Linear Classifiers

Jaime Arguello
INLS 613: Text Data Mining
jarguell@email.unc.edu
Overview

- Philosophical questions
- Derivatives: What are they good for?
- Linear regression
- Multiple linear regression
- Logistic regression
Philosophical Questions

- What would you do if …
- What does this have to do with linear classifiers?
Functions

\[ x \rightarrow f(x) \rightarrow y \]
Derivatives

\[ y = 2x \]
Derivatives

\[ \frac{d}{dx} (2x) = 2 \]
Derivatives

\[ y = 2x - 10 \]

Graph of \( y = x^2 \)
Derivatives

\[ \frac{d}{dx} (x^2) = 2x \]
Derivatives: What are they good for?

- The derivative of $f(x)$ outputs the slope of $f(x)$ for a particular value of $x$.
- A point of which the slope is zero is a point at which $f(x)$ is at its highest or lowest value.
- What does this have to do with machine learning?
Derivatives

\[
\frac{d}{dx} (x^2) = 2x
\]

\[
2x = 0 \quad \Rightarrow \quad x = 0
\]
Computation Graphs

\[ y = 3(a + bc) \]

\[ f(a, b, c) \rightarrow y \]

\[ a \]

\[ b \]

\[ c \]
Computation Graphs

\[ y = 3(a + bc) \]

Diagram:
- \( a \) and \( b \) as inputs to \( u = (a + v) \)
- \( v = bc \)
- \( y = 3u \) as output
Derivatives: Chain Rule

\[ y = 3(a + bc) \]

\[ \frac{dy}{dc} = \frac{dv}{dc} \times \frac{du}{dv} \times \frac{dy}{du} \]

[Diagram showing the relationships between variables and derivatives]
Derivatives: Chain Rule

\[ y = 3(a + bc) \]

\[ \frac{dy}{dc} = b \times 1 \times 3 = 3b \]
Overview

- Philosophical questions
- Derivatives: What are they good for?
- Linear regression
- Multiple linear regression
- Logistic regression
Linear Regression

\[ y = wx + b \]
Linear Regression

# yawns from students

Temperature in Rm 001
Linear Regression

# yawns from students

\[ y = wx + b \]
Linear Regression: Training

\[ y = wx + b \]

- **Input:** set of \( m \) training examples \((x, y)\)
- Find the value of \( w \) and \( b \) that minimize the error:

\[
\sum_{i=1}^{m} (y^{(i)} - \hat{y}^{(i)})^2
\]
Linear Regression: Training

\[ y = wx + b \]

- Find the value of \( w \) and \( b \) that minimize the error:

\[
\sum_{i=1}^{m} \left( y^{(i)} - \hat{y}^{(i)} \right)^2
\]

\[
\sum_{i=1}^{m} \left( y^{(i)} - wx^{(i)} - b \right)^2
\]
Linear Regression: Training

\[ y = wx + b \]

- Find the value of \( w \) and \( b \) that minimize the error:

\[ \sum_{i=1}^{m} (y^{(i)} - wx^{(i)} - b)^2 \]

- Take the derivative with respect to \( w \), set it equal to 0, and solve for \( w \).
- Take the derivative with respect to \( b \), set it equal to 0, and solve for \( b \).
Linear Regression: Training

- Find the value of $w$ and $b$ that minimize the error:

$$w = \frac{1}{m} \sum_{i=1}^{m} \left( x^{(i)} - \bar{x} \right) \left( y^{(i)} - \bar{y} \right) \frac{\sum_{i=1}^{m} \left( x^{(i)} - \bar{x} \right)^2}{\sum_{i=1}^{m} \left( x^{(i)} - \bar{x} \right)^2}$$

$$b = \bar{y} - w\bar{x}$$
Linear Regression: Training

- Find the value of $w$ and $b$ that minimize the error:

$$w = \frac{1}{m} \sum_{i=1}^{m} (x^{(i)} - \bar{x}) (y^{(i)} - \bar{y})$$

$$b = \bar{y} - w\bar{x}$$

Always positive!

It depends!
Linear Regression: Prediction

\[ y = wx + b \]
Overview

- Philosophical questions
- Derivatives: What are they good for?
- Linear regression
- Multiple linear regression
- Logistic regression
Multiple Linear Regression

\[ f(x_1, x_2, \ldots, x_n) \]

\[ y = \sum_{j=1}^{n} (w_j x_j) + b \]
Multiple Linear Regression

<table>
<thead>
<tr>
<th>Size (feet)</th>
<th>No. of bedrooms</th>
<th>No. of floors</th>
<th>Age (years)</th>
<th>Price (x$1000)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2,350</td>
<td>5</td>
<td>2</td>
<td>45</td>
<td>500</td>
</tr>
<tr>
<td>1,600</td>
<td>3</td>
<td>2</td>
<td>20</td>
<td>450</td>
</tr>
<tr>
<td>2,000</td>
<td>3</td>
<td>2</td>
<td>30</td>
<td>250</td>
</tr>
<tr>
<td>854</td>
<td>2</td>
<td>1</td>
<td>10</td>
<td>200</td>
</tr>
<tr>
<td>560</td>
<td>1</td>
<td>1</td>
<td>30</td>
<td>180</td>
</tr>
</tbody>
</table>
Multiple Linear Regression: Training

- Given:
  \[
  \{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \ldots, (x^{(m)}, y^{(m)})\}
  \]

- We want:
  \[
  \hat{y}^{(i)} \approx y^{(i)}
  \]
Multiple Linear Regression: Training

- **Loss Function:** the discrepancy between the predicted and actual output values for a single training instance

\[ \mathcal{L}(y^{(i)}, \hat{y}^{(i)}) = \frac{1}{2} (y^{(i)} - \hat{y}^{(i)})^2 \]
Multiple Linear Regression: Training

• **Cost Function**: the discrepancy between the predicted and actual output values for all training instances

\[
\mathcal{L}(y^{(i)}, \hat{y}^{(i)}) = \frac{1}{2} (y^{(i)} - \hat{y}^{(i)})^2
\]

\[
J(w, b) = \frac{1}{m} \sum_{i=1}^{m} \left( \frac{1}{2} (y^{(i)} - \hat{y}^{(i)})^2 \right)
\]
Derivatives

\[ \frac{d}{dx} (x^2) = 2x \]

\[ 2x = 0 \]

\[ x = 0 \]
Gradient Descent: Intuition

\[
\frac{d}{dx} (x^2) = 2x
\]
Gradient Descent: Intuition
Multiple Linear Regression

<table>
<thead>
<tr>
<th>Size (feet)</th>
<th>No. of bedrooms</th>
<th>No. of floors</th>
<th>Age (years)</th>
<th>Price (x$1000)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2,350</td>
<td>5</td>
<td>2</td>
<td>45</td>
<td>500</td>
</tr>
<tr>
<td>1,600</td>
<td>3</td>
<td>2</td>
<td>20</td>
<td>450</td>
</tr>
<tr>
<td>2,000</td>
<td>3</td>
<td>2</td>
<td>30</td>
<td>250</td>
</tr>
<tr>
<td>854</td>
<td>2</td>
<td>1</td>
<td>10</td>
<td>200</td>
</tr>
<tr>
<td>560</td>
<td>1</td>
<td>1</td>
<td>30</td>
<td>180</td>
</tr>
</tbody>
</table>
Multiple Linear Regression

\[ f(x_1, x_2, \ldots, x_n) \]

\[ y = \sum_{j=1}^{n} (w_j x_j) + b \]
Gradient Descent

• Loss Function: the discrepancy between the predicted and actual output values for a single training instance

\[ \mathcal{L}(y^{(i)}, \hat{y}^{(i)}) = \frac{1}{2} (y^{(i)} - \hat{y}^{(i)})^2 \]

• Let’s see what the slope of the loss function is with respect to parameter \( b \)!

• Note: this will only consider one training example!
Gradient Descent

• Derivative of the loss function with respect to $b$

$$\mathcal{L}(y^{(i)}, \hat{y}^{(i)}) = \frac{1}{2} (y^{(i)} - \hat{y}^{(i)})^2$$

$$\mathcal{L}(y^{(i)}, \hat{y}^{(i)}) = \frac{1}{2} \left( y^{(i)} - \sum_{j=1}^{n} (w_j x_j^{(i)}) - b \right)^2$$

$$\frac{d}{db} \mathcal{L}(y^{(i)}, \hat{y}^{(i)}) = - (y^{(i)} - \hat{y}^{(i)})$$

$$\frac{d}{db} \mathcal{L}(y^{(i)}, \hat{y}^{(i)}) = \hat{y}^{(i)} - y^{(i)}$$
Gradient Descent

\[
\frac{d}{db} \mathcal{L}(y^{(i)}, \hat{y}^{(i)}) = \hat{y}^{(i)} - y^{(i)}
\]

<table>
<thead>
<tr>
<th>Scenario</th>
<th>(\hat{y}^{(i)} - y^{(i)})</th>
<th>Action!</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\hat{y}^{(i)} &gt; y^{(i)})</td>
<td>+</td>
<td>Decrease (nudge left)</td>
</tr>
<tr>
<td>(\hat{y}^{(i)} &lt; y^{(i)})</td>
<td>--</td>
<td>Increase (nudge right)</td>
</tr>
<tr>
<td>(\hat{y}^{(i)} \approx y^{(i)})</td>
<td>0</td>
<td>Do nothing!</td>
</tr>
</tbody>
</table>

\[
y = \sum_{j=1}^{n} (w_j x_j) + b
\]
Gradient Descent

• Loss Function: the discrepancy between the predicted and actual output values for a single training instance

\[ L(y^{(i)}, \hat{y}^{(i)}) = \frac{1}{2}(y^{(i)} - \hat{y}^{(i)})^2 \]

• Let’s see what the slope of the loss function is with respect to parameter \( w_j \)!

• Note: this will only consider one training example!
Gradient Descent

• Derivative of the loss function with respect to $w_j$

$$\mathcal{L}(y^{(i)}, \hat{y}^{(i)}) = \frac{1}{2} (y^{(i)} - \hat{y}^{(i)})^2$$

$$\mathcal{L}(y^{(i)}, \hat{y}^{(i)}) = \frac{1}{2} \left( y^{(i)} - \sum_{j=1}^{n} (w_j x_j^{(i)}) - b \right)^2$$

$$\frac{d}{dw_j} \mathcal{L}(y^{(i)}, \hat{y}^{(i)}) = -x_j^{(i)} \left( y^{(i)} - \hat{y}^{(i)} \right)$$

$$\frac{d}{dw_j} \mathcal{L}(y^{(i)}, \hat{y}^{(i)}) = (\hat{y}^{(i)} - y^{(i)}) x_j$$
Gradient Descent

\[
\frac{d}{dw_j} \mathcal{L}(y(i), \hat{y}(i)) = (\hat{y}(i) - y(i))x_j
\]

<table>
<thead>
<tr>
<th>Scenario</th>
<th>( \hat{y}(i) - y(i) )</th>
<th>Action!</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \hat{y}(i) &gt; y(i) )</td>
<td>+</td>
<td>Go in the opposite direction as ( x_j^{(i)} )</td>
</tr>
<tr>
<td>( \hat{y}(i) &lt; y(i) )</td>
<td>-</td>
<td>Go in the same direction as ( x_j^{(i)} )</td>
</tr>
<tr>
<td>( \hat{y}(i) \approx y(i) )</td>
<td>0</td>
<td>Do nothing!</td>
</tr>
</tbody>
</table>

\[
y = \sum_{j=1}^{n} (w_j x_j) + b
\]
Gradient Descent

- **Loss Function:** the discrepancy between the predicted and actual output values for a single training instance

\[ \mathcal{L}(y^{(i)}, \hat{y}^{(i)}) = \frac{1}{2}(y^{(i)} - \hat{y}^{(i)})^2 \]

- Given one training example, we can take derivatives with respect to each parameter to see what direction we should be going to minimize the loss function.
Gradient Descent

• Repeat many times (or until convergence):

\[
b \leftarrow b - \alpha \frac{1}{m} \sum_{i=1}^{m} \left( \hat{y}^{(i)} - y^{(i)} \right)
\]

\[
w_j \leftarrow w_j - \alpha \frac{1}{m} \sum_{i=1}^{m} \left( \left( \hat{y}^{(i)} - y^{(i)} \right) x_j^{(i)} \right)
\]
Gradient Descent

- Repeat many times (or until convergence):

\[ b \leftarrow b - \alpha \frac{1}{m} \sum_{i=1}^{m} (\hat{y}^{(i)} - y^{(i)}) \]

- If we are overshooting the target, reduce \( b \)
- If we are undershooting the target, increase \( b \)
- Otherwise, do nothing
Gradient Descent

• Repeat many times (or until convergence):

\[
w_j \leftarrow w_j - \alpha \frac{1}{m} \sum_{i=1}^{m} \left( (\hat{y}^{(i)} - y^{(i)}) x_j^{(i)} \right)
\]

• If we are overshooting the target, reduce \( w_j \) proportional to the value of \( x_j \)

• If we are undershooting the target, increase \( w_j \) proportional to the value of \( x_j \)

• Otherwise, do nothing
Overview

- Philosophical questions
- Derivatives: What are they good for?
- Linear regression
- Multiple linear regression
- Logistic regression
Logistic Regression

- Linear regression: predict $y$ given $x$
- Multiple linear regression: predict $y$ given $x_1, x_2, \ldots, x_n$
Logistic Regression

- **Logistic Regression**: predict $P(y=1|x_1, x_2, ..., x_n)$
- We can use logistic regression to do binary classification.
Logistic Regression

\[ f(x_1, x_2, \ldots, x_n) \rightarrow P(y = 1|x) \]
Logistic Regression

\[ f(x_1, x_2, \ldots, x_n) \rightarrow P(y = 1|x) \]

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

\[ z = \sum_{j=1}^{n} (w_j x_j) + b \]
Logistic Regression

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

- **Loss Function:** the discrepancy between the predicted and actual output values for a single training instance.

\[
    z = \sum_{j=1}^{n} (w_j x_j) + b
\]

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

\[
    \mathcal{L}(y^{(i)}, \hat{y}^{(i)}) = - \left( y^{(i)} \log \hat{y}^{(i)} + (1 - y^{(i)}) \log(1 - \hat{y}^{(i)}) \right)
\]
Logistic Regression

- **Loss Function:** the discrepancy between the predicted and actual output values for a single training instance

\[
L(y^{(i)}, \hat{y}^{(i)}) = - \left( y^{(i)} \log \hat{y}^{(i)} + (1 - y^{(i)}) \log(1 - \hat{y}^{(i)}) \right)
\]

- If the true value is 1, we want the predicted value to be **high**.
- Remember: \(\log(1) = 0\)
Logistic Regression

- **Loss Function**: the discrepancy between the predicted and actual output values for a single training instance

\[
\mathcal{L}(y^{(i)}, \hat{y}^{(i)}) = - \left( y^{(i)} \log \hat{y}^{(i)} + (1 - y^{(i)}) \log(1 - \hat{y}^{(i)}) \right)
\]

- If the true value is 0, we want the predicted value to be **low**.
- Remember: \(\log(1) = 0\)
Logistic Regression

- **Loss Function:** the discrepancy between the predicted and actual output values for a single training instance

\[
z = \sum_{j=1}^{n} (w_j x_j) + b
\]

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

\[
\mathcal{L}(y^{(i)}, \hat{y}^{(i)}) = -\left( y^{(i)} \log \hat{y}^{(i)} + (1 - y^{(i)}) \log(1 - \hat{y}^{(i)}) \right)
\]

\[
\frac{d}{db} \mathcal{L}(y^{(i)}, \hat{y}^{(i)}) = \hat{y}^{(i)} - y^{(i)}
\]
Logistic Regression

- **Loss Function:** the discrepancy between the predicted and actual output values for a single training instance

\[ z = \sum_{j=1}^{n} (w_j x_j) + b \]

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

\[ \mathcal{L}(y^{(i)}, \hat{y}^{(i)}) = - \left( y^{(i)} \log \hat{y}^{(i)} + (1 - y^{(i)}) \log(1 - \hat{y}^{(i)}) \right) \]

\[ \frac{d}{dw_j} \mathcal{L}(y^{(i)}, \hat{y}^{(i)}) = (\hat{y}^{(i)} - y^{(i)}) x_j \]
Logistic Regression: 
Gradient Descent

• Repeat many times (or until convergence):

\[ b \leftarrow b - \alpha \frac{1}{m} \sum_{i=1}^{m} \left( \hat{y}^{(i)} - y^{(i)} \right) \]

\[ w_j \leftarrow w_j - \alpha \frac{1}{m} \sum_{i=1}^{m} \left( \hat{y}^{(i)} - y^{(i)} \right) x_j^{(i)} \]
Logistic Regression:
Gradient Descent

- Repeat many times (or until convergence):

\[ b \leftarrow b - \alpha \frac{1}{m} \sum_{i=1}^{m} \left( \hat{y}^{(i)} - y^{(i)} \right) \]

- If we are overshooting the target, reduce \( b \)
- If we are undershooting the target, increase \( b \)
- Otherwise, do nothing
Logistic Regression: Gradient Descent

• Repeat many times (or until convergence):

\[ w_j \leftarrow w_j - \alpha \frac{1}{m} \sum_{i=1}^{m} \left( (\hat{y}^{(i)} - y^{(i)}) x_j^{(i)} \right) \]

• If we are overshooting the target, reduce \( w_j \) proportional to the value of \( x_j \)

• If we are undershooting the target, increase \( w_j \) proportional to the value of \( x_j \)

• Otherwise, do nothing
Overview

- Philosophical questions
- Derivatives: What are they good for?
- Linear regression
- Multiple linear regression
- Logistic regression
The Big Picture!

- Linear regression, multiple linear regression and logistic regression are examples of linear models.
- Internally, linear models output a prediction based on a weighted combination of input features.
- Features that are positively correlated with the target output get a high weight.
- Features that are negatively correlated with the target output get a negative weight.
- Features that are uncorrelated with the target output get a zero weight.