# Classification
# Logistic Regression

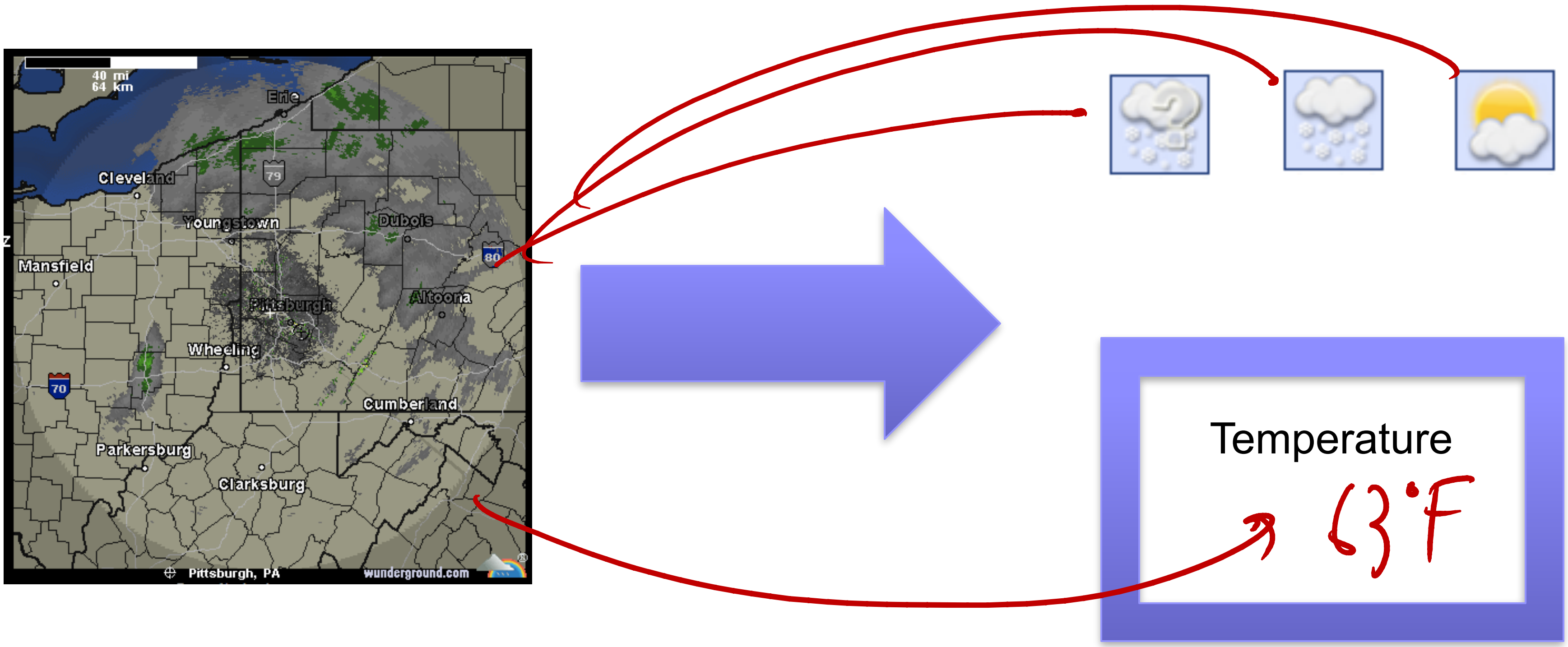
Machine Learning – CSE546

Kevin Jamieson
University of Washington

October 16, 2016

# THUS FAR, REGRESSION:
# PREDICT A CONTINUOUS VALUE GIVEN SOME INPUTS

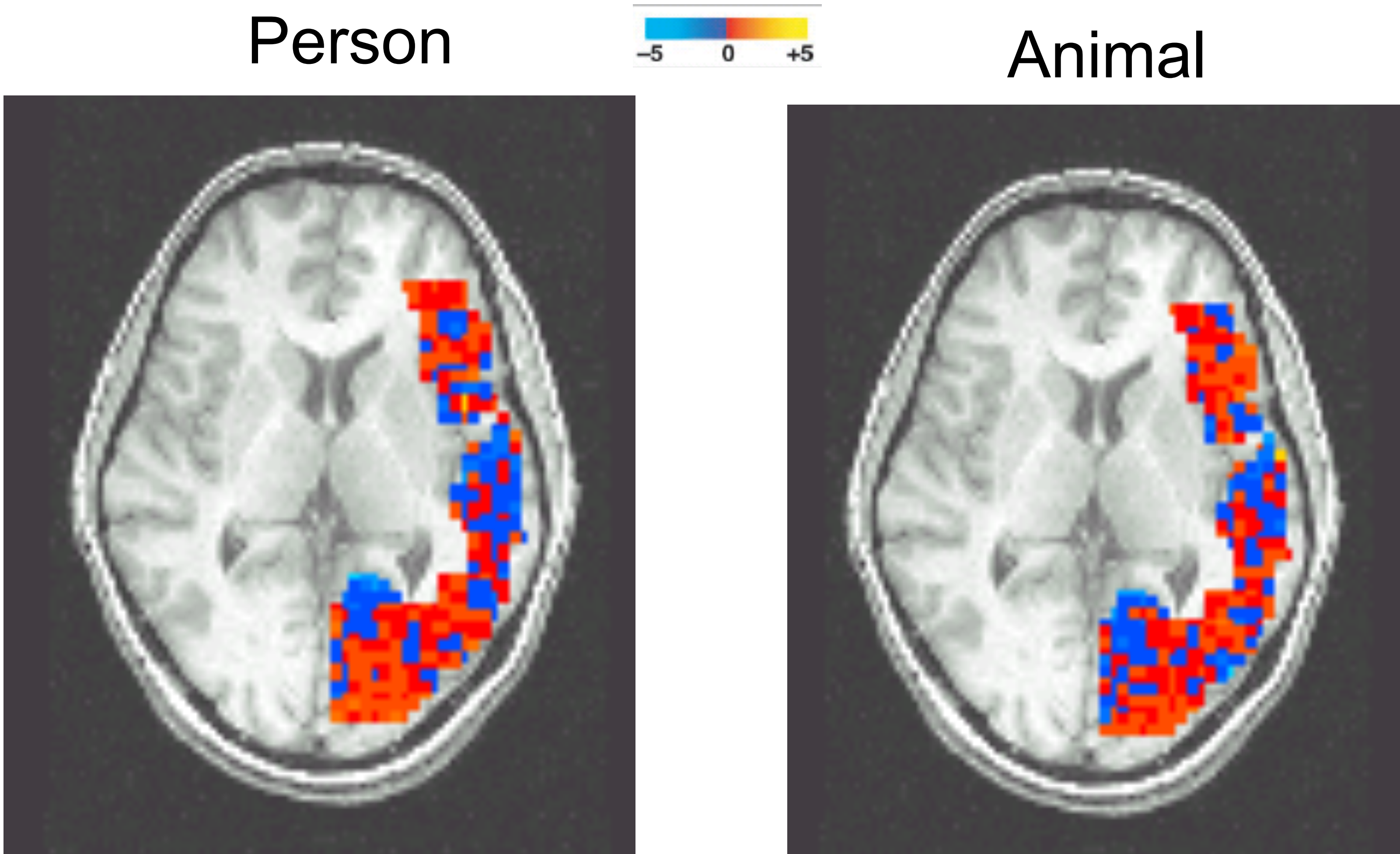
# Weather prediction revisted



# Reading Your Brain, Simple Example

[Mitchell et al.]

Pairwise classification accuracy: 85%



# Binary Classification

- **Learn**: f:**X —>**Y
  - **X** – features
  - Y – target classes

$$Y \in \{0, 1\}$$

- **Loss function:**
- **Expected loss of f:**
- Suppose you know P(Y|**X**) exactly, how should you classify?
  - Bayes optimal classifier:

# Binary Classification

- **Learn**: f:**X** —>Y
  - **X** – features
  - Y – target classes

$$Y \in \{0,1\}$$

- **Loss function:** $\ell(f(x),y) = \mathbf{1}\{f(x) \neq y\}$

- **Expected loss of f:**

$$\mathbb{E}_{XY}[\mathbf{1}\{f(X) \neq Y\}] = \mathbb{E}_X[\mathbb{E}_{Y|X}[\mathbf{1}\{f(x) \neq Y\}|X=x]]$$

$$\mathbb{E}_{Y|X}[\mathbf{1}\{f(x) \neq Y\}|X=x] = \sum_i P(Y=i|X=x)\mathbf{1}\{f(x) \neq i\} = \sum_{i \neq f(x)} P(Y=i|X=x)$$
$$= 1 - P(Y=f(x)|X=x)$$

- Suppose you know P(Y|**X**) exactly, how should you classify?
  - Bayes optimal classifier:

$$f(x) = \arg\max_y \mathbb{P}(Y=y|X=x)$$

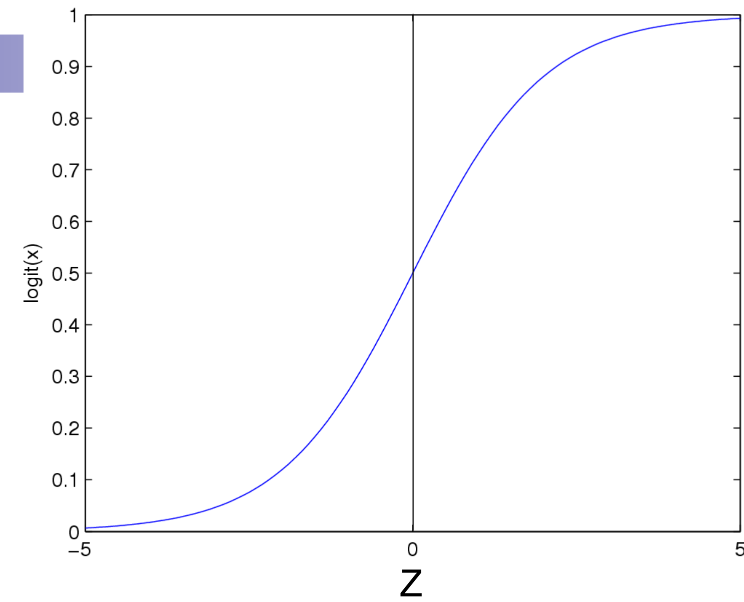
# Link Functions

- Estimating P(Y|**X**): Why not use standard linear regression?

- Combining regression and probability?
  - Need a mapping from real values to [0,1]
  - A link function!

# Logistic Regression

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




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

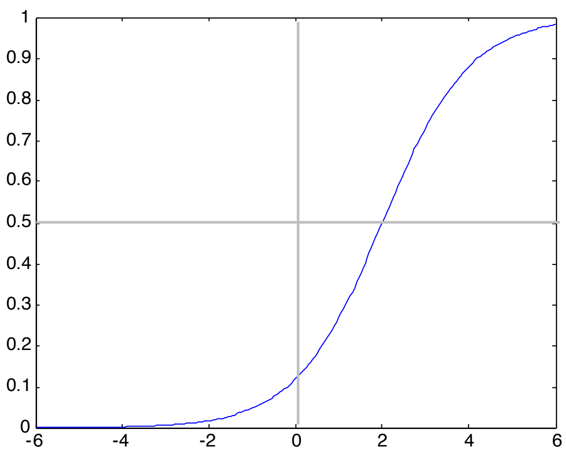
$$P(Y=0|X,W) = \frac{1}{1+exp(w_0+\sum_i w_i X_i)}$$

**Features can be discrete or continuous!**

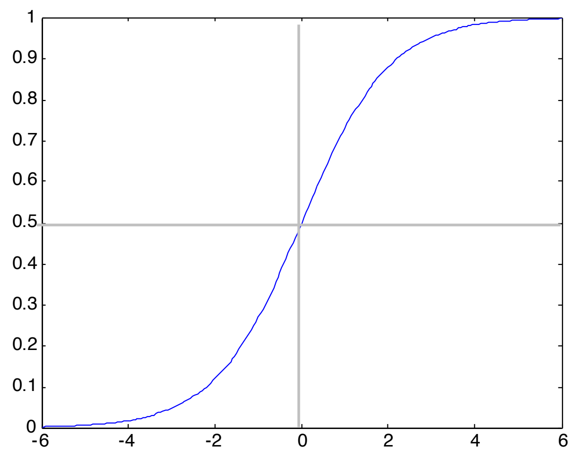
# Understanding the sigmoid

$$g(w_0 + \sum_i w_i x_i) = \frac{1}{1 + e^{w_0 + \sum_i w_i x_i}}$$

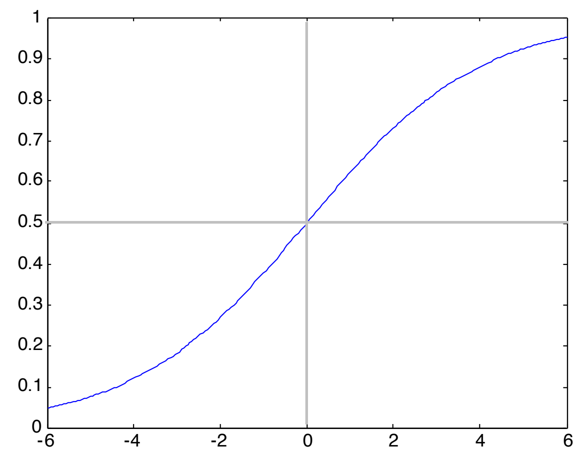
$w_0$=-2, $w_1$=-1




$w_0$=0, $w_1$=-1




$w_0$=0, $w_1$=-0.5



# Sigmoid for binary classes

$$\mathbb{P}(Y=0|w,X) = \frac{1}{1+\exp(w_0 + \sum_k w_k X_k)}$$

$$\mathbb{P}(Y=1|w,X) = 1 - \mathbb{P}(Y=0|w,X) = \frac{\exp(w_0 + \sum_k w_k X_k)}{1+\exp(w_0 + \sum_k w_k X_k)}$$

$$\frac{\mathbb{P}(Y=1|w,X)}{\mathbb{P}(Y=0|w,X)} =$$

# Sigmoid for binary classes

$$\mathbb{P}(Y=0|w,X) = \frac{1}{1+\exp(w_0 + \sum_k w_k X_k)}$$

$$\mathbb{P}(Y=1|w,X) = 1 - \mathbb{P}(Y=0|w,X) = \frac{\exp(w_0 + \sum_k w_k X_k)}{1+\exp(w_0 + \sum_k w_k X_k)}$$

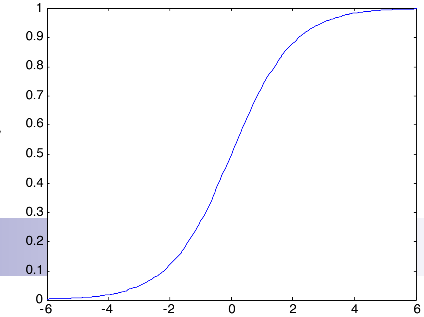
$$\frac{\mathbb{P}(Y=1|w,X)}{\mathbb{P}(Y=0|w,X)} = \exp(w_0 + \sum_k w_k X_k)$$

$$\log \frac{\mathbb{P}(Y=1|w,X)}{\mathbb{P}(Y=0|w,X)} = w_0 + \sum_k w_k X_k$$

**Linear Decision Rule!**

# Logistic Regression – a Linear classifier

$$\frac{1}{1+exp(-z)}$$




$$g(w_0 + \sum_i w_i x_i) \;=\; \frac{1}{1+e^{w_0+\sum_i w_i x_i}}$$

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

# Loss function: Conditional Likelihood

- Have a bunch of iid data of the form: $\{(x_i, y_i)\}_{i=1}^n \quad x_i \in \mathbb{R}^d, \quad y_i \in \{-1, 1\}$

$$P(Y = -1|x, w) = \frac{1}{1 + \exp(w^T x)}$$

$$P(Y = 1|x, w) = \frac{\exp(w^T x)}{1 + \exp(w^T x)}$$

- This is equivalent to:

$$P(Y = y|x, w) = \frac{1}{1 + \exp(-y\, w^T x)}$$

- So we can compute the maximum likelihood estimator:

$$\widehat{w}_{MLE} = \arg\max_w \prod_{i=1}^n P(y_i|x_i, w)$$

# Loss function: Conditional Likelihood

- Have a bunch of iid data of the form: $\{(x_i, y_i)\}_{i=1}^n \quad x_i \in \mathbb{R}^d, \quad y_i \in \{-1, 1\}$

$$\widehat{w}_{MLE} = \arg\max_w \prod_{i=1}^n P(y_i | x_i, w) \qquad P(Y = y | x, w) = \frac{1}{1 + \exp(-y\, w^T x)}$$

$$= \arg\min_w \sum_{i=1}^n \log(1 + \exp(-y_i\, x_i^T w))$$

# Loss function: Conditional Likelihood

- Have a bunch of iid data of the form: $\{(x_i, y_i)\}_{i=1}^n \quad x_i \in \mathbb{R}^d, \quad y_i \in \{-1, 1\}$

$$\widehat{w}_{MLE} = \arg\max_w \prod_{i=1}^n P(y_i|x_i, w) \qquad P(Y = y|x, w) = \frac{1}{1 + \exp(-y\, w^T x)}$$

$$= \arg\min_w \sum_{i=1}^n \log(1 + \exp(-y_i\, x_i^T w))$$

Logistic Loss: $\ell_i(w) = \log(1 + \exp(-y_i\, x_i^T w))$

Squared error Loss: $\ell_i(w) = (y_i - x_i^T w)^2$ (MLE for Gaussian noise)

# Loss function: Conditional Likelihood

- Have a bunch of iid data of the form: $\{(x_i, y_i)\}_{i=1}^n \quad x_i \in \mathbb{R}^d, \quad y_i \in \{-1, 1\}$

$$\widehat{w}_{MLE} = \arg\max_w \prod_{i=1}^n P(y_i|x_i, w) \qquad P(Y = y|x, w) = \frac{1}{1 + \exp(-y\, w^T x)}$$

$$= \arg\min_w \sum_{i=1}^n \log(1 + \exp(-y_i\, x_i^T w)) = J(w)$$

What does $J(w)$ look like? Is it convex?

# Loss function: Conditional Likelihood

- Have a bunch of iid data of the form: $\{(x_i, y_i)\}_{i=1}^n \quad x_i \in \mathbb{R}^d, \quad y_i \in \{-1, 1\}$

$$\widehat{w}_{MLE} = \arg\max_w \prod_{i=1}^n P(y_i|x_i, w) \qquad P(Y = y|x, w) = \frac{1}{1 + \exp(-y\, w^T x)}$$

$$= \arg\min_w \sum_{i=1}^n \log(1 + \exp(-y_i\, x_i^T w)) = J(w)$$

Good news: *J*(**w**) is convex function of **w**, no local optima problems

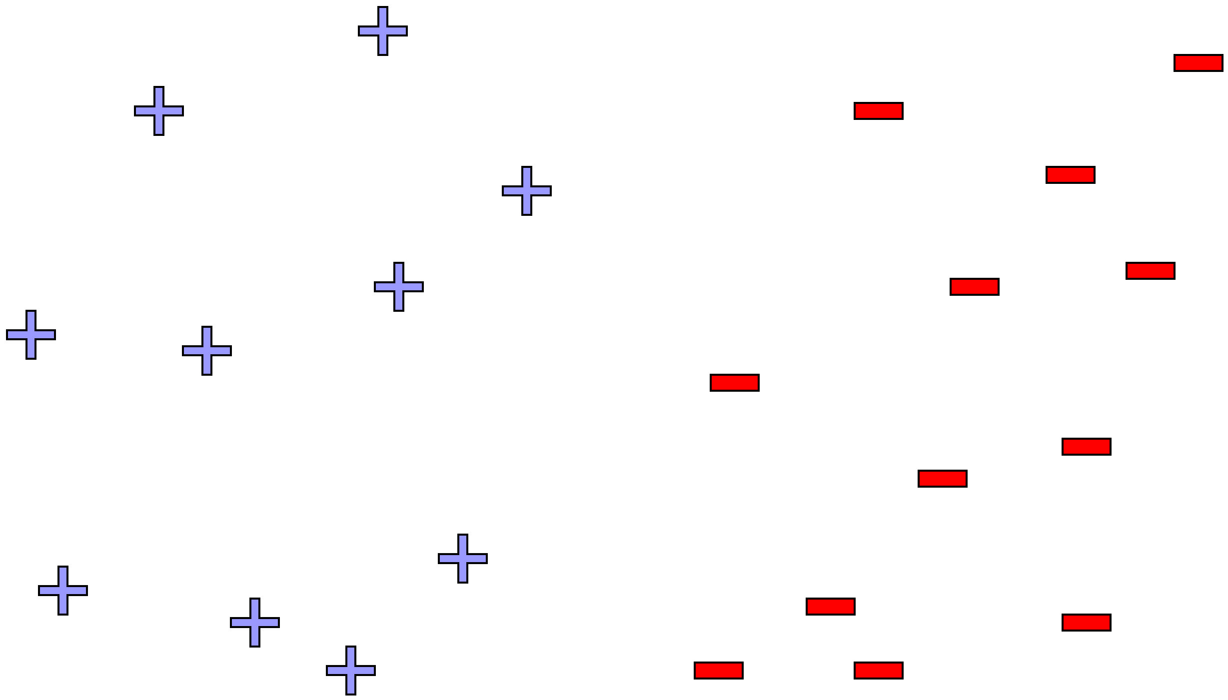
Bad news: no closed-form solution to maximize *J*(**w**)

Good news: convex functions easy to optimize

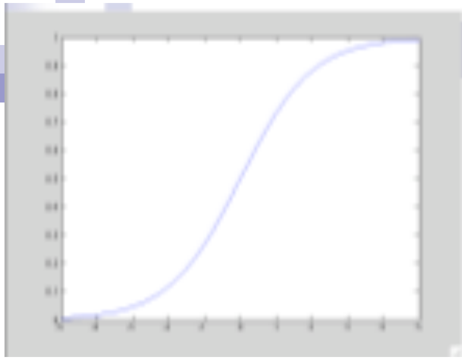
# Linear Separability

$$\arg\min_w \sum_{i=1}^n \log(1 + \exp(-y_i\, x_i^T w))$$

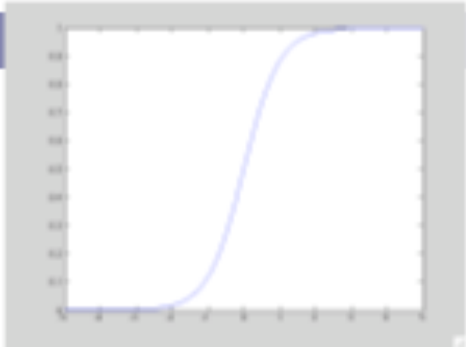
When is this loss small?



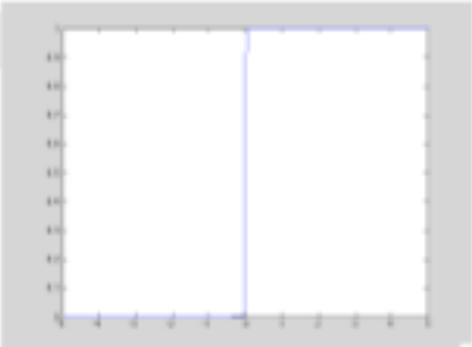
# Large parameters → Overfitting



$$\frac{1}{1+e^{-x}}$$



$$\frac{1}{1+e^{-2x}}$$



$$\frac{1}{1+e^{-100x}}$$

- If data is linearly separable, weights go to infinity
  - In general, leads to overfitting:
- Penalizing high weights can prevent overfitting…

# Regularized Conditional Log Likelihood

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

$$\arg\min_{w,b} \sum_{i=1}^{n} \log\left(1 + \exp(-y_i\left(x_i^T w + b\right))\right) + \lambda ||w||_2^2$$

Be sure to not regularize the offset $b$!

# Gradient Descent

Machine Learning – CSE546
Kevin Jamieson
University of Washington

October 16, 2016

# Machine Learning Problems

- Have a bunch of iid data of the form:

$$\{(x_i, y_i)\}_{i=1}^n \qquad x_i \in \mathbb{R}^d \qquad y_i \in \mathbb{R}$$

- Learning a model's parameters:

$$\sum_{i=1}^n \ell_i(w)$$

Each $\ell_i(w)$ is convex.

# Machine Learning Problems

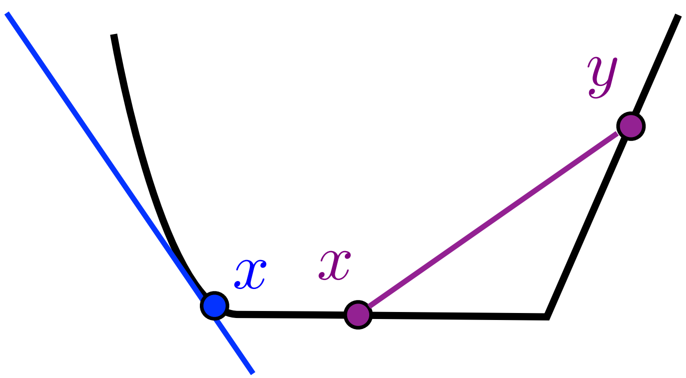
- Have a bunch of iid data of the form:

$$\{(x_i, y_i)\}_{i=1}^n \qquad x_i \in \mathbb{R}^d \qquad y_i \in \mathbb{R}$$

- Learning a model's parameters:

$$\sum_{i=1}^n \ell_i(w)$$

Each $\ell_i(w)$ is convex.




$g$ is a subgradient at $x$ if

$$f(y) \geq f(x) + g^T(y-x)$$

$f$ convex:

$$f(\lambda x + (1-\lambda)y) \leq \lambda f(x) + (1-\lambda) f(y) \qquad \forall x, y, \lambda \in [0,1]$$

$$f(y) \geq f(x) + \nabla f(x)^T (y-x) \qquad \forall x, y$$

# Machine Learning Problems

- Have a bunch of iid data of the form:

$$\{(x_i, y_i)\}_{i=1}^n \qquad x_i \in \mathbb{R}^d \qquad y_i \in \mathbb{R}$$

- Learning a model's parameters:

$$\sum_{i=1}^n \ell_i(w)$$

Each $\ell_i(w)$ is convex.

Logistic Loss: $\ell_i(w) = \log(1 + \exp(-y_i\, x_i^T w))$

Squared error Loss: $\ell_i(w) = (y_i - x_i^T w)^2$

# Least squares

- Have a bunch of iid data of the form:

$$\{(x_i, y_i)\}_{i=1}^n \qquad x_i \in \mathbb{R}^d \qquad y_i \in \mathbb{R}$$

- Learning a model's parameters: $$\sum_{i=1}^n \ell_i(w)$$

Each $\ell_i(w)$ is convex.

Squared error Loss: $\ell_i(w) = (y_i - x_i^T w)^2$

How does software solve: $\frac{1}{2}||\mathrm{X}w - \mathrm{y}||_2^2$

# Least squares

- Have a bunch of iid data of the form:

$$\{(x_i, y_i)\}_{i=1}^n \qquad x_i \in \mathbb{R}^d \qquad y_i \in \mathbb{R}$$

- Learning a model's parameters:

$$\sum_{i=1}^n \ell_i(w)$$

Each $\ell_i(w)$ is convex.

Squared error Loss: $\ell_i(w) = (y_i - x_i^T w)^2$

How does software solve: $\frac{1}{2}||\mathrm{X}w - \mathrm{y}||_2^2$

…its complicated:
(LAPACK, BLAS, MKL…)

Do you need high precision?
Is X column/row sparse?
Is $\widehat{w}_{LS}$ sparse?
Is $\mathrm{X}^T\mathrm{X}$ "well-conditioned"?
Can $\mathrm{X}^T\mathrm{X}$ fit in cache/memory?

# Taylor Series Approximation

- Taylor series in one dimension:

$$f(x+\delta) = f(x) + f'(x)\delta + \tfrac{1}{2}f''(x)\delta^2 + \dots$$

- Gradient descent:

# Taylor Series Approximation

- Taylor series in d dimensions:

$$f(x+v) = f(x) + \nabla f(x)^T v + \frac{1}{2} v^T \nabla^2 f(x) v + \dots$$

- Gradient descent:

# Gradient Descent $f(w) = \frac{1}{2}||\mathrm{X}w - \mathrm{y}||_2^2$

$$w_{t+1} = w_t - \eta \nabla f(w_t)$$

$$\nabla f(w) =$$

# Gradient Descent

$$f(w) = \frac{1}{2}||\mathrm{X}w - \mathrm{y}||_2^2$$

$$w_{t+1} = w_t - \eta \nabla f(w_t)$$

$$(w_{t+1} - w_*) = (I - \eta \mathrm{X}^T \mathrm{X})(w_t - w_*)$$
$$= (I - \eta \mathrm{X}^T \mathrm{X})^{t+1}(w_0 - w_*)$$

Example: $\mathrm{X} = \begin{bmatrix} 10^{-3} & 0 \\ 0 & 1 \end{bmatrix}$ $\mathrm{y} = \begin{bmatrix} 10^{-3} \\ 1 \end{bmatrix}$ $w_0 = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$ $w_* =$

# Taylor Series Approximation

- Taylor series in one dimension:

$$f(x+\delta) = f(x) + f'(x)\delta + \tfrac{1}{2}f''(x)\delta^2 + \dots$$

- Newton's method:

# Taylor Series Approximation

- Taylor series in d dimensions:

$$f(x+v) = f(x) + \nabla f(x)^T v + \frac{1}{2} v^T \nabla^2 f(x) v + \dots$$

- Newton's method:

# Newton's Method $f(w) = \frac{1}{2}||\mathrm{X}w - \mathrm{y}||_2^2$

$$\nabla f(w) =$$

$$\nabla^2 f(w) =$$

$$v_t \text{ is solution to} : \nabla^2 f(w_t) v_t = -\nabla f(w_t)$$

$$w_{t+1} = w_t + \eta v_t$$

# Newton's Method $f(w) = \frac{1}{2}||\mathrm{X}w - \mathrm{y}||_2^2$

$$\nabla f(w) = \mathrm{X}^T(\mathrm{X}w - \mathrm{y})$$

$$\nabla^2 f(w) = \mathrm{X}^T\mathrm{X}$$

$$v_t \text{ is solution to} : \nabla^2 f(w_t) v_t = -\nabla f(w_t)$$

$$w_{t+1} = w_t + \eta v_t$$

For quadratics, Newton's method converges in one step! (Not a surprise, why?)

$$w_1 = w_0 - \eta(\mathrm{X}^T\mathrm{X})^{-1}\mathrm{X}^T(\mathrm{X}w_0 - y) = w_*$$

# General case

In general for Newton's method to achieve $f(w_t) - f(w_*) \leq \epsilon$:

**So why are ML problems overwhelmingly solved by gradient methods?**

Hint: $v_t \text{ is solution to} : \nabla^2 f(w_t) v_t = -\nabla f(w_t)$

# General Convex case $f(w_t) - f(w_*) \leq \epsilon$

Clean converge nice proofs: Bubeck

**Newton's method:**

$$t \approx \log(\log(1/\epsilon))$$

**Gradient descent:**

- f is *smooth* and *strongly convex:* $aI \preceq \nabla^2 f(w) \preceq bI$
- f is *smooth*: $\nabla^2 f(w) \preceq bI$
- f is potentially non-differentiable: $||\nabla f(w)||_2 \leq c$

Nocedal +Wright, Bubeck

**Other:** BFGS, Heavy-ball, BCD, SVRG, ADAM, Adagrad,…

# Revisiting… Logistic Regression

Machine Learning – CSE546
Kevin Jamieson
University of Washington

October 16, 2016

# Loss function: Conditional Likelihood

- Have a bunch of iid data of the form: $\{(x_i, y_i)\}_{i=1}^n \quad x_i \in \mathbb{R}^d, \quad y_i \in \{-1, 1\}$

$$\widehat{w}_{MLE} = \arg\max_w \prod_{i=1}^n P(y_i|x_i, w) \qquad P(Y = y|x, w) = \frac{1}{1 + \exp(-y\, w^T x)}$$

$$f(w) = \arg\min_w \sum_{i=1}^n \log(1 + \exp(-y_i\, x_i^T w))$$

$$\nabla f(w) =$$