A Intuition of Machine Learning

Namely linear regression and perceptron

Which I really don't have a clue about

But have to present anyway...

Richard Zhong

SHUOSC

2017 / 10 / 13

SHUOSC

Machine Learning——What? How? When?

- What is Machine Learning?
  - Arthur Samuel:
    " Field of study that gives computers the ability to learn without being explicitly programmed. "

- How to achive Machine Learning?
  - With data
  - With model

- When to use Machine Learning?
  - With a black-box system or related data
  - Facing a problem hard to address by explicit programming

Types of Machine Learning

- Supervised Learning
  - Knowing a set of history input $X = \{x_1, x_2, \ldots\}$ and corresponding output $Y = \{y_1, y_2, \ldots\}$
  - Given a new input x', try to predict y'
- Unsupervised Learning
  - Knowing a lot of input $X = \{x_1, x_2, \ldots\}$
  - Find the structure within them.
- Reinforce Learning and Others
  - ......

Toy Problems

Problem 1

- $X = \{-3, -2, 9, 22\}$
  $Y = \{-2, -1, 10, 23\}$
- Given $x' = 5$, $y' = ?$

Problem 2

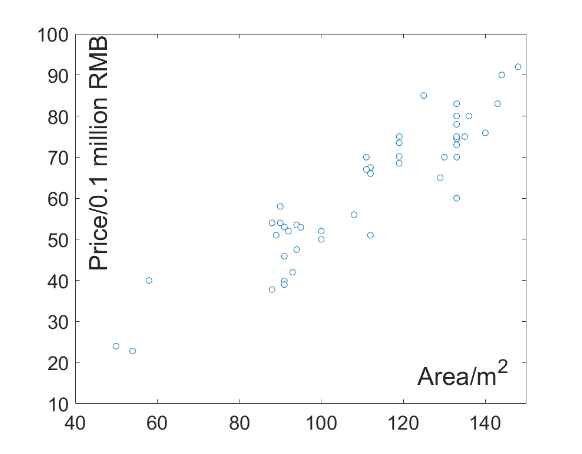
- $X = \{\sqrt{2}, \pi, 2.73, \frac{9}{8}\}$
  $Y = \{1, 1, 0, 0\}$
- Given $x' = 7.26$, $y' = ?$

Types of Supervised Learning

- Regression
  - y is a real(continuous) value
  - Given x, want to know exact how much is y
- Classification
  - y is a discrete value
  - Given x, want to know of all possible results, which would y be

House Price Predicting

Area-Price Dataset



Model Formula

- Observation
  - Somehow all the scatters fall into a region near a line
  - Let's assume this line is $y = wx + b$
  - Where w, b is a set of value we don't know yet, or so-called *parameters*
- Next thing to do
  - Get all the x-y pairs in dataset
  - Change w, b accordingly, to find the best line that is able to describe the data distribution

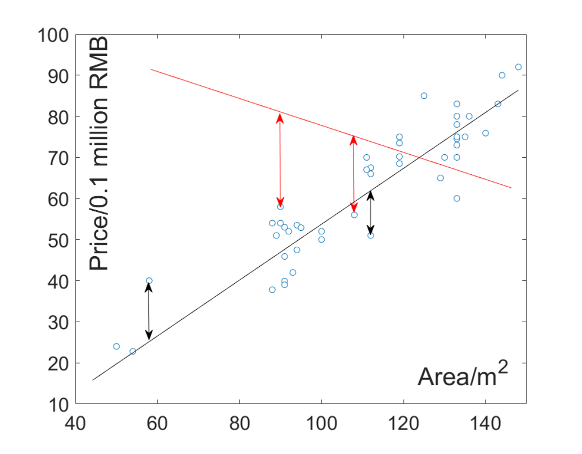
Mathematics is the gate and key of the sciences. —— Roger Bacon

But how do we choose parameters w and b anyway?
Well, anything mathematical is good.

- Find a metric of the performance of your model, given w_k, b_k
- Minimize/Maximize the metric in an analytic or arithmetic way
- Choose the set of parameters that optimizes the metric

Loss Function

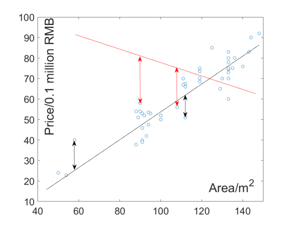
Intuition of Mean Square Error(distance)



Formula of MSE

$$J(w,b) = \frac{1}{2m}\left[(wx_1 + b - y_1)^2 + (wx_2 + b - y_2)^2 + (wx_3 + b - y_3)^2 + \ldots + (wx_m + b - y_m)^2\right]$$

$$= \frac{1}{2m}\sum_{i=1}^{m}(wx_i + b - y_i)^2$$



Partial Derivative of $J(w, b)$

$$\frac{\partial J}{\partial w} = \frac{1}{m}\sum_{i=1}^{m} x_i \left(wx_i + b - y_i\right)$$

$$= \frac{1}{m}[\sum_{i=1}^{m}\left(x_i^2\right) w + \sum_{i=1}^{m}(x_i b) - \sum_{i=1}^{m}\left(x_i y_i\right)]$$

$$\frac{\partial J}{\partial b} = \frac{1}{m}\sum_{i=1}^{m}\left(wx_i + b - y_i\right)$$

$$= \frac{1}{m}[\left(\sum_{i=1}^{m} x_i\right) w_i + mb - \sum_{i=1}^{m} y_i]$$

Optimization

Partial Derivative of $J(w, b)$

$$Define \quad S_x = \sum_{i=1}^{m} x_i \quad | \quad S_{xy} = \sum_{i=1}^{m} (x_i y_i)$$

$$\frac{\partial J}{\partial w} = \frac{1}{m}(S_{x^2} w + bS_x - S_{xy})$$

$$\frac{\partial J}{\partial b} = \frac{1}{m}(mb + S_x w - S_y)$$

Optimization

Best set of w, b

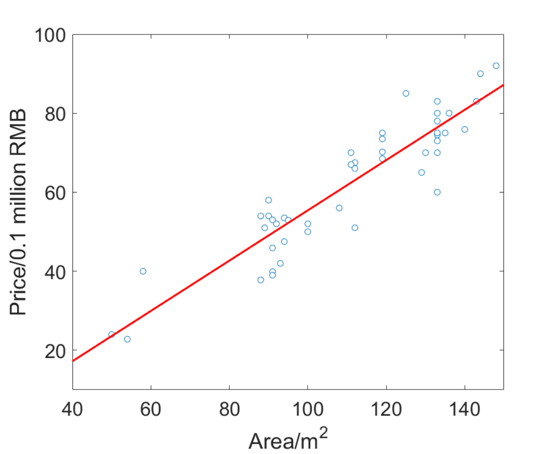
$$Let \quad \frac{\partial J}{\partial w} = \frac{\partial J}{\partial b} = 0$$

$$w = \frac{mS_{xy} - S_x S_y}{mS_{x^2} - (S_x)^2} = 0.6363$$

$$b = \frac{S_{x^2} S_y - S_x S_{xy}}{mS_{x^2} - (S_x)^2} = -8.217$$

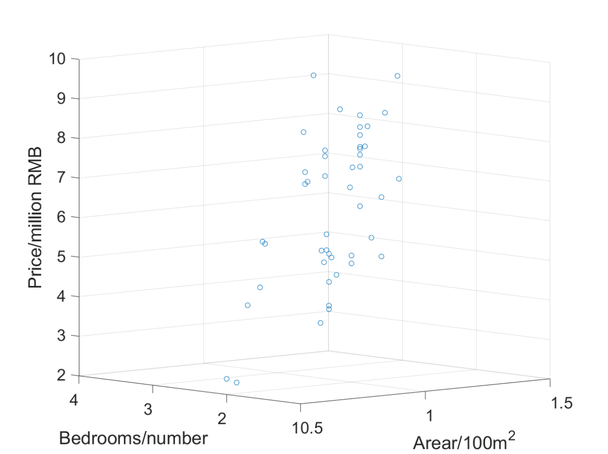
Analytic Fitting Result

Given $x' = 118$, $y' = 66.8664$



More Data

Area, Bedroom-Price Dataset



Model Formula

- Observation
  - Somehow all the scatters fall into a region near a plain
  - We can assume this plain is $y = w_1 x_1 + w_2 x_2 + b$
  - ...and pretend that this is not mathematically annoying

- Next thing to do
  - Get all the $x_1, x_2 - y$ pairs in dataset
  - Change w_1, w_2, b accordingly, to find the best plain that is able to describe the data distribution

Optimization

$J(w, b)$ and its Partial Derivative

$$J(w_1, w_2, b) = \frac{1}{2m} \sum_{i=1}^{m} (w_1 x_{i,1} + w_2 x_{i,2} + b)^2$$

$$\frac{\partial J}{\partial w_1} = \frac{1}{m} \sum_{i=1}^{m} x_{i,1} (w_1 x_{i,1} + w_2 x_{i,2} + b)$$

$$\frac{\partial J}{\partial w_2} = \frac{1}{m} \sum_{i=1}^{m} x_{i,2} (w_1 x_{i,1} + w_2 x_{i,2} + b)$$

$$\frac{\partial J}{\partial b} = \frac{1}{m} \sum_{i=1}^{m} (w_1 x_{i,1} + w_2 x_{i,2} + b)$$

Optimization

Best set of w, b

$$Let \quad \frac{\partial J}{\partial w_1} = \frac{\partial J}{\partial w_2} = \frac{\partial J}{\partial b} = 0$$

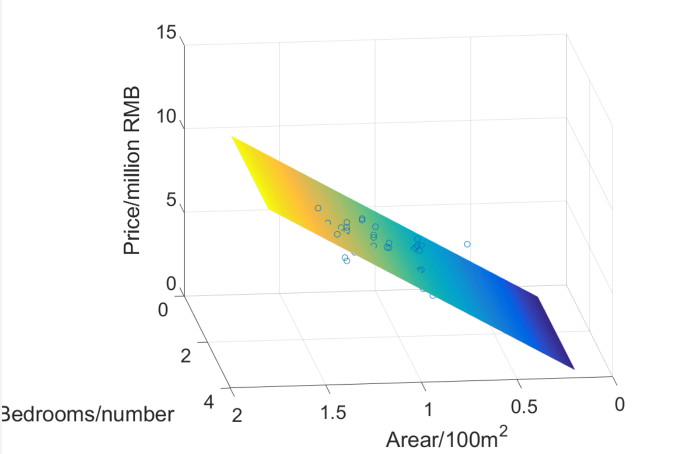
$$w_1 = 6.327$$

$$w_2 = 0.01874$$

$$b = -0.8298$$

Analytic Fitting Result

Given $x_1' = 1.18, x_2 = 4, y' = 6.7110$



Generalization

Given a set of input $X = \{x_1, x_2, x_3, \dots, x_m\}$,
Where $x_k = \{x_{k,1}, x_{k,2}, \dots, x_{k,n}\}$
And a set of output $Y = \{y_1, y_2, y_3, \dots, y_m\}$

The linear model predicts
$y(x_k) = w_0 + w_1 x_{k,1} + w_2 x_{k,2} + \dots + w_n x_{k,n}$

The loss function is $J(w_0, w_1, w_2, \dots, w_n) = \sum_{k=1}^{m} \left(y\,(x_k) - y_k\right)^2$
Optimization is to find $(w_0, w_1, \dots, w_n)$ that minimize J.

Vectorization

> The fundamental problem of linear algebra is to solve a system of linear equations.
>
> —— Gilbert Strang

$$
\begin{pmatrix}
1 & x_{1,1} & x_{1,2} & \cdots & x_{1,n} \\
1 & x_{2,1} & x_{2,2} & \cdots & x_{2,n} \\
\vdots & & & & \\
1 & x_{k,1} & x_{k,2} & \cdots & x_{k,n} \\
\vdots & & & & \\
1 & x_{m,1} & x_{m,2} & \cdots & x_{m,n}
\end{pmatrix}
\cdot
\begin{pmatrix}
w_0 \\ w_1 \\ w_2 \\ \vdots \\ w_n
\end{pmatrix}
=
\begin{pmatrix}
y(x_1) \\ y(x_2) \\ \vdots \\ y(x_k) \\ \vdots \\ y(x_m)
\end{pmatrix}
$$

Vectorization

The fundamental problem of linear algebra is to solve a system of linear equations.

—— Gilbert Strang

$$X \cdot W = Y(X)$$

$$\left(\begin{pmatrix} y(x_1) \\ y(x_2) \\ \vdots \\ y(x_k) \\ \vdots \\ y(x_m) \end{pmatrix} - \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_k \\ \vdots \\ y_m \end{pmatrix} \right)^{\odot 2} = \begin{pmatrix} (y(x_1) - y_1)^2 \\ (y(x_2) - y_2)^2 \\ \vdots \\ (y(x_k) - y_k)^2 \\ \vdots \\ (y(x_m) - y_m)^2 \end{pmatrix}$$

Normal Equation

> The fundamental problem of linear algebra is to solve a system of linear equations.
>
> —— Gilbert Strang

$$X \cdot W = Y(X)$$

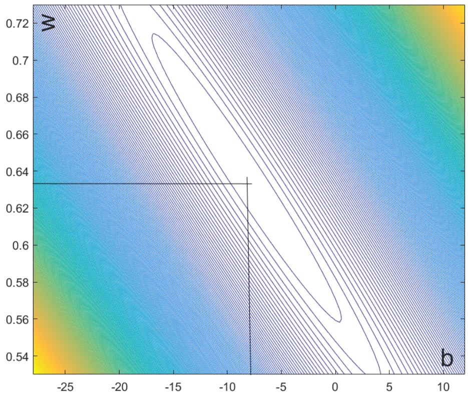
$$J(W) = sum((Y(x) - Y)\odot^2)$$

$$\nabla_W J(W) = X^T X W - X^T Y$$

$$min(J(W)) \Leftrightarrow W = (X^T X)^{-1} X^T Y$$

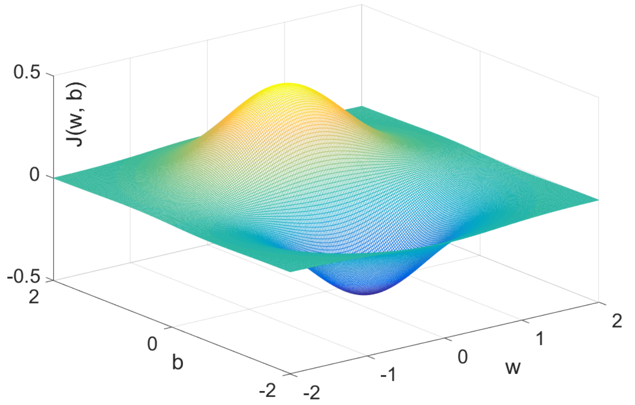
Intuition of Gradient Descent

Model: $y(x) = wx + b$, Loss: $J(w, b)$



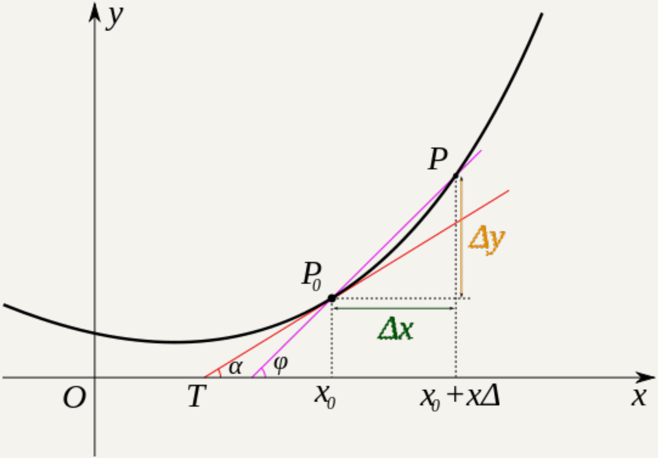
Gradient Descent

Intuition of Gradient Descent



Meaning of Derivative

$$x+ = \Delta x \frac{dy}{dx} \Leftrightarrow y+ = \Delta y$$



Gradient Descent

Formula

Randomly choose w, b

While $(\frac{\partial J}{\partial w}! = 0 \quad and \quad \frac{\partial J}{\partial b}! = 0)$ {

$$w- = \Delta w \frac{\partial J}{\partial w}$$

$$\Leftrightarrow \quad J(w, b)- = \Delta J$$

$$b- = \Delta b \frac{\partial J}{\partial b}$$

}

Let's see how this is done by wathcing a matlab program.

Gradient Descent Formula

While $(J(W) \quad is \quad decreasing)$ {

$$
\begin{gathered}
w_0 -= \alpha \frac{\partial J}{\partial w_0} \\
w_1 -= \alpha \frac{\partial J}{\partial w_1} \\
\vdots \\
w_m -= \alpha \frac{\partial J}{\partial w_m}
\end{gathered}
$$

}

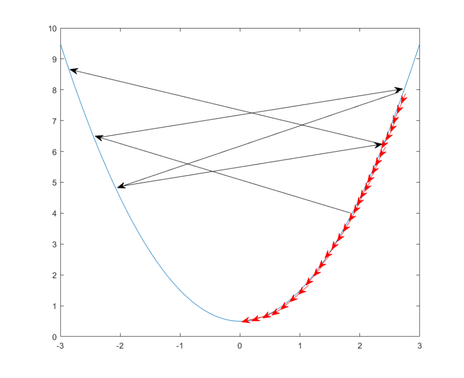
Where α is called the *Learning Rate*

Learning Rate

Bad Learning Rate

When α is too large, J fail to converge,

while when it's to small, J converge too slowly.



Learning Rate

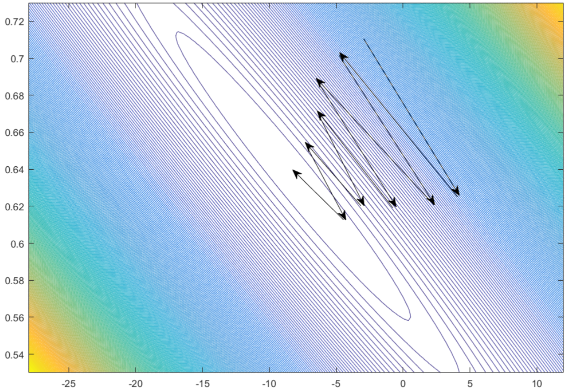
> When I do that usually it just gives a good learning rate for my problem ...
>
> —— Andrew Ng

Choose Learning Rate

- Try a value of α that is definitely too small, say 0.0001
- Try a value of α that is definitely too large, say 1
- Try to increase your α with a 3-fold step, say { 0.0003, 0.001, 0.003, 0.01, 0.03, 0.01, 0.03, 0.1 ...}

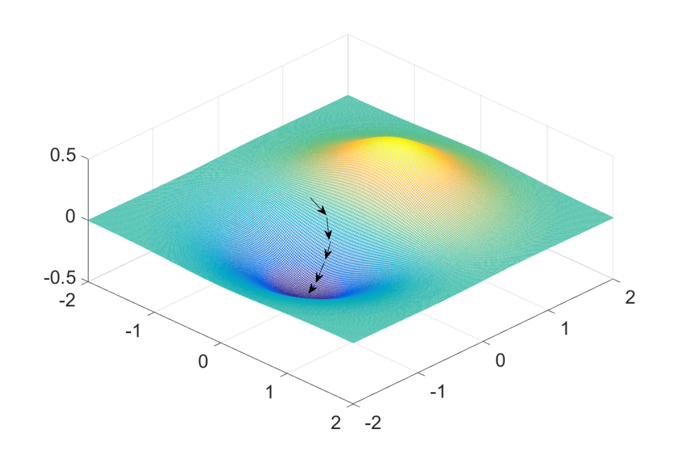
Overshoot of parameters

When $\dfrac{\partial J}{\partial w}$ and $\dfrac{\partial J}{\partial b}$ is too different ...



Gradient Descent

Ideal Scenario



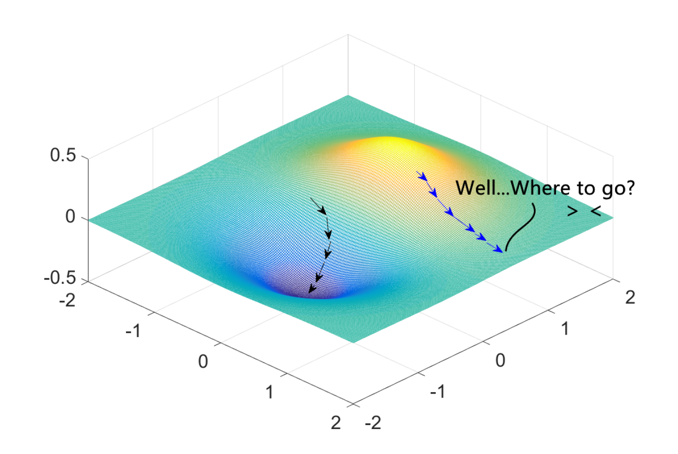
Comparison

Gradient Descent And Normal Equation

- Gradient Descent
  - Only need to know $\frac{\partial J}{\partial w}$(more general)
  - Need to choose α and do data scaling
  - Accuracy depends on α
  - Compatible with big data
- Normal Equation
  - Need to do a lot analytic works
  - No need to choose α and do data scaling
  - Perfectly accurate
  - Slow with large dataset($O(n^3)$ to compute $(X^T X)^{-1}$)

Gradient Descent

Local Optimal



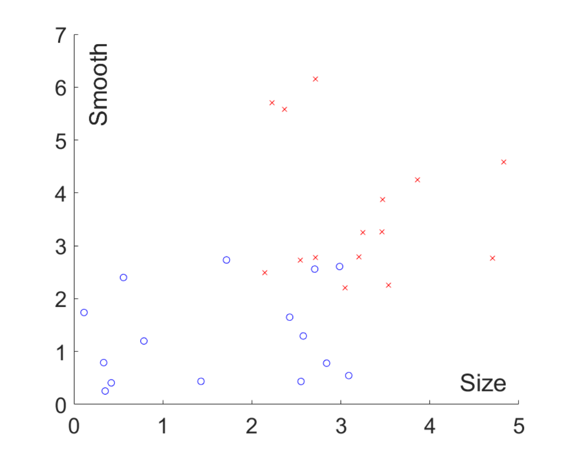
Let's observe it by watching a matlab program.

Paradox of Gradient Descent

Welcome to the world of Machine Learning

- Gradient Descent is likely to converge at local optimal(or saddle point) unless monotonicity is proven
- Proving a loss function has a global optimal is almost as difficult as to just find it
- It turns out for most problems analytic solution is impossible
- But people use gradient descent anyway...

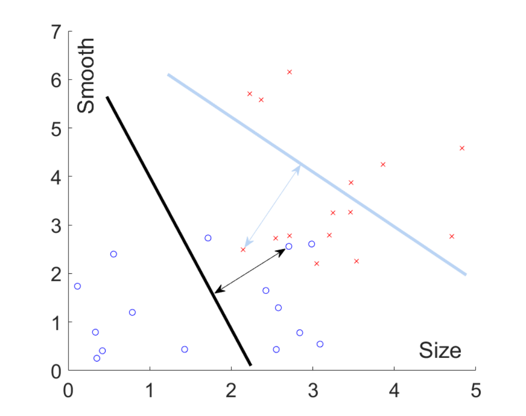
Tumor Dataset



Tumor Diagnosis

Design Perceptron Loss Function

J is small when: 1.less error points, 2.split line gets closer to error points.



Loss Function

Formula

- Split line defined by $w_1 x_1 + w_2 y_2 + b = 0$
- Distance from point $(x_{i,1}, x_{i,2})$ to this line is $\frac{|w_1 x_{i,1} + w_2 x_{i,2} + b|}{\sqrt{w_1^2 + w_2^2}}$
- For M is the error points set

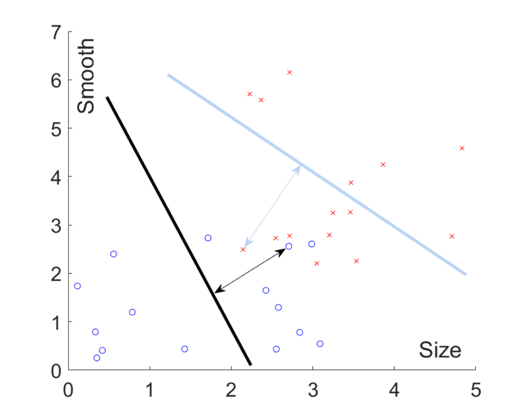
$$J(w_1, w_2, b) = \sum_{x_i \in M} \frac{|w_1 x_{i,1} + w_2 x_{i,2} + b|}{\sqrt{w_1^2 + w_2^2}}$$

Simplify Loss Function

Remove Absolute Mark

Predict 1 if $w_1 x_{i,1} + w_2 x_{i,2} + b > 0$, for $x_i \in M$

$$|w_1 x_{i,1} + w_2 x_{i,2} + b| \Leftrightarrow -y_i(w_1 x_{i,1} + w_2 x_{i,2} + b)$$



Simplify Loss Function

Remove 2th Norm

- $\sqrt{w_1^2 + w_2^2}$ can be just consider as a part of the leaning rate
- α of a perceptron will be noisy at the begining but finnaly converge
- ...which gives us a simplified loss function as:

$$J(w_1, w_2, b) = -\sum_{x_i \in M} y_i(w_1 x_{i,1} + w_2 x_{i,2} + b)$$

Simplified Loss Function

Formula

$$J(w_1, w_2, b) = -\sum_{x_i \in M} y_i(w_1 x_{i,1} + w_2 x_{i,2} + b)$$

$$\frac{\partial J}{\partial w_1} = -\sum_{x_i \in M} y_i x_{i,1}$$

$$\frac{\partial J}{\partial w_2} = -\sum_{x_i \in M} y_i x_{i,2}$$

$$\frac{\partial J}{\partial b} = -\sum_{x_i \in M} y_i$$

Optimization

Formula

Randomly choose w_1, w_2, b

While $(J(W) \quad is \quad decreasing)$ {

$$\left.\begin{aligned} w_1 \quad &-= \alpha \frac{\partial J}{\partial w_1} \\ w_2 \quad &-= \alpha \frac{\partial J}{\partial w_2} \\ b \quad &-= \alpha \frac{\partial J}{\partial b} \end{aligned}\right. \Leftrightarrow J(w_1, w_2, b) -= \Delta J$$

}

Let's see how this is done by wathcing a matlab program.

Generalization

Given a set of input $T = \{x_1, x_2, x_3, \ldots, x_m\}$,
Where $x_k = \{x_{k,1}, x_{k,2}, \ldots, x_{k,n}\}$
And a set of output $Y = \{y_1, y_2, y_3, \ldots, y_m\}$
Where $y_k \in \{-1, 1\}$ The perceptron predicts a sign function $y(x_k)$, that is:

$$\begin{array}{rll} 1, & if & (w_0 + w_1 x_{k,1} + w_2 x_{k,2} + \ldots + w_3 x_{k,n}) > 0 \\ 0, & if & (w_0 + w_1 x_{k,1} + w_2 x_{k,2} + \ldots + w_3 x_{k,n}) = 0 \\ -1, & if & (w_0 + w_1 x_{k,1} + w_2 x_{k,2} + \ldots + w_3 x_{k,n}) < 0 \end{array}$$

The loss function is

$$J(w_0, w_1, w_2, \ldots, w_n) = -\sum_{x_i \in M} y_i y(x_i)$$

Optimization is to find $(w_0, w_1, \ldots, w_n)$ that minimize J.

Vectorization

$$XW = Y(X) \Leftrightarrow \begin{pmatrix} 1 & x_{1,1} & x_{1,2} & \cdots & x_{1,n} \\ 1 & x_{2,1} & x_{2,2} & \cdots & x_{2,n} \\ \vdots & & & & \\ 1 & x_{k,1} & x_{k,2} & \cdots & x_{k,n} \\ \vdots & & & & \\ 1 & x_{m,1} & x_{m,2} & \cdots & x_{m,n} \end{pmatrix} \cdot \begin{pmatrix} w_0 \\ w_1 \\ w_2 \\ \vdots \\ w_n \end{pmatrix} = \begin{pmatrix} y(x_1) \\ y(x_2) \\ \vdots \\ y(x_k) \\ \vdots \\ y(x_m) \end{pmatrix}$$

$$\nabla_w J = \sum_{x_i \in M} \begin{pmatrix} w_0 \\ w_1 \\ \vdots \\ w_n \end{pmatrix} = \sum_{x_i \in M} \begin{pmatrix} y_i \\ y_i x_{i,1} \\ \vdots \\ y_i x_{i,n} \end{pmatrix} = M^T Y(M) | M \in X, sign(Y(x))! = Y$$

MNIST Data Set

MNIST Data

We will test on zeros and ones on it

Future Work

Future Works

- Logistic Regression and non-linear functions
- Simple Neural Network and Backpropagation
- Big Data set and Stochastic Gradient Descent
- Convolutional Network and Computer Vision
- Black magic, Witchcraft and Metaphysics

Thanks for listening

Thanks for the extra work done by: Eric Wang, Booker Zhao.