25: Logistic Regression

Jerry Cain
March 8, 2024

Lecture Discussion on Ed
Linear to Logical: Preamble
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?

$$g(\mathbf{X}) = \theta_0 X_0 + \theta_1 X_1 + \theta_2 X_2 + \cdots + \theta_m X_m$$

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 convenient.
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 that expresses a belief that something is true.

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

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

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

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

$$
= \arg \max_{\theta} LL(\theta)
$$

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

Linear Regression (Regression)

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

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

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

Naïve Bayes (Classification)

\[ X \quad \hat{P}(X|Y)\hat{P}(Y) \quad \hat{P}(X, Y) \]

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

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

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

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

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

- **Input features**: $X = [0, 1, 1]$
- **Parameter**: $\theta$
- **Conditional likelihood**: $P(Y = 1 | X = x) = \sigma(\theta_0 + \sum_{j=1}^{m} \theta_j x_j)$

Where $\sigma$ is the sigmoid function.
Logistic Regression: Key Metaphor

\[ \theta \text{ parameter} \]
Logistic Regression: Key Metaphor

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

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

Lisa Yan, Chris Piech, Mehran Sahami, and Jerry Cain, CS109, Winter 2024
Logistic Regression: Key Metaphor

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

\(X\), input features

\([0,1,1]\)
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

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

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

where \( \sigma \) is a squashing function between 0 and 1.
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

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

\( \theta_0 \)

\( \theta_1 \)

\( \theta_2 \)

\( \theta_3 \)

\( x_0 \)

\( x_1 \)

\( x_2 \)

\( x_3 \)

\( X, \text{ input features} \)

\([0,1,1]\)
**Different predictions for different inputs**

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

\(X\), input features 
\([0,0,1]\)

\(\theta_0\), \(\theta_1\), \(\theta_2\), \(\theta_3\)

\(z = -1.9\), \(\sigma(z) = 0.3\)

Lisa Yan, Chris Piech, Mehran Sahami, and Jerry Cain, CS109, Winter 2024
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

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

where \( x_0 = 1 \)
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

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

**Testing**

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

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

$$\theta = (\theta_0, \theta_1, \theta_2, \ldots, \theta_m)$$

Choose $$\theta$$ that optimizes some objective:

1. Determine objective function
2. Find gradient with respect to each $$\theta$$
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 wrt $\theta_j$, for $j = 0, 1, \ldots, m$

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

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

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

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

First: Interpret conditional likelihood with Logistic Regression

Second: Write a differentiable expression for log conditional likelihood
1. Determine objective function (interpret)

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

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)

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

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

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

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)**

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

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

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 wrt $\theta_j$, for $j = 0, 1, \ldots, m$:

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

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

Optimization problem:

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

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

Gradient wrt $\theta_j$, for $j = 0, 1, \ldots, m$:

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

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

Optimization problem:

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

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

Gradient wrt $\theta_j$, for $j = 0, 1, \ldots, m$:

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

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

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

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

1. Optimization problem:
   \[
   \theta_{MLE} = \arg \max_{\theta} \prod_{i=1}^{n} f(y^{(i)} | x^{(i)}, \theta) = \arg \max_{\theta} LL(\theta)
   \]
   \[
   LL(\theta) = \sum_{i=1}^{n} y^{(i)} \log \sigma(\theta^T x^{(i)}) + (1 - y^{(i)}) \log (1 - \sigma(\theta^T x^{(i)}))
   \]

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

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

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

for \( j = 0, 1, \ldots, m \)

repeat until convergence:

for all thetas:

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

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

initialize $\theta_j = 0$ for $0 \leq j \leq m$
repeat until convergence:

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

// TODO: your code here
// compute all gradient[$j$]'s
// based on n training examples

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

initialize $\theta_j = 0$ for $0 \leq j \leq m$
repeat until convergence:
  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
      $\theta_j += \eta \cdot \text{gradient}[j]$ for all $0 \leq j \leq m$
Training: Gradient Ascent

- Initialize $\theta_j = 0$ for $0 \leq j \leq m$
- Repeat until convergence:
  - Gradient $\text{Ascent Step}$
    $\theta_{\text{new}}^j = \theta_{\text{old}}^j + \eta \cdot \sum_{i=1}^{n} \left[ y^{(i)} - \sigma \left( \theta_{\text{old}}^T x^{(i)} \right) \right] x_j^{(i)}$

  - 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 $0 \leq j \leq m$

Some important details...
Training: Gradient Ascent

**Gradient Ascent Step**

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

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

**Initialize** \( \theta_j = 0 \) for \( 0 \leq j \leq m \)

repeat until convergence:

- gradient\[j\] = 0 for \( 0 \leq j \leq m \)

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

  for each \( 0 \leq j \leq m \):

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

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

- 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 the $j^{th}$ feature of input $x = (x_1, ..., x_m)$.

**Gradient Ascent Step**

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

**Initialization**

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

**Repeat until convergence**:

- $\text{gradient}[j] = 0$ 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 $0 \leq j \leq m$
Training: Gradient Ascent

initialize $\theta_j = 0$ for $0 \leq j \leq m$
repeat until convergence:

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

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

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

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

$\theta_j += \eta \ast \text{gradient}[j]$ 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
- $x_j$ is the $j^{th}$ feature of input $x = (x_1, \ldots, x_m)$
- Insert $x_0 = 1$ before training
Training: Gradient Ascent

**Training:**

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

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

  - \( \theta_j \text{ } +\text{=} \text{ } \eta \text{ } \ast \text{ } \text{gradient}[j] \text{ } \text{for all} \text{ } 0 \leq j \leq m \)

- **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 the** \( j^{th} \) **feature of input** \( x = (x_1, ..., x_m) \)
- **Insert** \( x_0 = 1 \) **before training**
Naïve Bayes vs Logistic Regression

\[ \hat{P}(X|Y)\hat{P}(Y) \]

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

\[ P(Y = 1|X) \]

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

Compare/contrast:
1. What distributions are we modeling?
2. After learning our parameters, could we randomly generate a new datapoint \((x, y)\)?
3. Could we model a continuous \(X_j\) feature (e.g., \(X_j \sim \text{Normal}\), or \(X_j \sim \text{Unknown}\))?
4. Could we model a non-binary discrete \(X_j\) (e.g., \(X_j \in \{1, 2, \ldots, 6\}\))?
### Tradeoffs:

<table>
<thead>
<tr>
<th>Modeling goal</th>
<th>Naïve Bayes</th>
<th>Logistic Regression</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>1. Modeling goal</strong></td>
<td>$P(X,Y)$</td>
<td>$P(Y</td>
</tr>
<tr>
<td><strong>2. Generative or discriminative?</strong></td>
<td>Generative: could use joint distribution to generate new points (⚠ but you might not need this extra effort)</td>
<td>Discriminative: just tries to discriminate $y = 0$ vs $y = 1$ (❌ cannot generate new points b/c no $P(X,Y)$)</td>
</tr>
<tr>
<td><strong>3. Continuous input features</strong></td>
<td>⚠ Needs parametric form (e.g., Gaussian) or discretized buckets (for multinomial features)</td>
<td>Yes, easily</td>
</tr>
<tr>
<td><strong>4. Discrete input features</strong></td>
<td>Yes, multi-value discrete data = multinomial $P(X_i</td>
<td>Y)$</td>
</tr>
</tbody>
</table>
Logistic Regression is trying to find the line that separates data instances where $y = 1$ from those where $y = 0$:

- We call such data (or functions generating that data) **linearly separable**.

- Naïve Bayes is linear too, because there is one parameter for each feature (and no parameters that involve multiple features).
Data is not always linearly separable

- Not possible to draw a line that **successfully** separates all the $y = 1$ points (green) from the $y = 0$ points (red)
- Despite this, Logistic Regression and Naive Bayes still often work well in practice
Extra: Gradient Derivation
Background: Calculus

**Calculus refresher #1:**
Derivative(sum) = sum(derivative)

\[
\frac{\partial}{\partial x} \sum_{i=1}^{n} f_i(x) = \sum_{i=1}^{n} \frac{\partial f_i(x)}{\partial x}
\]

**Calculus refresher #2:**
Chain rule 🌟🌟🌟

\[
\frac{df(x)}{dx} = \frac{df(z)}{dz} \frac{dz}{dx}
\]

Calculus Chain Rule

\[f(x) = f(z(x))\]

aka decomposition of composed functions
Our goal

Find: \( \frac{\partial LL(\theta)}{\partial \theta_j} \)

where

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

Two "pre-processing" steps to prepare for chain rule

1. Rewrite \( LL(\theta) \) with \( \hat{y} \)
2. Compute gradient of \( \hat{y} \)
1. Rewriting $LL(\theta)$ with $\hat{y}$

Find: \[ \frac{\partial LL(\theta)}{\partial \theta_j} \]

where

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

log conditional likelihood

Let $\hat{y}^{(i)} = \sigma(\theta^T x^{(i)})$
2. Compute gradient of $\hat{y} = \sigma(\theta^T x)$

Aside: Sigmoid has a beautiful derivative!

Sigmoid function:

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

Derivative:

$$\frac{d}{dz} \sigma(z) = \sigma(z)[1 - \sigma(z)]$$
2. Compute gradient of $\hat{y} = \sigma(\theta^T x)$

Sigmoid function:

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

Derivative:

$$\frac{d}{dz}\sigma(z) = \sigma(z)[1 - \sigma(z)]$$

What is $\frac{\partial}{\partial \theta_j} \hat{y} = \frac{\partial}{\partial \theta_j} \sigma(\theta^T x)$?

A. $\sigma(x_j)[1 - \sigma(x_j)]x_j$
B. $\sigma(\theta^T x)[1 - \sigma(\theta^T x)]x$
C. $\sigma(\theta^T x)[1 - \sigma(\theta^T x)]x_j$
D. $\sigma(\theta^T x)x_j[1 - \sigma(\theta^T x)x_j]$
E. None/other
2. Compute gradient of $\hat{y} = \sigma(\theta^T x)$

Sigmoid function:

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

Derivative:

\[
\frac{d}{dz} \sigma(z) = \sigma(z)[1 - \sigma(z)]
\]

What is $\frac{\partial}{\partial \theta_j} \sigma(\theta^T x)$?  

A. $\sigma(x_j)[1 - \sigma(x_j)]x_j$  
B. $\sigma(\theta^T x)[1 - \sigma(\theta^T x)]x$  
C. $\sigma(\theta^T x)[1 - \sigma(\theta^T x)]x_j$  
D. $\sigma(\theta^T x)x_j[1 - \sigma(\theta^T x)x_j]$  
E. None/other

Let $z = \theta^T x = \sum_{k=0}^{m} \theta_k x_k$.

\[
\frac{\partial}{\partial \theta_j} \sigma(\theta^T x) = \frac{\partial}{\partial \sigma(z)} \sigma(z) \cdot \frac{\partial z}{\partial \theta_j}
\]

(Chain Rule)

\[
= \sigma(\theta^T x)[1 - \sigma(\theta^T x)]x_j
\]
Compute gradient of log conditional likelihood

\[
\frac{\partial LL(\theta)}{\partial \theta_j} = \sum_{i=1}^{n} \frac{\partial}{\partial \theta_j} \left[ y^{(i)} \log(\hat{y}^{(i)}) + (1 - y^{(i)}) \log(1 - \hat{y}^{(i)}) \right]
\]

Let \( \hat{y}^{(i)} = \sigma(\theta^T x^{(i)}) \)

\[
= \sum_{i=1}^{n} \frac{\partial}{\partial \hat{y}^{(i)}} \left[ y^{(i)} \log(\hat{y}^{(i)}) + (1 - y^{(i)}) \log(1 - \hat{y}^{(i)}) \right] \cdot \frac{\partial \hat{y}^{(i)}}{\partial \theta_j}
\]

(Chain Rule)

\[
= \sum_{i=1}^{n} \left[ y^{(i)} \frac{1}{\hat{y}^{(i)}} - (1 - y^{(i)}) \frac{1}{1 - \hat{y}^{(i)}} \right] \cdot \hat{y}^{(i)} (1 - \hat{y}^{(i)}) x_j^{(i)}
\]

(calculus)

\[
= \sum_{i=1}^{n} [y^{(i)} - \hat{y}^{(i)}] x_j^{(i)}
\]

(simplify)
Compute gradient of log conditional likelihood

\[
\frac{\partial LL(\theta)}{\partial \theta_j} = \sum_{i=1}^{n} \frac{\partial}{\partial \theta_j} [y^{(i)} \log(\hat{y}^{(i)}) + (1 - y^{(i)}) \log(1 - \hat{y}^{(i)})] \\
\]

Let \( \hat{y}^{(i)} = \sigma(\theta^T x^{(i)}) \)

\[
= \sum_{i=1}^{n} \frac{\partial}{\partial \hat{y}^{(i)}} [y^{(i)} \log(\hat{y}^{(i)}) + (1 - y^{(i)}) \log(1 - \hat{y}^{(i)})] \cdot \frac{\partial \hat{y}^{(i)}}{\partial \theta_j} \\
(\text{Chain Rule})
\]

\[
= \sum_{i=1}^{n} \left[ y^{(i)} \frac{1}{\hat{y}^{(i)}} - (1 - y^{(i)}) \frac{1}{1 - \hat{y}^{(i)}} \right] \cdot \hat{y}^{(i)} (1 - \hat{y}^{(i)}) x_j^{(i)} \\
(\text{calculus})
\]

\[
= \sum_{i=1}^{n} [y^{(i)} - \hat{y}^{(i)}] x_j^{(i)} = \sum_{i=1}^{n} [y^{(i)} - \sigma(\theta^T x^{(i)})] x_j^{(i)} \\
(\text{simplify})
\]