Machine Learning (ML)

I. Linear Regression

1.Linear Function

Function Expression:

$$f(x_1, x_2, \dots x_n) = w_0 + w_1x_1 + w_2x_2 + \dots + w_nx_n \ (1)$$
$$= w_0 + \sum_{i=1}^{n} w_ix_i \ (2)$$

Vector Form:

$$f(x_1, x_2, \dots x_n) = w_0 + \begin{bmatrix} w_1 & w_2 & \cdots & w_n \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \dots \\ x_n \end{bmatrix} \ (3)$$
$$= w_0 + \vec{w}^T\vec{x} \ (4)$$

2.Regression

Form:

$$y = f(x_1, x_2, \dots x_n) + r \ (5)$$

In expression above:

$$f(x_1, x_2, \ldots x_n) \ : \ \ we\ named\ it\ as\ y's\ regression\ function$$
$$r \ : \ \ we\ named\ it\ as\ random\ difference$$

y is decided by computable *regression function* and random error *r* .

When *y*'s *regression function* in the form of linear we call it as ***Linear Regression***

3.Theory

As we know $(5)$and we can unfold its former part since (1) :

$$y = w_0 + w_1x_1 + w_2x_2 + \ldots + w_nx_n + r \ (6)$$

1.Conider (1) is unary

It goes:

$$y = w_0 + w_1x_1 + r \ (7)$$

Substitute m independent (x_i, y_i) into formula (7):

$$y_i = w_0 + w_1x_i + r_i \ \ (8)$$
$$(i = 1, \ldots, m)$$

2.Conider (1) is multivariate

We can easily generalize:

$$y^{(i)} = w_0 + w_1x_1^{(i)} + \cdots + w_nx_n^{(i)} + r^{(i)} \ (9)$$
$$(i = 1, \ldots, m)$$

4.Model Building

1.Linear Regression Model

From (9) we can model:

$$y = \begin{bmatrix} w_0 & w_1 & \ldots & w_n \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ \ldots \\ w_n \end{bmatrix} + b \ (10)$$
$$(x_0 = 1)$$

In the expression of form (2) and (4) :

$$y = \sum_{i=1}^{n} w_ix_i + b \ (11)$$
$$= \vec{w}^T\vec{x} + b \ (12)$$

- y : predict function
- $\vec{w}$: model parameter
- $\vec{x}$: feature input
- b : bias

And we make such assumptions:

1.variables are independent from each other

2.variables' effects can be superimposed

2.Loss function

1.Definition

Loss function is used for evaluating the degree of the prediction, aka *the difference between Real-value (y) and Predicted-value ($\hat{y}$)*

As for linear regression, it usually comes as form:

$$L = \frac{1}{2}(y - \hat{y})^2 \ (13)$$

2.Form in this model

Assume that the data set owns m training samples and n features, the loss formula goes :

$$L(w) = \frac{1}{2}\sum_{j=1}^{m}\left[y^{(j)} - \sum_{i=1}^{n} w_i x_i^{(j)} - b\right]^2 \ (14)$$

5.Model Training

1.Target

Figure out the value of the model parameter $(\vec{w})$

2.Principle

To figure out in what condition the loss function is at minimum

3.Menthod

1.Gradient descent

1.Principle

The formula goes:

$$w_(i+1) = w_i - \alpha \nabla f \ (15)$$

Means:

Iterate with the step α which given in the opposite direction of the gradient of the current point .

2.Model

$$w_{i+1} = w_i - \alpha \frac{\partial L(\vec{w})}{\partial(w_i)} \ (16)$$

'cause of

$$\frac{\partial L(\vec{w})}{\partial(w_i)} = -\sum_{j=1}^{m}\left[y^{(j)} - \sum_{i=1}^{n} w_i x_i^{(j)} - b\right] * x_i^{(j)} \ (17)$$

easily infer:

$$w_{i+1} = w_i + \alpha \left\{ \sum_{j=1}^{m} \left[y^{(j)} + \left(\sum_{i=1}^{n} w_i x_i^{(j)} + b \right) \right] * x_i^{(j)} \right\}$$

α : learning rate

$y^{(j)}$: sample value

$\sum_{i=1}^{n} w_i x_i^{(j)} + b$: predicted value

3.Code

View "./Code/Gradient"

HAVE FUN！！！

ShungFinn