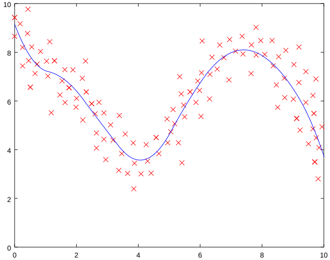
Regularization

Lecture 4 - DAMLF | ML1

human+machine INTELLIGENCE

Frankfurt School

Regression Analysis



Univariate Linear Regression:

$$h(x; \boldsymbol{\theta}) = \theta_0 + \theta_1 x$$

Multiple Linear Regression:

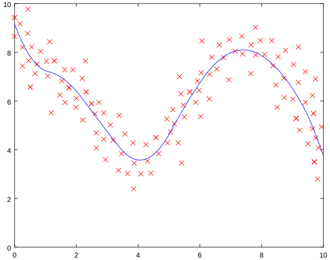
$$h(x; \boldsymbol{\theta}) = \theta_0 x_0 + \theta_1 x_1 + \theta_2 x_2 \cdots + \theta_n x_n$$

where $x_0 = 1$

Polynomial Regression:

$$h(x; \boldsymbol{\theta}) = \theta_0 + \theta_1 x + \theta_2 x^2 \cdots + \theta_n x^n$$

Basis Function Regression



Basis Function Regression:

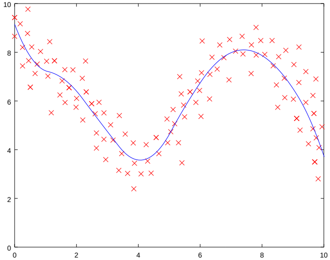
$$h(x; \boldsymbol{\theta}) = \theta_0 + \theta_1 z_1 + \theta_2 z_2 \cdots + \theta_n z_n$$

where $z_1, z_2, \ldots, z_n$ are built from a single input x:

$$f_i(x) = z_i, \text{ for } i = 1, 2, \ldots, n$$

The $f_i(\cdot)$ are called **basis functions**.

Basis Function Regression



Basis Function Regression:

$$h(x;\boldsymbol{\theta}) = \theta_0 + \theta_1 z_1 + \theta_2 z_2 \cdots + \theta_n z_n$$

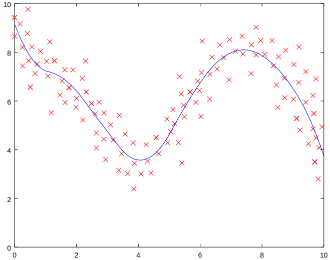
where $z_1, z_2, \ldots, z_n$ are built from a single input x:

$$f_i(x) = z_i, \text{ for } i = 1, 2, \ldots, n$$

The $f_i(\cdot)$ are called **basis functions**.

$h(x;\boldsymbol{\theta})$ is still a linear model: the coefficients θ_i never divide or multiply each other.

Polynomial Regression

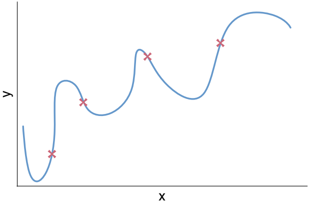


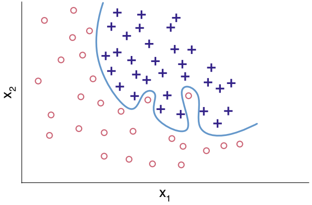
Polynomial Regression:

$$h(x; \boldsymbol{\theta}) = \theta_0 + \theta_1 x + \theta_2 x^2 \cdots + \theta_n x^n$$

where the basis function is $f_i(x) = x^i$, for $i = 1, 2, \ldots, n$

Overfitting





linear regression

$h(x; \boldsymbol{\theta}) = \theta^T x$

logistic regression

$h(x; \boldsymbol{\theta}) = g(\theta^T x)$

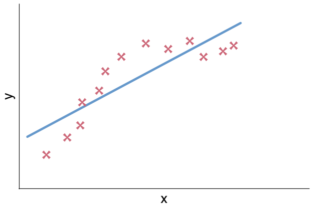
Overfitting occurs when there are **too many features/adjustable parameters**.

Although the learned hypothesis $h(x; \boldsymbol{\theta})$ fits the **training set** very well, that is:

$$J(\theta) \approx 0$$

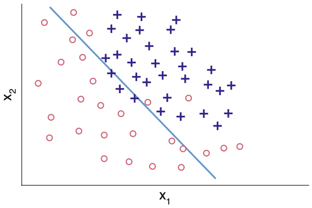
The model fails to give **accurate predictions** for new (out of sample) examples.

Underfitting



linear regression

$h(x; \boldsymbol{\theta}) = \theta^T x$



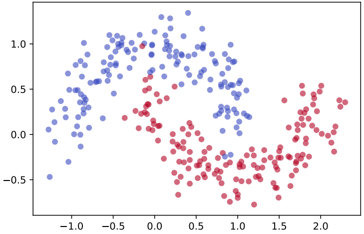
logistic regression

$h(x; \boldsymbol{\theta}) = g(\theta^T x)$

Underfitting occurs when there are **too few features**.

Although the learned hypothesis $h(x; \boldsymbol{\theta})$ is very **simple** and generalizable, the model introduces a **bias** that **skews predictions** for new examples.

Generalization

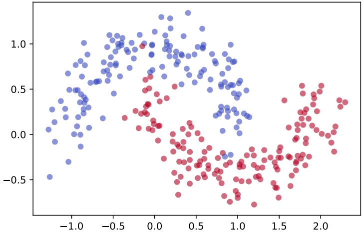


Fitting is easy. Predicting is hard.

The main challenge in machine learning is for a model to perform well on new, previously unseen inputs.

The ability to perform well on new, previously unobserved inputs is called **generalization**.

Model Capacity



Informally, a learning algorithm's **capacity** is its ability to fit a wide range of functions.

- *Low capacity* models are prone to underfitting.
- *High capacity* models are prone to overfitting.

The trick is is to match a model's capacity to the complexity of the task.

How to Address Overfitting

There are basically two options:

1. Reduce the number of Features

How to Address Overfitting

There are basically two options:

1. Reduce the number of Features
    - Manually pick out which features to keep and which to discard
    - **Model selection** algorithms (BIC, AIC, etc.)
      used less frequently in deep learning era

How to Address Overfitting

There are basically two options:

1. Reduce the number of Features
    - Manually pick out which features to keep and which to discard
    - **Model selection** algorithms (BIC, AIC, etc.)
      used less frequently in deep learning era
2. **Regularization**
    - Keep *all* features, but reduce the magnitude of parameters θ_j

Regularization

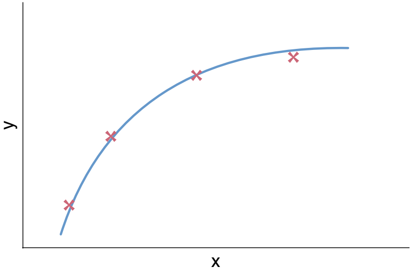
The **idea** behind regularization is to **discount** the role that multiple features play in a hypothesis $h(x; \boldsymbol{\theta})$.

Regularization

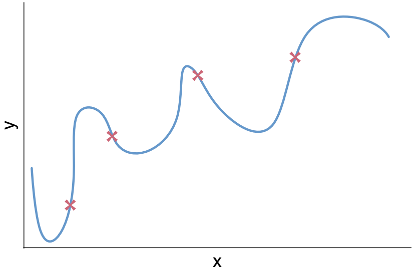
The **idea** behind regularization is to **discount** the role that multiple features play in a hypothesis $h(x; \boldsymbol{\theta})$.

How: *Make* θ_j *very small, close to zero.*

Intuition



$\theta_0 + \theta_1 x + \theta_2 x^2$



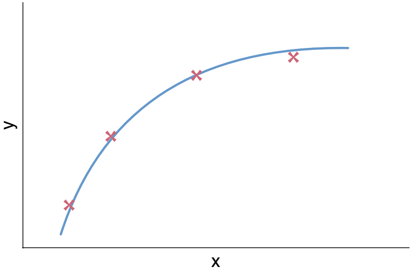
$\theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4$

overfit: high variance

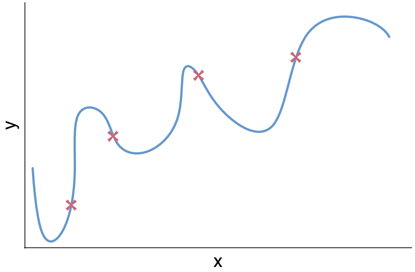
Suppose we made θ_3 and θ_4 **very small**.

How:

Intuition



$\theta_0 + \theta_1 x + \theta_2 x^2$



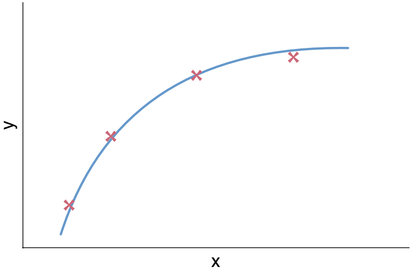
$\theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4$

overfit: high variance

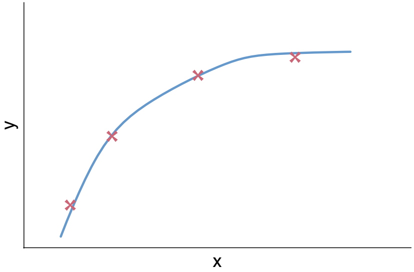
Suppose we made θ_3 and θ_4 **very small**.

How: $\min_\theta \frac{1}{2m} \sum_{i=1}^{m} (h_\theta^{(i)} - y^{(i)})^2$

Intuition



$\theta_0 + \theta_1 x + \theta_2 x^2$



$\theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4$

regularized θ_3, θ_4

Suppose we made θ_3 and θ_4 **very small**.

How: $\min_{\theta} \frac{1}{2m} \sum_{i=1}^{m} (h(x^{(i)}; \boldsymbol{\theta}) - y^{(i)})^2 + 1000 \cdot \theta_3^2 + 1000 \cdot \theta_4^2$

Intuition

Instead of picking some θ_js (θ_3, θ_4) to discount while leaving other θ_js (θ_1, θ_2) at full strength, **regularization** discounts **all** θ_js.

Example

Features: $x_1, x_2, \ldots, x_{1000}$

Parameters: $\theta_0 \theta_1, \theta_2, \ldots, \theta_{1000}$

Example

Features: $x_1, x_2, \ldots, x_{1000}$

Parameters: $\theta_0 \theta_1, \theta_2, \ldots, \theta_{1000}$

Unlike the previous example, where θ_3 and θ_4 were picked out for discounting, it is likely to be difficult to know in advance which of the 1001 parameters ought to be discounted.

Example

Features: $x_1, x_2, \ldots, x_{1000}$

Parameters: $\theta_0\theta_1, \theta_2, \ldots, \theta_{1000}$

Idea: Modify the **cost function** $J(\theta)$ to **discount** every $\theta_j \in \theta$.

Ridge Regularization

Regularized Cost Function

$$J(\theta) = \frac{1}{2m}\sum_{i=1}^{m}(h(x^{(i)};\boldsymbol{\theta}) - y^{(i)})^2 + \lambda\sum_{j=1}^{n}\theta_j^2$$

Ridge Regularization

Regularized Cost Function

$$J(\theta) = \frac{1}{2m}\sum_{i=1}^{m}(h(x^{(i)};\boldsymbol{\theta}) - y^{(i)})^2 + \lambda\sum_{j=1}^{n}\theta_j^2$$

Note: By convention, θ_0 is **not** discounted. Thus, since the regularization parameter λ applies to

$$\theta_1\theta_2,\ldots\theta_n,$$

the summation $\sum_{j=1}^{n}$ is indexed from 1 to n

Ridge Regularized Cost Function

Regularized Cost Function:

$$J(\theta) = \frac{1}{2m}\sum_{i=1}^{m}(h(x^{(i)};\boldsymbol{\theta}) - y^{(i)})^2 + \lambda\sum_{j=1}^{n}\theta_j^2$$

Regularization term: $\lambda\sum_{j=1}^{n}\theta_j^2$

Regularization parameter: λ

Fitting term: $\frac{1}{2m}\sum_{i=1}^{m}(h(x^{(i)};\boldsymbol{\theta}) - y^{(i)})^2$

Regularized Linear Regression

Regularized Linear Regression

The optimization objective is to choose θ to minimize the regularized cost function:

$$J(\theta) = \frac{1}{2m}\left(\sum_{i=1}^{m}\left(h(x^{(i)};\boldsymbol{\theta}) - y^{(i)}\right)^2 + \lambda\sum_{j=1}^{n}\theta_j^2\right)$$

Regularized Linear Regression

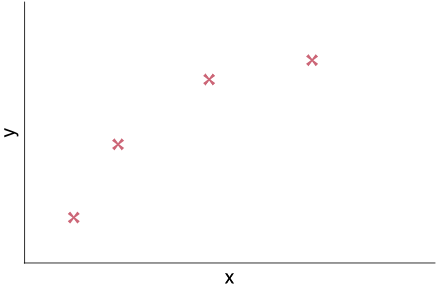
The optimization objective is to choose θ to minimize the regularized cost function:

$$J(\theta) = \frac{1}{2m}\left(\sum_{i=1}^{m}\left(h(x^{(i)};\boldsymbol{\theta}) - y^{(i)}\right)^2 + \lambda\sum_{j=1}^{n}\theta_j^2\right)$$

Question: What happens if we pick an extremely large λ?

Regularized Linear Regression

$$J(\theta) = \frac{1}{2m}\left(\sum_{i=1}^{m}(h(x;\boldsymbol{\theta}) - y^{(i)})^2 + \lambda \sum_{j=1}^{n} \theta_j^2\right)$$

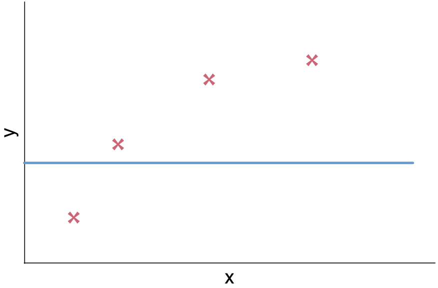


$\theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4$

Suppose $\lambda = 10^9$

Regularized Linear Regression

$$J(\theta) = \frac{1}{2m}\left(\sum_{i=1}^{m}(h(x;\boldsymbol{\theta}) - y^{(i)})^2 + \lambda\sum_{j=1}^{n}\theta_j^2\right)$$



$\theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4$

Suppose $\lambda = 10^9$

Then, $h(x;\boldsymbol{\theta}) \approx \theta_0$

and **underfits** the data.

Gradient Descent

Algorithm:
Gradient Descent

» Enter loop

Repeat:

$$\theta_0 := \theta_0 - \alpha \frac{1}{m} \sum_{i=1}^{m} \left(h(x^{(i)}; \boldsymbol{\theta}) - y^{(i)}\right) x_0^{(i)}$$

$$\theta_j := \theta_j - \alpha \frac{1}{m} \sum_{i=1}^{m} \left(h(x^{(i)}; \boldsymbol{\theta}) - y^{(i)}\right) x_j^{(i)}$$

$(j = 1, 2, \ldots, n)$

simultaneous update $\theta_0, \theta_j s$

Stop at convergence

» Exit loop

Remarks:

The **first step** is to rewrite the basic cost function to separate θ_0, which will not be discounted, from θ_1 to θ_n,

which will be discounted.

Gradient Descent for Regularized Linear Regression

Algorithm:
Gradient Descent

» Enter loop
Repeat:

$$\theta_0 := \theta_0 - \alpha \frac{1}{m} \sum_{i=1}^{m} \left(h(x^{(i)}; \boldsymbol{\theta}) - y^{(i)}\right) x_0^{(i)}$$

$$\theta_j := \theta_j - \alpha \left(\frac{1}{m} \sum_{i=1}^{m} \left(h(x^{(i)}; \boldsymbol{\theta}) - y^{(i)}\right) x_j^{(i)} + \frac{\lambda}{m}\theta_j\right)$$

$(j = 1, 2, \ldots, n)$
simultaneous update θ_0, θ_j*s*

Stop at convergence

» Exit loop

Remarks:

Then, add the **regularization term**.

Gradient Descent for Regularized Linear Regression

Algorithm:
Regularized Gradient Descent

» Enter loop
Repeat:

$$\theta_0 := \theta_0 - \alpha \frac{1}{m} \sum_{i=1}^{m} \left(h(x^{(i)}; \boldsymbol{\theta}) - y^{(i)} \right) x_0^{(i)}$$

$$\theta_j := \theta_j (1 - \alpha \frac{\lambda}{m}) - \alpha \frac{1}{m} \sum_{i=1}^{m} \left(h(x^{(i)}; \boldsymbol{\theta}) - y^{(i)} \right) x_j^{(i)}$$

$(j = 1, 2, \ldots, n)$
simultaneous update $\theta_0, \theta_j s$

Stop at convergence

» Exit loop

Remarks:

Finally, rearrange terms.

Algorithm:
Regularized Gradient Descent

» Enter loop

Repeat:

$$\theta_0 := \theta_0 - \alpha \frac{1}{m} \sum_{i=1}^{m} \left(h(x^{(i)}; \boldsymbol{\theta}) - y^{(i)} \right) x_0^{(i)}$$

$$\theta_j := \theta_j(1 - \alpha \frac{\lambda}{m}) - \alpha \frac{1}{m} \sum_{i=1}^{m} \left(h(x^{(i)}; \boldsymbol{\theta}) - y^{(i)} \right) x_j^{(i)}$$

$(j = 1, 2, \ldots, n)$

simultaneous update $\theta_0, \theta_j s$

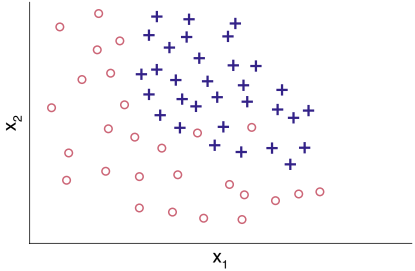
Stop at convergence

» Exit loop

Note: The term $0 \ll (1 - \alpha \frac{\lambda}{m}) < 1$ and $1 - \alpha \frac{\lambda}{m} \approx 1$

Regularized Logistic Regression

Regularized Logistic Regression

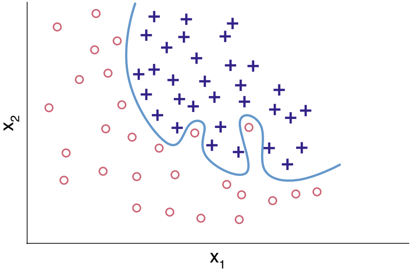


$$h(x;\boldsymbol{\theta}) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_1^2 + \theta_3 x_1^2 x_2 \\ + \theta_4 x_1^2 \theta_2^2 + \theta_5 x_1^2 x_2^3 + \cdots)$$

Cost Function:

$$J(\theta) = -\left(\frac{1}{m}\sum_{i=1}^{m} y^{(i)} \log h(x^{(i)};\boldsymbol{\theta}) + (1 - y^{(i)}) \log\left(1 - h_\theta(x^{(i)})\right)\right)$$

Regularized Logistic Regression

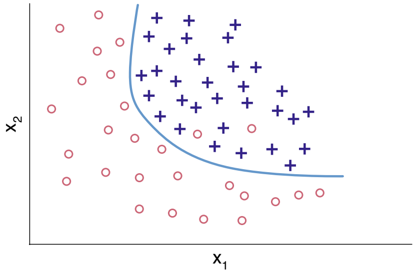


$$h(x;\boldsymbol{\theta}) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_1^2 + \theta_3 x_1^2 x_2 + \theta_4 x_1^2 \theta_2^2 + \theta_5 x_1^2 x_2^3 + \cdots)$$

Cost Function:

$$J(\theta) = -\left(\frac{1}{m}\sum_{i=1}^{m} y^{(i)} \log h(x^{(i)};\boldsymbol{\theta}) + (1-y^{(i)}) \log\left(1-h_\theta(x^{(i)})\right)\right)$$

Regularized Logistic Regression



$$h(x;\boldsymbol{\theta}) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_1^2 + \theta_3 x_1^2 x_2$$
$$+\theta_4 x_1^2 \theta_2^2 + \theta_5 x_1^2 x_2^3 + \cdots)$$

Cost Function:

$$J(\theta) = -\left(\frac{1}{m}\sum_{i=1}^{m} y^{(i)} \log h(x^{(i)};\boldsymbol{\theta}) + (1-y^{(i)}) \log\left(1 - h_\theta(x^{(i)})\right) + \frac{\lambda}{2m}\sum_{j=1}^{n}\theta_j^2\right)$$

Algorithm:
Regularized Gradient Descent

» Enter loop

Repeat:

$$\theta_0 := \theta_0 - \alpha \frac{1}{m} \sum_{i=1}^{m} \left(h(x^{(i)}; \boldsymbol{\theta}) - y^{(i)}\right) x_0^{(i)}$$

$$\theta_j := \theta_j(1 - \alpha \frac{\lambda}{m}) - \alpha \frac{1}{m} \sum_{i=1}^{m} \left(h(x^{(i)}; \boldsymbol{\theta}) - y^{(i)}\right) x_j^{(i)}$$

$(j = 1, 2, \ldots, n)$

simultaneous update $\theta_0, \theta_j s$

Stop at convergence

» Exit loop

Note: $h(x; \boldsymbol{\theta}) = \dfrac{1}{1 + e^{-\theta^T x}}$

L1 vs L2 Regularization

Ridge (L2) Regularization

$\lambda \sum_{j=1}^{n} \theta_j^2$

L2 norm:

- *sum of squared θ_i's*
- *unique*
- *not robust to outliers*
- *all features used*

Lasso (L1) Regularization

$\lambda \sum_{j=1}^{n} |\theta_j|$

L1 norm:

- *sum of absolute values of θ_i's*
- *not unique*
- *robust to outliers*
- *most features not used*

References