1. Linear Regression

To be simple, Regression is to make continuous values: The relationship or the gap between two values has some meaning.

The temperature, for example. 70F and 76F is not only different but the gap 6 is meaningful.

However, another task, classification, is not. This picture is a [dot] and it's a [cat]. Though different with each other, yet the gap has no meaning. And that's why we use different strategy to solve this two kind of problems.

Now, Focus on regression.

1.1 Intuition

Given data, $D = (x^i, y^i)|1 \leq i \leq m, x^i \in R^d, y^i \in R$.

We want to know what y would be when given a new data point x.

We just need to have a mapping function: $h_\theta : R^{d+1} \to R$

That means, $h_\theta = \theta^T[1; x] = \theta_0 + \theta_1 x_1 + \ldots + \theta_d x_d$.

So, we need to find θ to fit the test dataset.

Then we need a way to evaluate our θ, which can help us make the choice of θ.

That's Loss function.

1.2 Loss Function

Square Loss:

$$J(\theta) = \frac{1}{2m} \sum_{i=1}^{m} (y^i - h_\theta(x^i))^2$$

If $J(\theta)$ gets to the minimum, we find the best θ .

Now, it's an optimization problem.

1.3 Optimization

1.3.1 Gradient descent

We have loss function, all the data and we want the loss function to be minimized.

Just follow its gradient and change θ.

$$\begin{aligned}\theta_j &\leftarrow \theta_j - \alpha\frac{\partial J}{\partial \theta_j} \\ &\leftarrow \theta_j - \frac{\alpha}{m}\sum_{i=1}^{m}(y^i - \theta^T[1;x^i])x_j^i\end{aligned}$$

1.3.2 Stochastic Gradient descent

Then, stochastic gradient descent came out. That's because it'll take huge resources to compute all the data point for just one step of θ.

So, Stochastic gradient descent just use a batch of data points or even one single point.

The following it's the version of a single point.

$$\theta_j \leftarrow \theta_j + \frac{\alpha}{m}(y^i - \theta^T[1;x^i])x_j^i$$

1.3.3 Vectorize

Vectorizing (Matrix operation) is much more convenient and efficient than for loop.

$$\begin{aligned}\theta &\leftarrow \theta - \alpha\triangledown_\theta J(\theta) \\ &\leftarrow \theta - \frac{\alpha}{m}X^T(X\theta - y)\end{aligned}$$

1.3.4 Closed-form

$$\begin{aligned}&\frac{\partial J(\theta)}{\partial\theta} = 0 \\ &\theta = (X^TX)^{-1}X^Ty\end{aligned}$$

1.4 Probabilistic interpretation

Assume $(x^i, y^i) \sim N(0, \sigma^2)$, we can get $P((x^i, y^i)|\theta) = \frac{1}{\sqrt{2\pi\sigma^2}}exp(-\frac{(y^i - \theta^Tx^i)^2}{2\sigma^2})$.

And as for the $P(D|\theta)$, it's the product of each data points in the dataset.

We can get $P(D|\theta) = \Pi_{i=1}^m P((x^i, y^i))$.

And our goal is to get $\theta^*_{MLE} = argmax_\theta P(D|\theta)$. (MLE here stands for maximum likelihood estimation)

It's equal to $\theta^*_{MLE} = argmin_\theta - log(P(D|\theta))$. We use log here, since it can convert product to sum.

let $NLL(\theta)$, standing for **negative log likelihood**, represents $-log(p(D|\theta))$.

$$NLL(\theta) = -\sum_{i=1}^{m}(log\frac{1}{\sqrt{2\pi\sigma^2}} - \frac{(y^i - \theta^T x^i)^2}{2\sigma^2})$$

Recall that our goal is to find θ that can minimize $NLL(\theta)$, then only the squared error loss matters.

1.5 Non-Linear

One simple way to make model fit non-linear dataset is to use feature augmentation ϕ.

Eg, ϕ: $(x) \rightarrow (x, x^2, x^3 \dots)$

So, Just replace X with $\phi(x)$, we can get a non-linear model.

2. Ridge regression(L2)

2.1 Loss function

To fix the **overfit** problem, which means θ can fit the test data perfectly while it has bad performance on unseen data, we need to **regularize** θ.

Regularization is a trade-off between the **effect** of θ_j and the **value** of θ_j. In other words, model will focus more on important weights.

With regularization, the loss function now is:

$$J(\theta) = \frac{1}{2m}\sum_{i=1}^{m}(y^i - h_\theta(\phi^i))^2 + \frac{\lambda}{2m}\sum_{j=1}^{d}\theta_j^2$$

The following things are the same as Linear regression: solve the optimization problem.

2.2 Optimization

2.2.1 Batch Gradient Descent

$$\theta_j \leftarrow \theta_j - \alpha[\frac{1}{m}\phi^T(\phi\theta - y) + \frac{\lambda}{m}\theta_j(j \neq 0)]$$

2.2.2 Stochastic Gradient Descent

$$\theta_j \leftarrow \theta_j - \alpha[\frac{1}{m}(\theta^T\phi^i - y^i)\phi^i + \frac{\lambda}{m}\theta_j(j \neq 0)]$$

2.2.3 Closed form

$$\theta^* = (\phi^T\phi + \lambda I)^{-1}\phi^T y$$

3. MAP: Maximum A Priori estimation of θ

With Bayesian Rule, we have

$$P(\theta|D) \propto P(D|\theta)P(\theta)$$

$P(D|\theta)$ is the likelihood of data D given the parameter θ.

$P(\theta)$ is the priori distribution on the parameter θ.

Our goal is to maximize the $P(\theta|D)$.

Assume that:

$$y^i = \theta^T x^i + \epsilon^i, \;\; \epsilon^i \sim N(0, \sigma^2)$$
$$P(\theta) = N(\theta|0, \alpha^2 I)$$

We can have:

$$P(D|\theta) = \Pi_{i=1}^m \frac{1}{\sqrt{2\pi\sigma^2}} exp(-\frac{(y^i - \theta^T x^i)^2}{2\sigma^2})$$
$$P(\theta) = (\frac{1}{2\pi\alpha^2})^{\frac{\alpha+1}{2}} exp(-\frac{\theta^T\theta}{2\alpha^2})$$

We can then get $P(\theta|D)$, and $\theta_{MAP} = argmax_\theta P(\theta|D)$. We can use the same strategy like SGD to solve the optimization problem. And you can change the assumption to any other distribution.