Fundamentals of Machine Learning

Il-Chul Moon
Dept. of Industrial and Systems Engineering
KAIST
icmoon@kaist.ac.kr

Weekly Objectives

- Learn the most classical methods of machine learning
  - Rule based approach
  - Classical statistics approach
  - Information theory appraoch
- Rule based machine learning
  - How to find the specialized and the generalized rules
  - Why the rules are easily broken
- Decision Tree
  - How to create a decision tree given a training dataset
  - Why the tree becomes a weak learner with a new dataset
- Linear Regression
  - How to infer a parameter set from a training dataset
  - Why the feature engineering has its limit

LINEAR REGRESSION

How about statistical approach?

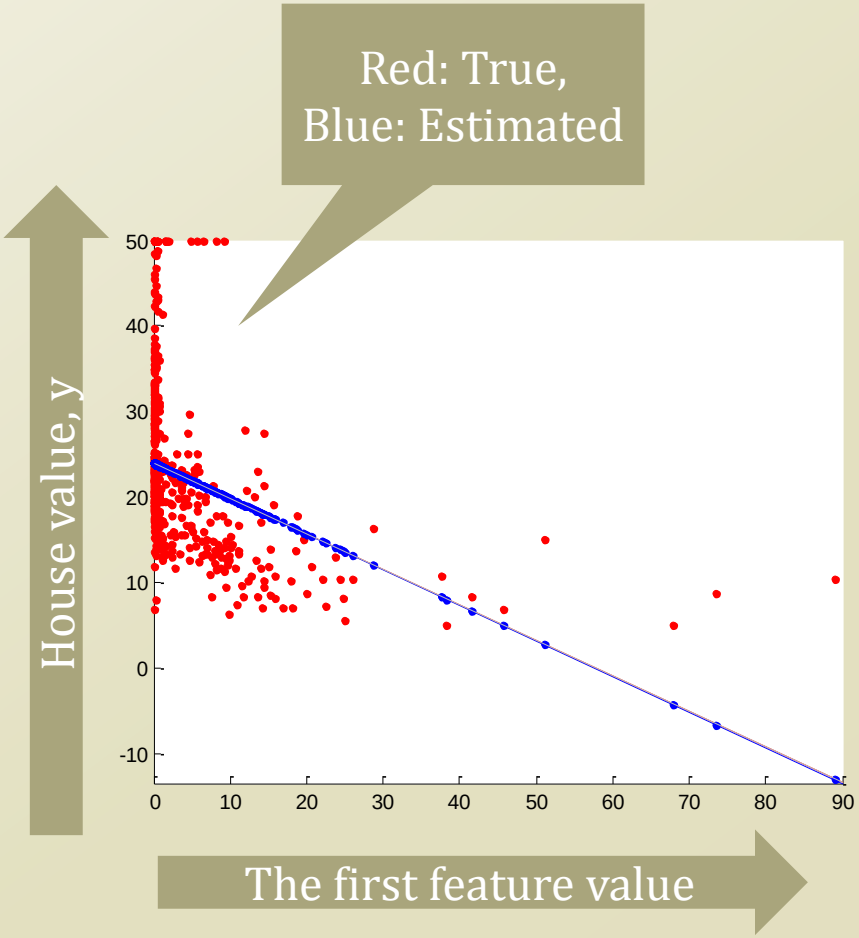
- http://archive.ics.uci.edu/ml/datasets/Housing
- Housing dataset
  - 13 numerical independent values
  - 1 numerical dependent value
- How to create an approximated function?
  - Do you remember that the machine learning is the function approximation process?
- Here,
  - Our hypothesis is
    - The house value will be the linearly weighted sum of the feature values.
    - $h: \hat{f}(x;\theta) = \theta_0 + \sum_{i=1}^{n} \theta_i x_i = \sum_{i=0}^{n} \theta_i x_i$
    - $\boldsymbol{n}$ is the number of the feature values.
    - Two aspects: the linearly weight sum (the model), the parameter $\boldsymbol{\theta}$
  - The first effort is finding the better $\boldsymbol{\theta}$, just like the thumbtack

Finding $\boldsymbol{\theta}$ in Linear Regression

- To make the hypothesis better, we need to find the better $\boldsymbol{\theta}$
  - $h: \hat{f}(x;\theta) = \sum_{i=0}^{n} \theta_i x_i \rightarrow \hat{f} = X\theta$
  - $X = \begin{pmatrix} 1 & \cdots & x_n^D \\ \vdots & \ddots & \vdots \\ 1 & \cdots & x_n^D \end{pmatrix}, \theta = \begin{pmatrix} \theta_0 \\ \theta_1 \\ \cdots \\ \theta_n \end{pmatrix}$
- The reality would be the noisy, so….
  - $f(x;\theta) = \sum_{i=0}^{n} \theta_i x_i + e = y \rightarrow f = X\theta + e = Y$
- The difference is the error from the noise, so let's make it minimum
  - $\hat{\theta} = argmin_\theta (f - \hat{f})^2 = argmin_\theta (Y - X\theta)^2$
    $= argmin_\theta (Y - X\theta)^T (Y - X\theta) = argmin_\theta (Y - X\theta)^T (Y - X\theta)$
    $= argmin_\theta (\theta^T X^T X\theta - 2\theta^T X^T Y + Y^T Y) = argmin_\theta (\theta^T X^T X\theta - 2\theta^T X^T Y)$
- Now, we need to optimize $\boldsymbol{\theta}$

Optimized θ

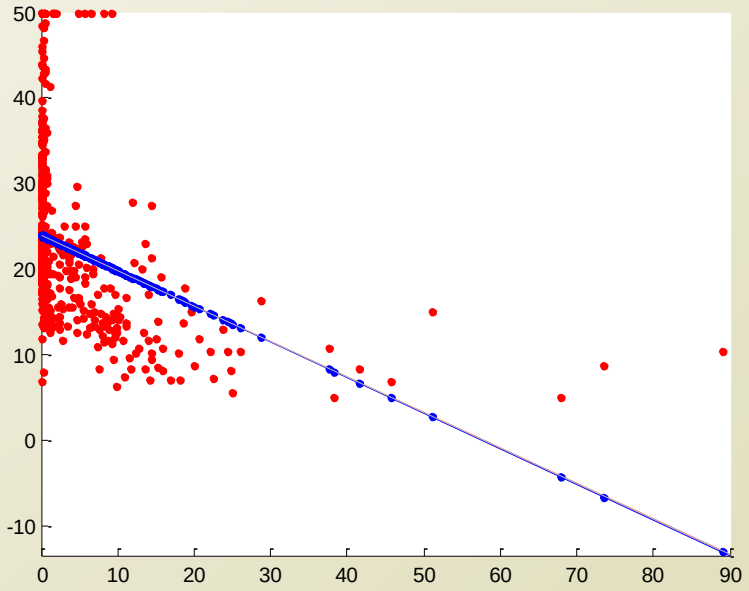
- $\hat{\theta}$
  $= argmin_{\theta}(\theta^T X^T X\theta - 2\theta^T X^T Y)$
- Same technique as in Thumbtack
  - $\nabla_{\theta}(\theta^T X^T X\theta - 2\theta^T X^T Y)$=0
  - $2X^T X\theta - 2X^T Y = 0$
  - $\theta = (X^T X)^{-1} X^T Y$
- Great! We know X and Y, so we can compute θ
- Let's calculate and watch the performance!
  - For demonstration purpose, we limit the dimension to the constant and the first feature
- MLE if error follows the normal distribution

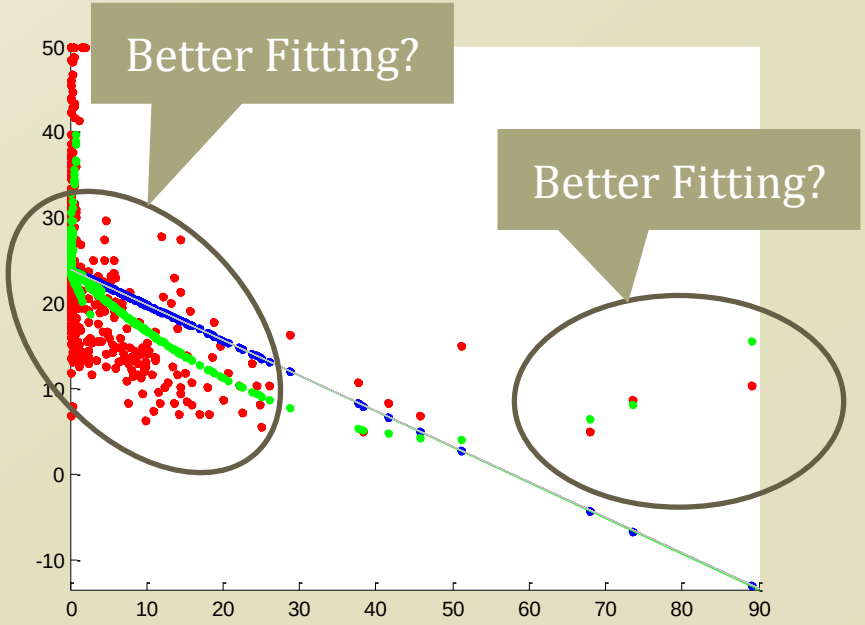


If you want more….

- Actually, you can increase the number of features, a.k.a. dimension
  - You have a value *x* in the previous fitted figure
  - How about adding x^2, x^3, x^4….?
- You can improve the result!
  - Is that right?
- We are going to come back!

$$h: \hat{f}(x;\theta) = \sum_{i=0}^{n}\sum_{j=1}^{m} \theta_{i,j}\phi_j(x_i)$$





Too Brittle to Be Used Naively

- What we are doing
  - Approximating a function to the dataset
  - The function can be
    - Discrete logics
    - Statistical model
  - Often, the function type is given
    - The parameters of the functions are the target of the analysis
- Alternatives in finding the parameter
  - MLE or MAP
  - Engineering the features
  - Setting the generalization and the specialization level
- Best choice among the alternatives
  - If we have the perfect data, we will know
  - But we don't