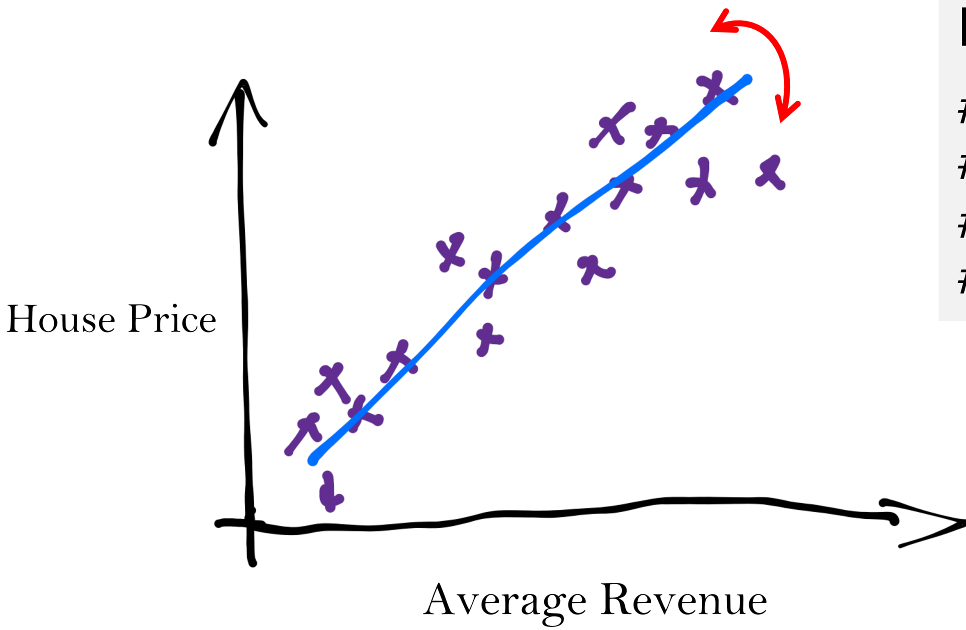
Machine Learning

Chapter 2: Linear Regression

Fall 2022

Instructor: Xiaodong Gu

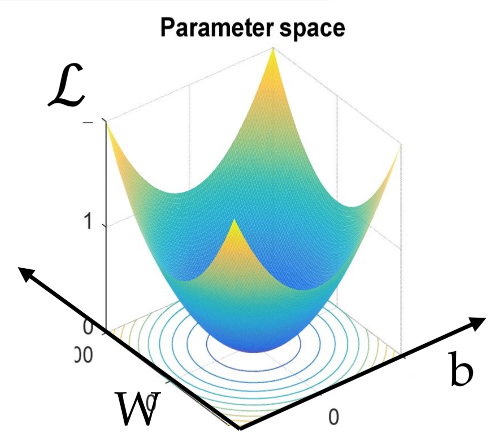
Recall: Key Elements of Machine Learning



Elements:

#1 Data (Experience)
#2 Model (Hypothesis)
#3 Loss Function (Objective)
#4 Optimization (Improve)

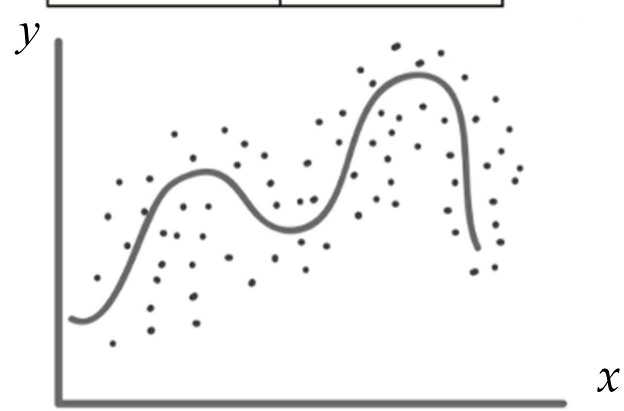
$$\theta^* = \underset{\theta}{\operatorname{argmin}} \mathcal{L}(\theta|\mathcal{D})$$



Regression in Machine Learning

Machine Learning
- **Supervised Learning**
  - Regression (✓)
  - Classification
  - …
- Unsupervised Learning
- Reinforcement Learning

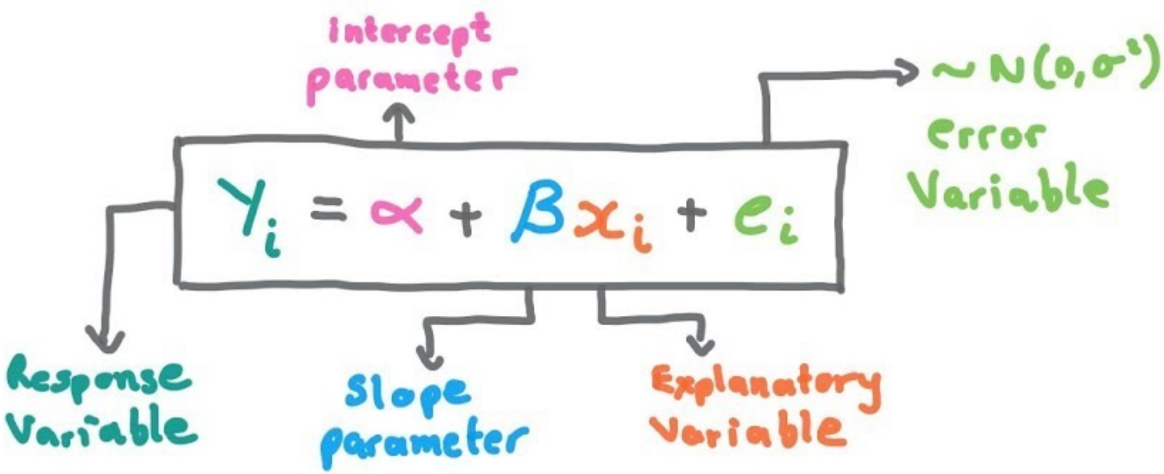
Advertisement	Sales
$90	$1000
$120	$1300
$150	$1800
$100	$1200
$130	$1380
$200	??



Regression: predicts real-valued labels

Regression Model

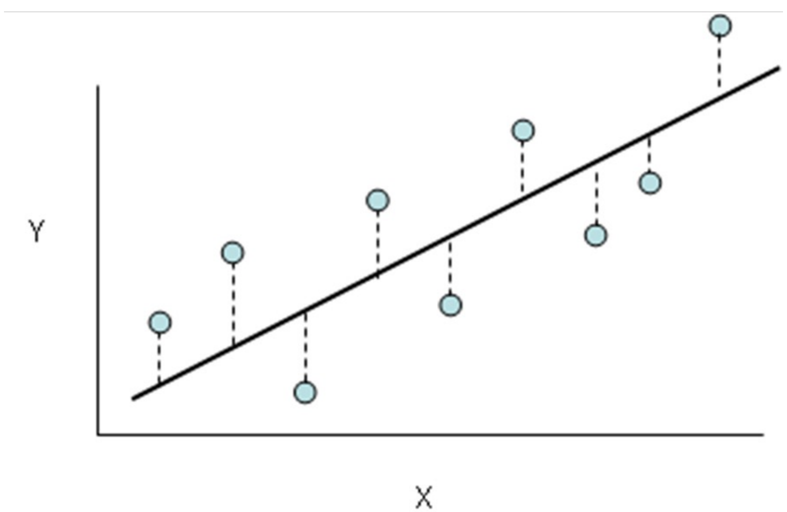
- A regression model provides a function that describes the relationship between one or more **independent variables** and a response, **dependent**, or target variable.

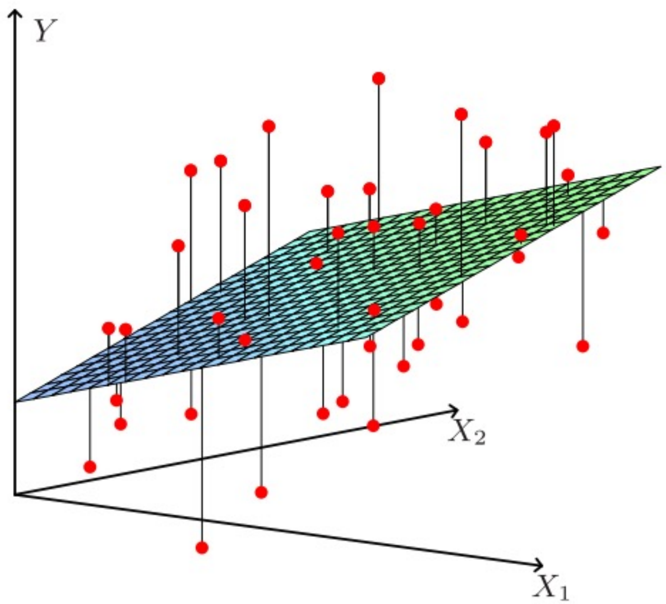


Linear Regression

A linear function for regression

$$y = f(x) = \boldsymbol{w}^T x + w_0$$

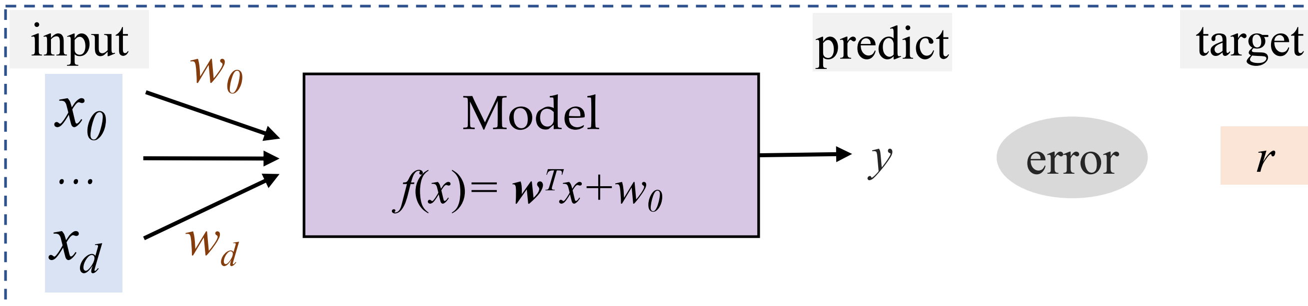




Linear model for regression is a (d+1)-dimensional hyperplane

Model Architecture

A simple linear function.



- Train:
  - estimate the parameters $\boldsymbol{w}$ and w_0 from data
- Test:
  - calculate $f(x) = \boldsymbol{w}^{\mathsf{T}} x + w_0$.

Loss Function

- For a given input x, the model outputs a real value y. Let $r \in \mathrm{R}$ be target value, the square error is :

$$l\,(\boldsymbol{w}, w_0 \,|\, x, r) = (r - y)^2$$

- Given: $D = \{(x^{(1)}, r^{(1)}), \ldots, (x^{(N)}, r^{(N)})\}$, the loss over the dataset is defined as the mean square error (MSE):

$$L(\boldsymbol{w}, w_0 \,|\, D) = \frac{1}{2N} \sum_{\ell=1}^{N} (r^{(\ell)} - y^{(\ell)})^2$$

这里的2据老师说只是为了之后求导好看一点。。。。

Optimization

Given: $D = \{(x^{(1)}, r^{(1)}), \ldots, (x^{(N)}, r^{(N)})\}$

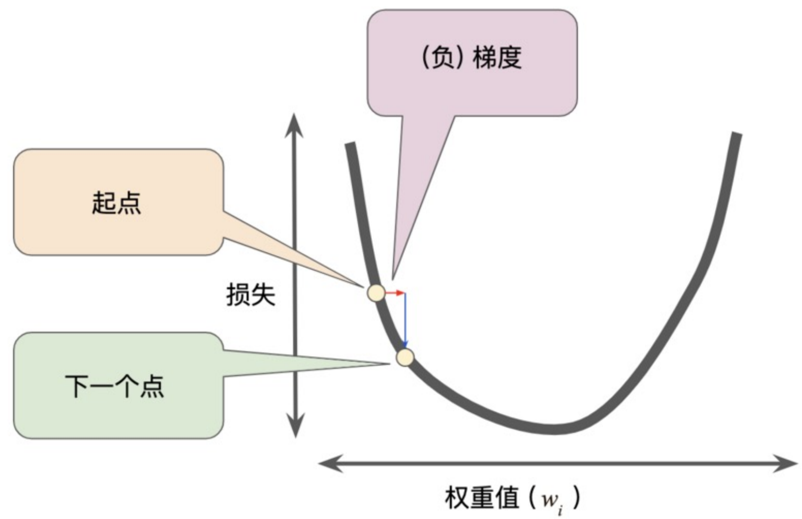
minimize the loss function using gradient descend:

- Goal:

$$\min_w L(\boldsymbol{w})$$

- Iteration:

$$\boldsymbol{w}_{t+1} = \boldsymbol{w}_t - \eta_t \frac{\partial L}{\partial w}$$



What is $\frac{\partial L}{\partial w}$? 梯度

Optimization – Gradient Descend

$$L(\boldsymbol{w}, w_0 \mid D) = -\,1/2\mathrm{N} \sum_{\ell=1}^{N} (r^{(\ell)} - y^{(\ell)})^2$$

What is $\frac{\partial L}{\partial w}$?

For each $\boldsymbol{w}_j$ ($j = 1, \ldots, d$):

此处将w作为自变量，y作为因变量，则将y=w*x+w_0两边求导就可以得到下面式子。

$$\frac{\partial L}{\partial w_j} = -\frac{1}{N} \sum_{\ell} \underbrace{\left(r^{(\ell)} - y^{(\ell)}\right) \frac{\partial y^{(\ell)}}{\partial w_j}}_{\text{Chain rule}} = -\frac{1}{N} \sum_{\ell} \left(r^{(\ell)} - y^{(\ell)}\right) x^{(\ell)}$$

$$\boldsymbol{w}_{\mathrm{new}} = \boldsymbol{w}_{\mathrm{old}} + \frac{1}{N} \sum_{\ell=1}^{N} (r^{(\ell)} - y^{(\ell)})\, x^{(\ell)}$$

The Algorithm

Gradient Descend for Liner Regression

Input: $D = \{(\mathrm{x}^{(l)}, \mathrm{r}^{(l)})\}$ $(l = 1{:}N)$

for $j = 0, \ldots, \mathrm{d}$

  $\boldsymbol{w}_{\mathrm{j}} \leftarrow rand\,(-0.01, 0.01)$

repeat

  for $j = 0, \ldots, d$

    $\Delta\boldsymbol{w}_{\mathrm{j}} \leftarrow 0$

  for $l = 1, \ldots, N$

    $y \leftarrow 0$

    for $j = 0, \ldots, d$

      $y \leftarrow y + \boldsymbol{w}_{\mathrm{j}} x_{\mathrm{j}}^{(l)}$

    $\Delta\boldsymbol{w}_{\mathrm{j}} \leftarrow \Delta\boldsymbol{w}_{\mathrm{j}} + (r^{(l)} - y) x_{\mathrm{j}}^{(l)}$

  $\Delta\boldsymbol{w}_{\mathrm{j}} = \Delta\boldsymbol{w}_{\mathrm{j}} / \mathrm{N}$

  for $j = 0, \ldots, d$

    $w_{\mathrm{j}} \leftarrow w_{\mathrm{j}} + \eta \Delta w_{\mathrm{j}}$

until convergence

The Matrix Form

$$\boldsymbol{X} = \begin{bmatrix} \boldsymbol{x}^{(1)} \\ \boldsymbol{x}^{(2)} \\ \vdots \\ \boldsymbol{x}^{(N)} \end{bmatrix} = \begin{bmatrix} x_0^{(1)} & x_1^{(1)} & x_2^{(1)} & \cdots & x_d^{(1)} \\ x_0^{(2)} & x_1^{(2)} & x_2^{(2)} & \cdots & x_d^{(2)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_0^{(N)} & x_1^{(N)} & x_2^{(N)} & \cdots & x_d^{(N)} \end{bmatrix} \qquad \boldsymbol{w} = \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_d \end{bmatrix} \qquad \boldsymbol{r} = \begin{bmatrix} r^{(1)} \\ r^{(2)} \\ \vdots \\ r^{(N)} \end{bmatrix}$$

- Prediction: $\boldsymbol{y} = \boldsymbol{X}\boldsymbol{w} = \begin{bmatrix} \boldsymbol{x}^{(1)}\boldsymbol{w} \\ \boldsymbol{x}^{(2)}\boldsymbol{w} \\ \vdots \\ \boldsymbol{x}^{(N)}\boldsymbol{w} \end{bmatrix}$
- Objective: $L(\boldsymbol{w}) = \frac{1}{2}(\boldsymbol{r}-\boldsymbol{y})^T(\boldsymbol{r}-\boldsymbol{y}) = \frac{1}{2}(\boldsymbol{r}-\boldsymbol{X}\boldsymbol{w})^T(\boldsymbol{r}-\boldsymbol{X}\boldsymbol{w})$

The Matrix Form

- Gradient

$$\frac{\partial L(\boldsymbol{w})}{\partial \boldsymbol{w}} = -\boldsymbol{X}^T(\boldsymbol{r} - \boldsymbol{X}\boldsymbol{w})$$

- Solution

$$\frac{\partial L(\boldsymbol{w})}{\partial \boldsymbol{w}} = 0 \Rightarrow \boldsymbol{X}^T(\boldsymbol{r} - \boldsymbol{X}\boldsymbol{w}) = 0$$

$$\Rightarrow \boldsymbol{X}^T\boldsymbol{r} = \boldsymbol{X}^T\boldsymbol{X}\boldsymbol{w}$$

$$\Rightarrow \boldsymbol{w}* = (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^{\mathrm{T}}\boldsymbol{r}$$

The Matrix Form

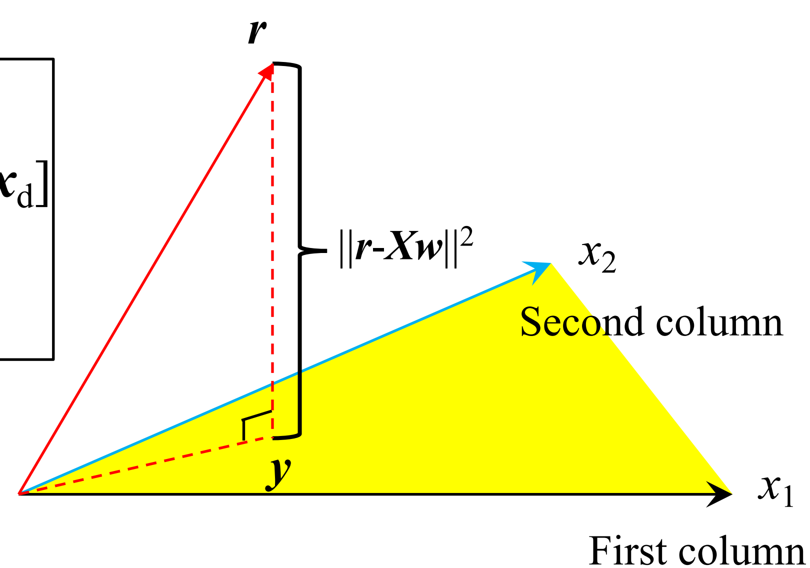
- Then the predicted values are

$$\boldsymbol{y} = \boldsymbol{X}(\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^{\mathrm{T}}\boldsymbol{r}$$
$$= \boldsymbol{H}\boldsymbol{r}$$

Geometrical Explanation

- The column vectors $[\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_d]$ form a subspace of $\mathbb{R}^n$.
- **H** is a least square projection

几何意义：偏差最小时即为x平面与偏差垂直的时候，此时y正好为r在x方向的分量。



Matrix Form with Regularization

奇异矩阵，行列式为0

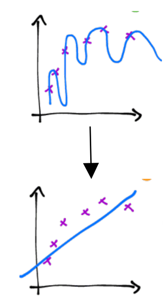
Problem: X^TX might be singular

When some column vectors are not independent (e.g., $\boldsymbol{x}_2=3\boldsymbol{x}_1$), then $\boldsymbol{X}^T\boldsymbol{X}$ is singular, thus $\boldsymbol{w}^* = (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{r}$ cannot be directly calculated.

奇异矩阵-->正则化

Solution: Regularization

$$L(\boldsymbol{w})=\frac{1}{2}(\boldsymbol{r}-\boldsymbol{y})^T(\boldsymbol{r}-\boldsymbol{y})=\frac{1}{2}(\boldsymbol{r}-\boldsymbol{X}\boldsymbol{w})^T(\boldsymbol{r}-\boldsymbol{X}\boldsymbol{w})+\frac{\lambda}{2}\|\boldsymbol{w}\|_2^2$$



New gradient: $$\frac{\partial L(\boldsymbol{w})}{\partial \boldsymbol{w}}=-\boldsymbol{X}^T(\boldsymbol{r}-\boldsymbol{X}\boldsymbol{w})+\lambda\boldsymbol{w}$$

New optimal solution: $$\frac{\partial L(\boldsymbol{w})}{\partial \boldsymbol{w}}=0 \Rightarrow -\boldsymbol{X}^T(\boldsymbol{r}-\boldsymbol{X}\boldsymbol{w})+\lambda\boldsymbol{w}=0$$

$$\Rightarrow \boldsymbol{X}^T\boldsymbol{r}=(\boldsymbol{X}^T\boldsymbol{X}+\lambda\mathbf{I})\boldsymbol{w}$$

$$\Rightarrow \boldsymbol{w}^*=(\boldsymbol{X}^T\boldsymbol{X}+\lambda\mathbf{I})^{-1}\boldsymbol{X}^T\boldsymbol{r}$$

Python Tutorial

https://colab.research.google.com/github/cs231n/cs231n.github.io/blob/master/python-colab.ipynb

https://www.kaggle.com/code/sudhirnl7/linear-regression-tutorial/data?select=insurance.csv

What’s Next?

Classifications

Find a decision boundary that maximizes the margin between two classes.

Machine Learning
- **Supervised Learning**
  - Regression
  - Classification(✓)
  - …
- Unsupervised Learning
- Reinforcement Learning