Linear Regression

Cost Function

$$L(\mathrm{w}) = \frac{1}{2N}\sum_{i=1}^{N}\left(y_i - x_i^T\mathrm{w}\right)^2 = \frac{1}{2N}\left\|\mathrm{y} - \mathrm{X}^T\mathrm{w}\right\|_2^2$$

- Need to optimize the cost function: $\mathrm{w} = \arg\min_{\mathrm{w}} L(\mathrm{w})$
- Solution for cost function: $\frac{\nabla L(\mathrm{w})}{\nabla \mathrm{w}} = \frac{1}{N}\mathrm{X}\left(\mathrm{X}^T\mathrm{w} - \mathrm{y}\right) = 0 \Leftrightarrow \mathrm{X}\mathrm{X}^T\mathrm{w} = \mathrm{X}\mathrm{y}$
  - If $\mathrm{X}\mathrm{X}^T$ is invertible, $\mathrm{w} = \left(\mathrm{X}\mathrm{X}^T\right)^{-1}\mathrm{X}\mathrm{y}$
  - o.w. we use pseudo-inverse, $\mathrm{w} = \left(\mathrm{X}\mathrm{X}^T\right)^{+}\mathrm{X}\mathrm{y}$

Bias Trick

- To make $y = \mathrm{x}^T\mathrm{w} + b$ become $y = \mathrm{x}^T\mathrm{w}$ $\begin{cases} \mathrm{x} = \begin{bmatrix} x_1 & x_2 & \dots & x_n \end{bmatrix}^T \rightarrow \mathrm{x} = \begin{bmatrix} x_1 & x_2 & \dots & x_n & 1 \end{bmatrix}^T \\ \mathrm{w} = \begin{bmatrix} w_0 & w_1 & \dots & w_n \end{bmatrix}^T \rightarrow \mathrm{w} = \begin{bmatrix} w_0 & w_1 & \dots & w_n & b \end{bmatrix}^T \end{cases}$

Drawbacks

- Linear regression is very ***sensitive to noise***, so we need to
  - Pre-processing data before applying linear regression
  - Change the cost function to make it ***robust to noise***: **<u>Huber Regression</u>**
- Linear Regression cannot represent the high order function

Ridge Regression

- Aside from the purpose of avoiding overfitting, we can apply ridge when $\mathrm{X}\mathrm{X}^T$ is not invertible and we don't want to apply pseudo-inverse

$$L_2(\mathrm{w}) = \frac{1}{2N}\left(\left\|\mathrm{y} - X^T\mathrm{w}\right\|_2^2 + \lambda\left\|\mathrm{w}\right\|_2^2\right)$$

$$\frac{\nabla L_2(\mathrm{w})}{\nabla \mathrm{w}} = 0 \Leftrightarrow \frac{1}{N}\left(\mathrm{X}\left(\mathrm{X}^T\mathrm{w} - \mathrm{y}\right) + \lambda\mathrm{w}\right) = 0 \Leftrightarrow \left(XX^T + \lambda\mathrm{I}\right)\mathrm{w} = \mathrm{X}\mathrm{y}$$

- Prove $A = XX^T + \lambda\mathrm{I}$ is positive definite, so A is always invertible

$$\mathrm{w}^T A\mathrm{w} = \mathrm{w}^T\left(\mathrm{X}\mathrm{X}^T + \lambda\mathrm{I}\right)\mathrm{w} = \mathrm{w}^T\mathrm{X}\mathrm{X}^T\mathrm{w} + \lambda\mathrm{w}^T\mathrm{w} = \left\|\mathrm{X}^T\mathrm{w}\right\|_2^2 + \lambda\left\|\mathrm{w}\right\|_2^2 > 0$$

- So, $\frac{\nabla L_2(\mathrm{w})}{\nabla \mathrm{w}} = 0$ always have 1 solution: $\mathrm{w} = \left(XX^T + \lambda\mathrm{I}\right)^{-1}\mathrm{X}\mathrm{y}$

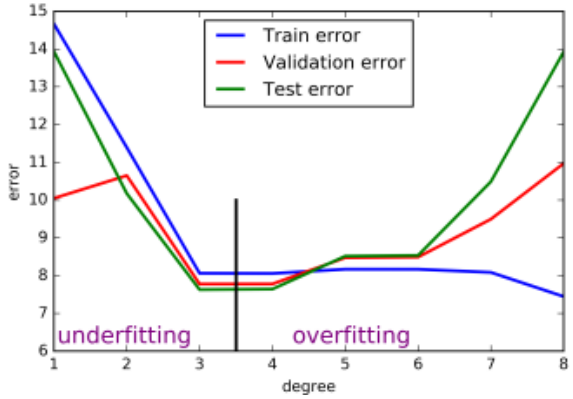
Overfitting

- If our model is too fit with the data, the model is more likely to lose its generalization
- We can use the metrics like MSE, Cross entropy, Hinge Loss (Multi class SVM) to measure the error of training and test set
- If the metric of error in training set < test set, we say it's overfitting

Validation

- Validation set is small dataset extracted from training set
- Its analogy: Suppose we are reviewing before taking exam. Exam questions in previous years is considered training set and exam questions in the exam we are gonna take is test set. When reviewing, we separate the training set into 2 parts: first part has questions and answer, second part just has questions

- So now, our model will have 3 errors, so our model needs to have small *training error* and *validation error*, so it's more likely to have small *test error*
- **Based on this method, we can tune the model or find different model on the training set and apply them on validation set to find small validation error**
- E.g. In regression, we increase the degree, the training error definitely decreases and validation error also decreases. But when we increase the degree to some extent, the validation error will stop decreasing and start to increase, so we will stop tune the degree at this point



Hình 8.2: Lựa chọn mô hình dựa trên validation error.

- Note: Test set and Validation set need to have something in common like distribution

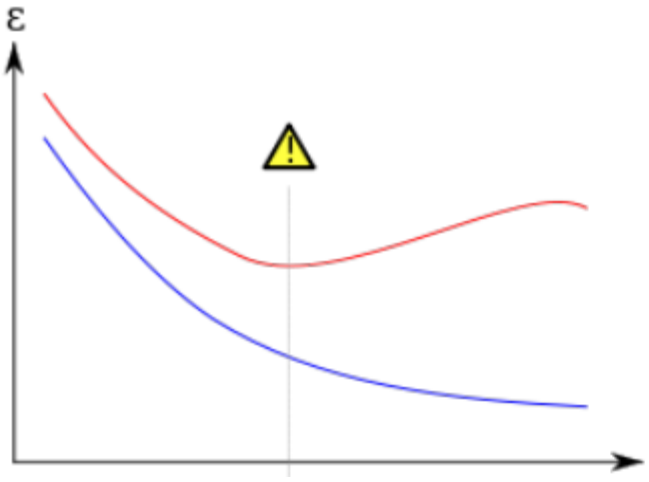
Cross Validation

- Suppose we just have not so big training set to build the model
- If we extract to much training data to make validation set, training set will note have enough data to build the model. So now validation set needs to be small. But when validation set is small, overfitting may occur in validation set
- Cross Validation will separate training set into k subsets. We run the model k times, each time, one of k subset will be considered as validation set, the model will be built based on k-1 subsets as training set
- Final model will be the mean of training error and mean of validation error
- **Leave-one-out**: k = number of data points, extract 1 data point to make validation set

Regularization

- Big disadvantage of cross validation is building the model k times
- In the regression above, we just need to tune the degree, but in the other models, we have to tune other parameters and the range of parameter is large, so cross validation is too complex in such models like this

Early Stopping



Hình 8.3: Early stopping. Đường màu xanh là training error, màu đỏ là validation error. Trục x thể hiện số lượng vòng lặp, trục y là giá trị error. Thuật toán huấn luyện dừng lại tại vòng lặp mà validation error đạt giá trị nhỏ nhất (Nguồn: Over-fitting – Wikipedia).

Add to Cost function

$$L_{reg}(\theta) = L(\theta) + \lambda R(\theta)$$

- LASSO Regression: $R(\theta) = \|\mathbf{w}\|_1$, parameter **w** is sparse vector. Data points which value in **w** ≠ 0 means this data point is important, contribute to predict output, otherwise this data point is note important. So LASSO regression is considered as FEATUE SELECTION and robust to noise

- But the LASSO regression has disadvantage: it's hard to find derivatives of $\|\mathbf{w}\|_1$