Lecture: starts Tuesdays 9:35am (Boston time zone)
Course website: introml.odl.mit.edu
Who’s talking? Prof. Tamara Broderick
Questions? discourse.odl.mit.edu ("Lecture 4" category)
Materials: Will all be available at course website

Last Time(s)
I. Linear classifiers
II. Perceptron algorithm
III. A more-complete ML analysis

Today’s Plan
I. Linear logistic classification/logistic regression
II. Gradient descent
Recall
Recall

- Perceptron struggles with data that’s not linearly separable
Recall

- Perceptron struggles with data that’s not linearly separable
Recall

• Perceptron struggles with data that’s not linearly separable

Notice

• Perceptron doesn’t have a notion of uncertainty (how well do we know what we know?)
Recall

• Perceptron struggles with data that’s not linearly separable

Notice

• Perceptron doesn’t have a notion of uncertainty (how well do we know what we know?)
Recall

- Perceptron struggles with data that’s not linearly separable

Notice

- Perceptron doesn’t have a notion of uncertainty (how well do we know what we know?)

![Graph showing data points for different species of penguins with axes for flipper length (mm) and body mass (g).]
Recall

• Perceptron struggles with data that’s not linearly separable

Notice

• Perceptron doesn’t have a notion of uncertainty (how well do we know what we know?)

Am I wearing a coat?

Wind speed (kph)

Temperature (C)
Recall

• Perceptron struggles with data that’s not linearly separable

Notice

• Perceptron doesn’t have a notion of uncertainty (how well do we know what we know?)
Recall

- Perceptron struggles with data that’s not linearly separable

Notice

- Perceptron doesn’t have a notion of uncertainty (how well do we know what we know?)
Recall

- Perceptron struggles with data that’s not linearly separable

Notice

- Perceptron doesn’t have a notion of uncertainty (how well do we know what we know?)
Recall

- Perceptron struggles with data that’s not linearly separable

Notice

- Perceptron doesn’t have a notion of uncertainty (how well do we know what we know?)

Am I wearing a coat?
Recall

- Perceptron struggles with data that’s not linearly separable

Notice

- Perceptron doesn’t have a notion of uncertainty (how well do we know what we know?)
Capturing uncertainty
Capturing uncertainty

Probability of wearing a coat

Temperature (C)
Capturing uncertainty

Temperature (C)

Probability of wearing a coat

0.5

0

1
Capturing uncertainty

Temperature (C)

Probability of wearing a coat

0 0.5 1
Capturing uncertainty

Probability of wearing a coat

Temperature (C)

Label: am I wearing a coat?

Temperature (C)
Capturing uncertainty

Probability of wearing a coat

Temperature (C)

0 0.5 1

Label: am I wearing a coat?

Temperature (C)
Capturing uncertainty

Temperature (C)

Probability of wearing a coat

Label: am I wearing a coat?
Capturing uncertainty

Temperature (C)

Probability of wearing a coat

Label: am I wearing a coat?
Capturing uncertainty

Probability of wearing a coat

Label: am I wearing a coat?

Temperature (°C)

Temperature (°C)
Capturing uncertainty

Temperature (C)

Probability of wearing a coat

Label: am I wearing a coat?
Capturing uncertainty

Temperature (C)

Probability of wearing a coat

Label: am I wearing a coat?
Capturing uncertainty

Temperature (C)

Probability of wearing a coat

Label: am I wearing a coat?

Temperature (C)
Capturing uncertainty

Temperature (C)

Probability of wearing a coat

Label: am I wearing a coat?
Capturing uncertainty

Temperature (°C)

Probability of wearing a coat
0
0.5
1

Label: am I wearing a coat?

Temperature (°C)
Capturing uncertainty

Temperature (C)

Probability of wearing a coat

+ + + + + + + +

Label: am I wearing a coat?

Temperature (C)
Capturing uncertainty

Temperature (°C)

Probability of wearing a coat

Label: am I wearing a coat?

Temperature (°C)
Capturing uncertainty

Label: am I wearing a coat?

Temperature (C)

Probability of wearing a coat

Temperature (C)
Capturing uncertainty

Probability of wearing a coat

Temperature (C)

Label: am I wearing a coat?

Temperature (C)
Capturing uncertainty

Probability of wearing a coat

Temperature (°C)

Label: am I wearing a coat?

Temperature (°C)
Capturing uncertainty

Temperature (C)

Probability of wearing a coat

Label: am I wearing a coat?
Capturing uncertainty

Temperature (C)

Probability of wearing a coat

Label: am I wearing a coat?
Capturing uncertainty

Probability of wearing a coat

Temperature (C)

Label: am I wearing a coat?

Temperature (C)
Capturing uncertainty

- Temperature (C)
- Probability of wearing a coat

Label: am I wearing a coat?

Temperature (C)
Capturing uncertainty

Probability of wearing a coat

Temperature (C)

Label: am I wearing a coat?

Temperature (C)
Capturing uncertainty

Temperature (C)

Probability of wearing a coat

0

0.5

1

Label: am I wearing a coat?

+++ ++

Temperature (C)

--- --- --- ---
Capturing uncertainty

Temperature (C)

Probability of wearing a coat

Label: am I wearing a coat?
Capturing uncertainty

Label: am I wearing a coat?

Temperature (C)

Probability of wearing a coat

Temperature (C)

[Graphs showing probability of wearing a coat based on temperature]
Capturing uncertainty

Probability of wearing a coat

Temperature (C)

Label: am I wearing a coat?

Temperature (C)
Capturing uncertainty

Temperature (C)

Probability of wearing a coat

Label: am I wearing a coat?
Capturing uncertainty

- How to make this shape?
Capturing uncertainty

- How to make this shape?
Capturing uncertainty

- How to make this shape?
Capturing uncertainty

- How to make this shape?
  - Sigmoid/logistic function
Capturing uncertainty

- How to make this shape?
  - Sigmoid/logistic function

\[ \sigma(z) = \frac{1}{1 + \exp(-z)} \]
Capturing uncertainty

• How to make this shape?
• Sigmoid/logistic function

\[ \sigma(z) = \frac{1}{1 + \exp(-z)} \]
Capturing uncertainty

- How to make this shape?
- Sigmoid/logistic function

\[ \sigma(z) = \frac{1}{1 + \exp(-z)} \]
Capturing uncertainty

- How to make this shape?
  - Sigmoid/logistic function

\[ \sigma(z) = \frac{1}{1 + \exp(-z)} \]
Capturing uncertainty

Temperature (C)

Probability of wearing a coat

0

0.5

1

Label: am I wearing a coat?

Temperature (C)

How to make this shape?

Sigmoid/logistic function

\[ \sigma(z) = \frac{1}{1 + \exp(-z)} \]
Capturing uncertainty

- How to make this shape?
  - Sigmoid/logistic function

\[ \sigma(z) = \frac{1}{1 + \exp(-z)} \]
Capturing uncertainty

- How to make this shape?
  - Sigmoid/logistic function

$$\sigma(z) = \frac{1}{1 + \exp(-z)}$$
Capturing uncertainty

- How to make this shape?
  - Sigmoid/logistic function

\[
\sigma(z) = \frac{1}{1 + \exp(-z)}
\]
Capturing uncertainty

• How to make this shape?
• Sigmoid/logistic function

\[ \sigma(z) = \frac{1}{1 + \exp(-z)} \]
Capturing uncertainty

- How to make this shape?
- Sigmoid/logistic function

\[ \sigma(z) = \frac{1}{1 + \exp(-z)} \]
Capturing uncertainty

- How to make this shape?
- Sigmoid/logistic function

$$\sigma(z) = \frac{1}{1 + \exp(-z)}$$
Capturing uncertainty

• How to make this shape?
  • Sigmoid/logistic function

\[ \sigma(z) = \frac{1}{1 + \exp(-z)} \]
Capturing uncertainty

• How to make this shape?
  • Sigmoid/logistic function

\[ \sigma(z) = \frac{1}{1 + \exp(-z)} \]
Capturing uncertainty

• How to make this shape?
• Sigmoid/logistic function

\[ \sigma(z) = \frac{1}{1 + \exp(-z)} \]
Capturing uncertainty

- How to make this shape?
- Sigmoid/logistic function

\[
\sigma(z) = \frac{1}{1 + \exp(-z)}
\]
Capturing uncertainty

• How to make this shape?
  • Sigmoid/logistic function

\[ \sigma(z) = \frac{1}{1 + \exp(-z)} \]
Capturing uncertainty

• How to make this shape?
• Sigmoid/logistic function

\[ \sigma(z) = \frac{1}{1 + \exp(-z)} \]
Capturing uncertainty

- How to make this shape?
- Sigmoid/logistic function

$$\sigma(z) = \frac{1}{1 + \exp(-z)}$$
Capturing uncertainty

\[ g(x) = \sigma(\theta x + \theta_0) \]

- How to make this shape?
- Sigmoid/logistic function

\[ \sigma(z) = \frac{1}{1 + \exp(-z)} \]
Capturing uncertainty

\[ g(x) = \sigma(\theta x + \theta_0) = \frac{1}{1 + \exp\left\{-(\theta x + \theta_0)\right\}} \]

• How to make this shape?
  • Sigmoid/logistic function
    \[ \sigma(z) = \frac{1}{1 + \exp(-z)} \]
Capturing uncertainty
Capturing uncertainty

1 feature:
Capturing uncertainty

1 feature:

\[ g(x) = \sigma(\theta x + \theta_0) \]

\[ = \frac{1}{1 + \exp \left\{ - (\theta x + \theta_0) \right\}} \]
Capturing uncertainty

1 feature:

\[ g(x) = \sigma(\theta x + \theta_0) = \frac{1}{1 + \exp\{- (\theta x + \theta_0)\}} \]
Capturing uncertainty

1 feature:

\[ g(x) = \sigma(\theta x + \theta_0) \]

\[ = \frac{1}{1 + \exp \left\{ -(\theta x + \theta_0) \right\}} \]
Capturing uncertainty

1 feature:

\[ g(x) = \sigma(\theta x + \theta_0) \]

\[ = \frac{1}{1 + \exp\left\{ -(\theta x + \theta_0) \right\}} \]
Capturing uncertainty

1 feature:

\[ g(x) = \sigma(\theta x + \theta_0) \]

\[ = \frac{1}{1 + \exp\{-(\theta x + \theta_0)\}} \]

2 features:

\[ g(x) = \sigma(\theta^\top x + \theta_0) \]

\[ = \frac{1}{1 + \exp\{-(\theta^\top x + \theta_0)\}} \]
Capturing uncertainty

1 feature:

\[ g(x) = \sigma(\theta x + \theta_0) \]

\[ = \frac{1}{1 + \exp \{-(\theta x + \theta_0)\}} \]

2 features:

\[ g(x) = \sigma(\theta^\top x + \theta_0) \]

\[ = \frac{1}{1 + \exp \{-(\theta^\top x + \theta_0)\}} \]
Capturing uncertainty

1 feature:

\[ g(x) = \sigma(\theta x + \theta_0) = \frac{1}{1 + \exp\left\{- (\theta x + \theta_0)\right\}} \]

2 features:

\[ g(x) = \sigma(\theta^\top x + \theta_0) = \frac{1}{1 + \exp\left\{-(\theta^\top x + \theta_0)\right\}} \]
Capturing uncertainty

1 feature:

\[ g(x) = \sigma(\theta x + \theta_0) \]

\[ = \frac{1}{1 + \exp\left\{-(\theta x + \theta_0)\right\}} \]

2 features:

\[ g(x) = \sigma(\theta^\top x + \theta_0) \]

\[ = \frac{1}{1 + \exp\left\{-(\theta^\top x + \theta_0)\right\}} \]
Capturing uncertainty

1 feature:

\[ g(x) = \sigma(\theta x + \theta_0) \]

\[ = \frac{1}{1 + \exp \{-(\theta x + \theta_0)\}} \]

2 features:

\[ g(x) = \sigma(\theta^\top x + \theta_0) \]

\[ = \frac{1}{1 + \exp \{-(\theta^\top x + \theta_0)\}} \]
Capturing uncertainty

1 feature:

\[ g(x) = \sigma(\theta x + \theta_0) = \frac{1}{1 + \exp \left\{ -(\theta x + \theta_0) \right\}} \]

2 features:

\[ g(x) = \sigma(\theta^\top x + \theta_0) = \frac{1}{1 + \exp \left\{ -(\theta^\top x + \theta_0) \right\}} \]
Capturing uncertainty

1 feature:

\( g(x) = \sigma(\theta x + \theta_0) \)

\[ = \frac{1}{1 + \exp\{-(\theta x + \theta_0)\}} \]

2 features:

\( g(x) = \sigma(\theta^\top x + \theta_0) \)

\[ = \frac{1}{1 + \exp\{-(\theta^\top x + \theta_0)\}} \]
Capturing uncertainty

1 feature:
\[ g(x) = \sigma(\theta x + \theta_0) \]
\[ = \frac{1}{1 + \exp\left\{ - (\theta x + \theta_0) \right\}} \]

2 features:
\[ g(x) = \sigma(\theta^\top x + \theta_0) \]
\[ = \frac{1}{1 + \exp\left\{ - (\theta^\top x + \theta_0) \right\}} \]
Capturing uncertainty

1 feature:

\[ g(x) = \sigma(\theta x + \theta_0) \]

\[ = \frac{1}{1 + \exp\left\{-(\theta x + \theta_0)\right\}} \]

2 features:

\[ g(x) = \sigma(\theta^\top x + \theta_0) \]

\[ = \frac{1}{1 + \exp\left\{-(\theta^\top x + \theta_0)\right\}} \]
Capturing uncertainty

1 feature:

\[ g(x) = \sigma(\theta x + \theta_0) \]

\[ = \frac{1}{1 + \exp \left\{ - (\theta x + \theta_0) \right\}} \]

2 features:

\[ g(x) = \sigma(\theta^T x + \theta_0) \]

\[ = \frac{1}{1 + \exp \left\{ - (\theta^T x + \theta_0) \right\}} \]
Capturing uncertainty

1 feature:

\[ g(x) = \sigma(\theta x + \theta_0) \]

\[
= \frac{1}{1 + \exp\{-(\theta x + \theta_0)\}}
\]

2 features:

\[ g(x) = \sigma(\theta^\top x + \theta_0) \]

\[
= \frac{1}{1 + \exp\{-(\theta^\top x + \theta_0)\}}
\]
Capturing uncertainty

1 feature:

\[ g(x) = \sigma(\theta x + \theta_0) \]

\[ = \frac{1}{1 + \exp\left\{-(\theta x + \theta_0)\right\}} \]

2 features:

\[ g(x) = \sigma(\theta^\top x + \theta_0) \]

\[ = \frac{1}{1 + \exp\left\{-(\theta^\top x + \theta_0)\right\}} \]
Capturing uncertainty

1 feature:

\[ g(x) = \sigma(\theta x + \theta_0) \]

\[ = \frac{1}{1 + \exp\{- (\theta x + \theta_0)\}} \]

2 features:

\[ g(x) = \sigma(\theta^\top x + \theta_0) \]

\[ = \frac{1}{1 + \exp\{- (\theta^\top x + \theta_0)\}} \]
Capturing uncertainty

1 feature:

\[ g(x) = \sigma(\theta x + \theta_0) \]

\[ = \frac{1}{1 + \exp \{-(\theta x + \theta_0)\}} \]

2 features:

\[ g(x) = \sigma(\theta^\top x + \theta_0) \]

\[ = \frac{1}{1 + \exp \{- (\theta^\top x + \theta_0)\}} \]
Capturing uncertainty

1 feature:

\[ g(x) = \sigma(\theta x + \theta_0) = \frac{1}{1 + \exp\{- (\theta x + \theta_0)\}} \]

2 features:

\[ g(x) = \sigma(\theta^T x + \theta_0) = \frac{1}{1 + \exp\{- (\theta^T x + \theta_0)\}} \]
Capturing uncertainty

1 feature:

\[ g(x) = \sigma(\theta x + \theta_0) \]

\[ = \frac{1}{1 + \exp \{-(\theta x + \theta_0)\}} \]

\[
g(x) = \sigma(\theta^\top x + \theta_0) \frac{1}{1 + \exp \{-(\theta^\top x + \theta_0)\}}
\]
Capturing uncertainty

1 feature:

\[ g(x) = \sigma(\theta x + \theta_0) \]

\[ = \frac{1}{1 + \exp\{-(\theta x + \theta_0)\}} \]

2 features:

\[ g(x) = \sigma(\theta^\top x + \theta_0) \]

\[ = \frac{1}{1 + \exp\{-(\theta^\top x + \theta_0)\}} \]

\[ g(x) \]

\[ x \]

\[ x_1 \]

\[ x_2 \]

\[ \text{Temperature (C)} \]

\[ \text{Wind speed (kph)} \]
Capturing uncertainty

1 feature:

\[ g(x) = \sigma(\theta x + \theta_0) \]

\[ = \frac{1}{1 + \exp \left\{ -(\theta x + \theta_0) \right\}} \]

2 features:

\[ g(x) = \sigma(\theta^\top x + \theta_0) \]

\[ = \frac{1}{1 + \exp \left\{ -((\theta^\top x + \theta_0)) \right\}} \]
Linear logistic classification

aka logistic regression
Linear logistic classification

• How do we learn a classifier (i.e. learn $\theta, \theta_0$)?
Linear logistic classification

- How do we learn a classifier (i.e. learn $\theta, \theta_0$)?
Linear logistic classification

• How do we learn a classifier (i.e. learn $\theta$, $\theta_0$)?

aka logistic regression
Linear logistic classification

- How do we learn a classifier (i.e. learn $\theta, \theta_0$)?
- How do we make predictions?
Linear logistic classification

• How do we learn a classifier (i.e. learn $\theta, \theta_0$)?
• How do we make predictions?

aka logistic regression
Linear logistic classification

- How do we learn a classifier (i.e. learn $\theta, \theta_0$)?
- How do we make predictions?

aka logistic regression
Linear logistic classification

• How do we learn a classifier (i.e. learn $\theta$, $\theta_0$)?
• How do we make predictions?
  • Idea: predict +1 if:

aka logistic regression
Linear logistic classification

• How do we learn a classifier (i.e. learn $\theta, \theta_0$)?
• How do we make predictions?
  • Idea: predict +1 if: probability $> 0.5$
Linear logistic classification

• How do we learn a classifier (i.e. learn $\theta, \theta_0$)?
• How do we make predictions?

• Idea: predict +1 if: probability $> 0.5$

$$
\sigma(\theta^T x + \theta_0) > 0.5
$$
Linear logistic classification

• How do we learn a classifier (i.e. learn $\theta$, $\theta_0$)?
• How do we make predictions?

- Idea: predict $+1$ if: probability $> 0.5$
  \[
  \sigma(\theta^T x + \theta_0) > 0.5
  \]
  \[
  \frac{1}{1 + \exp \left\{ - (\theta^T x + \theta_0) \right\}} > 0.5
  \]
Linear logistic classification

• How do we learn a classifier (i.e. learn $\theta, \theta_0$)?
• How do we make predictions?

• Idea: predict +1 if: probability $> 0.5$

$$\sigma(\theta^\top x + \theta_0) > 0.5$$

$$\frac{1}{1 + \exp \left\{ -(\theta^\top x + \theta_0) \right\}} > 0.5$$

$$\exp \left\{ -(\theta^\top x + \theta_0) \right\} < 1$$
Linear logistic classification

• How do we learn a classifier (i.e. learn \( \theta, \theta_0 \))? 
• How do we make predictions?

• Idea: predict +1 if: probability > 0.5
  \[
  \frac{1}{1 + \exp\{-(\theta^\top x + \theta_0)\}} > 0.5 \\
  \exp\{-(\theta^\top x + \theta_0)\} < 1 \\
  \theta^\top x + \theta_0 > 0
  \]
Linear logistic classification

- How do we learn a classifier (i.e. learn $\theta, \theta_0$)?
- How do we make predictions?

- Idea: predict +1 if: probability $> 0.5$

$$\sigma(\theta^T x + \theta_0) > 0.5$$

$$\frac{1}{1 + \exp \left\{ -(\theta^T x + \theta_0) \right\}} > 0.5$$

$$\exp \left\{ -(\theta^T x + \theta_0) \right\} < 1$$

$$\theta^T x + \theta_0 > 0$$

- Same hypothesis class as before!
Linear logistic classification

• How do we learn a classifier (i.e. learn $\theta$, $\theta_0$)?
• How do we make predictions?

- Idea: predict +1 if: probability > 0.5

$$\sigma(\theta^T x + \theta_0) > 0.5$$

$$\frac{1}{1 + \exp \{- (\theta^T x + \theta_0)\}} > 0.5$$

$$\exp \{- (\theta^T x + \theta_0)\} < 1$$

$$\theta^T x + \theta_0 > 0$$

- Same hypothesis class as before! But we will get:
Linear logistic classification

• How do we learn a classifier (i.e. learn $\theta, \theta_0$)?
• How do we make predictions?

• Idea: predict +1 if: probability $> 0.5$
  \[ \frac{1}{1 + \exp \left\{ -(\theta^T x + \theta_0) \right\}} > 0.5 \]
  \[ \exp \left\{ -(\theta^T x + \theta_0) \right\} < 1 \]
  \[ \theta^T x + \theta_0 > 0 \]

• Same hypothesis class as before! But we will get:
  • Uncertainties
Linear logistic classification

• How do we learn a classifier (i.e. learn $\theta, \theta_0$)?
• How do we make predictions?

- Idea: predict +1 if: probability $> 0.5$
  $$\sigma(\theta^T x + \theta_0) > 0.5$$
  $$\frac{1}{1 + \exp \left\{ - (\theta^T x + \theta_0) \right\}} > 0.5$$
  $$\exp \left\{ - (\theta^T x + \theta_0) \right\} < 1$$
  $$\theta^T x + \theta_0 > 0$$

- Same hypothesis class as before! But we will get:
  • Uncertainties
  • Quality guarantees when data not linearly separable
Linear logistic classification

• How do we learn a classifier (i.e. learn $\theta, \theta_0$)?
Linear logistic classification

• How do we learn a classifier (i.e. learn $\theta, \theta_0$)?

 aka logistic regression
Linear logistic classification

• How do we learn a classifier (i.e. learn $\theta, \theta_0$)?
Linear logistic classification

• How do we learn a classifier (i.e. learn $\theta, \theta_0$)?

aka logistic regression
Linear logistic classification

• How do we learn a classifier (i.e. learn \( \theta, \theta_0 \))?

aka logistic regression
Linear logistic classification

• How do we learn a classifier (i.e. learn $\theta, \theta_0$)?
Linear logistic classification

• How do we learn a classifier (i.e. learn $\theta, \theta_0$)?

aka logistic regression
Linear logistic classification

• How do we learn a classifier (i.e. learn $\theta$, $\theta_0$)?

Probability(data)

aka logistic regression
Linear logistic classification

- How do we learn a classifier (i.e. learn $\theta, \theta_0$)?

$\text{Probability(data)} = \prod_{i=1}^{n} \text{Probability(data point } i\text{)}$
Linear logistic classification

• How do we learn a classifier (i.e. learn $\theta, \theta_0$)?

$$\text{Probability(data)} = \prod_{i=1}^{n} \text{Probability(data point } i)$$

aka logistic regression
Linear logistic classification

- How do we learn a classifier (i.e. learn \( \theta, \theta_0 \))?

\[
\text{Probability(data)} = \prod_{i=1}^{n} \text{Probability(data point } i) \quad [\text{Let } g^{(i)} = \sigma(\theta^\top x^{(i)} + \theta_0)]
\]
Linear logistic classification

• How do we learn a classifier (i.e. learn \( \theta, \theta_0 \))?

Probability(data)

\[
= \prod_{i=1}^{n} \text{Probability(data point } i) \\
= \prod_{i=1}^{n} \left[ \text{Let } g^{(i)} = \sigma(\theta^T x^{(i)} + \theta_0) \right] \\
= \prod_{i=1}^{n} \left\{ \begin{array}{ll}
g^{(i)} & \text{if } y^{(i)} = +1 \\
(1 - g^{(i)}) & \text{else}
\end{array} \right.
\]

aka logistic regression
Linear logistic classification

- How do we learn a classifier (i.e. learn $\theta$, $\theta_0$)?

Probability(data)

\[
\text{Probability(data)} = \prod_{i=1}^{n} \text{Probability(data point } i) \\
= \prod_{i=1}^{n} \left[ \text{Let } g^{(i)} = \sigma(\theta^T x^{(i)} + \theta_0) \right] \\
= \prod_{i=1}^{n} \left\{ \begin{array}{ll}
g^{(i)} & \text{if } y^{(i)} = +1 \\
(1 - g^{(i)}) & \text{else} \end{array} \right.
\]

\[
= \prod_{i=1}^{n} (g^{(i)})^{1\{y^{(i)}=+1\}} (1 - g^{(i)})^{1\{y^{(i)} \neq +1\}}
\]
Linear logistic classification

• How do we learn a classifier (i.e. learn $\theta, \theta_0$)?

Probability(data)

$$= \prod_{i=1}^{n} \text{Probability(data point } i)$$

$$= \prod_{i=1}^{n} \left[ \text{Let } g^{(i)} = \sigma(\theta^T x^{(i)} + \theta_0) \right]$$

$$= \prod_{i=1}^{n} \begin{cases} g^{(i)} & \text{if } y^{(i)} = +1 \\ (1 - g^{(i)}) & \text{else} \end{cases}$$

$$= \prod_{i=1}^{n} (g^{(i)})^{1\{y^{(i)}=+1\}} (1 - g^{(i)})^{1\{y^{(i)}\neq+1\}}$$
Linear logistic classification

- How do we learn a classifier (i.e. learn $\theta$, $\theta_0$)?

Probability(data)

$$= \prod_{i=1}^{n} \text{Probability(data point } i)$$

$$= \prod_{i=1}^{n} \left\{ \begin{array}{cl} g^{(i)} & \text{if } y^{(i)} = +1 \\ (1 - g^{(i)}) & \text{else} \end{array} \right. = \prod_{i=1}^{n} (g^{(i)})^{1\{y^{(i)}=+1\}} (1 - g^{(i)})^{1\{y^{(i)}\neq+1\}}$$

aka logistic regression
Linear logistic classification

• How do we learn a classifier (i.e. learn $\theta, \theta_0$)?

Probability(data)

$$\text{Probability(data)} = \prod_{i=1}^{n} \text{Probability(data point } i)$$

$$= \prod_{i=1}^{n} \left[ \text{Let } g^{(i)} = \sigma(\theta^\top x^{(i)} + \theta_0) \right]$$

$$= \prod_{i=1}^{n} \left\{ \begin{array}{ll}
g^{(i)} & \text{if } y^{(i)} = +1 \\
(1 - g^{(i)}) & \text{else}
\end{array} \right.$$  

$$= \prod_{i=1}^{n} (g^{(i)})^{1\{y^{(i)}=+1\}} (1 - g^{(i)})^{1\{y^{(i)}\neq+1\}}$$
Linear logistic classification

• How do we learn a classifier (i.e. learn $\theta$, $\theta_0$)?

Probability(data)

$$\text{Probability(data)} = \prod_{i=1}^{n} \text{Probability(data point } i)$$

[Let $g^{(i)} = \sigma(\theta^\top x^{(i)} + \theta_0)$]

$$= \prod_{i=1}^{n} \begin{cases} g^{(i)} & \text{if } y^{(i)} = +1 \\ (1 - g^{(i)}) & \text{else} \end{cases}$$

$$= \prod_{i=1}^{n} (g^{(i)})^{1\{y^{(i)}=+1\}} (1 - g^{(i)})^{1\{y^{(i)}\neq+1\}}$$

aka logistic regression
Linear logistic classification

- How do we learn a classifier (i.e. learn $\theta, \theta_0$)?

Probability(data)

$$\text{Probability(data)} = \prod_{i=1}^{n} \text{Probability(data point } i)$$

$$= \prod_{i=1}^{n} [\text{Let } g^{(i)} = \sigma(\theta^T x^{(i)} + \theta_0)]$$

$$= \prod_{i=1}^{n} \begin{cases} g^{(i)} & \text{if } y^{(i)} = +1 \\ (1 - g^{(i)}) & \text{else} \end{cases}$$

$$= \prod_{i=1}^{n} \left( g^{(i)} \right)^{1\{y^{(i)}=+1\}} \left( 1 - g^{(i)} \right)^{1\{y^{(i)}\neq+1\}}$$

log probability(data)

aka logistic regression
Linear logistic classification

• How do we learn a classifier (i.e. learn $\theta, \theta_0$)?

Probability(data)

$$= \prod_{i=1}^{n} \text{Probability(data point } i)$$

$$= \prod_{i=1}^{n} \left[ \text{Let } g^{(i)} = \sigma (\theta^T x^{(i)} + \theta_0) \right]$$

$$= \prod_{i=1}^{n} \left\{ \begin{array}{ll}
g^{(i)} & \text{if } y^{(i)} = +1 \\
(1 - g^{(i)}) & \text{else}
\end{array} \right\}$$

$$= \prod_{i=1}^{n} \left( g^{(i)} \right)^{1\{y^{(i)} = +1\}} \left( 1 - g^{(i)} \right)^{1\{y^{(i)} \neq +1\}}$$

Loss(data) = $-\log \text{ probability(data)}$
Linear logistic classification

• How do we learn a classifier (i.e. learn $\theta, \theta_0$)?

Probability(data)

\[
\prod_{i=1}^{n} \text{Probability(data point } i) \\
= \prod_{i=1}^{n} \left\{ \begin{array}{ll}
g(i) & \text{if } y(i) = +1 \\ (1 - g(i)) & \text{else} \end{array} \right.
\]

\[
= \prod_{i=1}^{n} (g(i))^{1\{y(i)=+1\}} (1 - g(i))^{1\{y(i)\neq+1\}}
\]

Loss(data) = -log probability(data)

\[
= \sum_{i=1}^{n} - \left( 1\{y(i) = +1\} \log g(i) + 1\{y(i) \neq +1\} \log(1 - g(i)) \right)
\]
Linear logistic classification

• How do we learn a classifier (i.e. learn \( \theta, \theta_0 \))?

**Probability(data)**

\[
\text{Probability(data)} = \prod_{i=1}^{n} \text{Probability(data point } i) = \prod_{i=1}^{n} \left\{ \begin{array}{ll}
g(i) & \text{if } y(i) = +1 \\
(1 - g(i)) & \text{else}
\end{array} \right.
\]

\[
= \prod_{i=1}^{n} (g(i))^{1\{y(i)=+1\}} (1 - g(i))^{1\{y(i)\neq+1\}}
\]

**Loss(data)**

\[
\text{Loss(data)} = -\log \text{probability(data)} = \sum_{i=1}^{n} - \left(1\{y(i) = +1\} \log g(i) + 1\{y(i) \neq +1\} \log(1 - g(i)) \right)
\]
Linear logistic classification

- How do we learn a classifier (i.e. learn $\theta$, $\theta_0$)?

Probability(data)

$\prod_{i=1}^{n} \text{Probability(data point } i) = \prod_{i=1}^{n} \left\{ \begin{array}{ll} g(i) & \text{if } y(i) = +1 \\ (1 - g(i)) & \text{else} \end{array} \right.$

$= \prod_{i=1}^{n} (g(i))^{1 \{y(i) = +1\}} (1 - g(i))^{1 \{y(i) \neq +1\}}$

Loss(data) = $-\log \text{ probability(data)}$

$= \sum_{i=1}^{n} - \left(1 \{y(i) = +1\} \log g(i) + 1 \{y(i) \neq +1\} \log (1 - g(i)) \right)$
Linear logistic classification

• How do we learn a classifier (i.e. learn $\theta, \theta_0$)?

Probability(data)

\[
\frac{1}{n} \prod_{i=1}^{n} \text{Probability(data point } i) \quad \text{[Let } g^{(i)} = \sigma(\theta^\top x^{(i)} + \theta_0) \text{]} \\
= \prod_{i=1}^{n} \begin{cases} 
    g^{(i)} & \text{if } y^{(i)} = +1 \\
    (1 - g^{(i)}) & \text{else}
\end{cases} \\
= \prod_{i=1}^{n} (g^{(i)})^{1\{y^{(i)}=+1\}} (1 - g^{(i)})^{1\{y^{(i)}\neq+1\}}
\]

Loss(data) = \(-\log \text{ probability(data)}\)

\[
= \sum_{i=1}^{n} - \left( 1\{y^{(i)} = +1\} \log g^{(i)} + 1\{y^{(i)} \neq +1\} \log(1 - g^{(i)}) \right)
\]
Linear logistic classification

- How do we learn a classifier (i.e. learn $\theta, \theta_0$)?

Probability(data)

$$\begin{align*}
\text{Probability(data)} & = \prod_{i=1}^{n} \text{Probability(data point } i) \\
& = \prod_{i=1}^{n} \left\{ \begin{array}{ll}
g^{(i)} & \text{if } y^{(i)} = +1 \\
(1 - g^{(i)}) & \text{else}
\end{array} \right.
\end{align*}$$

Loss(data) = -log probability(data)

$$\begin{align*}
\text{Loss(data)} = & \sum_{i=1}^{n} - \left( 1\{y^{(i)} = +1\} \log g^{(i)} + 1\{y^{(i)} \neq +1\} \log(1 - g^{(i)}) \right)
\end{align*}$$
Linear logistic classification

- How do we learn a classifier (i.e. learn $\theta$, $\theta_0$)?

Probability(data)

$$\prod_{i=1}^{n} \text{Probability(data point } i)$$

$$\prod_{i=1}^{n} [\text{Let } g^{(i)} = \sigma(\theta^T x^{(i)} + \theta_0)]$$

$$= \prod_{i=1}^{n} \left\{ \begin{array}{ll}
g^{(i)} & \text{if } y^{(i)} = +1 \\
1 - g^{(i)} & \text{else}
\end{array} \right.$$

$$= \prod_{i=1}^{n} (g^{(i)})^{1 \{y^{(i)} = +1\}} (1 - g^{(i)})^{1 \{y^{(i)} \neq +1\}}$$

Loss(data) = $-\log$ probability(data)

$$= \frac{1}{n} \sum_{i=1}^{n} - \left( 1 \{y^{(i)} = +1\} \log g^{(i)} + 1 \{y^{(i)} \neq +1\} \log(1 - g^{(i)}) \right)$$
Linear logistic classification

How do we learn a classifier (i.e. learn $\theta, \theta_0$)?

Probability(data)

$$\prod_{i=1}^{n} \text{Probability(data point } i)$$

[Let $g^{(i)} = \sigma(\theta^T x^{(i)} + \theta_0)$ ]

$$= \prod_{i=1}^{n} \begin{cases} 
 g^{(i)} \text{ if } y^{(i)} = +1 \\
 (1 - g^{(i)}) \text{ else }
\end{cases}$$

$$= \prod_{i=1}^{n} (g^{(i)})^{1\{y^{(i)}=+1\}} (1 - g^{(i)})^{1\{y^{(i)}\neq+1\}}$$

Loss(data) = -(1/n) * log probability(data)

$$= \frac{1}{n} \sum_{i=1}^{n} -\left(1\{y^{(i)} = +1\} \log g^{(i)} + 1\{y^{(i)} \neq +1\} \log(1 - g^{(i)})\right)$$
Linear logistic classification

• How do we learn a classifier (i.e. learn $\theta, \theta_0$)?

Probability(data)

$$\text{Probability(data)} = \prod_{i=1}^{n} \text{Probability(data point } i)$$

$$= \prod_{i=1}^{n} \left[ \text{Let } g^{(i)} = \sigma(\theta^T x^{(i)} + \theta_0) \right]$$

$$= \prod_{i=1}^{n} \left\{ \begin{array}{ll}
g^{(i)} & \text{if } y^{(i)} = +1 \\
(1 - g^{(i)}) & \text{else}
\end{array} \right.$$ 

$$= \prod_{i=1}^{n} (g^{(i)})^{1\{y^{(i)}=+1\}} (1 - g^{(i)})^{1\{y^{(i)}\neq+1\}}$$

Loss(data) = $-(1/n) \times \log \text{probability(data)}$

$$= \frac{1}{n} \sum_{i=1}^{n} - \left(1\{y^{(i)} = +1\} \log g^{(i)} + 1\{y^{(i)} \neq +1\} \log(1 - g^{(i)}) \right)$$
Linear logistic classification

- How do we learn a classifier (i.e. learn $\theta, \theta_0$)?

Probability(data)

$$\prod_{i=1}^{n} \text{Probability (data point } i)$$

$$= \prod_{i=1}^{n} [\text{Let } g^{(i)} = \sigma(\theta^\top x^{(i)} + \theta_0)]$$

$$= \prod_{i=1}^{n} \begin{cases} g^{(i)} & \text{if } y^{(i)} = +1 \\ (1 - g^{(i)}) & \text{else} \end{cases}$$

$$= \prod_{i=1}^{n} (g^{(i)})^{1\{y^{(i)}=+1\}} (1 - g^{(i)})^{1\{y^{(i)}\neq+1\}}$$

Loss(data) = $-(1/n) \times \log \text{probability(data)}$

$$= \frac{1}{n} \sum_{i=1}^{n} - \left( 1\{y^{(i)} = +1\} \log g^{(i)} + 1\{y^{(i)} \neq +1\} \log(1 - g^{(i)}) \right)$$

Negative log likelihood loss ($g$ for guess, $a$ for actual):
Linear logistic classification

- How do we learn a classifier (i.e. learn $\theta$, $\theta_0$)?

**Probability(data)**

\[
= \prod_{i=1}^{n} \text{Probability(data point } i) \\
= \prod_{i=1}^{n} \begin{cases} 
  g(i) & \text{if } y(i) = +1 \\
  (1 - g(i)) & \text{else}
\end{cases}
\]

\[
= \prod_{i=1}^{n} (g(i))^{1\{y(i)=+1\}} (1 - g(i))^{1\{y(i)\neq+1\}}
\]

**Loss(data)**

\[
= -(1/n) \times \log \text{probability(data)} \\
= \frac{1}{n} \sum_{i=1}^{n} - \left( 1\{y(i) = +1\} \log g(i) + 1\{y(i) \neq +1\} \log(1 - g(i)) \right)
\]

Negative log likelihood loss ($g$ for guess, $a$ for actual):

\[
-L_{nll}(g, a) = (1\{a = +1\} \log g + 1\{a \neq +1\} \log(1 - g))
\]
Linear logistic classification

- How do we learn a classifier (i.e. learn $\theta, \theta_0$)?
Linear logistic classification

• How do we learn a classifier (i.e. learn $\theta$, $\theta_0$)?
• Want to find parameter values to minimize average (negative log likelihood) loss across the data

aka logistic regression
Linear logistic classification

• How do we learn a classifier (i.e. learn $\theta, \theta_0$)?
• Want to find parameter values to minimize average (negative log likelihood) loss across the data

$$\frac{1}{n} \sum_{i=1}^{n} L_{nll}(\sigma(\theta^\top x^{(i)} + \theta_0), y^{(i)})$$
Linear logistic classification

- How do we learn a classifier (i.e. learn $\theta, \theta_0$)?
- Want to find parameter values to minimize average (negative log likelihood) loss across the data

$$J_{lr}(\theta, \theta_0) = \frac{1}{n} \sum_{i=1}^{n} L_{nll}(\sigma(\theta^\top x^{(i)} + \theta_0), y^{(i)})$$
Linear logistic classification

- How do we learn a classifier (i.e. learn $\theta, \theta_0$)?
- Want to find parameter values to minimize average (negative log likelihood) loss across the data

$$J_{lr}(\Theta) = J_{lr}(\theta, \theta_0) = \frac{1}{n} \sum_{i=1}^{n} L_{nll}(\sigma(\theta^\top x^{(i)} + \theta_0), y^{(i)})$$
Linear logistic classification

- How do we learn a classifier (i.e. learn $\theta$, $\theta_0$)?
- Want to find parameter values to minimize average (negative log likelihood) loss across the data

\[ J_{lr}(\Theta) = J_{lr}(\theta, \theta_0) = \frac{1}{n} \sum_{i=1}^{n} L_{nll}(\sigma(\theta^T x^{(i)} + \theta_0), y^{(i)}) \]

\[ L_{nll}(\sigma(\theta^T x^{(i)} + \theta_0), y^{(i)}) \]

Temperature (C)

0 0.5 1

Temperature (C)
Linear logistic classification

- How do we learn a classifier (i.e. learn $\theta, \theta_0$)?
- Want to find parameter values to minimize average (negative log likelihood) loss across the data

\[
J_{lr}(\Theta) = J_{lr}(\theta, \theta_0) = \frac{1}{n} \sum_{i=1}^{n} L_{nll}(\sigma(\theta^T x^{(i)} + \theta_0), y^{(i)})
\]
Linear logistic classification

• How do we learn a classifier (i.e. learn $\theta, \theta_0$)?
• Want to find parameter values to minimize average (negative log likelihood) loss across the data

$$J_{lr}(\Theta) = J_{lr}(\theta, \theta_0) = \frac{1}{n} \sum_{i=1}^{n} L_{nll}(\sigma(\theta^\top x^{(i)} + \theta_0), y^{(i)})$$
Linear logistic classification

• How do we learn a classifier (i.e. learn $\theta, \theta_0$)?
• Want to find parameter values to minimize average
  (negative log likelihood) loss across the data

\[
J_{lr}(\Theta) = J_{lr}(\theta, \theta_0) = \frac{1}{n} \sum_{i=1}^{n} L_{nll}(\sigma(\theta^\top x^{(i)} + \theta_0), y^{(i)})
\]
Linear logistic classification

• How do we learn a classifier (i.e. learn \( \theta, \theta_0 \))?  
• Want to find parameter values to minimize average (negative log likelihood) loss across the data

\[
J_{lr}(\Theta) = J_{lr}(\theta, \theta_0) = \frac{1}{n} \sum_{i=1}^{n} L_{nll}(\sigma(\theta^\top x^{(i)} + \theta_0), y^{(i)})
\]
Linear logistic classification

- How do we learn a classifier (i.e. learn $\theta, \theta_0$)?
- Want to find parameter values to minimize average (negative log likelihood) loss across the data

$$J_{lr}(\Theta) = J_{lr}(\theta, \theta_0) = \frac{1}{n} \sum_{i=1}^{n} L_{nll}(\sigma(\theta^\top x^{(i)} + \theta_0), y^{(i)})$$
Linear logistic classification

- How do we learn a classifier (i.e. learn $\theta, \theta_0$)?
- Want to find parameter values to minimize average (negative log likelihood) loss across the data

$$J_{lr}(\Theta) = J_{lr}(\theta, \theta_0) = \frac{1}{n} \sum_{i=1}^{n} L_{nll}(\sigma(\theta^\top x^{(i)} + \theta_0), y^{(i)})$$

Temperature (C)

0

0.5

1

aka logistic regression
Gradient descent
Gradient descent
Gradient descent
Gradient descent
Gradient descent

\[ f(\Theta) \]

\[ \Theta_1 \]

\[ \Theta_2 \]
Gradient descent
Gradient descent

- Gradient $\nabla_{\Theta} f = \left[ \frac{\partial f}{\partial \Theta_1}, \ldots, \frac{\partial f}{\partial \Theta_m} \right]^T$
- with $\Theta \in \mathbb{R}^m$
Gradient descent

- Gradient $\nabla_\Theta f = \left[ \frac{\partial f}{\partial \Theta_1}, \cdots, \frac{\partial f}{\partial \Theta_m} \right]^T$
- with $\Theta \in \mathbb{R}^m$
Gradient descent

- Gradient $\nabla_{\Theta} f = \left[ \frac{\partial f}{\partial \Theta_1}, \ldots, \frac{\partial f}{\partial \Theta_m} \right]^T$

- with $\Theta \in \mathbb{R}^m$
Gradient descent

- Gradient $\nabla_\Theta f = \left[ \frac{\partial f}{\partial \Theta_1}, \ldots, \frac{\partial f}{\partial \Theta_m} \right]^T$
- with $\Theta \in \mathbb{R}^m$
Gradient descent

- Gradient $\nabla_{\Theta} f = \left[ \frac{\partial f}{\partial \Theta_1}, \ldots, \frac{\partial f}{\partial \Theta_m} \right]^T$
- with $\Theta \in \mathbb{R}^m$

Gradient-Descent $(\Theta_{\text{init}}, \eta, f, \nabla_{\Theta} f, \epsilon)$
Gradient descent

- Gradient $\nabla_\Theta f = \left[ \frac{\partial f}{\partial \Theta_1}, \ldots, \frac{\partial f}{\partial \Theta_m} \right]^T$
- with $\Theta \in \mathbb{R}^m$

Gradient-Descent($\Theta_{\text{init}}, \eta, f, \nabla_\Theta f, \epsilon$)

Initialize $\Theta^{(0)} = \Theta_{\text{init}}$
Gradient descent

- Gradient $\nabla_\Theta f = \left[ \frac{\partial f}{\partial \Theta_1}, \ldots, \frac{\partial f}{\partial \Theta_m} \right]^T$
- with $\Theta \in \mathbb{R}^m$

Gradient-Descent ($\Theta_{\text{init}}, \eta, f, \nabla_\Theta f, \epsilon$)

Initialize $\Theta^{(0)} = \Theta_{\text{init}}$

Initialize $t = 0$
Gradient descent

- Gradient $\nabla_\Theta f = \left[ \frac{\partial f}{\partial \Theta_1}, \ldots, \frac{\partial f}{\partial \Theta_m} \right]^T$
- with $\Theta \in \mathbb{R}^m$

Gradient-Descent($\Theta_{init}, \eta, f, \nabla_\Theta f, \epsilon$)
Initialize $\Theta^{(0)} = \Theta_{init}$
Initialize $t = 0$

repeat
Gradient descent

- Gradient $\nabla \Theta f = \left[ \frac{\partial f}{\partial \Theta_1}, \ldots, \frac{\partial f}{\partial \Theta_m} \right]^T$
- with $\Theta \in \mathbb{R}^m$

**Gradient-Descent** ($\Theta_{\text{init}}, \eta, f, \nabla \Theta f, \epsilon$)

Initialize $\Theta^{(0)} = \Theta_{\text{init}}$

Initialize $t = 0$

repeat

$t = t + 1$
Gradient descent

- Gradient $\nabla_{\Theta} f = \left[ \frac{\partial f}{\partial \Theta_1}, \ldots, \frac{\partial f}{\partial \Theta_m} \right]^T$
- with $\Theta \in \mathbb{R}^m$

Gradient-Descent($\Theta_{\text{init}}, \eta, f, \nabla_{\Theta} f, \epsilon$)

Initialize $\Theta^{(0)} = \Theta_{\text{init}}$

Initialize $t = 0$

repeat

$t = t + 1$

$\Theta^{(t)} = \Theta^{(t-1)} - \eta \nabla_{\Theta} f(\Theta^{(t-1)})$
Gradient descent

- Gradient $\nabla_\Theta f = \left[ \frac{\partial f}{\partial \Theta_1}, \ldots, \frac{\partial f}{\partial \Theta_m} \right]^\top$
- with $\Theta \in \mathbb{R}^m$

Gradient-Descent ($\Theta_{\text{init}}, \eta, f, \nabla_\Theta f, \epsilon$)

Initialize $\Theta^{(0)} = \Theta_{\text{init}}$
Initialize $t = 0$

repeat

\[ t = t + 1 \]
\[ \Theta^{(t)} = \Theta^{(t-1)} - \eta \nabla_\Theta f(\Theta^{(t-1)}) \]
Gradient descent

- Gradient $\nabla_{\Theta} f = \begin{bmatrix} \frac{\partial f}{\partial \Theta_1} , \cdots , \frac{\partial f}{\partial \Theta_m} \end{bmatrix}^T$

- with $\Theta \in \mathbb{R}^m$

Gradient-Descent $(\Theta_{\text{init}}, \eta, f, \nabla_{\Theta} f, \epsilon)$

1. Initialize $\Theta^{(0)} = \Theta_{\text{init}}$
2. Initialize $t = 0$

repeat

   $t = t + 1$

   $\Theta^{(t)} = \Theta^{(t-1)} - \eta \nabla_{\Theta} f(\Theta^{(t-1)})$
Gradient descent

- Gradient $\nabla_{\Theta} f = \left[ \frac{\partial f}{\partial \Theta_1}, \ldots, \frac{\partial f}{\partial \Theta_m} \right]^T$
- with $\Theta \in \mathbb{R}^m$

Gradient-Descent($\Theta_{\text{init}}, \eta, f, \nabla_{\Theta} f, \epsilon$)
Initialize $\Theta^{(0)} = \Theta_{\text{init}}$
Initialize $t = 0$

repeat
  $t = t + 1$
  $\Theta^{(t)} = \Theta^{(t-1)} - \eta \nabla_{\Theta} f(\Theta^{(t-1)})$
Gradient descent

- Gradient $\nabla_{\Theta} f = \left[ \frac{\partial f}{\partial \Theta_1}, \ldots, \frac{\partial f}{\partial \Theta_m} \right]^T$

- with $\Theta \in \mathbb{R}^m$

Gradient-Descent ($\Theta_{\text{init}}, \eta, f, \nabla_{\Theta} f, \epsilon$)

Initialize $\Theta^{(0)} = \Theta_{\text{init}}$

Initialize $t = 0$

repeat

$t = t + 1$

$\Theta^{(t)} = \Theta^{(t-1)} - \eta \nabla_{\Theta} f(\Theta^{(t-1)})$

until
Gradient descent

- Gradient $\nabla_{\Theta} f = \left[ \frac{\partial f}{\partial \Theta_1}, \ldots, \frac{\partial f}{\partial \Theta_m} \right]^T$
- with $\Theta \in \mathbb{R}^m$

Gradient-Descent($\Theta_{\text{init}}, \eta, f, \nabla_{\Theta} f, \epsilon$)

Initialize $\Theta^{(0)} = \Theta_{\text{init}}$

Initialize $t = 0$

repeat
  $t = t + 1$
  $\Theta^{(t)} = \Theta^{(t-1)} - \eta \nabla_{\Theta} f(\Theta^{(t-1)})$

until $\left| f(\Theta^{(t)}) - f(\Theta^{(t-1)}) \right| < \epsilon$
Gradient descent

- Gradient $\nabla_\Theta f = \left[ \frac{\partial f}{\partial \Theta_1}, \ldots, \frac{\partial f}{\partial \Theta_m} \right]^T$
- with $\Theta \in \mathbb{R}^m$

Gradient-Descent ($\Theta_{\text{init}}, \eta, f, \nabla_\Theta f, \epsilon$)

Initialize $\Theta^{(0)} = \Theta_{\text{init}}$

Initialize $t = 0$

repeat
   $t = t + 1$
   $\Theta^{(t)} = \Theta^{(t-1)} - \eta \nabla_\Theta f(\Theta^{(t-1)})$
until $|f(\Theta^{(t)}) - f(\Theta^{(t-1)})| < \epsilon$

Return $\Theta^{(t)}$
Gradient descent

- Gradient $\nabla_{\Theta} f = \left[ \frac{\partial f}{\partial \Theta_1}, \ldots, \frac{\partial f}{\partial \Theta_m} \right]^T$
- with $\Theta \in \mathbb{R}^m$

Gradient-Descent $(\Theta_{\text{init}}, \eta, f, \nabla_{\Theta} f, \epsilon)$

Initialize $\Theta^{(0)} = \Theta_{\text{init}}$

Initialize $t = 0$

**repeat**

$t = t + 1$

$\Theta^{(t)} = \Theta^{(t-1)} - \eta \nabla_{\Theta} f(\Theta^{(t-1)})$

**until** $\left| f(\Theta^{(t)}) - f(\Theta^{(t-1)}) \right| < \epsilon$

**Return** $\Theta^{(t)}$

- Other possible stopping criteria:
Gradient descent

- Gradient $\nabla_\Theta f = \left[ \frac{\partial f}{\partial \Theta_1}, \ldots, \frac{\partial f}{\partial \Theta_m} \right]^T$

- with $\Theta \in \mathbb{R}^m$

Gradient-Descent ($\Theta_{\text{init}}, \eta, f, \nabla_\Theta f, \epsilon$)

Initialize $\Theta^{(0)} = \Theta_{\text{init}}$

Initialize $t = 0$

repeat

$t = t + 1$

$\Theta^{(t)} = \Theta^{(t-1)} - \eta \nabla_\Theta f(\Theta^{(t-1)})$

until $|f(\Theta^{(t)}) - f(\Theta^{(t-1)})| < \epsilon$

Return $\Theta^{(t)}$

- Other possible stopping criteria:
  - Max number of iterations $T$
Gradient descent

- Gradient $\nabla_\Theta f = \left[ \frac{\partial f}{\partial \Theta_1}, \ldots, \frac{\partial f}{\partial \Theta_m} \right]^T$
  - with $\Theta \in \mathbb{R}^m$

Gradient-Descent$(\Theta_{\text{init}}, \eta, f, \nabla_\Theta f, \epsilon)$

Initialize $\Theta^{(0)} = \Theta_{\text{init}}$

Initialize $t = 0$

repeat
  $t = t + 1$
  $\Theta^{(t)} = \Theta^{(t-1)} - \eta \nabla_\Theta f(\Theta^{(t-1)})$
until $\left| f(\Theta^{(t)}) - f(\Theta^{(t-1)}) \right| < \epsilon$

Return $\Theta^{(t)}$

- Other possible stopping criteria:
  - Max number of iterations $T$
  - $|\Theta^{(t)} - \Theta^{(t-1)}| < \epsilon$
Gradient descent

- Gradient \( \nabla_\Theta f = \left[ \frac{\partial f}{\partial \Theta_1}, \ldots, \frac{\partial f}{\partial \Theta_m} \right]^T \)
  - with \( \Theta \in \mathbb{R}^m \)

Gradient-Descent \((\Theta_{\text{init}}, \eta, f, \nabla_\Theta f, \epsilon)\)

Initialize \( \Theta^{(0)} = \Theta_{\text{init}} \)

Initialize \( t = 0 \)

repeat
  \( t = t + 1 \)
  \( \Theta^{(t)} = \Theta^{(t-1)} - \eta \nabla_\Theta f(\Theta^{(t-1)}) \)
until \( \left| f(\Theta^{(t)}) - f(\Theta^{(t-1)}) \right| < \epsilon \)

Return \( \Theta^{(t)} \)

- Other possible stopping criteria:
  - Max number of iterations \( T \)
  - \( |\Theta^{(t)} - \Theta^{(t-1)}| < \epsilon \)
  - \( \| \nabla_\Theta f(\Theta^{(t)}) \| < \epsilon \)
Gradient descent properties

• A function $f$ on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of $f$ lies above or on the graph
Gradient descent properties

- A function $f$ on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of $f$ lies above or on the graph.
Gradient descent properties

- A function $f$ on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of $f$ lies above or on the graph.
Gradient descent properties

- A function $f$ on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of $f$ lies above or on the graph.
Gradient descent properties

- A function $f$ on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of $f$ lies above or on the graph.
Gradient descent properties

- A function $f$ on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of $f$ lies above or on the graph.
Gradient descent properties

- A function $f$ on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of $f$ lies above or on the graph.
Gradient descent properties

- A function $f$ on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of $f$ lies above or on the graph.
Gradient descent properties

- A function $f$ on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of $f$ lies above or on the graph.
Gradient descent properties

• A function $f$ on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of $f$ lies above or on the graph.
Gradient descent properties

• A function $f$ on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of $f$ lies above or on the graph.
Gradient descent properties

- A function $f$ on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of $f$ lies above or on the graph.
Gradient descent properties

- A function $f$ on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of $f$ lies above or on the graph.
Gradient descent properties

• A function $f$ on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of $f$ lies above or on the graph.
Gradient descent properties

- A function $f$ on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of $f$ lies above or on the graph.
Gradient descent properties

- A function $f$ on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of $f$ lies above or on the graph.
Gradient descent properties

- A function $f$ on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of $f$ lies above or on the graph.
Gradient descent properties

• A function $f$ on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of $f$ lies above or on the graph.

• **Theorem**: Gradient descent performance
Gradient descent properties

• A function $f$ on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of $f$ lies above or on the graph

• **Theorem**: Gradient descent performance

• **Assumptions**: 
Gradient descent properties

- A function $f$ on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of $f$ lies above or on the graph.

- **Theorem**: Gradient descent performance

- **Assumptions**: (Choose any $\tilde{\epsilon} > 0$)
Gradient descent properties

• A function $f$ on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of $f$ lies above or on the graph

• **Theorem:** Gradient descent performance

• **Assumptions:** (Choose any $\tilde{\epsilon} > 0$)
  • $f$ is sufficiently “smooth” and convex
Gradient descent properties

• A function $f$ on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of $f$ lies above or on the graph

• **Theorem:** Gradient descent performance
• **Assumptions:** (Choose any $\tilde{\epsilon} > 0$)
  • $f$ is sufficiently “smooth” and convex
  • $f$ has at least one global optimum
Gradient descent properties

- A function $f$ on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of $f$ lies above or on the graph.

- **Theorem:** Gradient descent performance
  - **Assumptions:** (Choose any $\tilde{\epsilon} > 0$)
    - $f$ is sufficiently “smooth” and convex
    - $f$ has at least one global optimum
Gradient descent properties

• A function $f$ on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of $f$ lies above or on the graph.

• **Theorem**: Gradient descent performance
  
  • **Assumptions**: (Choose any $\tilde{\varepsilon} > 0$)
    
    • $f$ is sufficiently “smooth” and convex
    • $f$ has at least one global optimum
Gradient descent properties

• A function $f$ on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of $f$ lies above or on the graph.

• **Theorem**: Gradient descent performance

• **Assumptions**: (Choose any $\tilde{\epsilon} > 0$)
  • $f$ is sufficiently “smooth” and convex
  • $f$ has at least one global optimum
Gradient descent properties

• A function $f$ on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of $f$ lies above or on the graph

• **Theorem**: Gradient descent performance
  • **Assumptions**: (Choose any $\tilde{\epsilon} > 0$)
    • $f$ is sufficiently “smooth” and convex
    • $f$ has at least one global optimum
    • $\eta$ is sufficiently small
Gradient descent properties

• A function $f$ on $\mathbb{R}^m$ is convex if any line segment connecting two points of the graph of $f$ lies above or on the graph.

**Theorem**: Gradient descent performance

**Assumptions**: (Choose any $\tilde{\epsilon} > 0$)

• $f$ is sufficiently “smooth” and convex
• $f$ has at least one global optimum
• $\eta$ is sufficiently small

**Conclusion**: If run long enough, gradient descent will return a value within $\tilde{\epsilon}$ of a global optimum $\Theta$. 
Gradient descent for logistic regression
Gradient descent for logistic regression

• Loss $J_{lr}(\Theta) = J_{lr}(\theta, \theta_0)$ is differentiable
Gradient descent for logistic regression

- Loss $J_{lr}(\Theta) = J_{lr}(\theta, \theta_0)$ is differentiable & convex
Gradient descent for logistic regression

- Loss $J_{lr}(\Theta) = J_{lr}(\theta, \theta_0)$ is differentiable & convex
- Run Gradient-Descent($\Theta_{\text{init}}, \eta, J_{lr}, \nabla_\Theta J_{lr}, \epsilon$)
Gradient descent for logistic regression

- Loss $J_{lr}(\Theta) = J_{lr}(\theta, \theta_0)$ is differentiable & convex
- Run Gradient-Descent ( $\Theta_{\text{init}}, \eta, J_{lr}, \nabla \Theta J_{lr}, \epsilon$ )

Wear a coat?

+ + + + +

Temperature (°C)

- - - - -
Gradient descent for logistic regression

- Loss $J_{lr}(\Theta) = J_{lr}(\theta, \theta_0)$ is differentiable & convex
- Run Gradient-Descent($\Theta_{\text{init}}, \eta, J_{lr}, \nabla_{\Theta} J_{lr}, \epsilon$)

Wear a coat?

Temperature (C)

15
Gradient descent for logistic regression

- Loss $J_{lr}(\Theta) = J_{lr}(\theta, \theta_0)$ is differentiable & convex
- Run $\text{Gradient Descent}(\Theta_{\text{init}}, \eta, J_{lr}, \nabla \Theta J_{lr}, \epsilon)$
Gradient descent for logistic regression

- Loss $J_{lr}(\Theta) = J_{lr}(\theta, \theta_0)$ is differentiable & convex
- Run Gradient-Descent ( $\Theta_{\text{init}}, \eta, J_{lr}, \nabla_{\Theta} J_{lr}, \epsilon$ )

Wear a coat?

+ + + + + +

15 Temperature (C)

Wear a swimsuit?

+ + + + + + +

23 Temperature (C)
Gradient descent for logistic regression

- Loss $J_{lr}(\Theta) = J_{lr}(\theta, \theta_0)$ is differentiable & convex
- Run $\text{Gradient-Descent}(\Theta_{\text{init}}, \eta, J_{lr}, \nabla_{\Theta} J_{lr}, \epsilon)$
Gradient descent for logistic regression

• Loss $J_{lr}(\Theta) = J_{lr}(\theta, \theta_0)$ is differentiable & convex

• Run Gradient-Descent ($\Theta_{init}, \eta, J_{lr}, \nabla\Theta J_{lr}, \epsilon$)
Gradient descent for logistic regression

- Loss $J_{lr}(\Theta) = J_{lr}(\theta, \theta_0)$ is differentiable & convex
- Run Gradient-Descent ($\Theta_{\text{init}}, \eta, J_{lr}, \nabla_{\Theta} J_{lr}, \epsilon$)

Wear base layer?  

---

Temperature (C)
Gradient descent for logistic regression

- Loss $J_{lr}(\Theta) = J_{lr}(\theta, \theta_0)$ is differentiable & convex
- Run $\text{Gradient-Descent}(\Theta_{\text{init}}, \eta, J_{lr}, \nabla \Theta J_{lr}, \epsilon)$
Gradient descent for logistic regression

- Loss $J_{lr}(\Theta) = J_{lr}(\theta, \theta_0)$ is differentiable & convex
- Run Gradient-Descent ($\Theta_{init}, \eta, J_{lr}, \nabla_{\Theta} J_{lr}, \epsilon$)

Wear base layer?
Gradient descent for logistic regression

- Loss $J_{lr}(\Theta) = J_{lr}(\theta, \theta_0)$ is differentiable & convex
- Run $\text{Gradient-Descent}(\Theta_{\text{init}}, \eta, J_{lr}, \nabla \Theta J_{lr}, \epsilon)$

Wear base layer?
Gradient descent for logistic regression

- Loss $J_{lr}(\Theta) = J_{lr}(\theta, \theta_0)$ is differentiable & convex
- Run Gradient-Descent ($\Theta_{init}, \eta, J_{lr}, \nabla \Theta J_{lr}, \epsilon$)
Gradient descent for logistic regression

- Loss $J_{lr}(\Theta) = J_{lr}(\theta, \theta_0)$ is differentiable & convex
- Run $\text{Gradient-Descent} (\Theta_{\text{init}}, \eta, J_{lr}, \nabla_{\Theta} J_{lr}, \epsilon)$
Gradient descent for logistic regression

- Loss $J_{lr}(\Theta) = J_{lr}(\theta, \theta_0)$ is differentiable & convex
- Run $\text{Gradient-Descent} \left( \Theta_{\text{init}}, \eta, J_{lr}, \nabla_\Theta J_{lr}, \epsilon \right)$

$J_{lr}(\theta, \theta_0)$
$\left( \theta_0 = 0 \right)$
Gradient descent for logistic regression

- Loss $J_{lr}(\Theta) = J_{lr}(\theta, \theta_0)$ is differentiable & convex
- Run $\text{Gradient-Descent}(\Theta_{\text{init}}, \eta, J_{lr}, \nabla\Theta J_{lr}, \epsilon)$

\[
J_{lr}(\theta, \theta_0) \\
(\theta_0 = 0)
\]
Logistic regression loss revisited

\[
J_{lr}(\Theta) = J_{lr}(\theta, \theta_0) = \frac{1}{n} \sum_{i=1}^{n} L_{nll}(\sigma(\theta^T x^{(i)} + \theta_0), y^{(i)})
\]
Logistic regression loss revisited

\[ J_{lr}(\Theta) = J_{lr}(\theta, \theta_0) \]

\[ = \frac{1}{n} \sum_{i=1}^{n} L_{nll}(\sigma(\theta^\top x^{(i)} + \theta_0), y^{(i)}) + \lambda \|\theta\|^2 \quad (\lambda \geq 0) \]
Logistic regression loss revisited

\[ J_{lr}(\Theta) = J_{lr}(\theta, \theta_0) \]
\[ = \frac{1}{n} \sum_{i=1}^{n} L_{\text{nll}}(\sigma(\theta^\top x^{(i)} + \theta_0), y^{(i)}) + \lambda \| \theta \|^2 \quad (\lambda \geq 0) \]

- A “regularizer” or “penalty” \( R(\theta) = \lambda \| \theta \|^2 \)
Logistic regression loss revisited

\[ J_{lr}(\Theta) = J_{lr}(\theta, \theta_0) \]

\[ = \frac{1}{n} \sum_{i=1}^{n} L_{nll}(\sigma(\theta^T x^{(i)} + \theta_0), y^{(i)}) + \lambda \|\theta\|^2 \quad (\lambda \geq 0) \]

- A “regularizer” or “penalty” \( R(\theta) = \lambda \|\theta\|^2 \)
- Penalizes being overly certain
Logistic regression loss revisited

\[ J_{lr}(\Theta) = J_{lr}(\theta, \theta_0) = \frac{1}{n} \sum_{i=1}^{n} L_{nll}(\sigma(\theta^\top x^{(i)} + \theta_0), y^{(i)}) + \lambda \|\theta\|^2 \quad (\lambda \geq 0) \]

- A “regularizer” or “penalty” \( R(\theta) = \lambda \|\theta\|^2 \)
- Penalizes being overly certain
- Objective is still differentiable & convex (gradient descent)
Logistic regression loss revisited

\[ J_{lr}(\Theta) = J_{lr}(\theta, \theta_0) \]

\[ = \frac{1}{n} \sum_{i=1}^{n} L_{nll}(\sigma(\theta^T x^{(i)} + \theta_0), y^{(i)}) + \lambda \|\theta\|^2 \quad (\lambda \geq 0) \]

• A “regularizer” or “penalty” \( R(\theta) = \lambda \|\theta\|^2 \)
• Penalizes being overly certain
• Objective is still differentiable & convex (gradient descent)
Logistic regression loss revisited

\[ J_{lr}(\Theta) = J_{lr}(\theta, \theta_0) \]

\[ = \frac{1}{n} \sum_{i=1}^{n} L_{\text{nll}}(\sigma(\theta^\top x^{(i)} + \theta_0), y^{(i)}) + \lambda \|\theta\|^2 \quad (\lambda \geq 0) \]

- A “regularizer” or “penalty” \( R(\theta) = \lambda \|\theta\|^2 \)
- Penalizes being overly certain
- Objective is still differentiable & convex (gradient descent)

\[ \lambda = 0 \quad \lambda = 0.01 \]
Logistic regression loss revisited

\[ J_{lr}(\Theta) = J_{lr}(\theta, \theta_0) = \frac{1}{n} \sum_{i=1}^{n} L_{nll}(\sigma(\theta^T x^{(i)} + \theta_0), y^{(i)}) + \lambda \|\theta\|^2 \quad (\lambda \geq 0) \]

- A “regularizer” or “penalty” \( R(\theta) = \lambda \|\theta\|^2 \)
- Penalizes being overly certain
- Objective is still differentiable & convex (gradient descent)

\[ \lambda = 0 \quad \lambda = 0.01 \quad \lambda = 0.1 \]
Logistic regression loss revisited

\[ J_{lr}(\Theta) = J_{lr}(\theta, \theta_0) \]
\[ = \frac{1}{n} \sum_{i=1}^{n} L_{nll}(\sigma(\theta^T x^{(i)} + \theta_0), y^{(i)}) + \lambda \| \theta \|^2 \quad (\lambda \geq 0) \]

- A “regularizer” or “penalty” \( R(\theta) = \lambda \| \theta \|^2 \)
- Penalizes being overly certain
- Objective is still differentiable & convex (gradient descent)

\[ \lambda = 0 \quad \lambda = 0.01 \quad \lambda = 0.1 \]

- How to choose hyperparameters? One option: consider a handful of possible values and compare via CV
Logistic regression learning algorithm

Exactly gradient descent with $f$ given by logistic regression objective
Logistic regression learning algorithm

LR-Gradient-Descent\left(\theta_{\text{init}}, \theta_{0,\text{init}}, \eta, \epsilon\right)

Exactly gradient descent with \( f \) given by logistic regression objective
Logistic regression learning algorithm

LR-Gradient-Descent(\(\theta_{\text{init}}, \theta_{0,\text{init}}, \eta, \epsilon\))

Initialize \(\theta^{(0)} = \theta_{\text{init}}\)

Initialize \(\theta_0^{(0)} = \theta_{0,\text{init}}\)

Exactly gradient descent with \(f\) given by logistic regression objective
Logistic regression learning algorithm

\text{LR-Gradient-Descent}(\theta_{init}, \theta_{0,init}, \eta, \epsilon)

Initialize \( \theta^{(0)} = \theta_{init} \)
Initialize \( \theta_0^{(0)} = \theta_{0,init} \)
Initialize \( t = 0 \)

Exactly gradient descent with \( f \) given by logistic regression objective
Logistic regression learning algorithm

**LR-Gradient-Descent**

\[
\begin{align*}
\text{Initialize } & \theta^{(0)} = \theta_{\text{init}} \\
\text{Initialize } & \theta_0^{(0)} = \theta_{0,\text{init}} \\
\text{Initialize } & t = 0 \\
\text{repeat}
\end{align*}
\]

Exactly gradient descent with \( f \) given by logistic regression objective.
Logistic regression learning algorithm

\[ \text{LR-Gradient-Descent} \left( \theta_{\text{init}}, \theta_{0,\text{init}}, \eta, \epsilon \right) \]

- Initialize \( \theta^{(0)} = \theta_{\text{init}} \)
- Initialize \( \theta_0^{(0)} = \theta_{0,\text{init}} \)
- Initialize \( t = 0 \)

repeat
- \( t = t + 1 \)

Exactly gradient descent with \( f \) given by logistic regression objective
Logistic regression learning algorithm

\text{LR-Gradient-Descent}(\theta_{\text{init}}, \theta_{0,\text{init}}, \eta, \epsilon)

Initialize $\theta^{(0)} = \theta_{\text{init}}$

Initialize $\theta_0^{(0)} = \theta_{0,\text{init}}$

Initialize $t = 0$

repeat

\begin{align*}
  t &= t + 1 \\
  \theta^{(t)} &= \theta^{(t-1)} - \eta \left\{ \frac{1}{n} \sum_{i=1}^{n} \left[ \sigma(\theta^{(t-1)} \mathbf{x}^{(i)} + \theta_0^{(t-1)}) + \theta_0^{(t-1)} \right] \mathbf{x}^{(i)} \\
  \theta_0^{(t)} &= \theta_0^{(t-1)} - \eta \left\{ \frac{1}{n} \sum_{i=1}^{n} \left[ \sigma(\theta^{(t-1)} \mathbf{x}^{(i)} + \theta_0^{(t-1)}) - y^{(i)} \right] \right\} 
\end{align*}

Exactly gradient descent with $f$ given by logistic regression objective
Logistic regression learning algorithm

\textbf{LR-Gradient-Descent} \left( \theta_{\text{init}}, \theta_{0,\text{init}}, \eta, \epsilon \right)

Initialize \( \theta^{(0)} = \theta_{\text{init}} \)
Initialize \( \theta_{0}^{(0)} = \theta_{0,\text{init}} \)
Initialize \( t = 0 \)

\textbf{repeat}

\( t = t + 1 \)

\( \theta^{(t)} = \theta^{(t-1)} - \eta \left\{ \frac{1}{n} \sum_{i=1}^{n} \left[ \sigma(\theta^{(t-1)\top} x^{(i)} + \theta_{0}^{(t-1)}) - y^{(i)} \right] x^{(i)} \right\} \)

\( \theta_{0}^{(t)} = \theta_{0}^{(t-1)} - \eta \left\{ \frac{1}{n} \sum_{i=1}^{n} \left[ \sigma(\theta^{(t-1)\top} x^{(i)} + \theta_{0}^{(t-1)}) - y^{(i)} \right] \right\} \)

\textbf{until} \quad \left| J_{lr}(\theta^{(t)}, \theta_{0}^{(t)}) - J_{lr}(\theta^{(t-1)}, \theta_{0}^{(t-1)}) \right| < \epsilon

Exactly gradient descent with \( f \) given by logistic regression objective
Logistic regression learning algorithm

LR-Gradient-Descent($\theta_{\text{init}}, \theta_{0,\text{init}}, \eta, \epsilon$)

Initialize $\theta^{(0)} = \theta_{\text{init}}$
Initialize $\theta_0^{(0)} = \theta_{0,\text{init}}$
Initialize $t = 0$

repeat

$t = t + 1$

$\theta^{(t)} = \theta^{(t-1)} - \eta \left\{ \frac{1}{n} \sum_{i=1}^{n} \left[ \sigma(\theta^{(t-1)\top} x^{(i)} + \theta_0^{(t-1)}) - y^{(i)} \right] x^{(i)} + 2\lambda \theta^{(t-1)} \right\}$

$\theta_0^{(t)} = \theta_0^{(t-1)} - \eta \left\{ \frac{1}{n} \sum_{i=1}^{n} \left[ \sigma(\theta^{(t-1)\top} x^{(i)} + \theta_0^{(t-1)}) - y^{(i)} \right] \right\}$

until $|J_{lr}(\theta^{(t)}, \theta_0^{(t)}) - J_{lr}(\theta^{(t-1)}, \theta_0^{(t-1)})| < \epsilon$

Return $\theta^{(t)}, \theta_0^{(t)}$