Lecture 10 (part 1): Iterative Optimization with Gradient Descent

COMP90049
Introduction to Machine Learning

Semester 1, 2024

Ting Dang, CIS

Acknowledgement: Lea Frermann

Roadmap

So far...

- Naive Bayes Classifier – theory and practice
- MLE estimation of parameters
- Exact optimization

Now: Quick aside on iterative optimization

- Gradient Descent
- Global and local optima

Finding Optimal Points I

Finding the parameters that optimize a **target**

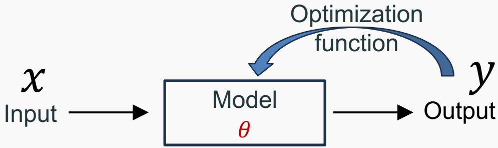
Ex1: Estimate the study time which leads to the **best grade** in COMP90049.

Ex2: Find the shoe price which leads to **maximum profit** of our shoe shop.

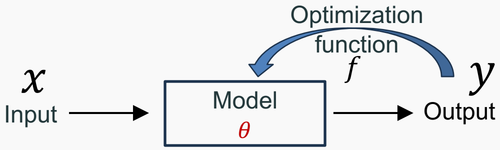
Ex3: Predicting **housing prices** from a weighted combination of house age and house location

Ex4: Find the parameters θ of a spam classifier which lead to the **lowest error**

Ex5: Find the parameters θ of a spam classifier which lead to the **highest data log likelihood**



Recipe for finding Minima / Maxima



1. Define your function of interest $f(y|x,\theta)$ (e.g., data log likelihood)
2. Compute its first derivative wrt its input θ
3. Set the derivative to zero
4. Solve for θ

Closed-form vs Iterative Optimization

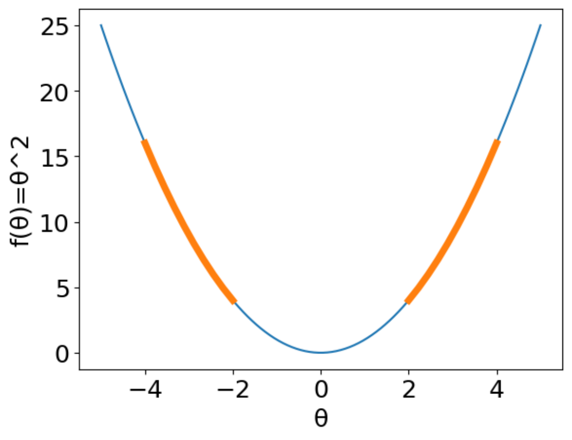
Closed-form solutions

- Previously, we computed the **closed form** solution for the MLE of the binomial distribution
- We follow our recipe, and arrive at a single solution

Unfortunately, life is not always as easy

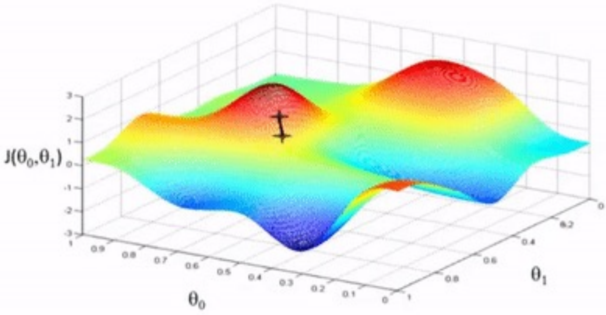
- Often, no closed-form solution exists
- Instead, we have to **iteratively** improve our estimate of $\hat{\theta}$ until we arrive at a satisfactory solution
- Gradient descent is one popular iterative optimization method

'Descending' the function to find the Optimum



- 1-dimensional case: find parameter θ that minimizes the function
- follow the curvature of the line step by step

‘Descending’ the function to find the Optimum



- 2-dimensional case: find parameters $\theta = [\theta_0, \theta_1]$ that minimize the function J
- follow the curvature step by step along the steepest way

Source: https://medium.com/binaryandmore/beginners-guide-to-deriving-and-implementing-backpropagation-e3c1a5a1e536

Intuition

- Descending a mountain (aka. our function) as fast as possible: at every position take the next step that takes you most directly into the valley
- We compute a series of solutions $\theta^{(0)}, \theta^{(1)}, \theta^{(2)}, \ldots$ by 'walking' along the function and taking steps in the direction with the steepest local slope (or gradient).
- each solution depends on the current location

Gradient Descent: Details

Learn the model parameters θ

- such that we **minimize the error**
- traverse over the loss function step by step ('descending into a valley')
- we would like an algorithm that tells how to update our parameters

$$\theta \leftarrow \theta + \Delta\theta$$

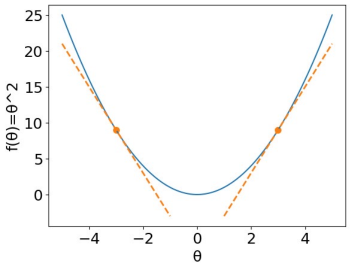
Gradient Descent: Details

Learn the model parameters θ

- such that we **minimize the error**
- traverse over the loss function step by step ('descending into a valley')
- we would like an algorithm that tells how to update our parameters

$$\theta \leftarrow \theta + \Delta\theta$$

- $\Delta\theta = \frac{\partial f(\theta)}{\partial \theta}$ is the **derivative**, a measure of change in the function f given a change in θ
- the derivative measures the **slope** or **gradient** of a function *f* at point θ
- for a function $f(\theta)$, $\frac{\partial f(\theta)}{\partial \theta}$ tells us how much *f* changes in response to a change in θ.



Gradient Descent: Details

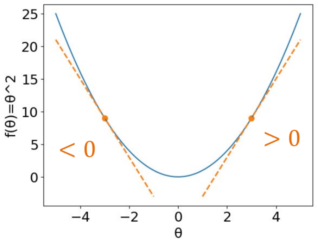
Learn the model parameters θ

- such that we **minimize the error**
- traverse over the loss function step by step ('descending into a valley')
- we would like an algorithm that tells how to update our parameters

$$\theta \leftarrow \theta + \Delta\theta$$

- if $\frac{\partial f(\theta)}{\partial \theta} > 0$: $f(\theta)$ increases as θ increases
- if $\frac{\partial f(\theta)}{\partial \theta} < 0$: $f(\theta)$ increases as θ decreases
- if $\frac{\partial f(\theta)}{\partial \theta} = 0$: we are at a minimum (or maximum)
- so, to approach the minimum:

$$\theta \leftarrow \theta - \eta \frac{\partial f}{\partial \theta}$$



Gradient Descent for multiple parameters

- Usually, our models have **several parameters** which need to be optimized to minimize the error
- We compute **partial derivatives** of $f(\theta)$ wrt. individual θ_i
- Partial derivatives measure change in a function of multiple parameters given a change in a single parameter, with all others held constant
- For example for $f(\theta_1, \theta_2)$ we can compute $\frac{\partial f}{\partial \theta_1}$ and $\frac{\partial f}{\partial \theta_2}$

Gradient Descent for multiple parameters

- Usually, our models have **several parameters** which need to be optimized to minimize the error
- We compute **partial derivatives** of $f(\theta)$ wrt. individual θ_i
- Partial derivatives measure change in a function of multiple parameters given a change in a single parameter, with all others held constant
- For example for $f(\theta_1, \theta_2)$ we can compute $\frac{\partial f}{\partial \theta_1}$ and $\frac{\partial f}{\partial \theta_2}$
- We then **update each parameter individually**

$$\theta_1 \leftarrow \theta_1 - \Delta\theta_1 \qquad \text{with } \Delta\theta_1 = \eta \frac{\partial f}{\partial \theta_1}$$

$$\theta_2 \leftarrow \theta_2 - \Delta\theta_2 \qquad \text{with } \Delta\theta_2 = \eta \frac{\partial f}{\partial \theta_2}$$

Gradient Descent: Recipe

Recipe for Gradient Descent (single parameter)

1: Define objective function $f(\theta)$
2: Initialize parameter $\theta^{(0)}$
3: **for** iteration $t \in \{0, 1, 2, \dots T\}$ **do**
4: Compute the first derivative of f at that point $\theta^{(t)}$: $\frac{\partial f}{\partial \theta^{\mathrm{t}}}$
5: Update your parameter by subtracting the (scaled) derivative

$$\theta^{t+1} \leftarrow \theta^{t} - \eta \frac{\partial f}{\partial \theta^{\mathrm{t}}}$$

- η is the **step size** or **learning rate**, a parameter
- When to stop? Fix number of iterations, or define other criteria

$$f(\theta^{t+1}) - f(\theta^{t}) < \gamma$$

Gradient Descent: Recipe

Recipe for Gradient Descent (multiple parameters)

Define objective function $f(\theta)$

Initialize parameters $\{\theta_1^{(0)}, \theta_2^{(0)}, \theta_3^{(0)}, \ldots\}$

for iteration $t \in \{0, 1, 2, \ldots T\}$ **do**

  Initialize vector of *gradients* $\leftarrow$ []

  for parameter $f \in \{1, 2, 3, \ldots F\}$ **do**

    Compute the first derivative of f at that point $\theta^{(t)}$: $\frac{\partial f}{\partial \theta_i^t}$

    append $\frac{\partial f}{\partial \theta_i^t}$ to *gradients*

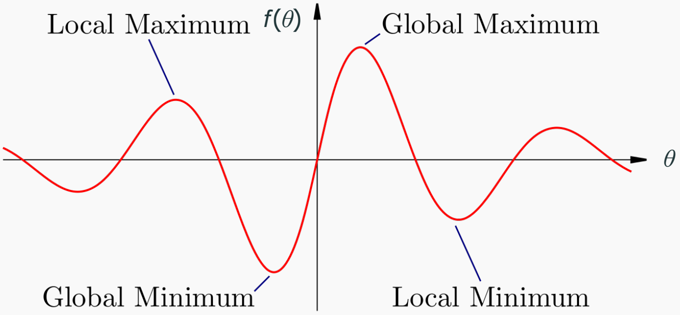
  Update all parameters by subtracting the (scaled) gradient

$$\theta_i^{t+1} \leftarrow \theta_i^t - \eta \frac{\partial f}{\partial \theta_i^t}$$

Aside: Global and Local Minima and Maxima

Possible issue: local maxima and minima!

- A global **maximum** is the single highest value of the function
- A global **minimum** is the single lowest value of the function
- A function is **convex** if a line between any two points of the function lies above the function



https://commons.wikimedia.org/wiki/File:Maxima_and_Minima.svg

Gradient Descent Guarantees

1. with an appropriate learning rate, GD will find the global minimum for differentiable convex functions
2. with an appropriate learning rate, GD will find a local minimum for differentiable non-convex functions

Equivalently, Gradient Ascent (“GA”) would find the global maximum (case 1.) and local maximum (case 2.)

Summary

Now you know:

- What optimization is, and why it's important
- How to do closed-form optimization (aka. "set the derivative of $f(\theta)$ to zero and solve for θ)
- That closed-form solutions are not always computable
- In that case, iterative optimization can help us
- Gradient descent is one instance of an iterative optimization method
- How gradient descent works!

Next lecture(s)

- Logistic Regression

Lecture 10 (part 2): Logistic Regression

COMP90049
Introduction to Machine Learning

Semester 1, 2024

Ting Dang, CIS

Acknowledgement: Lea Frermann

Roadmap

Sofar...

- Decision Trees, Naive Bayes and KNN
- Optimization (closed-form and iterative)
- Evaluation + Feature Selection

Today : back to classification!

- Logistic Regression

Logistic Regression

Quick Refresher

Recall **Naive Bayes**

$$\color{red}{P(x,y)} = P(y)P(x|y) = \prod_{i=1}^{N} P(y^i) \prod_{m=1}^{M} P(x_m^i|y^i)$$

- a **probabilistic generative model** of the joint probability $P(x,y)$
- optimized to maximize the likelihood of the observed data
- **naive** due to unrealistic feature indepencence assumptions

Quick Refresher

Recall **Naive Bayes**

$$P(x,y) = P(y)P(x|y) = \prod_{i=1}^{N} P(y^i) \prod_{m=1}^{M} P(x_m^i|y^i)$$

- a **probabilistic generative model** of the joint probability $P(x,y)$
- optimized to maximize the likelihood of the observed data
- **naive** due to unrealistic feature indepencence assumptions

For **prediction**, we apply **Bayes Rule** to obtain the conditional distribution

$$P(x,y) = P(y)P(x|y) = P(x)P(y|x)$$

$$P(y|x) = \frac{P(y)P(x|y)}{P(x)}$$

$$\hat{y} = \underset{y}{\operatorname{argmax}}\, P(y|x) \approx P(y)P(x|y)$$

How about we model $P(y|x)$ directly? → **Logistic Regression**

Logistic Regression on a high level

- Is a **binary** classification model
- Is a **probabilistic discriminative model** because it optimizes $P(y|x)$ directly
- Learns to optimally discriminate between inputs which belong to different classes
- No model of $P(x|y) \rightarrow$ no conditional feature independence assumption

Aside: Linear Regression

- Regression: predict a real-valued quantity y given features x, e.g.,

$$\begin{array}{lll}
\text{housing price} & \text{given} & \{\text{location, size, age, } \ldots\} \\
\text{success of movie (\$)} & \text{given} & \{\text{cast, genre, budget, } \ldots\} \\
\text{air quality} & \text{given} & \{\text{temperature, timeOfDay, CO2, } \ldots\}
\end{array}$$

Aside: Linear Regression

- Regression: predict a real-valued quantity y given features x, e.g.,

housing price	given	$\{\text{location, size, age, } \ldots\}$
success of movie (\$)	given	$\{\text{cast, genre, budget, } \ldots\}$
air quality	given	$\{\text{temperature, timeOfDay, CO2, } \ldots\}$

- **linear regression** is the simplet regression model
- a real-valued $\hat{y}$ is predicted as a linear combination of weighted feature values

$$\hat{y} = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots = \theta_0 + \sum_i \theta_i x_i$$

- The weights $\theta_0, \theta_1, \ldots$ are model parameters, and need to be optimized during training
- Loss (error) is the sum of squared errors (SSE): $L = \sum_{i=1}^{N} \left(\hat{y}^i - y^i\right)^2$

Logistic Regression: Derivation I

- Let's assume a **binary** classification task, y is $\texttt{true}$ (1) or $\texttt{false}$ (0).
- We model **probabilites** $P(y = 1|x; \theta)$ as a function of observations x under parameters θ.
- We want to use a **regression** approach

Logistic Regression: Derivation I

- Let's assume a **binary** classification task, y is `true` (1) or `false` (0).
- We model **probabilites** $P(y = 1|x; \theta)$ as a function of observations x under parameters θ.
- We want to use a **regression** approach
- How about: $P(y = 1|x; \theta)$ as a linear function of x. Problem: probabilities are bounded in 0 and 1, linear functions are not.

$$P(y = 1|x; \theta) = \theta_0 + \theta_1 x_1 + \ldots \theta_F x_F$$

Logistic Regression: Derivation I

- Let's assume a **binary** classification task, y is $\mathtt{true}$ (1) or $\mathtt{false}$ (0).
- We model **probabilites** $P(y = 1|x; \theta)$ as a function of observations x under parameters θ.
- We want to use a **regression** approach
- ~~How about: $P(y = 1|x; \theta)$ as a linear function of x. Problem: probabilities are bounded in 0 and 1, linear functions are not.~~

Logistic Regression: Derivation I

- Let's assume a **binary** classification task, y is $\texttt{true}$ (1) or $\texttt{false}$ (0).
- We model **probabilites** $P(y = 1|x; \theta)$ as a function of observations x under parameters θ.
- We want to use a **regression** approach
- ~~How about: $P(y = 1|x; \theta)$ as a linear function of x. Problem: probabilities are bounded in 0 and 1, linear functions are not.~~
- How about: *log* $P(y = 1|x; \theta)$ as a linear function of x. Problem: *log* is bounded in one direction, linear functions are not.

$$\log P(y = 1|x; \theta) = \theta_0 + \theta_1 x_1 + \ldots \theta_F x_F$$

Logistic Regression: Derivation I

- Let's assume a **binary** classification task, y is $\texttt{true}$ (1) or $\texttt{false}$ (0).
- We model **probabilites** $P(y = 1|x; \theta)$ as a function of observations x under parameters θ.
- We want to use a **regression** approach
- ~~How about: $P(y = 1|x; \theta)$ as a linear function of x. Problem: probabilities are bounded in 0 and 1, linear functions are not.~~
- ~~How about: $log\ P(y = 1|x; \theta)$ as a linear function of x. Problem: *log* is bounded in one direction, linear functions are not.~~

Logistic Regression: Derivation I

- Let's assume a **binary** classification task, y is $\mathtt{true}$ (1) or $\mathtt{false}$ (0).
- We model **probabilites** $P(y = 1/x; \theta)$ as a function of observations x under parameters θ.
- We want to use a **regression** approach
- ~~How about: $P(y = 1/x; \theta)$ as a linear function of x. Problem: probabilities are bounded in 0 and 1, linear functions are not.~~
- ~~How about: *log* $P(y = 1/x; \theta)$ as a linear function of x. Problem: *log* is bounded in one direction, linear functions are not.~~
- How about: minimally modifying *log* $P(y = 1/x; \theta)$ such that is is unbounded, by applying the **logistic** transformation

$$log \frac{P(y=1|x;\theta)}{1-P(y=1|x;\theta)} = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_F x_F$$

Logistic Regression: Derivation II

$$log \frac{P(y=1|x;\theta)}{1-P(y=1|x;\theta)} = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_F x_F$$

- also called the **log odds**
- the **odds** are defined as the fraction of success over the fraction of failures

$$odds = \frac{P(success)}{P(failures)} = \frac{P(success)}{1-P(success)}$$

- e.g., the odds of rolling a 6 with a fair dice are:

$$\frac{1/6}{1-(1/6)} = \frac{0.17}{0.83} = 0.2$$

Logistic Regression: Derivation III

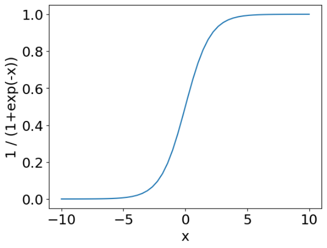
$$log \frac{P(y=1|x;\theta)}{1-P(y=1|x;\theta)} = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_F x_F$$

If we rearrange and solve for $P(y=1|x;\theta)$, we get

$$P(y=1|x;\theta) = \frac{\exp(\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_F x_F)}{1+\exp(\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_F x_F)}$$

$$= \frac{\exp(\theta_0 + \sum_i \theta_i x_i)}{1+\exp\left(\theta_0 + \sum_i \theta_i x_i\right)} = \frac{1}{1+\exp\left(-(\theta_0 + \sum_i \theta_i x_i)\right)}$$

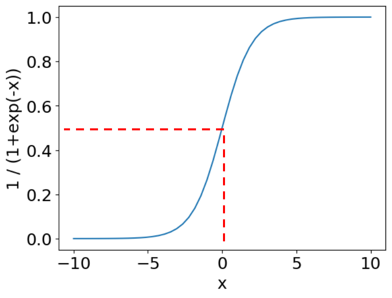
- The inverse logit (or **logistic function**)
- we pass a regression model through the logistic function to obtain a valid probability predicton



$$P(y=1|x;\theta) = \frac{1}{1+\exp\left(-(\theta_0 + \sum_i \theta_i x_i)\right)}$$

A closer look at the logistic function

Most inputs lead to $P(y|x)$=0 or $P(y|x)$=1. That is intended, because all true labels are either 0 or 1.



- $(\theta_0 + \sum_i \theta_i x_i) > 0$ means y=1
- $(\theta_0 + \sum_i \theta_i x_i) \approx 0$ means most uncertainty
- $(\theta_0 + \sum_i \theta_i x_i) < 0$ means y=0

Example!

$$P(y=1|x;\theta) = \frac{1}{1+\exp\left(-(\theta_0 + \sum_i \theta_i x_i)\right)} = \frac{1}{1+\exp(-(\boldsymbol{\theta}^{\boldsymbol{T}}\boldsymbol{X}))} = \sigma(\boldsymbol{\theta}^{\boldsymbol{T}}\boldsymbol{X})$$

Model parameters

θ = [0.1, −3.5, 0.7, 2.1]

$\theta_0 \quad \theta_1 \quad \theta_2 \quad \theta_3$

Feature Function

$$x_0 = 1\ (bias\ term)$$

$$x_1 = \begin{cases} 1\ if\ outlook = sunny \\ 2\ if\ outlook = overcast \\ 3\ if\ outlook = rainy \end{cases}$$

$$x_2 = \begin{cases} 1\ if\ temp = hot \\ 2\ if\ temp = mild \\ 3\ if\ temp = cool \end{cases}$$

$$x_3 = \begin{cases} 1\ if\ humidity = normal \\ 2\ if\ humidity = high \end{cases}$$

(Small) Test Data set

Outlook	Temp	Humidity	Class
rainy	*cool*	*normal*	0
sunny	*hot*	*high*	1

Parameter Estimation

What are the four steps we would follow in finding the optimal parameters?

Objective Function

Mimimize the **Negative conditional log likelihood**

$$L(\theta) = -P(y|x;\theta) = -\prod_{i=1}^{N} P(y_i|x_i;\theta)$$

note that

$$P(y = 1|x;\theta) = \sigma(\boldsymbol{\theta}^{\boldsymbol{T}}\boldsymbol{X})$$

$$P(y = 0|x;\theta) = 1 - \sigma(\boldsymbol{\theta}^{\boldsymbol{T}}\boldsymbol{X})$$

Objective Function

Mimimize the **Negative conditional log likelihood**

$$L(\theta) = -P(y|x;\theta) = -\prod_{i=1}^{N} P(y_i|x_i;\theta)$$

note that

$$P(y=1|x;\theta) = \sigma(\boldsymbol{\theta}^T \boldsymbol{X})$$

$$P(y=0|x;\theta) = 1 - \sigma(\boldsymbol{\theta}^T \boldsymbol{X})$$

so

$$L(\theta) = -P(y|x;\theta) = -\prod_{i=1}^{N} P(y_i|x_i;\theta)$$

$$= -\prod_{i=1}^{N} \left(\sigma(\boldsymbol{\theta}^T \boldsymbol{X})\right)^{y_i} \left(1 - \sigma(\boldsymbol{\theta}^T \boldsymbol{X})\right)^{1-y_i}$$

take the log of this function

$$\log L(\theta) = -\sum_{i=1}^{N} \left[y_i log\sigma(\boldsymbol{\theta}^T \boldsymbol{X}) + (1-y_i)\log\left(1 - \sigma(\boldsymbol{\theta}^T \boldsymbol{X})\right)\right]$$

Take 1st Derivative of the Objective Function

$$\log L(\theta) = -\sum_{i=1}^{N} \left[y_i log\sigma\left(\boldsymbol{\theta^T X}\right) + (1 - y_i) \log\left(1 - \sigma\left(\boldsymbol{\theta^T X}\right)\right)\right]$$

Preliminaries

- The derivative of the logistic (sigmoid) function is $\frac{\partial\sigma(z)}{\partial z} = \sigma(z)[1 - \sigma(z)]$
- The chain rule tells us that $\frac{\partial A}{\partial D} = \frac{\partial A}{\partial B} \times \frac{\partial B}{\partial C} \times \frac{\partial C}{\partial D}$

Take 1st Derivative of the Objective Function

$$L = \log L(\theta) = -\sum_{i=1}^{N}\left[y_i log\sigma\left(\boldsymbol{\theta}^{\boldsymbol{T}}\boldsymbol{X}\right) + (1-y_i)\log\left(1-\sigma\left(\boldsymbol{\theta}^{\boldsymbol{T}}\boldsymbol{X}\right)\right)\right]$$

Preliminaries

- The derivative of the logistic (sigmoid) function is $\frac{\partial\sigma(z)}{\partial z} = \sigma(z)[1-\sigma(z)]$
- The chain rule tells us that $\frac{\partial A}{\partial D} = \frac{\partial A}{\partial B} \times \frac{\partial B}{\partial C} \times \frac{\partial C}{\partial D}$

Also

- Derivative of sum = sum of derivatives → focus on 1 training input
- Compute $\frac{\partial L}{\partial \theta_j}$ for each θ_j individually, so focus on 1 θ_j

Take 1st Derivative of the Objective Function

$$L = \log L(\theta) = -\sum_{i=1}^{N}\left[y_i log\sigma\left(\boldsymbol{\theta^T X}\right) + (1-y_i)\log\left(1-\sigma\left(\boldsymbol{\theta^T X}\right)\right)\right]$$

Preliminaries

- The derivative of the logistic (sigmoid) function is $\frac{\partial\sigma(z)}{\partial z} = \sigma(z)[1-\sigma(z)]$
- The chain rule tells us that $\frac{\partial A}{\partial D} = \frac{\partial A}{\partial B}\times\frac{\partial B}{\partial C}\times\frac{\partial C}{\partial D}$

$$\frac{\partial \log L(\theta)}{\partial\theta_j} = \frac{\partial \log L(\theta)}{\partial P}\times\frac{\partial P}{\partial z}\times\frac{\partial z}{\partial\theta_j} \quad where\ P = \sigma\left(\boldsymbol{\theta^T X}\right)\ and\quad z = \boldsymbol{\theta^T X}$$

Take 1st Derivative of the Objective Function

$$L = \log L(\theta) = -\sum_{i=1}^{N}\left[y_i log\sigma\left(\boldsymbol{\theta^T X}\right) + (1-y_i)\log\left(1-\sigma\left(\boldsymbol{\theta^T X}\right)\right)\right]$$

Preliminaries

- The derivative of the logistic (sigmoid) function is $\frac{\partial\sigma(z)}{\partial z} = \sigma(z)[1-\sigma(z)]$
- The chain rule tells us that $\frac{\partial A}{\partial D} = \frac{\partial A}{\partial B}\times\frac{\partial B}{\partial C}\times\frac{\partial C}{\partial D}$

$$\frac{\partial \log L(\theta)}{\partial \theta_j} = \frac{\partial \log L(\theta)}{\partial P}\times\frac{\partial P}{\partial z}\times\frac{\partial z}{\partial \theta_j} \quad where\ P = \sigma\left(\boldsymbol{\theta^T X}\right)\ and\ \ z = \boldsymbol{\theta^T X}$$

$$\frac{\partial \log L(\theta)}{\partial P} = -\left(\frac{y}{P} - \frac{1-y}{1-P}\right)$$

because $\log L(\theta) = -[\,ylogP + (1-y)\log(1-P)]$

Take 1st Derivative of the Objective Function

$$\log L(\theta) = -\sum_{i=1}^{N} y_i log\sigma\left(\boldsymbol{\theta^T X_i}\right) + (1 - y_i)\log(1 - \sigma\left(\boldsymbol{\theta^T X_i}\right))$$

Preliminaries

- The derivative of the logistic (sigmoid) function is $\frac{\partial\sigma(z)}{\partial z} = \sigma(z)[1 - \sigma(z)]$
- The chain rule tells us that $\frac{\partial A}{\partial D} = \frac{\partial A}{\partial B} \times \frac{\partial B}{\partial C} \times \frac{\partial C}{\partial D}$

$$\frac{\partial \log L(\theta)}{\partial \theta_j} = \frac{\partial \log L(\theta)}{\partial P} \times \frac{\partial P}{\partial z} \times \frac{\partial z}{\partial \theta_j} \quad where\ P = \sigma\left(\boldsymbol{\theta^T X}\right)\ and \quad z = \boldsymbol{\theta^T X}$$

$$\frac{\partial P}{\partial z} = \frac{\partial \sigma(z)}{\partial z} = \sigma(z)(1 - \sigma(z))$$

Take 1st Derivative of the Objective Function

$$\log L(\theta) = -\sum_{i=1}^{N} y_i log\sigma\left(\boldsymbol{\theta^T X_i}\right) + (1 - y_i)\log(1 - \sigma\left(\boldsymbol{\theta^T X_i}\right))$$

Preliminaries

- The derivative of the logistic (sigmoid) function is $\frac{\partial\sigma(z)}{\partial z} = \sigma(z)[1 - \sigma(z)]$
- The chain rule tells us that $\frac{\partial A}{\partial D} = \frac{\partial A}{\partial B} \times \frac{\partial B}{\partial C} \times \frac{\partial C}{\partial D}$

$$\frac{\partial \log L(\theta)}{\partial \theta_j} = \frac{\partial \log L(\theta)}{\partial P} \times \frac{\partial P}{\partial z} \times \frac{\partial z}{\partial \theta_j} \quad where\ P = \sigma\left(\boldsymbol{\theta^T X}\right)\ and \quad z = \boldsymbol{\theta^T X}$$

$$\frac{\partial z}{\partial \theta_j} = \frac{\partial \boldsymbol{\theta^T X}}{\partial \theta_j} = x^j$$

Take 1st Derivative of the Objective Function

$$\log L(\theta) = -\sum_{i=1}^{N} y_i log\sigma\left(\boldsymbol{\theta^T X_i}\right) + (1 - y_i)\log(1 - \sigma\left(\boldsymbol{\theta^T X_i}\right))$$

Preliminaries

- The derivative of the logistic (sigmoid) function is $\frac{\partial\sigma(z)}{\partial z} = \sigma(z)[1 - \sigma(z)]$
- The chain rule tells us that $\frac{\partial A}{\partial D} = \frac{\partial A}{\partial B} \times \frac{\partial B}{\partial C} \times \frac{\partial C}{\partial D}$

$$\frac{\partial \log L(\theta)}{\partial \theta_j} = \frac{\partial \log L(\theta)}{\partial P} \times \frac{\partial P}{\partial z} \times \frac{\partial z}{\partial \theta_j} \quad where\ P = \sigma\left(\boldsymbol{\theta^T X}\right)\ and \quad z = \boldsymbol{\theta^T X}$$

$$= -\left(\frac{y}{P} - \frac{1 - y}{1 - P}\right) \times \sigma(z)\left(1 - \sigma(z)\right) \times x^j$$

Take 1st Derivative of the Objective Function

$$\log L(\theta) = -\sum_{i=1}^{N} y_i log\sigma(\boldsymbol{\theta^T X_i}) + (1 - y_i)\log(1 - \sigma(\boldsymbol{\theta^T X_i}))$$

Preliminaries

- The derivative of the logistic (sigmoid) function is $\frac{\partial\sigma(z)}{\partial z} = \sigma(z)[1 - \sigma(z)]$
- The chain rule tells us that $\frac{\partial A}{\partial D} = \frac{\partial A}{\partial B} \times \frac{\partial B}{\partial C} \times \frac{\partial C}{\partial D}$

$$\frac{\partial \log L(\theta)}{\partial \theta_j} = \frac{\partial \log L(\theta)}{\partial P} \times \frac{\partial P}{\partial z} \times \frac{\partial z}{\partial \theta_j} \quad where\ P = \sigma(\boldsymbol{\theta^T X})\ and$$

$$= -\left(\frac{y}{P} - \frac{1-y}{1-P}\right) \times \sigma(z)\big(1 - \sigma(z)\big) \times x^j$$

$$= \left[\sigma(\boldsymbol{\theta^T X}) - y\right] \times x^j$$

Logistic Regression: Parameter Estimation III

The derivative of the log likelihood wrt. a single parameter θ_j for **all** training examples

$$\frac{\partial \log L(\theta)}{\partial \theta_j} = \sum_{i=1}^{N} \left(\sigma\left(\theta^T x^i\right) - y^i\right) x_j^i$$

- Now, we would set derivatives to zero (**Step 3**) and solve for θ (**Step 4**)
- Unfortunately, that's not straightforward here (as for Naive Bayes)
- Instead, we will use iterative method: **Gradient Descent**

Logistic Regression: Parameter Estimation III

The derivative of the log likelihood wrt. a single parameter θ_j for **all** training examples

$$\frac{\partial \log L(\theta)}{\partial \theta_j} = \sum_{i=1}^{N} \left(\sigma\left(\theta^T x^i\right) - y^i\right) x_j^i$$

- Now, we would set derivatives to zero (**Step 3**) and solve for θ (**Step 4**)
- Unfortunately, that's not straightforward here (as for Naive Bayes)
- Instead, we will use an iterative method: **Gradient Descent**

$$\theta_j^{(new)} \leftarrow \theta_j^{(old)} - \eta \frac{\partial \log L(\theta)}{\partial \theta_j}$$

THE UNIVERSITY OF MELBOURNE

Multinomial Logistic Regression

- So far we looked at problems where either $y = 0$ or $y = 1$ (e.g., spam classification: $y \in \{\text{play, not play}\}$)

$$P(y = 1|x;\theta) = \sigma(\theta^T x) \qquad = \frac{exp(\theta^T x)}{1 + exp(\theta^T x)}$$

$$P(y = 0|x;\theta) = 1 - \sigma(\theta^T x) \qquad = 1 - \frac{exp(\theta^T x)}{1 + exp(\theta^T x)}$$

Multinomial Logistic Regression

- So far we looked at problems where either $y = 0$ or $y = 1$ (e.g., spam classification: $y \in \{\text{play, not play}\}$)

$$P(y = 1|x;\theta) = \sigma(\theta^T x) \qquad = \frac{exp(\theta^T x)}{1 + exp(\theta^T x)}$$

$$P(y = 0|x;\theta) = 1 - \sigma(\theta^T x) \qquad = 1 - \frac{exp(\theta^T x)}{1 + exp(\theta^T x)}$$

- But what if we have more than 2 classes, e.g., $y \in \{\text{positive, negative, neutral}\}$
- we predict the probability of each class c by passing the input representation through the **softmax** function, a generalization of the sigmoid

$$P(y = c|x;\theta) = \frac{\exp(\theta_c x)}{\sum_k \exp(\theta_k x)}$$

- we learn a parameter vector θ_c for each class c

Example! Multi-class with 1-hot features

$$P(y = c|x; \theta) = \frac{\exp(\theta_c x)}{\sum_k \exp(\theta_k x)}$$

(Small) Test Data set

Outlook	Temp	Humidity	Class
rainy	*cool*	*normal*	0 (don't play)
sunny	*cool*	*normal*	1 (maybe play)
sunny	*hot*	*high*	2 (play)

Feature Function

$$x_0 = 1\ (bias\ term)$$

$$x_1 = \begin{cases} 1\ \ if\ outlook = sunny \\ 2\ if\ outlook = overcast \\ 3\ if\ outlook = rainy \end{cases}$$

$$x_2 = \begin{cases} 1\ \ if\ temp = hot \\ 2\ \ if\ temp = mild \\ 3\ if\ temp = cool \end{cases}$$

$$x_3 = \begin{cases} 1\ if\ humidity = normal \\ 2\ if\ humidity = high \end{cases}$$

Example! Multi-class with 1-hot features

$$P(y = c|x;\theta) = \frac{\exp(\theta_c x)}{\sum_k \exp(\theta_k x)}$$

(Small) Test Data set

Outlook	Temp	Humidity	Class
rainy	*cool*	*normal*	0 (don't play)
sunny	*cool*	*normal*	1 (maybe play)
sunny	*hot*	*high*	2 (play)

Feature Function

$$x_0 = 1\ (bias\ term)$$

$$x_1 = \begin{cases} 1\ \ if\ outlook = sunny \\ 2\ if\ outlook = overcast \\ 3\ if\ outlook = rainy \end{cases}$$

$$x_2 = \begin{cases} 1\ \ if\ temp = hot \\ 2\ \ if\ temp = mild \\ 3\ if\ temp = cool \end{cases}$$

$$x_3 = \begin{cases} 1\ if\ humidity = normal \\ 2\ if\ humidity = high \end{cases}$$

Model parameters

θ_{c0}

[0.1, 0.7, 0.2, −3.5, −3.5, −3.5, 0.7, 2.1]

θ_{c1}

[0.6, 0.1, 0.9, 2.5, 2.5, 2.5, 2.7, −2.1]

θ_{c2}

[3.1, 3.4, 4.1, 1.5, 1.5, 1.5, 0.7, 3.6]

THE UNIVERSITY OF MELBOURNE

Logistic Regression: Final Thoughts

Pros

- Probabilistic interpretation
- No restrictive assumptions on features
- Often outperforms Naive Bayes
- Particularly suited to frequency-based features (so, popular in NLP)

Cons

- Can only learn *linear* feature-data relationships
- Some feature scaling issues
- Often needs a lot of data to work well
- Regularisation a nuisance, but important since overfitting can be a big problem

THE UNIVERSITY OF MELBOURNE

Summary

- Derivation of logistic regression
- Prediction
- Derivation of maximum likelihood

References

Cosma Shalizi. *Advanced Data Analysis from an Elementary Point of View*. Chapters 11.1 and 11.2. Online Draft.
https://www.stat.cmu.edu/~cshalizi/ADAfaEPoV/ADAfaEPoV.pdf

Dan Jurafsky and James H. Martin. *Speech and Language Processing*. Chapter 5. Online Draft V3.0.
https://web.stanford.edu/~jurafsky/slp3/

THE UNIVERSITY OF MELBOURNE

Optional: Detailed Parameter Estimation

Step 2 Differentiate the loglikelihood wrt. the parameters

$$\log L(\theta) = -\sum_{i=1}^{N} y_i log\sigma\left(\boldsymbol{\theta^T X_i}\right) + (1 - y_i)\log(1 - \sigma\left(\boldsymbol{\theta^T X_i}\right))$$

Preliminaries

- The derivative of the logistic (sigmoid) function is $\frac{\partial\sigma(z)}{\partial z} = \sigma(z)[1 - \sigma(z)]$
- The chain rule tells us that $\frac{\partial A}{\partial D} = \frac{\partial A}{\partial B} \times \frac{\partial B}{\partial C} \times \frac{\partial C}{\partial D}$

THE UNIVERSITY OF MELBOURNE

Optional: Detailed Parameter Estimation

$$\log L(\theta) = -\sum_{i=1}^{N} y_i log\sigma\left(\boldsymbol{\theta^T X_i}\right) + (1 - y_i)\log(1 - \sigma\left(\boldsymbol{\theta^T X_i}\right))$$

Preliminaries

- The derivative of the logistic (sigmoid) function is $\frac{\partial\sigma(z)}{\partial z} = \sigma(z)[1 - \sigma(z)]$
- The chain rule tells us that $\frac{\partial A}{\partial D} = \frac{\partial A}{\partial B} \times \frac{\partial B}{\partial C} \times \frac{\partial C}{\partial D}$

Also

- Derivative of sum = sum of derivatives → focus on 1 training input
- Compute $\frac{\partial L}{\partial \theta_j}$ for each θ_j individually, so focus on 1 θ_j

Optional: Detailed Parameter Estimation

$$\log L(\theta) = -\sum_{i=1}^{N} y_i log\sigma\left(\boldsymbol{\theta^T X_i}\right) + (1 - y_i)\log(1 - \sigma\left(\boldsymbol{\theta^T X_i}\right))$$

Preliminaries

- The derivative of the logistic (sigmoid) function is $\frac{\partial\sigma(z)}{\partial z} = \sigma(z)[1 - \sigma(z)]$
- The chain rule tells us that $\frac{\partial A}{\partial D} = \frac{\partial A}{\partial B} \times \frac{\partial B}{\partial C} \times \frac{\partial C}{\partial D}$

$$\frac{\partial \log L(\theta)}{\partial \theta_j} = \frac{\partial \log L(\theta)}{\partial P} \times \frac{\partial P}{\partial z} \times \frac{\partial z}{\partial \theta_j} \quad where\ P = \sigma\left(\boldsymbol{\theta^T X}\right)\ and$$

Optional: Detailed Parameter Estimation

$$L = \log L(\theta) = -\sum_{i=1}^{N}\left[y_i log\sigma\left(\boldsymbol{\theta^T X}\right) + (1-y_i)\log\left(1-\sigma\left(\boldsymbol{\theta^T X}\right)\right)\right]$$

Preliminaries

- The derivative of the logistic (sigmoid) function is $\frac{\partial\sigma(z)}{\partial z} = \sigma(z)[1-\sigma(z)]$
- The chain rule tells us that $\frac{\partial A}{\partial D} = \frac{\partial A}{\partial B}\times\frac{\partial B}{\partial C}\times\frac{\partial C}{\partial D}$

$$\frac{\partial \log L(\theta)}{\partial\theta_j} = \frac{\partial \log L(\theta)}{\partial P}\times\frac{\partial P}{\partial z}\times\frac{\partial z}{\partial\theta_j} \quad where\ P = \sigma\left(\boldsymbol{\theta^T X}\right)\ and\ \ z = \boldsymbol{\theta^T X}$$

$$\frac{\partial \log L(\theta)}{\partial P} = -\frac{y}{P} - \frac{1-y}{1-P}$$

because $\log L(\theta) = -[\,ylogP + (1-y)\log(1-P)]$

Optional: Detailed Parameter Estimation

$$\log L(\theta) = -\sum_{i=1}^{N} y_i log\sigma\left(\boldsymbol{\theta^T X_i}\right) + (1 - y_i)\log(1 - \sigma\left(\boldsymbol{\theta^T X_i}\right))$$

Preliminaries

- The derivative of the logistic (sigmoid) function is $\frac{\partial\sigma(z)}{\partial z} = \sigma(z)[1 - \sigma(z)]$
- The chain rule tells us that $\frac{\partial A}{\partial D} = \frac{\partial A}{\partial B} \times \frac{\partial B}{\partial C} \times \frac{\partial C}{\partial D}$

$$\frac{\partial \log L(\theta)}{\partial \theta_j} = \frac{\partial \log L(\theta)}{\partial P} \times \frac{\partial P}{\partial z} \times \frac{\partial z}{\partial \theta_j} \quad where\ P = \sigma\left(\boldsymbol{\theta^T X}\right)\ and \quad z = \boldsymbol{\theta^T X}$$

$$\frac{\partial P}{\partial z} = \frac{\partial \sigma(z)}{\partial z} = \sigma(z)(1 - \sigma(z))$$

Optional: Detailed Parameter Estimation

$$\log L(\theta) = -\sum_{i=1}^{N} y_i log\sigma\left(\boldsymbol{\theta^T X_i}\right) + (1 - y_i)\log(1 - \sigma\left(\boldsymbol{\theta^T X_i}\right))$$

Preliminaries

- The derivative of the logistic (sigmoid) function is $\frac{\partial\sigma(z)}{\partial z} = \sigma(z)[1 - \sigma(z)]$
- The chain rule tells us that $\frac{\partial A}{\partial D} = \frac{\partial A}{\partial B} \times \frac{\partial B}{\partial C} \times \frac{\partial C}{\partial D}$

$$\frac{\partial \log L(\theta)}{\partial \theta_j} = \frac{\partial \log L(\theta)}{\partial P} \times \frac{\partial P}{\partial z} \times \frac{\partial z}{\partial \theta_j} \quad where\ P = \sigma\left(\boldsymbol{\theta^T X}\right)\ and \quad z = \boldsymbol{\theta^T X}$$

$$\frac{\partial z}{\partial \theta_j} = \frac{\partial \boldsymbol{\theta^T X}}{\partial \theta_j} = x^j$$

Optional: Detailed Parameter Estimation

$$\log L(\theta) = -\sum_{i=1}^{N} y_i log\sigma\left(\boldsymbol{\theta^T X_i}\right) + (1 - y_i)\log(1 - \sigma\left(\boldsymbol{\theta^T X_i}\right))$$

Preliminaries

- The derivative of the logistic (sigmoid) function is $\frac{\partial \sigma(z)}{\partial z} = \sigma(z)[1 - \sigma(z)]$
- The chain rule tells us that $\frac{\partial A}{\partial D} = \frac{\partial A}{\partial B} \times \frac{\partial B}{\partial C} \times \frac{\partial C}{\partial D}$

$$\frac{\partial \log L(\theta)}{\partial \theta_j} = \frac{\partial \log L(\theta)}{\partial P} \times \frac{\partial P}{\partial z} \times \frac{\partial z}{\partial \theta_j} \quad where\ P = \sigma\left(\boldsymbol{\theta^T X}\right)\ and \quad z = \boldsymbol{\theta^T X}$$

$$= -\left(\frac{y}{P} - \frac{1 - y}{1 - P}\right) \times \sigma(z)\left(1 - \sigma(z)\right) \times x^j$$

Optional: Detailed Parameter Estimation

Step 2 Differentiate the loglikelihood wrt. the parameters

$$\log L(\theta) = -\sum_{i=1}^{N} y_i log\sigma\left(\boldsymbol{\theta}^{\boldsymbol{T}}\boldsymbol{X}_{\boldsymbol{i}}\right) + (1 - y_i)\log(1 - \sigma\left(\boldsymbol{\theta}^{\boldsymbol{T}}\boldsymbol{X}_{\boldsymbol{i}}\right))$$

Preliminaries

- The derivative of the logistic (sigmoid) function is $\frac{\partial\sigma(z)}{\partial z} = \sigma(z)[1 - \sigma(z)]$
- The chain rule tells us that $\frac{\partial A}{\partial D} = \frac{\partial A}{\partial B} \times \frac{\partial B}{\partial C} \times \frac{\partial C}{\partial D}$

$$\frac{\partial \log L(\theta)}{\partial \theta_j} = \frac{\partial \log L(\theta)}{\partial P} \times \frac{\partial P}{\partial z} \times \frac{\partial z}{\partial \theta_j}$$

where $p = \sigma(\theta^T x)$ and $z = \theta^T x$

$$= -\left(\frac{y}{P} - \frac{1 - y}{1 - P}\right) \times \sigma(z)\left(1 - \sigma(z)\right) \times x^j$$

[[combine 3 derivatives]]

$$= -\left(\frac{y(1 - P)}{P(1 - P)} - \frac{P(1 - y)}{P(1 - P)}\right) \times P(1 - P) \times x^j$$

[[$\sigma(z) = p$]]

$$= -\left(\frac{y(1 - P)}{P(1 - P)} - \frac{P(1 - y)}{P(1 - P)}\right) \times P(1 - P) \times x^j$$

[[$\times \frac{P(1 - P)}{P(1 - P)}$]]

$$= -(y(1 - P) - P(1 - y)) \times x^j$$

[[cancel terms]]

THE UNIVERSITY OF MELBOURNE

Optional: Detailed Parameter Estimation

Step 2 Differentiate the loglikelihood wrt. the parameters

$$\log L(\theta) = -\sum_{i=1}^{N} y_i log\sigma\left(\boldsymbol{\theta}^{\boldsymbol{T}}\boldsymbol{X}_{\boldsymbol{i}}\right) + (1 - y_i)\log(1 - \sigma\left(\boldsymbol{\theta}^{\boldsymbol{T}}\boldsymbol{X}_{\boldsymbol{i}}\right))$$

Preliminaries

- The derivative of the logistic (sigmoid) function is $\frac{\partial\sigma(z)}{\partial z} = \sigma(z)[1 - \sigma(z)]$
- The chain rule tells us that $\frac{\partial A}{\partial D} = \frac{\partial A}{\partial B} \times \frac{\partial B}{\partial C} \times \frac{\partial C}{\partial D}$

$$\frac{\partial \log L(\theta)}{\partial \theta_j} = \frac{\partial \log L(\theta)}{\partial P} \times \frac{\partial P}{\partial z} \times \frac{\partial z}{\partial \theta_j}$$ where $p = \sigma(\theta^T x)$ and $z = \theta^T x$

$= -\left(y(1 - P) - P(1 - y)\right) \times x^j$ [[copy from last slide]]

$= -(y - yP\ - P + \ yP) \times x^j$ [[multiply out]]

$= -(y - P) \times x^j$ $[[-yP + yP = 0]]$

$= (P\ -\ y) \times x^j$ $[[-(y\ -\ P) = -y + P = P - y]]$

$= \left(\sigma\left(\boldsymbol{\theta}^{\boldsymbol{T}}\boldsymbol{X}\right) -\ y\right) \times x^j$ $[[P = \sigma(z), z = \theta^T x]]$