CST 413 Machine Learning
Module 2

Module 2 (7 hours)

- **Supervised Learning**
- **Regression** - Linear regression with one variable, Linear regression with multiple variables, solution using gradient descent algorithm and matrix method, basic idea of overfitting in regression.
- **Linear Methods for Classification-** Logistic regression, Naive Bayes, Decision tree algorithm ID3.

Regression

Linear Regression – Refer Notes

- Linear regression with one variable,
- Linear regression with multiple variables,
- Solution using matrix method (derivation not required)

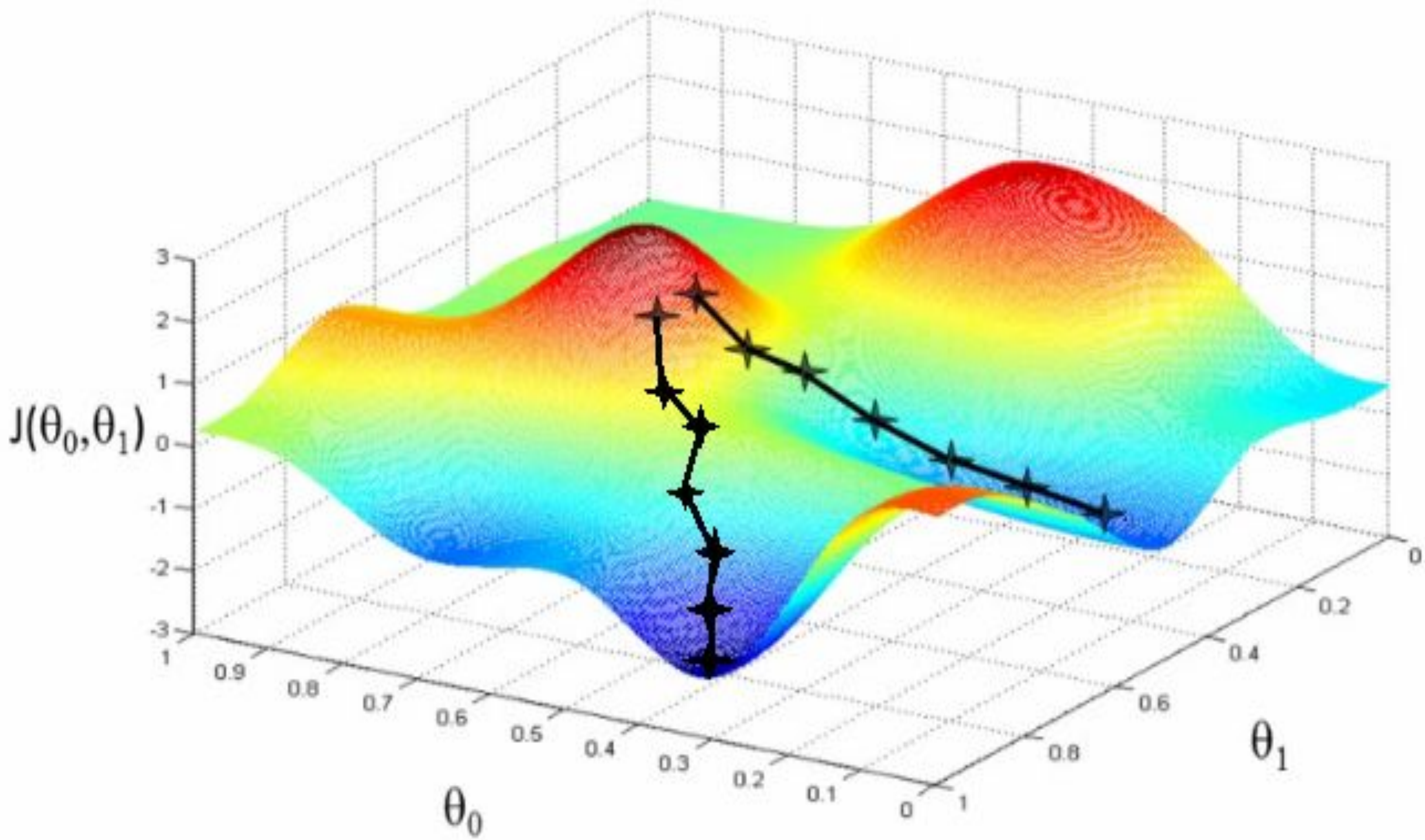
Gradient Descent Algorithm

- **Gradient Descent** is the most common optimization algorithm in *machine learning* and *deep learning*.

- It is a first-order optimization algorithm.

- This means it only takes into account the first derivative when performing the updates on the parameters.

Gradient Descent

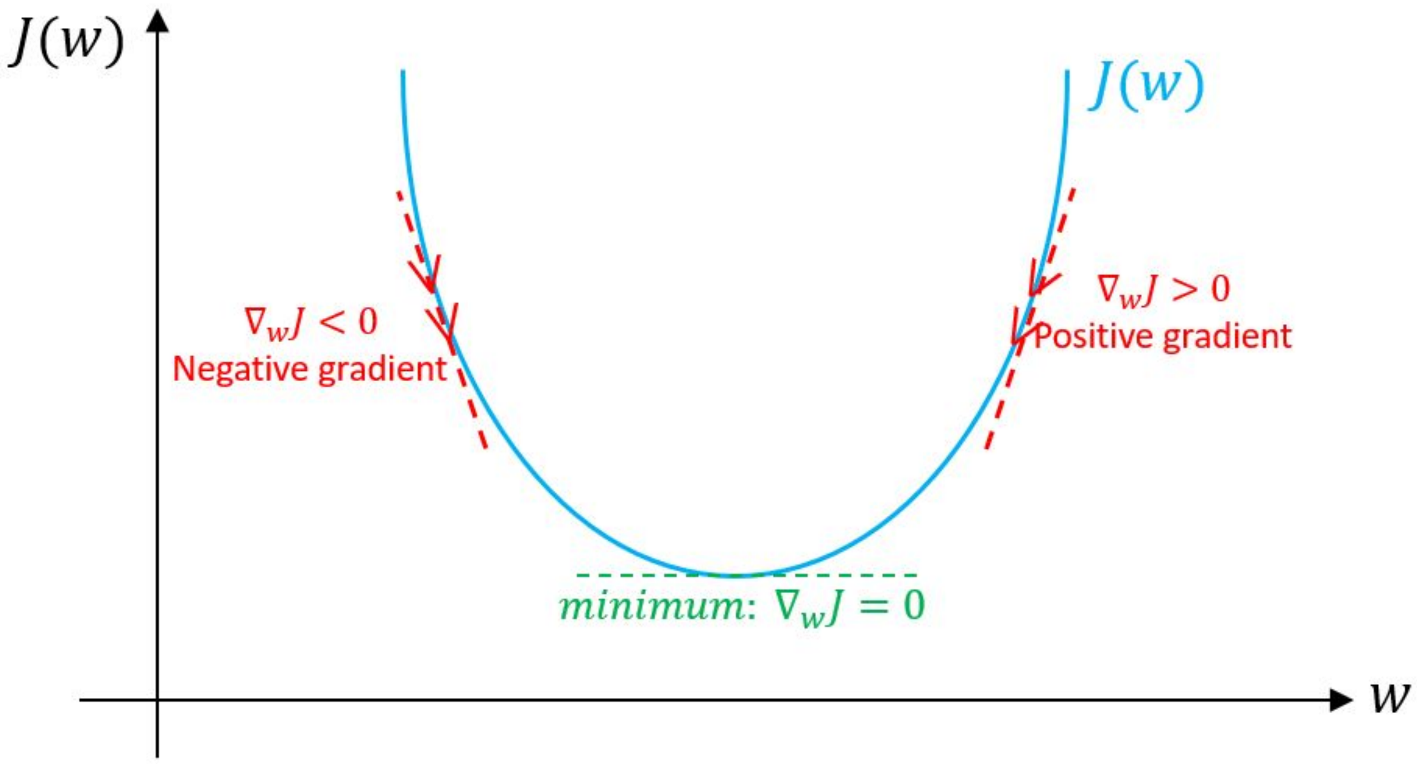


- On each iteration, we update the parameters in the opposite direction of the gradient of the objective function *J(w)* w.r.t the parameters where the gradient gives the direction of the steepest ascent.

- The size of the step we take on each iteration to reach the local minimum is determined by the learning rate α.

- Therefore, we follow the direction of the slope downhill until we reach a local minimum.

Gradient Descent



Linear Regression Solution using Gradient Descent

- Cost Function or Objective Function to be minimized is given by the Mean Square Error (MSE) in approximating data

$$J(\boldsymbol{\theta}) = \frac{1}{2n} \sum_{i=1}^{n} \left(h_{\boldsymbol{\theta}} \left(\boldsymbol{x}^{(i)} \right) - y^{(i)} \right)^2$$

- Initialize $\boldsymbol{\theta}$
- Repeat until convergence

$$\theta_j \leftarrow \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\boldsymbol{\theta})$$

simultaneous update for j = 0 ... d

$$
\begin{aligned}
\frac{\partial}{\partial \theta_j} J(\boldsymbol{\theta}) &= \frac{\partial}{\partial \theta_j} \frac{1}{2n} \sum_{i=1}^{n} \left(h_{\boldsymbol{\theta}} \left(\boldsymbol{x}^{(i)} \right) - y^{(i)} \right)^2 \\
&= \frac{\partial}{\partial \theta_j} \frac{1}{2n} \sum_{i=1}^{n} \left(\sum_{k=0}^{d} \theta_k x_k^{(i)} - y^{(i)} \right)^2 \\
&= \frac{1}{n} \sum_{i=1}^{n} \left(\sum_{k=0}^{d} \theta_k x_k^{(i)} - y^{(i)} \right) \times \frac{\partial}{\partial \theta_j} \left(\sum_{k=0}^{d} \theta_k x_k^{(i)} - y^{(i)} \right) \\
&= \frac{1}{n} \sum_{i=1}^{n} \left(\sum_{k=0}^{d} \theta_k x_k^{(i)} - y^{(i)} \right) x_j^{(i)}
\end{aligned}
$$

- Initialize $\boldsymbol{\theta}$
- Repeat until convergence

$$\theta_j \leftarrow \theta_j - \alpha \frac{1}{n} \sum_{i=1}^{n} \left(h_{\boldsymbol{\theta}} \left(\boldsymbol{x}^{(i)} \right) - y^{(i)} \right) x_j^{(i)}$$

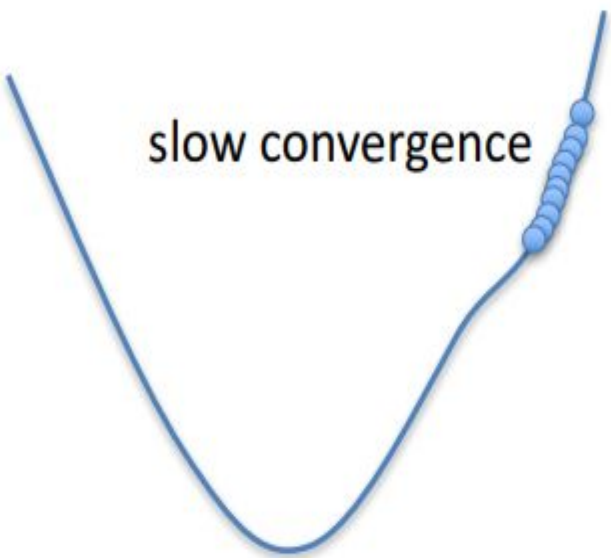
simultaneous update for j = 0 ... d

Stochastic gradient descent algorithm

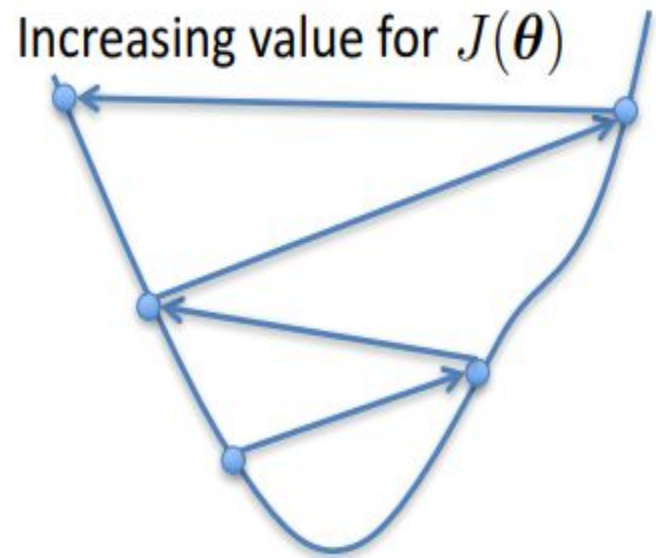
- ***On-line gradient descent, also known as sequential gradient descent or stochastic gradient descent***, makes an update to the weight vector based on one data point at a time.

Choosing α

α too small



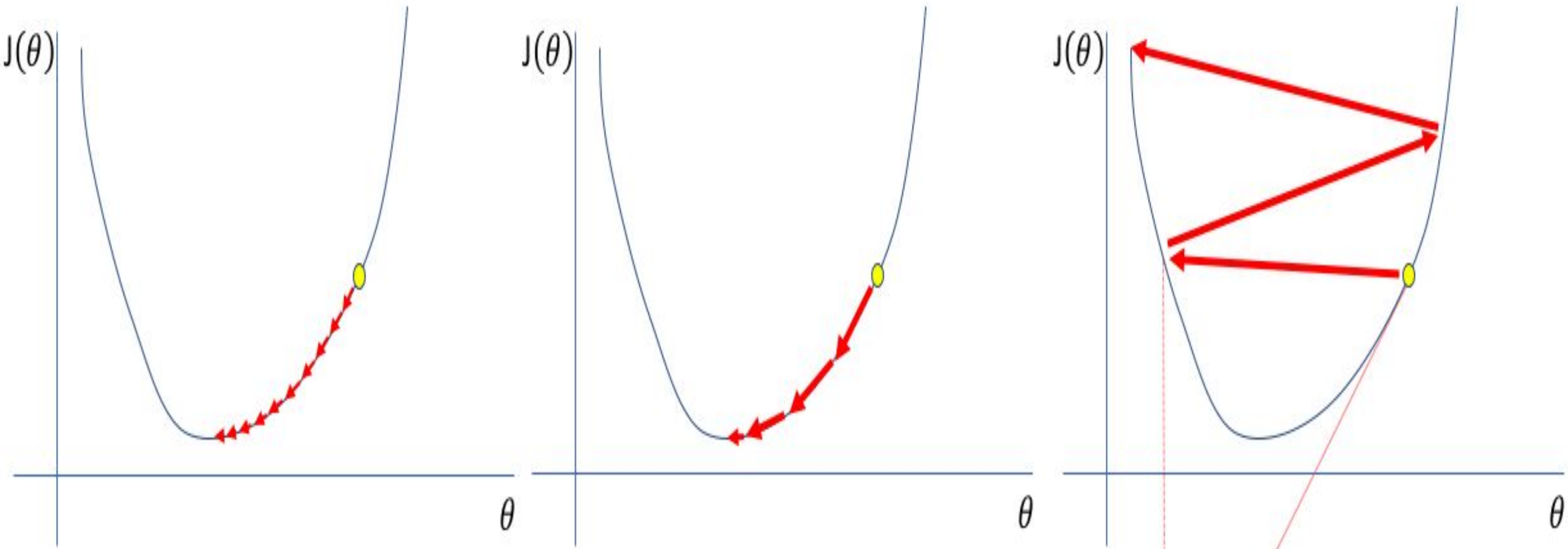
α too large



- May overshoot the minimum
- May fail to converge
- May even diverge

https://faculty.cc.gatech.edu/~bboots3/CS4641-Fall2018/Lecture3/03_LinearRegression.pdf

Too low | **Just right** | **Too high**

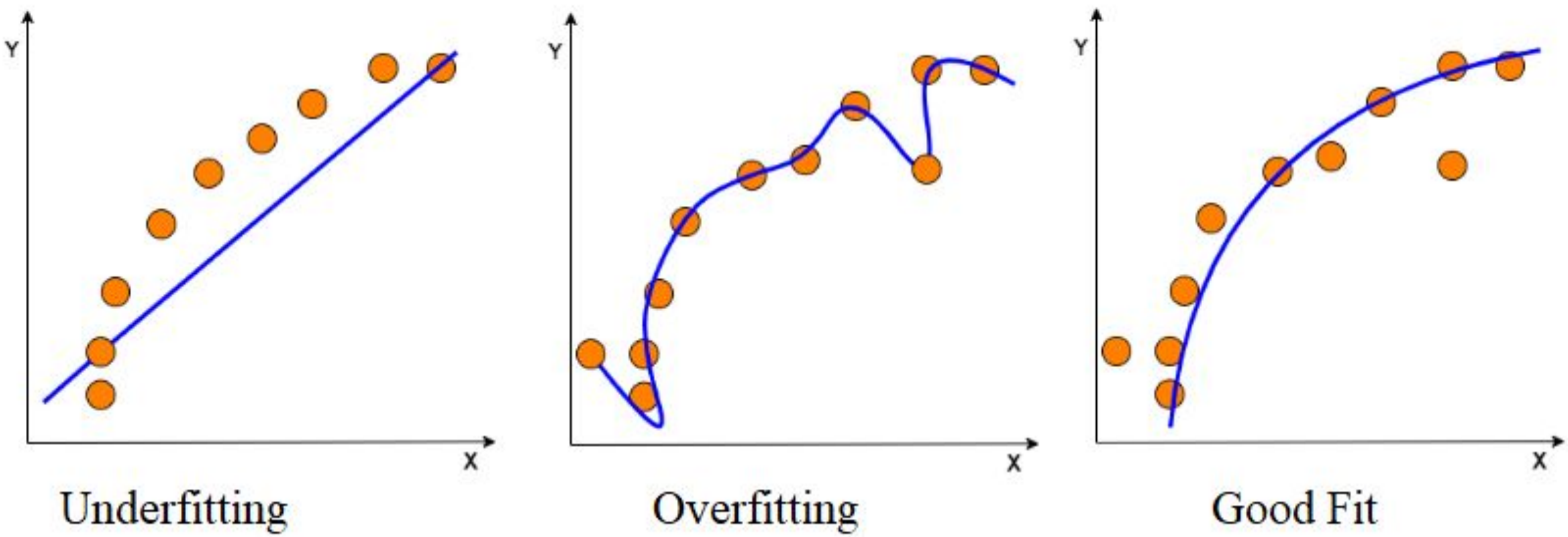


A small learning rate requires many updates before reaching the minimum point

The optimal learning rate swiftly reaches the minimum point

Too large of a learning rate causes drastic updates which lead to divergent behaviors

Overfitting and Underfitting

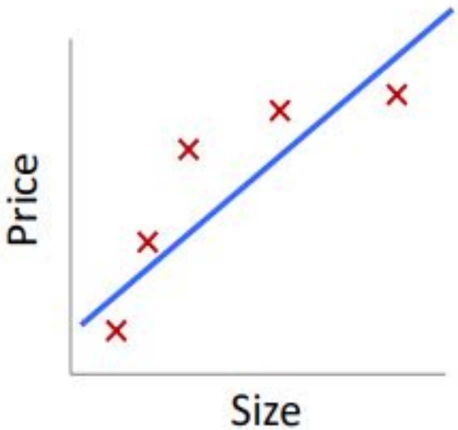


Training error – high
Testing error – high
High bias

Training error – less
Testing error – high
High variance

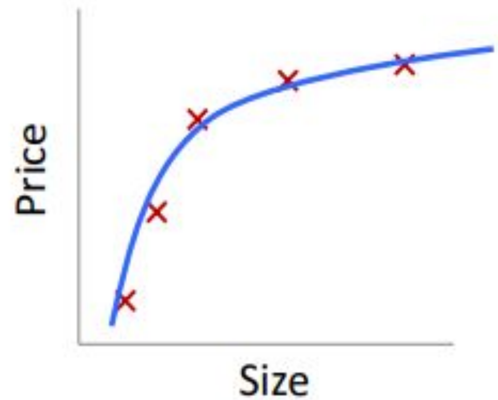
Training error – minimum
Testing error – minimum
Bias-variance trade-off

Quality of Fit



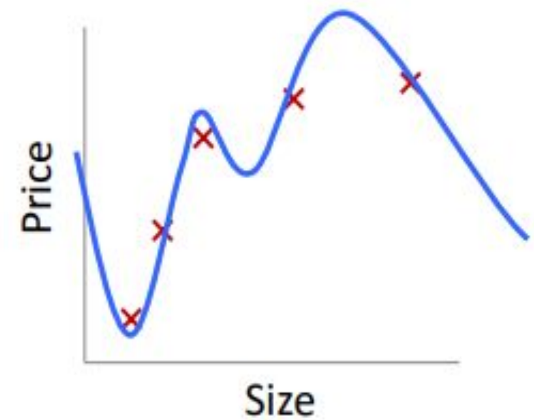
$\theta_0 + \theta_1 x$

Underfitting
(high bias)



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

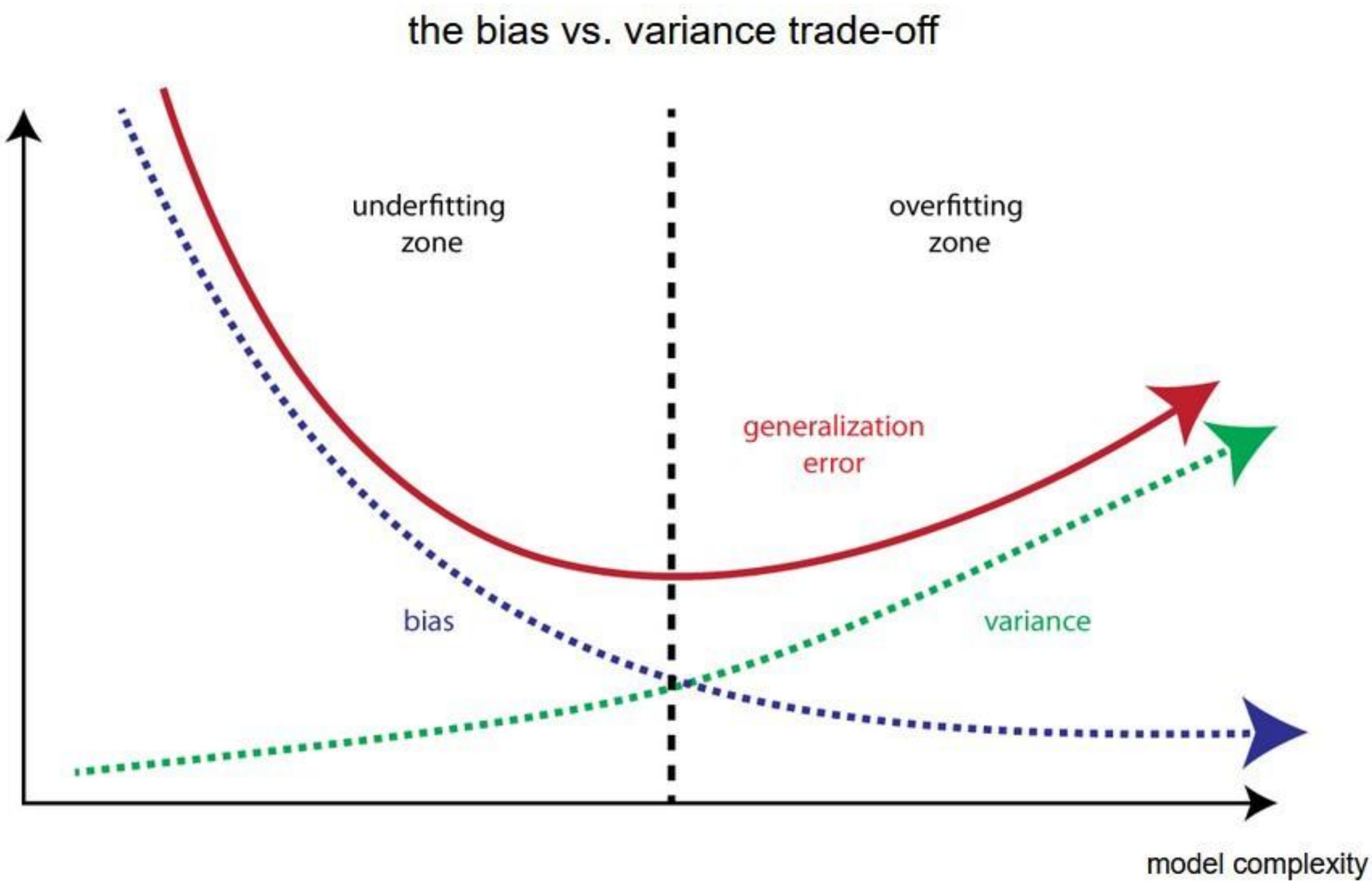
Correct fit



$\theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4$

Overfitting
(high variance)

Bias – Variance Trade-off



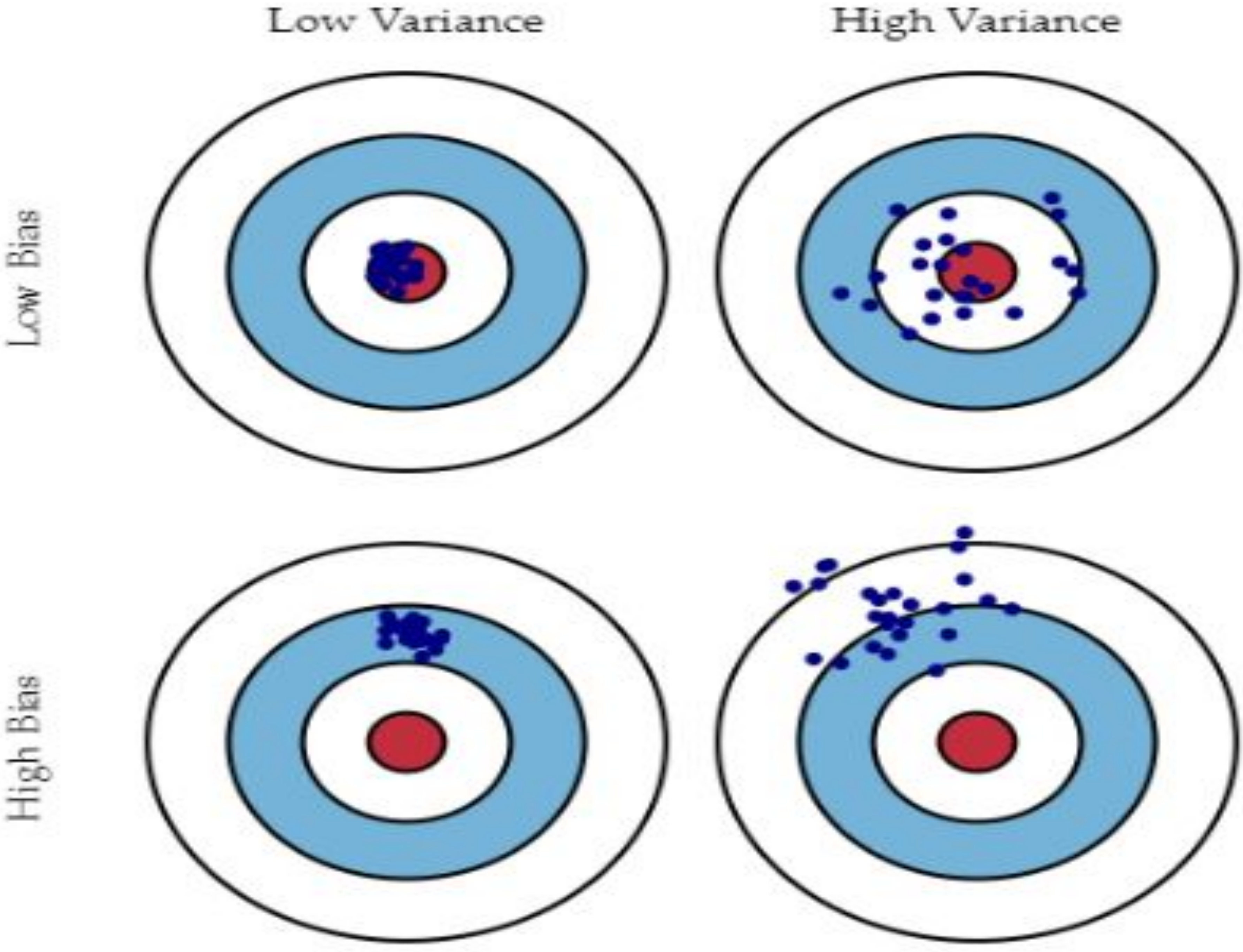
- **What is bias?**
- Bias is the difference between the average prediction of our model and the correct value which we are trying to predict.
- Model with high bias pays very little attention to the training data and oversimplifies the model.
- It always leads to high error on training and test data.

- **What is variance?**
- Variance is the variability of model prediction for a given data point or a value which tells us spread of our data.

- Model with high variance pays a lot of attention to training data and does not generalize on the data which it hasn’t seen before.

- Such models perform very well on training data but has high error rates on test data.

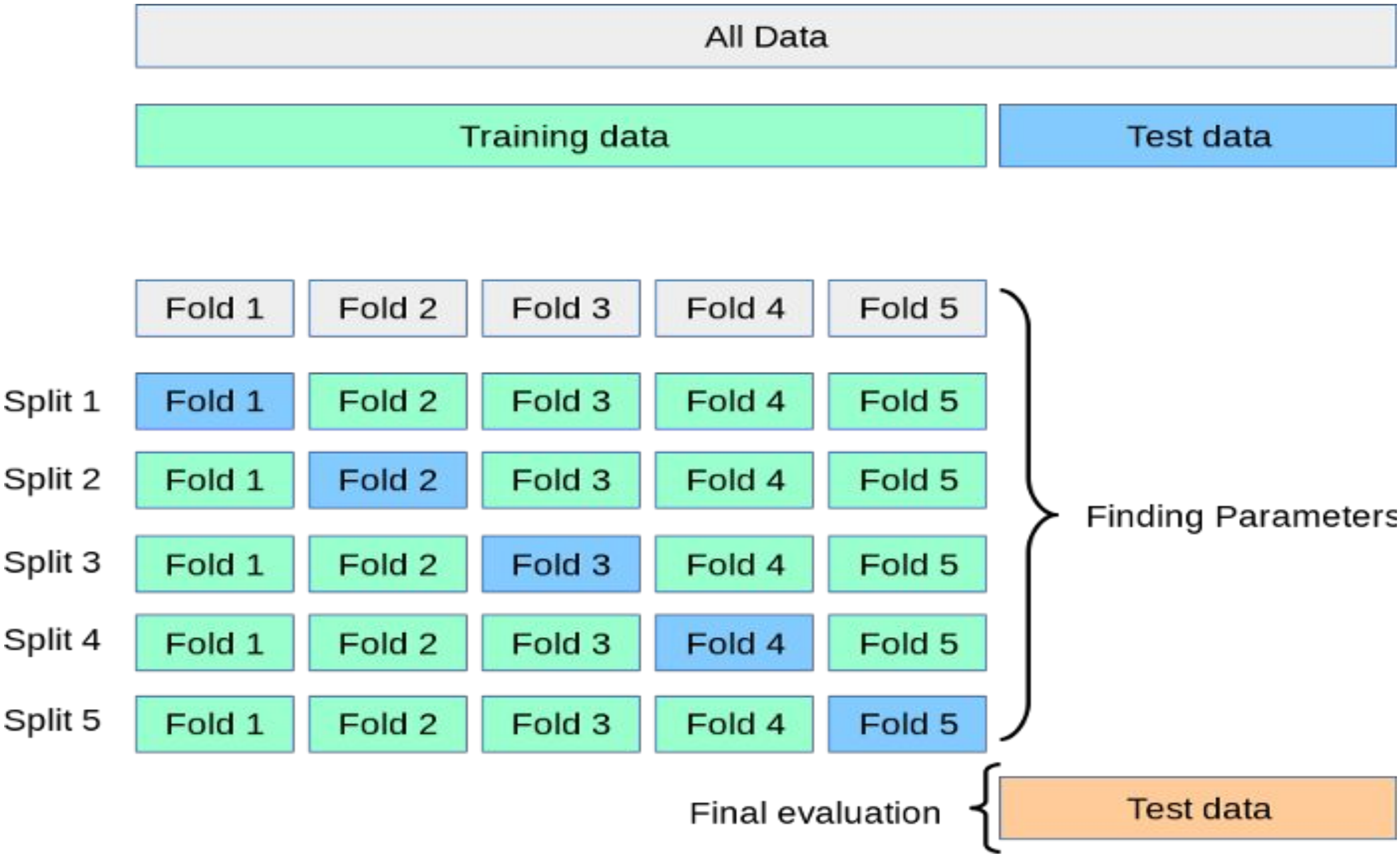
- **Low Bias:** The average prediction is very close to the target value
- **High Bias:** The predictions differ too much from the actual value
- **Low Variance:** The data points are compact and do not vary much from their mean value
- **High Variance:** Scattered data points with huge variations from the mean value and other data points


Low Variance
High Variance
Low Bias
High Bias

Overfitting

- The learned hypothesis may fit the training set very well.
- ...but may fail to generalize to new examples.
- **How To Avoid Overfitting?**
- Since overfitting algorithm captures the noise in data, reducing the number of features will help.
- K-fold cross validation
- Increase the training data
- Regularization

K-fold cross validation



Regularization

- **L**east **A**bsolute **S**hrinkage and **S**election **O**perator (LASSO) Regression - **L1 regularization**

  - adds penalty equivalent to **absolute value of the magnitude** of coefficients

  - Minimization objective = LS Obj + **λ*** (sum of absolute value of coefficients)

$$\sum_{i=1}^{M}(y_i - \hat{y}_i)^2 = \sum_{i=1}^{M}\left(y_i - \sum_{j=0}^{p} w_j \times x_{ij}\right)^2 + \lambda \sum_{j=0}^{p} |w_j|$$

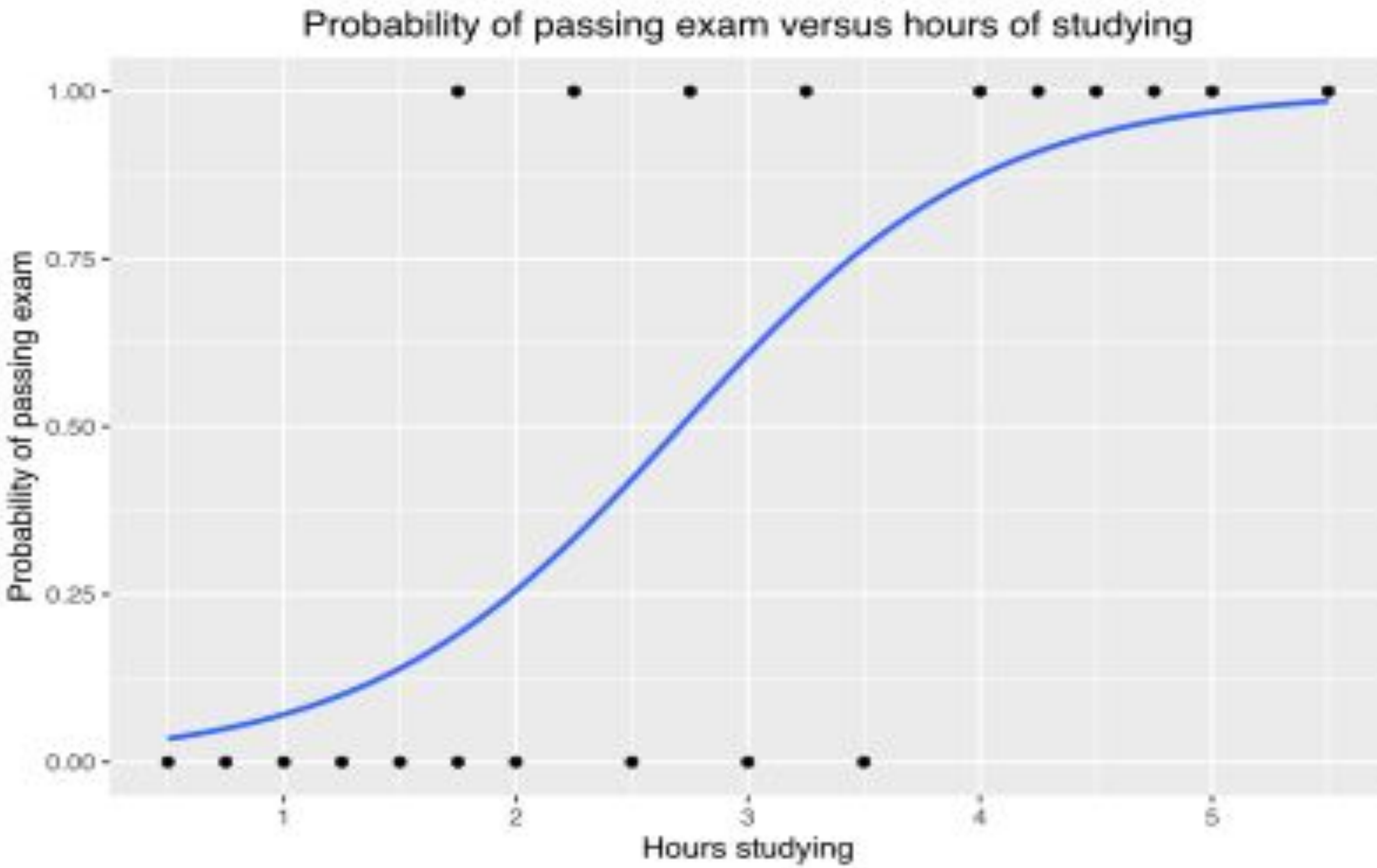
- **Ridge Regression – L2 regularization**
  - adds penalty equivalent to **square of the magnitude** of coefficients
  - Minimization objective = LS Obj + **λ** * (sum of square of coefficients)

$$\sum_{i=1}^{M}(y_i-\hat{y}_i)^2=\sum_{i=1}^{M}\left(y_i-\sum_{j=0}^{p}w_j\times x_{ij}\right)^2+\lambda\sum_{j=0}^{p}w_j^2$$

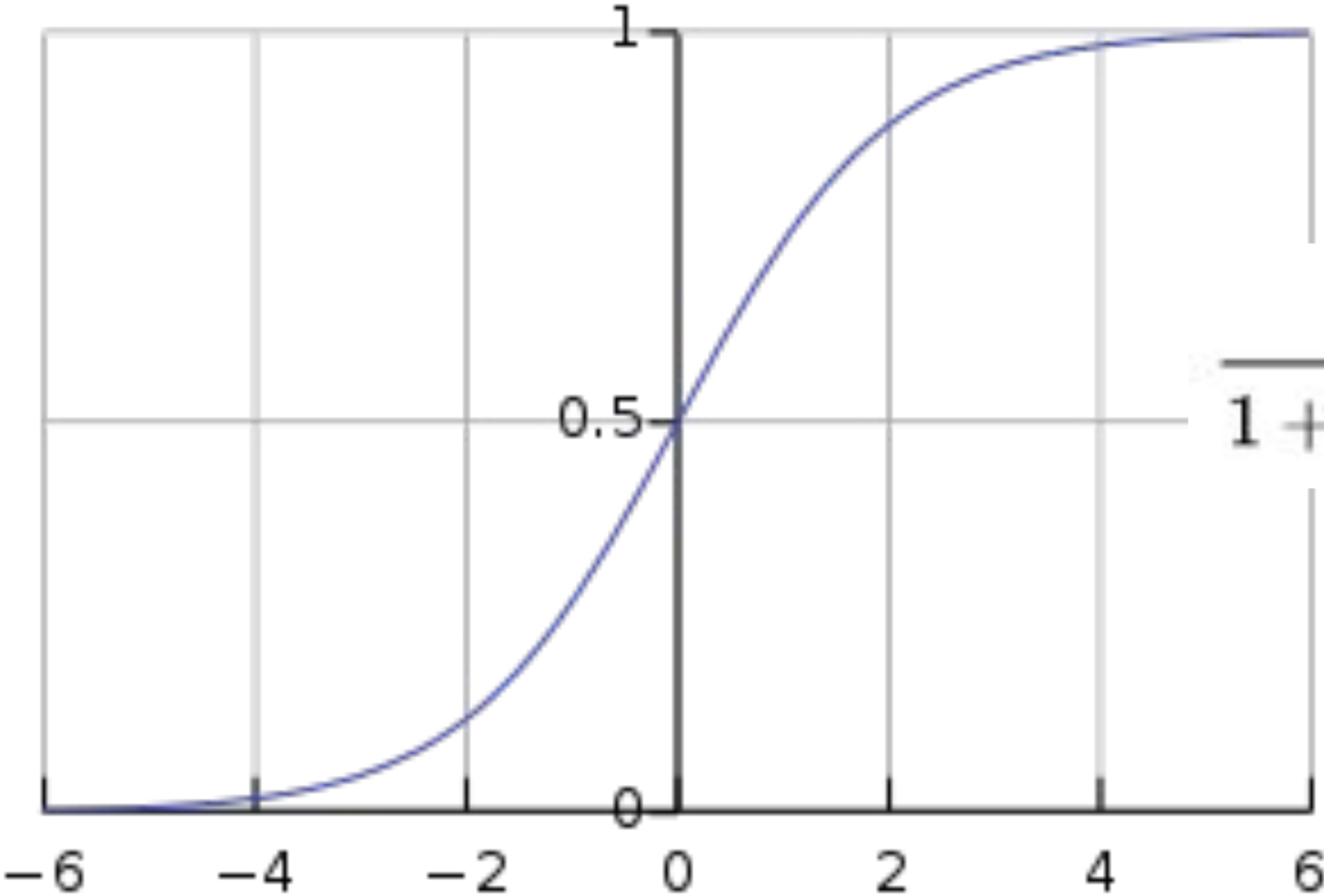
- **How To Avoid Underfitting?**
- Increasing the model complexity. e.g. If linear function under fit then try using polynomial features

Linear Methods for Classification

Logistic Regression



Logistic or sigmoid function

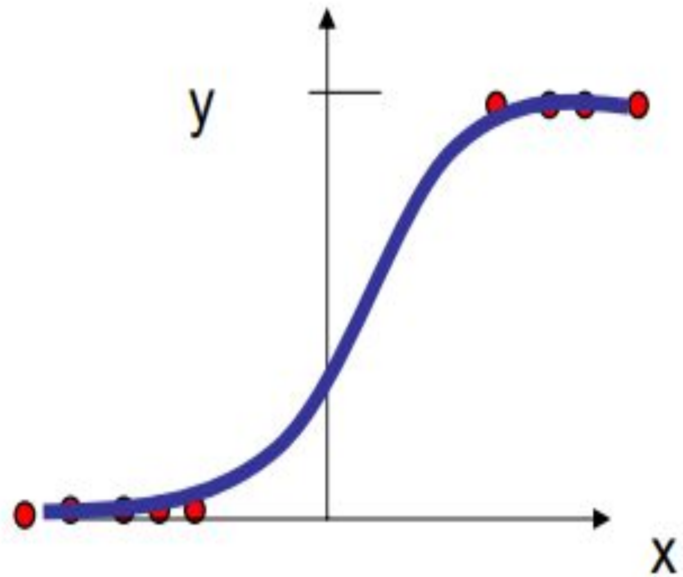


$$\frac{1}{1+e^{-(\beta_0+\beta_1 x)}}$$

Logit

$$p_i = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x_i)}}$$

$$p_i = \frac{e^{\beta_0 + \beta_1 x_i}}{e^{\beta_0 + \beta_1 x_i} + 1}$$



Log odds

- **Odds (odds of success):** It is defined as the chances of success divided by the chances of failure.

- Odds = p/(1-p)

- **Log odds**: It is the logarithm of the odds ratio.

- Log odds = log[p/(1-p)]

Log odds or Logit function

$$p = \frac{e^{b_0 + b_1 x_1}}{1 + e^{b_0 + b_1 x_1}}$$

$$1 - p = 1 - \frac{e^{b_0 + b_1 x_1}}{1 + e^{b_0 + b_1 x_1}} = \frac{1}{1 + e^{b_0 + b_1 x_1}}$$

$$\frac{p}{1 - p} = e^{b_0 + b_1 x_1} \longrightarrow \boxed{\log\left(\frac{p}{1 - p}\right) = b_0 + b_1 X_1}$$

https://digitaschools.com/binary-logistic-regression-introduction/

Logistic Regression

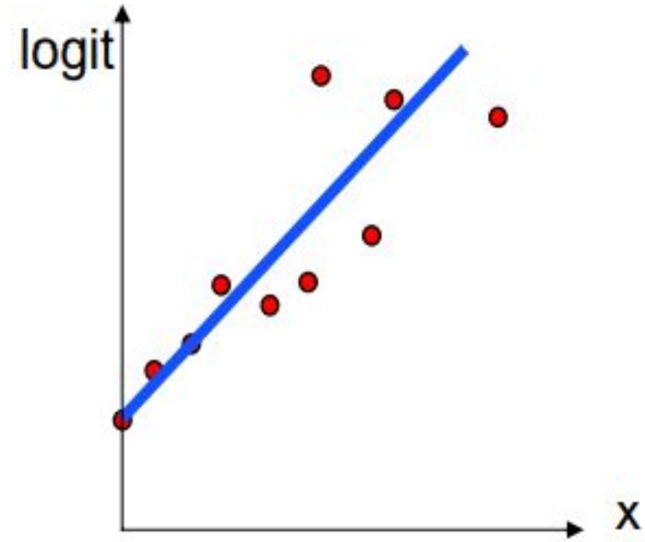
LOGISTIC REGRESSION MODEL

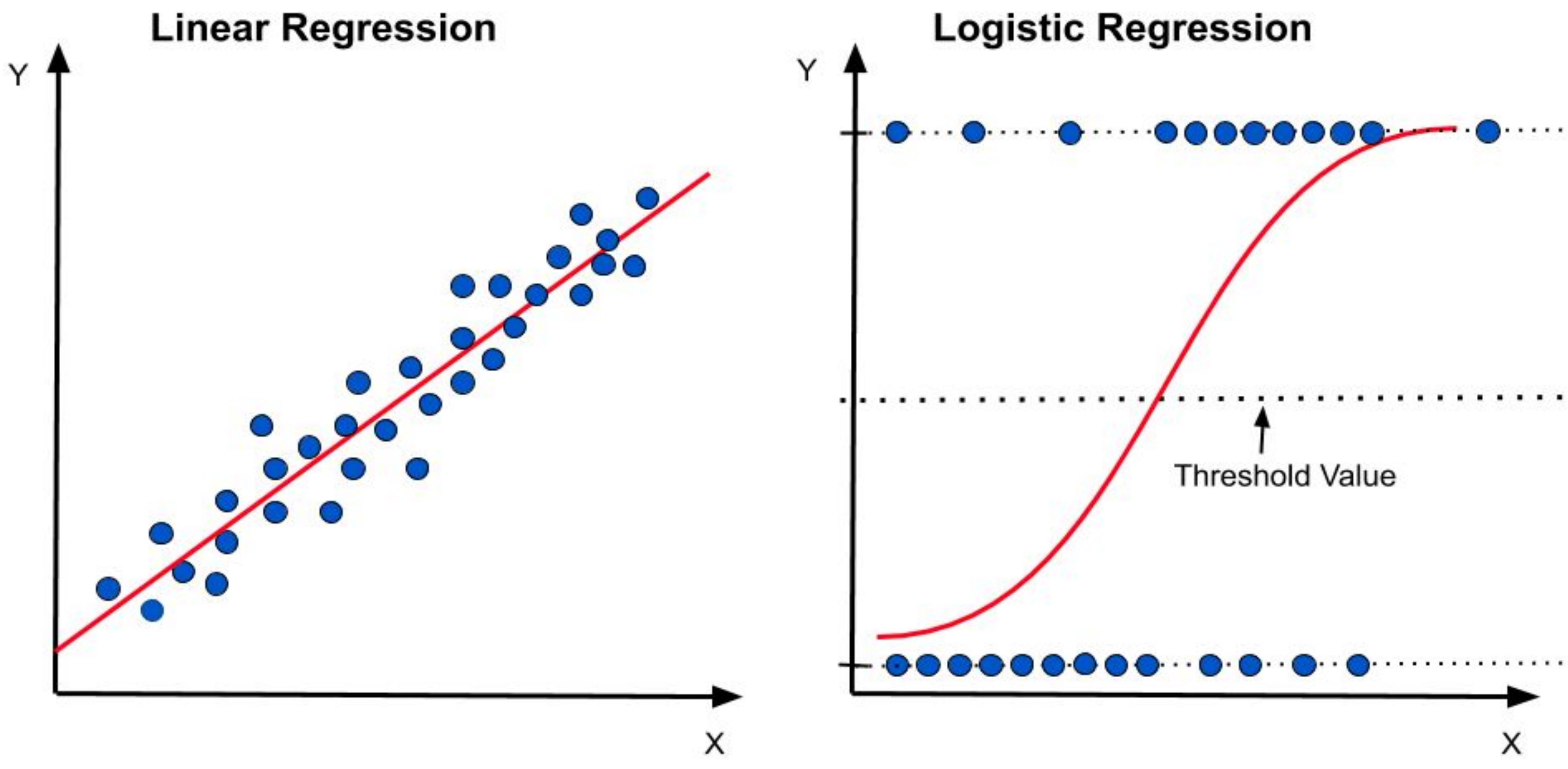
The **statistical model for logistic regression** is

$$\log\left(\frac{p}{1-p}\right) = \beta_0 + \beta_1 x$$

where p is a binomial proportion and x is the explanatory variable. The parameters of the logistic regression model are β_0 and β_1.

$$\log\left[\frac{p_i}{1-p_i}\right] = \beta_0 1 + \beta_1 x_i$$




Linear Regression
Y
X
Logistic Regression
Y
Threshold Value
X

	Linear Regression	Logistic Regression
Response Variable	Continuous (e.g. price, age, height, distance)	Categorical (yes/no, male/female, win/not win)
Equation Used	$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots$	$p(Y) = e^{(\beta 0 + \beta 1X1 + \beta 2X2 + \ldots)} / (1 + e^{(\beta 0 + \beta 1X1 + \beta 2X2 + \ldots)})$
Method Used to Fit Equation	Ordinary Least Squares	Maximum Likelihood Estimation
Output to Predict	Continuous value ($150, 40 years, 10 feet, etc.)	Probability (0.741, 0.122, 0.345, etc.)

Cost function of logistic regression

$$Cost((h_\theta(x), y) = \frac{1}{m}\sum_{i=0}^{m} -y^i \log\left(h_\theta\left(x^i\right)\right) - (1 - y^i)log(1 - h_\theta\left(x^i\right))$$

Naïve Bayes Classifier

(Based on Bayes' Theorem)

Bayes' Theorem

likelihoo
d

prio
r

posterior

$$P(H|\mathbf{X}) = \frac{P(\mathbf{X}|H)P(H)}{P(\mathbf{X})}$$

evidence

X - data sample ("*evidence*").

H - *hypothesis* that X belongs to class C.

P(**X**): probability that sample data is observed.

P(H) - *prior probability*: the initial probability.

P(H|**X**) - *posteriori probability:* the probability that the hypothesis holds given the observed data sample **X.**

P(**X**|H) (likelihood): the probability of observing the sample **X**, given that the hypothesis holds.

Bayesian Classification: Why?

- A statistical classifier: performs *probabilistic prediction, i.e.,* predicts class membership probabilities.
- Foundation: Based on Bayes' Theorem.
- Incremental: Each training example can incrementally increase/decrease the probability that a hypothesis is correct — prior knowledge can be combined with observed data.

Classification - Derive the Maximum Posteriori

- Let D be a training set of tuples and their associated class labels, and each tuple is represented by an n-D attribute vector $\mathbf{X} = (x_1, x_2, \ldots, x_n)$.
- Suppose there are *m* classes $C_1, C_2, \ldots, C_m$.
- Classification is to derive the maximum posteriori, i.e., the maximal $P(C_i|\mathbf{X})$. This can be derived from Bayes' theorem,

$$P(C_i|\mathbf{X}) = \frac{P(\mathbf{X}|C_i)P(C_i)}{P(\mathbf{X})}$$

Naïve Bayes Classifier

- A simplified assumption: attributes are conditionally independent (i.e., no dependence relation between attributes):

$$P(\mathbf{X} \mid C_i) = \prod_{k=1}^{n} P(x_k \mid C_i)$$

$$= P(x_1 \mid C_i) \times P(x_2 \mid C_i) \times \ldots \times P(x_n \mid C_i)$$

Naïve Bayes Classifier: Training Dataset

Class:

C1:buys_computer = 'yes'

C2:buys_computer = 'no'

Data to be classified:

X = (age <=30,

Income = medium,

Student = yes

Credit_rating = Fair)

age	income	student	credit_rating	com
<=30	high	no	fair	no
<=30	high	no	excellent	no
31…40	high	no	fair	yes
>40	medium	no	fair	yes
>40	low	yes	fair	yes
>40	low	yes	excellent	no
31…40	low	yes	excellent	yes
<=30	medium	no	fair	no
<=30	low	yes	fair	yes
>40	medium	yes	fair	yes
<=30	medium	yes	excellent	yes
31…40	medium	no	excellent	yes
31…40	high	yes	fair	yes
>40	medium	no	excellent	no

Naïve Bayes Classifier: An Example

- $P(C_i)$: P(buys_computer = “yes”) = 9/14 = 0.643
  P(buys_computer = “no”) = 5/14= 0.357
- Compute $P(X|C_i)$ for each class
  P(age = “<=30” | buys_computer = “yes”) = 2/9 = 0.222
  P(age = “<= 30” | buys_computer = “no”) = 3/5 = 0.6
  P(income = “medium” | buys_computer = “yes”)
  = 4/9 = 0.444
  P(income = “medium” | buys_computer = “no”)
  = 2/5 = 0.4
  P(student = “yes” | buys_computer = “yes)
  = 6/9 = 0.667
  P(student = “yes” | buys_computer = “no”) = 1/5 = 0.2
  P(credit_rating = “fair” | buys_computer = “yes”)
  = 6/9 = 0.667

P(credit_rating = “fair” | buys_computer = “no”) = 2/5 = 0.4

- **X = (age <= 30 , income = medium, student = yes, credit_rating = fair)**

$P(X|C_i)$:

- P(X|buys_computer = “yes”)

  = 0.222 x 0.444 x 0.667 x 0.667 = 0.044

- P(X|buys_computer = “no”)

  = 0.6 x 0.4 x 0.2 x 0.4 = 0.019

$P(X|C_i)*P(C_i)$:

- P(X|buys_computer = “yes”) * P(buys_computer = “yes”) = 0.028
- P(X|buys_computer = “no”) * P(buys_computer = “no”) = 0.007

Therefore, X belongs to class (“buys_computer = yes”)

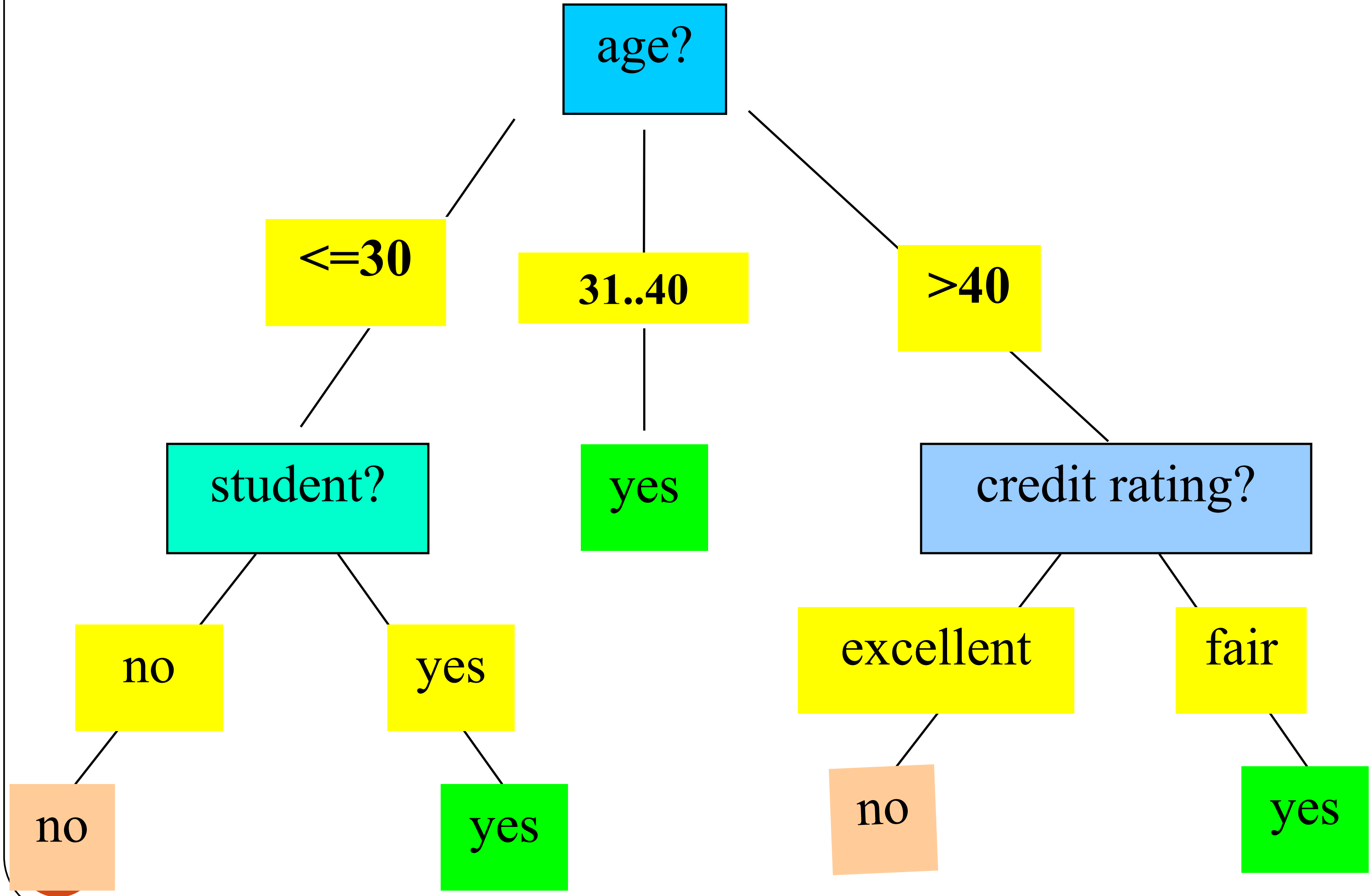
Decision Tree Induction – ID3 (Iterative Dichotomiser)

(by ROSS QUINLAN, 1986)

https://link.springer.com/content/pdf/10.1007/BF00116251.pdf

Decision Tree Induction: An Example

age	income	student	credit_rating	buys_computer
<=30	high	no	fair	no
<=30	high	no	excellent	no
31…40	high	no	fair	yes
>40	medium	no	fair	yes
>40	low	yes	fair	yes
>40	low	yes	excellent	no
31…40	low	yes	excellent	yes
<=30	medium	no	fair	no
<=30	low	yes	fair	yes
>40	medium	yes	fair	yes
<=30	medium	yes	excellent	yes
31…40	medium	no	excellent	yes
31…40	high	yes	fair	yes
>40	medium	no	excellent	no


age?
<=30
31..40
>40
student?
yes
credit rating?
no
yes
excellent
fair
no
yes
no
yes

Algorithm for Decision Tree Induction

- Basic algorithm (a greedy algorithm)
  - At start, all the training examples are at the root.
  - Attributes are categorical (discretize if continuous-valued).
  - Examples are partitioned recursively based on selected attributes.
  - Test attributes are selected on the basis of a heuristic or statistical measure (e.g., information gain).

- Conditions for stopping partitioning.
  - All samples for a given node belong to the same class.
  - There are no remaining attributes for further partitioning – majority voting is employed for classifying the leaf.
  - There are no samples left.

Attribute Selection Measures – Information Gain

- Let p_i be the probability that an arbitrary tuple in D belongs to class C_i, estimated by $|C_{i,D}|/|D|$.

- Expected information (entropy) needed to classify a tuple in D:

$$Info(D) = -\sum_{i=1}^{m} p_i \log_2(p_i)$$

- Information needed (after using A to split D into v partitions) to classify D:

$$Info_A(D) = \sum_{j=1}^{v} \frac{|D_j|}{|D|} \times Info(D_j)$$

- Information gained by branching on attribute A

$$Gain(A) = Info(D) - Info_A(D)$$

- Select the attribute with the highest information gain.

Attribute Selection: Information Gain (ID3)

- Class P: buys_computer = "yes"
- Class N: buys_computer = "no"

age	yes	no	I(yes, no)
<=30	2	3	0.971
31…40	4	0	0
>40	3	2	0.971

$$Info(D) = I(9,5)$$

$$= -\frac{9}{14}\log_2\left(\frac{9}{14}\right) - \frac{5}{14}\log_2\left(\frac{5}{14}\right)$$

$$=0.940$$

$$Info_{age}(D) = \frac{5}{14}I(2,3) + \frac{4}{14}I(4,0) + \frac{5}{14}I(3,2)$$

$$= 0.694$$

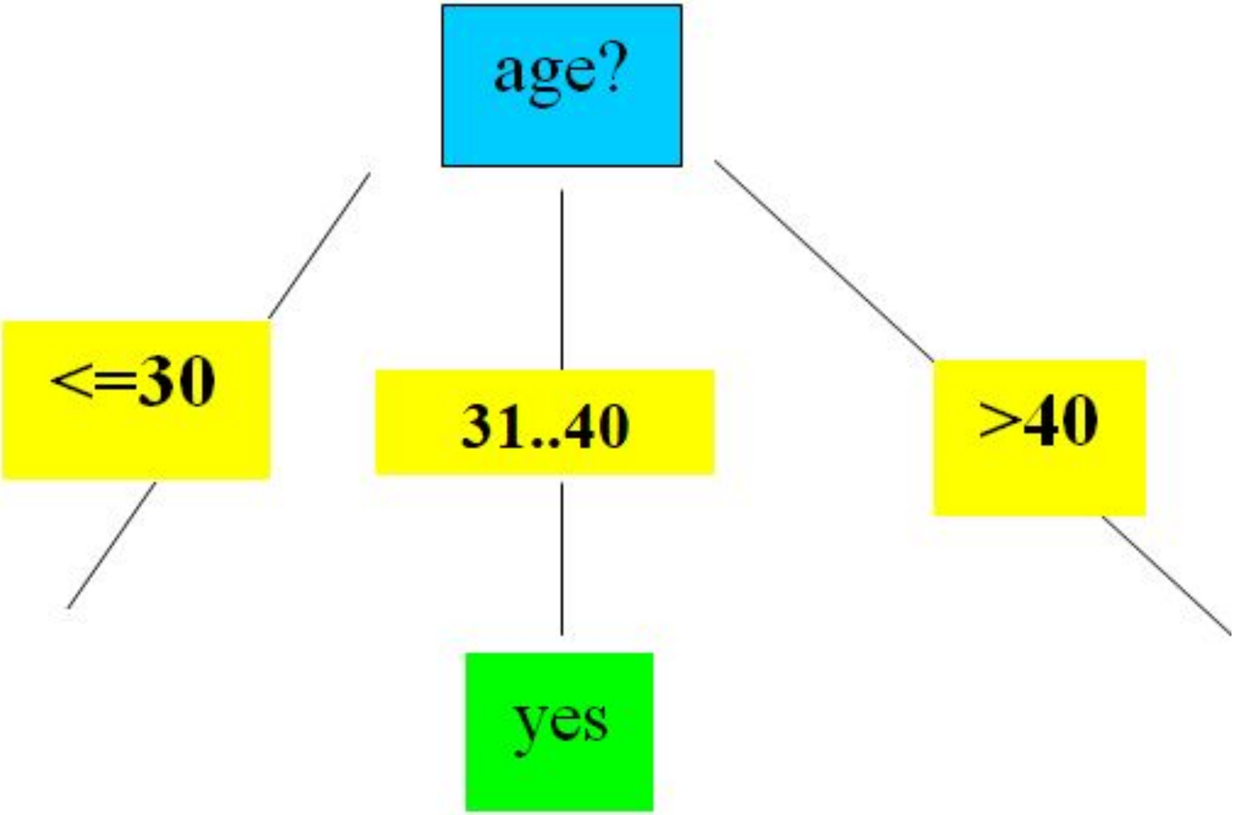
$\frac{5}{14}I(2,3)$ means "age <=30" has 5 out of 14 samples, with 2 yes'es and 3 no's.

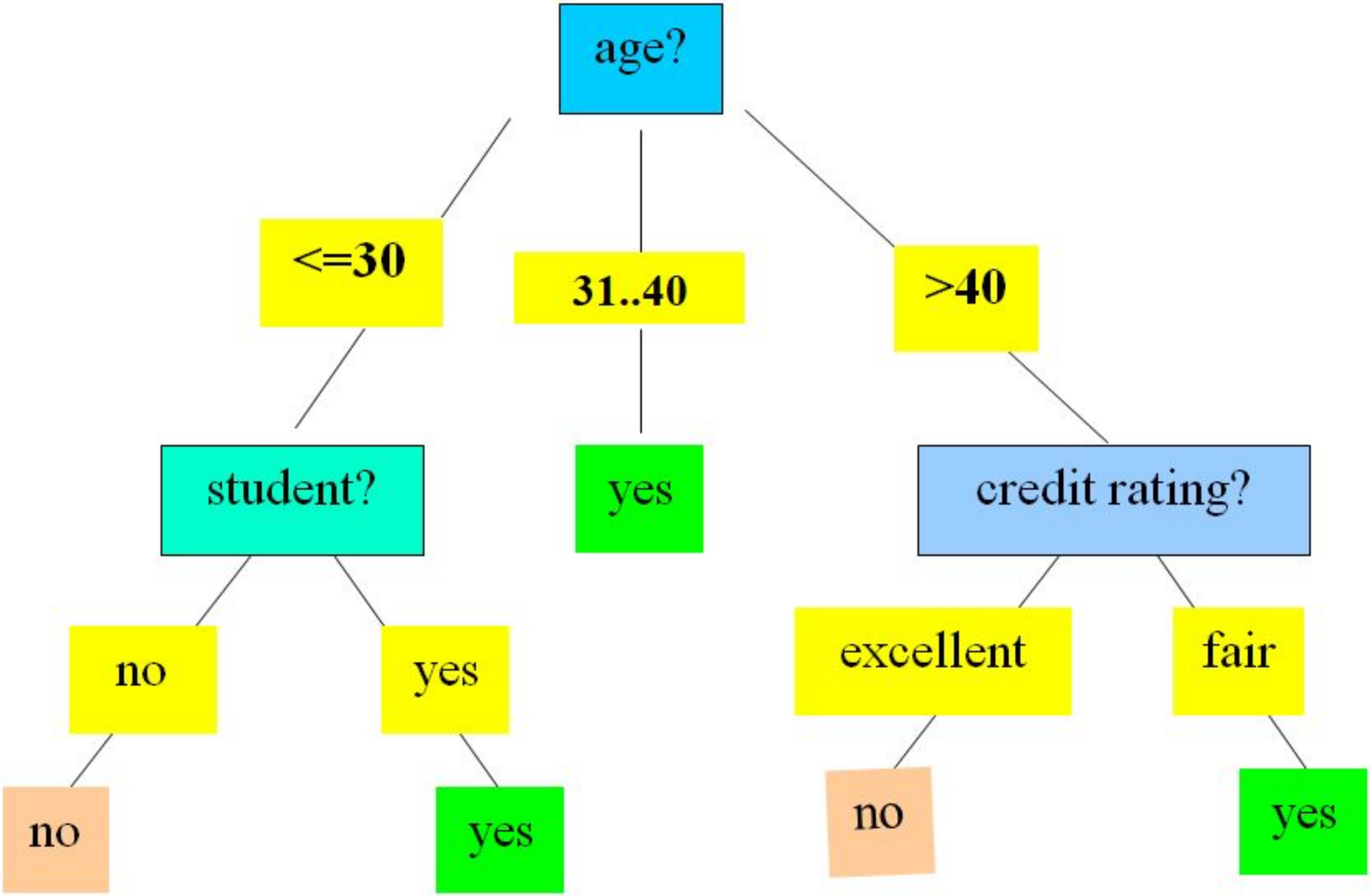
Hence,

$$Gain(age) = Info(D) - Info_{age}(D) = 0.246$$

Similarly,

$$Gain(income) = 0.029$$
$$Gain(student) = 0.151$$
$$Gain(credit\_rating) = 0.048$$


age?
<=30
31..40
>40
yes


age?
<=30
31..40
>40
student?
yes
credit rating?
no
yes
excellent
fair
no
yes
no
yes

11/1/2022

Overfitting and Tree Pruning

- Overfitting: A decision tree may overfit the training data.
- Two approaches to avoid overfitting:
  - Prepruning
  - Postpruning