GBDi
Government Big Data Institute
สถาบันส่งเสริมการวิเคราะห์และบริหารข้อมูลขนาดใหญ่ภาครัฐ (สวข.)

depa

Introduction to Supervised Learning: Linear Regression

Papoj Thamjaroenporn

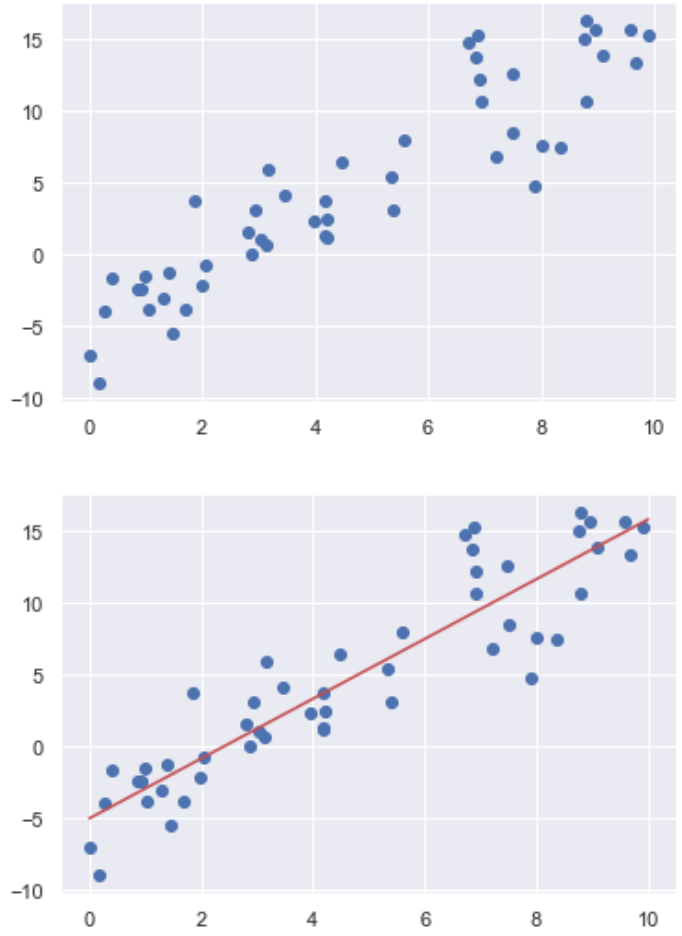
Original materials developed by Patipan Prasertsom

GBDi

Notebooks and Data

https://bit.ly/3z6Geeo

Regression



- Just like many other supervised learning tasks, regression task attempt to unravel the relationship between input features and target output.
- The model can then predict a numeral value of a target output when given new input data.
- For linear regression, a linear equation is used as model to be represent said relationships by attempting to find the most appropriate weights for the equation

$$y = w_0 + w_1 x_1 + w_2 x_2 + \cdots + w_m x_m$$

Linear function review

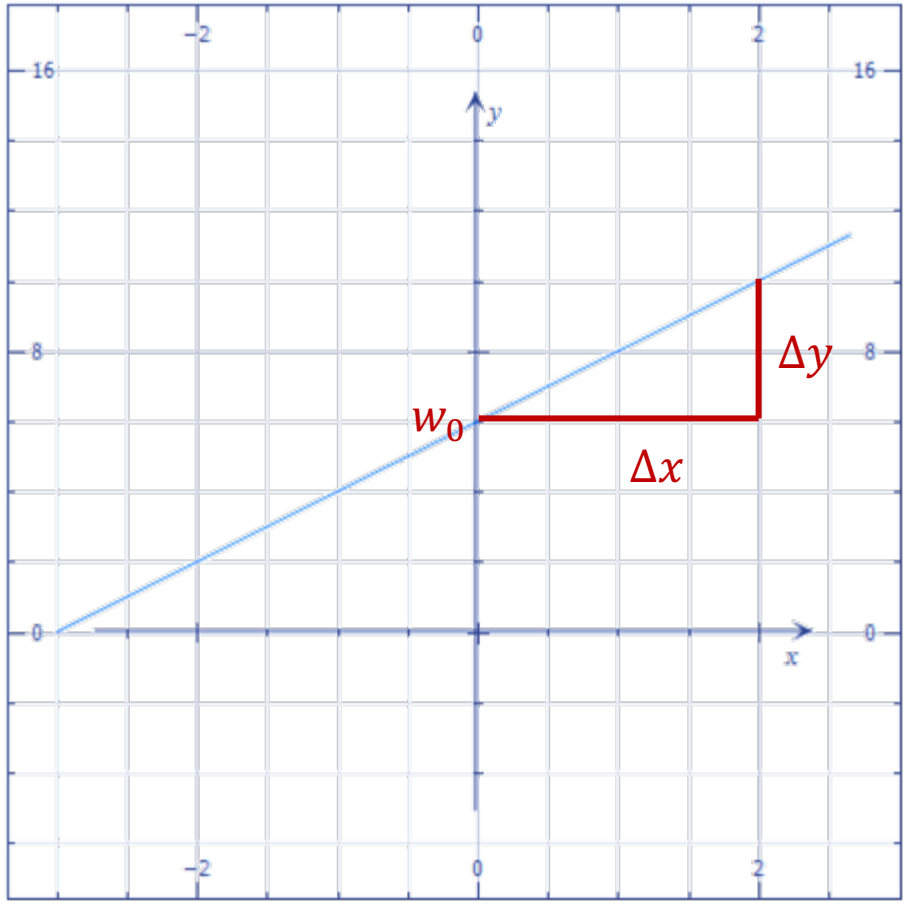
- Recall that in the case of 1 variable, a linear equation has the form of

$$y = w_0 + w_1 x_1$$

- Here, w_1 represent the gradient (slope) of the line, which is the rate of change in y with respect to the change in x

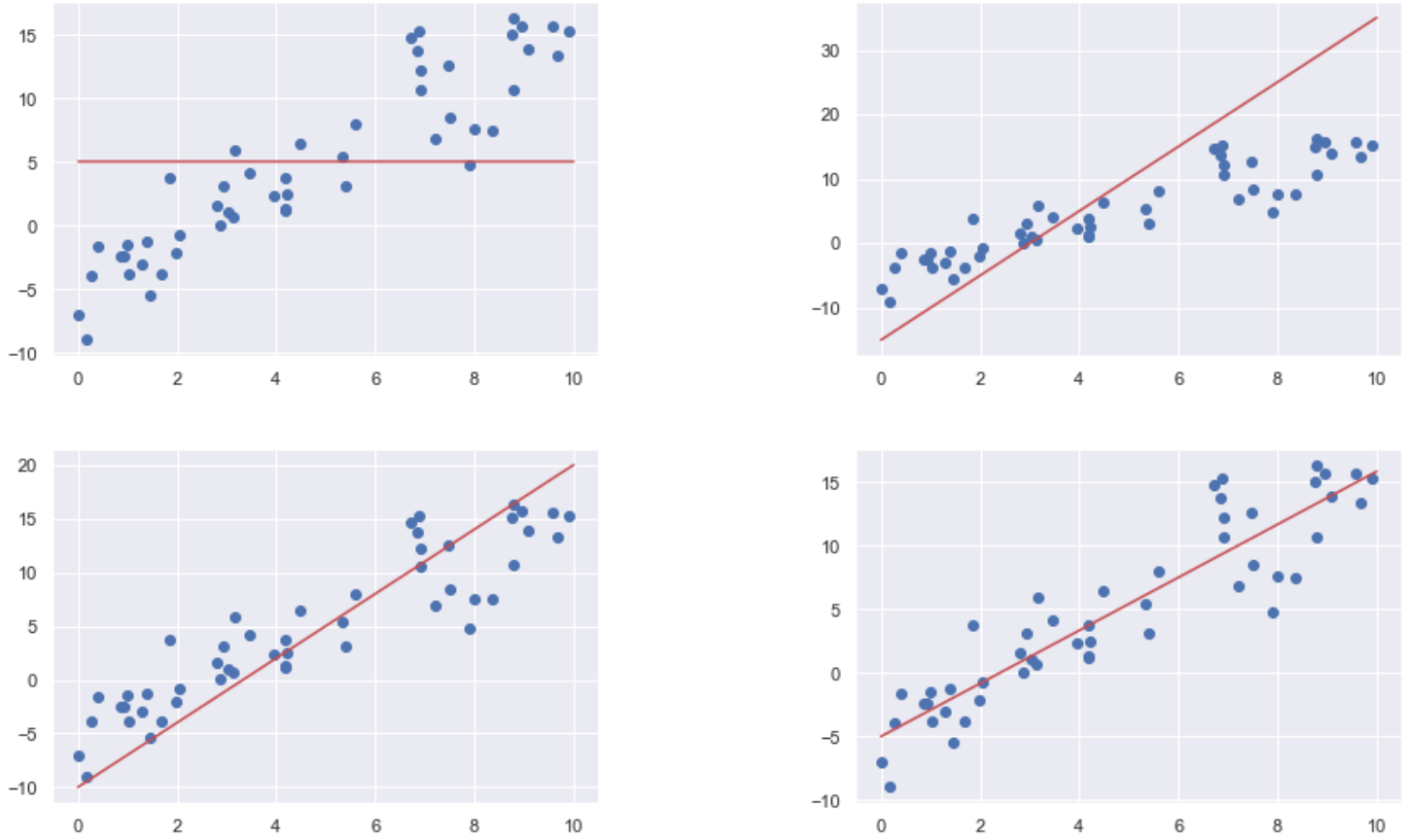
$$w_1 = \frac{\Delta y}{\Delta x}$$

- w_0 represent the y-intercept (bias)



Figures from Adrien1018 (https://commons.wikimedia.org/wiki/File:Linear_equation.png)

Multiple Curve Fitting Options



- Which line fits the best among these 4 lines? How do we compare them?

Error

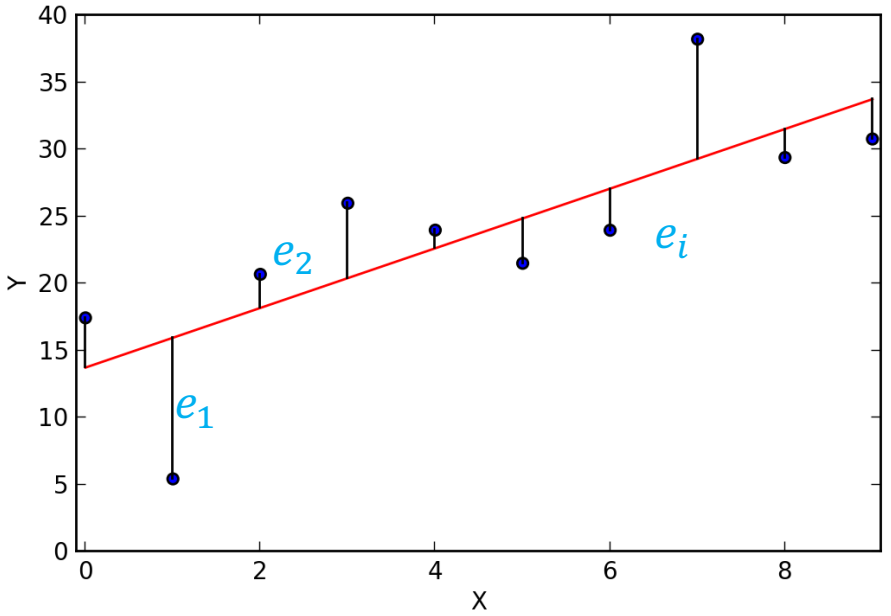
- For each data point, the error of a prediction is

$$e^{(i)} = \hat{y}_i^{(i)} - y^{(i)}$$

- In order to evaluate the overall error of a prediction model, Mean Square Error (MSE) is utilized.

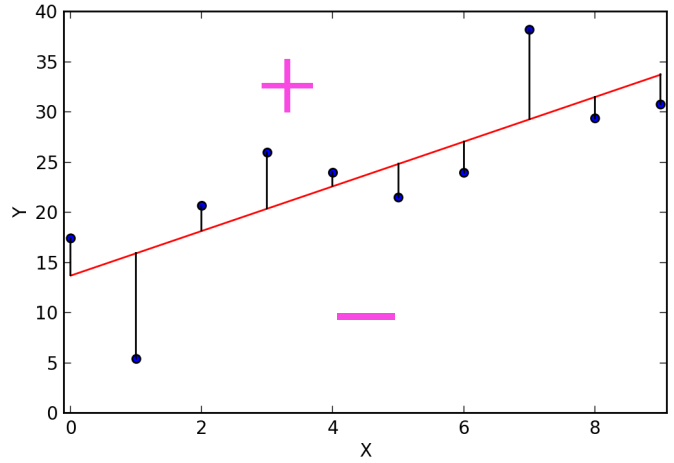
$$MSE = \frac{1}{m}\sum_{i=1}^{m}\left(\hat{y}_i^{(i)} - y^{(i)}\right)^2$$

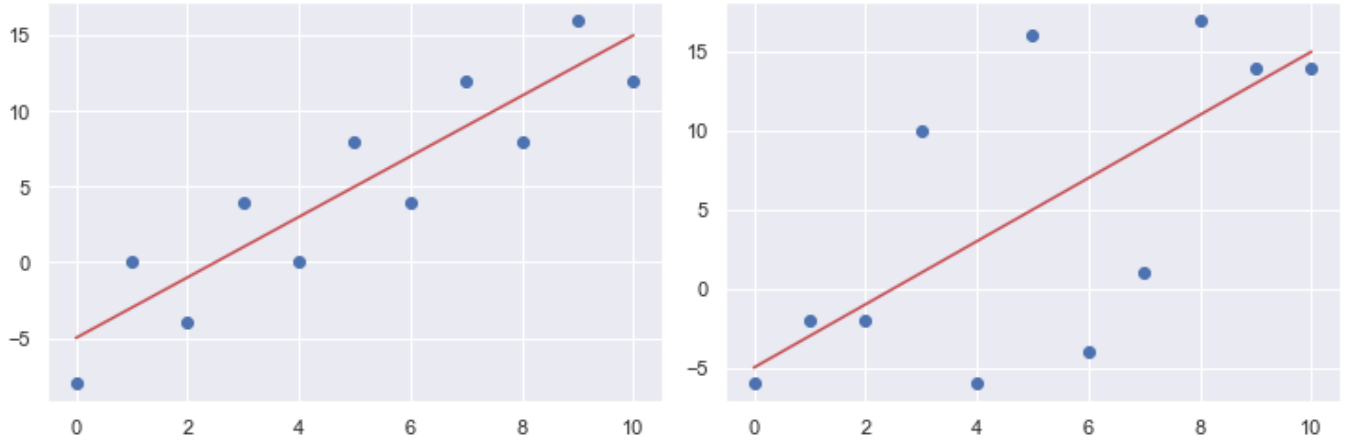
- If we want the overall error of the model, why don't we just sum/average the errors of all points?



Figures from Thomas.haslwanter (https://commons.wikimedia.org/wiki/File:Residuals_for_Linear_Regression_Fit.png)

Why MSE?





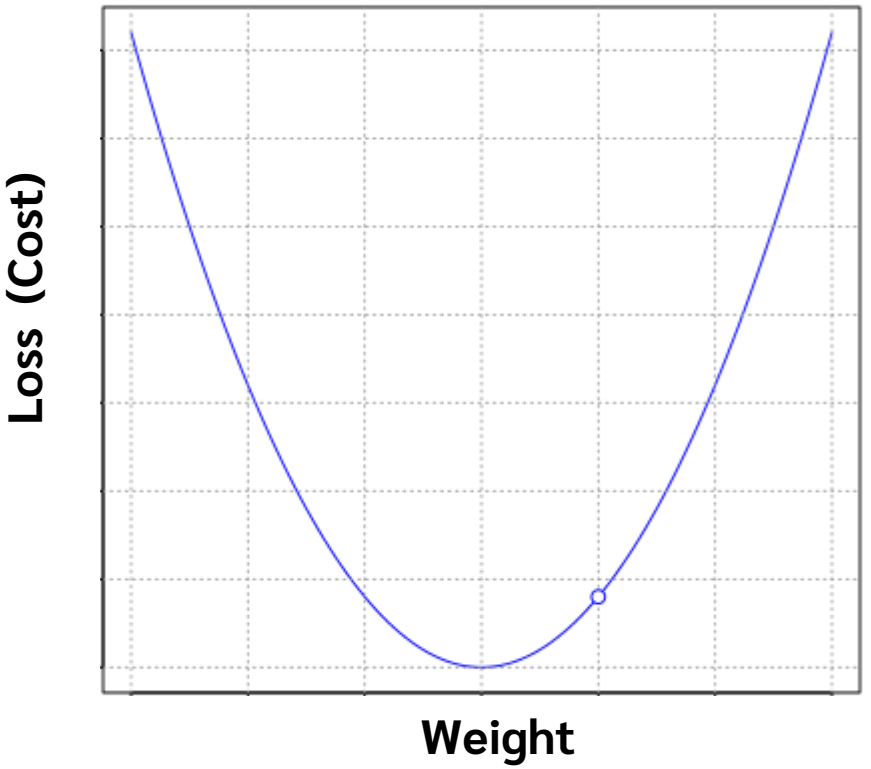
- The sum of errors will cancel each other!
- However, if we only want the errors to be positive,

why don't we simply use absolute value of errors?

- Which one of the line fits better?
- In addition to making errors positive, MSE does not treat an increase in errors linearly in scale
  - Specifically, by squaring the error, it heavily punish the points that are further from the predicted values

Adjusting Weights

- A goal of a linear regression (and most supervised machine learning models) is to find the mathematical model that best represent the relationship between input features and target output.
- In order to do so, we define a "**loss function**" (or cost function) that we're trying to minimize.
- For linear regression, a commonly used loss function is **MSE** (with modified constant) i.e. $\text{Loss} = \frac{1}{m}\sum_{i=1}^{m}\left(\hat{y}_i{}^{(i)} - y^{(i)}\right)^2$
  - This means the model will adjust its weights to minimize overall error
- How would the algorithm know which way to adjust the weights so the model can reach the minimum error?



Solving Regression Problem Numerically: Gradient descent

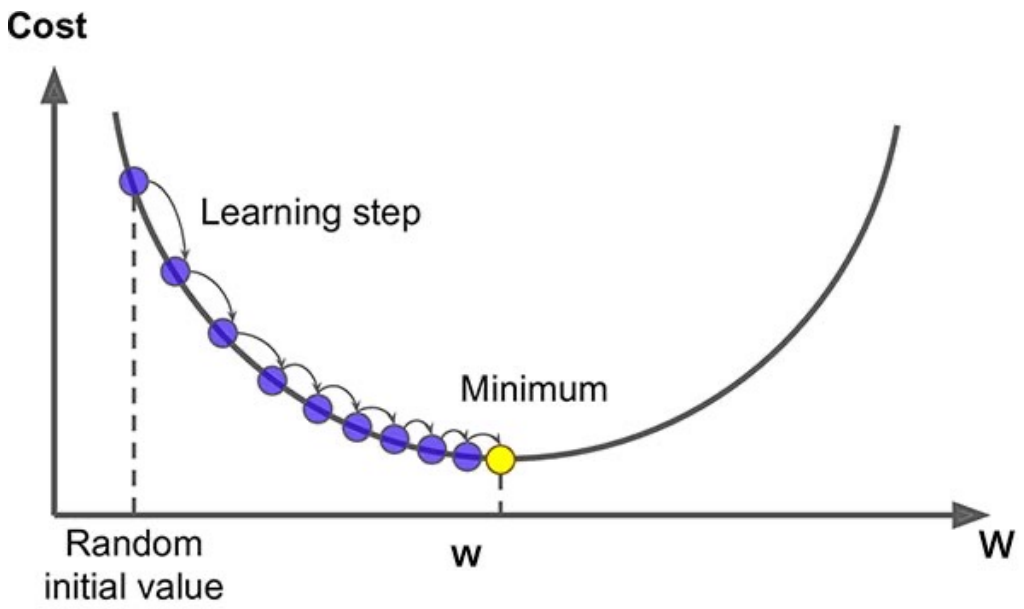
- The *gradient* (slope) of the cost function with respect to each weight at each given point informs the algorithm of the direction it should adjust the weight.
- For each iteration, the parameter is adjusted toward minimizing the loss function

$$w_i = w_i - \alpha \frac{\partial Loss}{\partial w_i}$$

New Weight

Old Weight

Learning Rate: A constant defined to determine how fast the weight is adjusting

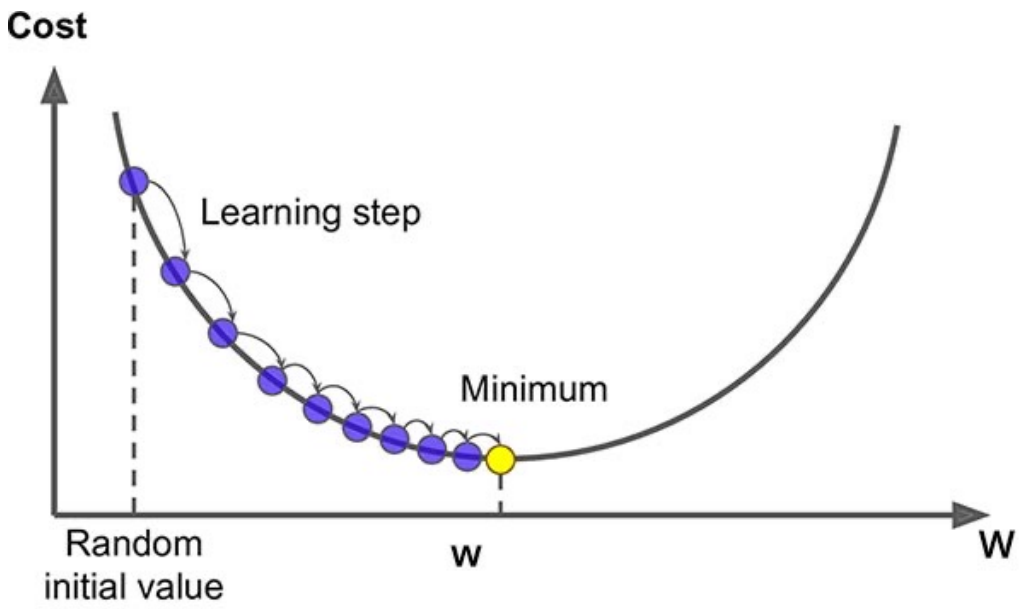


Figures by Saugat Bhattarai (https://saugatbhattarai.com.np/what-is-gradient-descent-in-machine-learning/)

Gradient (slope): The higher the magnitude, the further away from the minimum. Therefore, the algorithm will try to adjust quicker

Solving Regression Problem Numerically: Gradient descent

- When do we stop adjusting the weight?
  - The loss function improvement is below threshold (converged)
  - Reached an iteration limit.
- In machine learning library, parameters such as learning rate and iteration count are configurable
- Smaller learning rate can result in model training taking a long time
- Large learning rate can result in oscillating performance



Figures by Saugat Bhattarai (https://saugatbhattarai.com.np/what-is-gradient-descent-in-machine-learning/)

Multiple Linear Regression (MLR)

- While the examples of graphs presented so far are showing only one feature, the gradient descent algorithm applies updates to all the weights simultaneously inside the equation in each iteration.

$$y = w_0 + w_1 x_1 + w_2 x_2 + \cdots + w_m x_m$$

$$w_i = w_i - \alpha \frac{\partial Loss}{\partial w_i}$$

- In many cases, there are multiple relevant features that can (and should) be used to train a regression model.
- However, having multiple input features introduces more factors that may affect the performance such as
  - Variety of features' scales
  - The dependencies among features

Feature Scaling

- If the features of are not on similar scale, the process of gradient descent might take a long time before reaching convergence as the value of the cost function might end up oscillating.
- One of the common practices is to get features to be approximately within the range of -1 and 1

  ⇒In order to do so, we can first apply **mean normalization** (replacing x_i with $x_i - \mu_i$) to center data at 0

  ⇒Then we can perform normalization techniques like **mean normalization** or **standardization** to scale down the size of feature.
- In other words,

Mean normalization $x'_i = \dfrac{x_i - \overline{x_i}}{\max(x_i) - \min(x_i)}$ **Standardization** $x'_i = \dfrac{x_i - \overline{x_i}}{S_i}$

- There are functions available in the popular machine learning libraries to accomplish this.

(Optional) Solving Regression Analytically: Normal Equation

Constant (x0)	Feature 1 (x1)	Feature 2 (x2)	Feature 3 (x3)	Feature 4 (x4)	Target (y)
1	25	7	789	1	413
1	42	1	546	0	217
1	14	4	420	0	921
…	…	…	…	…	…
1	9	5	312	1	135

- Using linear algebra knowledge, the above equations can be written as $y = Xw$, where

$$X = \begin{bmatrix} 1 & 25 & 7 & 789 & 1 \\ 1 & 42 & 1 & 546 & 0 \\ 1 & 14 & 4 & 420 & 0 \\ \dots & \dots & \dots & \dots & \dots \\ 1 & 9 & 5 & 312 & 1 \end{bmatrix} \quad w = \begin{bmatrix} w_0 \\ w_1 \\ w_2 \\ w_3 \\ w_4 \end{bmatrix} \quad y = \begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ \dots \\ y_m \end{bmatrix}$$

Input features **Weights** **Target results**

- We can then find the value of weights by using

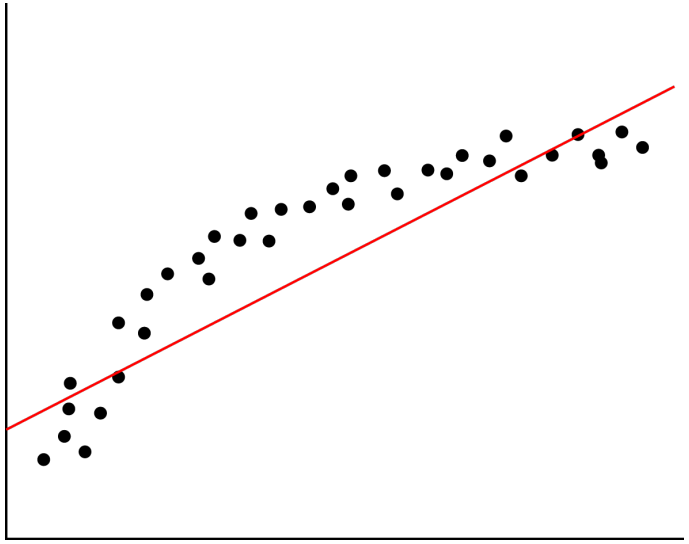
$$w = (X^T X)^{-1} X^T y$$

This method is slow if there are large amount of information

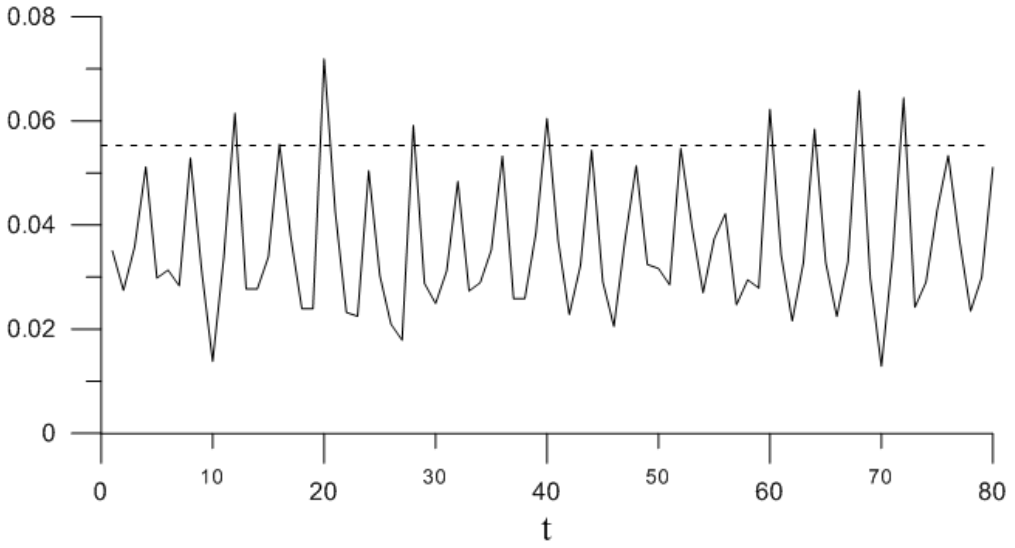
Linear Regression Assumptions

- **L**inearity – The relationship between input features and output should be linear
  - If the relationship is non-linear, higher degree model or non-linear transformation should be applied.
- **I**ndependence – There should be no correlation among error terms of data points (no autocorrelation)
  - If there is autocorrelation, using time series models might be more appropriate.
- **N**ormality – The error terms should be normally distributed
  - If the distribution is not normal, then there are abnormal data points. Handling outliers and perform non-linear transformation might help.
- **E**qual Variance
  - Generally non-constant variance occurs when there are outliers or extreme leverage value. Non-linear transformation might help.
- Multicollinearity – The input features should not be correlated

Linear Regression Assumptions



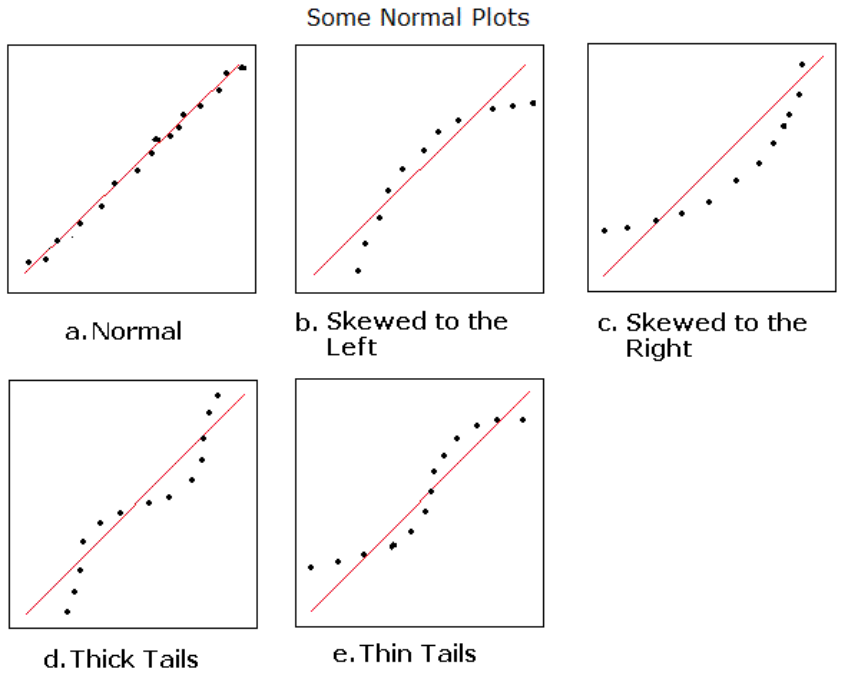
Non-Linearity



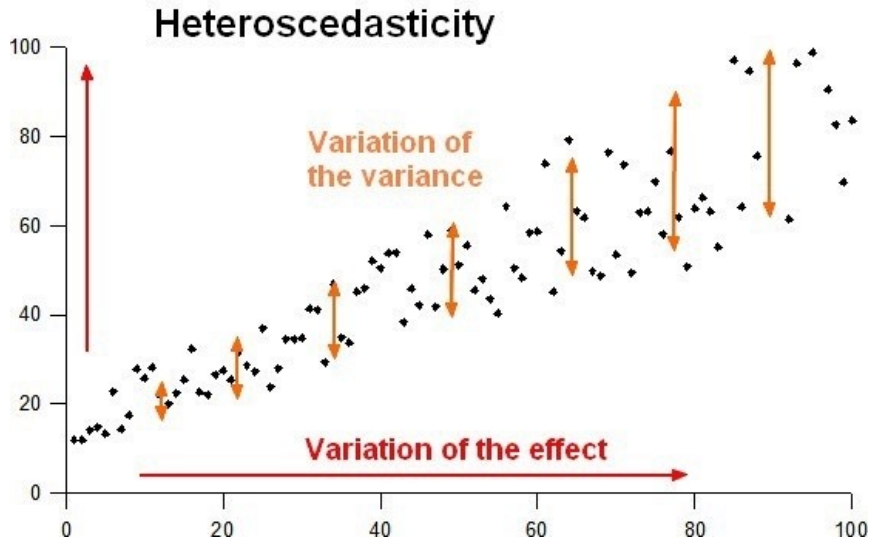
Autocorrelation

Figures from Mike de Waard (https://xyclade.github.io/MachineLearning/) and Zorroxied (https://upload.wikimedia.org/wikipedia/commons/8/8f/Autocorrelation_cryptanalisys_exmp.png)

Linear Regression Assumptions



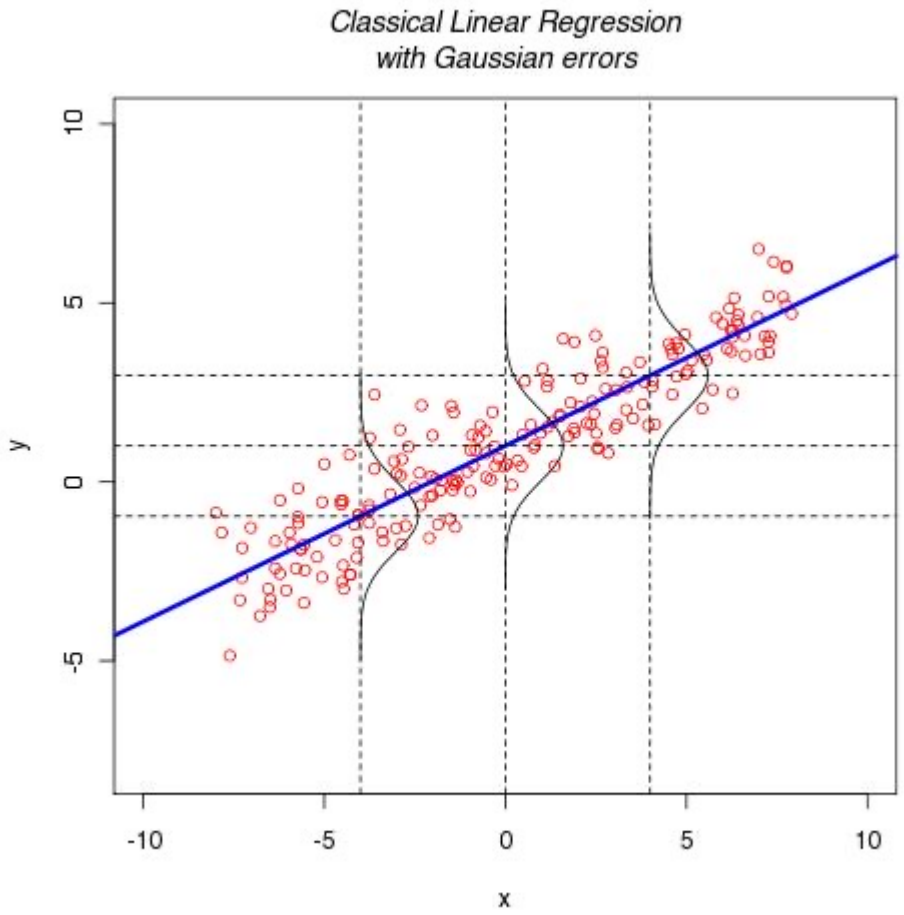
Normality



Unequal Variance

Figures from Steve D. Jost @DePaul University (http://facweb.cs.depaul.edu/sjost/csc423/documents/nplots.gif) and Balaji Pitchai Kannu (https://www.quora.com/What-is-heteroscedasticity-1)

Error term distribution



Figures from Vinh Nguyen
(http://blog.nguyenvq.com/blog/2009/05/12/linear-regression-plot-with-normal-curves-for-error-sideways/)

Note on Correlations

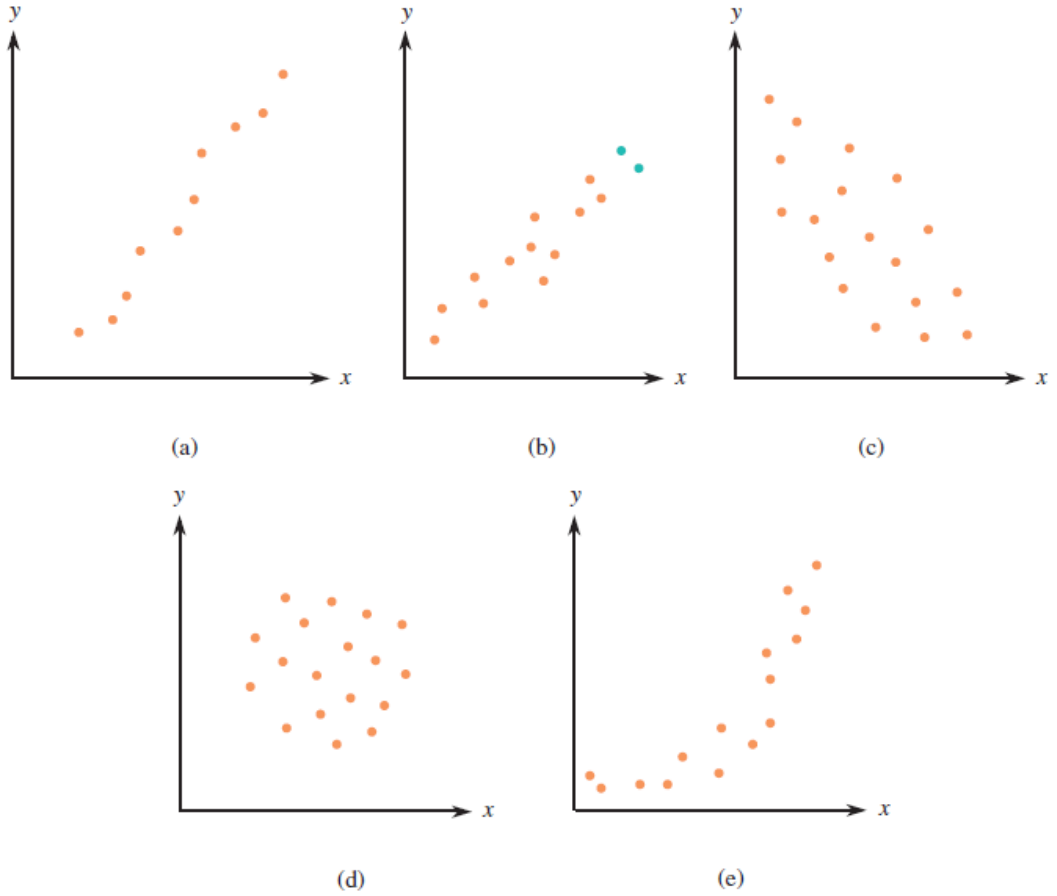


FIGURE 5.1
Scatterplots illustrating various types of relationships: (a) positive linear relationship; (b) another positive linear relationship; (c) negative linear relationship; (d) no relationship; (e) curved relationship.

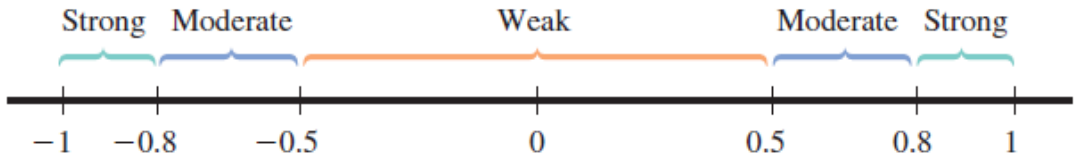


FIGURE 5.4
Describing the strength of a linear relationship.

- Correlation coefficient has a value between -1 and 1
  - The closer the magnitude is to 1, the stronger the correlation
- Pandas has corr() function that we can use to find correlation among features/with target
  - Usage: df.corr()

Figures from Introduction to Statistics & Data Analysis by Peck et. al

Coefficient of determination (r^2)

- Coefficient of determination is a measure of the proportion of variability in the target variable that can be explained by the linear relationship between the input features and the target.

$$r^2 = 1 - \frac{SSResid}{SSTotal}$$

Unexplainable Variance

Explainable Variance

- Where $SSResid = \sum(y - \hat{y})^2$ and $SSTotal = \sum(y - \bar{y})^2$
- Can be used to measure how well the regression equation is doing as it tell us how much better our predictive model is performing to just using mean as predictor.
- r^2 value is between 0 and 1. The higher the value, the more useful the model

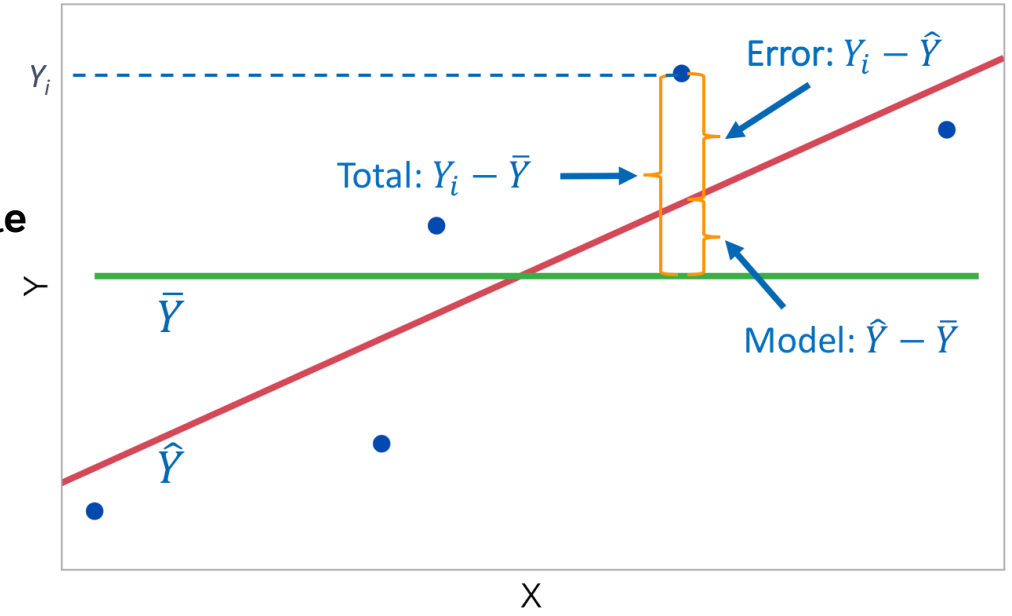


Figure from https://www.jmp.com/en_us/statistics-knowledge-portal/what-is-regression/interpreting-regression-results.html

Content from Introduction to Statistics & Data Analysis by Peck et. al and KMUTT Big Data Experience Center's lecture on Linear Regression

Machine Learning Library: Scikit-learn (sklearn)

- Scikit-learn is one of the most popular open source machine learning library for python.
- Large user base globally and constant updates for a (approximately) 3-month cycle.
- Provides a large variety of machine learning models and tools including but not limited to.
  - Supervised learning models such as regression, tree-based models, ensemble models, neural network, etc.
  - Unsupervised learning models such as clustering models, GMM, density estimation, PCA, etc.
  - Model selection, hyperparameters tuning, and evaluation tools such as grid search, cross-validation, various evaluation metrics, etc.

Linear Regression Example

- Colab!
- Tutorial - Simple Linear Regression
- Short Exercise - Simple Linear Regression

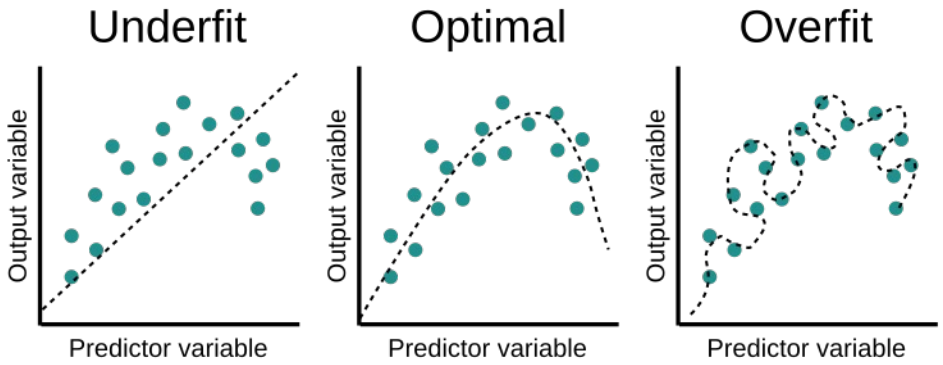
Recap: Overfitting vs. Underfitting

Overfitting

Overfitting occurs when the trained model adjust its parameters to fit the training data too well and became unable to generalize well. This behavior is observed when the model performs well with during the development with training data but performs poorly on the unseen data.

Underfitting

Underfitting happens when the trained model is unable to adjust itself to sufficiently fit the training data, resulting in poor performances overall. This behavior is observed when the model is unable to perform well even during the development with training data.



Dealing with Overfitting: Regularization

- There are many causes of overfitting ranging from the presents of outliers, the lack of data, to the complexity of the model itself being too high, making it fits the training data *too well*.
- **Regularization** is one of the methods used to address overfitting. It penalizes the model with large coefficients and causes them to shrink during optimization process. This is done by introducing penalty terms into the loss function.

$$Loss = \frac{1}{2m}\sum_{i=1}^{m}\left(\hat{y}_i^{(i)} - y^{(i)}\right)^2 + \lambda \boxed{\sum_{i=1}^{n} w_i^2}$$

Example of penalty terms

- Small coefficients will help reduce the impact of less relevant input features
- The regularization constant λ determines the strength of the penalty. It is a tunable hyperparameter of the model.
  - The higher it is, the more penalty imposed. This can help the model becomes more generalizable.
  - Too large regularization constant might leads to underfitting.

Regularization: Ridge & Lasso (& Elastic Net) Regression

- Ridge regression adds an L2 regularization penalty to the cost function

$$Loss = \frac{1}{2m}\sum_{i=1}^{m}\left(\hat{y}_i^{(i)} - y^{(i)}\right)^2 + \lambda\sum_{i=1}^{n} w_i^2 \rightarrow \textbf{L2 Penalty}$$

$$= \frac{1}{2m}\sum_{i=1}^{m}\left(w_0 + w_1 x_1^{(i)} + \cdots + w_n x_n^{(i)} - y^{(i)}\right)^2 + \lambda\sum_{i=1}^{n} w_i^2$$

$$= \frac{1}{2m}\sum_{i=1}^{m}\left(w_0 + \sum_{j=1}^{n} w_j x_j^{(i)} - y^{(i)}\right)^2 + \lambda\sum_{i=1}^{n} w_i^2$$

- Lasso regressions adds an L1 regularization penalty to the cost function

$$Loss = \frac{1}{2m}\sum_{i=1}^{m}\left(\hat{y}_i^{(i)} - y^{(i)}\right)^2 + \lambda\sum_{i=1}^{n} |w_i| \rightarrow \textbf{L1 Penalty}$$

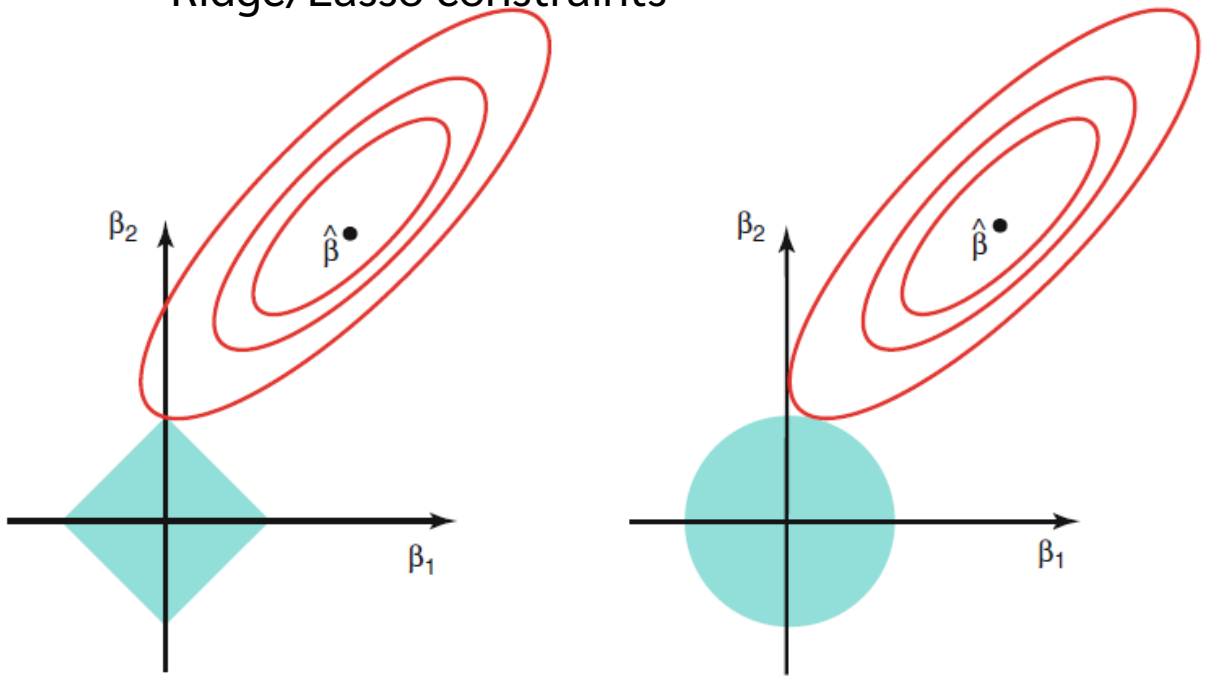
$$= \frac{1}{2m}\sum_{i=1}^{m}\left(w_0 + \sum_{j=1}^{n} w_j x_j^{(i)} - y^{(i)}\right)^2 + \lambda\sum_{i=1}^{n} |w_i|$$

- Elastic Net uses both L1 and L2 penalty

Ridge & Lasso Regression

- For Ridge (L2) regularization, the constraints have a degree of 2 (hence the circular shape in the plot).
  - This causes the weights solution that satisfy the constraints to generally not appear on an axis.
- For Lasso (L1) regularization, the constraints are linear in nature (hence the rectangular shape in the plot).
  - This causes the weight solution to often satisfy the constraints at an axis
  - **<u>This means less relevant coefficient will be forced to 0</u>**

$\hat{\beta}$ represents the of the weights without Ridge/Lasso constraints

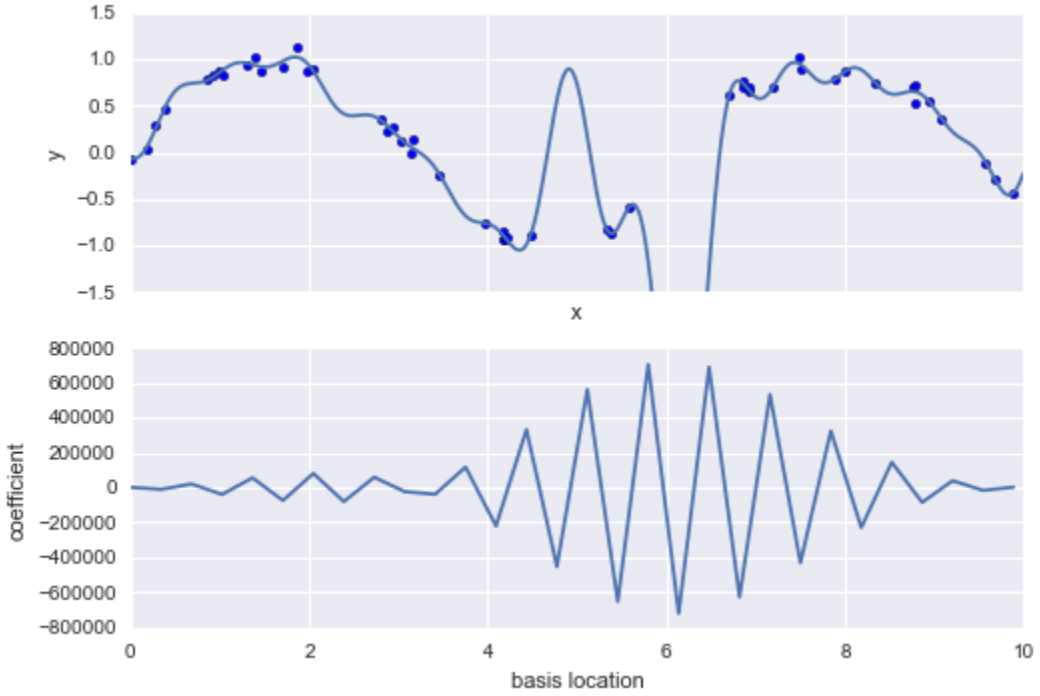


Lasso Regression

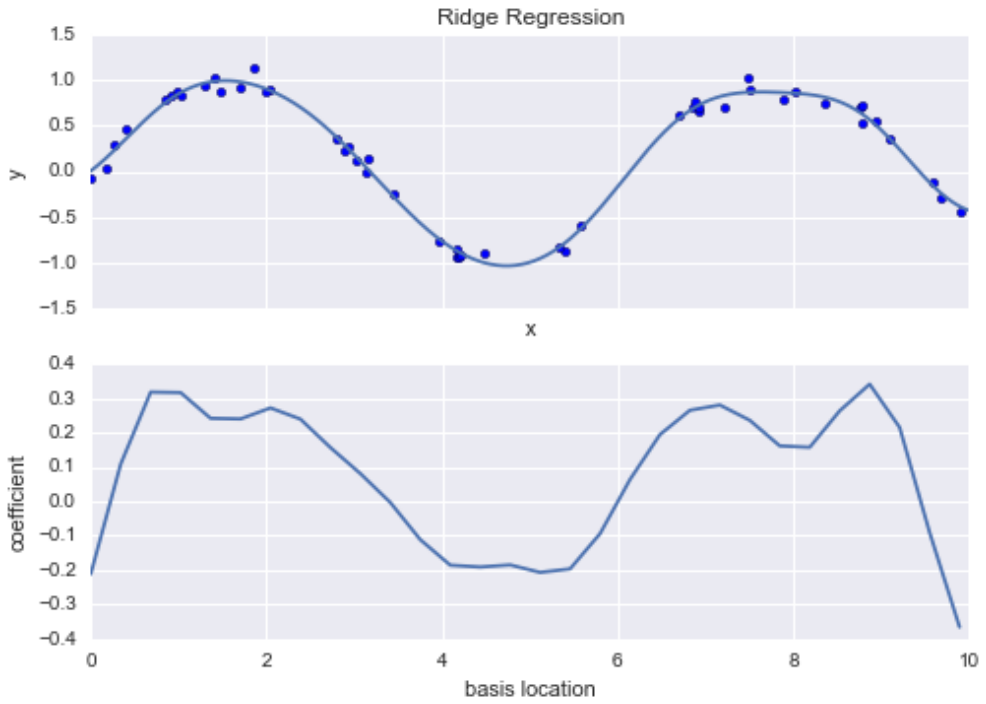
Ridge Regression

Contents and figures from An Introduction to Statistical Learning by Hames et. Al.

Regularization Impact: Ridge

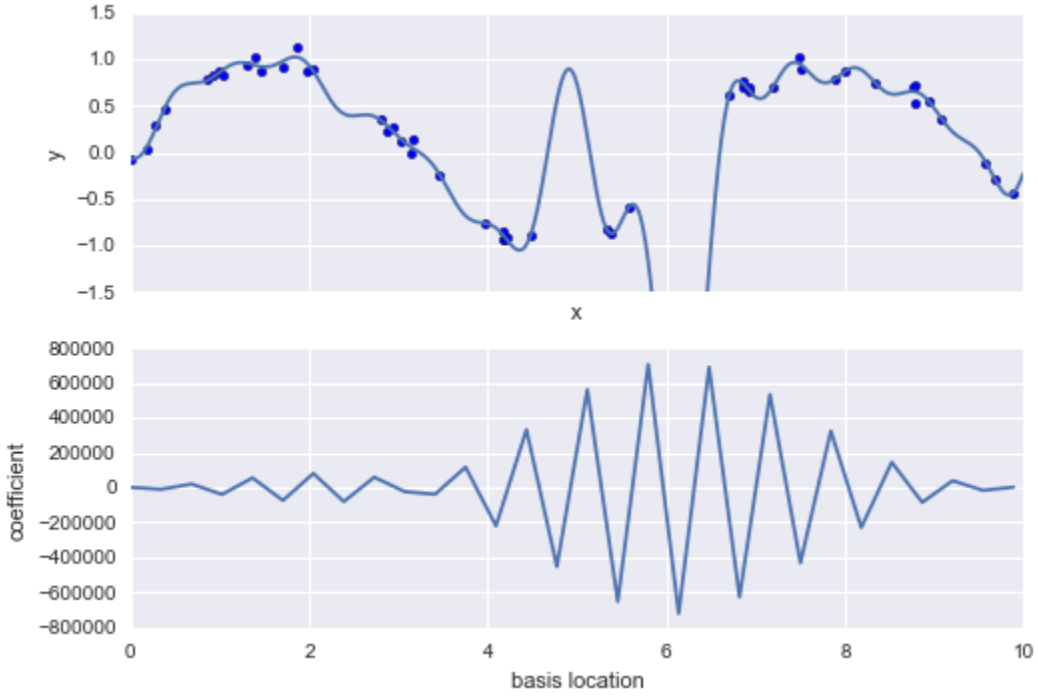


No Regularization

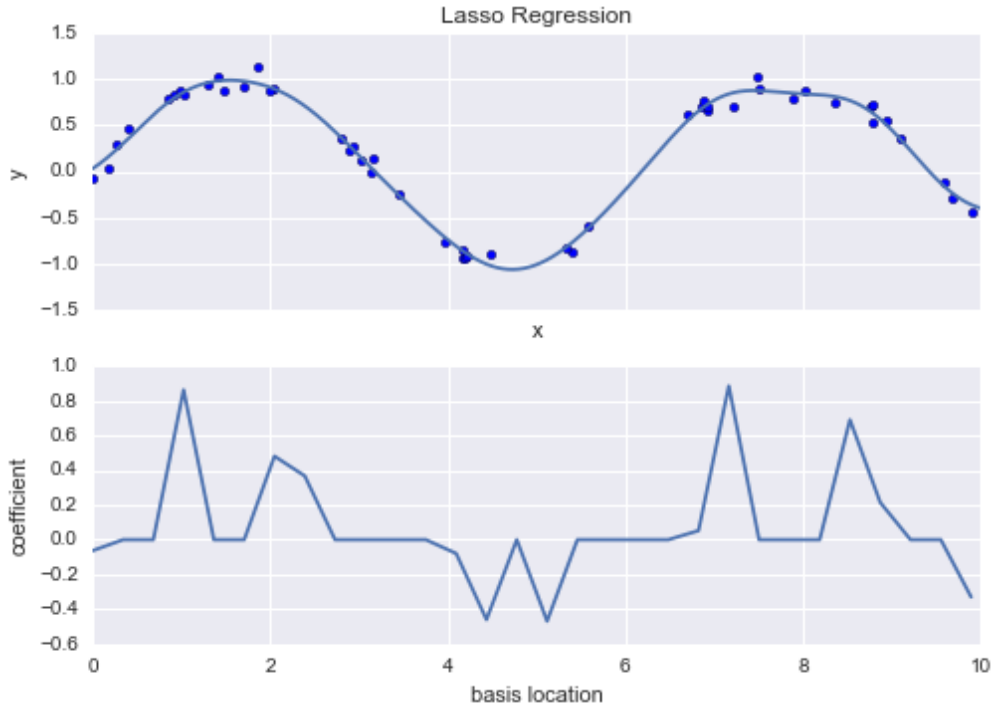


Ridge

Regularization Impact: Lasso



No Regularization



Lasso

Multiple Regression Example

- Colab!
- Tutorial - Simple House Price

Polynomial Regression (Higher Degree Fitting)

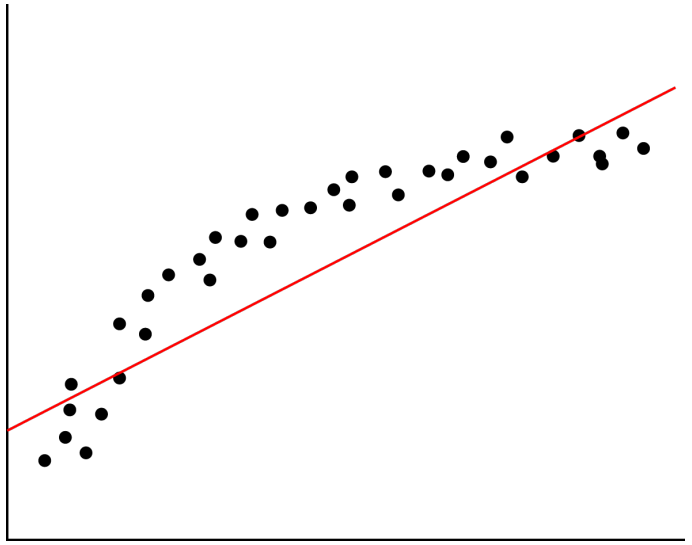
- In the case that relationship between input features of the data and the target output isn't linear, it may be more appropriate to use a polynomial equation

$$y = w_0 + w_1 x + w_2 x^2 + \cdots + w_n x^n$$

- Ex: when fitting a polynomial of degree 3, the equation will be

$$y = w_0 + w_1 x + w_2 x^2 + w_3 x^3$$

- Do we need to develop a new model for this?



Figures from Mike de Waard (https://xyclade.github.io/MachineLearning/)

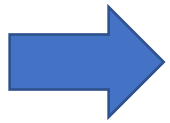
Polynomial Features

- Considering the general linear equation.

$$y = w_0 + w_1 x_1 + w_2 x_2 + \cdots + w_m x_m$$

- Notice that if we let $x_1 = x, x_2 = x^2, x_3 = x^3$, then we are fitting the polynomial model.
- By transforming the input features into desired degree, it is still possible to utilize the linear regression model

X
1
2
3
…



X	X^2	X^3
1	1	1
2	4	8
3	9	27
…	…	…

- Again, there is a library command to do perform this task.
- Note that Ridge and Lasso regularizations can be applied on polynomial regression as well

Thank You

GBDi
Government Big Data Institute
สถาบันส่งเสริมการวิเคราะห์และบริหารข้อมูลขนาดใหญ่ภาครัฐ (สวช.)

Facebook

Twitter

Blockdit

govbigdata

Follow us on

GBDi

gbdi.depa.or.th

YouTube

Government Big Data Institute (GBDi)

Line Official

@gbdi

BIG DATA
THAILAND

Follow us on

BIG DATA
THAILAND

Website

Facebook

Blockdit

Twitter