26: Logistic Regression

Jerry Cain
May 25, 2022
Linear to Logical: Preparation
1. Weighted sum

If $X = (X_1, X_2, \ldots, X_m)$:

$$Z = \theta_1 X_1 + \theta_2 X_2 + \cdots + \theta_m X_m$$

$$= \sum_{j=1}^{m} \theta_j X_j$$

weighted sum

$$= \theta^T X$$

dot product
1. Weighted sum

Recall the linear regression model, where \( \mathbf{X} = (X_1, X_2, \ldots, X_m) \) and \( Y \in \mathbb{R} \):

\[
g(\mathbf{X}) = \theta_0 + \sum_{j=1}^{m} \theta_j X_j
\]

How would you rewrite this expression as a single dot product?

\[
g(\mathbf{X}) = \theta_0 X_0 + \theta_1 X_1 + \theta_2 X_2 + \cdots + \theta_m X_m \quad \text{Define } X_0 = 1
\]

\[
= \theta^T \mathbf{X}
\]

New \( \mathbf{X} = (1, X_1, X_2, \ldots, X_m) \)

Prepending \( X_0 = 1 \) to each feature vector \( \mathbf{X} \) makes matrix operators more accessible.
2. Sigmoid function $\sigma(z)$

- The sigmoid function:

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

- Sigmoid squashes $z$ to a number between 0 and 1.

- Recall definition of probability: A number between 0 and 1

$\sigma(z)$ can represent a probability.
3. Conditional likelihood function

Training data (n datapoints):
- \((x^{(i)}, y^{(i)})\) drawn iid from a distribution \(f(X = x^{(i)}, Y = y^{(i)}|\theta) = f(x^{(i)}, y^{(i)}|\theta)\)

\[
\theta_{MLE} = \arg \max_{\theta} \prod_{i=1}^{n} f(y^{(i)}|x^{(i)}, \theta) \quad \text{conditional likelihood of training data}
\]

\[
= \arg \max_{\theta} \sum_{i=1}^{n} \log f(y^{(i)}|x^{(i)}, \theta) \quad \text{log conditional likelihood}
\]

\[
= \arg \max_{\theta} LL(\theta)
\]

- MLE here is estimator that maximizes \textit{conditional} likelihood
- Confusingly, log conditional likelihood is also written as \(LL(\theta)\)
Logistic Regression
Prediction models so far

Linear Regression (Regression)

\[
\hat{Y} = \theta_0 + \sum_{j=1}^{m} \theta_j X_j
\]

- \(\mathbf{X}\) can be dependent
- Regression model (\(\hat{Y} \in \mathbb{R}\), not discrete)

Naïve Bayes (Classification)

\[
\hat{Y} = \arg \max_{y=\{0,1\}} P(Y | X)
\]

- Tractable with NB assumption, but...
- Realistically, \(X_j\) features not always conditionally independent
- Actually models \(P(X, Y)\), not \(P(Y | X)\)

Lisa Yan, Chris Piech, Mehran Sahami, and Jerry Cain, CS109, Spring 2022
Logistic Regression

Logistic Regression Model:

\[
P(Y = 1|X = x) = \sigma \left( \theta_0 + \sum_{j=1}^{m} \theta_j x_j \right)
\]

Predict \( \hat{Y} \) as the more likely \( Y \) given our observation \( X = x \):

\[
\hat{Y} = \arg \max_{y \in \{0,1\}} P(Y \mid X)
\]

- Since \( Y \in \{0,1\} \),
  \[
P(Y = 0|X = x) = 1 - \sigma \left( \theta_0 + \sum_{j=1}^{m} \theta_j x_j \right)
  \]
- Sigmoid function also known as logit function

Lisa Yan, Chris Piech, Mehran Sahami, and Jerry Cain, CS109, Spring 2022
Logistic Regression

\[ x = [0, 1, 1] \]

\[ P(Y = 1 | X = x) \]
conditional likelihood

\[ P(Y = 1 | X = x) = \sigma \left( \theta_0 + \sum_{j=1}^{m} \theta_j x_j \right) \]

Lisa Yan, Chris Piech, Mehran Sahami, and Jerry Cain, CS109, Spring 2022
Logistic Regression: Key Metaphor

$\theta$ parameter
Logistic Regression: Key Metaphor

\[
P(Y = 1|X = x) = \sigma \left( \theta_0 + \sum_{j=1}^{m} \theta_j x_j \right)
\]
Logistic Regression: Key Metaphor

\[ P(Y = 1 | X = x) = \sigma \left( \theta_0 + \sum_{j=1}^{m} \theta_j x_j \right) \]

\( \mathbf{X} \), input features 
\([0,1,1]\)

\( \mathbf{X} \), input features 
\([0,1,1]\)

\( \hat{Y} \), output
Components of Logistic Regression

\[ P(Y = 1|X = x) = \sigma \left( \theta_0 + \sum_{j=1}^{m} \theta_j x_j \right) \]
Components of Logistic Regression

\[ P(Y = 1|X = x) = \sigma \left( \theta_0 + \sum_{j=1}^{m} \theta_j x_j \right) \]

where \( \sigma(z) \) is the sigmoid function.
Components of Logistic Regression

\[ P(Y = 1 | X = x) = \sigma \left( \theta_0 + \sum_{j=1}^{m} \theta_j x_j \right) \]
Components of Logistic Regression

\[ P(Y = 1|X = x) = \sigma \left( \theta_0 + \sum_{j=1}^{m} \theta_j x_j \right) \]
Different predictions for different inputs

\[ P(Y = 1 | X = x) = \sigma \left( \theta_0 + \sum_{j=1}^{m} \theta_j x_j \right) \]

\( X, \) input features \([0, 1, 1]\)
Different predictions for different inputs

\[ P(Y = 1 | X = x) = \sigma \left( \theta_0 + \sum_{j=1}^{m} \theta_j x_j \right) \]

\( X, \text{ input features } [0,0,1] \)

\( z = -1.9 \)

\( \sigma(z) = 0.3 \)
Parameters affect prediction

\[ P(Y = 1 | X = x) = \sigma \left( \theta_0 + \sum_{j=1}^{m} \theta_j x_j \right) \]
Parameters affect prediction

\[ P(Y = 1|X = x) = \sigma \left( \theta_0 + \sum_{j=1}^{m} \theta_j x_j \right) \]
Parameters affect prediction

\[ P(Y = 1|X = x) = \sigma \left( \theta_0 + \sum_{j=1}^{m} \theta_j x_j \right) \]

\[ P(Y = 1|X = x) = \sigma \left( \sum_{j=0}^{m} \theta_j x_j \right) = \sigma (\theta^T x) \quad \text{where } x_0 = 1 \]
Logistic regression classifier

\[ \hat{Y} = \arg \max_{y \in \{0, 1\}} P(Y | X) \]

\[ P(Y = 1 | X = x) = \sigma \left( \sum_{j=0}^{m} \theta_j x_j \right) = \sigma (\theta^T x) \]

**Training**

Estimate parameters from training data

\[ \theta = (\theta_0, \theta_1, \theta_2, \ldots, \theta_m) \]

**Testing**

Given an observation \( X = (X_1, X_2, \ldots, X_m) \), predict

\[ \hat{Y} = \arg \max_{y \in \{0, 1\}} P(Y | X) \]
Training: The big picture
Logistic regression classifier

\[
\hat{Y} = \arg\max_{y=\{0, 1\}} P(Y|X) \\
\]

\[
P(Y = 1|X = x) = \sigma(\sum_{j=0}^{m} \theta_j x_j) = \sigma(\theta^T x)
\]

**Training**

Estimate parameters from training data

\[\theta = (\theta_0, \theta_1, \theta_2, ..., \theta_m)\]

Choose \(\theta\) that optimizes some objective:
1. Determine objective function
2. Find gradient with respect to \(\theta\)
3. Solve analytically by setting to 0, or solve computationally with gradient ascent

We are modeling \(P(Y|X)\) directly, so we maximize the **conditional likelihood** of training data.
Estimating $\theta$

1. Determine objective function

$$\theta_{MLE} = \arg \max_\theta \prod_{i=1}^{n} f(y^{(i)}| x^{(i)}, \theta)$$

2. Gradient w.r.t. $\theta_j$, for $j = 0, 1, \ldots, m$

3. Solve
   - No analytical derivation of $\theta_{MLE}$...
   - ...but can still compute $\theta_{MLE}$ with gradient ascent!

initialize $x$
repeat many times:
  compute gradient
  $x += \eta \times$ gradient
1. Determine objective function

\[ \theta_{MLE} = \arg \max_{\theta} \prod_{i=1}^{n} f(y^{(i)} | x^{(i)}, \theta) = \arg \max_{\theta} LL(\theta) \]

\[ P(Y = 1 | X = x) = \sigma \left( \sum_{j=0}^{m} \theta_j x_j \right) = \sigma (\theta^T x) \]

First: Interpret conditional likelihood with Logistic Regression
Second: Write a differentiable expression for log conditional likelihood
1. Determine objective function (interpret)

\[
\theta_{MLE} = \arg \max_\theta \prod_{i=1}^n f(y^{(i)} | x^{(i)}, \theta) = \arg \max_\theta LL(\theta)
\]

\[
P(Y = 1 | X = x) = \sigma(\sum_{j=0}^m \theta_j x_j)
= \sigma(\theta^T x)
\]

Suppose you have \(n = 2\) training datapoints: \((x^{(1)}, 1), (x^{(2)}, 0)\)

Consider the following expressions for a given \(\theta\):

A. \(\sigma(\theta^T x^{(1)}) \sigma(\theta^T x^{(2)})\)

B. \((1 - \sigma(\theta^T x^{(1)})) \sigma(\theta^T x^{(2)})\)

C. \(\sigma(\theta^T x^{(1)}) \left(1 - \sigma(\theta^T x^{(2)})\right)\)

D. \((1 - \sigma(\theta^T x^{(1)})) \left(1 - \sigma(\theta^T x^{(2)})\right)\)

1. Interpret the above expressions as probabilities.
2. If we let \(\theta = \theta_{MLE}\), which probability should be the highest?
1. Determine objective function (write)

\[ \theta_{MLE} = \arg \max_{\theta} \prod_{i=1}^{n} f(y^{(i)} | x^{(i)}, \theta) = \arg \max_{\theta} LL(\theta) \]

\[ P(Y = 1 | X = x) = \sigma \left( \sum_{j=0}^{m} \theta_j x_j \right) = \sigma (\theta^T x) \]

1. What is a differentiable expression for \( P(Y = y | X = x) \)?

\[ P(Y = y | X = x) = \begin{cases} \sigma (\theta^T x) & \text{if } y = 1 \\ 1 - \sigma (\theta^T x) & \text{if } y = 0 \end{cases} \]

2. What is a differentiable expression for \( LL(\theta) \), log conditional likelihood?

\[ LL(\theta) = \log \prod_{i=1}^{n} f(y^{(i)} | x^{(i)}, \theta) \]
1. Determine objective function (write)

\[ \theta_{MLE} = \arg \max_{\theta} \prod_{i=1}^{n} f(y^{(i)} | x^{(i)}, \theta) = \arg \max_{\theta} LL(\theta) \]

\[ P(Y = 1 | X = x) = \sigma \left( \sum_{j=0}^{m} \theta_j x_j \right) = \sigma(\theta^T x) \]

1. What is a differentiable expression for \( P(Y = y | X = x) \)?

\[ P(Y = y | X = x) = \begin{cases} \sigma(\theta^T x) & \text{if } y = 1 \\ 1 - \sigma(\theta^T x) & \text{if } y = 0 \end{cases} \]

Recall
Bernoulli MLE!

2. What is a differentiable expression for \( LL(\theta) \), log conditional likelihood?

\[ LL(\theta) = \log \prod_{i=1}^{n} f(y^{(i)} | x^{(i)}, \theta) \]
1. Determine objective function (write)

\[ \theta_{\text{MLE}} = \arg \max_{\theta} \prod_{i=1}^{n} f(y^{(i)}| x^{(i)}, \theta) = \arg \max_{\theta} LL(\theta) \]

\[ P(Y = 1|X = x) = \sigma(\sum_{j=0}^{m} \theta_j x_j) = \sigma(\theta^T x) \]

1. What is a differentiable expression for \( P(Y = y| X = x) \)?

\[ P(Y = y|X = x) = (\sigma(\theta^T x))^y (1 - \sigma(\theta^T x))^{1-y} \]

2. What is a differentiable expression for \( LL(\theta) \), log conditional likelihood?

\[ LL(\theta) = \sum_{i=1}^{n} y^{(i)} \log \sigma(\theta^T x^{(i)}) + (1 - y^{(i)}) \log \left(1 - \sigma(\theta^T x^{(i)})\right) \]
2. Find gradient with respect to $\theta$

Optimization problem:

$$\theta_{MLE} = \arg \max_{\theta} \prod_{i=1}^{n} f(y^{(i)}|x^{(i)}, \theta) = \arg \max_{\theta} LL(\theta)$$

$$LL(\theta) = \sum_{i=1}^{n} y^{(i)} \log \sigma(\theta^T x^{(i)}) + (1 - y^{(i)}) \log (1 - \sigma(\theta^T x^{(i)}))$$

Gradient w.r.t. $\theta_j$, for $j = 0, 1, \ldots, m$:

$$\frac{\partial LL(\theta)}{\partial \theta_j} = \sum_{i=1}^{n} \left[ y^{(i)} - \sigma(\theta^T x^{(i)}) \right] x_j^{(i)} \quad \text{(presented without proof)}$$

How do we interpret the gradient contribution of the i-th training datapoint?
2. Find gradient with respect to $\theta$

Optimization problem:

$$\theta_{MLE} = \arg \max_{\theta} \prod_{i=1}^{n} f(y^{(i)} \mid x^{(i)}, \theta) = \arg \max_{\theta} LL(\theta)$$

$$LL(\theta) = \sum_{i=1}^{n} y^{(i)} \log \sigma(\theta^T x^{(i)}) + (1 - y^{(i)}) \log (1 - \sigma(\theta^T x^{(i)}))$$

Gradient w.r.t. $\theta_j$, for $j = 0, 1, \ldots, m$:

$$\frac{\partial LL(\theta)}{\partial \theta_j} = \sum_{i=1}^{n} [y^{(i)} - \sigma(\theta^T x^{(i)})] x^{(i)}_j$$

(derived later)

scale by j-th feature
2. Find gradient with respect to $\theta$

Optimization problem:

$$\theta_{MLE} = \arg\max_{\theta} \prod_{i=1}^{n} f(y^{(i)}|x^{(i)}, \theta) = \arg\max_{\theta} LL(\theta)$$

$$LL(\theta) = \sum_{i=1}^{n} y^{(i)} \log \sigma(\theta^T x^{(i)}) + (1 - y^{(i)}) \log (1 - \sigma(\theta^T x^{(i)}))$$

Gradient w.r.t. $\theta_j$, for $j = 0, 1, ..., m$:

$$\frac{\partial LL(\theta)}{\partial \theta_j} = \sum_{i=1}^{n} [y^{(i)} - \sigma(\theta^T x^{(i)})] x_j^{(i)}$$

(presented without proof)
2. Find gradient with respect to $\theta$

Optimization problem:

$$\theta_{MLE} = \arg \max_{\theta} \prod_{i=1}^{n} f(y^{(i)} | x^{(i)}, \theta) = \arg \max_{\theta} LL(\theta)$$

$$LL(\theta) = \sum_{i=1}^{n} y^{(i)} \log \sigma(\theta^T x^{(i)}) + (1 - y^{(i)}) \log (1 - \sigma(\theta^T x^{(i)}))$$

Gradient w.r.t. $\theta_j$, for $j = 0, 1, ..., m$:

$$\frac{\partial LL(\theta)}{\partial \theta_j} = \sum_{i=1}^{n} \left[ y^{(i)} - \sigma(\theta^T x^{(i)}) \right] x_j^{(i)}$$

(derived later)

Suppose $y^{(i)} = 1$ (the true class label for $i$-th datapoint):

- If $\sigma(\theta^T x^{(i)}) \geq 0.5$, correct
- If $\sigma(\theta^T x^{(i)}) < 0.5$, incorrect $\rightarrow$ change $\theta_j$ more
3. Solve

1. Optimization problem:

\[ \theta_{MLE} = \arg \max_{\theta} \prod_{i=1}^{n} f(y^{(i)} | x^{(i)}, \theta) = \arg \max_{\theta} LL(\theta) \]

\[ LL(\theta) = \sum_{i=1}^{n} y^{(i)} \log \sigma(\theta^T x^{(i)}) + (1 - y^{(i)}) \log \left(1 - \sigma(\theta^T x^{(i)})\right) \]

2. Gradient w.r.t. \( \theta_j \), for \( j = 0, 1, \ldots, m \):

\[ \frac{\partial LL(\theta)}{\partial \theta_j} = \sum_{i=1}^{n} [y^{(i)} - \sigma(\theta^T x^{(i)})] x^{(i)}_j \]

3. Solve using gradient ascent
Training:
The details
Training: Gradient ascent step

\[
\frac{\partial LL(\theta)}{\partial \theta_j} = \sum_{i=1}^{n} [y^{(i)} - \sigma(\theta^T x^{(i)})] x_j^{(i)} \quad \text{for } j = 0, 1, \ldots, m
\]

repeat many times:

for all thetas:

\[
\theta_j^{\text{new}} = \theta_j^{\text{old}} + \eta \cdot \frac{\partial LL(\theta^{\text{old}})}{\partial \theta_j^{\text{old}}}
\]

\[
= \theta_j^{\text{old}} + \eta \cdot \sum_{i=1}^{n} [y^{(i)} - \sigma(\theta^{\text{old}}^T x^{(i)})] x_j^{(i)}
\]

What does this look like in code?
Training: Gradient Ascent

initialize $\theta_j = 0$ for $0 \leq j \leq m$

repeat many times:

$\text{gradient}[j] = 0$ for $0 \leq j \leq m$

// TODO: your code here

// compute all gradient[j]'s
// based on n training examples

$\theta_j += \eta \times \text{gradient}[j]$ for all $0 \leq j \leq m$
Training: Gradient Ascent

initialize $\theta_j = 0$ for $0 \leq j \leq m$
repeat many times:

gradient[$j$] = 0 for $0 \leq j \leq m$

for each training example $(x,y)$:

for each $0 \leq j \leq m$:

// update gradient[$j$] for current $(x,y)$ example

$\theta_j += \eta \ast \text{gradient}[j]$ for all $0 \leq j \leq m$
Training: Gradient Ascent

initialize $\theta_j = 0$ for $0 \leq j \leq m$
repeat many times:

gradient $[j] = 0$ for $0 \leq j \leq m$
for each training example $(x, y)$:

for each $0 \leq j \leq m$:

$$\text{gradient}[j] += \left[ y - \frac{1}{1 + e^{-\theta^Tx}} \right] x_j$$

$$\theta_j += \eta \times \text{gradient}[j] \text{ for all } 0 \leq j \leq m$$

Gradient Ascent Step

for $j = 0, 1, ..., m$:

$$\theta_{j}^{\text{new}} = \theta_{j}^{\text{old}} + \eta \cdot \sum_{i=1}^{n} \left[ y^{(i)} - \sigma \left( \theta_{old}^T x^{(i)} \right) \right] x_j^{(i)}$$

compute

outer loop

inner loop

Some important details...
Training: Gradient Ascent

**Gradient Ascent Step**

\[
\theta_j^{\text{new}} = \theta_j^{\text{old}} + \eta \cdot \sum_{i=1}^{n} \left[ y^{(i)} - \sigma \left( \theta^{\text{old}} x^{(i)} \right) \right] x_j^{(i)}
\]

- Initialize \( \theta_j = 0 \) for \( 0 \leq j \leq m \)
- Repeat many times:
  - Gradient \( j \) = 0 for \( 0 \leq j \leq m \)
  - For each training example \((x, y)\):
    - For each \( 0 \leq j \leq m \):
      \[
      \text{gradient}[j] += \left[ y - \frac{1}{1 + e^{-\theta^T x}} \right] x_j
      \]
    - \( \theta_j += \eta \times \text{gradient}[j] \) for all \( 0 \leq j \leq m \)

- Finish computing gradient with \( \theta^{\text{old}} \) prior to any \( \theta \) update
Training: Gradient Ascent

initialize $\theta_j = 0$ for $0 \leq j \leq m$
repeat many times:

gradient$[j] = 0$ for $0 \leq j \leq m$

for each training example $(x, y)$:

for each $0 \leq j \leq m$:

\[
\text{gradient}[j] += \left[ y - \frac{1}{1 + e^{-\theta^T x}} \right] x_j
\]

$\theta_j += \eta \times \text{gradient}[j]$ for all $0 \leq j \leq m$

- Finish computing gradient with $\theta^{old}$ prior to any $\theta$ update
- Learning rate $\eta$ is a constant you set before training
Training: Gradient Ascent

**Gradient Ascent Step**

\[
\theta_j^{\text{new}} = \theta_j^{\text{old}} + \eta \cdot \sum_{i=1}^{n} \left[ y^{(i)} - \sigma(\theta^{\text{old}}^T x^{(i)}) \right] x_j^{(i)}
\]

- Initialize \( \theta_j = 0 \) for \( 0 \leq j \leq m \)
- Repeat many times:
  - \( \text{gradient}[j] = 0 \) for \( 0 \leq j \leq m \)
  - For each training example \( (x, y) \):
    - For each \( 0 \leq j \leq m \):
      \[
      \text{gradient}[j] += \left[ y - \frac{1}{1 + e^{-\theta^T x}} \right] x_j
      \]
    - \( \theta_j += \eta \cdot \text{gradient}[j] \) for all \( 0 \leq j \leq m \)

- Finish computing gradient with \( \theta^{\text{old}} \) prior to any \( \theta \) update
- Learning rate \( \eta \) is a constant you set before training
- \( x_j \) is \( j \)-th feature of input \( x = (x_1, \ldots, x_m) \)
Training: Gradient Ascent

initialize $\theta_j = 0$ for $0 \leq j \leq m$
repeat many times:

gradient[j] = 0 for $0 \leq j \leq m$

for each training example $(x,y)$:

for each $0 \leq j \leq m$:

```
gradient[j] += \left[ y - \frac{1}{1 + e^{-\theta^T x}} \right] x_j
```

$\theta_j += \eta \times gradient[j]$ for all $0 \leq j \leq m$

Gradient Ascent Step

$$
\theta_j^{new} = \theta_j^{old} + \eta \cdot \sum_{i=1}^{n} \left[ y^{(i)} - \sigma \left( \theta^{old^T} x^{(i)} \right) \right] x_j^{(i)}
$$

- Finish computing gradient with $\theta^{old}$ prior to any $\theta$ update
- Learning rate $\eta$ is a constant you set before training
- $x_j$ is $j$-th feature of input $x = (x_1, \ldots, x_m)$
- Insert $x_0 = 1$ before training
Training: Gradient Ascent

Initialize $\theta_j = 0$ for $0 \leq j \leq m$

Repeat many times:

- For each training example $(x,y)$:
  - For each $0 \leq j \leq m$:
    - $\text{gradient}[j] = 0$ for $0 \leq j \leq m$
    - $\text{gradient}[j] += \left[ y - \frac{1}{1 + e^{-\theta^T x}} \right] x_j$

$\theta_j += \eta \times \text{gradient}[j]$ for all $0 \leq j \leq m$

Gradient Ascent Step

$$
\theta_j^{\text{new}} = \theta_j^{\text{old}} + \eta \cdot \sum_{i=1}^{n} \left[ y^{(i)} - \sigma \left( \theta^{\text{old}}^T x^{(i)} \right) \right] x_j^{(i)}
$$

- Finish computing gradient with $\theta^{\text{old}}$ prior to any $\theta$ update
- Learning rate $\eta$ is a constant you set before training
- $x_j$ is $j$-th feature of input $x = (x_1, ..., x_m)$
- Insert $x_0 = 1$ before training