Data Science

Introduction to Machine Learning:
Gradient Descent

March 30, 2022

Machine Learning

Many Machine Learning problems take the following form:

$$\text{minimize}_\theta \sum_{i=1}^{m} l(h_\theta(x^{(i)}), y^{(i)})$$

Machine Learning

Many Machine Learning problems take the following form:

$$\text{minimize}_\theta \sum_{i=1}^{m} l(h_\theta(x^{(i)}), y^{(i)})$$

Let's go through this, bit by bit

Machine Learning

We have some input data we'd like to learn from:

$$\text{minimize}_\theta \sum_{i=1}^{m} l(h_\theta(x^{(i)}), y^{(i)})$$

Machine Learning

We have some known output data:

$$\text{minimize}_\theta \sum_{i=1}^{m} l(h_\theta(x^{(i)}), y^{(i)})$$

Machine Learning

We have a *hypothesis function*, with unkown parameters:

$$\text{minimize}_\theta \sum_{i=1}^{m} l(h_\theta(x^{(i)}), y^{(i)})$$

Machine Learning

We have a *loss* function that tells us how wrong we are:

$$\text{minimize}_\theta \sum_{i=1}^{m} l(h_\theta(x^{(i)}), y^{(i)})$$

Machine Learning

We want to sum the 'loss' from all of our input/output pairs:

$$\text{minimize}_\theta \sum_{i=1}^{m} l(h_\theta(x^{(i)}), y^{(i)})$$

Machine Learning

We want to minimize the 'loss' by changing the parameters to our hypothesis function:

$$\text{minimize}_\theta \sum_{i=1}^{m} l(h_\theta(x^{(i)}), y^{(i)})$$

One Approach

Gradient Descent!

One Approach

Gradient Descent!

1. The term gradient comes from calculus (a vector of partial derivatives)

One Approach

Gradient Descent!

1. The term gradient comes from calculus (a vector of partial derivatives)
2. We can 'ride' the gradient to some minimum (or maximum)

One Approach

Gradient Descent is search! The basic algorith:

One Approach

Gradient Descent is search! The basic algorith:

1. pick a starting point

One Approach

Gradient Descent is search! The basic algorith:

1. pick a starting point
2. compute the sum of the loss over learning set

One Approach

Gradient Descent is search! The basic algorith:

1. pick a starting point
2. compute the sum of the loss over learning set
3. compute the sum of the loss for points ‘nearby’

One Approach

Gradient Descent is search! The basic algorith:

1. pick a starting point
2. compute the sum of the loss over learning set
3. compute the sum of the loss for points 'nearby'
4. pick new parameters based on the gradient from the previous steps

One Approach

Gradient Descent is search! The basic algorith:

1. pick a starting point
2. compute the sum of the loss over learning set
3. compute the sum of the loss for points ‘nearby’
4. pick new parameters based on the gradient from the previous steps
5. repeat

One Approach

Gradient Descent is search! The basic algorith:

1. pick a starting point
2. compute the sum of the loss over learning set
3. compute the sum of the loss for points 'nearby'
4. pick new parameters based on the gradient from the previous steps
5. repeat
6. When do we stop?

One Approach

Gradient Descent is search! The basic algorith:

1. pick a starting point
2. compute the sum of the loss over learning set
3. compute the sum of the loss for points ‘nearby’
4. pick new parameters based on the gradient from the previous steps
5. repeat
6. When do we stop?
7. What assumptions have we baked in?

Gradient Descent

Assumptions

Gradient Descent

Assumptions

1. That the loss function has a gadient!

Gradient Descent

Assumptions

1. That the loss function has a gadient!
2. That there's only one minimum (maximum)

Gradient Descent

Assumptions

1. That the loss function has a gadient!
2. That there's only one minimum (maximum)
3. What can we do about this?

Loss Functions

What we want:

Loss Functions

What we want:

1. Continuity

Loss Functions

What we want:

1. Continuity
2. Global minimum

Loss Functions

What we want:

1. Continuity
2. Global minimum
3. Cheap

Loss Functions

What we want:

1. Continuity
2. Global minimum
3. Cheap
4. Convex (why?)

Loss Functions

What we want:

1. Continuity
2. Global minimum
3. Cheap
4. Convex (why?)
5. A function is convex if a lin between two points always lies above the function.

Loss Functions

What we have:

Loss Functions

What we have:

1. Almost none of these things.

Loss Functions

What we have:

1. Almost none of these things.
2. Most functions don't have these nice properties
3. Instead we *approximate* the loss function

Surrogate Loss Functions

Let's just make a function with the properties we care about!
Some alternatives:

Surrogate Loss Functions

Let's just make a function with the properties we care about!
Some alternatives:

1. 0/1 Loss

Surrogate Loss Functions

Let's just make a function with the properties we care about!
Some alternatives:

1. 0/1 Loss
2. Hinge

Surrogate Loss Functions

Let's just make a function with the properties we care about!
Some alternatives:

1. 0/1 Loss
2. Hinge
3. Exponential

Surrogate Loss Functions

Let's just make a function with the properties we care about!
Some alternatives:

1. 0/1 Loss
2. Hinge
3. Exponential
4. Squared Loss (very common)

On Wednesday we will:

On Wednesday we will:

1. Show examples of each loss function

On Wednesday we will:

1. Show examples of each loss function
2. Use Gradient descent to learn a linear model

On Wednesday we will:

1. Show examples of each loss function
2. Use Gradient descent to learn a linear model
3. Use our hypothesis testing to see if it's any good!

Thanks for your time!

:)