Regression Supervised Learning

講者：Isaac

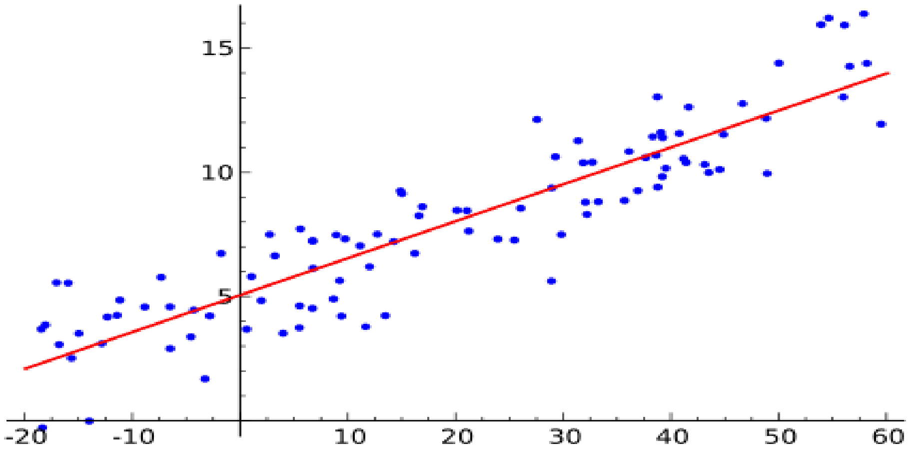
Outline

- Linear Regression
- Polynomial Regression
- Ridge, Lasso, ElasticNet

Linear Regression

What's Linear Regression

- A linear approach for modelling the relationship between a scalar dependent variable y and one or more independent variables X



Linear Regression

Size ($feet^2$)	Number of bedrooms	Number of floors	Age of home (years)	Price ($1000)
2104	5	1	45	460
1416	3	2	40	232
1534	3	2	30	315
852	2	1	36	178
...	...	...	...	...

How to predict house price?

Data normalization

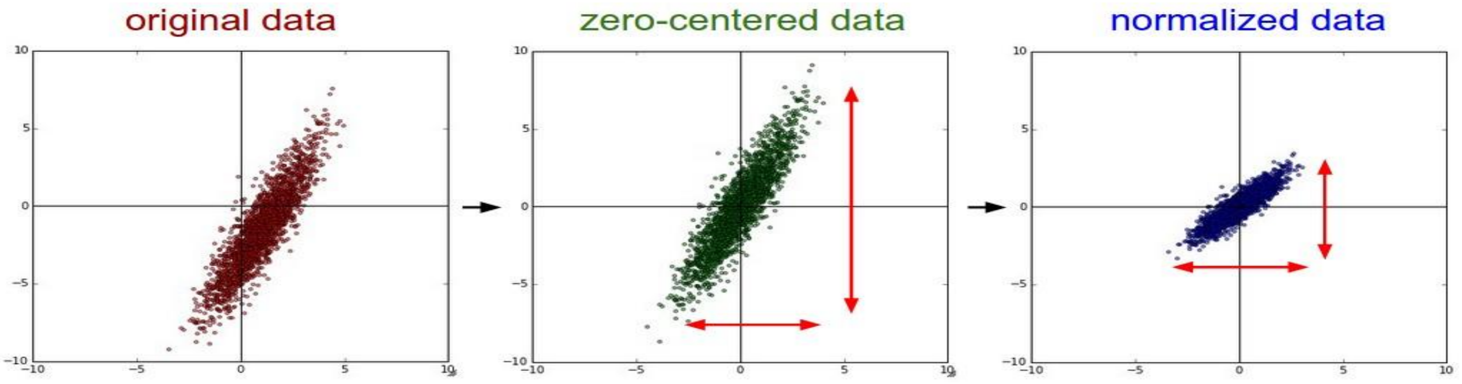
House price prediction

Size (feet²)	Number of bedrooms	Number of floors	Age of home (years)	Price ($1000)
2104	5	1	45	460
1416	3	2	40	232
1534	3	2	30	315
852	2	1	36	178
…	…	…	…	…

Order of magnitude is quite different

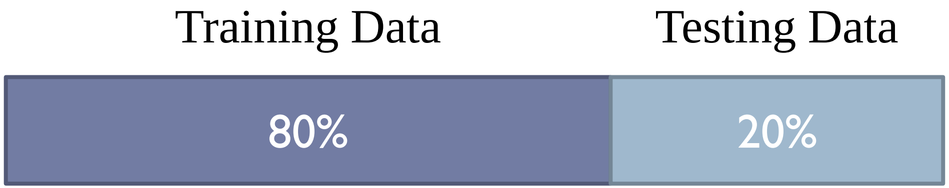
Data normalization

- Try to scale all features into [0, 1] or [-1, 1]
- Some common normalization method
  - $\frac{x_i - x_{min}}{x_{max} - x_{min}}$
  - $x_i - \mu$



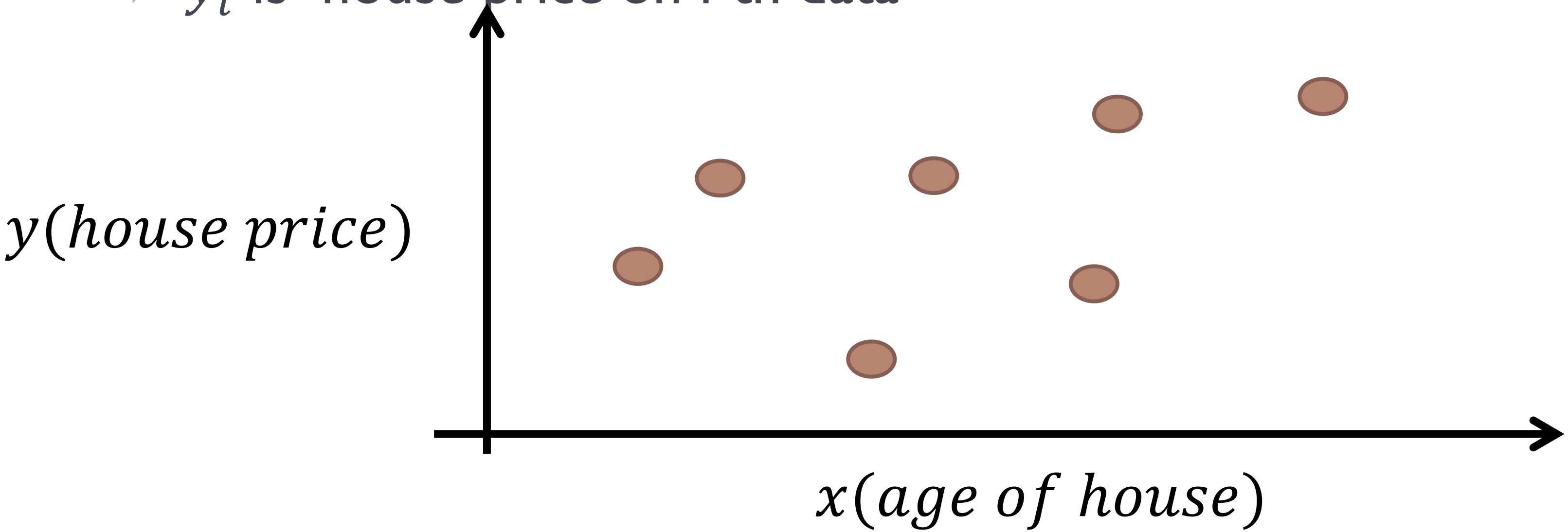
Split data

- Split total house price data into two part
  - 80% as training data
  - 20% as testing data
- Training data is used to make machine learn
- Testing data is used to validate if machine actually learn well



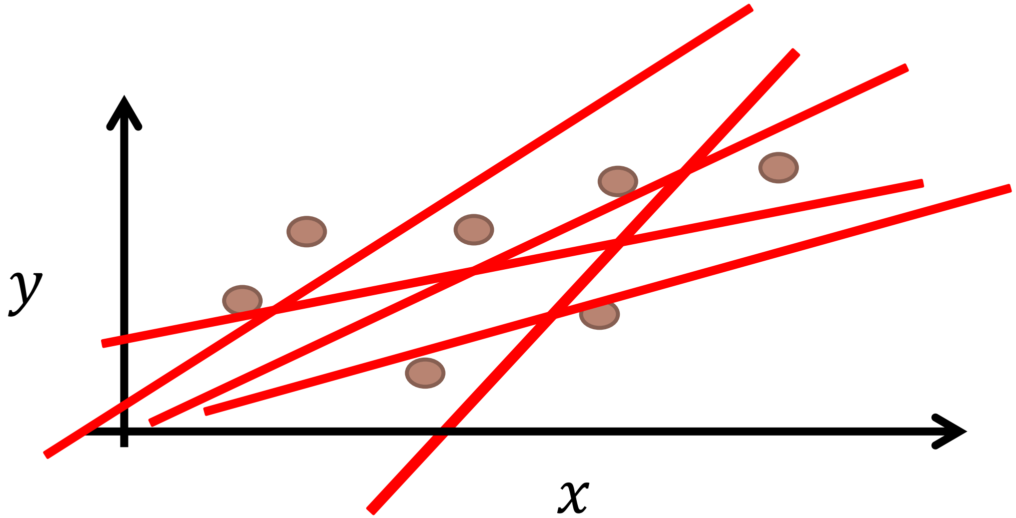
Simple regression case

- Assume all (x_i, y_i) pairs is as following
  - x_i is age of house on i-th data
  - y_i is house price on i-th data



Simple regression case

- We would like to find the best line that fit these data
  - One of famous solution is linear regression

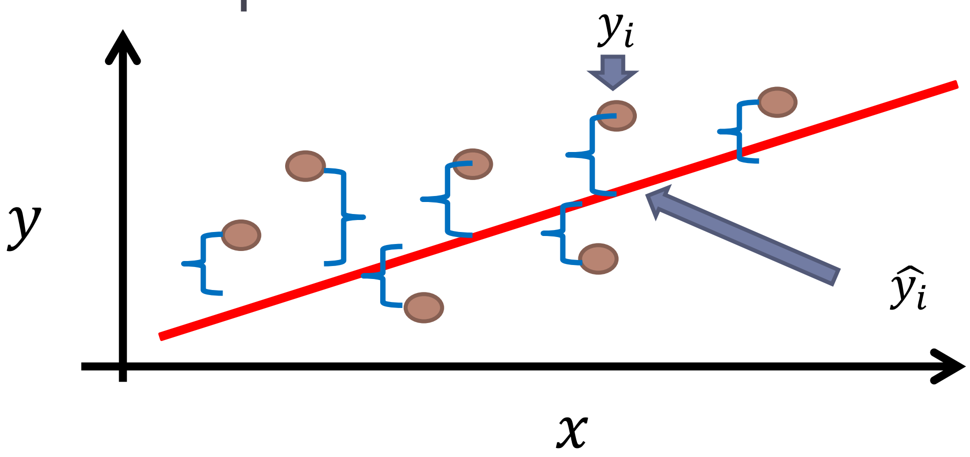


$$\hat{y} = \theta_1 x + \theta_0$$

Build model

Simple regression case

- Best fit' means difference between actual *y* values and predicted *y* values are a minimum
  - We usually use least squares which minimizes the sum of the squared differences



$$Cost = \sum_{i=1}^{m} (y_i - \hat{y}_i)^2$$

Define cost

Simple regression case

- We can use calculus to find that the best fit line

$$\hat{y} = \theta_1 x + \theta_0$$

$$Cost = \sum_{i=1}^{m} (y_i - \hat{y_i})^2$$

$$\Big\Downarrow \quad \textbf{Optimization}$$

$$\boxed{\begin{aligned} \theta_1{}^* &= \frac{\sum_{i=1}^{m} x_i y_i - \frac{\left(\sum_{i=1}^{m} x_i\right)\left(\sum_{i=1}^{m} y_i\right)}{m}}{\left(\sum_{i=1}^{m} x_i{}^2\right) - \frac{\left(\sum_{i=1}^{m} x_i\right)^2}{m}} \\ \theta_0{}^* &= \bar{y} - \theta_1{}^* \, \bar{x} \end{aligned}}$$

Best parameters

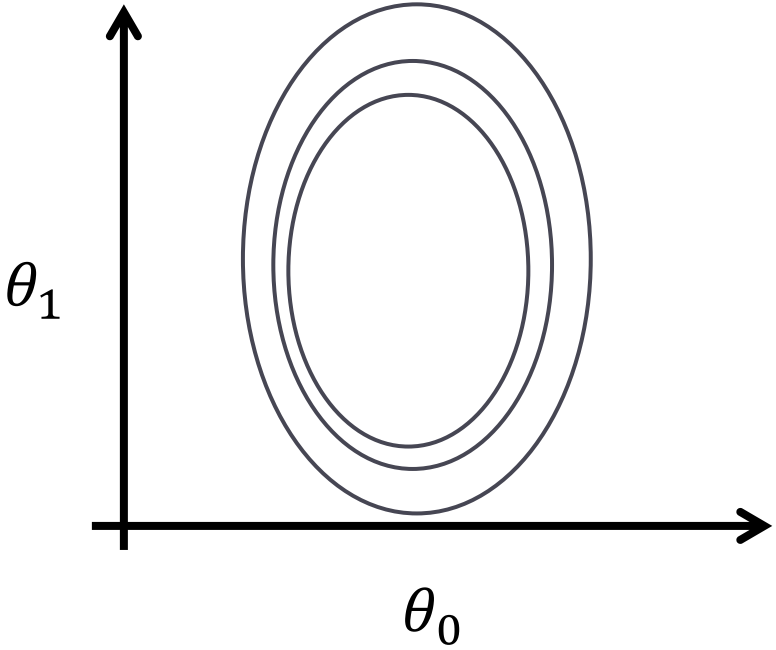
validate trained model - regression

- MSE
  - average of the squares of the errors/deviations(difference between the estimator and what is estimated)
- R squared(coefficient of determination)
  - proportion of the variance in the dependent variable that is predictable from the independent variable
  - R squared is close to 1 mean that the model is fitting better

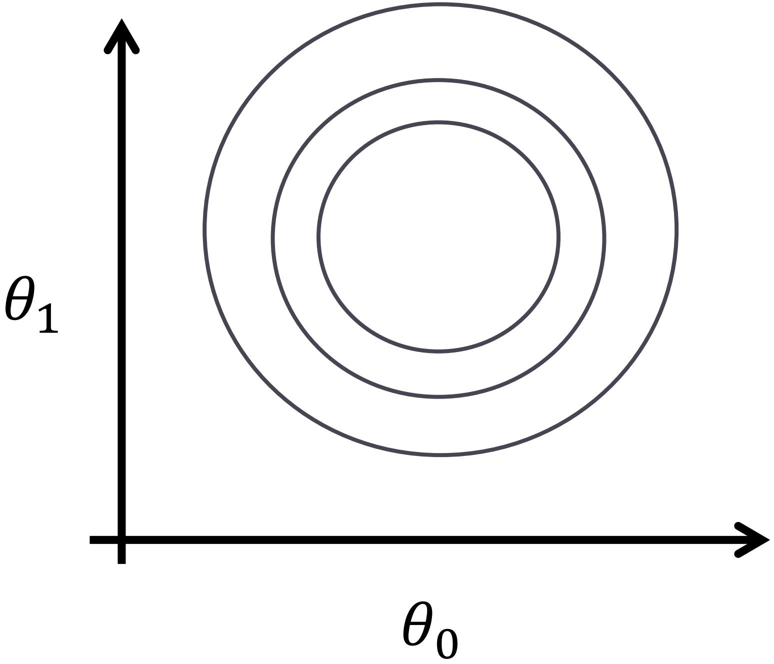
$$MSE = \frac{1}{n}\sum_{i=1}^{n}(f_i - \hat{y}_i)^2$$

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(f_i - \bar{y})^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}$$

Importance of data normalization



without data normalization



with data normalization

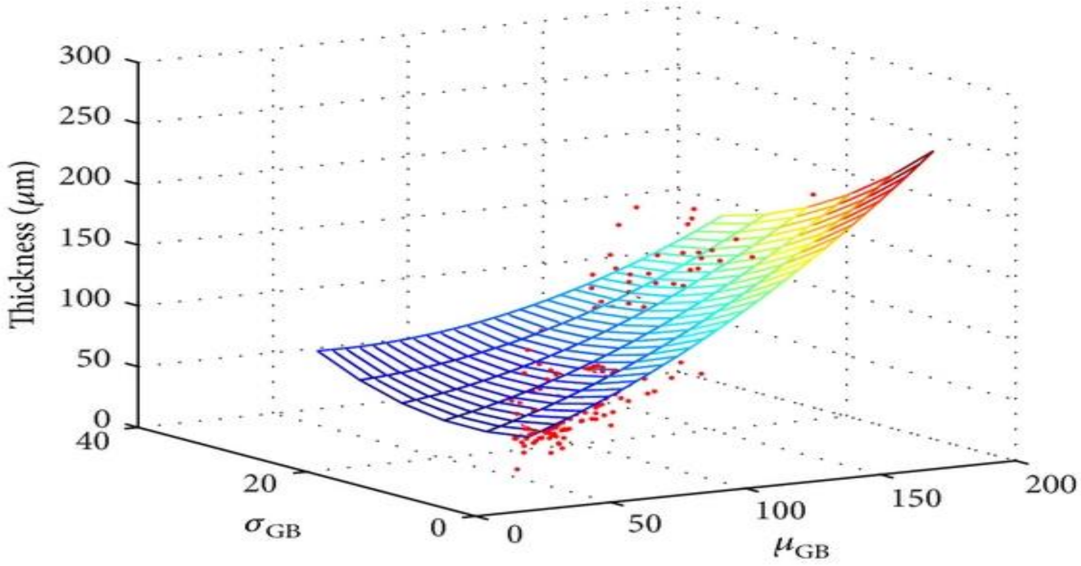
Multivariate Linear Regression Models

$$Model\colon\ h_\theta = \theta^T X = \theta_0 x_0 + \theta_1 x_1 + \cdots + \theta_n x_n$$

$$Learned\ Parameters\colon\ \theta_0, \theta_1, \ldots, \theta_n$$

$$Cost\ Function\colon C(\theta_0, \theta_1, \ldots, \theta_n) = \frac{1}{2m}\sum_{i=1}^{m}(h_\theta\left(x^{(i)}\right) - y^{(i)})^2$$

Multivariate Linear Regression Models

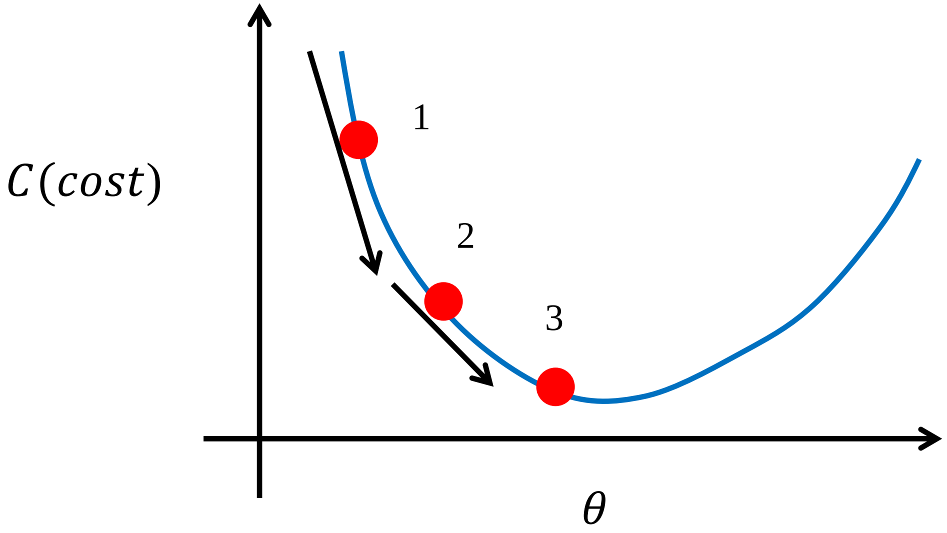


$Best\ \theta$:

$$\theta = (X^T X)^{-1} X^T y$$

Gradient Descent

- An algorithm that find the minimum of a function



$Randomly\ select\ \theta_1 as\ start\ point$

$$Compute\ \frac{dC(\theta_1)}{d\theta}$$

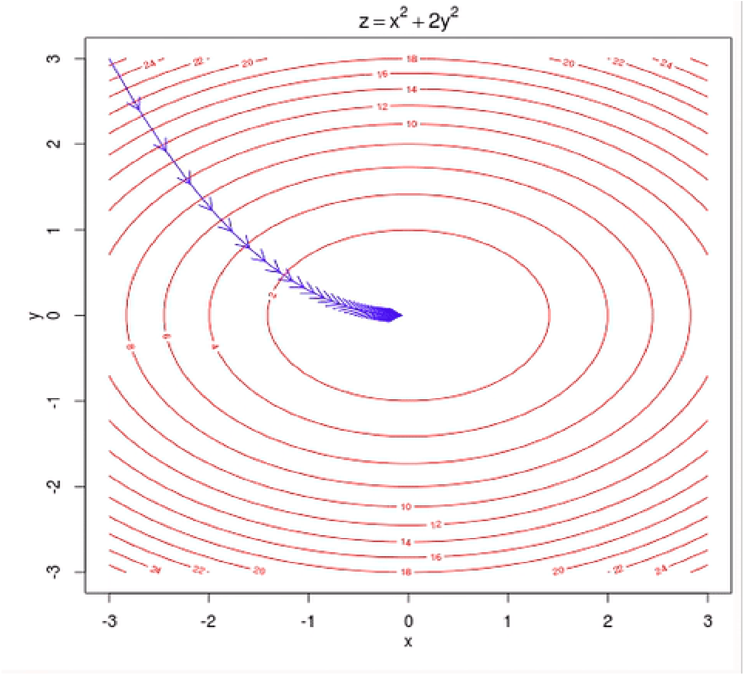
$$\theta_2 \leftarrow \theta_1 - \eta \frac{dC(\theta_1)}{d\theta}$$

$$Compute\ \frac{dC(\theta_2)}{d\theta}$$

$$\theta_3 \leftarrow \theta_2 - \boxed{\eta} \frac{dC(\theta_2)}{d\theta}$$

Learning rate

Gradient Descent



$$\theta = \begin{bmatrix} x \\ y \end{bmatrix} \qquad \nabla C(\theta) = \begin{bmatrix} dz/dx \\ dz/dy \end{bmatrix}$$

$Randomly\ select\ \theta_1 as\ start\ point$

$Compute\ \nabla C(\theta_1)$

$\theta_2 \leftarrow \theta_1 - \eta \nabla C(\theta_1)$

$Compute\ \nabla C(\theta_2)$

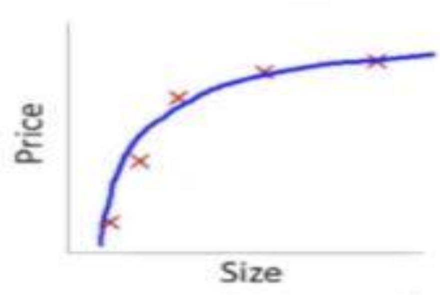
$\theta_3 \leftarrow \theta_2 - \eta \nabla C(\theta_2)$

Gradient Descent V.S. Normal Equation

Gradient Descent	Normal Equation
• Need to choose learning rate • Need many iterations • Works well when n is large	• No need to choose learning rate • Need to compute matrix inverse • If n is large, it is very slow

Overfitting

$\theta_0 + \theta_1 x$



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

$\theta_0 + \theta_1 x + \theta_2 x^2 + \cdots + \theta_l x^l$

Too many variable would cause Overfitting

Example and Practice

- Example
  - linear regression
    - example/regression
- Practice
  - Try to use linear regression to predict house price
    - dataset/house.csv
    - practice/regression
  - More information about the dataset
    - https://www.kaggle.com/c/boston-housing

Polynomial Regression

Polynomial Regression

House price prediction

Size ($feet^2$)	Price ($1000)
2104	460
1416	232
1534	315
852	178
...	...

Assume house price is only dependent on house size

(consider single variable first)

nth-degree Polynomial Regression

house size

$$Model:\ h_\theta = \theta_0 + \theta_1 {x_0}^1 + \cdots + \theta_n {x_0}^n$$

$$Learned\ Parameters:\ \theta_0, \theta_1, \ldots, \theta_n$$

$$Cost\ Function: C(\theta_0, \theta_1, \ldots, \theta_n) = \frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2$$

Polynomial Regression

- Different models

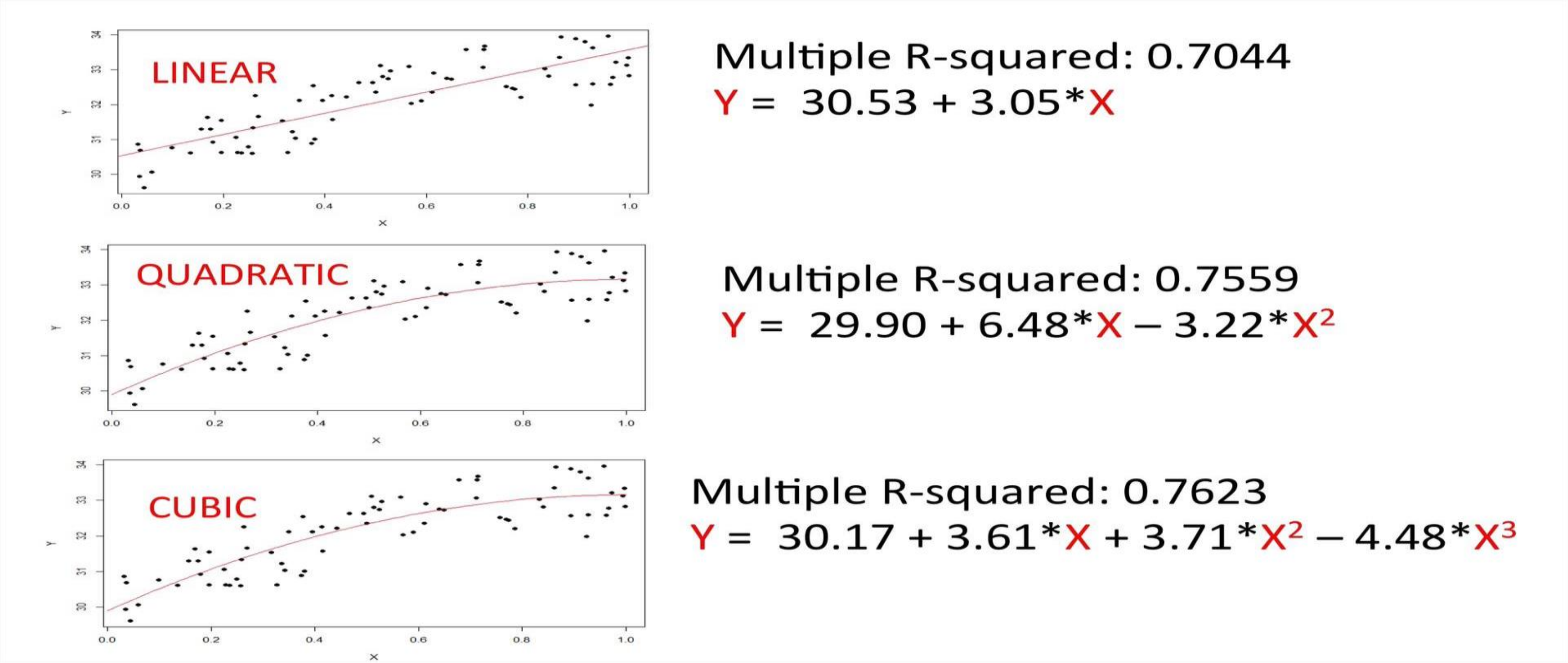
$$Model\colon\ h_\theta = \theta_0 + \theta_1 {x_0}^1 + \cdots + \theta_n {x_0}^n$$

$$\vdots$$

$$Model\colon\ h_\theta = \theta_0 + \theta_1 {x_0}^1 + \theta_2 \sqrt{x_0}$$

$$\vdots$$

Polynomial Regression



Polynomial Regression

multiple variable in polynomial regression
(take two variable for example)

house size

\# room

$$Model:\ h_\theta = \theta_0 + \theta_1 x_0 + \theta_{11} {x_0}^2 + \theta_2 x_1 + \theta_{22} {x_1}^2$$
$$+\theta_{12} \boxed{x_0 x_1}$$

cross term

Polynomial Regression

we can think that
we use five different attributes($x_0, x_1, {x_0}^2, {x_1}^2, x_0 x_1$) to do linear regression

$$Model\colon\ h_\theta = \theta_0 + \theta_1 x_0 + \theta_{11} {x_0}^2 + \theta_2 x_1 + \theta_{22} {x_1}^2 + \theta_{12} x_0 x_1$$

Example and Practice

- Example
  - polynomial regression
    - example/regression
- Practice
  - Try to use linear regression to predict wine quality
    - dataset/winequality-red.csv
    - practice/regression
  - More information about the dataset
    - https://archive.ics.uci.edu/ml/datasets/wine+quality

Ridge, Lasso, ElasticNet

Lasso Regression

- Lasso (least absolute shrinkage and selection operator) is a regression analysis method
  - perform both variable selection and regularization

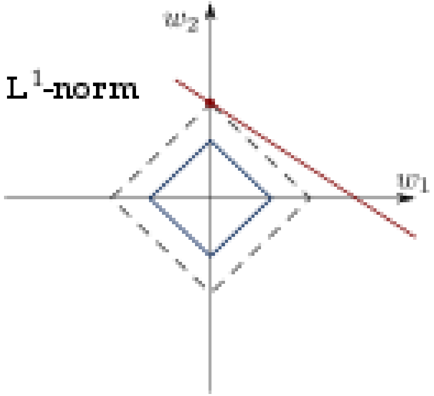
$$Model\colon\ h_\theta = \theta^T X = \theta_0 x_0 + \theta_1 x_1 + \cdots + \theta_n x_n$$

$$Learned\ Parameters\colon\ \theta_0, \theta_1, \dots, \theta_n$$

$$Cost\ Function\colon C(\theta_0, \theta_1, \dots, \theta_n) = \frac{1}{2m}\sum_{i=1}^{m}\left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right)^2 + \lambda \sum_{j=1}^{n}\left|\theta_j\right|$$

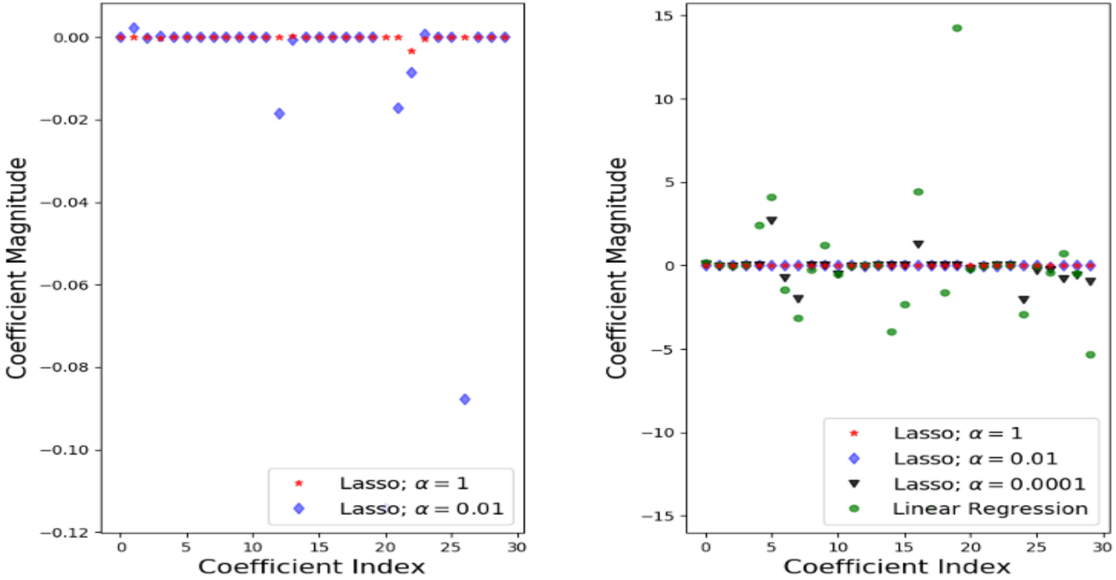
Lasso Regression

$$Cost\ Function: C(\theta_0, \theta_1, \dots, \theta_n) = \frac{1}{2m}\sum_{i=1}^{m}\left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right)^2 + \lambda\sum_{j=1}^{n}\left|\theta_j\right|$$



some of the features are completely neglected for the evaluation of output(feature selection)

Different parameters under Lasso



ref: https://towardsdatascience.com/ridge-and-lasso-regression-a-complete-guide-with-python-scikit-learn-e20e34bcbf0b

Ridge regression

- Ridge is a regression analysis method which add L2 regularization to cost function

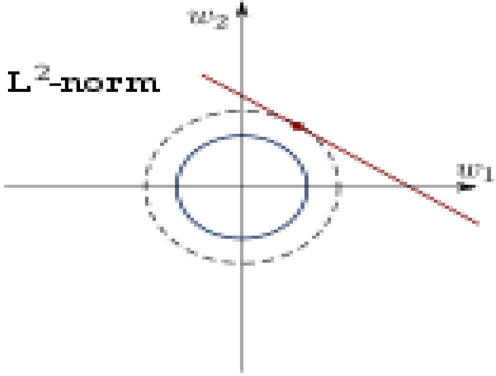
$$Model\colon\ h_\theta = \theta^T X = \theta_0 x_0 + \theta_1 x_1 + \cdots + \theta_n x_n$$

$$Learned\ Parameters\colon\ \theta_0, \theta_1, \dots, \theta_n$$

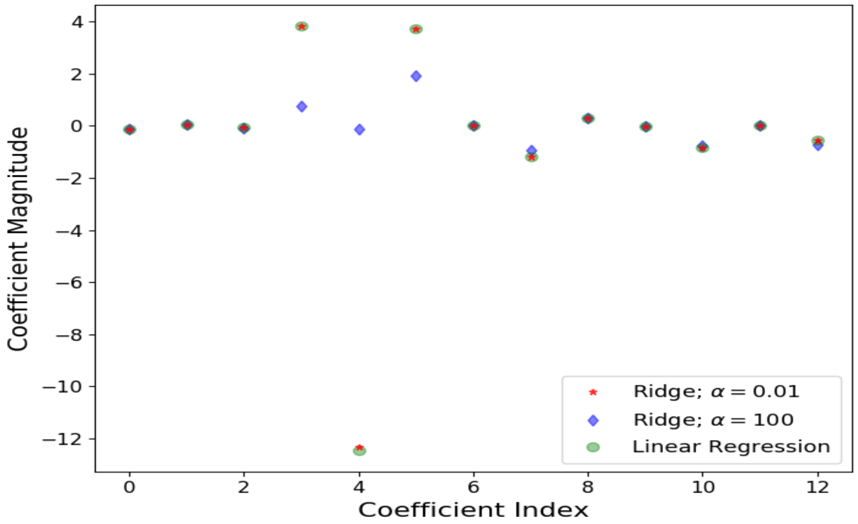
$$Cost\ Function\colon C(\theta_0, \theta_1, \dots, \theta_n) = \frac{1}{2m}\sum_{i=1}^{m}(h_\theta\left(x^{(i)}\right) - y^{(i)})^2 + \lambda\sum_{j=1}^{n}(\theta_j)^2$$

Ridge regression

$$Cost\ Function: C(\theta_0, \theta_1, \ldots, \theta_n) = \frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2 + \lambda\sum_{j=1}^{n}(\theta_j)^2$$



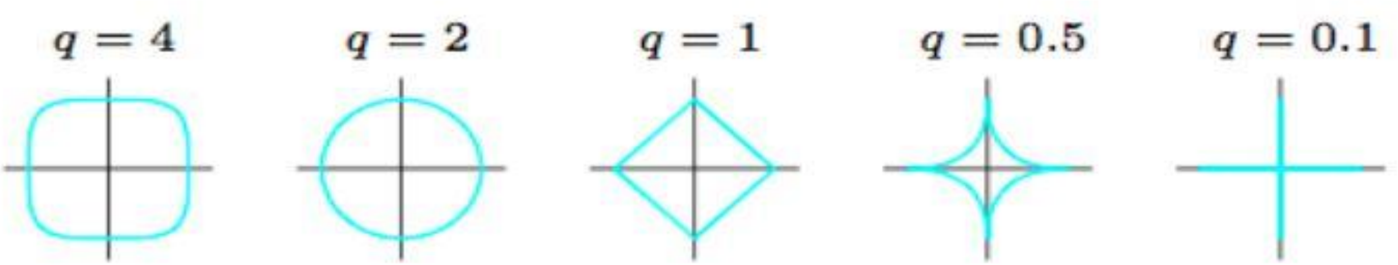
Different parameters under Ridge



ref: https://towardsdatascience.com/ridge-and-lasso-regression-a-complete-guide-with-python-scikit-learn-e20e34bcbf0b

Different Norm Constraint

Generalize to L_q $norm$: $||w||^q$



Elastic net

- Elastic net is a regularized regression method that linearly combines the L1 and L2 penalties of the lasso and ridge methods.

$$Model:\ h_\theta = \theta^T X = \theta_0 x_0 + \theta_1 x_1 + \cdots + \theta_n x_n$$

$$Learned\ Parameters:\ \theta_0, \theta_1, \ldots, \theta_n$$

$$Cost\ Function: C(\theta_0, \theta_1, \ldots, \theta_n)$$

$$= \frac{1}{2m} \sum_{i=1}^{m} (h_\theta\left(x^{(i)}\right) - y^{(i)})^2 + \lambda_1 \sum_{j=1}^{n} \left|\theta_j\right| + \lambda_2 \sum_{j=1}^{n} (\theta_j)^2$$

Constrain in Ridge Lasso Elastic Net