Logistics:

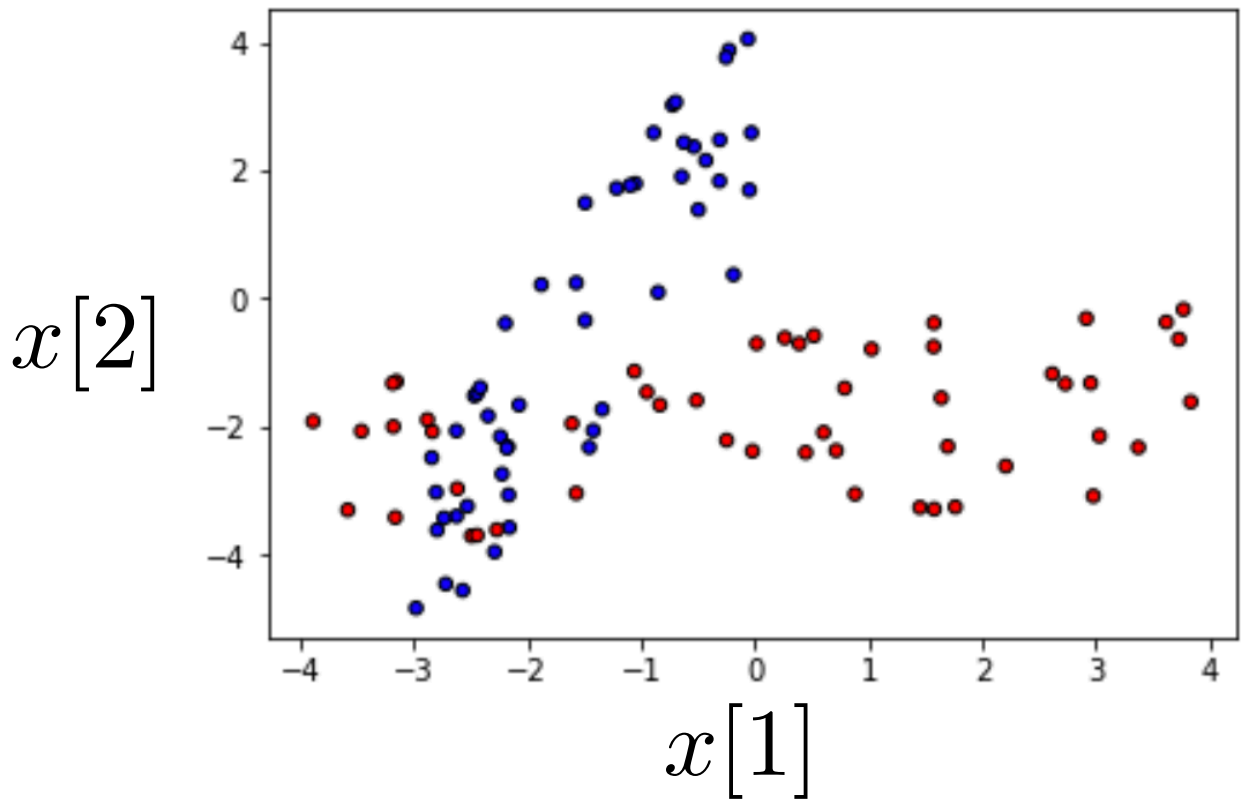
- New Office Hours now on course website. Some on zoom some in-person.

Lecture 12: Classification with logistic regression (continued)

- Regression: label is continuous valued
- Classification: label is discrete valued, e.g., {0,1}

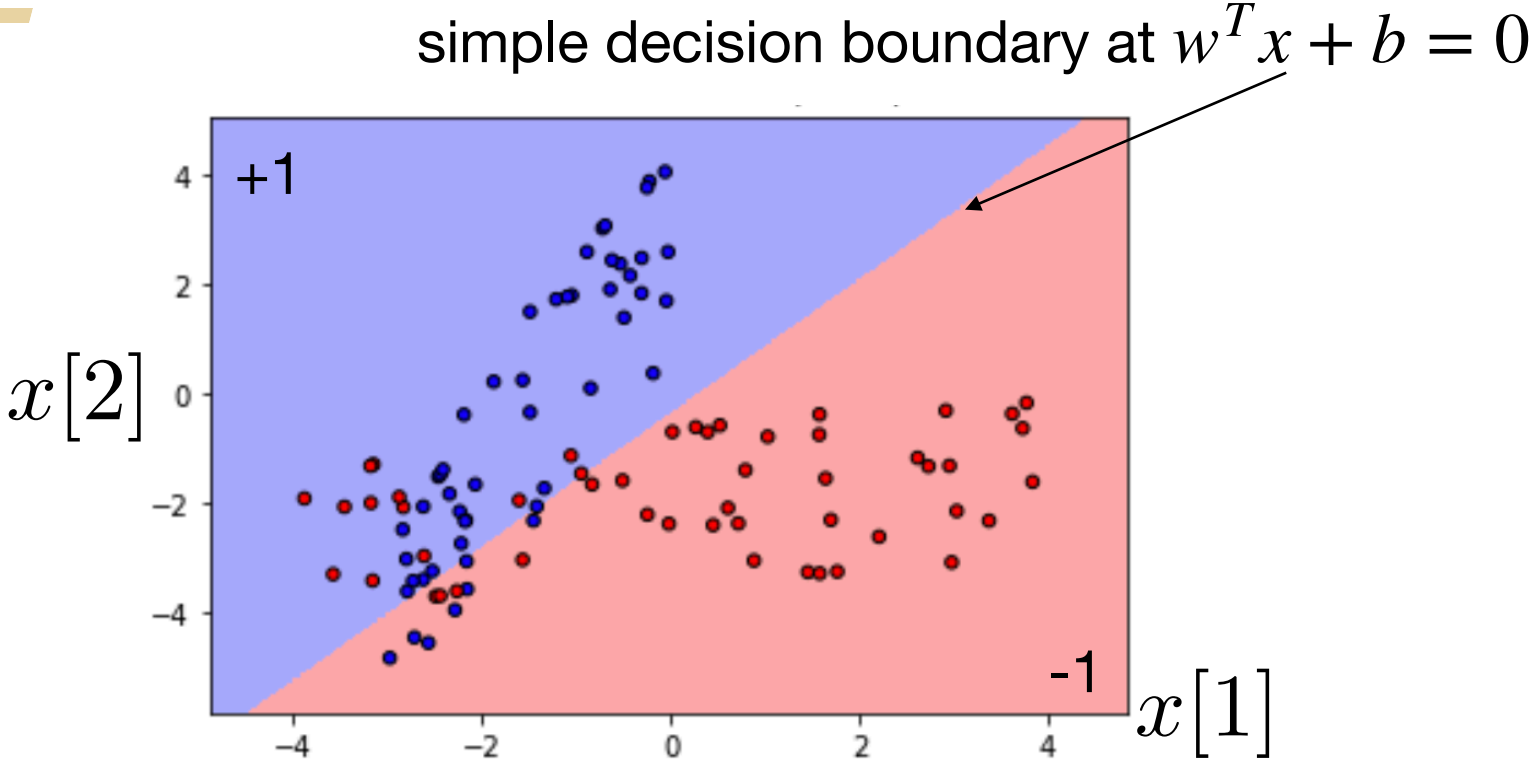
- Note that logistic regression is a classification algorithm not a regression algorithm

Training data for a binary classification problem



- in this example, each input is $x_i \in \mathbb{R}^2$
- Red points have label y_i=-1, blue points have label y_i=1
- We want a predictor that maps any $x \in \mathbb{R}^2$ to a prediction $\hat{y} \in \{-1, +1\}$

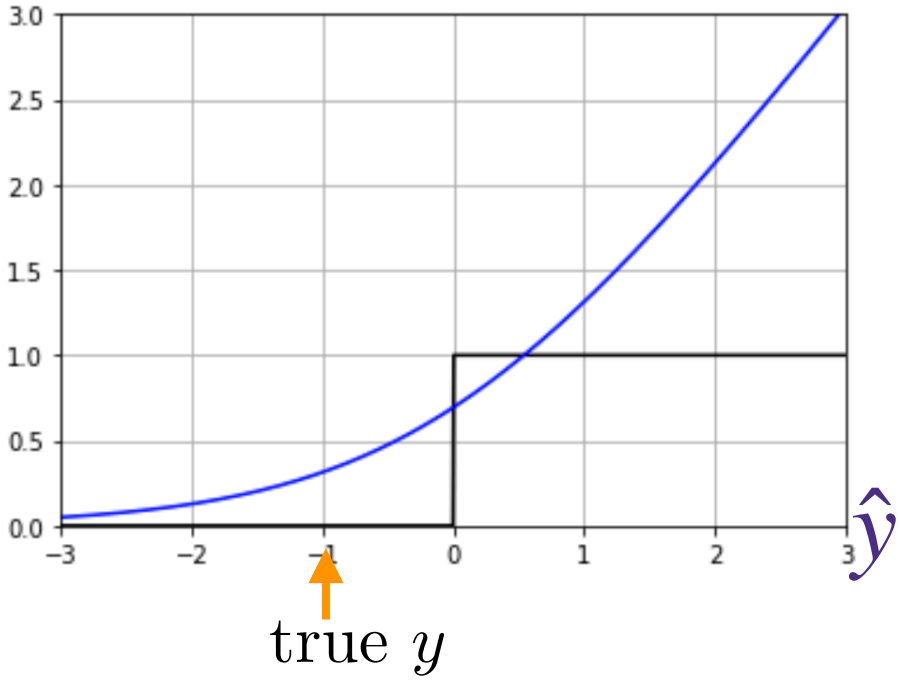
Example: linear classifier trained on 100 samples



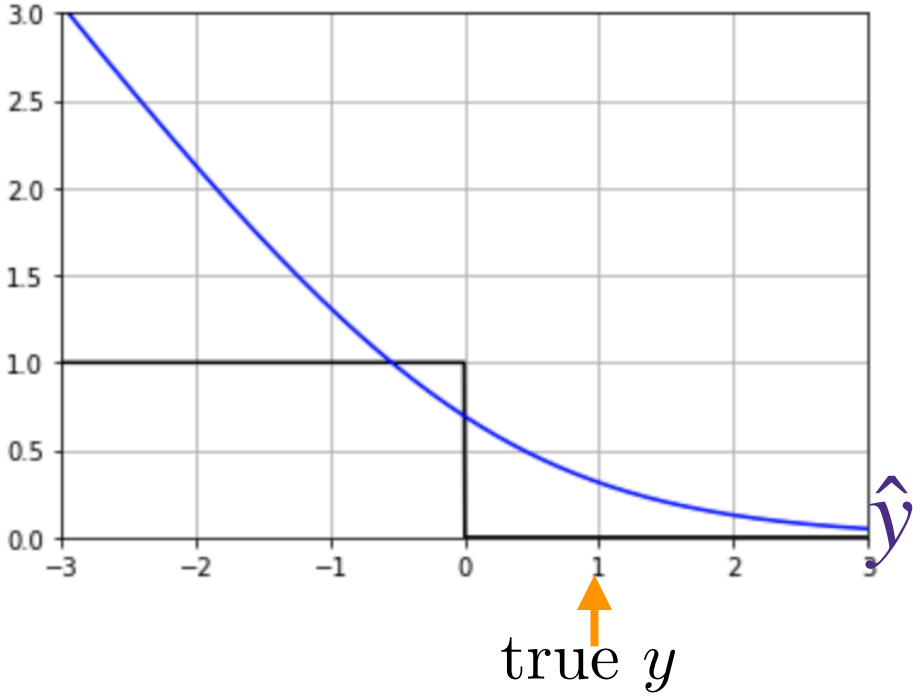
- We fit a linear model: $w_0 + w_1 x[1] + w_2 x[2] = 0.8 - 1.1x[1] + 0.9x[2]$
- predict using $\hat{y} = \text{sign}(0.8 - 1.1x[1] + 0.9x[2])$
- decision boundary is the line (or hyperplane in higher dimensions) defined by
$$0.8 - 1.1x[1] + 0.9x[2] = 0$$
- note that a model $2w^T x + 2b$ has the same predictions as $w^T x + b$
- How do we find such a good linear classifier that fits the data?

Logistic loss $\ell(\hat{y}, y) = \log(1 + e^{-y\hat{y}})$

$\ell(\hat{y}, -1) = \log(1 + e^{\hat{y}})$



$\ell(\hat{y}, +1) = \log(1 + e^{-\hat{y}})$



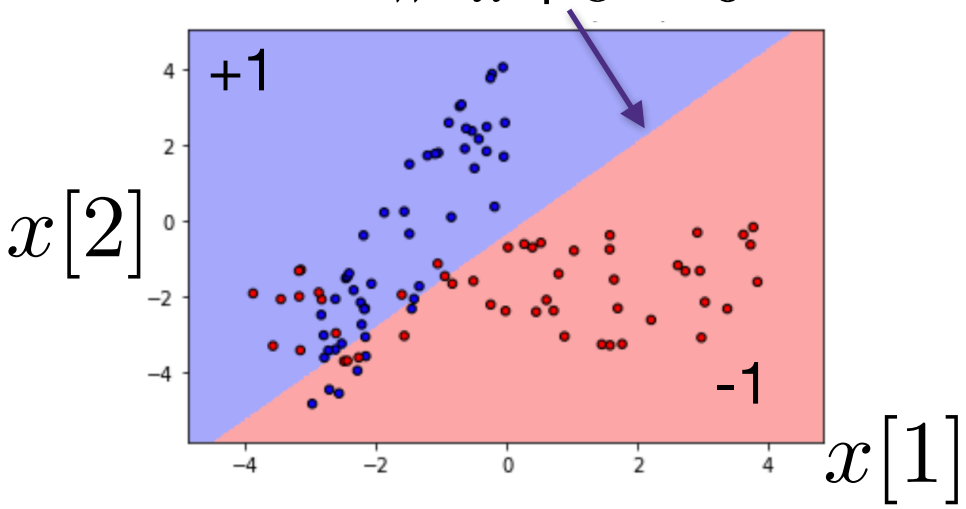
- differentiable and convex in $\hat{y}$
- how do we show $\ell(\,\cdot\,, y)$ is convex?
- approximation of 0-1
- Most popular choice of a loss function for classification problems

Logistic regression for binary classification

- Data $\mathscr{D} = \{(x_i \in \mathbb{R}^d, y_i \in \{-1, +1\})\}_{i=1}^n$
- Model: $\hat{y} = x^T w + b$
- Loss function: logistic loss $\ell(\hat{y}, y) = \log(1 + e^{-y\hat{y}})$
- Optimization: solve for

$$(\hat{b}, \hat{w}) = \arg\min_{b,w} \sum_{i=1}^{n} \log(1 + e^{-y_i(b + x_i^T w)})$$

- As this is a **smooth convex** optimization, it can be solved efficiently using gradient descent
- Prediction: $\text{sign}(b + x^T w)$

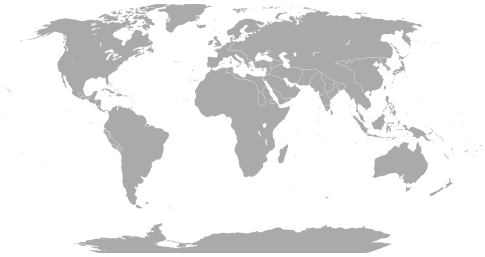


Multi-class regression

How do we encode general categorical data *y*?

- so far, we considered Binary case where there are two categories
- encoding y is simple: {+1,-1}
- We output a scalar prediction $\hat{y} = b + w^T x$, and take the sign

- multi-class classification predicts categorial y
- taking values in $y \in C = \{c_1, \ldots, c_k\}$
- c_j's are called **classes** or **labels**
- examples:



Country of birth
(Argentina, Brazil, USA,...)

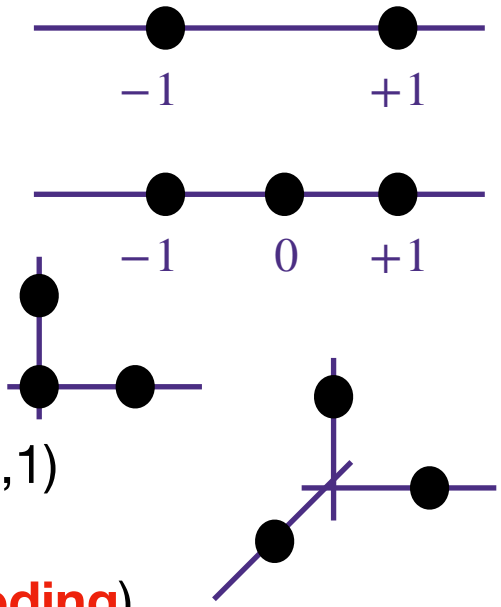
Zipcode
(10005, 98195,...)

All English words

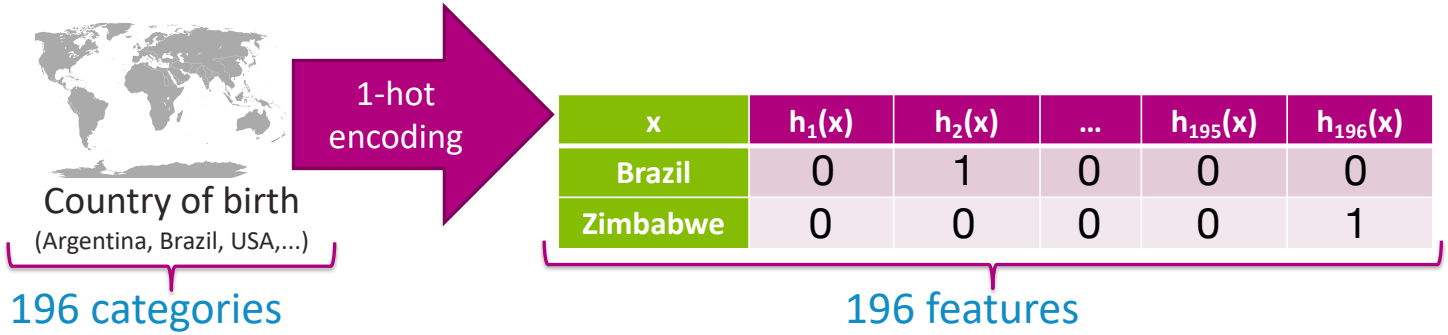
- a **k-class classifier** predicts y given x, but we need to represent categorical y with numerical values

Embedding c_j's in real values

- Since we rely on optimization, which are numerical solvers, we need to **embed** raw categorical c_j's into real valued vectors
- there are many ways to embed categorial data
  - True->1, False->-1, can we use it for all binary classification?
  - Yes->1, Maybe->0, No->-1 can we use it for all ternary classification?
  - Yes->(1,0), Maybe->(0,0), No->(0,1)
  - Apple->(1,0,0), Orange->(0,1,0), Banana->(0,0,1)



- we use **one-hot embedding** (a.k.a. **one-hot encoding**), because it is neutral representation with no domain knowledge
  - each class is a standard basis vector in $k-$dimension



x	$h_1(x)$	$h_2(x)$	...	$h_{195}(x)$	$h_{196}(x)$
Brazil	0	1	0	0	0
Zimbabwe	0	0	0	0	1

Multi-class logistic regression

- **data**: categorical y in $\{c_1, \ldots, c_k\}$ with k categories

  we use **one-hot encoding**, s.t. $y = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$ implies that $y = c_1$

- **model**: linear vector-function makes a linear prediction $\hat{y} \in \mathbb{R}^k$

$$\hat{y}_i = f(x_i) = w^T x_i \quad \in \mathbb{R}^k$$

with model parameter matrix $w \in \mathbb{R}^{d \times k}$ and sample $x_i \in \mathbb{R}^d$

$$f(x_i) = \begin{bmatrix} f_1(x_i) \\ f_2(x_i) \\ \vdots \\ f_k(x_i) \end{bmatrix} = \underbrace{\begin{bmatrix} w_{1,0} & w_{1,1} & w_{1,2} & \cdots \\ w_{2,0} & w_{2,1} & w_{2,2} & \cdots \\ \vdots & & & \\ w_{k,0} & w_{k,1} & w_{k,2} & \cdots \end{bmatrix}}_{w^T} \underbrace{\begin{bmatrix} 1 \\ x_i[1] \\ \vdots \\ x_i[d] \end{bmatrix}}_{x_i} = \begin{bmatrix} w_{1,0} + w_{1,1}x_i[1] + w_{1,2}x_i[2] + \cdots \\ w_{2,0} + w_{2,1}x_i[1] + w_{2,2}x_i[2] + \cdots \\ \vdots \\ w_{k,0} + w_{k,1}x_i[1] + w_{k,2}x_i[2] + \cdots \end{bmatrix}$$

$$w = \begin{bmatrix} w[:,1] & w[:,2] & \cdots & w[:,k] \end{bmatrix}$$

Model parameter for category 1

2 equivalent approaches for binary Logistic regression

- What we learned so far:
  single model parameter $w \in \mathbb{R}^d$

$$\mathbb{P}(y_i = +1 \,|\, x_i) = \frac{1}{1 + e^{-w^T x_i}} = \frac{e^{w^T x_i}}{1 + e^{w^T x_i}}$$

$$\mathbb{P}(y_i = -1 \,|\, x_i) = 1 - \mathbb{P}(y_i = +1 \,|\, x_i)$$

- Another approach: two parameters $w_1, w_2 \in \mathbb{R}^d$
  - $w_1^T x$ represents how much category 1 is more likely at a data point x (relatively to $w_2^T x$)
  - $w_2^T x$ represents how much category 2 is more likely at a data point x

$$\mathbb{P}(y_i = 1 \,|\, x_i) = \frac{e^{w_1^T x_i}}{e^{w_1^T x_i} + e^{w_2^T x_i}}$$

$$\mathbb{P}(y_i = 2 \,|\, x_i) = \frac{e^{w_2^T x_i}}{e^{w_1^T x_i} + e^{w_2^T x_i}}$$

Maximum Likelihood Estimator

$$\text{maximize}_w \; \frac{1}{n} \sum_{i=1}^{n} \log(\mathbb{P}(y_i \,|\, x_i))$$

$$\text{maximize}_{w \in \mathbb{R}^d} \; \frac{1}{n} \sum_{i=1}^{n} \log\left(\frac{e^{y_i w^T x_i}}{1 + e^{y_i w^T x_i}} \right)$$

Representing the two categories as {-1,+1} helps simplify the notations, where we used the fact that $\frac{e^{-w^T x}}{1 + e^{-w^T x}} = 1 - \frac{e^{w^T x}}{1 + e^{w^T x}}$

$$\text{maximize}_{w_1, w_2 \in \mathbb{R}^d} \frac{1}{n} \sum_{i=1}^{n} \left\{ \log\left(\frac{e^{w_{y_i}^T x_i}}{e^{w_1^T x_i} + e^{w_2^T x_i}} \right) \right\}$$

Representing the two categories as {1,2} is inline with the notation for the parameters w_1 and w_2, and we simplify the numerator using w_{y_i}

- Do these two approaches give different solutions?
- The second approach is more symmetric, and hence generalizes to larger classes

Logistic regression

2 classes

$$\mathbb{P}(y_i = 1 \,|\, x_i) = \frac{e^{w_1^T x_i}}{e^{w_1^T x_i} + e^{w_2^T x_i}}$$

$$\mathbb{P}(y_i = 2 \,|\, x_i) = \frac{e^{w_2^T x_i}}{e^{w_1^T x_i} + e^{w_2^T x_i}}$$

k classes

$$\mathbb{P}(y_i = 1 \,|\, x_i) = \frac{e^{w[:,1]^T x_i}}{e^{w[:,1]^T x_i} + \cdots + e^{w[:,k]^T x_i}}$$

$$\vdots$$

$$\mathbb{P}(y_i = k \,|\, x_i) = \frac{e^{w[:,k]^T x_i}}{e^{w[:,1]^T x_i} + \cdots + e^{w[:,k]^T x_i}}$$

Maximum Likelihood Estimator

$$\text{maximize}_w \ \frac{1}{n} \sum_{i=1}^{n} \log(\mathbb{P}(y_i \,|\, x_i))$$

$$\text{maximize}_{w_1, w_2 \in \mathbb{R}^d} \frac{1}{n} \sum_{i=1}^{n} \left\{ \log\left(\frac{e^{w_{y_i}^T x_i}}{e^{w_1^T x_i} + e^{w_2^T x_i}} \right) \right\}$$

$$\text{maximize}_{w \in \mathbb{R}^{d \times k}} \frac{1}{n} \sum_{i=1}^{n} \log\left(\frac{e^{w[:,y_i]^T x_i}}{\sum_{j=1}^{k} e^{w[:,j]^T x_i}} \right)$$

Gradient Descent

- how are we going to find the solution for

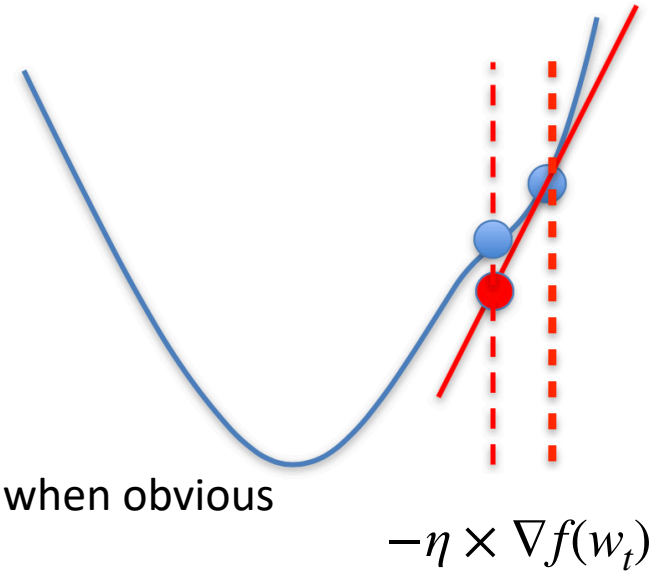
$$\arg\min_{b,w} \sum_{i=1}^{n} \ell(b + w^T x_i, y_i)$$

- e.g., Lasso, Logistic Regression do not have closed form solution for

$$\nabla_{b,w} \mathscr{L}(b, w) = 0$$

Gradient descent

Example of a general non-convex $f(w)$



- Initialize: $w_0 = 0$
- **For** t=0,1,2,…
  - $w_{t+1} \leftarrow w_t - \eta \cdot \nabla_w f(w_t)$

I will omit w in ∇_w when obvious

- Learning rate or step-size
- A hyper-parameter to be chosen by the analyst
- Can be fixed over the iterations,
  or can also change, in which case we use η_t
  to emphasize the fact that it changes over iteration t

Running example: linear regression

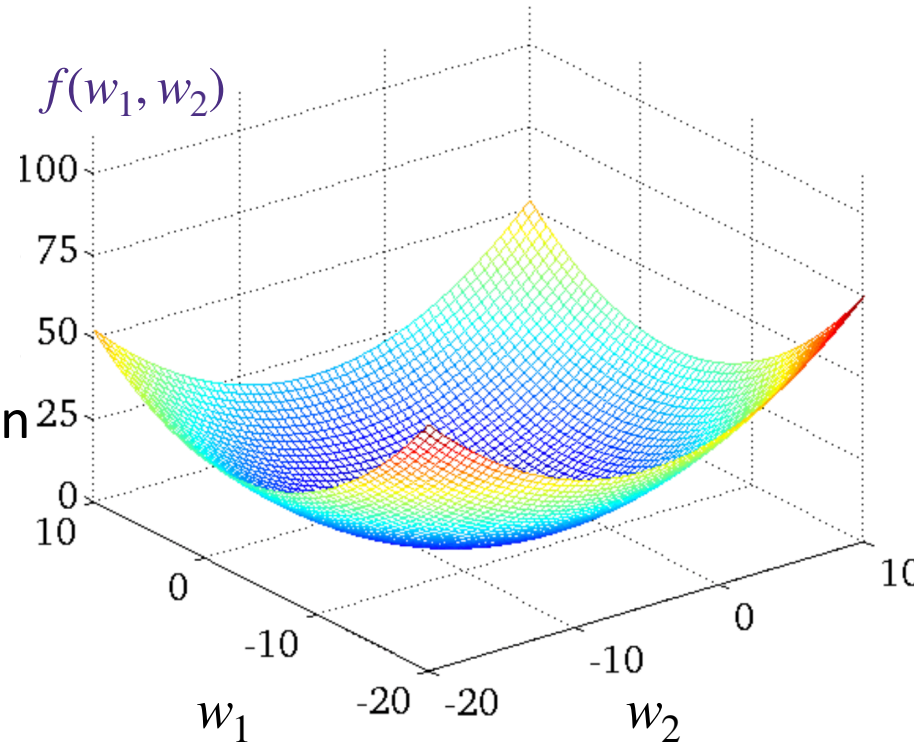
- **Given data:**

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

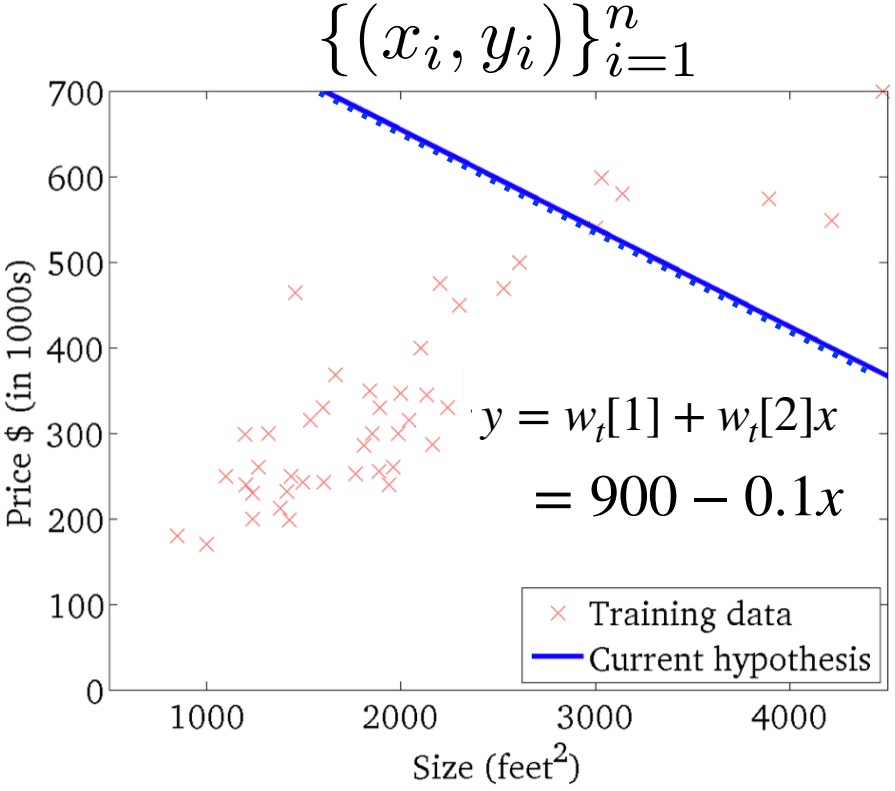
- **Learning model parameters:**

$$\hat{w}_{\text{LS}} = \arg\min_{w \in \mathbb{R}^d} \underbrace{\|\mathbf{y} - \mathbf{X}w\|_2^2}_{f(w)}$$

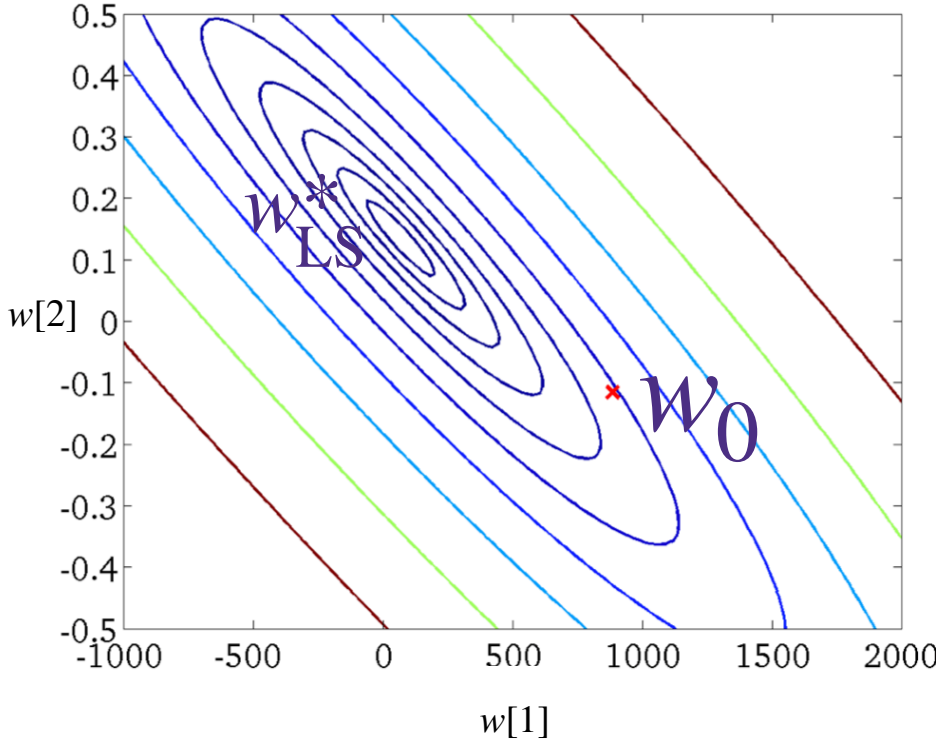
- Although we know the optimal solution in a closed form, we will use this as a running example to understand GD



- $w_0 = (900, -0.1)$
- **For** t=0,1,2,…
  - $w_{t+1} \leftarrow w_t - \eta \cdot \nabla_w f(w_t)$



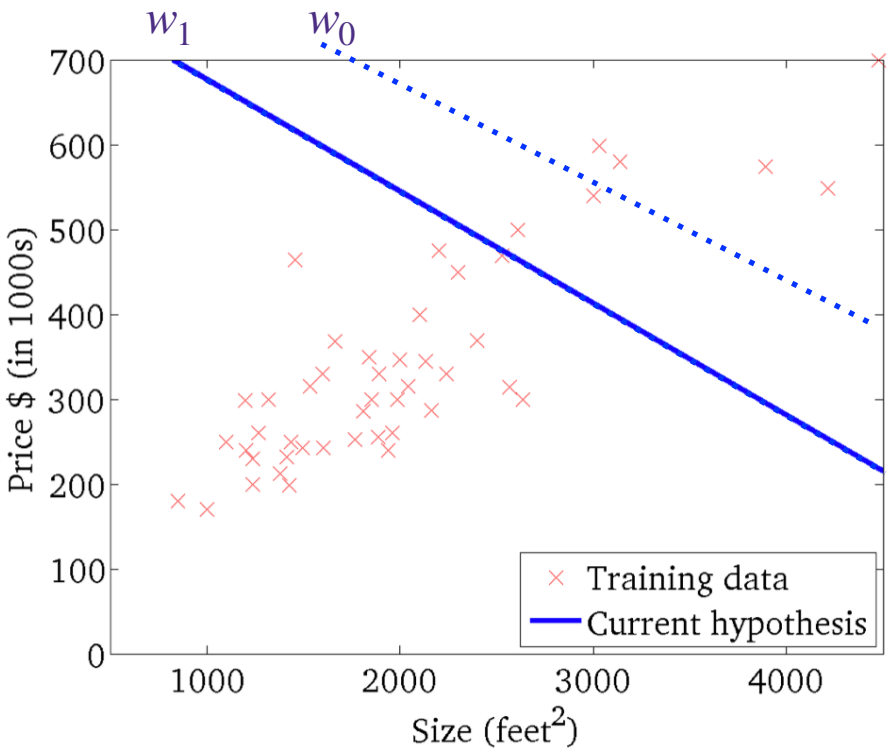
Evolution of the predictor



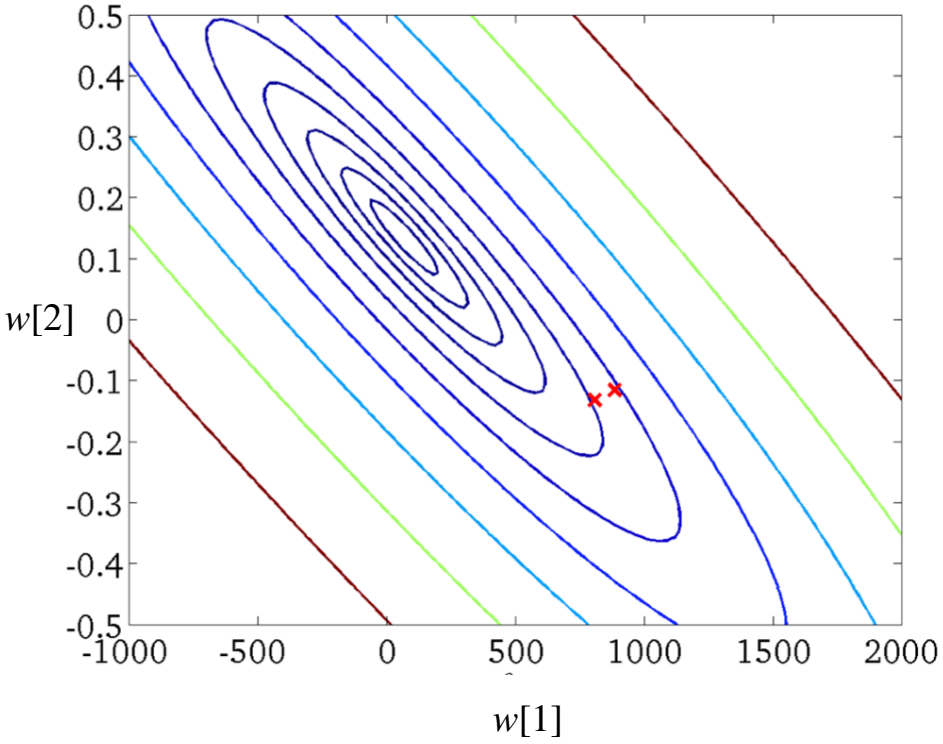
GD dynamics in the Parameter space

- Which direction will the GD move?

- $w_0 = (900, -0.1)$
- **For** t=0,1,2,…
  - $w_{t+1} \leftarrow w_t - \eta \cdot \nabla_w f(w_t)$

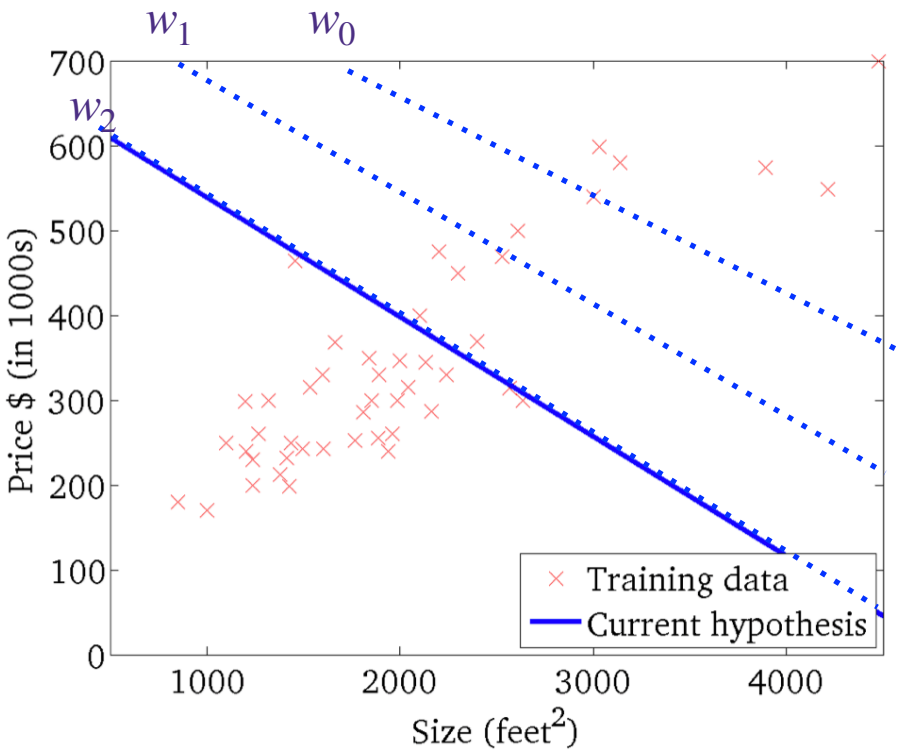


Evolution of the predictor

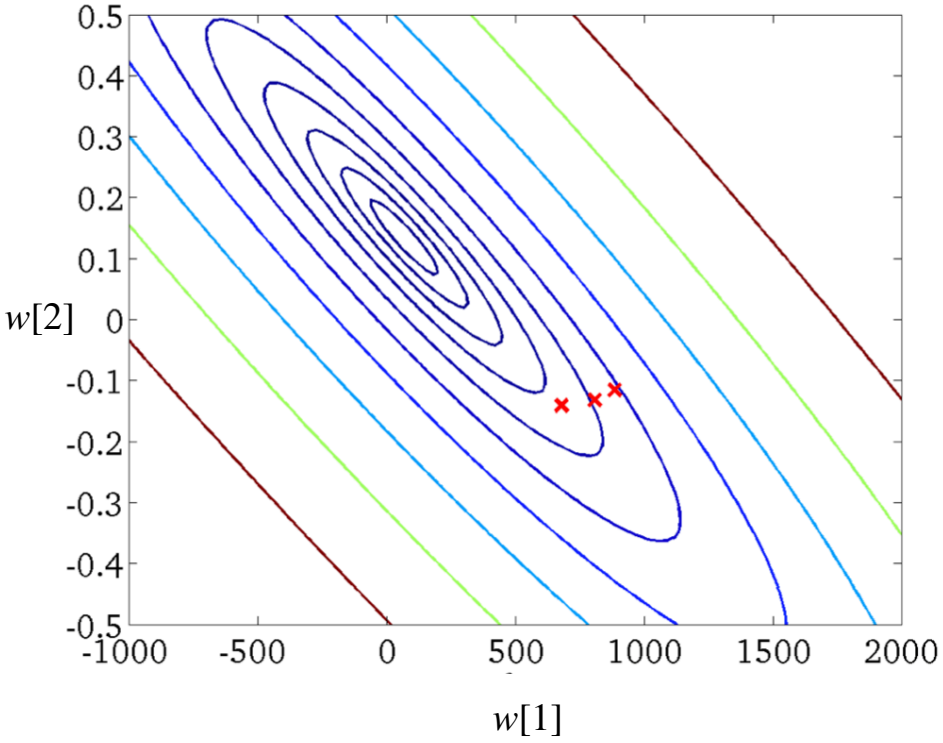


GD dynamics in the Parameter space

- $w_0 = (900, -0.1)$
- **For** t=0,1,2,…
  - $w_{t+1} \leftarrow w_t - \eta \cdot \nabla_w f(w_t)$

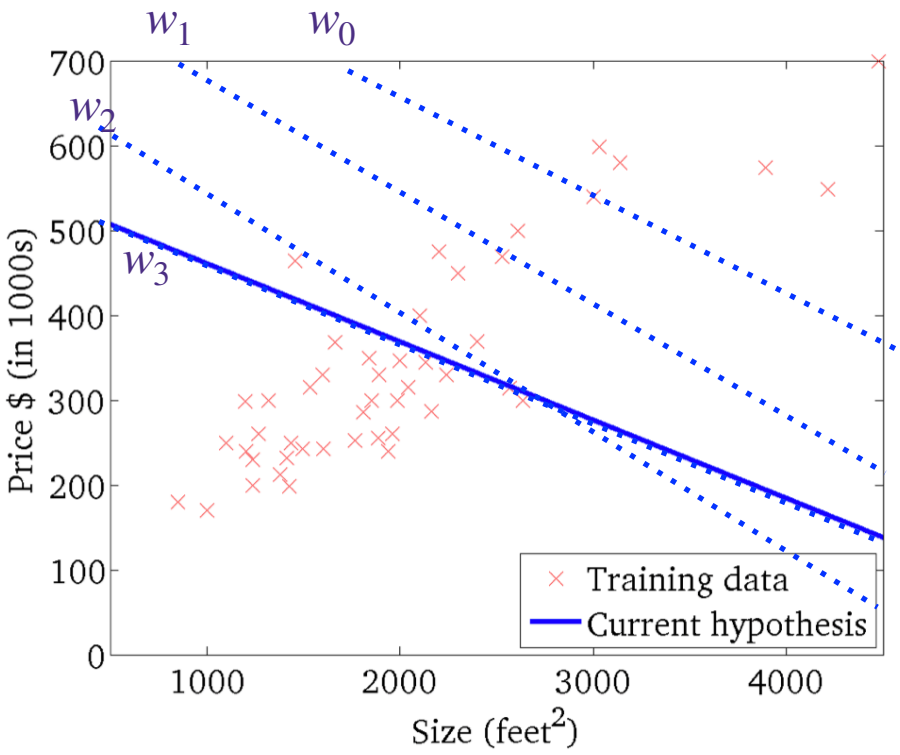


Evolution of the predictor

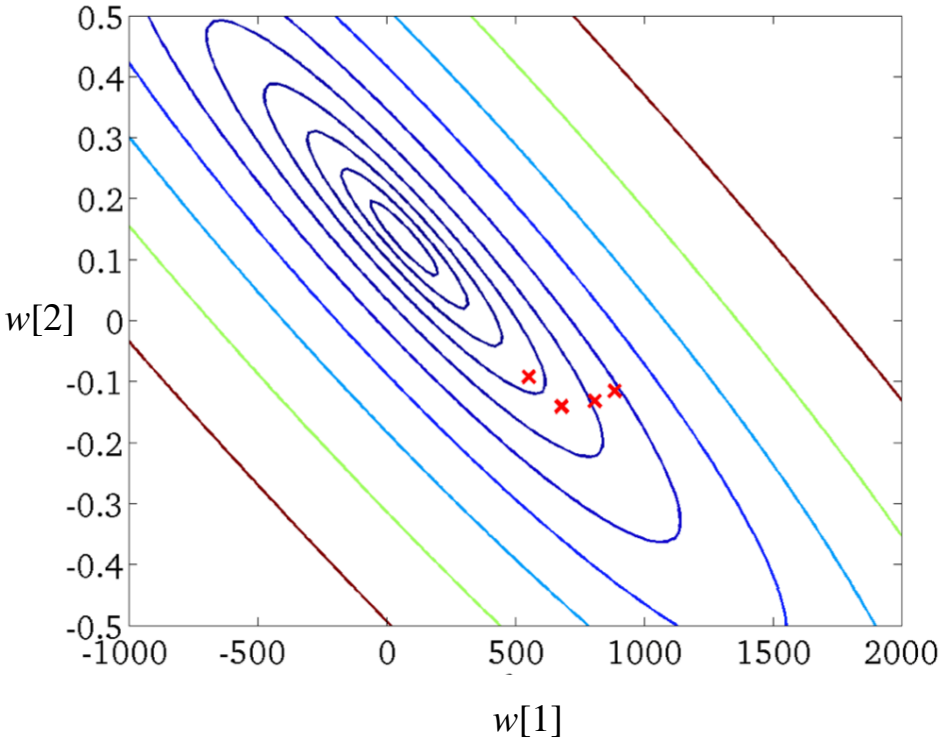


GD dynamics in the Parameter space

- $w_0 = (900, -0.1)$
- **For** t=0,1,2,…
  - $w_{t+1} \leftarrow w_t - \eta \cdot \nabla_w f(w_t)$

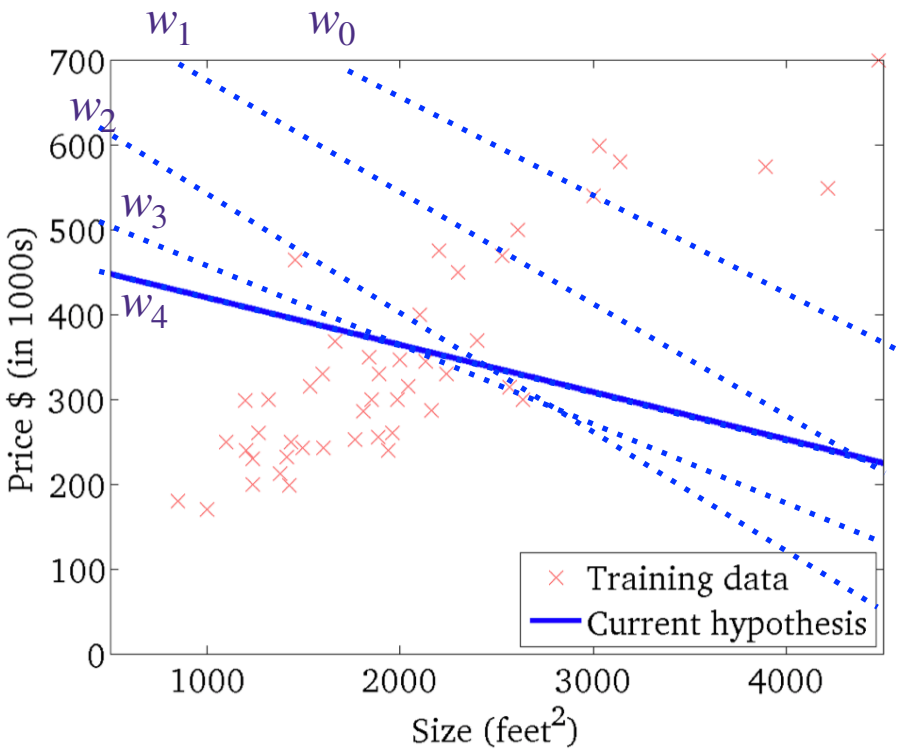


Evolution of the predictor

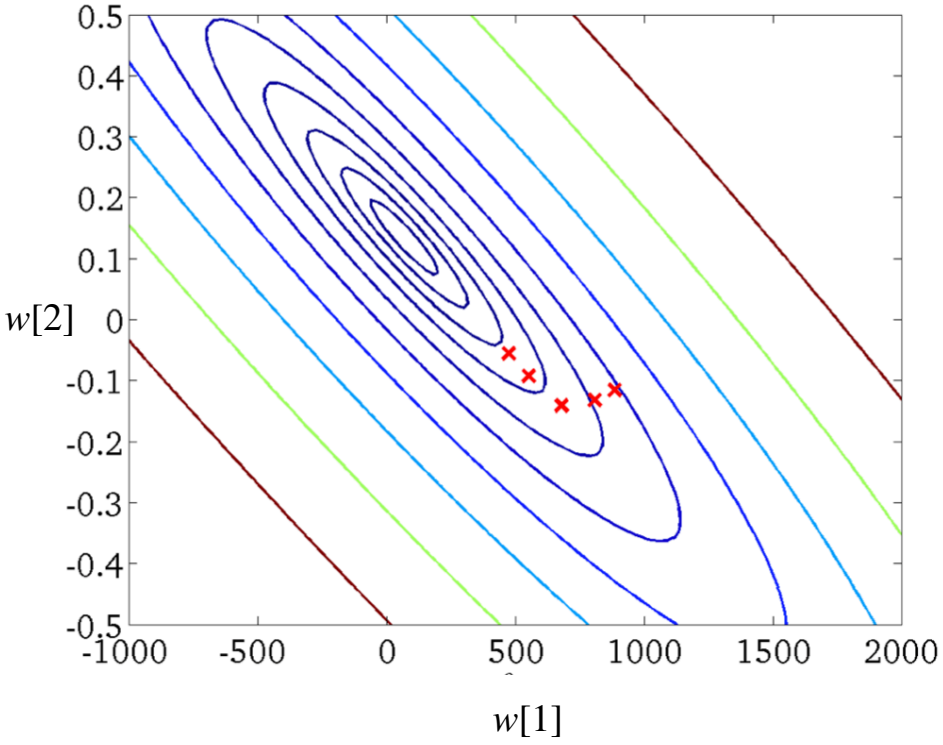


GD dynamics in the Parameter space

- $w_0 = (900, -0.1)$
- **For** t=0,1,2,…
  - $w_{t+1} \leftarrow w_t - \eta \cdot \nabla_w f(w_t)$

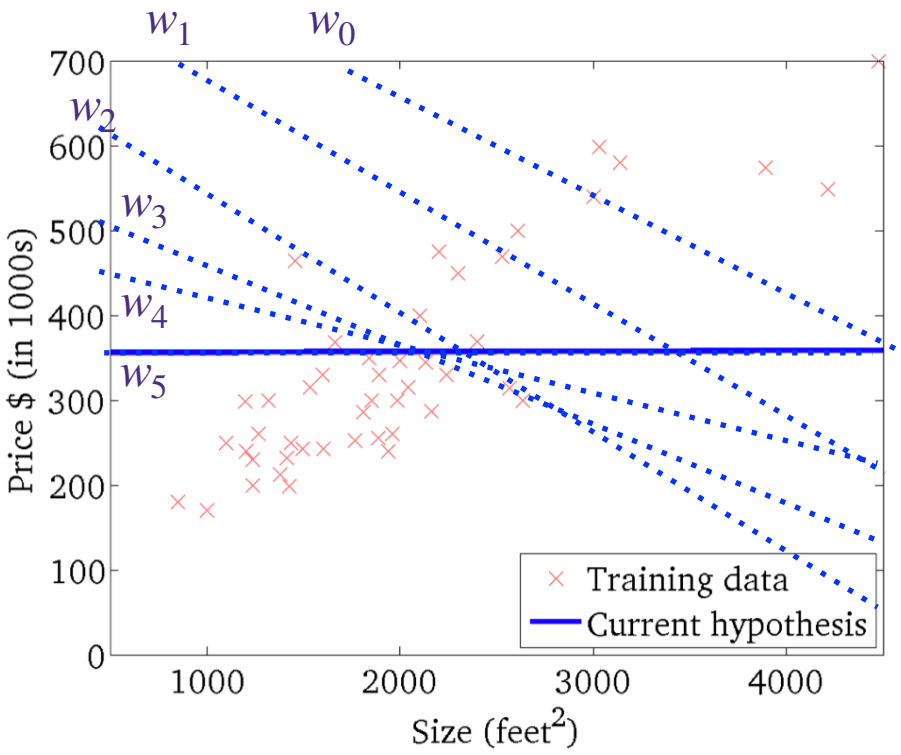


Evolution of the predictor

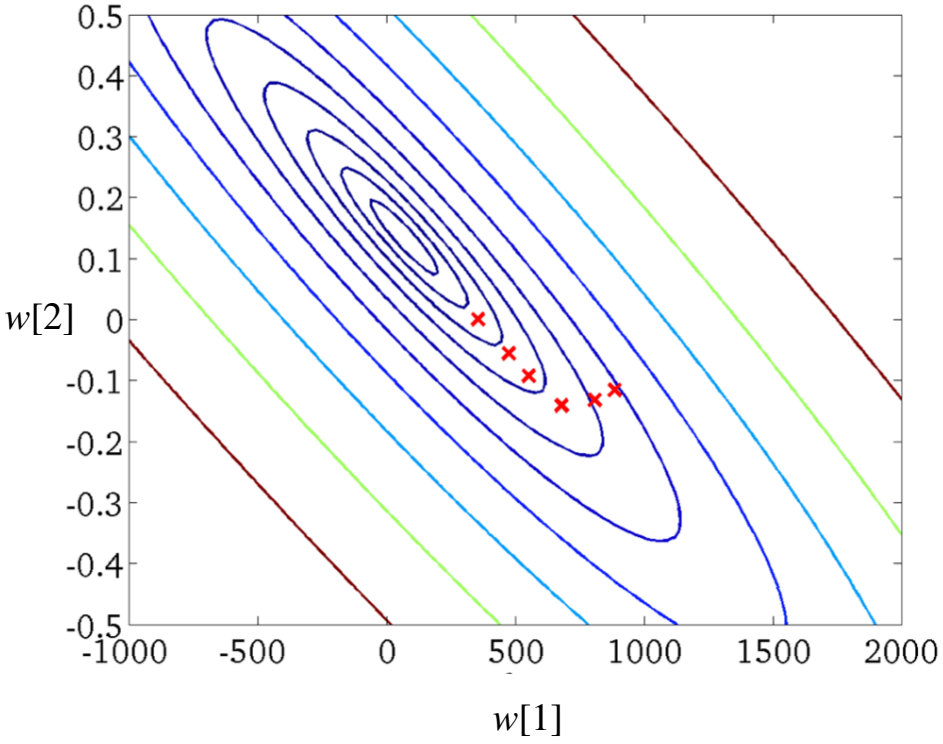


GD dynamics in the Parameter space

- $w_0 = (900, -0.1)$
- **For** t=0,1,2,…
  - $w_{t+1} \leftarrow w_t - \eta \cdot \nabla_w f(w_t)$

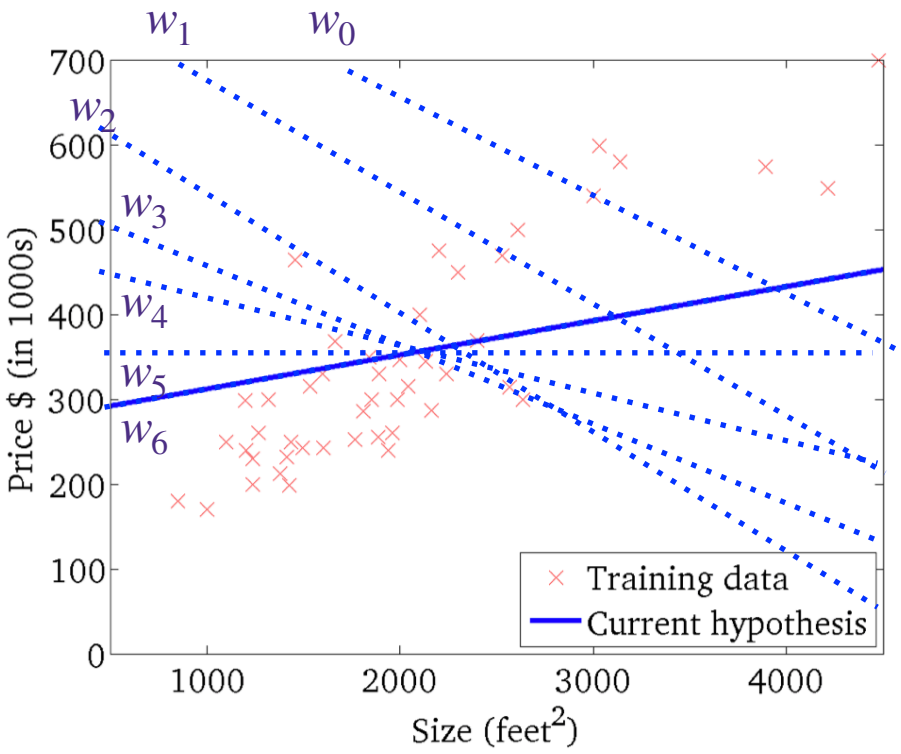


Evolution of the predictor

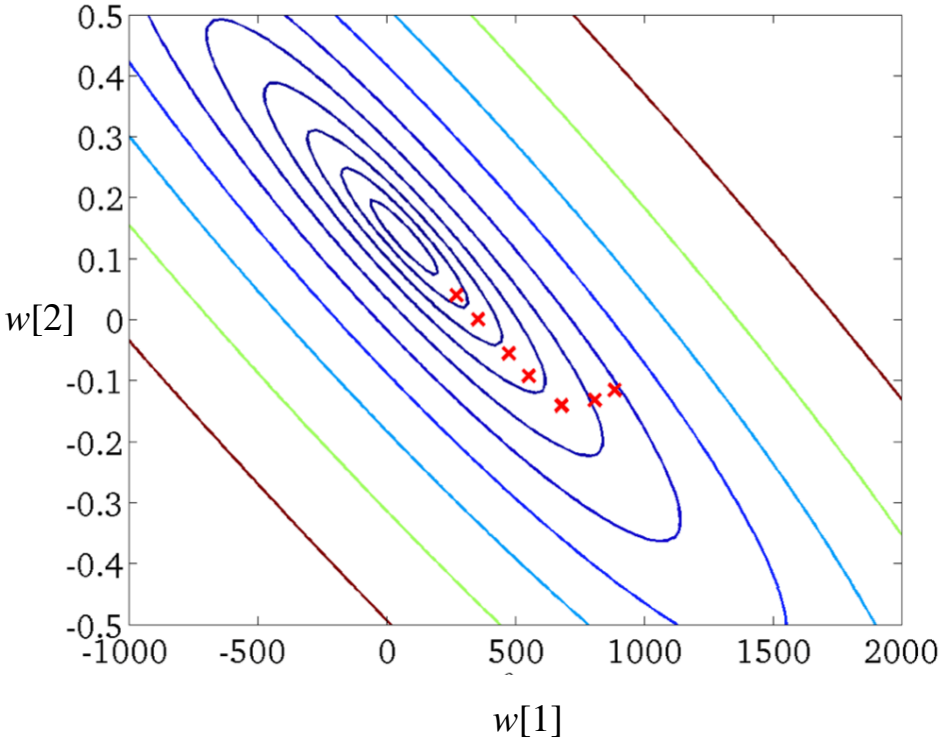


GD dynamics in the Parameter space

- $w_0 = (900, -0.1)$
- **For** t=0,1,2,…
  - $w_{t+1} \leftarrow w_t - \eta \cdot \nabla_w f(w_t)$

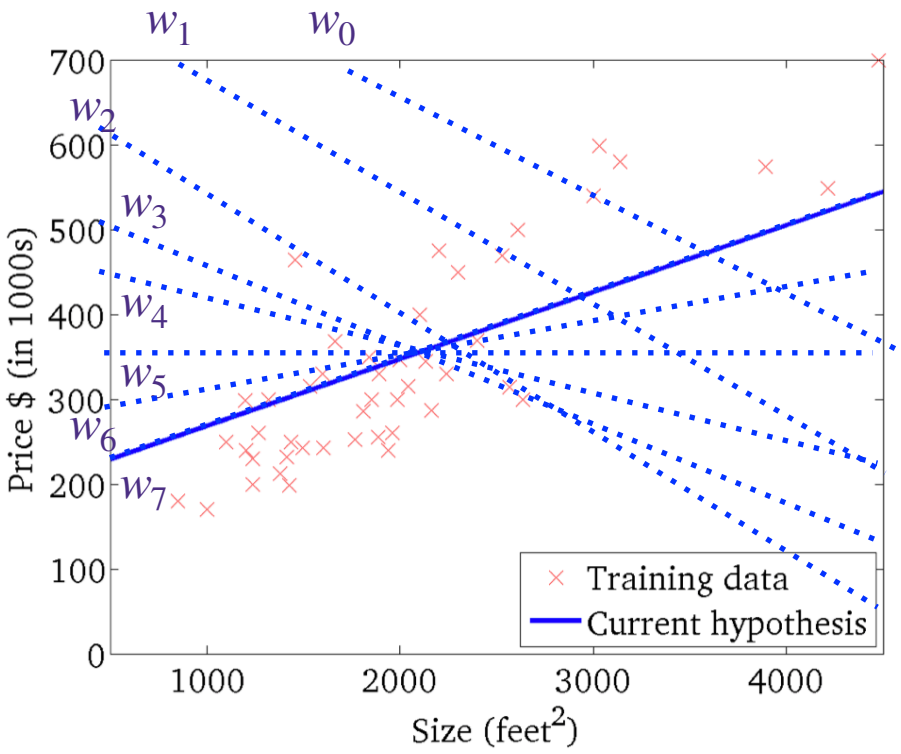


Evolution of the predictor

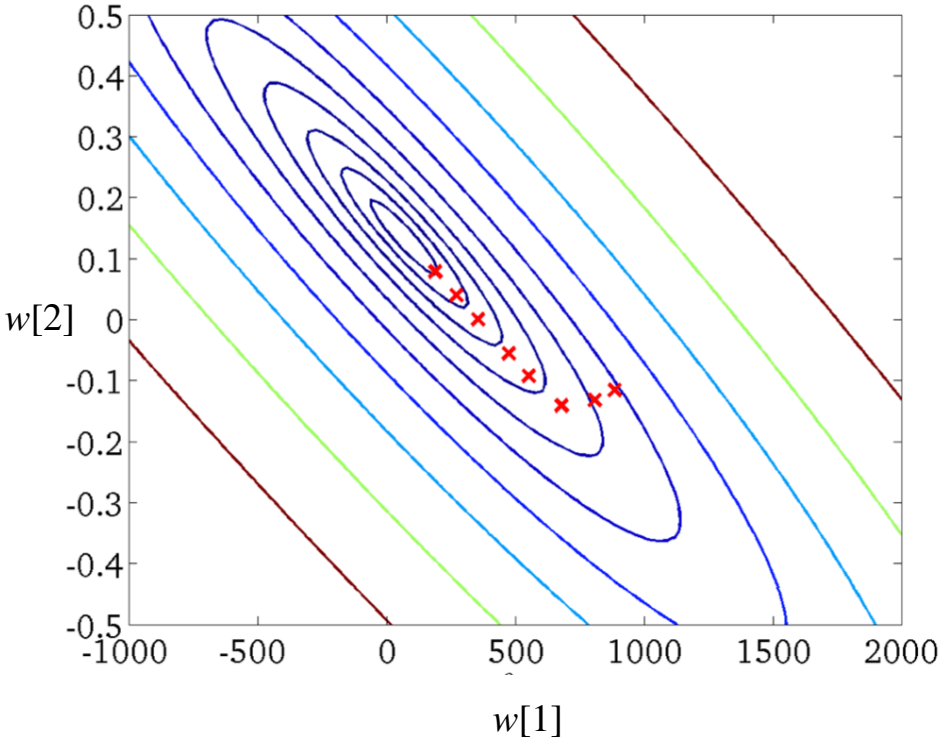


GD dynamics in the Parameter space

- $w_0 = (900, -0.1)$
- **For** t=0,1,2,…
  - $w_{t+1} \leftarrow w_t - \eta \cdot \nabla_w f(w_t)$

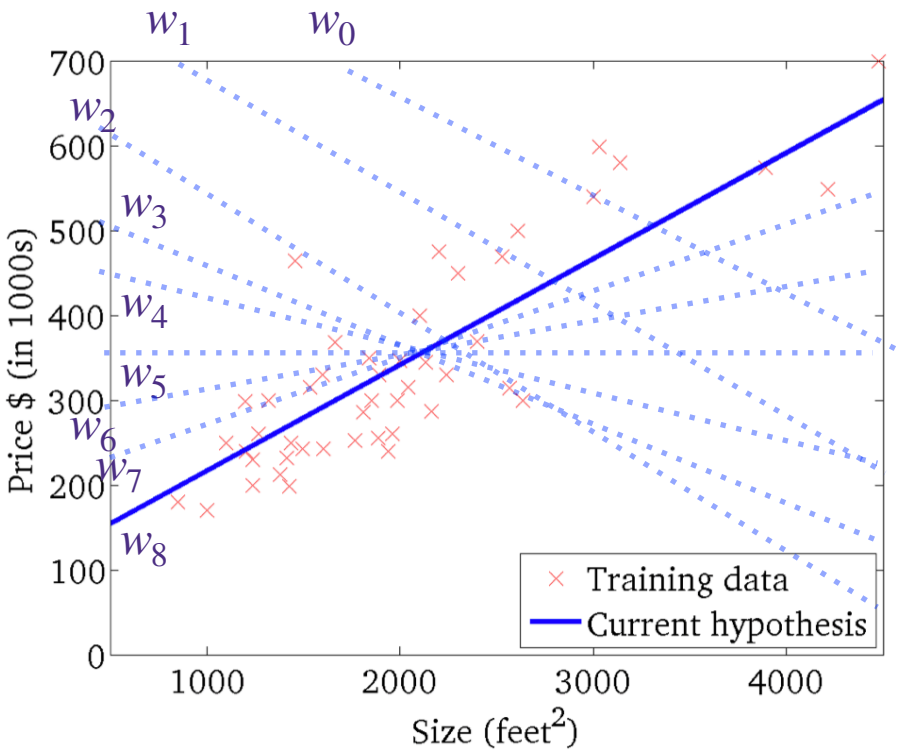


Evolution of the predictor

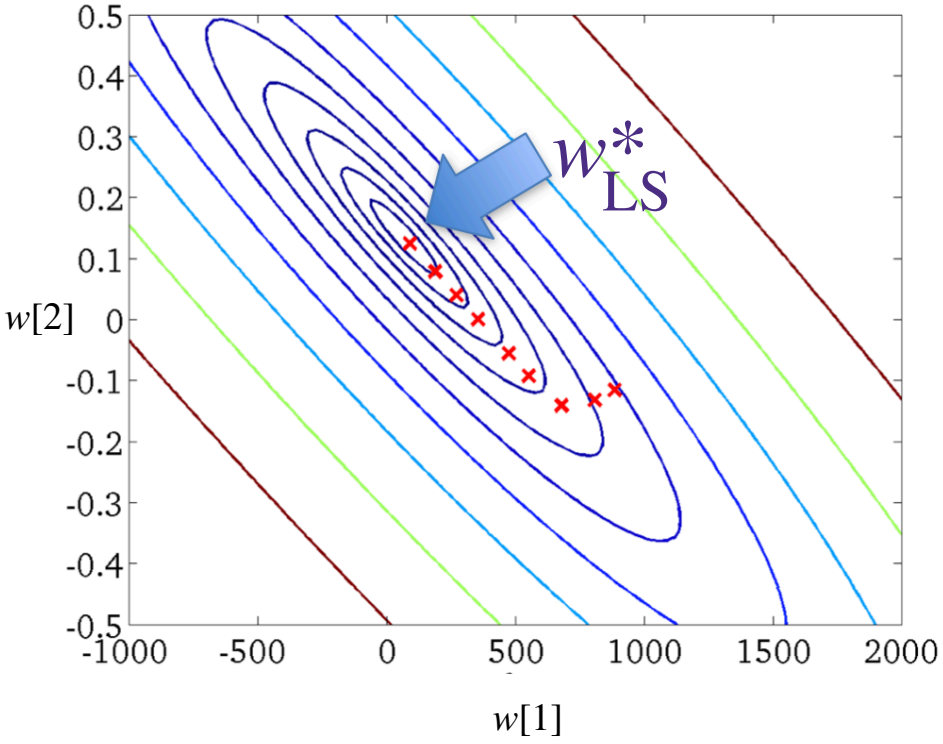


GD dynamics in the Parameter space

- $w_0 = (900, -0.1)$
- **For** t=0,1,2,…
  - $w_{t+1} \leftarrow w_t - \eta \cdot \nabla_w f(w_t)$



Evolution of the predictor



GD dynamics in the Parameter space

Gradient descent for linear regression

- In this example of linear regression, we can derive exactly the gradient descent trajectory
- Initialize: $w_0 = 0$
- **For** t=0,1,2,…
  - $w_{t+1} \leftarrow w_t - \eta \cdot \nabla_w f(w_t)$

For linear regression, we have

$$\hat{w}_{\mathrm{LS}} = \arg\min_{w \in \mathbb{R}^d} \underbrace{\|\mathbf{y} - \mathbf{X}w\|_2^2}_{f(w)}$$

Gradient descent for linear regression

- Initialize: $w_0 = 0$
- **For** t=0,1,2,…
  - $w_{t+1} \leftarrow w_t - \eta \cdot \nabla_w f(w_t)$

For linear regression, we have

$$\hat{w}_{\text{LS}} = \arg\min_{w \in \mathbb{R}^d} \underbrace{\|\mathbf{y} - \mathbf{X}w\|_2^2}_{f(w)}$$

$$\nabla f(w_t) = -2\mathbf{X}^T(\mathbf{y} - \mathbf{X}w_t)$$

$$w_{t+1} = w_t + \eta 2\mathbf{X}^T(\mathbf{y} - \mathbf{X}w_t) = (\mathbf{I} - 2\eta\mathbf{X}^T\mathbf{X})w_t + 2\eta\mathbf{X}^T\mathbf{y}$$

Let the least-squares solution be $w^* = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$

$$\begin{aligned} w_{t+1} - w^* &= (\mathbf{I} - 2\eta\mathbf{X}^T\mathbf{X})w_t + 2\eta\mathbf{X}^T\mathbf{y} - w^* \\ &= (\mathbf{I} - 2\eta\mathbf{X}^T\mathbf{X})(w_t - w^*) + 2\eta\mathbf{X}^T\mathbf{y} - 2\eta\mathbf{X}^T\mathbf{X}w^* \\ &= (\mathbf{I} - 2\eta\mathbf{X}^T\mathbf{X})(w_t - w^*) \end{aligned}$$

Gradient Descent (GD) for Linear Regression (LR)

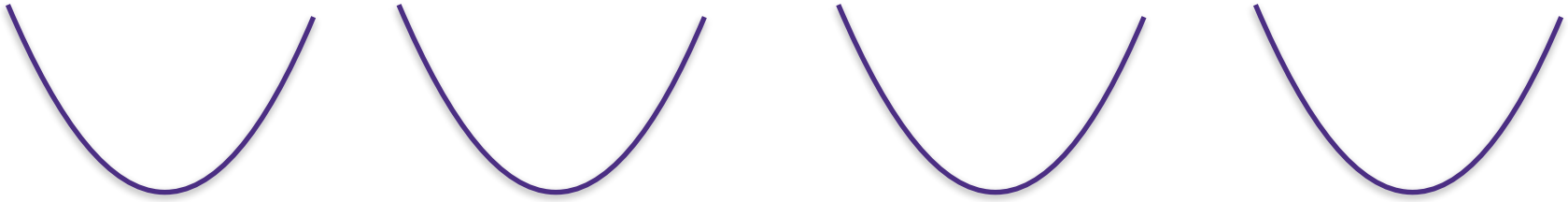
- We use this analytical derivation of GD for LR to understand how the choice of step size impacts the algorithm

$$w_{t+1} = w_t - \eta \nabla f(w_t) \quad \Longrightarrow \quad w_{t+1} - w^* \; = \; (\mathbf{I} - 2\eta \mathbf{X}^T \mathbf{X})(w_t - w^*)$$

Gradient descent for linear regression

$$w_{t+1} = w_t - \eta \nabla f(w_t) \quad \Longrightarrow \quad \begin{aligned} w_{t+1} - w^* &= (\mathbf{I} - 2\eta \mathbf{X}^T\mathbf{X})(w_t - w^*) \\ &= (\mathbf{I} - 2\eta \mathbf{X}^T\mathbf{X})^2(w_{t-1} - w^*) \\ &\vdots \\ &= (\mathbf{I} - 2\eta \mathbf{X}^T\mathbf{X})^{t+1}(w_0 - w^*) \end{aligned}$$

In one dimension, $2\mathbf{X}^T\mathbf{X} = a$ is a scalar, and $w_{t+1} - w^* = (1 - \eta a)^{t+1}(w_0 - w^*)$



$0 < \eta < 1/a$ $\quad$ $\eta = 1/a$ $\quad$ $1/a < \eta < 2/a$ $\quad$ $\eta > 2/a$

$1 > 1 - \eta a > 0$ $\quad$ $1 - \eta a = 0$ $\quad$ $0 > 1 - \eta a > -1$ $\quad$ $-1 > 1 - \eta a$

Gradient descent for linear regression

$$w_{t+1} = w_t - \eta \nabla f(w_t) \quad \Longrightarrow \quad w_{t+1} - w^* \; = \; (\mathbf{I} - 2\eta \mathbf{X}^T \mathbf{X})^{t+1}(w_0 - w^*)$$

- In multi dimensions, **eigenvalues** of $\mathbf{X}^T\mathbf{X}$ are important
  (you will see why I consider eigenvalues of $\mathbf{X}^T\mathbf{X}$,
  instead of $\mathbf{I} - 2\eta\mathbf{X}^T\mathbf{X}$ in couple of slides)
- Let the eigenvalue decomposition of $\mathbf{I} - 2\eta\mathbf{X}^T\mathbf{X}$ be $Q^{-1}DQ$

Gradient descent for linear regression

$$w_{t+1} = w_t - \eta \nabla f(w_t) \quad \Longrightarrow \quad w_{t+1} - w^* \;=\; (\mathbf{I} - 2\eta \mathbf{X}^T\mathbf{X})^{t+1}(w_0 - w^*)$$

- In multi dimensions, **eigenvalues** of $\mathbf{X}^T\mathbf{X}$ are important
  (you will see why I consider eigenvalues of $\mathbf{X}^T\mathbf{X}$,
  instead of $\mathbf{I} - 2\eta\mathbf{X}^T\mathbf{X}$ in couple of slides)
- Let the eigenvalue decomposition of $\mathbf{I} - 2\eta\mathbf{X}^T\mathbf{X}$ be $Q^{-1}DQ$

  Then, $w_{t+1} - w^* \;=\; (Q^{-1}DQ)^{t+1}(w_0 - w^*)$

$$= \underbrace{Q^{-1}DQQ^{-1}DQ\cdots Q^{-1}DQ}_{t+1 \text{ times}}(w_0 - w^*)$$

$$= Q^{-1}D^{t+1}Q\,(w_0 - w^*)$$

$$Q(w_{t+1} - w^*) \;=\; D^{t+1}Q\,(w_0 - w^*)$$

Gradient descent for linear regression

$$Q(w_{t+1} - w^*) = D^{t+1} Q (w_0 - w^*)$$

- Where eigenvalue decomposition of $\mathbf{I} - 2\eta \mathbf{X}^T \mathbf{X}$ is $Q^{-1} D Q$
- Let $Q = \begin{bmatrix} - & q_1^T & - \\ - & q_2^T & - \\ & \vdots & \end{bmatrix}$, then the above multi-dimensional dynamics of GD can be decomposed into multiple 1-d dynamics we saw before
- In direction q_1, the error decreases multiplicatively according to

$$q_1^T (w_{t+1} - w^*) = D_{11}^{t+1} q_1^T (w_0 - w^*)$$

$$q_2^T (w_{t+1} - w^*) = D_{22}^{t+1} q_2^T (w_0 - w^*)$$

$$\vdots$$

Gradient descent for linear regression

$$w_{t+1} = w_t - \eta \nabla f(w_t) \implies w_{t+1} - w^* = (\mathbf{I} - 2\eta \mathbf{X}^T \mathbf{X})^{t+1}(w_0 - w^*)$$

$$\implies Q(w_{t+1} - w^*) = D^{t+1} Q\,(w_0 - w^*)$$

$$q_1^T(w_{t+1} - w^*) = D_{11}^{t+1}\, q_1^T(w_0 - w^*)$$

$$q_2^T(w_{t+1} - w^*) = D_{22}^{t+1}\, q_2^T(w_0 - w^*)$$

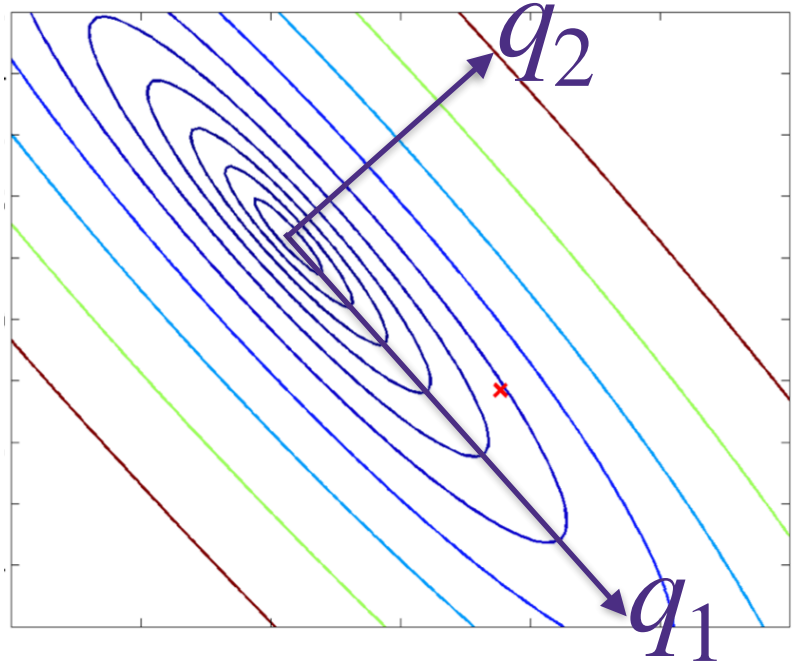
- For example suppose, the step size η is chosen such that

In direction q_1

$0 < D_{11} < 1$

In direction q_2

$-1 < D_{22} < 0$

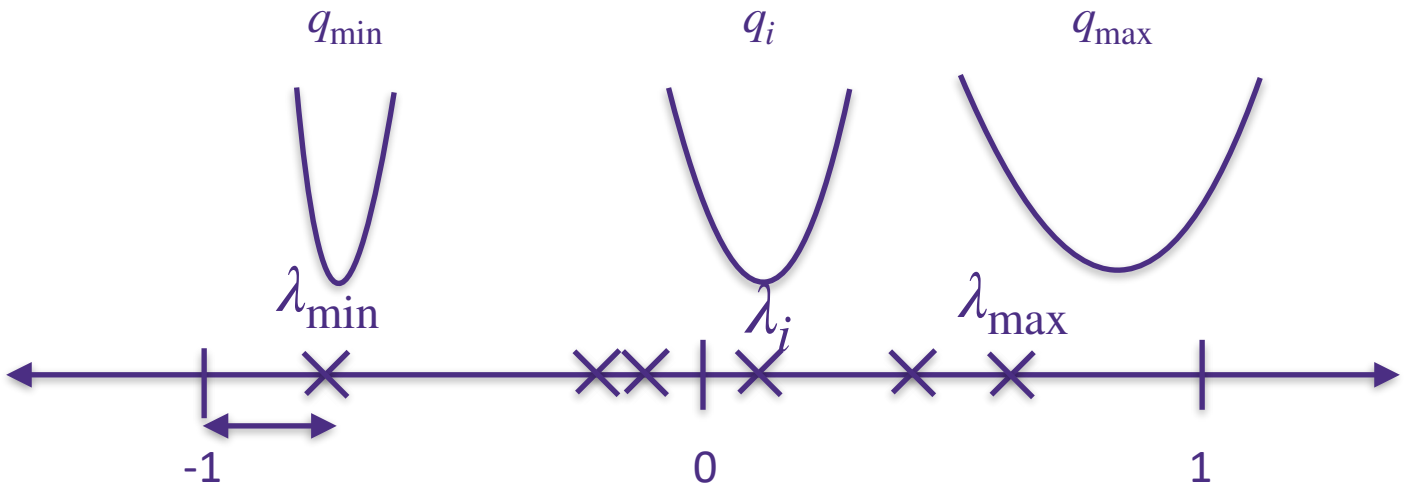


Gradient descent for linear regression

- Note that $D_{ii} := \lambda_i(\mathbf{I} - 2\eta\mathbf{X}^T\mathbf{X})$ is defined as the i-th Eigen value of the matrix $\mathbf{I} - 2\eta\mathbf{X}^T\mathbf{X}$
- Recall that in each eigen direction error evolves as

$$q_i^T(w_{t+1} - w^*) \;=\; \left(\lambda_i(\mathbf{I} - 2\eta\mathbf{X}^T\mathbf{X})\right)^{t+1} q_i^T(w_0 - w^*)$$

- We want the error to decay fast in all directions, whose bottleneck are the largest and the smallest eigen values: $\lambda_{\min}(\mathbf{I} - 2\eta\mathbf{X}^T\mathbf{X})$ and $\lambda_{\max}(\mathbf{I} - 2\eta\mathbf{X}^T\mathbf{X})$



We want to choose the learning rate η such that

$$-1 \ll \lambda_{\min}(\mathbf{I} - 2\eta\mathbf{X}^T\mathbf{X}) \leq \cdots \leq \lambda_{\max}(\mathbf{I} - 2\eta\mathbf{X}^T\mathbf{X}) \ll 1$$

Gradient descent for linear regression

- We use linear algebra to answer the question:
  how does the rate of error decay change with step size η
- Recall that in each eigen direction error decays as

$$q_i^T(w_{t+1} - w^*) \;=\; \lambda_i(\mathbf{I} - 2\eta\mathbf{X}^T\mathbf{X})^{t+1}\, q_i^T(w_0 - w^*)$$

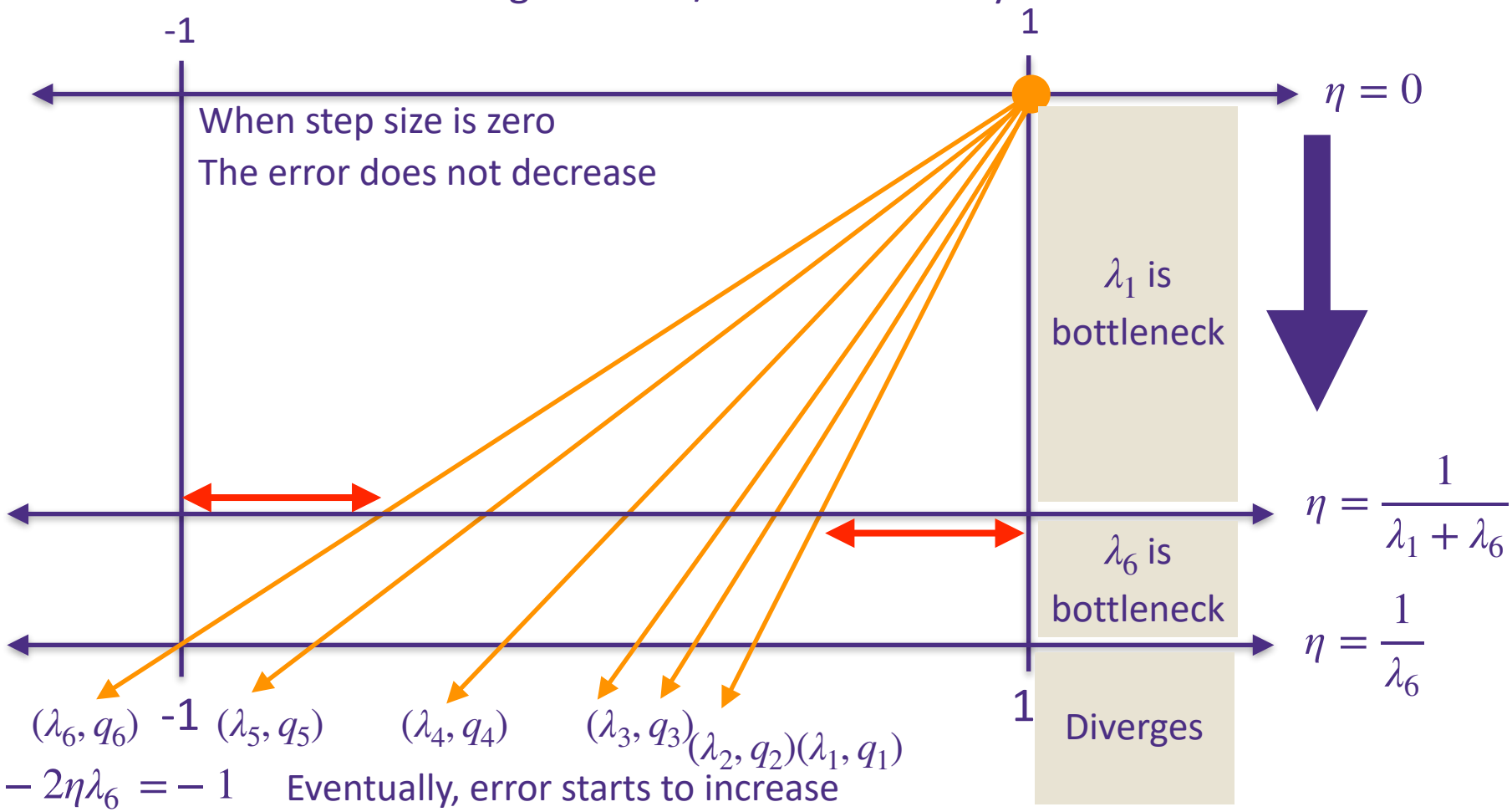
- I will not prove the facts that $\|q_i\|_2 = 1$ and $q_i^T q_j = 0$

Claim: $\lambda_i(\mathbf{I} - 2\eta\mathbf{X}^T\mathbf{X}) = 1 - 2\eta\lambda_i(\mathbf{X}^T\mathbf{X})$

Claim: $\lambda_i(\mathbf{X}^T\mathbf{X}) \geq 0$

Gradient descent for linear regression

- Claim: $\lambda_i(\mathbf{I} - 2\eta\mathbf{X}^T\mathbf{X}) = 1 - 2\eta\lambda_i(\mathbf{X}^T\mathbf{X})$, and $\lambda_i(\mathbf{X}^T\mathbf{X}) \geq 0$
- We plot $\{ 1 - 2\eta\lambda_i(\mathbf{X}^T\mathbf{X}) \}$ for increasing step sizes, the largest $\left| 1 - 2\eta\lambda_i(\mathbf{X}^T\mathbf{X}) \right|$ is the bottleneck determining how fast/slow error decays



Gradient descent for logistic regression

- Now we know how to find the global minimum of a logistic regression problem, numerically

Loss function: Conditional Likelihood

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

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

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

$$\nabla f(w) = \sum_{i=1}^n \frac{1}{1 + \exp(-y_i x_i^T w)} \exp(-y_i x_i^T w)(-y_i x_i)$$

What is known for Gradient descent

- $f(\cdot)$ is L-smooth if $\|\nabla f(w) - \nabla f(v)\|_2 \leq L\|w - v\|_2$ for all $w, v \in \mathbb{R}^d$
- $f(\cdot)$ is μ-strongly convex if $f(w) \geq f(v) + \nabla f(v)^T(w - v) + \frac{\mu}{2}\|w - v\|_2^2$
- For L-smooth functions, with a fixed step size $\eta < 1/L$
  - if $f(w)$ is convex,

$$f(w_t) - f(w^*) \leq \frac{\|w_0 - w^*\|_2^2}{2\eta t}$$

  - if $f(w)$ is μ-strongly convex,

$$f(w_t) - f(w^*) \leq \left(1 - \eta\mu\right)^t \left(f(w_0) - f(w^*)\right)$$

- What can we do for non-smooth function $f(w)$?
  - for example, LASSO

$$\hat{w}_{\text{Lasso}} = \arg\min_{w \in \mathbb{R}^d} \underbrace{\|\mathbf{y} - \mathbf{X}w\|_2^2 + \lambda\|w\|_1}_{f(w)}$$

Questions?