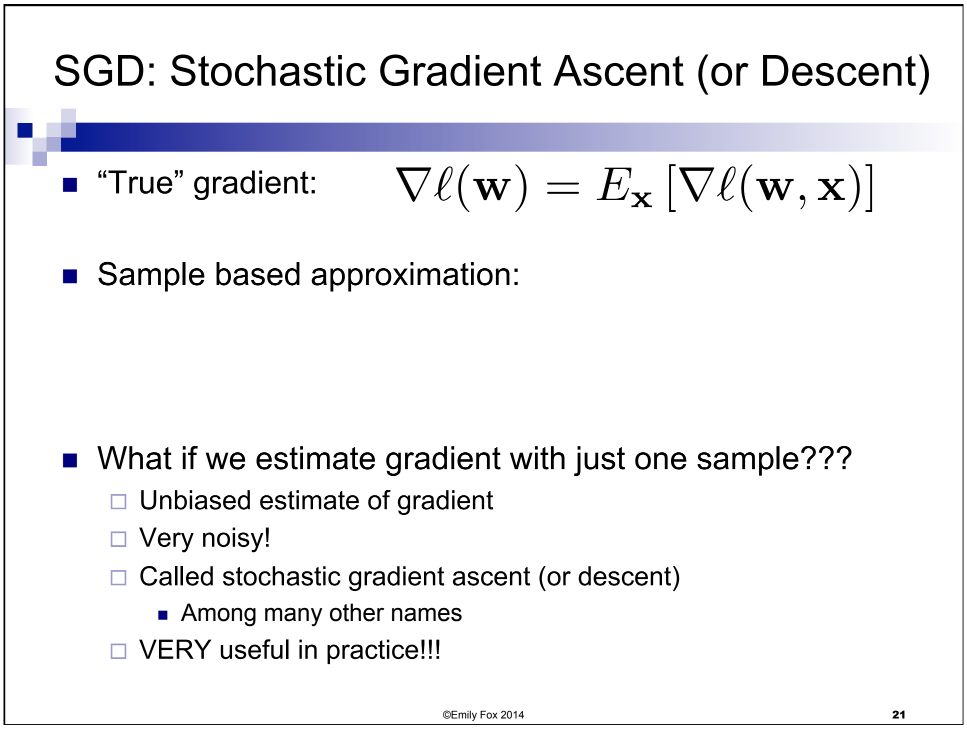
- "True" objective function:

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

- Taking the gradient:

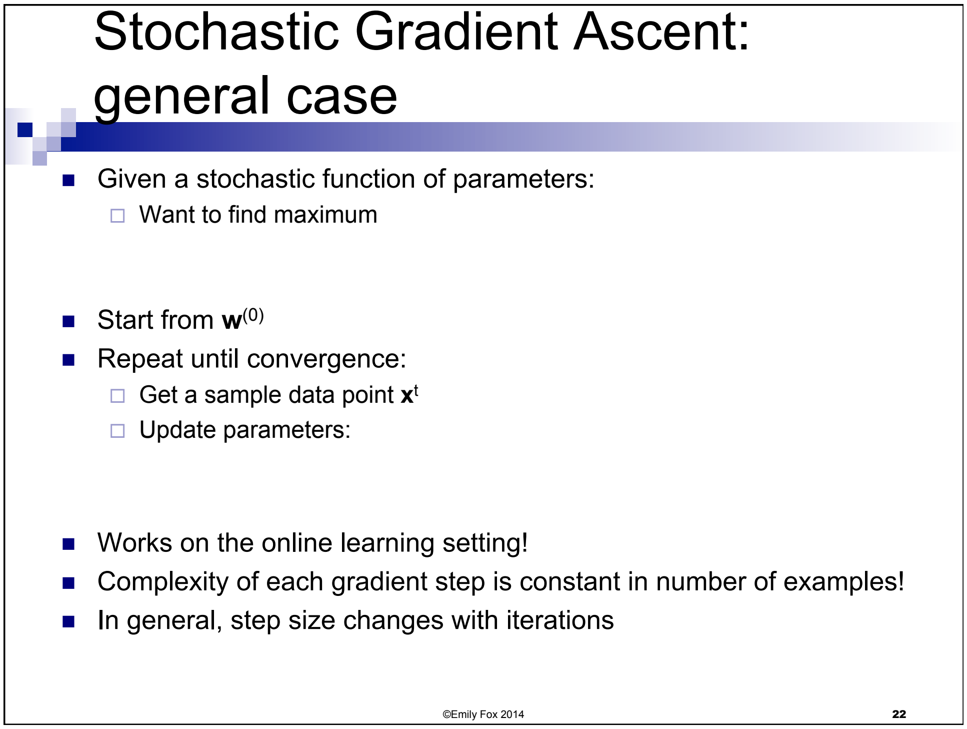
- "True" gradient ascent rule:

- How do we estimate expected gradient?



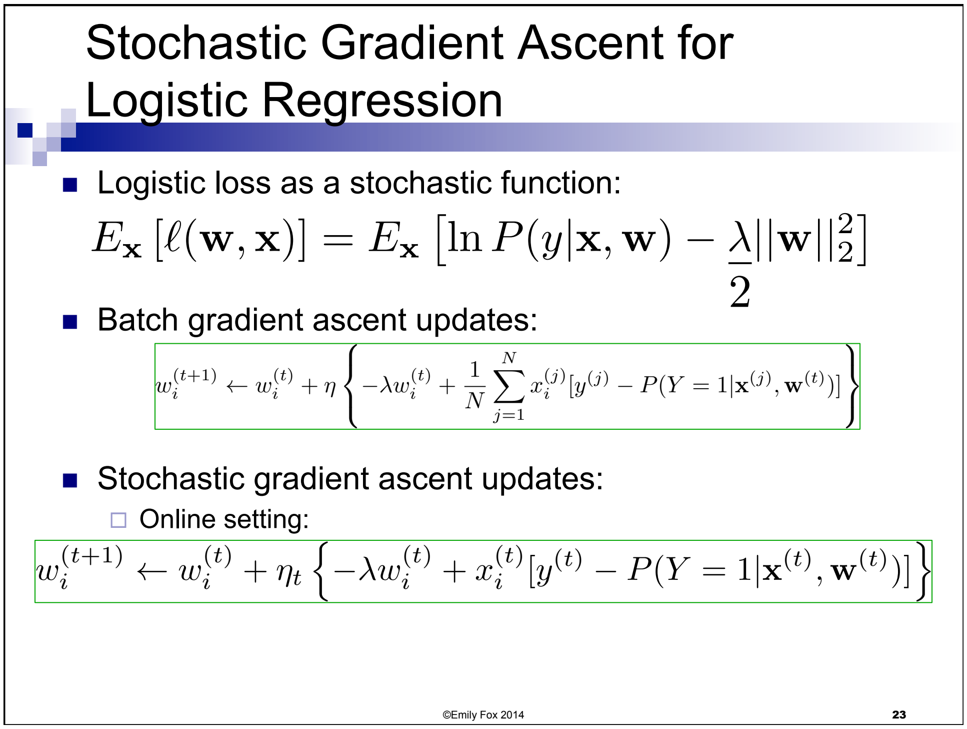
SGD: Stochastic Gradient Ascent (or Descent)

- "True" gradient: $\nabla \ell(\mathbf{w}) = E_{\mathbf{x}}\left[\nabla \ell(\mathbf{w}, \mathbf{x})\right]$
- Sample based approximation:
- 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
- Start from $\mathbf{w}^{(0)}$
- Repeat until convergence:
  - □ Get a sample data point $\mathbf{x}^t$
  - □ Update parameters:
- Works on 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:

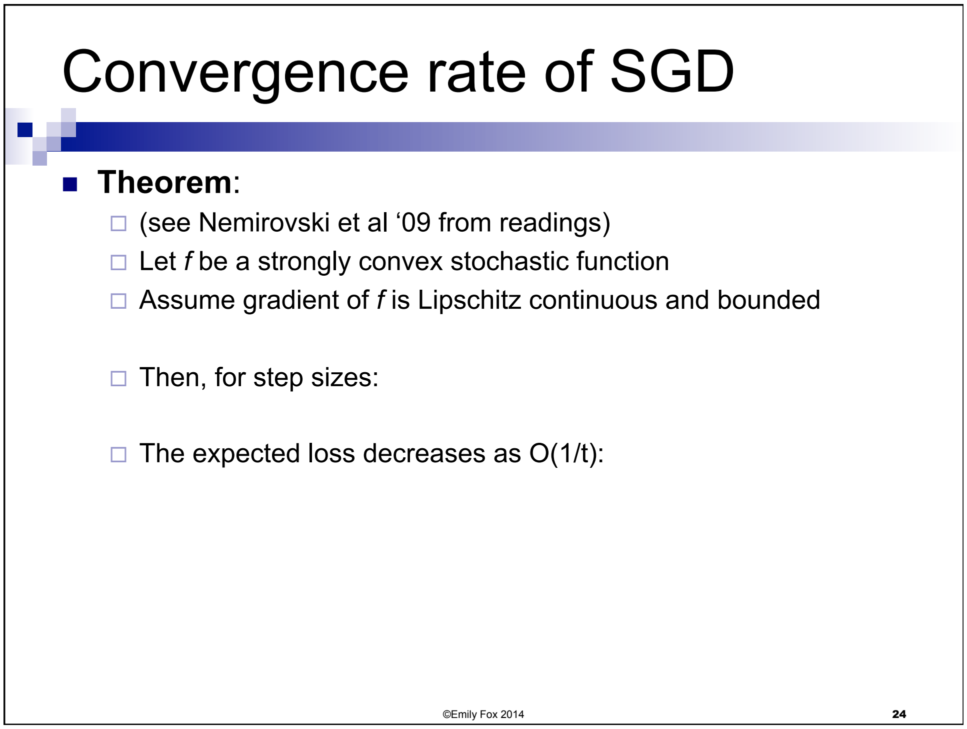
$$E_{\mathbf{x}}\left[\ell(\mathbf{w},\mathbf{x})\right] = E_{\mathbf{x}}\left[\ln P(y|\mathbf{x},\mathbf{w}) - \frac{\lambda}{2}||\mathbf{w}||_2^2\right]$$

- 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:

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



Convergence rate of SGD

- **Theorem**:
  - □ (see Nemirovski et al '09 from readings)
  - □ Let *f* be a strongly convex stochastic function
  - □ Assume gradient of *f* is Lipschitz continuous and bounded
  - □ Then, for step sizes:
  - □ The expected loss decreases as O(1/t):

Convergence rates for gradient descent/ascent versus 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
  - And, when analyzing true error, situation even more subtle… expected running time about the same, see readings

Motivating AdaGrad (Duchi, Hazan, Singer 2011)

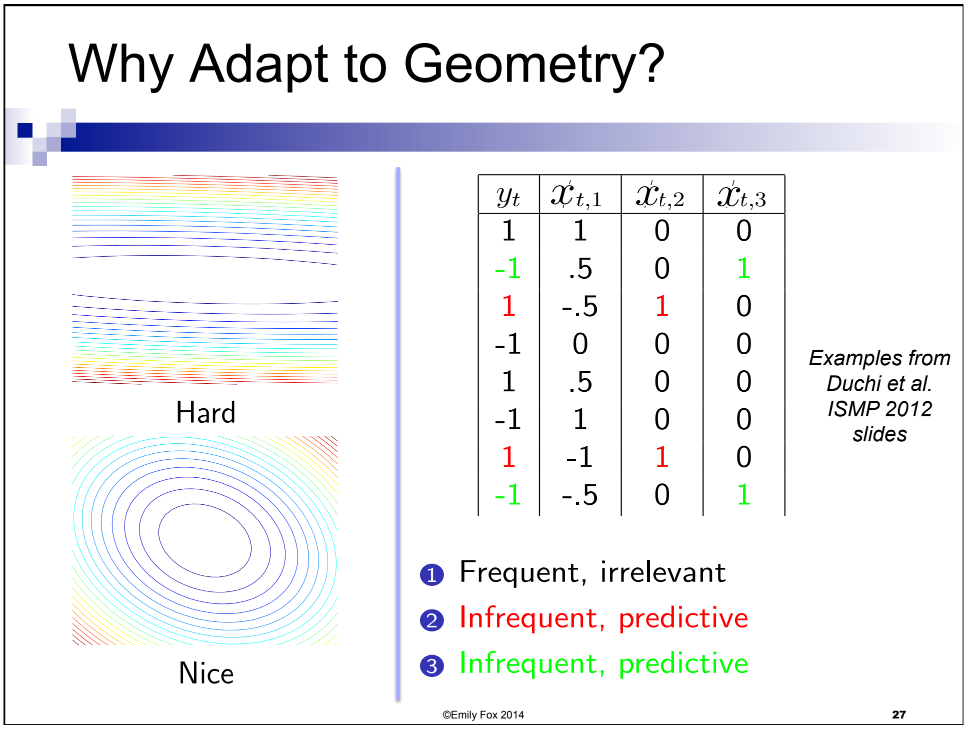
- Assuming $\mathbf{w} \in \mathbb{R}^d$, standard stochastic (sub)gradient descent updates are of the form:

$$w_i^{(t+1)} \leftarrow w_i^{(t)} - \eta g_{t,i}$$

- Should all features share the same learning rate?

- Often have high-dimensional feature spaces
  - Many features are irrelevant
  - Rare features are often very informative

- Adagrad provides a feature-specific adaptive learning rate by incorporating knowledge of the geometry of past observations



Why Adapt to Geometry?

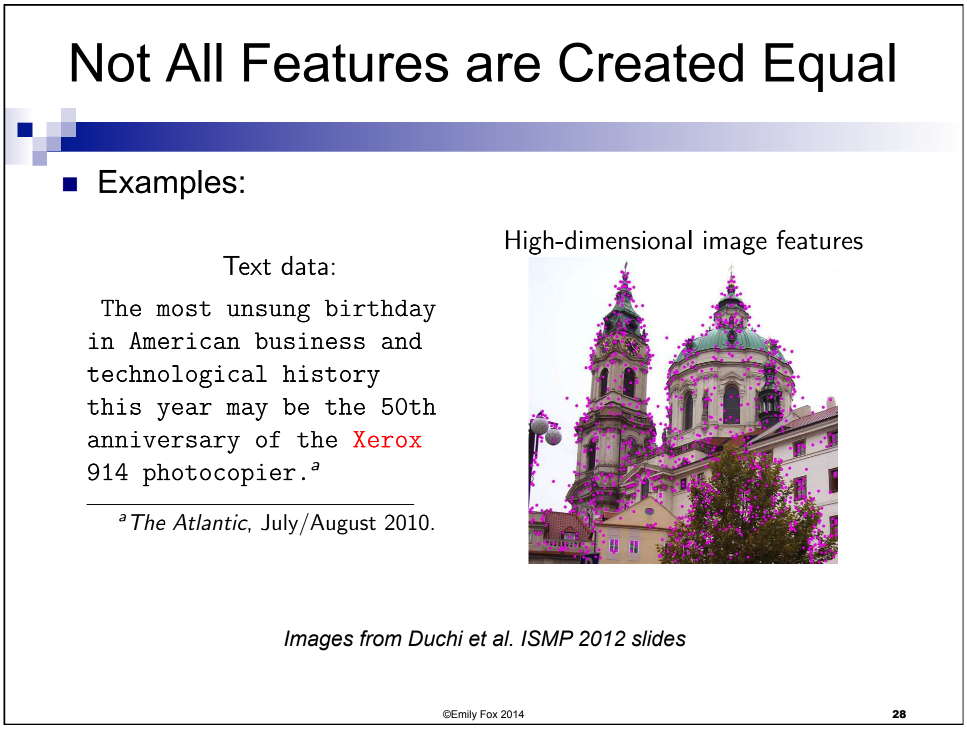
Hard

Nice

y_t	$x_{t,1}$	$x_{t,2}$	$x_{t,3}$
1	1	0	0
-1	.5	0	1
1	-.5	1	0
-1	0	0	0
1	.5	0	0
-1	1	0	0
1	-1	1	0
-1	-.5	0	1

Examples from Duchi et al. ISMP 2012 slides

1. Frequent, irrelevant
2. Infrequent, predictive
3. Infrequent, predictive



Not All Features are Created Equal

- Examples:

Text data:

```
The most unsung birthday
in American business and
technological history
this year may be the 50th
anniversary of the Xerox
914 photocopier.ᵃ
```

[a] *The Atlantic*, July/August 2010.

High-dimensional image features

Images from Duchi et al. ISMP 2012 slides

Projected Gradient

$$w_i^{(t+1)} \leftarrow w_i^{(t)} - \eta g_{t,i}$$

- Brief aside…
- Consider an arbitrary feature space $\mathbf{w} \in \mathcal{W}$
- If $\mathbf{w} \in \mathcal{W}$, can use ***projected gradient*** for (sub)gradient descent

$$\mathbf{w}^{(t+1)} =$$

Regret Minimization

- How do we assess the performance of an online algorithm?
- Algorithm iteratively predicts $\mathbf{w}^{(t)}$
- Incur ***loss*** $f_t(\mathbf{w}^{(t)})$
- ***Regret***:
  What is the total incurred loss of algorithm relative to the best choice of $\mathbf{w}$ that could have been made ***retrospectively***

$$R(T) = \sum_{t=1}^{T} f_t(\mathbf{w}^{(t)}) - \inf_{\mathbf{w}\in\mathcal{W}} \sum_{t=1}^{T} f_t(\mathbf{w})$$

Regret Bounds for Standard SGD

- Standard projected gradient stochastic updates:

$$\mathbf{w}^{(t+1)} = \arg\min_{\mathbf{w}\in\mathcal{W}} ||\mathbf{w} - (\mathbf{w}^{(t)} - \eta g_t)||_2^2$$

- Standard regret bound:

$$\sum_{t=1}^{T} f_t(\mathbf{w}^{(t)}) - f_t(\mathbf{w}^*) \leq \frac{1}{2\eta}||\mathbf{w}^{(1)} - \mathbf{w}^*||_2^2 + \frac{\eta}{2}\sum_{t=1}^{T}||g_t||_2^2$$

Projected Gradient using Mahalanobis

- Standard projected gradient stochastic updates:

$$\mathbf{w}^{(t+1)} = \arg\min_{\mathbf{w}\in\mathcal{W}} ||\mathbf{w} - (\mathbf{w}^{(t)} - \eta g_t)||_2^2$$

- What if instead of an L_2 metric for projection, we considered the ***Mahalanobis*** norm

$$\mathbf{w}^{(t+1)} = \arg\min_{\mathbf{w}\in\mathcal{W}} ||\mathbf{w} - (\mathbf{w}^{(t)} - \eta A^{-1} g_t)||_A^2$$

Mahalanobis Regret Bounds

$$\mathbf{w}^{(t+1)} = \arg\min_{\mathbf{w}\in\mathcal{W}} ||\mathbf{w} - (\mathbf{w}^{(t)} - \eta A^{-1} g_t)||_A^2$$

- **What *A* to choose?**

- Regret bound now:

$$\sum_{t=1}^{T} f_t(\mathbf{w}^{(t)}) - f_t(\mathbf{w}^*) \leq \frac{1}{2\eta}||\mathbf{w}^{(1)} - \mathbf{w}^*||_A^2 + \frac{\eta}{2}\sum_{t=1}^{T}||g_t||_{A^{-1}}^2$$

- What if we minimize upper bound on regret w.r.t. *A* in hindsight?

$$\min_A \sum_{t=1}^{T} \langle g_t, A^{-1} g_t \rangle$$

Mahalanobis Regret Minimization

- Objective:

$$\min_A \sum_{t=1}^{T} \langle g_t, A^{-1} g_t \rangle \quad \text{subject to } A \succeq 0, \mathrm{tr}(A) \leq C$$

- Solution:

$$A = c\left(\sum_{t=1}^{T} g_t g_t^T\right)^{\frac{1}{2}}$$

For proof, see Appendix E, Lemma 15 of Duchi et al. 2011.
Uses “trace trick” and Lagrangian.

- *A* defines the norm of the metric space we should be operating in

AdaGrad Algorithm

$$\mathbf{w}^{(t+1)} = \arg\min_{\mathbf{w}\in\mathcal{W}} ||\mathbf{w} - (\mathbf{w}^{(t)} - \eta A^{-1} g_t)||_A^2$$

- At time *t*, estimate optimal (sub)gradient modification *A* by

$$A_t = \left(\sum_{\tau=1}^{t} g_\tau g_\tau^T\right)^{\frac{1}{2}}$$

- For *d* large, A_t is computationally intensive to compute. Instead,

- Then, algorithm is a simple modification of normal updates:

$$\mathbf{w}^{(t+1)} = \arg\min_{\mathbf{w}\in\mathcal{W}} ||\mathbf{w} - (\mathbf{w}^{(t)} - \eta \text{diag}(A_t)^{-1} g_t)||_{\text{diag}(A_t)}^2$$

AdaGrad in Euclidean Space

- For $\mathcal{W} = \mathbb{R}^d$,

- For each feature dimension,

$$w_i^{(t+1)} \leftarrow w_i^{(t)} - \eta_{t,i} g_{t,i}$$

where

$$\eta_{t,i} =$$

- That is,

$$w_i^{(t+1)} \leftarrow w_i^{(t)} - \frac{\eta}{\sqrt{\sum_{\tau=1}^{t} g_{\tau,i}^2}} g_{t,i}$$

- Each feature dimension has it's own learning rate!
  - □ Adapts with *t*
  - □ Takes geometry of the past observations into account
  - □ Primary role of η is determining rate the first time a feature is encountered

AdaGrad Theoretical Guarantees

- AdaGrad regret bound:

$$\sum_{t=1}^{T} f_t(\mathbf{w}^{(t)}) - f_t(\mathbf{w}^*) \leq 2R_\infty \sum_{i=1}^{d} ||g_{1:T,j}||_2$$

$$R_\infty := \max_t ||\mathbf{w}^{(t)} - \mathbf{w}^*||_\infty$$

- So, what does this mean in practice?

- Many cool examples. This really is used in practice!
- Let's just examine one…

AdaGrad Theoretical Example

- Expect to out-perform when gradient vectors are sparse
- SVM hinge loss example:

$f_t(\mathbf{w}) = [1 - y^t \left\langle \mathbf{x}^t, \mathbf{w} \right\rangle]_+$ where $\mathbf{x}^t \in \{-1, 0, 1\}^d$

- If $x_j^t \neq 0$ with probability $\propto j^{-\alpha}, \quad \alpha > 1$

$$\mathbb{E}\left[f\left(\frac{1}{T}\sum_{t=1}^{T}\mathbf{w}^{(t)}\right)\right] - f(\mathbf{w}^*) = \mathcal{O}\left(\frac{||\mathbf{w}^*||_\infty}{\sqrt{T}} \cdot \max\{\log d, d^{1-\alpha/2}\}\right)$$

- Previously best known method:

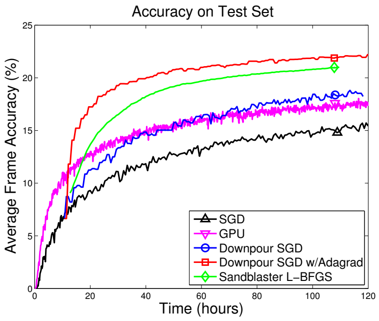
$$\mathbb{E}\left[f\left(\frac{1}{T}\sum_{t=1}^{T}\mathbf{w}^{(t)}\right)\right] - f(\mathbf{w}^*) = \mathcal{O}\left(\frac{||\mathbf{w}^*||_\infty}{\sqrt{T}} \cdot \sqrt{d}\right)$$

Neural Network Learning

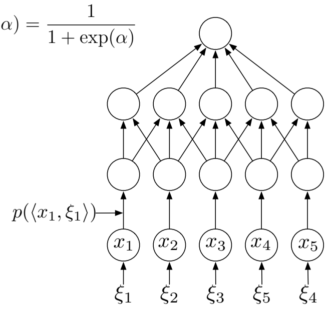
- Very non-convex problem, but use SGD methods anyway

$$f(x;\xi) = \log\left(1+\exp\left(\langle[p(\langle x_1,\xi_1\rangle)\ \cdots\ p(\langle x_k,\xi_k\rangle)],\xi_0\rangle\right)\right)$$

$$p(\alpha) = \frac{1}{1+\exp(\alpha)}$$



(Dean et al. 2012)



Distributed, $d = 1.7 \cdot 10^9$ parameters. SGD and AdaGrad use 80 machines (1000 cores), L-BFGS uses 800 (10000 cores)

Images from Duchi et al. ISMP 2012 slides

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
- AdaGrad motivation, derivation, and algorithm