University of Colorado **Boulder**

Department of Computer Science

CSCI 5622: Machine Learning

Chris Ketelsen

Lecture 5:
SGD Tips and Tricks and Multiclass Classification

Tips and Tricks for SGD Roadmap

- Feature Scaling
- Learning Rate Schedules
- Momentum

Feature Scaling with Regularization

$$NLL(\mathbf{w}) = -\sum_{i=1}^{m} \left[y_i \log \text{sigm}(\mathbf{w}^T \mathbf{x}_i) + (1 - y_i) \log(1 - \text{sigm}(\mathbf{w}^T \mathbf{x}_i)) \right] + \lambda \underbrace{\sum_{k=1}^{D} w_k^2}_{\text{REGULARIZATION}}$$

x_2 = # floors

x_5 = Age

$w_2 x_2$ vs $w_5 x_5$

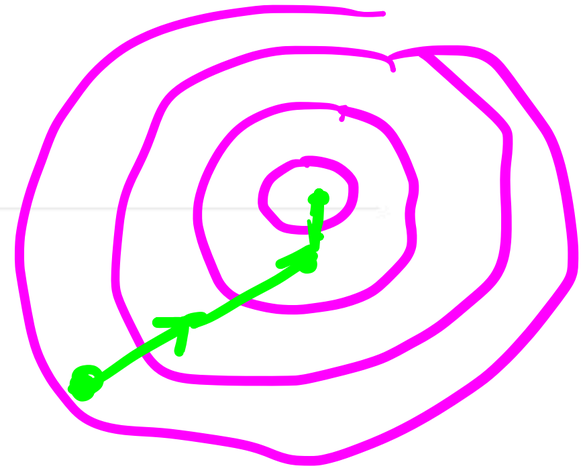
w_2 ↗ Big, x_2 ↑ small

w_5 ↑ small, x_5 ↑ BIG

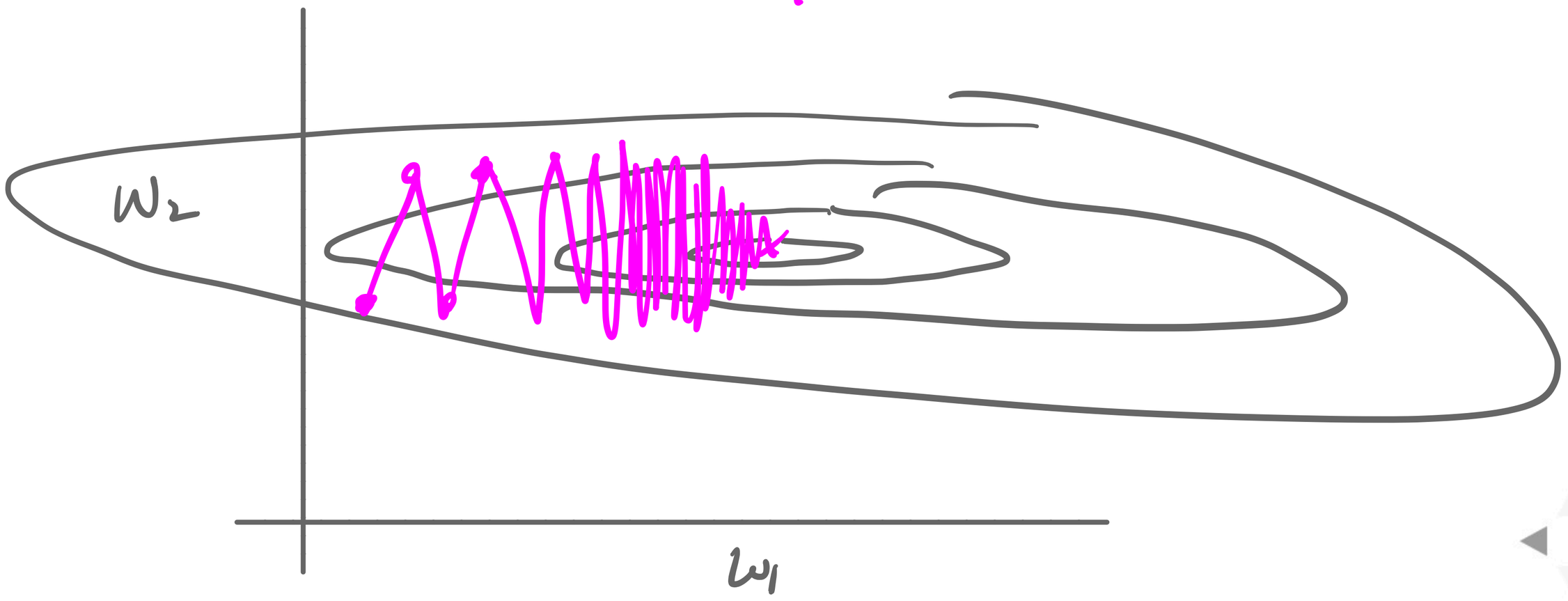
REG SEES w_2 AS ENEMY BEFORE w_5

Feature Scaling for SGD Convergence

$$RSS = \sum_{i=1}^{m} (w_1 x_{i1} + w_2 x_{i2} - y_i)^2$$

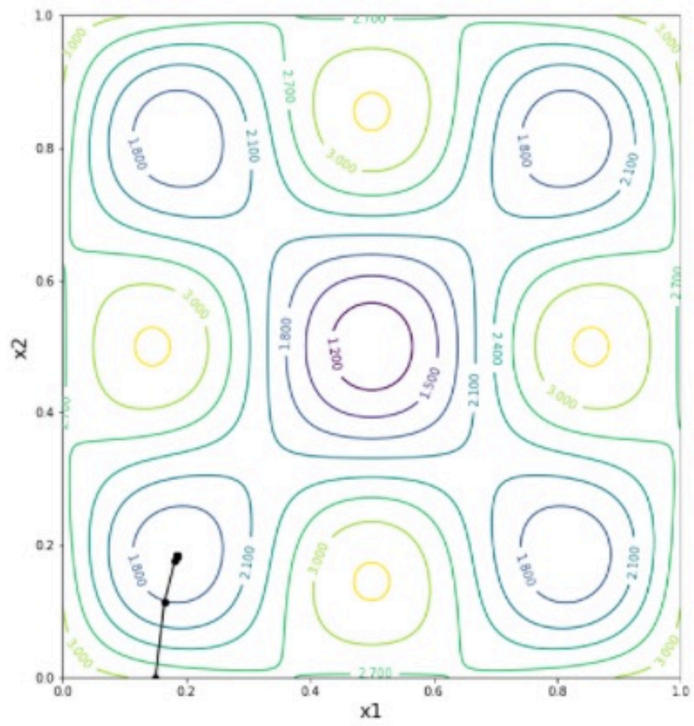


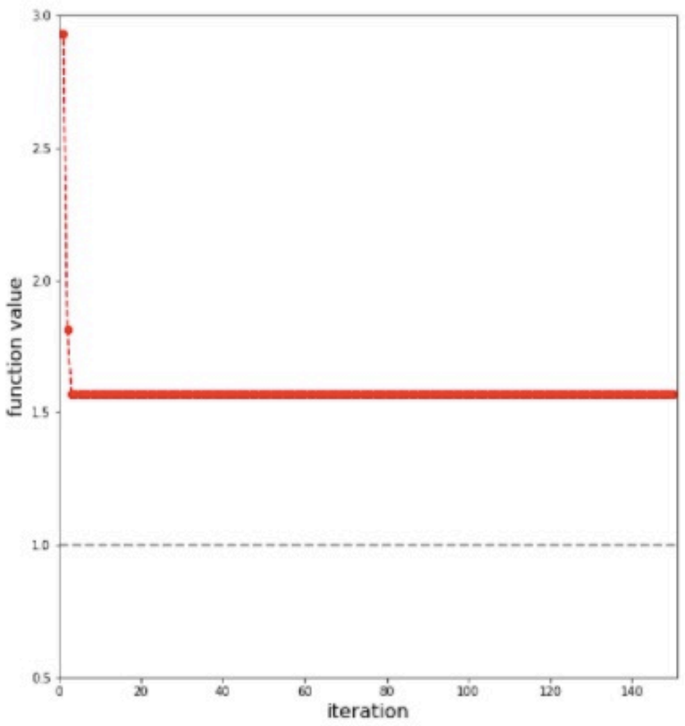
$$\overset{Ex}{=} (w_1 \cdot 1 + w_2 \cdot 100 - 1)^2 + (w_1 \cdot 1 + w_2 \cdot (-100) - 0)^2$$



Learning Rate Schedules

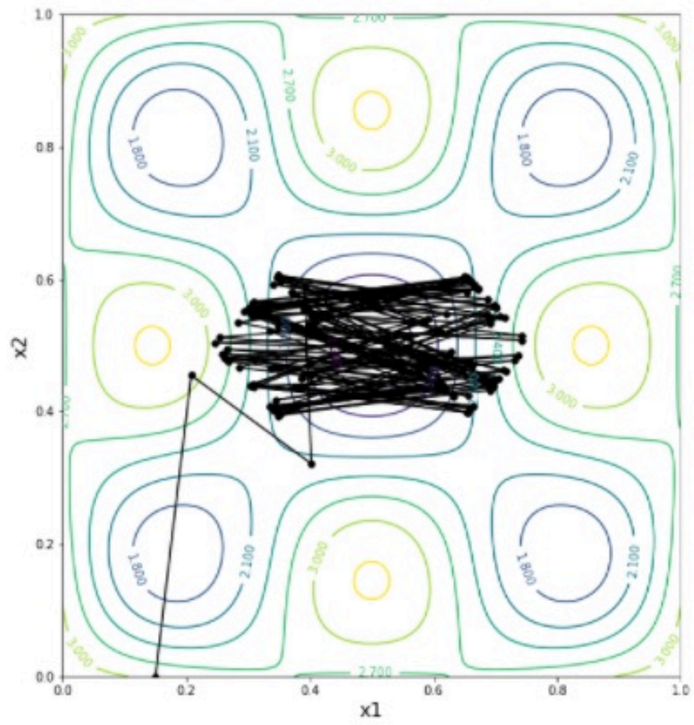
Allowing the learning rate to evolve (usually shrink) over time can aid convergence in both the convex and nonconvex cases

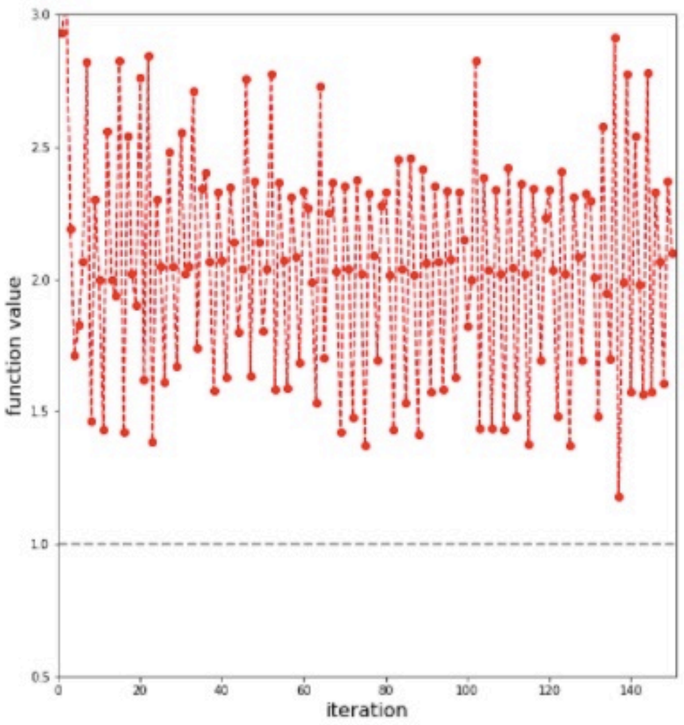




Learning Rate Schedules

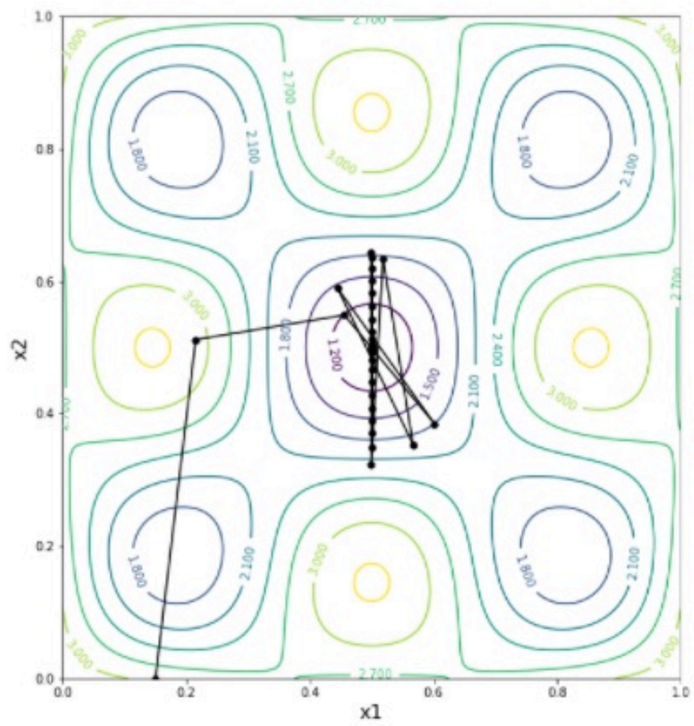
Allowing the learning rate to evolve (usually shrink) over time can aid convergence in both the convex and nonconvex cases

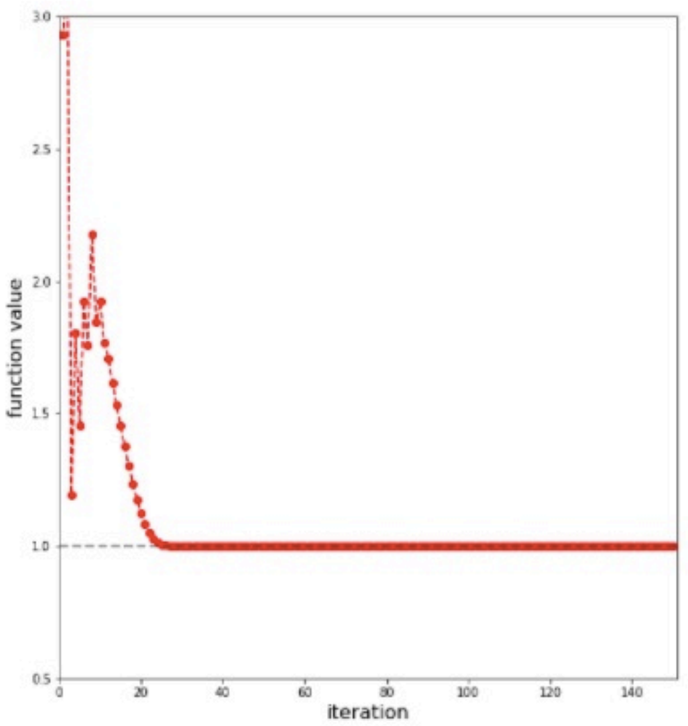




Learning Rate Schedules

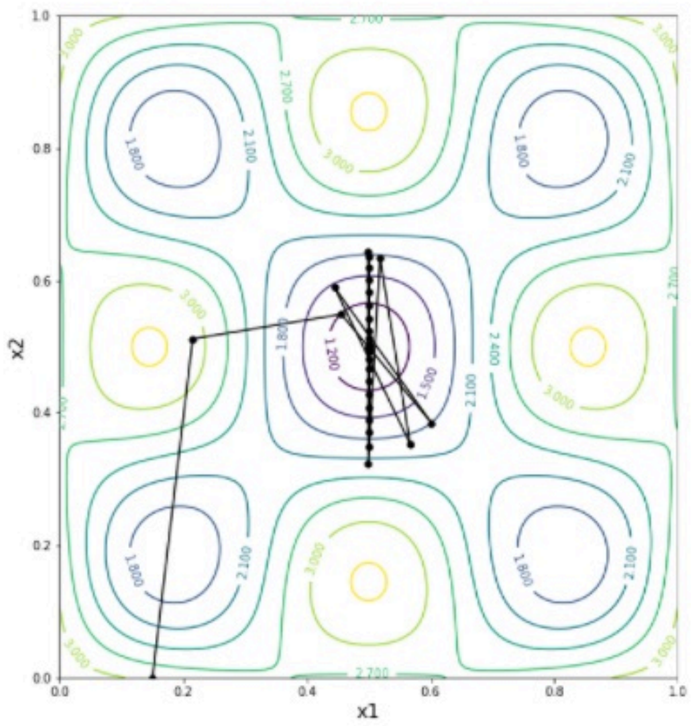
Allowing the learning rate to evolve (usually shrink) over time can aid convergence in both the convex and nonconvex cases

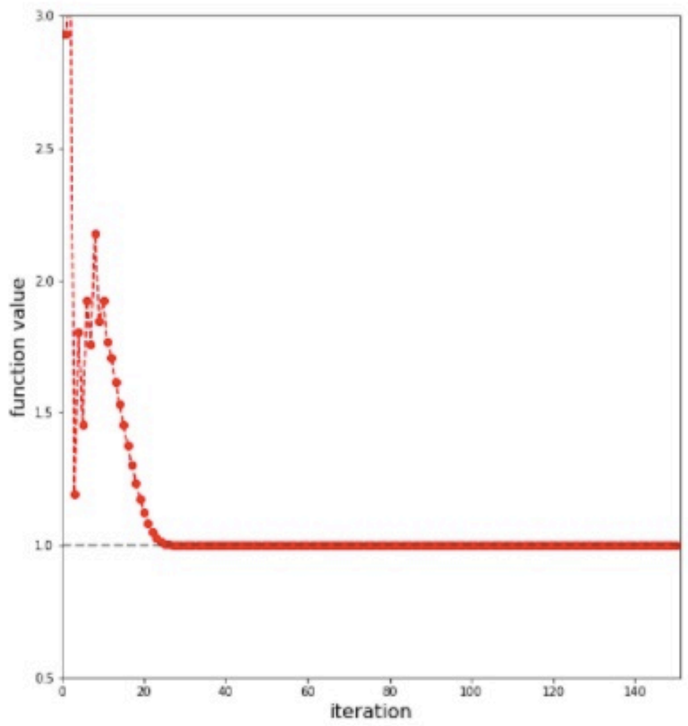




Learning Rate Schedules

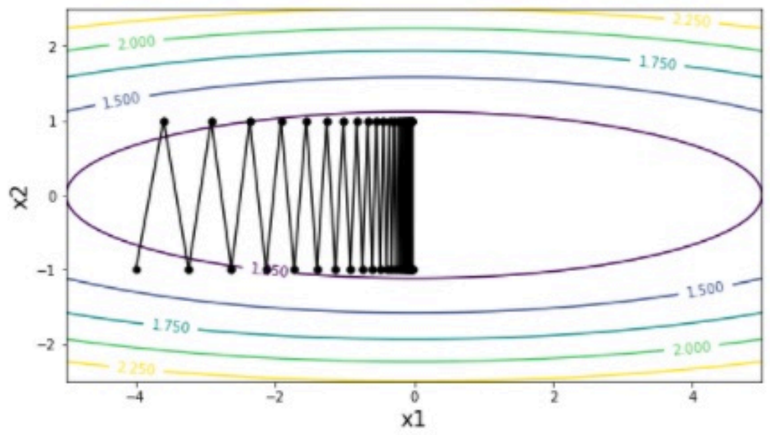
A common, simple learning rate schedule: $\eta_k = \dfrac{\eta_0}{1 + \alpha\, k\,/\,n}$

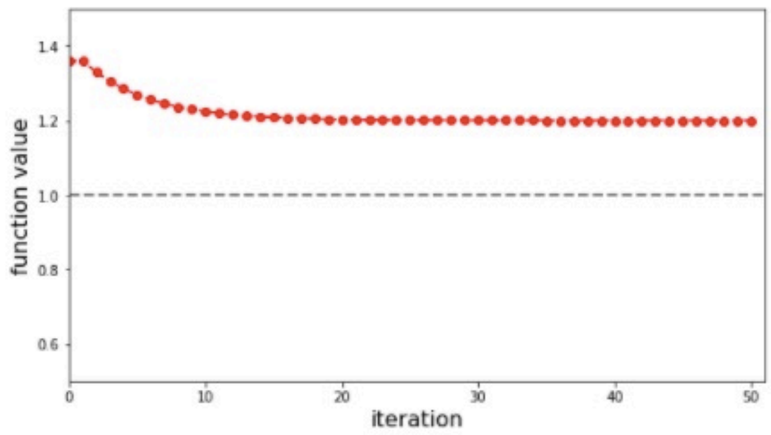




Momentum

SGA tends to hop around quite a bit when determining step direction based on one or a small sample of training examples



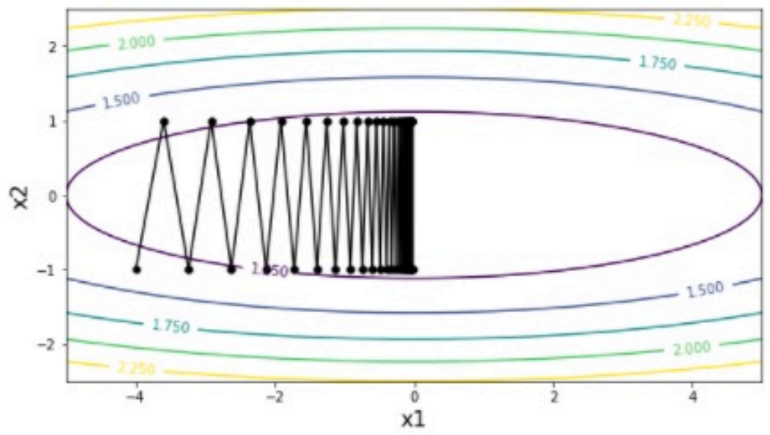


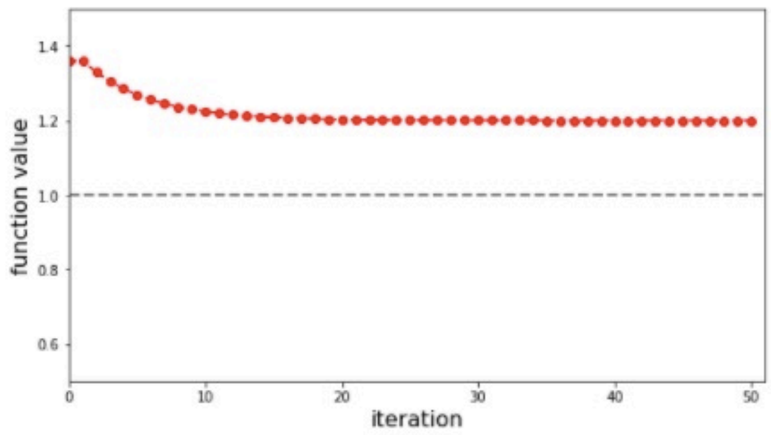
SGA is a *memoryless* algorithm

Keeping and using information about the past is a common acceleration technique in optimization

Momentum

SGA tends to hop around quite a bit when determining step direction based on one or a small sample of training examples





Adding a **momentum** term is popular:

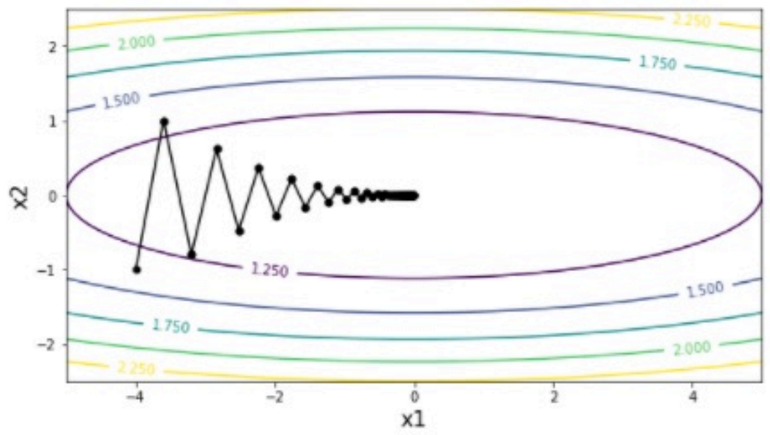
-NLL

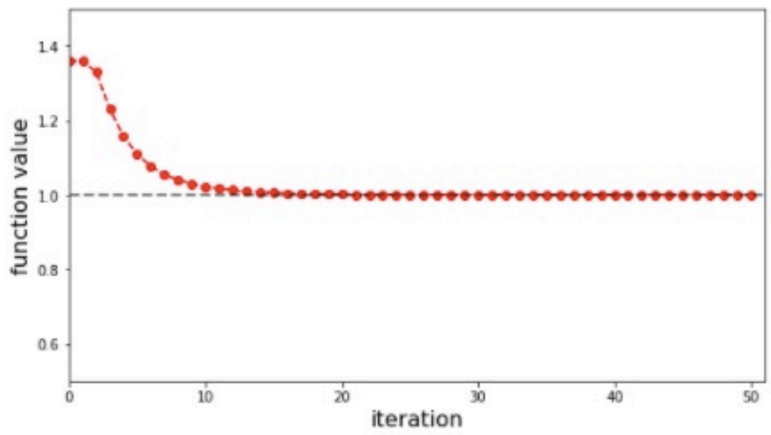
$$\mathbf{v}^{(k)} = \eta \, \nabla_{\mathbf{w}} J(\mathbf{w}^{(k-1)}, \mathbf{x}_i)$$

$$\mathbf{w}^{(k)} = \mathbf{w}^{(k-1)} - \mathbf{v}^{(k)}$$

Momentum

SGA tends to hop around quite a bit when determining step direction based on one or a small sample of training examples





Adding a **momentum** term is popular:

$$\mathbf{v}^{(k)} = \eta \nabla_{\mathbf{w}} J(\mathbf{w}^{(k-1)}, \mathbf{x}_i) + \mu \mathbf{v}^{(k-1)}$$

$$\mathbf{w}^{(k)} = \mathbf{w}^{(k-1)} - \mathbf{v}^{(k)}$$

GRADIENT

PREVIOUS STEP

μ: tuning PARAM

Classification Roadmap

- Inherently multiclass classifiers
- Multiclass classifiers from binary classifiers
  - One-vs-All
  - All Pairs
  - Error Correcting Codes (in-class exercises)

General Classification

Given: $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{m}$ training examples

$$\mathbf{x}_i \in \mathbb{R}^D \qquad y_i \in \{1, 2, \ldots, K\}$$

Goal: Given new data $\mathbf{x}$, predict its label y

Some problems are inherently binary:

- Spam vs Ham

Some problems are inherently multiclass:

- Is there a car/truck/bike/person in this image
- Should self-driving car speed up / slow down / hold speed

Inherently Multiclass Classification

Some methods we've seen are inherently multiclass

Naive Bayes

For each class c_k, estimate

$$p(y = c_k \mid \mathbf{x}, \mathcal{D})$$

Assign to $\mathbf{x}$ the class with highest probability

$$\hat{y} = \arg\max_c p(y = c \mid \mathbf{x}, \mathcal{D})$$

Inherently Multiclass Classification

Some methods we've seen are inherently multiclass

K-Nearest Neighbors

1. Find $N_K(\mathbf{x}, \mathcal{D})$: the K training examples in $\mathcal{D}$ "nearest" to $\mathbf{x}$
2. Assign $\hat{y}$ the majority label of N_K

Inherently Multiclass Classification

Some methods we've seen are inherently multiclass

Logistic Regression

ODDS

For binary classification we modeled

$$\log \frac{p(y=1 \mid \mathbf{x}, D)}{p(y=0 \mid \mathbf{x}, D)} = w_0 + w_1 x_1 + \cdots + w_D x_D = \mathbf{w}^T \mathbf{x}$$

$$\log \frac{p(y=1 \mid \mathbf{x}, D)}{1 - p(y=1 \mid \mathbf{x}, D)} = w_0 + w_1 x_1 + \cdots + w_D x_D = \mathbf{w}^T \mathbf{x}$$

with

$$p(y = 1 \mid \mathbf{x}) = \text{sigm}(\mathbf{w}^T \mathbf{x}), \quad p(y = 0 \mid \mathbf{x}) = 1 - \text{sigm}(\mathbf{w}^T \mathbf{x})$$

Inherently Multiclass Classification

RATIO: $y=1$ to $y=K$

RATIO: $y=2$ to $y=K$

For multiclass classification with K classes we can do

$$\log \frac{p(y=1 \mid \mathbf{x},D)}{p(y=K \mid \mathbf{x},D)} = w_{10} + w_{11}x_1 + \cdots + w_{1D}x_D = \mathbf{w}_1^T\mathbf{x}$$

$$\log \frac{p(y=2 \mid \mathbf{x},D)}{p(y=K \mid \mathbf{x},D)} = w_{20} + w_{21}x_1 + \cdots + w_{2D}x_D = \mathbf{w}_2^T\mathbf{x}$$

$$\vdots$$

$$\log \frac{p(y=K-1 \mid \mathbf{x},D)}{p(y=K \mid \mathbf{x},D)} = w_{(K-1)0} + w_{(K-1)1}x_1 + \cdots + w_{(K-1)D}x_D = \mathbf{w}_{K-1}^T\mathbf{x}$$

with

$$p(y=k \mid \mathbf{x}) = \frac{\exp(\mathbf{w}_k^T\mathbf{x})}{1+\sum_{\ell=1}^{K-1}\exp(\mathbf{w}_\ell^T\mathbf{x})} \quad k=1,\ldots,K-1, \qquad p(y=K \mid \mathbf{x}) = \frac{1}{1+\sum_{\ell=1}^{K-1}\exp(\mathbf{w}_\ell^T\mathbf{x})}$$

Inherently Binary Classification

Some methods can be implemented as binary classification

- Logistic Regression

Some common methods we'll see later are inherently binary

- Perceptron
- Support Vector Machines
- Boosted Decision Stumps

Want to leverage these to do multiclass labeling

Inherently Binary Classification

Example Data:

$$\langle \texttt{name=Cindy, age=5, sex=F} \rangle \quad ,$$
$$\langle \texttt{name=Marcia, age=15, sex=F} \rangle \quad ,$$
$$\langle \texttt{name=Bobby, age=6, sex=M} \rangle \quad ,$$
$$\langle \texttt{name=Jan, age=12, sex=F} \rangle \quad ,$$
$$\langle \texttt{name=Peter, age=13, sex=M} \rangle \quad ,$$

We'll look at two strategies

- **One-vs-All**
- **All-Pairs**

One-vs-All

BLACK VS NOT BLACK

	■	■	■	■
$\mathbf{x}_1$ ■	−	+	−	−
$\mathbf{x}_2$ ■	−	−	+	−
$\mathbf{x}_3$ ■	−	−	−	+
$\mathbf{x}_4$ ■	−	+	−	−
$\mathbf{x}_5$ ■	+	−	−	−
	h_1	h_2	h_3	h_4

- Build K binary problems of form *Class k vs Not Class k*
- Evaluate all h's. Choose w/ highest probability / confidence

All-Pairs

	■ v ■	■ v ■	■ v ■	■ v ■	■ v ■	■ v ■
$\mathbf{x}_1$ ■	−			−		−
$\mathbf{x}_2$ ■		−	+			+
$\mathbf{x}_3$ ■			−	+	−	
$\mathbf{x}_4$ ■	−			−		−
$\mathbf{x}_5$ ■	+	+			+	
	h_{12}	h_{13}	h_{34}	h_{42}	h_{14}	h_{32}

- Build one binary problem for each pair of classes
- Define h_{ij} case when $i \rightarrow +$ and $j \rightarrow -$
- **Note**: $h_{ij} = -h_{ji}$
- Classify with $H(\mathbf{x}) = \arg\max_i(\sum_j h_{ij}(\mathbf{x}))$

One-vs-All vs All-Pairs

m = # TRAINING EXAMPLES

Time Complexity:

Suppose training time is $\mathcal{O}(m^\alpha)$ and testing time is $\mathcal{O}(T)$

	Train	Test
OVA	$\mathcal{O}(km^\alpha)$	$\mathcal{O}(kT)$
APs	$\mathcal{O}(k^2(m/k)^\alpha)$	$\mathcal{O}(k^2T)$

- One-vs-All wins at Test time
- All-Pairs wins at Training time
- All-Pairs generally faster and more accurate

Multiclass Wrap-Up

- Lots of choices for multiclass classification
- Pretty much plug-and-play with binary classifiers
- Can come up with multiclass theory based on binary theory

In Class

- **Get out laptops and get in groups!**

In Class

In Class

In Class

In Class

In Class

In Class

In Class

In Class

In Class

In Class

In Class