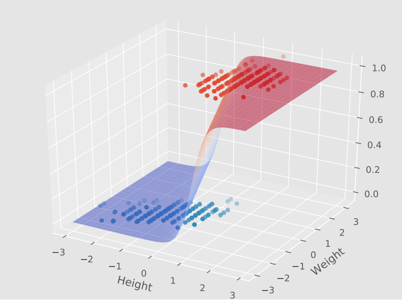


CSE 151A

Intro to Machine Learning

Lecture 09 – Part 01

About the Midterm

The Midterm

- First midterm is Friday.
- Covers everything from Weeks 01 – 04.
  - excluding logistic regression.
- Focus is on the **essentials**.

Format

- Canvas quiz.
  - Random order/subset, you can change answers.
- Multiple choice, T/F, short answer.
  - Result of simple calculation, explanation, etc.
- Open book, open notes, open Google, etc.
  - No proctoring software/webcam needed.
- However, **no collaboration**.

Logistics

- Exam will be posted on Canvas at 00:00 AM PST.
- Exam will disappear at 22:30 PM PST.
- You can start whenever, you'll have 1.5 hours.
- Open book, open notes, open Google, etc.
  - Exam designed to take ≈ 50 minutes.

Corrections and Clarifications

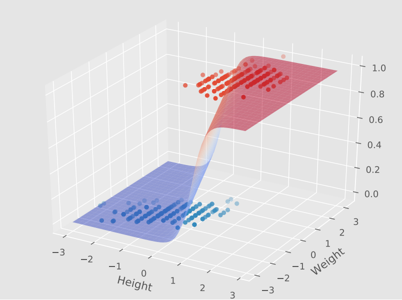
- This makes clarifications/corrections difficult to do fairly.
- **Unfortunately, no corrections/clarifications can be made.**
- Of course, if a question contains an error, it will be thrown out after the fact.

Studying

- No practice exam.
- Focus is on **essentials**.
- Here's a sample question:

A straight line is fit to a data set $\{(x_i, y_i)\}$ using least squares regression; the slope is found to be m. The data is changed by adding 10 to each y to create a new data set $\{(x_i, y_i + 10)\}$, and least squares is used again. The new slope is m'. Which is true?

a) $m = m'$ b) $m < m'$ c) $m > m'$



CSE 151A

Intro to Machine Learning

Lecture 09 – Part 02

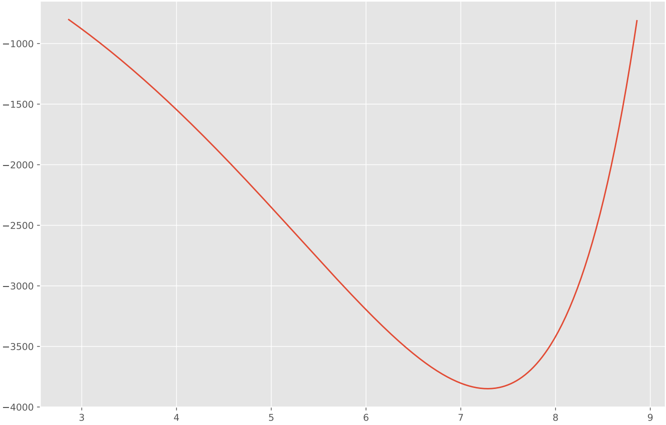
Motivating Gradient Descent

Last time...

- Set up logistic regression as optimization problem.
- Claimed: we can't solve it explicitly.
- Today: solve it using **gradient descent**.

But first...

- Minimize $f(x) = e^x - 100x^2$



Minimizing via Calculus

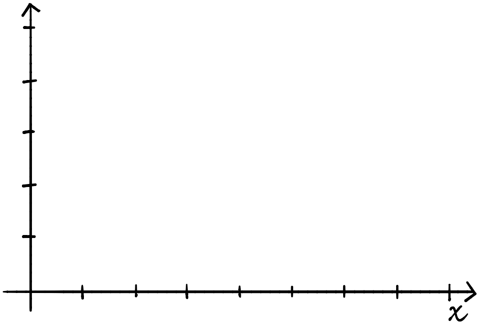
- Try setting derivative to zero, solving:

Minimizing via Calculus

- f **is differentiable.**
- But there is **no explicit solution** for $f'(x) = 0$.
- Can we use the derivative in some other way?

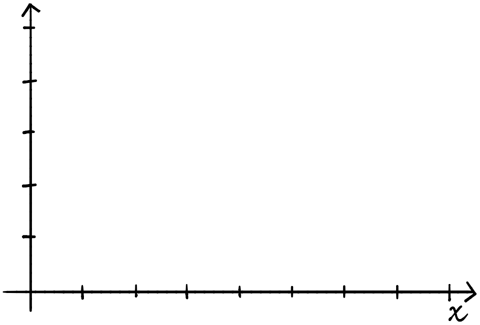
Meaning of the Derivative

- Meaning of **differentiable**: locally, f looks linear.
- $f'(x)$ is a function; it gives the **slope** at x.



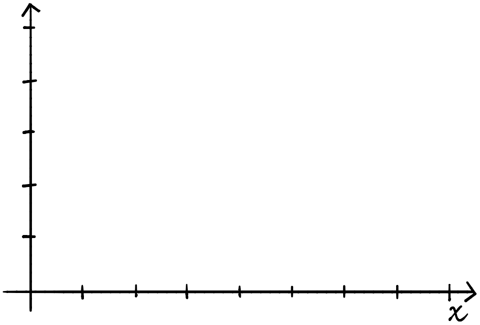
Key Idea Behind Gradient Descent

- Derivative at x tells us which way to go.
  - If the slope of f at x is **positive** then moving to the **left** decreases the value of R.



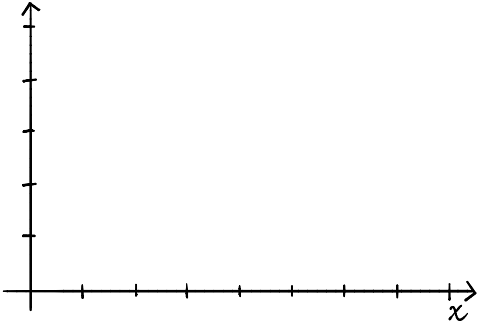
Key Idea Behind Gradient Descent

- Derivative at x tells us which way to go.
  - If the slope of f at x is **negative** then moving to the **right** decreases the value of R.



Key Idea Behind Gradient Descent

- Derivative at x tells us which way to go.
    - If the slope of f at x is **zero** then we are at a local optimum.



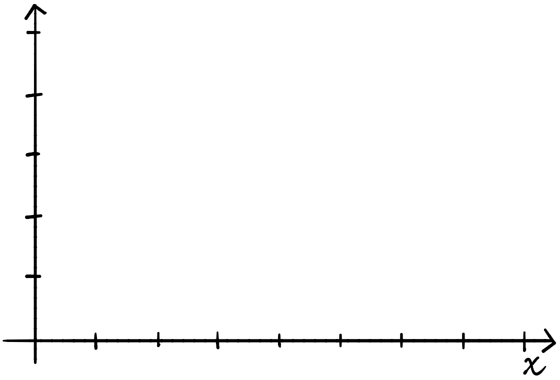
Taking a Step

- Suppose we are at x_0. Where do we go next?
- Slope at x_0 negative? Then **increase** x_0.
  - Step **right**.
- Slope at x_0 positive? Then **decrease** x_0.
  - Step **left**.
- This will work:

$$x_1 = x_0 - f'(x_0)$$

Gradient Descent

- Pick α to be a positive number.
  - It is the **learning rate**.
- Pick a starting guess, x_0.
- On step i, perform update $x_i = x_{i-1} - \alpha \cdot f'(x_{i-1})$
- Repeat until convergence
  - when x doesn't change much
  - equivalently, when $f'(x_i)$ is small


x

```python
def gradient_descent(derivative, x, alpha, tol=1e-12):
    """Minimize using gradient descent."""
    while True:
        x_next = x - alpha * derivative(x)
        if abs(x_next - x) < tol:
            break
        x = x_next
    return x
```

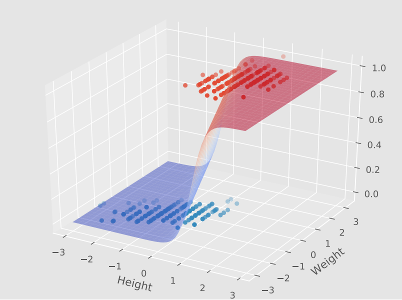
(demo)

Gradient Ascent

- Pick α to be a positive number.
  - It is the **learning rate**.
- Pick a starting guess, x_0.
- On step i, perform update $x_i = x_{i-1} + \alpha \cdot f'(x_{i-1})$
- Repeat until convergence
  - when h doesn't change much
  - equivalently, when $f'(x_i)$ is small

Ascent vs. Descent

- Maximizing f is equivalent to minimizing $-f$.



CSE 151A

Intro to Machine Learning

Lecture 09 – Part 02

Logistic Regression

Recall: Logistic Regression

- Predict probability that person has heart disease.
- Prediction rule:

$$H_{\vec{w}}(\vec{x}) = \sigma(\vec{w} \cdot \text{Aug}(\vec{x}))$$

where

$$\sigma(t) = \frac{1}{1 + e^{-t}}$$

is the **logistic function**.

Recall: Logistic Regression

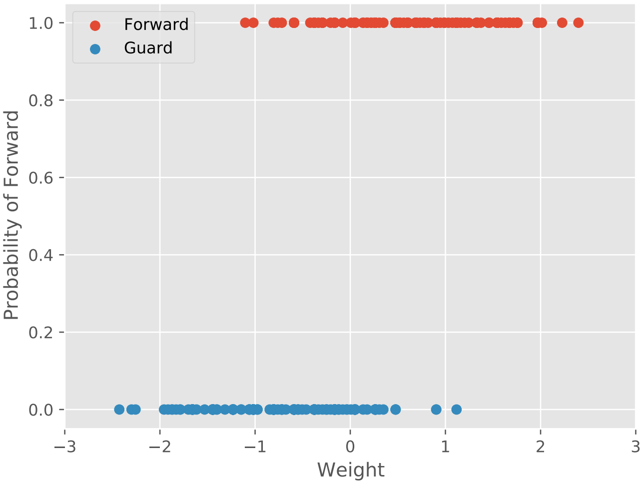
- Find **most likely** $\vec{w}$ using data.
- **Goal**: maximize the **log likelihood**,

$$\log \mathcal{L}(\vec{w}) = -\sum_{i=1}^{n} \log\left[1 + e^{-y_i \vec{w} \cdot \text{Aug}(\vec{x}^{(i)})}\right]$$

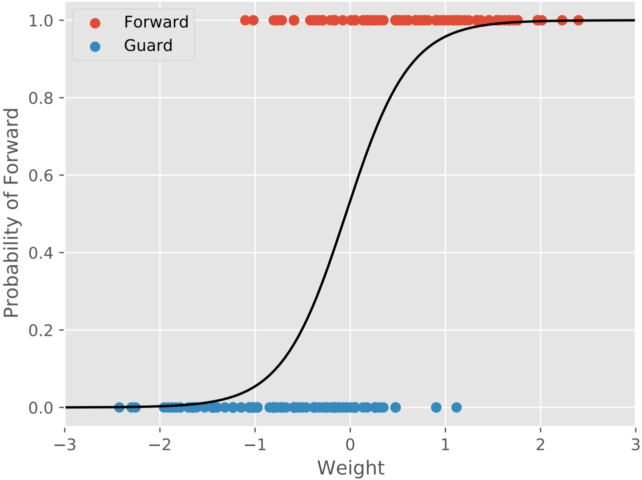
Another Example

- Given the weight of an NBA player.
- Predict probability that they are a forward.

Guards vs. Forwards



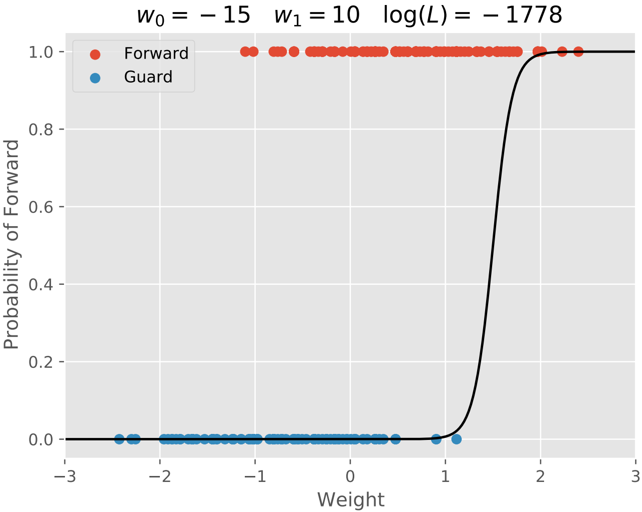
Guards vs. Forwards



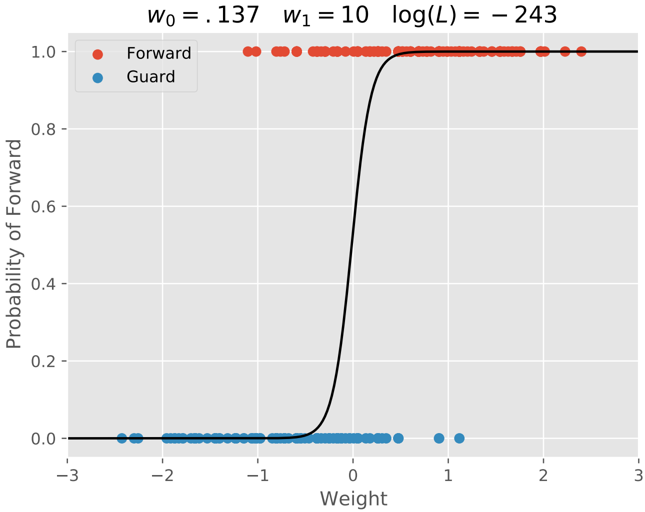
Prediction Rule Parameters

$$
\begin{aligned}
H_{\vec{w}}(\vec{x}) &= \sigma(\vec{w} \cdot \text{Aug}(\vec{x})) \\
&= \sigma(w_0 + w_1 \times \text{Weight})
\end{aligned}
$$

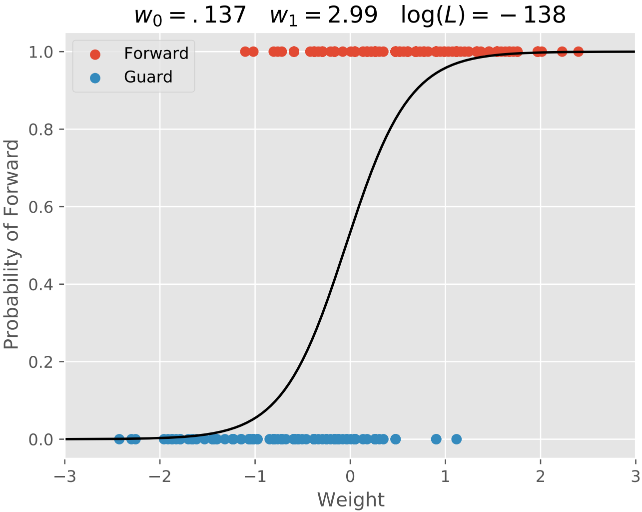
Prediction Rule Parameters



Prediction Rule Parameters



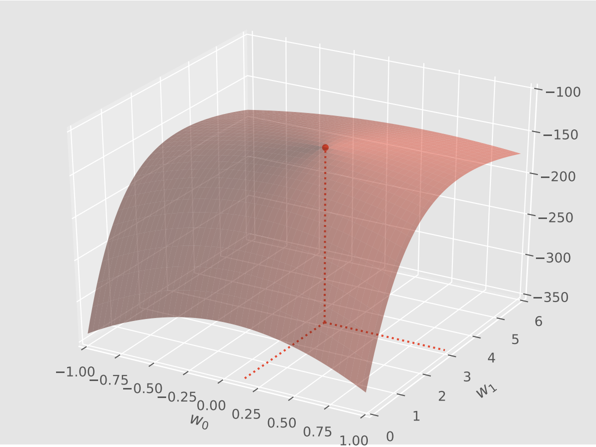
Prediction Rule Parameters



Learning

- Goal: find $\vec{w}$ maximizing $f(\vec{w}) = \log L(\vec{w})$.

The Log Likelihood



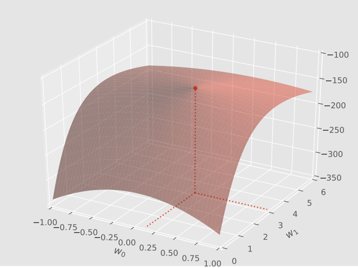
Maximizing

- Try setting gradient to zero, solving:
  - $f(\vec{w}) = -\sum_{i=1}^{n} \log\left(1 + \exp\left(-y_i \vec{w} \cdot \vec{x}^{(i)}\right)\right)$

Meaning of the Gradient

- Meaning of **differentiable**: locally, f looks linear.

- $\nabla f(\vec{w})$ is a function; it returns a vector pointing in direction of steepest ascent.



Gradient Ascent

- Pick α to be a positive number.
  - It is the **learning rate**.
- Pick a starting guess, $\vec{w}^{(0)}$.
- On step i, update $\vec{w}^{(i)} = \vec{w}^{(i-1)} + \alpha \cdot \nabla f(\vec{w}^{(i-1)})$
- Repeat until convergence
  - when $\vec{w}$ doesn't change much
  - equivalently, when $\|\nabla f(\vec{w}^{(i)})\|$ is small

Gradient Ascent for Logistic Regression

- Recall:

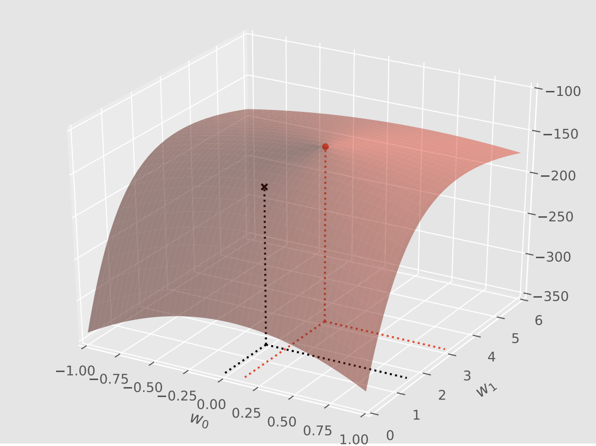
$$\nabla f(\vec{w}) = \sum_{k=1}^{n} y_k \vec{x}^{(k)} \frac{e^{-y_k \vec{w} \cdot \vec{x}^{(k)}}}{1 + e^{-y_k \vec{w} \cdot \vec{x}^{(k)}}} = \sum_{k=1}^{n} y_k \vec{x}^{(k)} \frac{1}{1 + e^{y_k \vec{w} \cdot \vec{x}^{(k)}}}$$

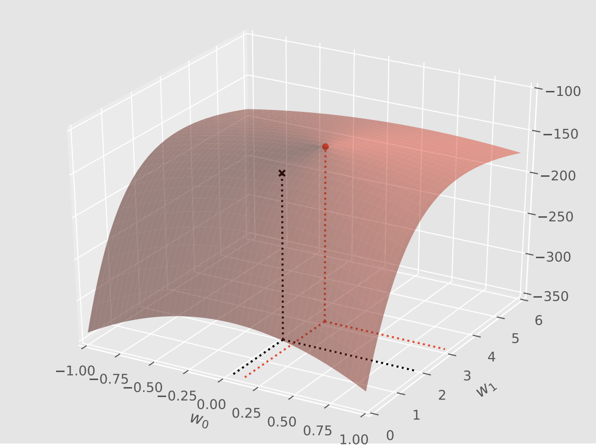
- Can show: $\nabla f(\vec{w}) = \sum_{k=1}^{n} y_k \vec{x}^{(k)} H_{\vec{w}}(-y_k \vec{x}^{(k)})$

Gradient Ascent for Logistic Regression

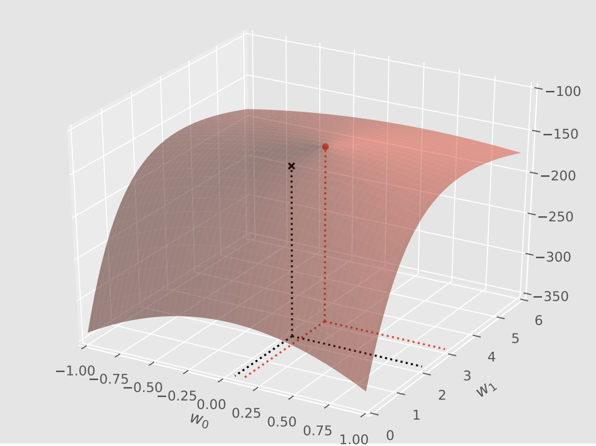
- On step i, update

$$\vec{w}^{(i)} = \vec{w}^{(i-1)} + \alpha \cdot \sum_{k=1}^{n} y_k \vec{x}^{(k)} H_{\vec{w}^{(i-1)}}(-y_k \vec{x}^{(k)})$$

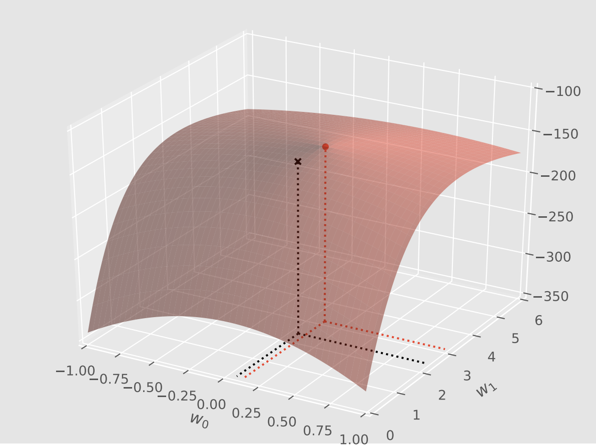

−100
−150
−200
−250
−300
−350
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00
w_0
0
1
2
3
4
5
6
w_1

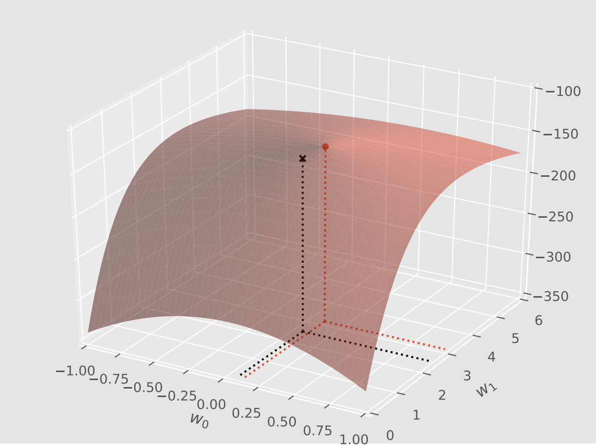


−100
−150
−200
−250
−300
−350
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00
w_0
0
1
2
3
4
5
6
w_1

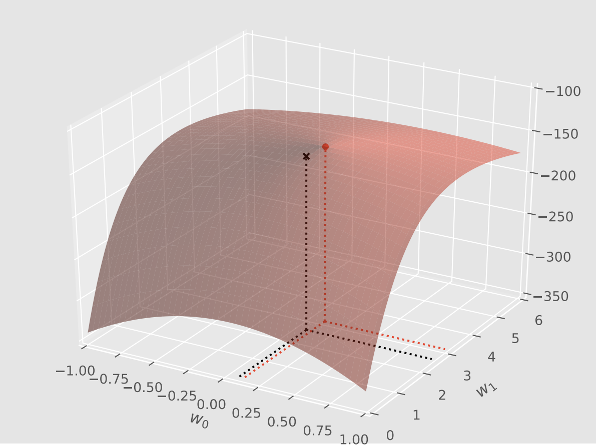


−100
−150
−200
−250
−300
−350
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00
w_0
0
1
2
3
4
5
6
w_1

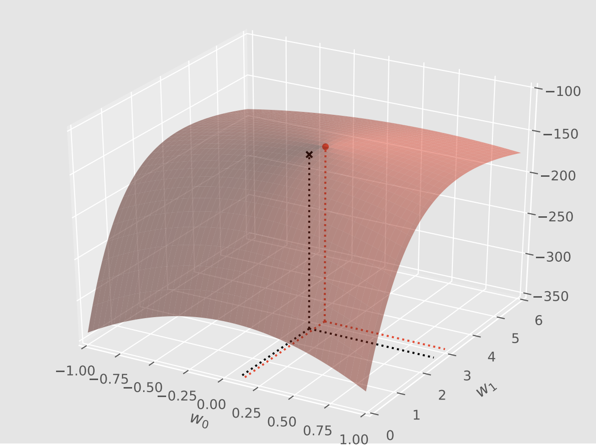

−100
−150
−200
−250
−300
−350
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00
w_0
0
1
2
3
4
5
6
w_1



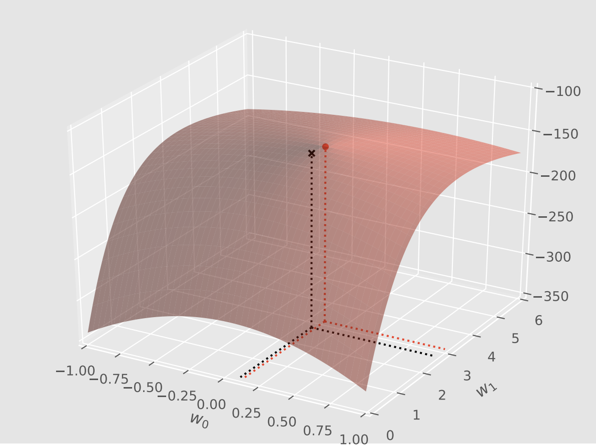
−100
−150
−200
−250
−300
−350
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00
w_0
0
1
2
3
4
5
6
w_1



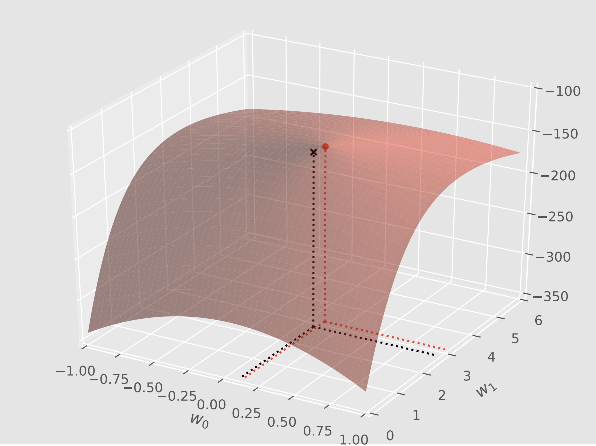
−100
−150
−200
−250
−300
−350
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00
w_0
0
1
2
3
4
5
6
w_1



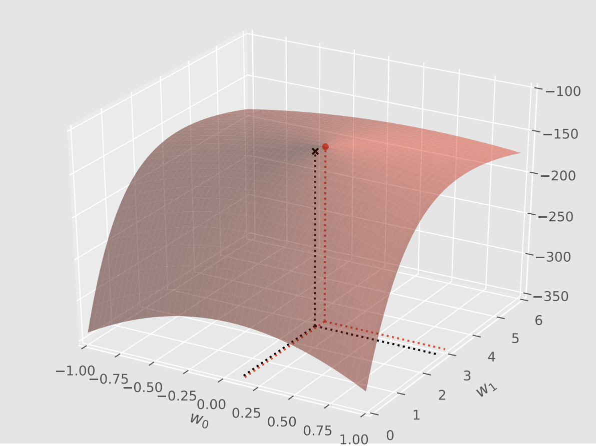
−100
−150
−200
−250
−300
−350
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00
w_0
0
1
2
3
4
5
6
w_1



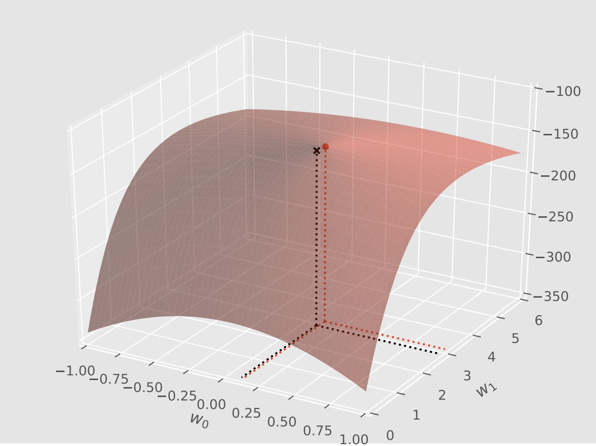
−100
−150
−200
−250
−300
−350
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00
w_0
0
1
2
3
4
5
6
w_1



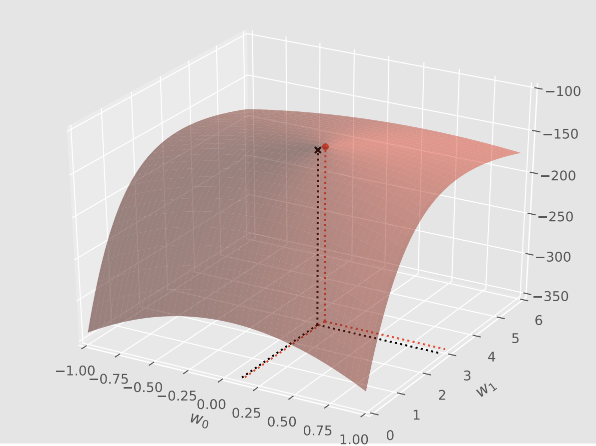
−100
−150
−200
−250
−300
−350
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00
w_0
0
1
2
3
4
5
6
w_1



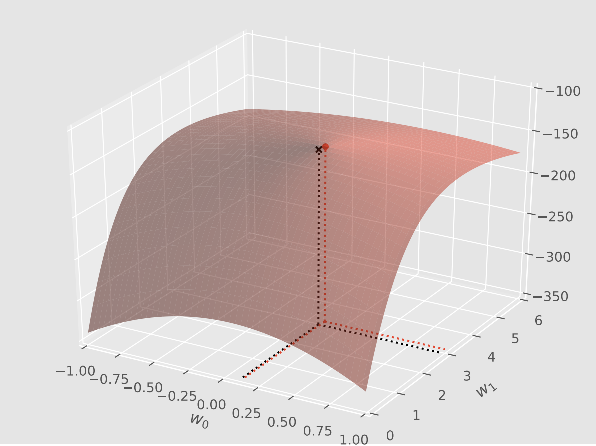
−100
−150
−200
−250
−300
−350
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00
w_0
0
1
2
3
4
5
6
w_1

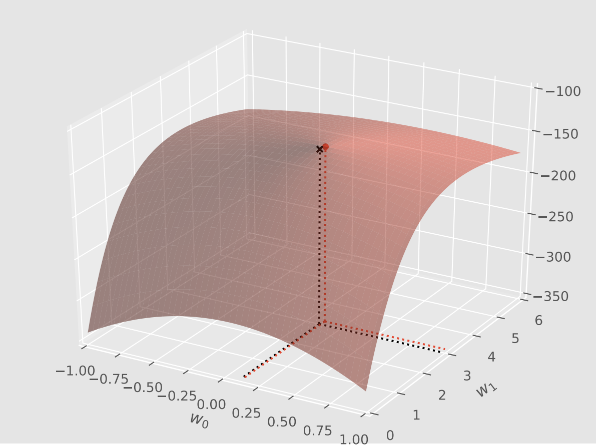


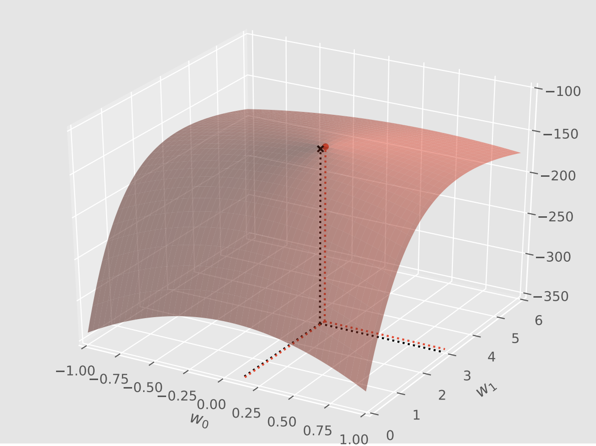
−100
−150
−200
−250
−300
−350
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00
w_0
0
1
2
3
4
5
6
w_1



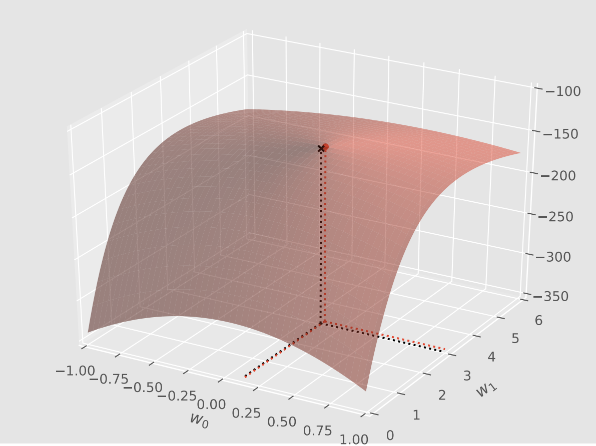
−100
−150
−200
−250
−300
−350
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00
w_0
0
1
2
3
4
5
6
w_1


−100
−150
−200
−250
−300
−350
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00
w_0
0
1
2
3
4
5
6
w_1

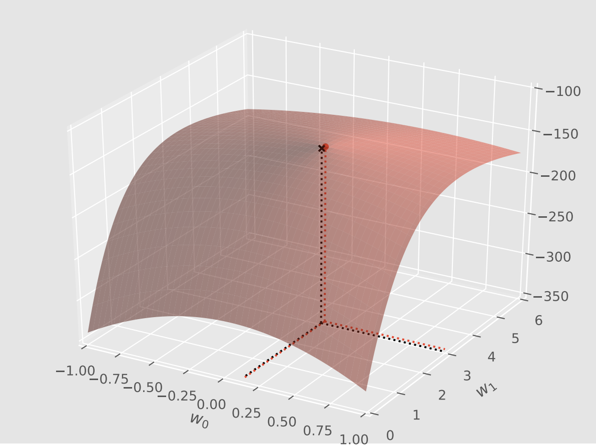

−100
−150
−200
−250
−300
−350
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00
w_0
0
1
2
3
4
5
6
w_1



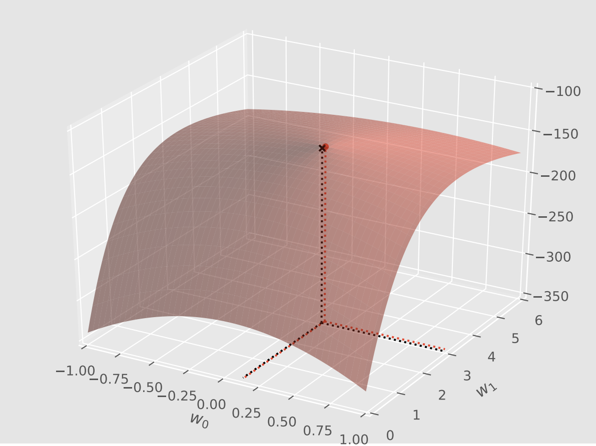
−100
−150
−200
−250
−300
−350
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00
w_0
0
1
2
3
4
5
6
w_1



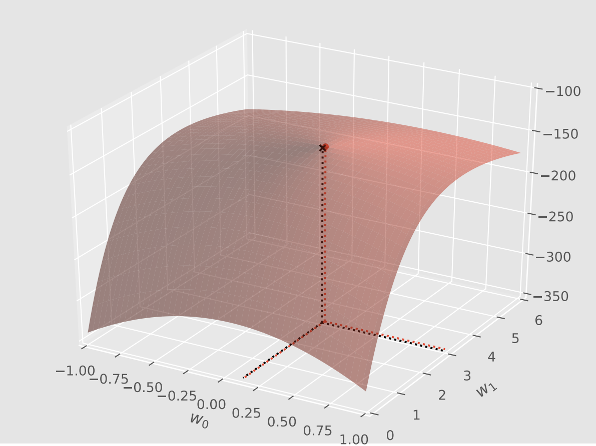
−100
−150
−200
−250
−300
−350
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00
w_0
0
1
2
3
4
5
6
w_1

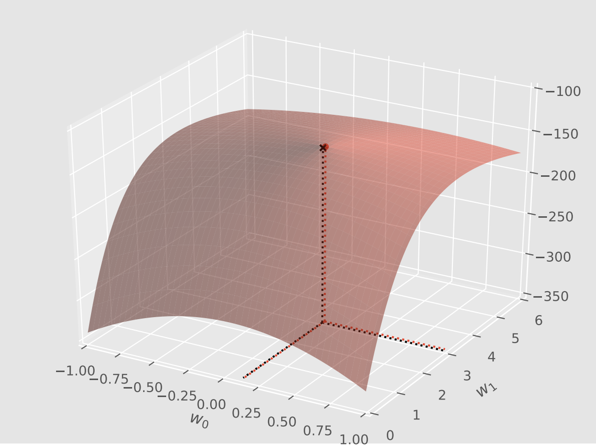


−100
−150
−200
−250
−300
−350
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00
w_0
0
1
2
3
4
5
6
w_1



−100
−150
−200
−250
−300
−350
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00
w_0
0
1
2
3
4
5
6
w_1


−100
−150
−200
−250
−300
−350
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00
w_0
0
1
2
3
4
5
6
w_1



−100
−150
−200
−250
−300
−350
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00
w_0
0
1
2
3
4
5
6
w_1

Gradient Descent

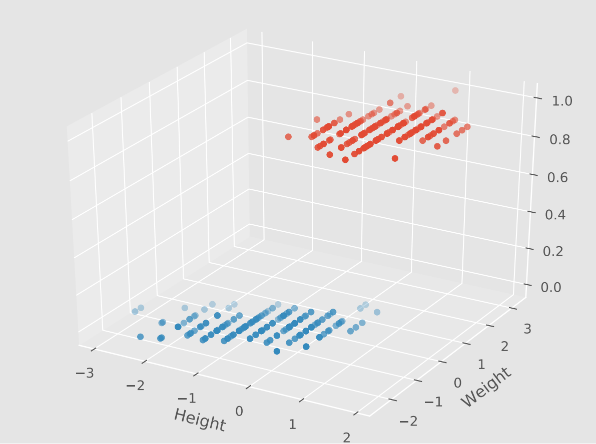
- Pick α to be a positive number.
  - It is the **learning rate**.
- Pick a starting guess, $\vec{w}^{(0)}$.
- On step i, update $\vec{w}^{(i)} = \vec{w}^{(i-1)} - \alpha \cdot \nabla f(\vec{w}^{(i-1)})$
- Repeat until convergence
  - when $\vec{w}$ doesn't change much
  - equivalently, when $\|\nabla f(\vec{w}^{(i)})\|$ is small

```python
def gradient_descent(gradient, w, alpha, tol=1e-12):
    """Minimize using gradient descent."""
    while True:
        w_next = w - alpha * gradient(x)
        if np.linalg.norm(w_next - w) < tol:
            break
        w = w_next
    return w
```

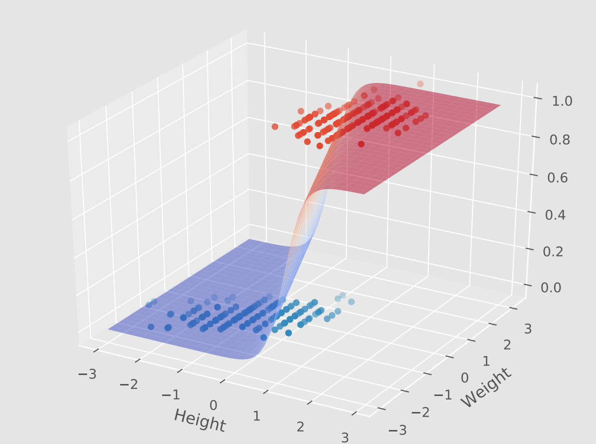
Adding Another Feature

- Use weight *and* height to predict position.
- Now $\text{Aug}(\vec{x}) \in \mathbb{R}^3$ and $\vec{w} \in \mathbb{R}^3$.

The Data



After Gradient Ascent



Making Classifications

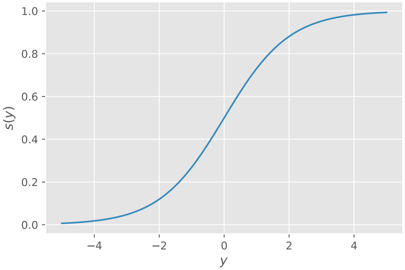
- Logistic regression predicts a probability:

$$H_{\vec{w}}(\vec{x}) = \sigma(\vec{w} \cdot \vec{x})$$

- Can turn into classification in two ways.

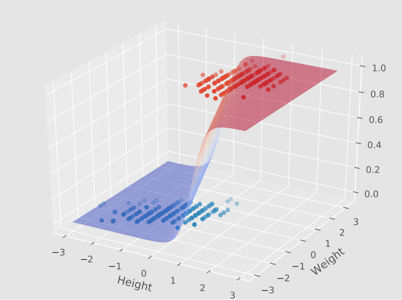
Approach 1

- If $H_{\vec{w}}(\vec{x}) > 0.5$, predict class 1; else predict class -1.
- Equivalently, predict class 1 if $\vec{x} \cdot \vec{w} > 0$.



Approach 2

- More generally, predict class 1 if $H_{\vec{w}}(\vec{x}) > \tau$
- Equivalently, predict class 1 if $\vec{x} \cdot \vec{w} > t$
- How to pick τ / t? Cross-validation!



CSE 151A

Intro to Machine Learning

Lecture 09 – Part 03

Demo: Heart Disease Dataset