Lecture 10
Review

GEOL 4397: Data analytics and machine learning for geoscientists

Jiajia Sun, Ph.D.

March 19th, 2019

UNIVERSITY of HOUSTON

YOU ARE THE PRIDE

EARTH AND ATMOSPHERIC SCIENCES

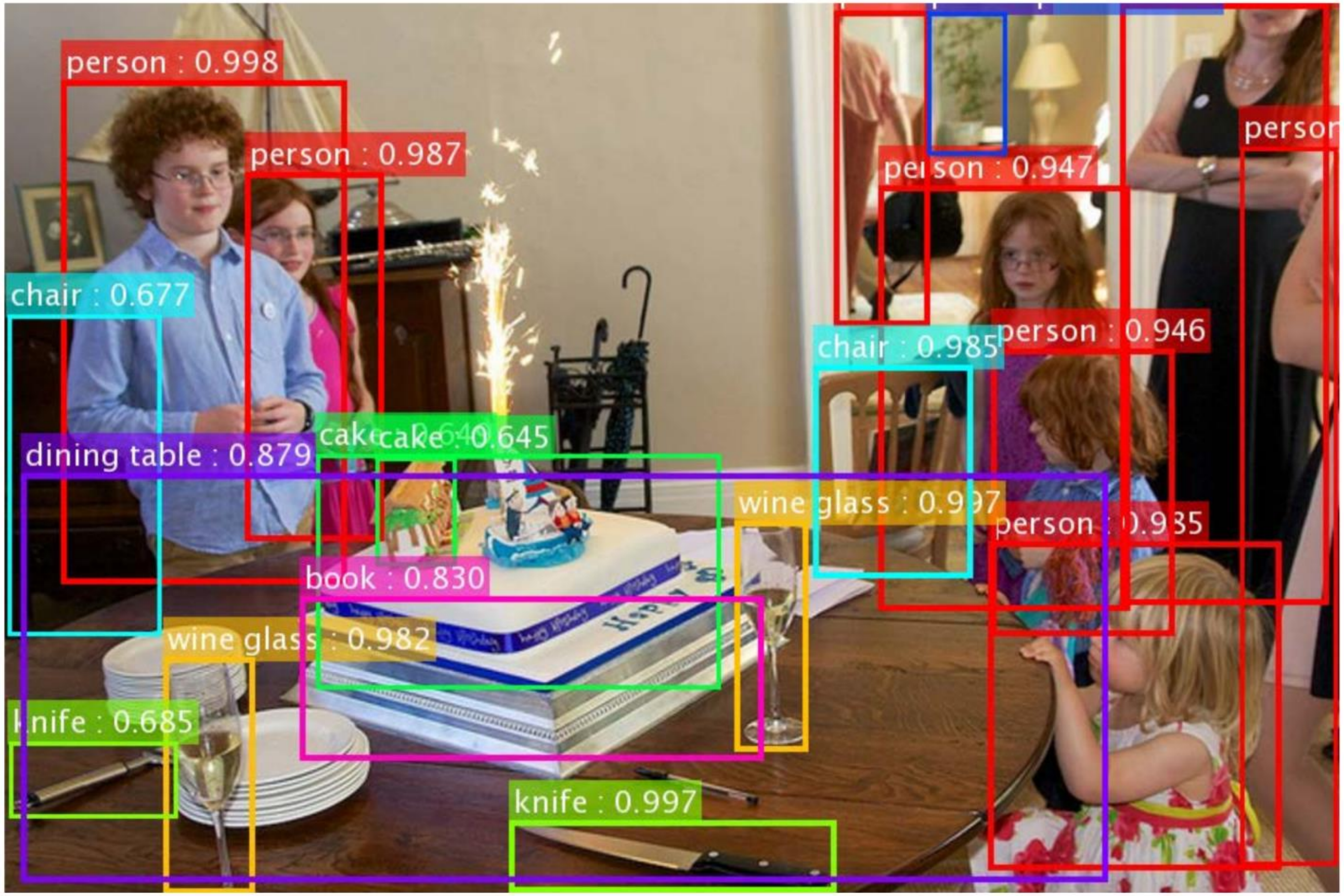
Announcement

- The lab report on ensemble learning due at 5:30 pm, March 21st.
- Exam on March 21st from 5:30 to 6:50 pm.

Outline

- Machine learning basics
  - Example applications
  - Definitions
- Training
  - Cost function
  - Optimization
- Optimization algorithms
  - Batch gradient descent
  - Stochastic gradient descent
  - Mini-batch gradient descent
- Types of machine learning
  - Supervised vs. unsupervised
  - Regression vs. Classification
- Overfit vs Underfit
  - Diagnose
  - Remedy

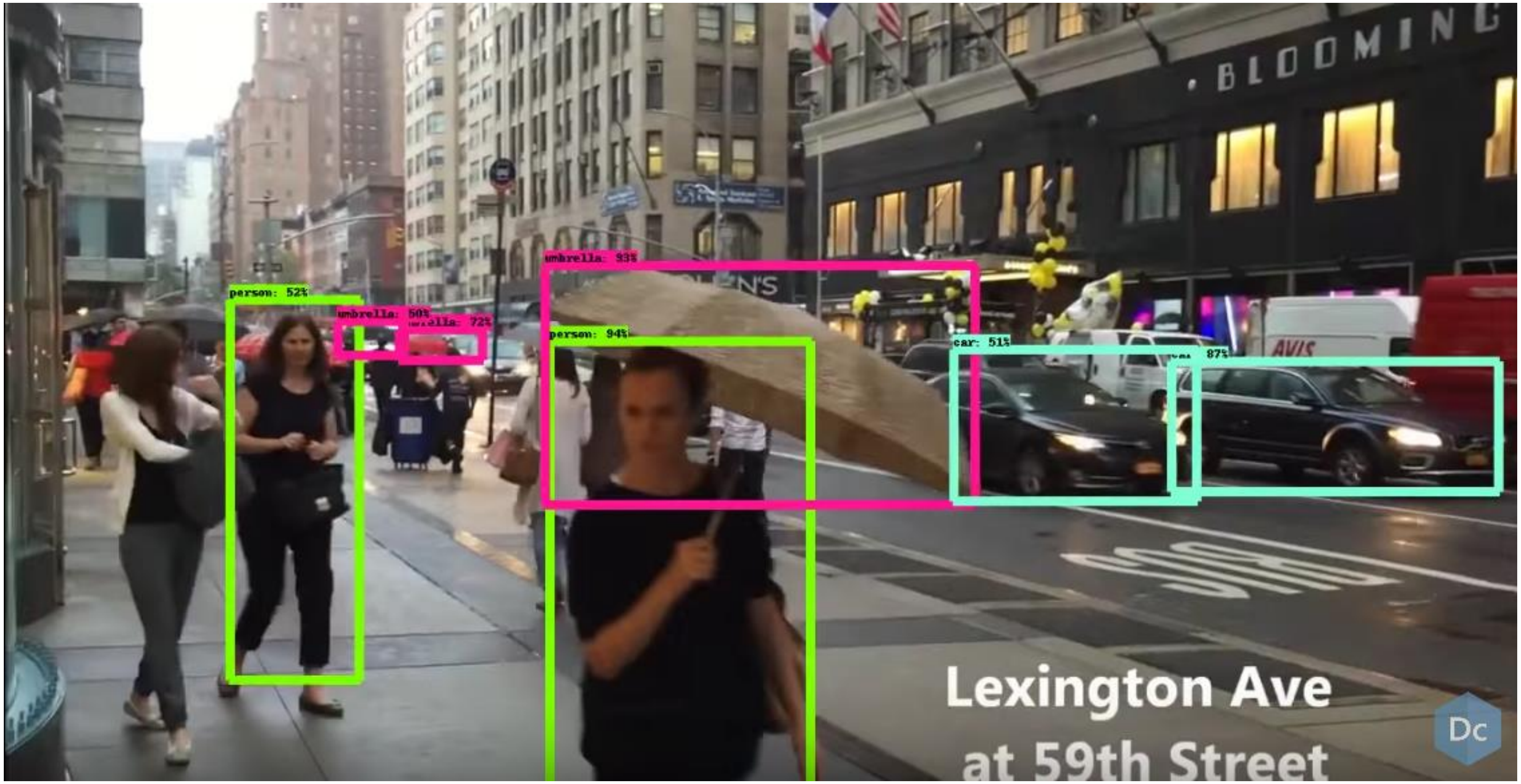
Object detection



ResNet applied to COCO dataset.

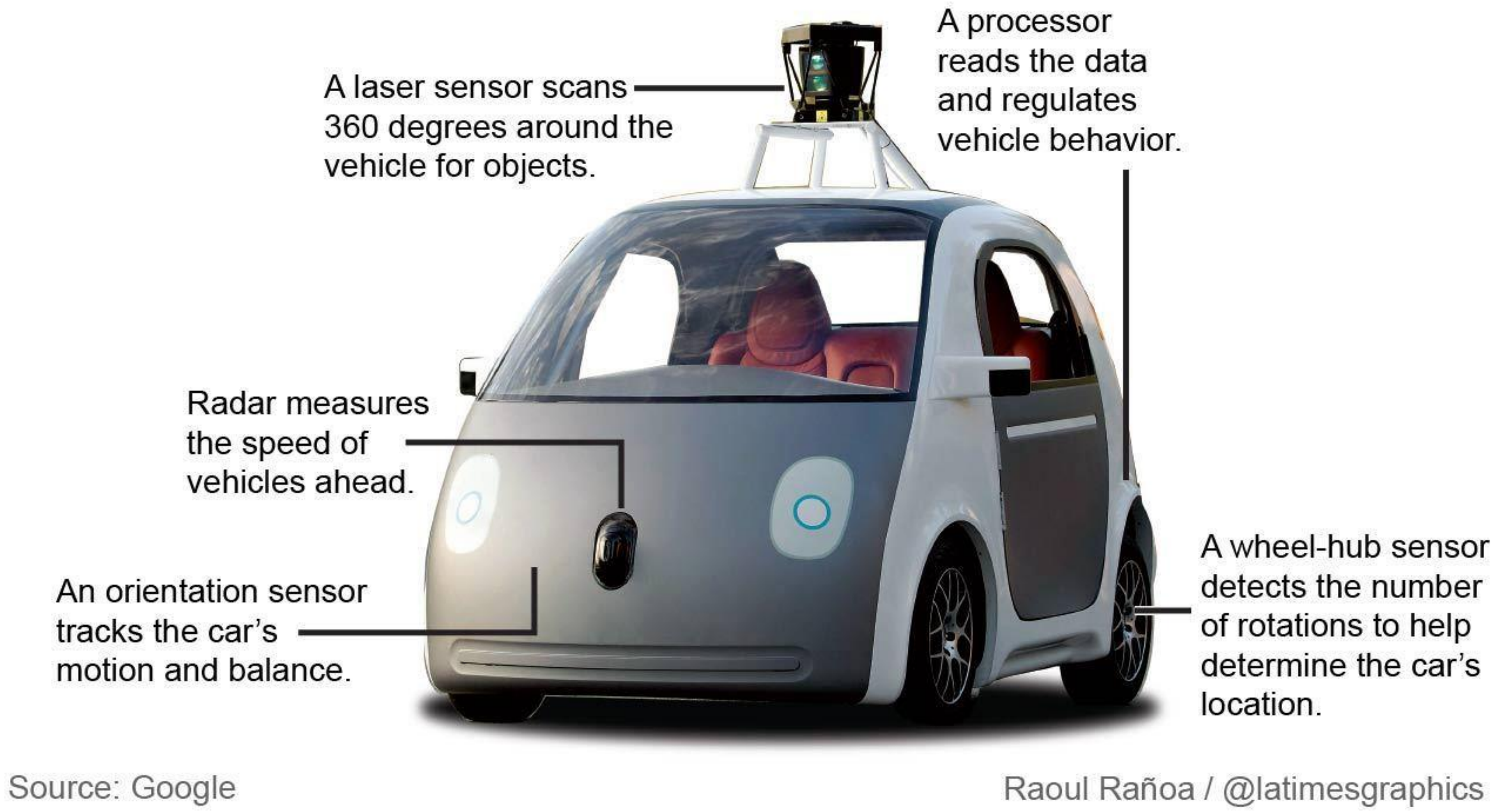
Source: He et al., Deep residual learning for image recognition, CVPR, 2016

Real time object detection



Video online: https://www.youtube.com/watch?v=_zZe27JYi8Y

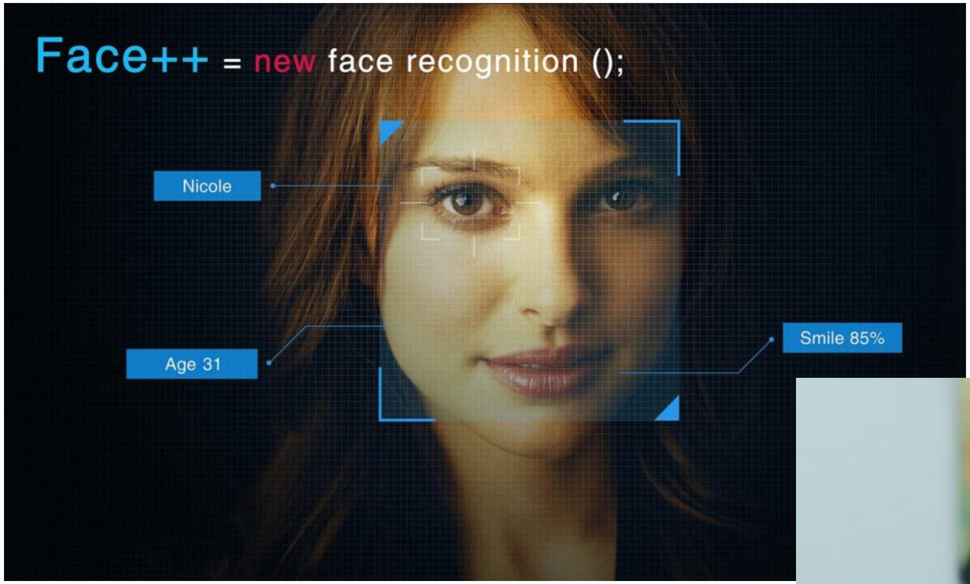
Self-driving car



Voice recognition

Source: https://biostore.co.uk/company/news-articles/voice-recognition-biometrics-making-noise/

Face recognition



Source: https://www.pinterest.com/pin/135600638759424941/?lp=true

Hand written digit recognition

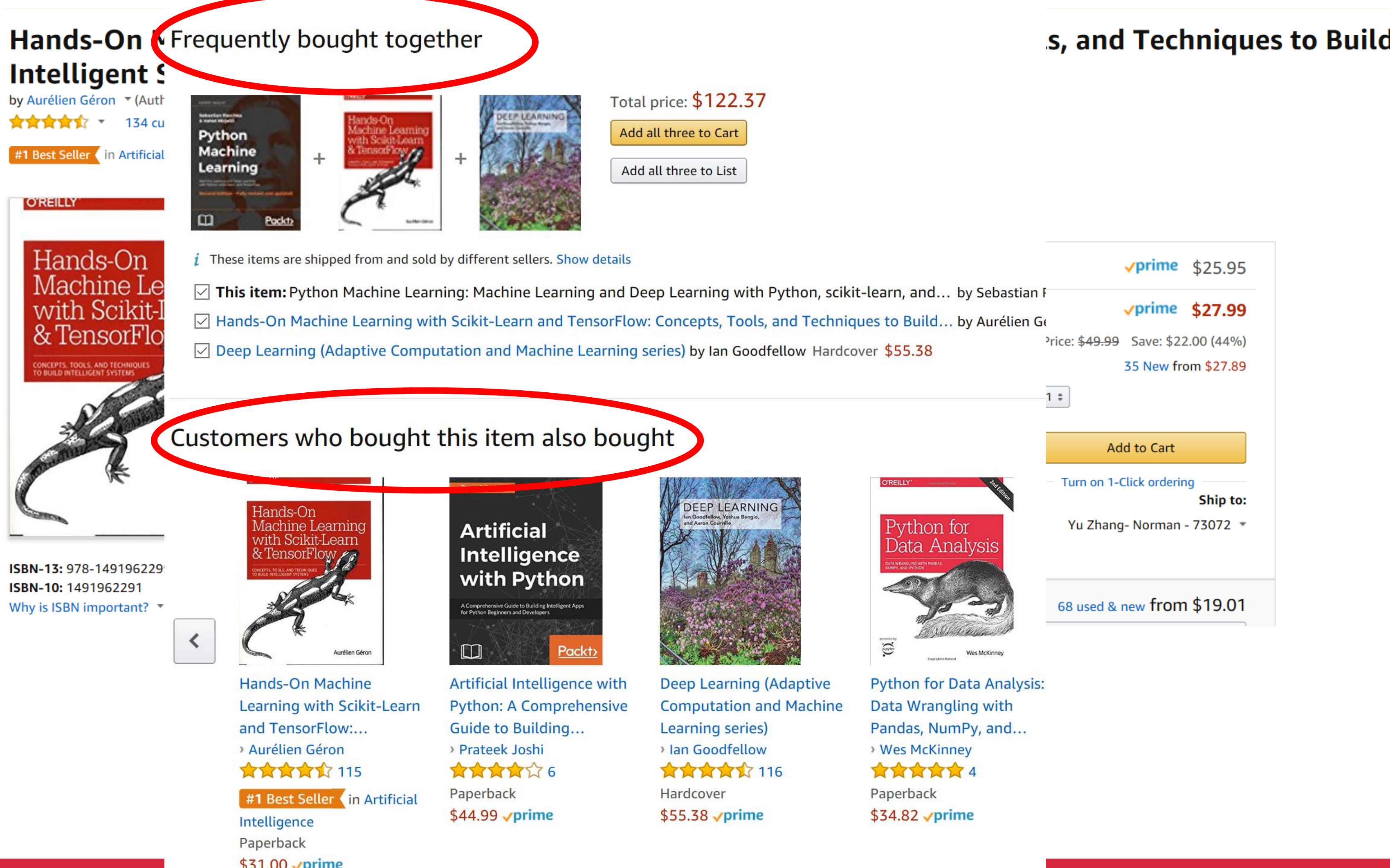
MNIST data set

Spam filer

Fraud detection

Source: http://tsigroup.com

Recommender system



Go game

Image credit: Nature

Image credit: theverge.com

What is machine learning?

- the field of study that gives computers the ability to learn from data (e.g., discovering patterns and relations among input data), and make predictions.

Outline

- Machine learning basics
  - Example applications
  - Definitions
- Training
  - Cost function
  - Optimization
- Optimization algorithms
  - Batch gradient descent
  - Stochastic gradient descent
  - Mini-batch gradient descent
- Types of machine learning
  - Supervised vs. unsupervised
  - Regression vs. Classification
- Overfit vs Underfit
  - Diagnose
  - Remedy

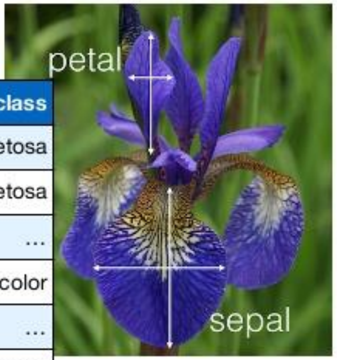
Nomenclature

IRIS

https://archive.ics.uci.edu/ml/datasets/Iris

Instances (samples, observations)

	sepal_length	sepal_width	petal_length	petal_width	class
1	5.1	3.5	1.4	0.2	setosa
2	4.9	3.0	1.4	0.2	setosa
...	...	...	...	...	...
50	6.4	3.2	4.5	1.5	veriscolor
...	...	...	...	...	...
150	5.9	3.0	5.1	1.8	virginica



Features (attributes, dimensions)

Classes (targets)

$$\mathbf{X} = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1D} \\ x_{21} & x_{22} & \cdots & x_{2D} \\ x_{31} & x_{32} & \cdots & x_{3D} \\ \vdots & \vdots & \ddots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ x_{N1} & x_{N2} & \cdots & x_{ND} \end{bmatrix}$$

$$\mathbf{y} = [y_1, y_2, y_3, \cdots y_N]$$

https://www.slideshare.net/SebastianRaschka/nextgen-talk-022015

A simple example

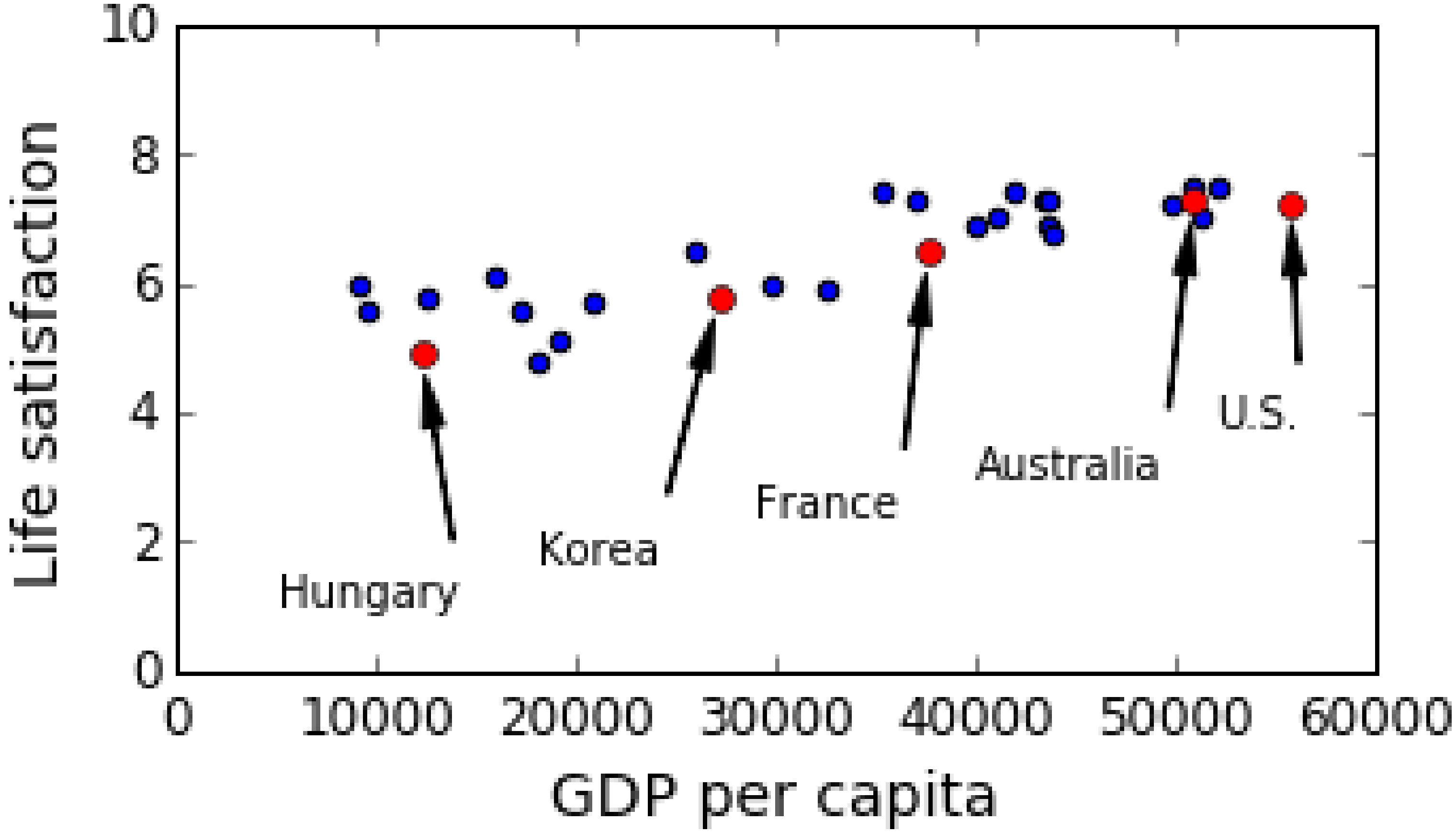


Figure from Aurelien Geron's ML book, page 19

Goal

- Learn a model from the data

Goal

- Build a model from the data

Goal

- Train a model from the data

Goal

- Train a model from the training data

Goal

- Train a model from the training data
- Make predictions

Training/Learning

- Which ones to choose?
- Or, a more general question is, How to train/learn these model parameters??
- The answer is by

defining a cost function

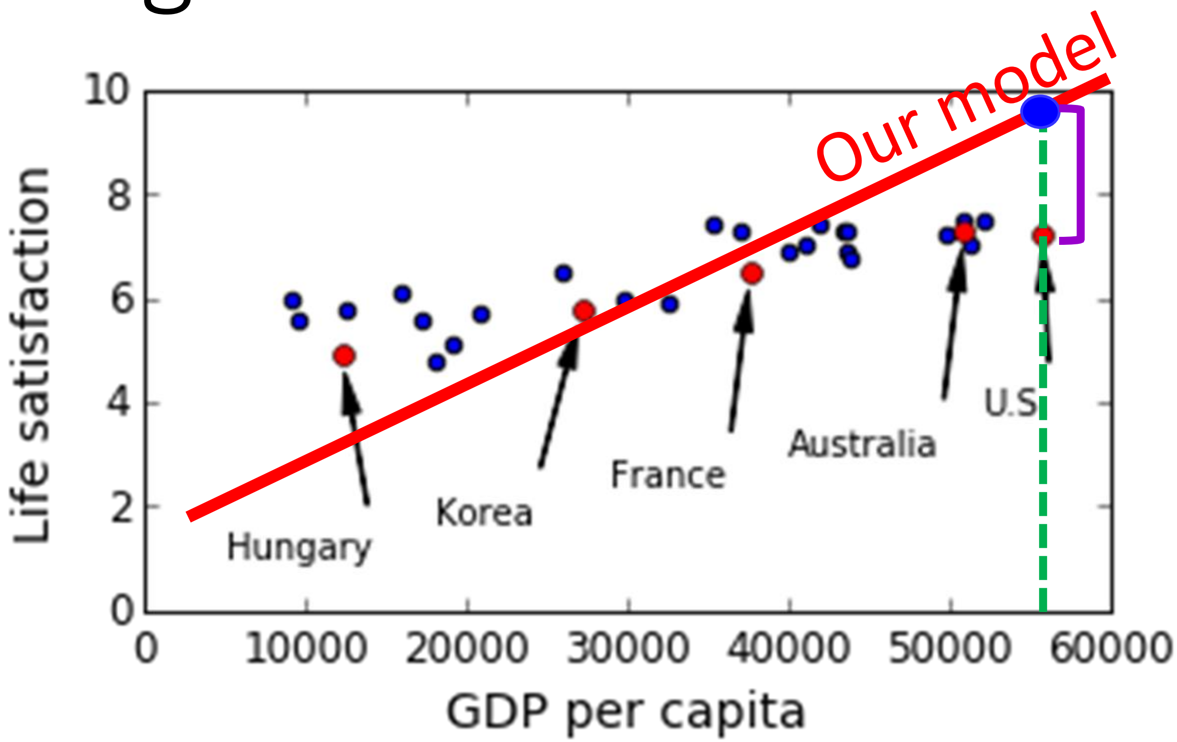
&

minimizing it

Cost function: basic idea

- Measures how bad (or good) a candidate model is
- Specifically, measure the difference between predictions from our model and the training data
- The objective is to minimize the cost function so as to minimize the difference between predictions and observations

Building a cost function



For simplicity, let us focus on one country, say, U.S.

- What is the predicted value for life satisfaction?
- What is the value from training data?

Difference between predicted and true values

- In our training data, we have M countries. Suppose U.S. is the i^{th} country.
- Predicted value: $h_\theta(x^{(i)})$
- True value: $y^{(i)}$
- Difference:

$$(h_\theta(x^{(i)}) - y^{(i)})^2$$

- This is only for U.S.
- Remember that we want our model to fit all of our training data, not just one of them. Therefore, we sum the differences over all countries

$$\sum_{i=1}^{M}(h_\theta(x^{(i)}) - y^{(i)})^2$$

Cost function

$$\sum_{i=1}^{M}(h_\theta(x^{(i)}) - y^{(i)})^{2}$$

Remember

$$h_\theta(x^{(i)}) = \theta_0 + \theta_1 x^{(i)}$$

Cost function

$$\sum_{i=1}^{M}\left(h_\theta(x^{(i)}) - y^{(i)}\right)^2$$

Remember

$$h_\theta(x^{(i)}) = \theta_0 + \theta_1 x^{(i)}$$

Therefore,

$$\sum_{i=1}^{M}\left(\theta_0 + \theta_1 x^{(i)} - y^{(i)}\right)^2$$

Cost function

$$\sum_{i=1}^{M} (h_\theta(x^{(i)}) - y^{(i)})^{2}$$

Remember

$$h_\theta(x^{(i)}) = \theta_0 + \theta_1 x^{(i)}$$

Therefore,

$$J(\theta_0, \theta_1) = \sum_{i=1}^{M} (\theta_0 + \theta_1 x^{(i)} - y^{(i)})^{2}$$

Minimization

- Cost function measures the difference between predicted and true values
- Remember that, we want to minimize this difference, i.e.,

$$\min\ J(\theta_0, \theta_1) = \sum_{i=1}^{M} (\theta_0 + \theta_1 x^{(i)} - y^{(i)})^2$$

- Learning/training = Minimizing a cost function

- The process of learning a model from training data is essentially the process of minimizing a cost function.

Optimization

- Cost function measures the difference between predicted and true values
- Remember that, we want to minimize this difference, i.e.,

$$\min \ J(\theta_0, \theta_1) = \sum_{i=1}^{M} (\theta_0 + \theta_1 x^{(i)} - y^{(i)})^2$$

- Learning/training = Minimizing a cost function

- The process of learning a model from training data is essentially the process of minimizing a cost function.
- Optimization: finding optimal parameter values that minimize a cost function

Optimization

- Cost function measures the difference between predicted and true values
- Remember that, we want to minimize this difference, i.e.,

$$\min \ J(\theta_0, \theta_1) = \sum_{i=1}^{M} (\theta_0 + \theta_1 x^{(i)} - y^{(i)})^2$$

- $\boxed{\text{Learning/training = Minimizing a cost function}}$

- The process of learning a model from training data is essentially the process of optimization.
- Optimization: finding optimal parameter values that minimize a cost function

Best fit model

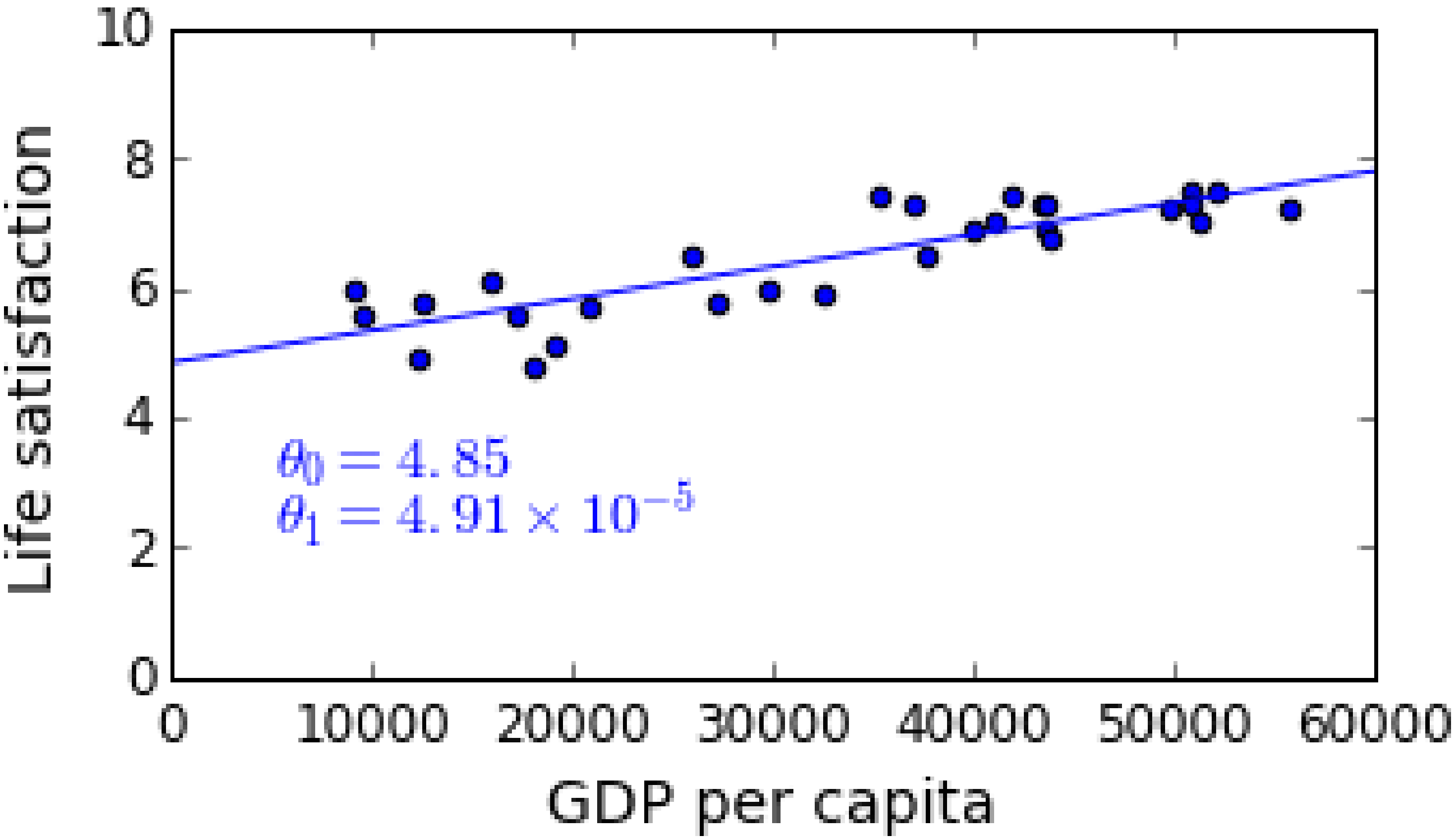


Figure from Aurelien Geron's ML book, page 20

Outline

- Machine learning basics
  - Example applications
  - Definitions
- Training
  - Cost function
  - Optimization
- Optimization algorithms
  - Batch gradient descent
  - Stochastic gradient descent
  - Mini-batch gradient descent
- Types of machine learning
  - Supervised vs. unsupervised
  - Regression vs. Classification
- Overfit vs Underfit
  - Diagnose
  - Remedy

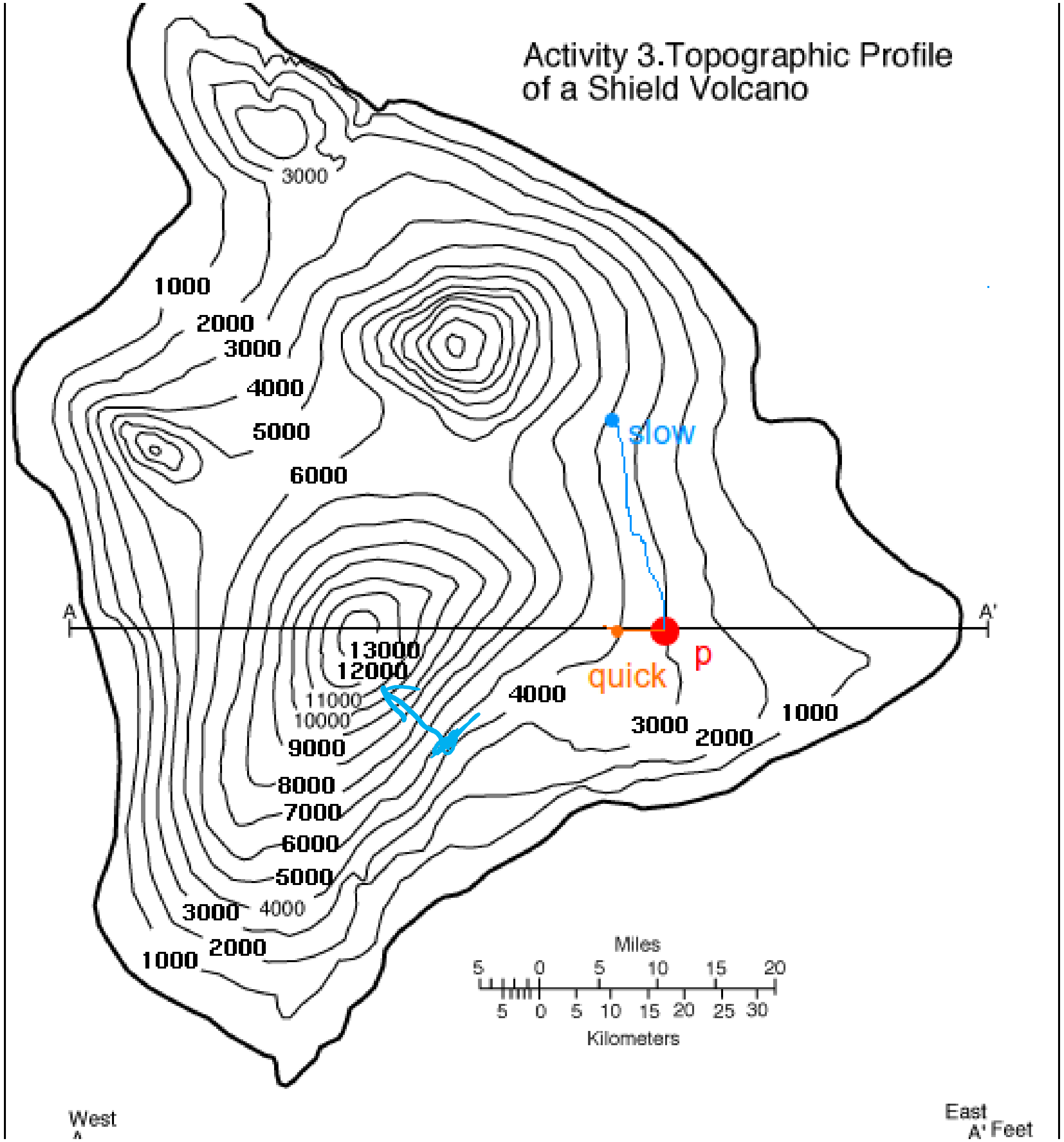
Gradient

- Gradient of a function $f(x, y)$ is defined as

$$\nabla f(x, y) = \begin{bmatrix} \dfrac{\partial f}{\partial x} \\ \\ \dfrac{\partial f}{\partial y} \end{bmatrix}$$

Understanding gradient

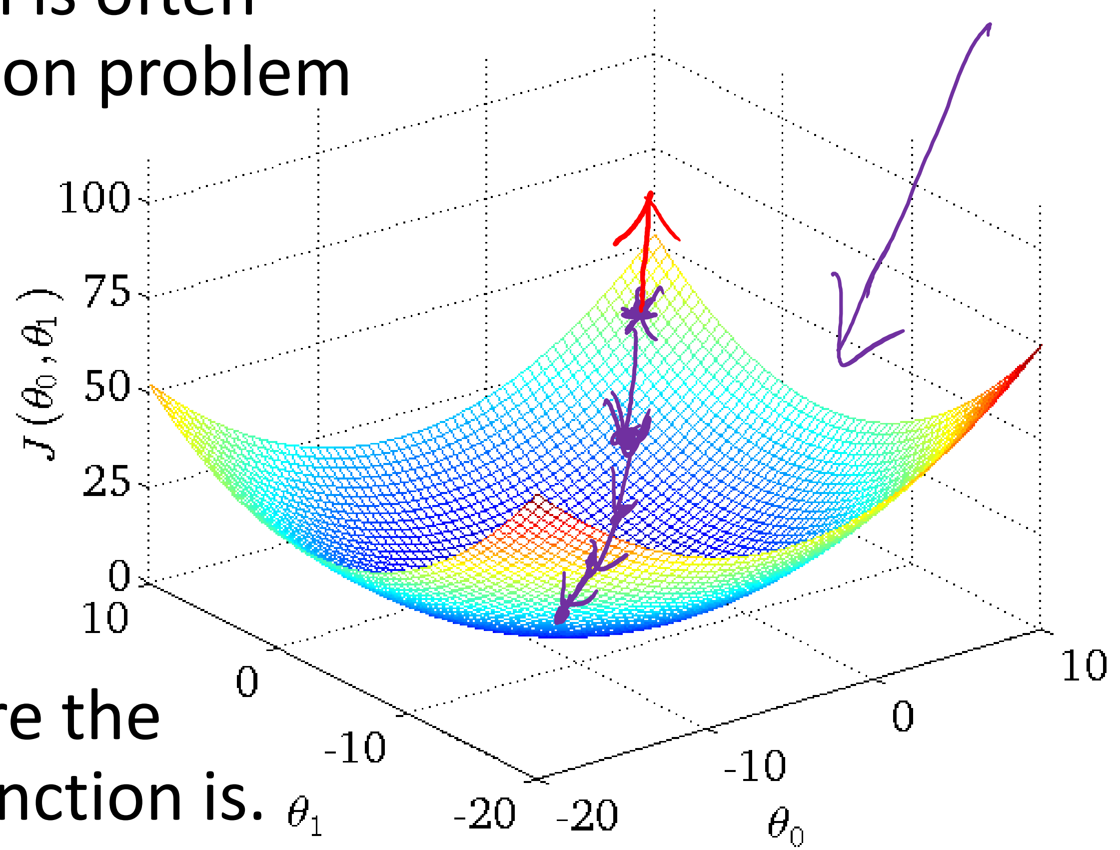
- Consider the topography as a 2D function $f(x, y)$
- The gradient direction tells you the fastest way up



Picture taken from https://mathoverflow.net/questions/1977/why-is-the-gradient-normal

Gradient in the context of optimization

- Optimization problem is often posed as a minimization problem



- We want to find where the minimum of a cost function is.

Picture taken from Andrew Ng's Machine Learning class on Coursera.org

Gradient descent algorithm

- Given initial values $\boldsymbol{\theta}^{(0)} = \left[\theta_0^{(0)}, \theta_1^{(0)}\right]$
- While (not convergence):

$$\boldsymbol{\theta}^{(j)} = \boldsymbol{\theta}^{(j-1)} - \alpha \nabla J(\boldsymbol{\theta}^{(j-1)})$$

Gradient descent algorithm

- Given initial values $\boldsymbol{\theta}^{(0)} = \left[\theta_0^{(0)}, \theta_1^{(0)}\right]$
- While (not convergence):

$$\theta_0^{(j)} = \theta_0^{(j-1)} - \alpha \frac{\partial}{\partial \theta_0} \mathrm{J}(\theta_0^{(j-1)}, \theta_1^{(j-1)})$$
$$\theta_1^{(j)} = \theta_1^{(j-1)} - \alpha \frac{\partial}{\partial \theta_1} \mathrm{J}(\theta_0^{(j-1)}, \theta_1^{(j-1)})$$

Gradient descent algorithm for linear regression

- Given initial values $\boldsymbol{\theta}^{(0)} = \left[\theta_0^{(0)}, \theta_1^{(0)}\right]$
- While (not convergence):

$$\theta_0^{(j)} = \theta_0^{(j-1)} - \alpha \frac{\partial}{\partial \theta_0} \mathrm{J}(\theta_0^{(j-1)}, \theta_1^{(j-1)})$$

$$\theta_1^{(j)} = \theta_1^{(j-1)} - \alpha \frac{\partial}{\partial \theta_1} \mathrm{J}(\theta_0^{(j-1)}, \theta_1^{(j-1)})$$

$$\min \quad J(\theta_0, \theta_1) = \frac{1}{2} \sum_{i=1}^{M} (\theta_0 + \theta_1 x^{(i)} - y^{(i)})^2$$

Gradient descent algorithm for linear regression

- Given initial values $\boldsymbol{\theta}^{(0)} = \left[\theta_0^{(0)}, \theta_1^{(0)}\right]$
- While (not convergence):

$$\theta_0^{(j)} = \theta_0^{(j-1)} - \alpha \sum_{i=1}^{M} (\theta_0 + \theta_1 x^{(i)} - y^{(i)})$$

$$\theta_1^{(j)} = \theta_1^{(j-1)} - \alpha \sum_{i=1}^{M} (\theta_0 + \theta_1 x^{(i)} - y^{(i)}) x^{(i)}$$

- Each step of gradient descent uses ALL the training examples.

Batch gradient descent

- Each step of gradient descent uses ALL the training examples.

Problem with batch gradient descent

- When the number of training data is huge, say, M = 300,000,000, batch gradient descent becomes very slow.

Gradient descent algorithm

- Given initial values $\boldsymbol{\theta}^{(0)} = \left[\theta_0^{(0)}, \theta_1^{(0)}\right]$
- While (not convergence):

$$\theta_0 = \theta_0 - \alpha \frac{1}{M} \sum_{i=1}^{M} (\theta_0 + \theta_1 x^{(i)} - y^{(i)})$$

$$\theta_1 = \theta_1 - \alpha \frac{1}{M} \sum_{i=1}^{M} (\theta_0 + \theta_1 x^{(i)} - y^{(i)}) x^{(i)}$$

Stochastic gradient descent

- Given initial values $\boldsymbol{\theta}^{(0)} = \left[\theta_0^{(0)}, \theta_1^{(0)}\right]$

  Repeat {

  Shuffle the training set

  For $i = 1, \ldots, m$ {

  $$\theta_0 = \theta_0 - \alpha\left(\theta_0 + \theta_1 x^{(i)} - y^{(i)}\right)$$

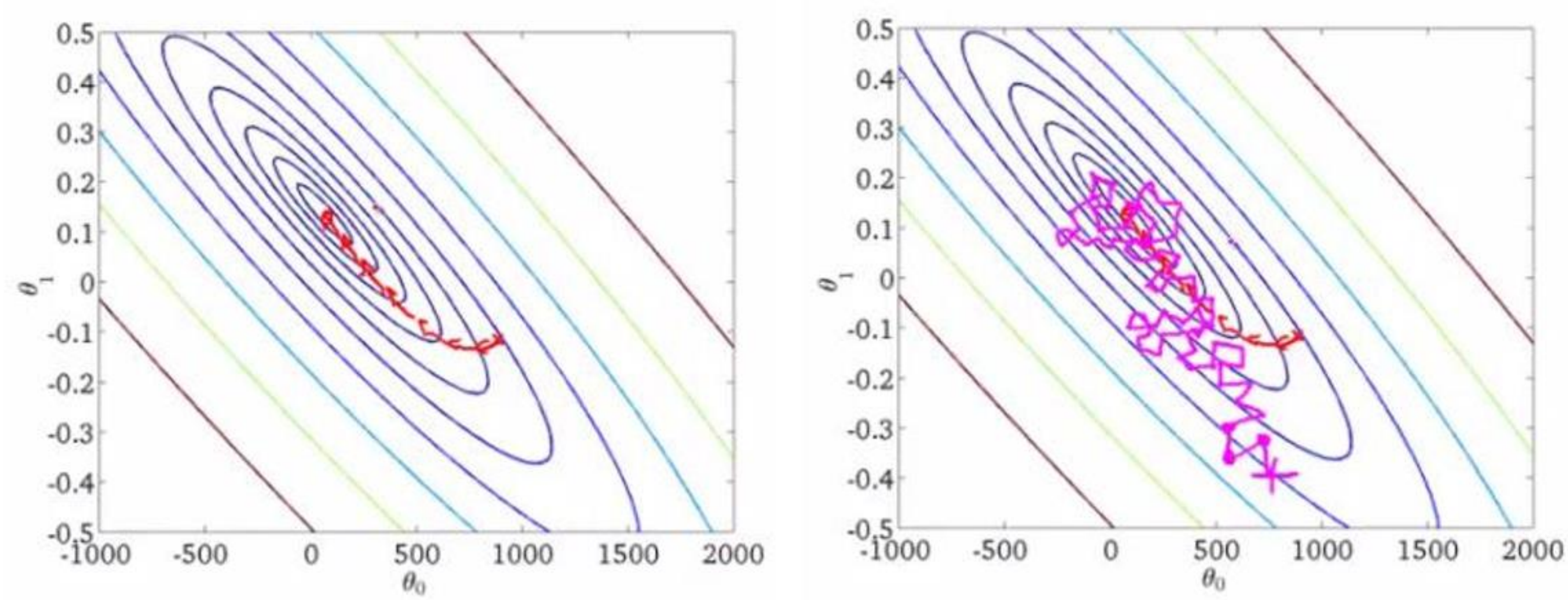
  $$\theta_1 = \theta_1 - \alpha(\theta_0 + \theta_1 x^{(i)} - y^{(i)})\, x^{(i)}$$

  }

  }

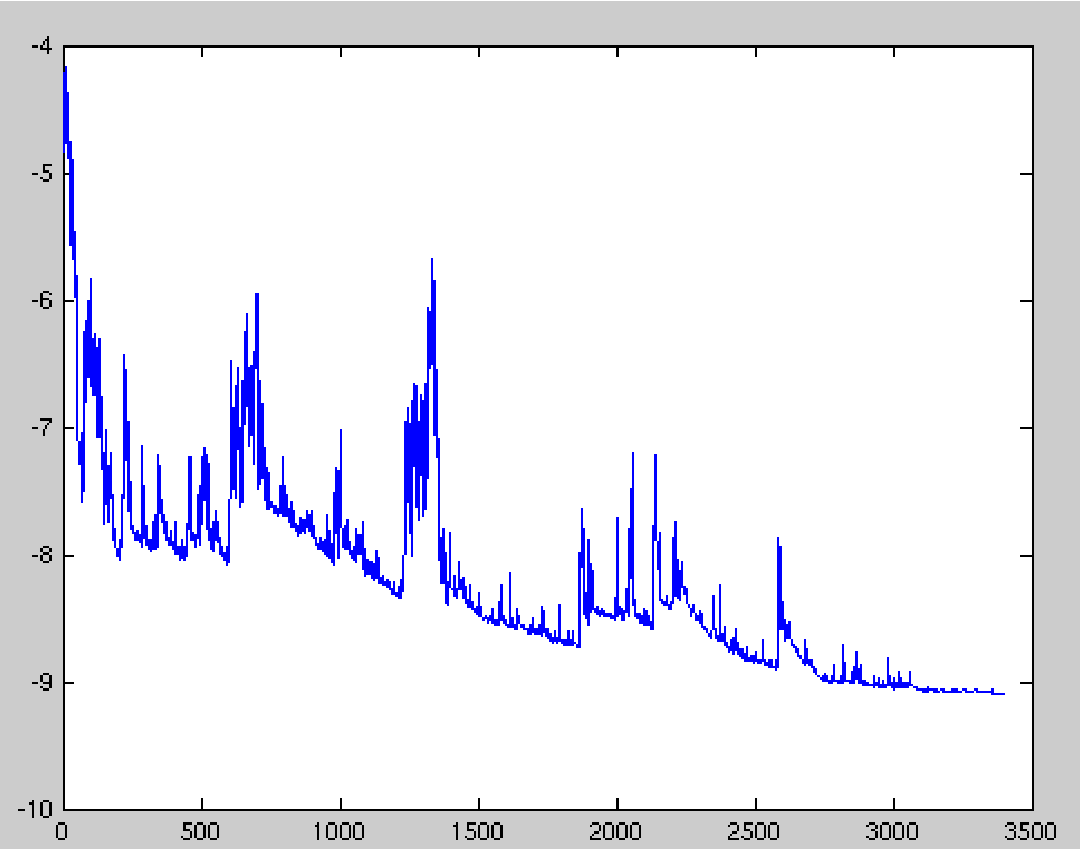
SGD vs BGD

- SGD much faster
- Search path very irregular
- Cost function bounces up and down, decreasing only on average
- Over time, it ends up close to minimum, but never settles down.



Picture taken from
https://www.cs.cmu.edu/~yuxiangw/docs/SSGD.pdf

SGD cost function



https://upload.wikimedia.org/wikipedia/commons/f/f3/Stogra.png

Mini-batch gradient descent

- Mini-batch uses a small number ($1 < \# < M$)of training examples to update model parameters

Batch vs stochastic vs Mini-batch

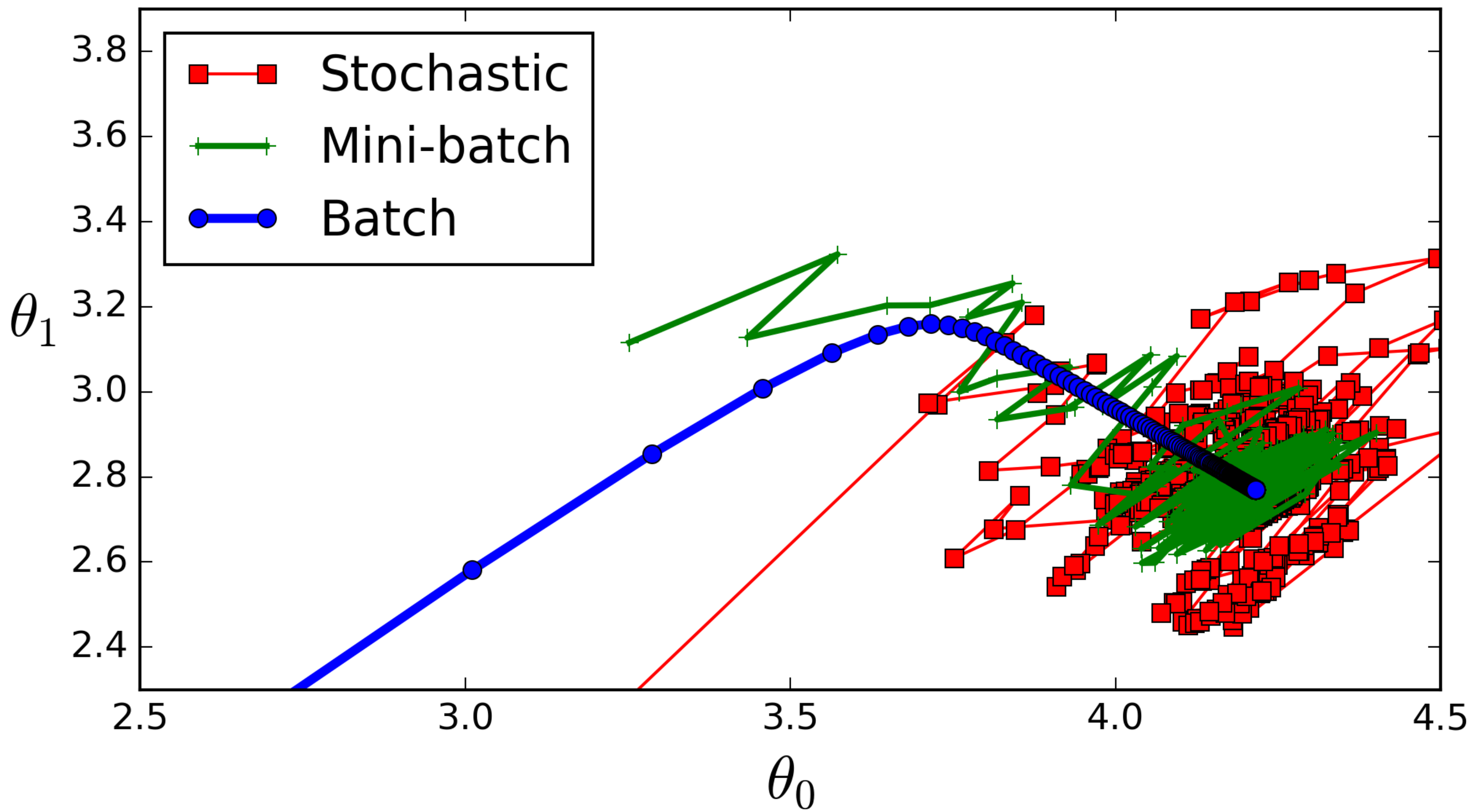


Figure from Aurelien Geron's ML book, page 120

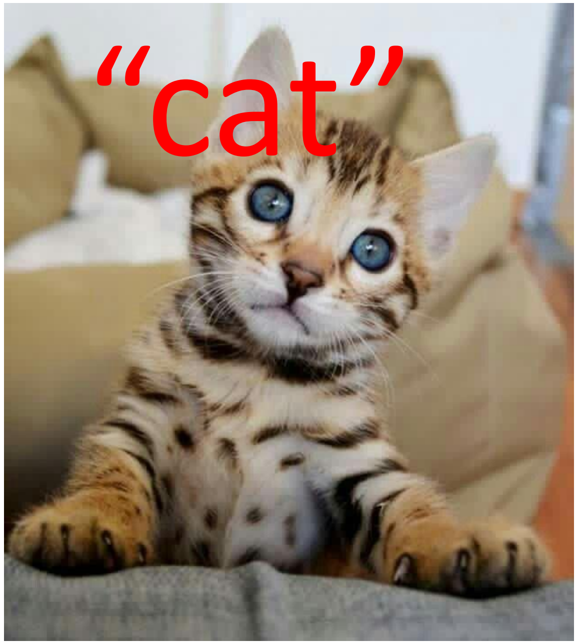
Outline

- Machine learning basics
  - Example applications
  - Definitions
- Training
  - Cost function
  - Optimization
- Optimization algorithms
  - Batch gradient descent
  - Stochastic gradient descent
  - Mini-batch gradient descent
- Types of machine learning
  - Supervised vs. unsupervised
  - Regression vs. Classification
- Overfit vs Underfit
  - Diagnose
  - Remedy

Supervised learning

- Training data come with labels (i.e., answers)
- Also termed labeled data
- E.g., cat classifier


"cat"


"cat"

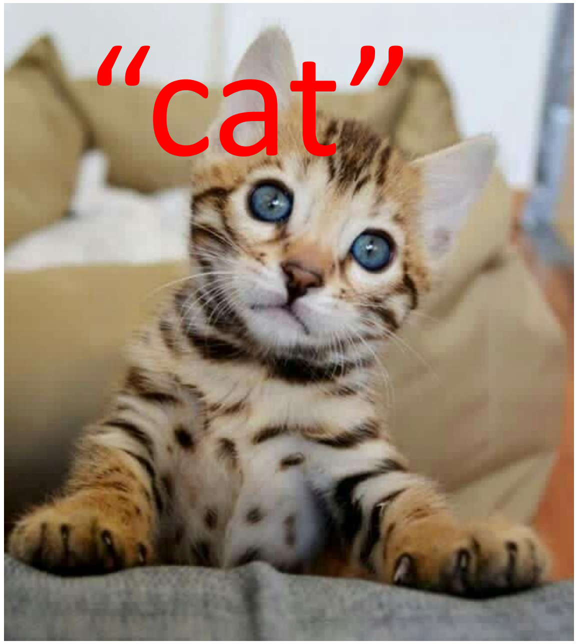

"cat"


"cat"


“cat”


“cat”


“cat”


“cat”


“non-cat”


“non-cat”


“non-cat”

Supervised learning: what is it?

- Let us consider each image as an input variable, x
- Also, consider each label as an output variable, y
- Supervised learning is all about learning a mapping function from the input to the output

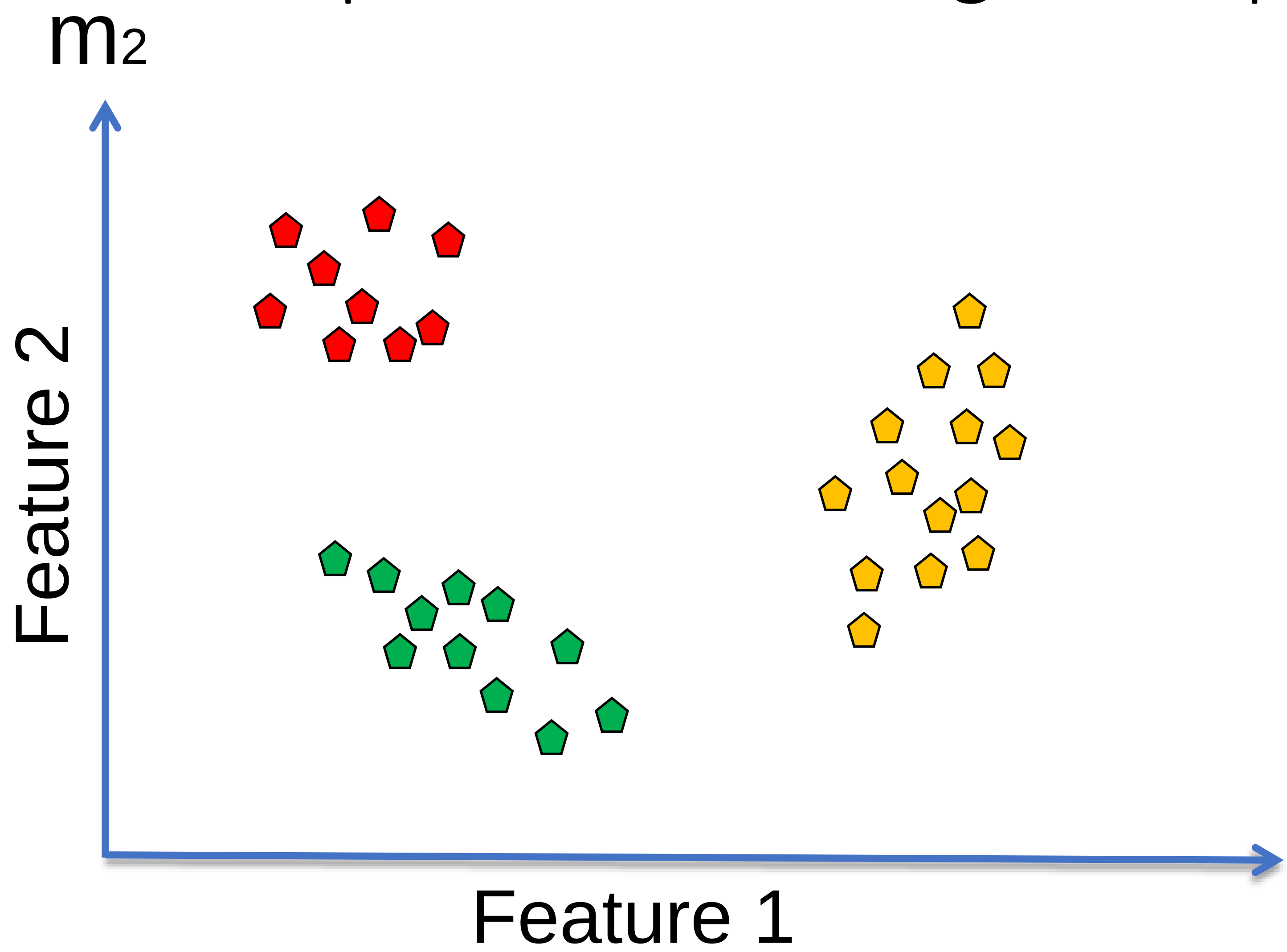
$$y = f(x)$$

- So that, given a new image, x, your model learning model can predict y.

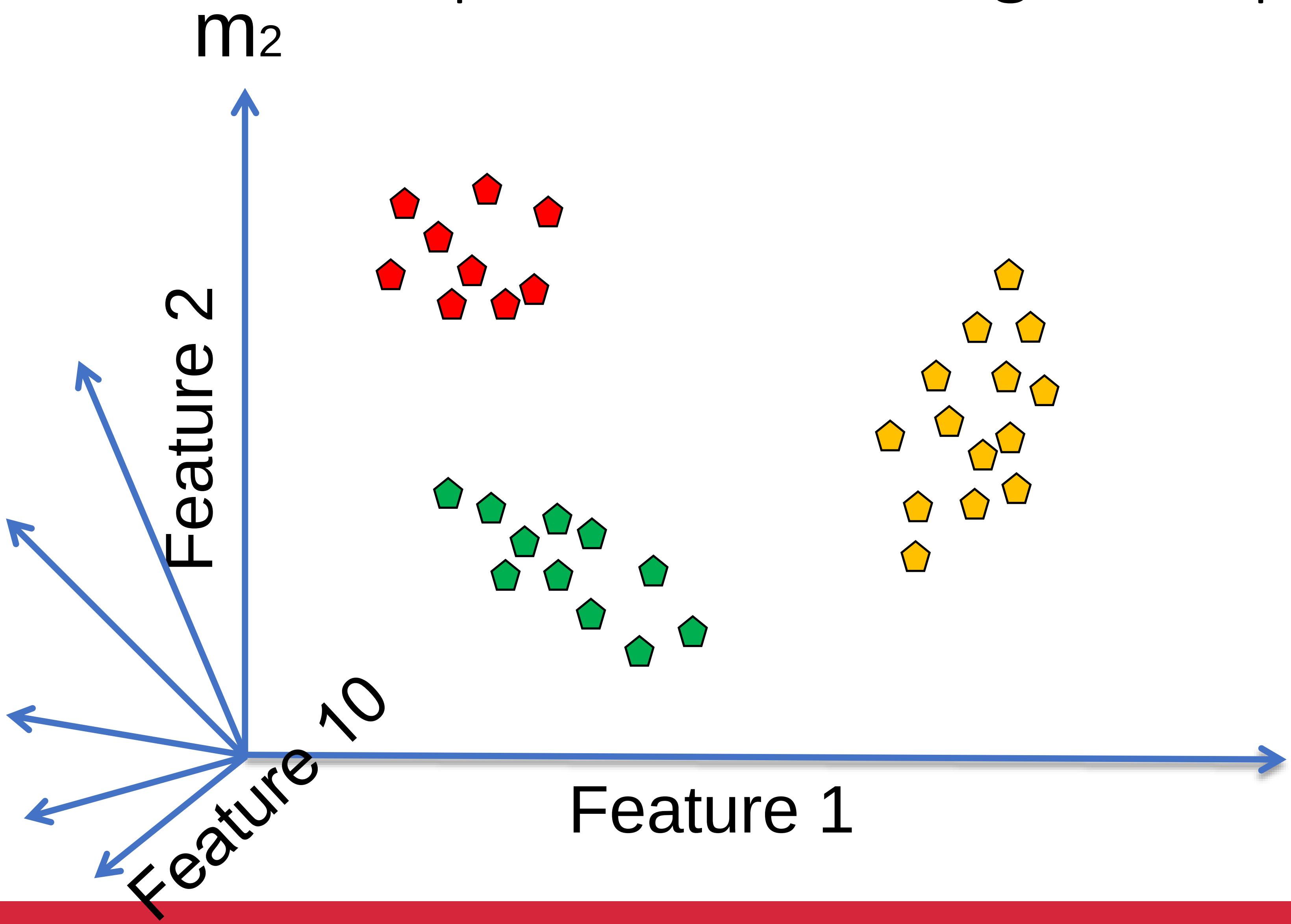
Unsupervised learning

- Training data does not have any label
- Unlabeled data
- The goal is to discover the intrinsic, and often complicated structures among data for better decision-making.
- No answer available. Algorithms are left to their own to discover the interesting structures in data.

Unsupervised learning: example

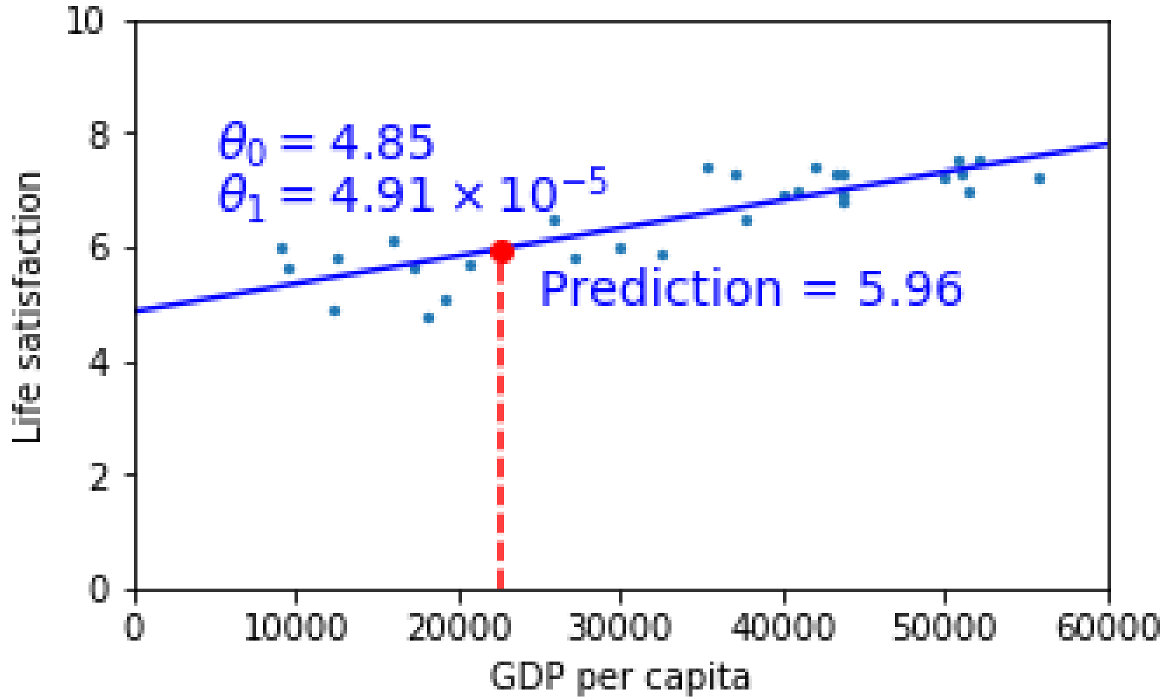


Unsupervised learning: example



Regression

- Predict continuous numerical values, such as prices, temperatures, etc.

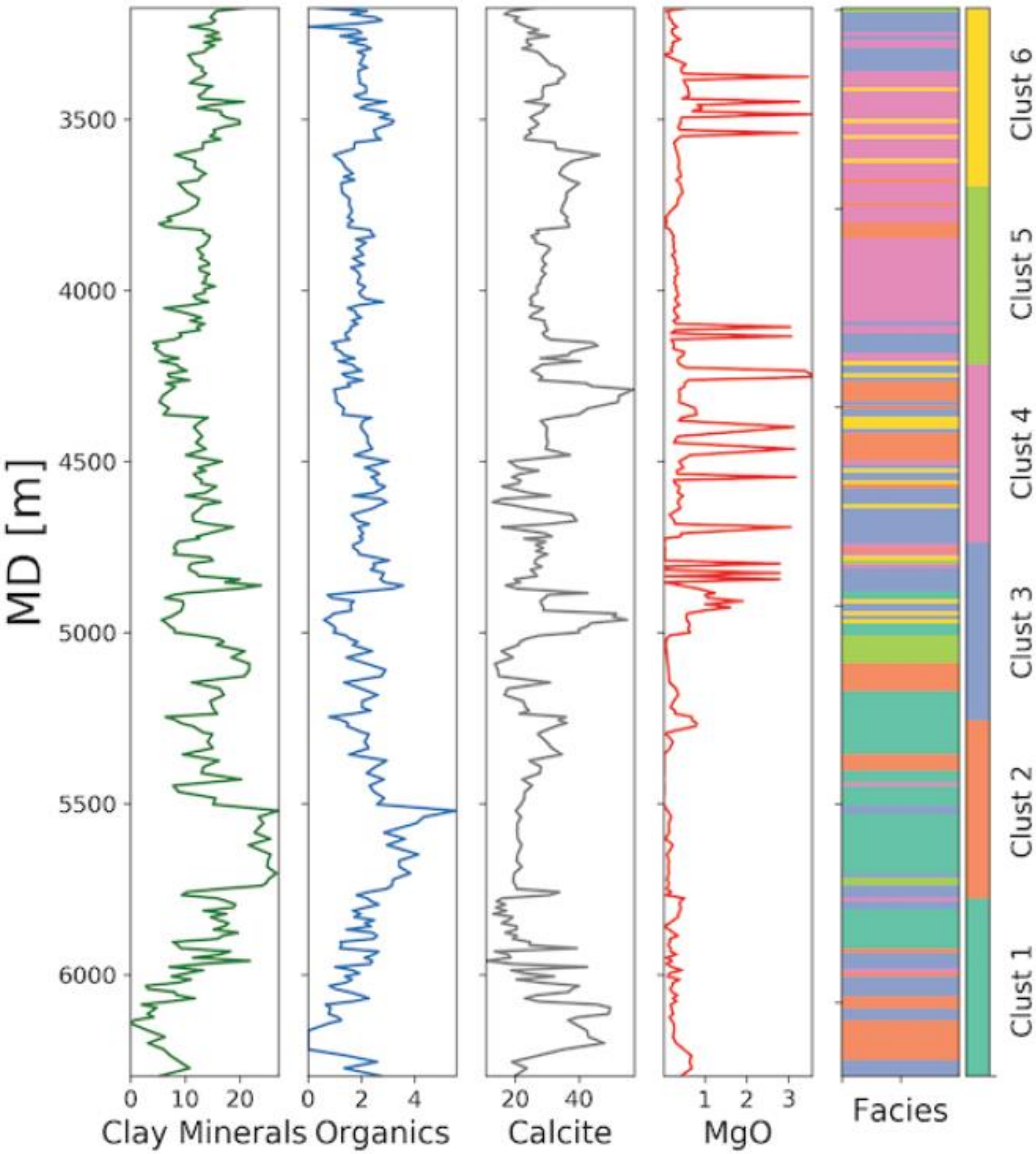


Classification

- Predict discrete categorical values, such as class 1, 2, 3, etc.

Clas

- Pred class 1, 2, 3,



Outline

- Machine learning basics
  - Example applications
  - Definitions
- Training
  - Cost function
  - Optimization
- Optimization algorithms
  - Batch gradient descent
  - Stochastic gradient descent
  - Mini-batch gradient descent
- Types of machine learning
  - Supervised vs. unsupervised
  - Regression vs. Classification
- Overfit vs Underfit
  - Diagnose
  - Remedy

Overfit: example

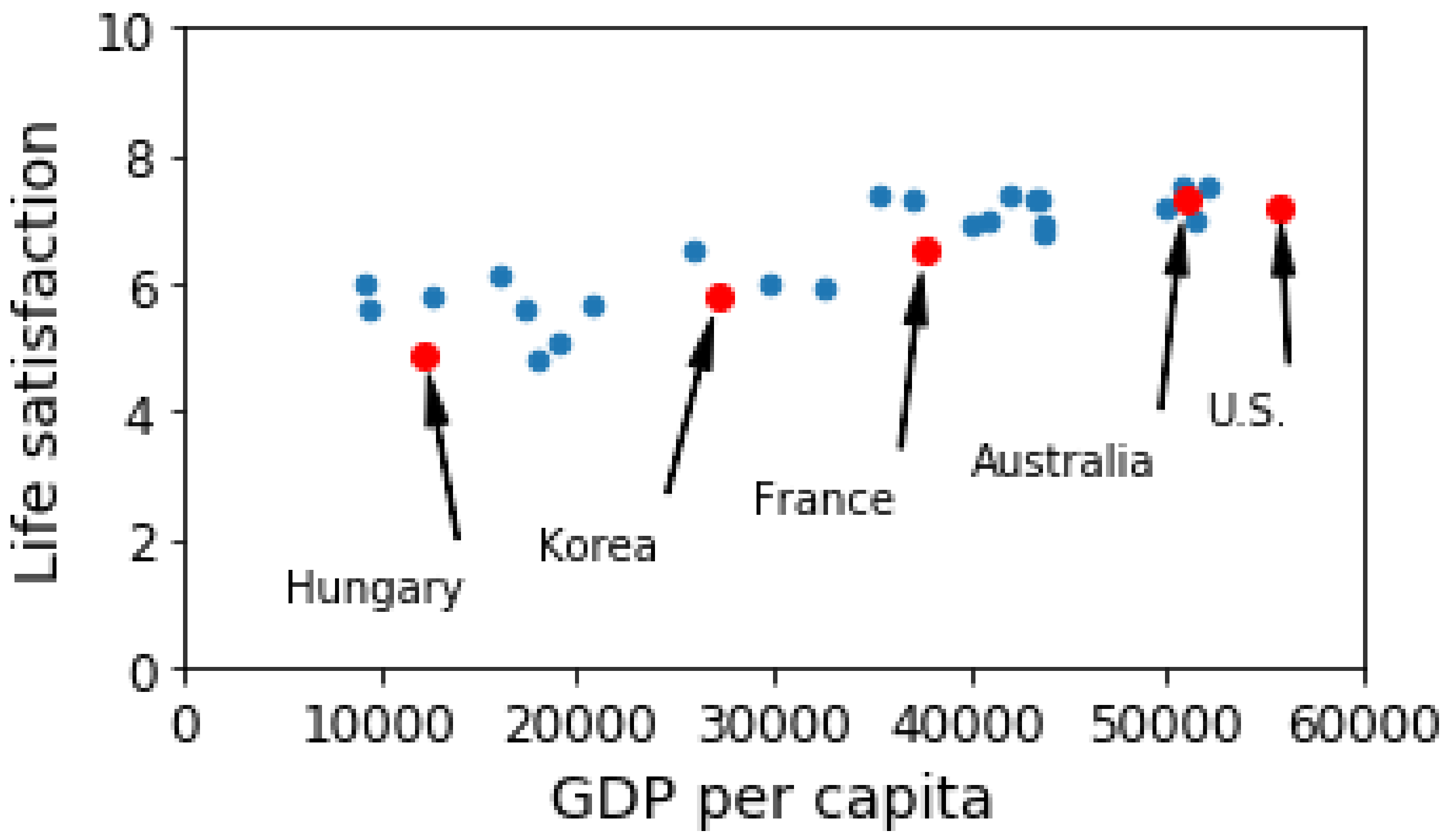


Figure from Aurelien Geron's ML book, page 19

Good fit

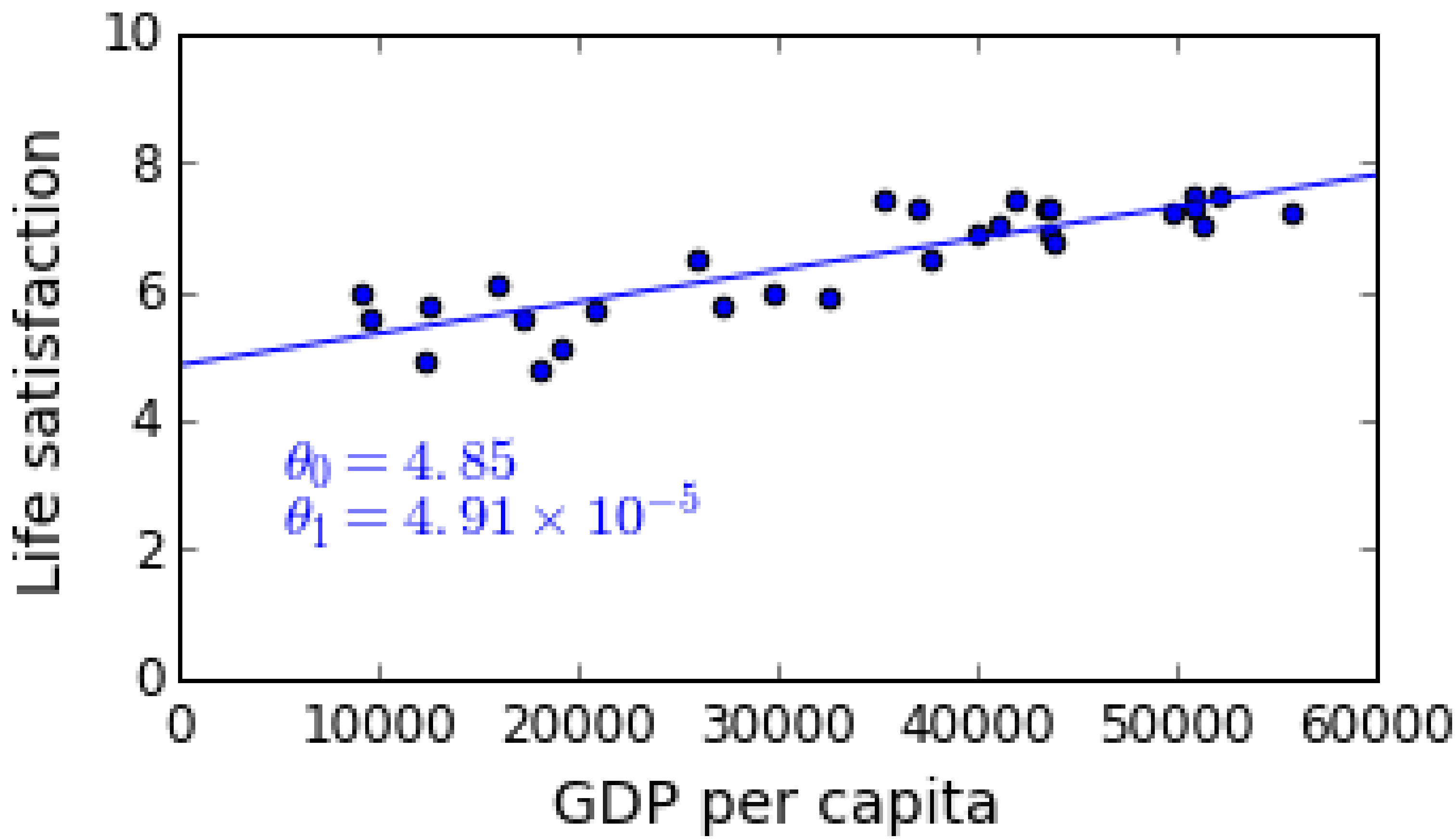


Figure from Aurelien Geron's ML book, page 20

Overfit (polynomial degree = 60)

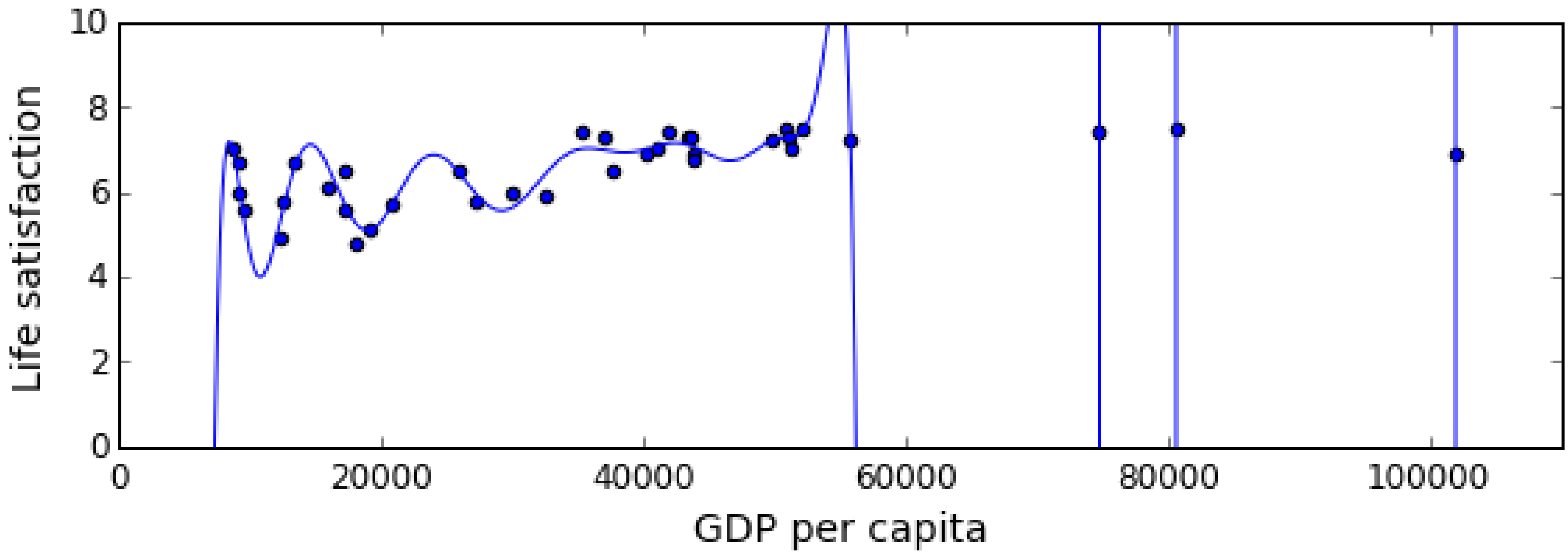


Figure from Aurelien Geron's ML book, page 26

Overfit

- Fit the training data very well (actually too well)
- But, does not generalize well to new data.
- That is, predictions on new data will be bad!
- Remember, the whole purpose of machine learning is to make predications.
- If a machine learning model only works well on training data, but not on new (i.e., unseen) data, it is NOT a good model/product.

How to tell if you are overfitting?

- Split your training data into three parts:

1. Training set
2. Validation set
3. Test set

How to tell if you are overfitting?

- Split your training data into three parts:

1. Training set
2. Validation set
3. Test set

- Use only training set for training (put the other two sets of data aside), calculate the prediction error J_{train}
- After training, apply the learned model to cross-validation set, calculate the prediction error J_{cv}
- If J_{train} is very small, J_{cv} is large, you overfit your data!

Remedy for overfitting

- Overfitting happens when your ML model is overly complex
- Therefore, possible solutions are:
  1. Collect more training data
  2. Reduce data noise
  3. Simplify model
     - using linear model rather than a high-degree polynomial model
     - using regularization
     - …

Underfit

- The opposite of overfitting
- Your model is too simple to capture the meaningful information/structures/relations in the data.

Underfit: example

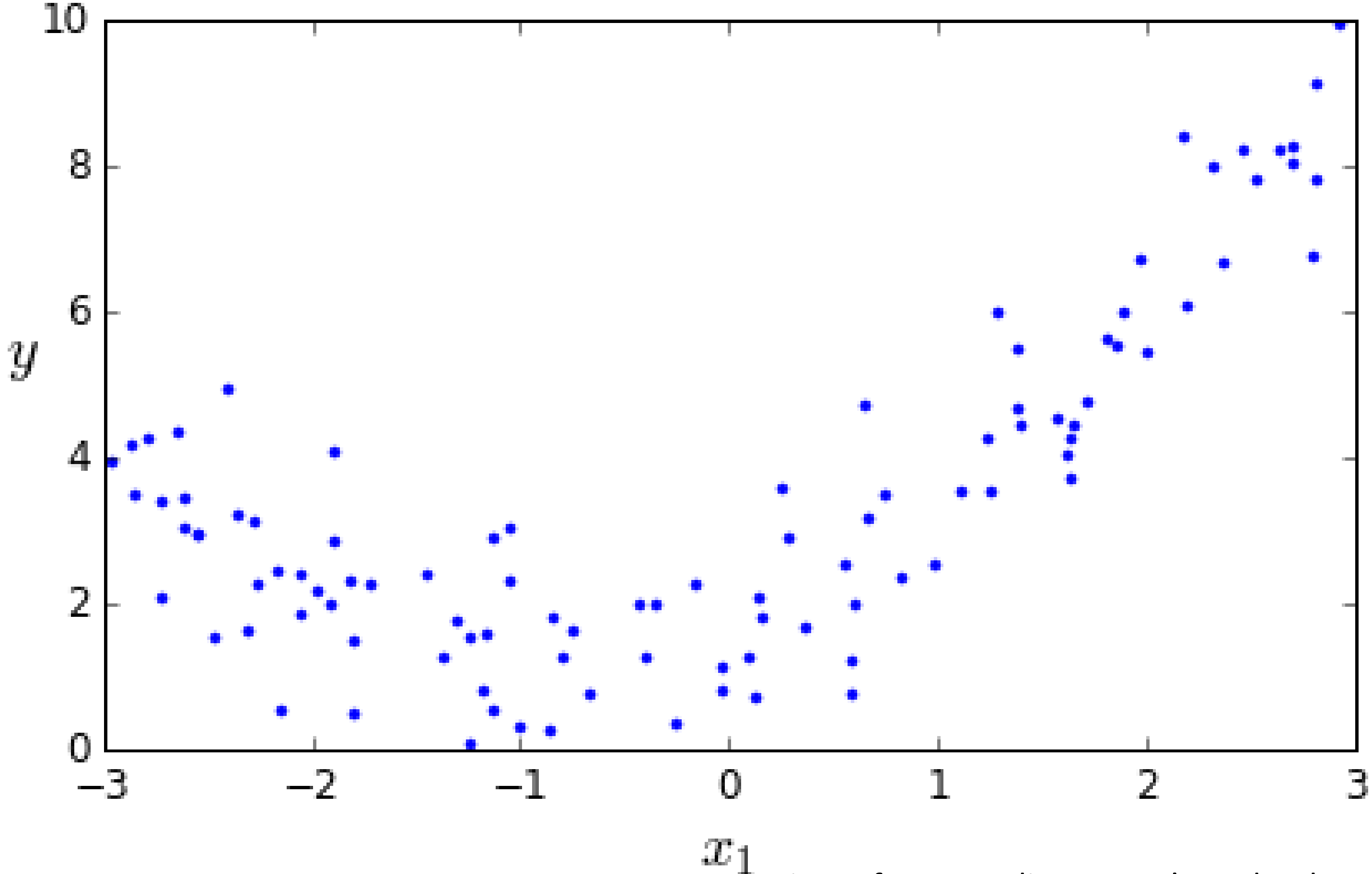
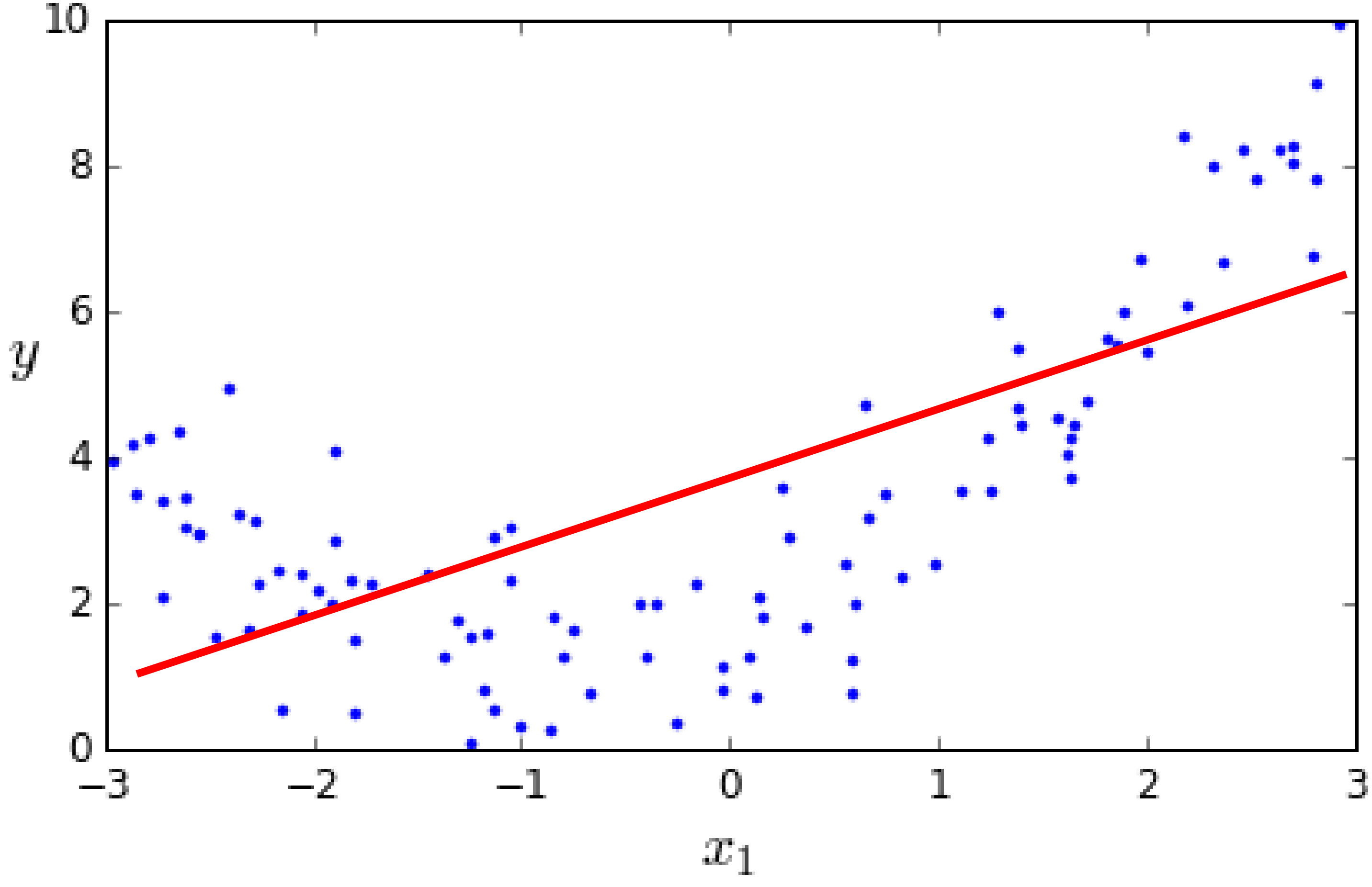


Figure from Aurelien Geron's ML book, page 121

Underfit: example



Overfit vs. underfit

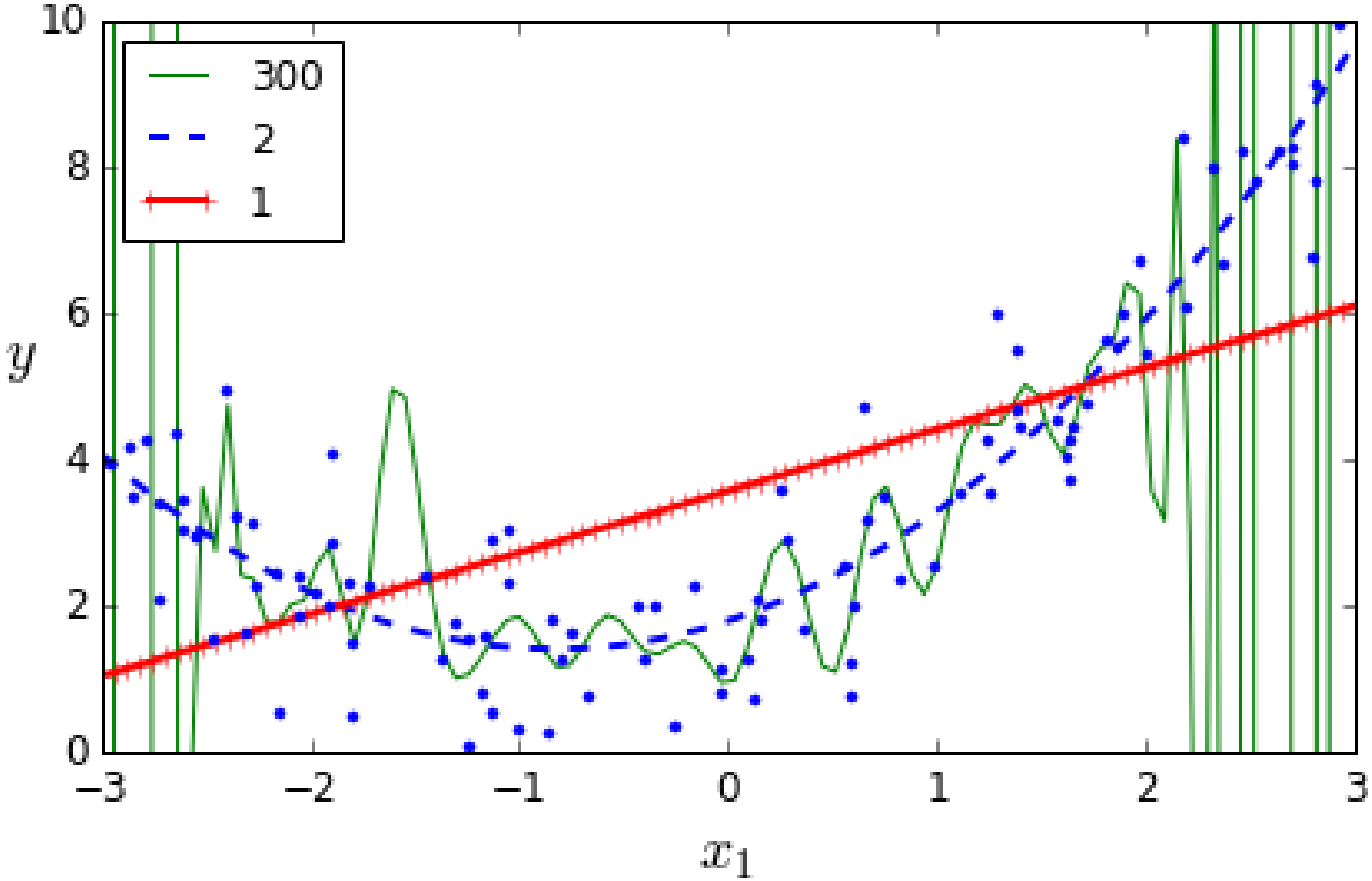


Figure from Aurelien Geron's ML book, page 123

How to tell if you are underfitting?

- Split your training data into three parts:

1. Training set
2. Validation set
3. Test set

How to tell if you are overfitting?

- Split your training data into three parts:

1. Training set
2. Validation set
3. Test set

- Use only training set for training (put the other two sets of data aside), calculate the prediction error E^{train}
- After training, apply the learned model to cross-validation set, calculate the prediction error E^{cv}
- If E^{train} is large, E^{cv} is large, you underfit your data!

Remedy for underfitting

- Underfitting happens when your ML model is overly simple
- Therefore, possible solutions are:
  1. ~~Collect more training data~~
  2. ~~Reduce data noise~~
  3. Make your model more complex
     - using a high-degree polynomial model rather than a linear model
     - using less regularization
     - Adding more features such as $(x_1^2, x_2^2, x_1 x_2)$ to the learning algorithm (feature engineering)

Remember,

- If you are underfitting your data, collecting more data won't help!

Bias vs. Variances

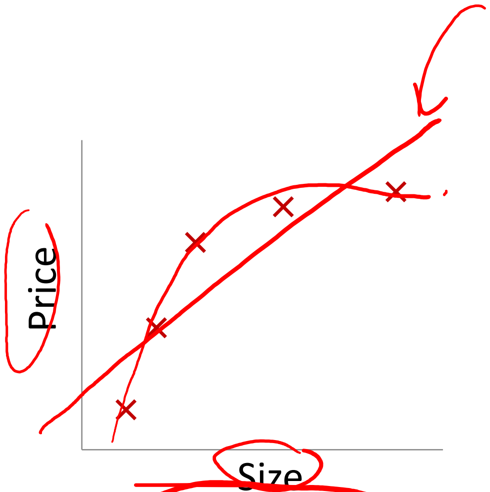
Bias

- Due to over-simplified assumptions
- Your model is heavily limited or biased by your assumptions.
- E.g., assuming a linear model when the training data are actually from a non-linear model
- Lead to underfitting the training data

Variance

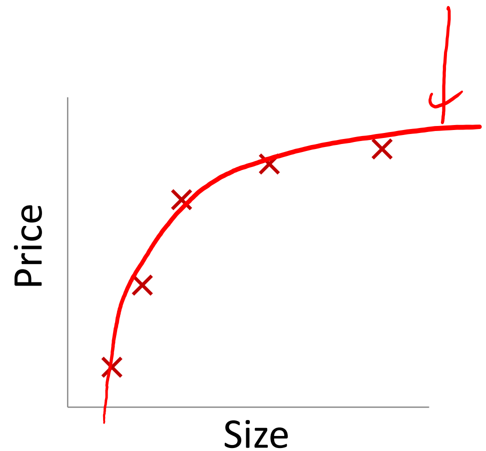
- Your model has too much flexibility (or, is allowed to have too many small-scale variations)
- E.g., assume a highly nonlinear model when the data are actually linear
- Lead to overfitting the data

Bias/variance



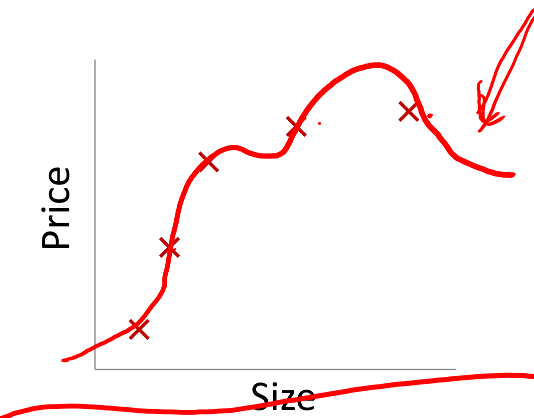
$\theta_0 + \theta_1 x$

High bias
(underfit)



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

"Just right"



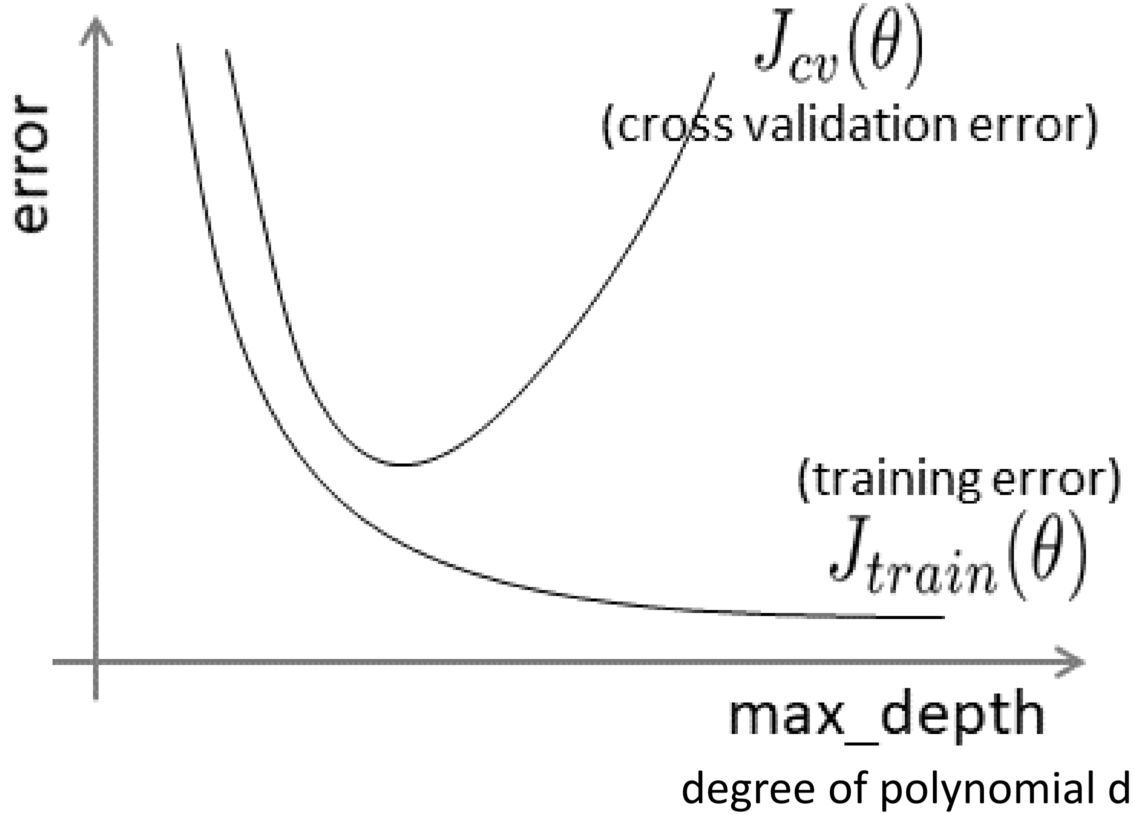
$\theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4$

High variance
(overfit)

This slide is taken from Andrew Ng's ML class on coursera

80

Diagnosing bias vs. variance



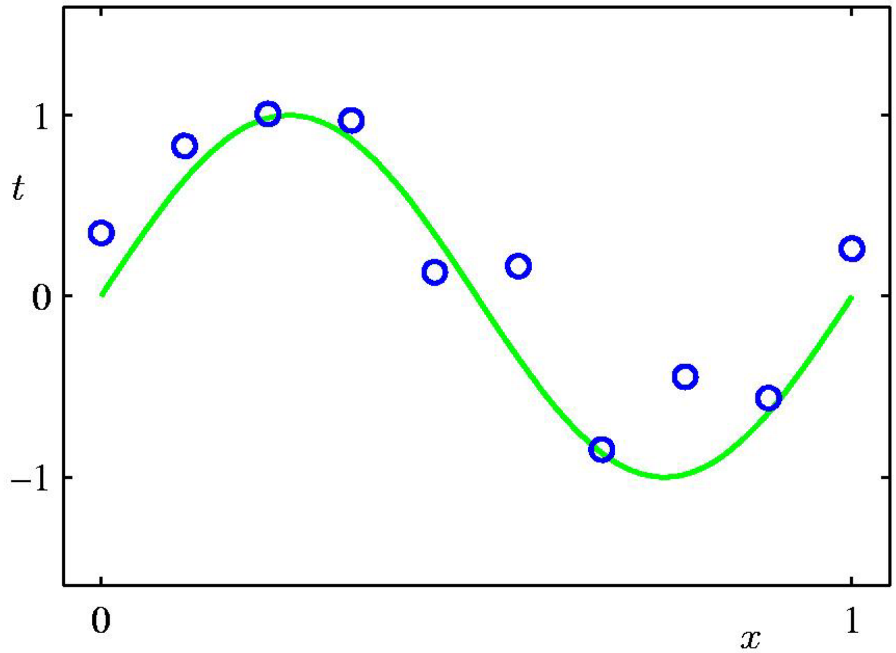
Sample Question

- Suppose you trained a ML model on training data, and obtained a prediction accuracy of 89% (i.e., prediction error of 11%)
- Suppose you also applied the learned model to your cross-validation data set, and obtained a prediction accuracy of 96% (i.e., a prediction error of 4%)
- Question: Is this an underfitting or overfitting problem?

Remedy for overfitting

- Overfitting happens when your ML model is overly complex
- Therefore, possible solutions are:
  1. Collect more training data
  2. Reduce data noise
  3. Simplify model
     - using linear model rather than a high-degree polynomial model
     - using regularization
     - …

Training data set of M = 10 points, each comprising an observation of input variable x and the corresponding target variable t.



The green curve shows the function $\sin(2\pi x)$ used to generate the data.

Goal: predict the value of t for some new value of x, based on the model learned from training data.

Polynomial curve fitting

Fit the data using a polynomial function of the form:

$$h_\theta(x) = \theta_0 + \theta_1 x + \theta_2 x^2 + \cdots + \theta_N x^N$$

where N is the degree of the polynomial

Polynomial curve fitting

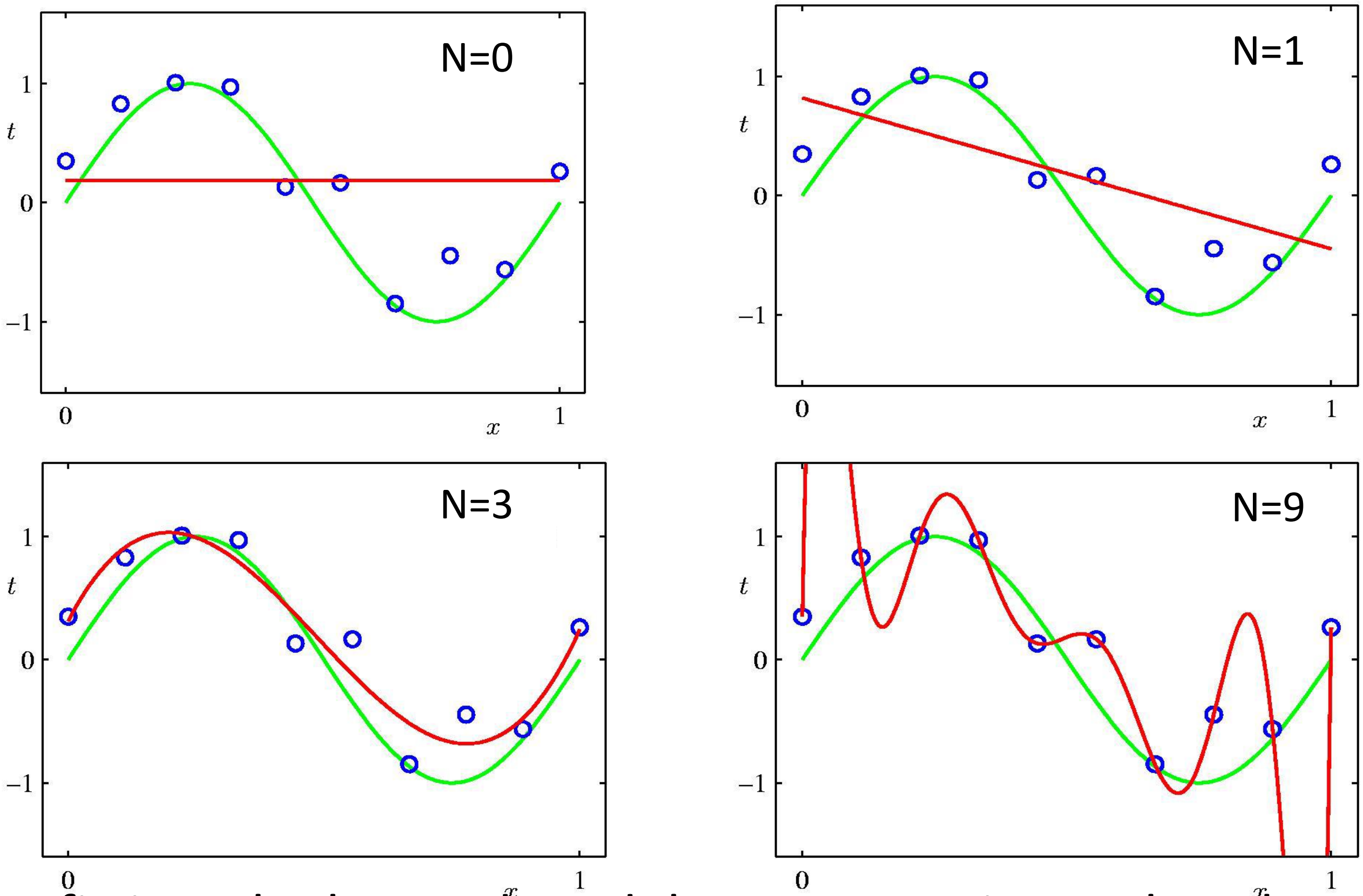
Fit the data using a polynomial function of the form:

$$h_\theta(x) = \theta_0 + \theta_1 x + \theta_2 x^2 + \cdots + \theta_N x^N$$

where N is the degree of the polynomial

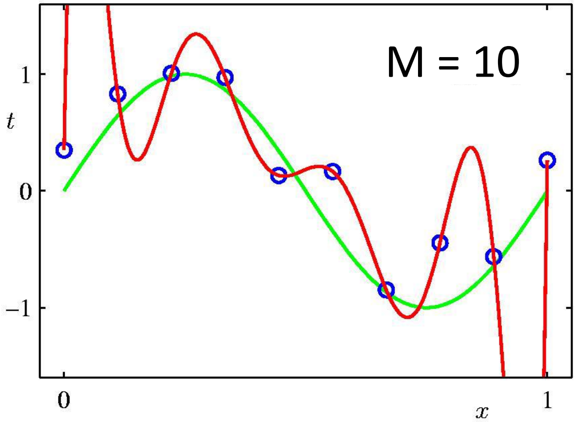
Question: how to select N?

Training models with N = 0, 1, 3, 9



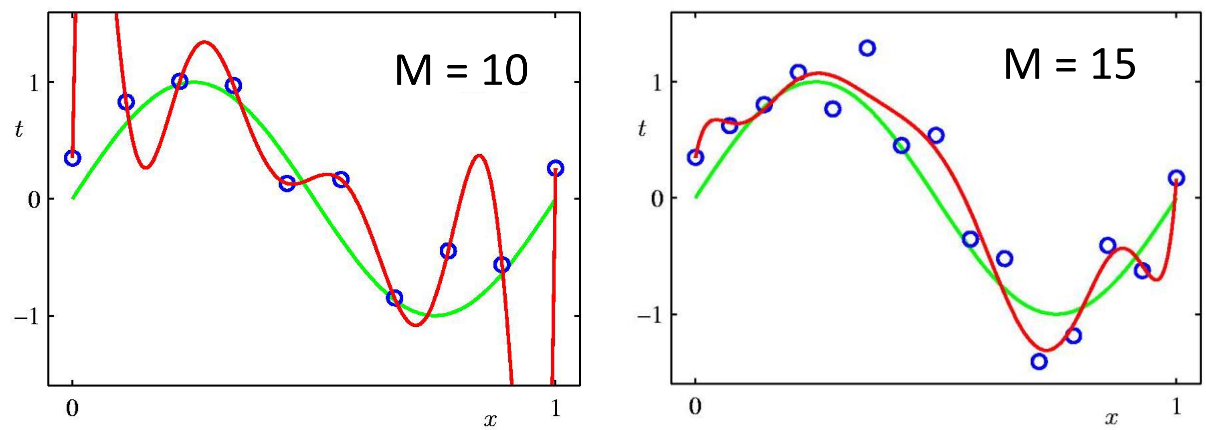
Overfitting: the learned model captures noise, rather than the true and meaningful features/trends among the training data.

More training data



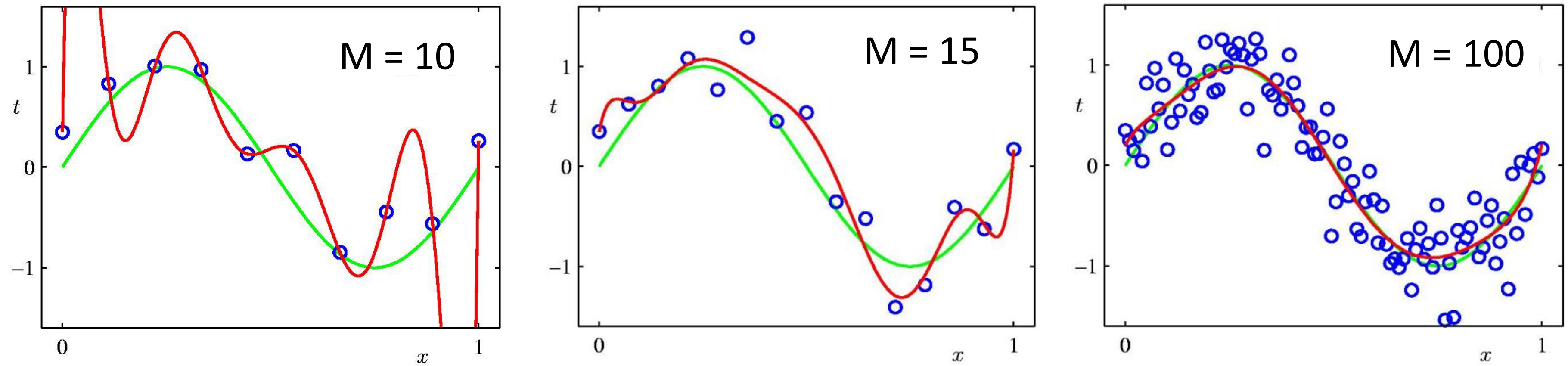
Learned models with N = 9 for M = 10, 15, and 100.

More training data



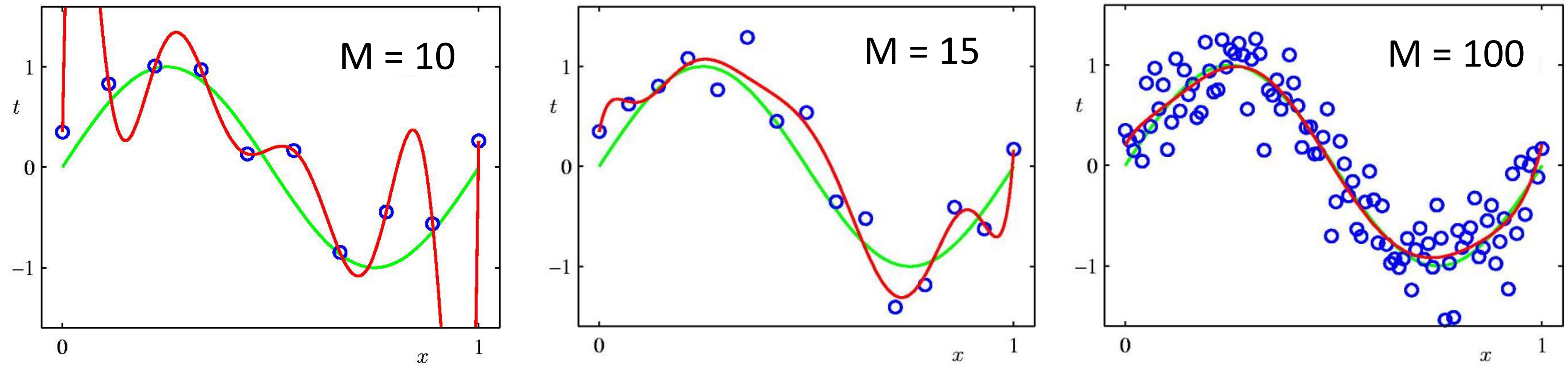
Learned models with N = 9 for M = 10, 15, and 100.

More training data



Learned models with N = 9 for M = 10, 15, and 100.

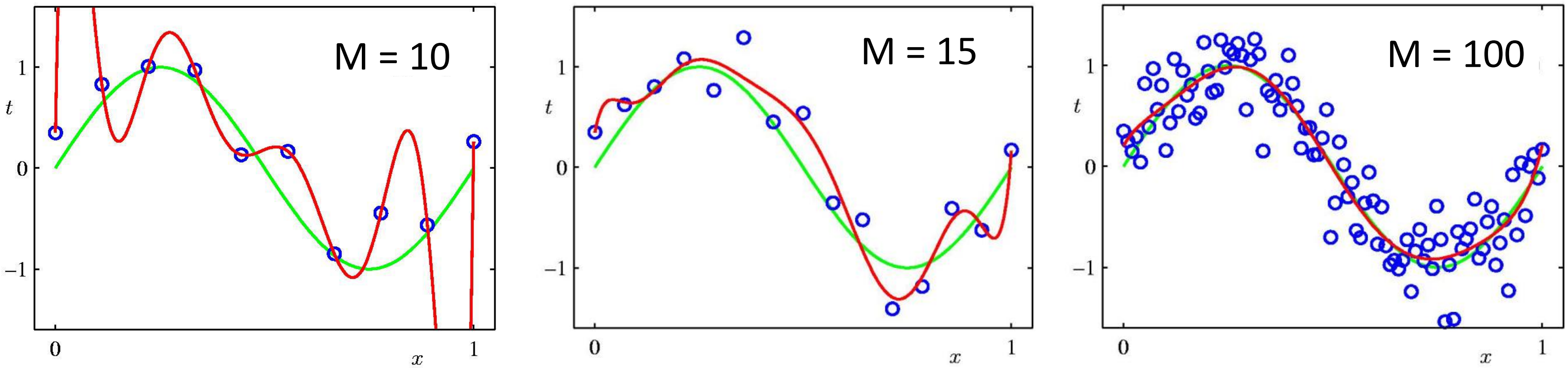
More training data



Learned models with N = 9 for M = 10, 15, and 100.

Increasing the size of training data set helps decrease the overfitting.

More training data



Learned models with N = 9 for M = 10, 15, and 100.

Increasing the size of training data set helps decrease the overfitting.

However, obtaining more training data is not always doable in reality.

Remedy for overfitting

- Overfitting happens when your ML model is overly complex
- Therefore, possible solutions are:
  1. Collect more training data
  2. Reduce data noise Not always feasible either!
  3. Simplify model
     - using linear model rather than a high-degree polynomial model
     - using regularization
     - …

	N = 0	N = 1	N = 3	N = 9
θ_0	0.19	0.82	0.31	0.35
θ_1		-1.27	7.99	232.37
θ_2			-25.43	-5321.83
θ_3			17.37	48568.31
θ_4				-231639.30
θ_5				640042.26
θ_6				-1061800.52
θ_7				1042400.18
θ_8				-557682.99
θ_9				125201.43

Table of the coefficients $\boldsymbol{\theta}$ learned from training data.

Note how the magnitudes of coefficients increase dramatically as the order of polynomial increases.

Regularization

- Discourage the learned model parameters (i.e., coefficients) from being too large.
- Keep the model simple.
- Keep the model from being unnecessarily complicated.

Regularization

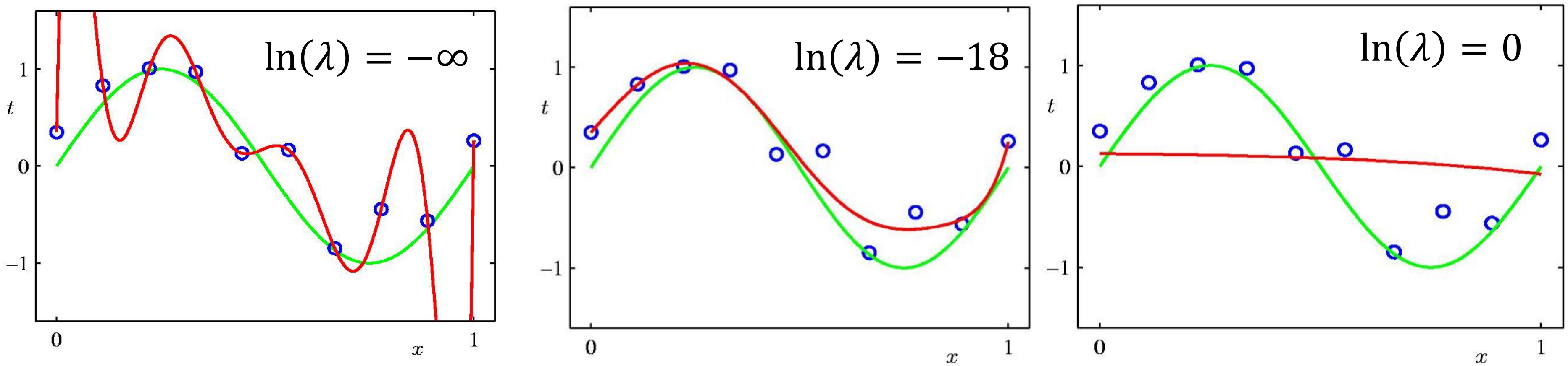
$$J(\boldsymbol{\theta}) = \frac{1}{2}\sum_{i=1}^{M}\left(h_\theta\left(x^{(i)}\right) - t^{(i)}\right)^2$$

Regularization

$$J(\boldsymbol{\theta}) = \frac{1}{2}\sum_{i=1}^{M}\left(h_\theta\left(x^{(i)}\right) - t^{(i)}\right)^2 + \frac{1}{2}\lambda\sum_{j=1}^{N}\theta_j^2$$

Also known as shrinkage because it shrinks/reduces the values of the model parameters

Regularized curve fitting

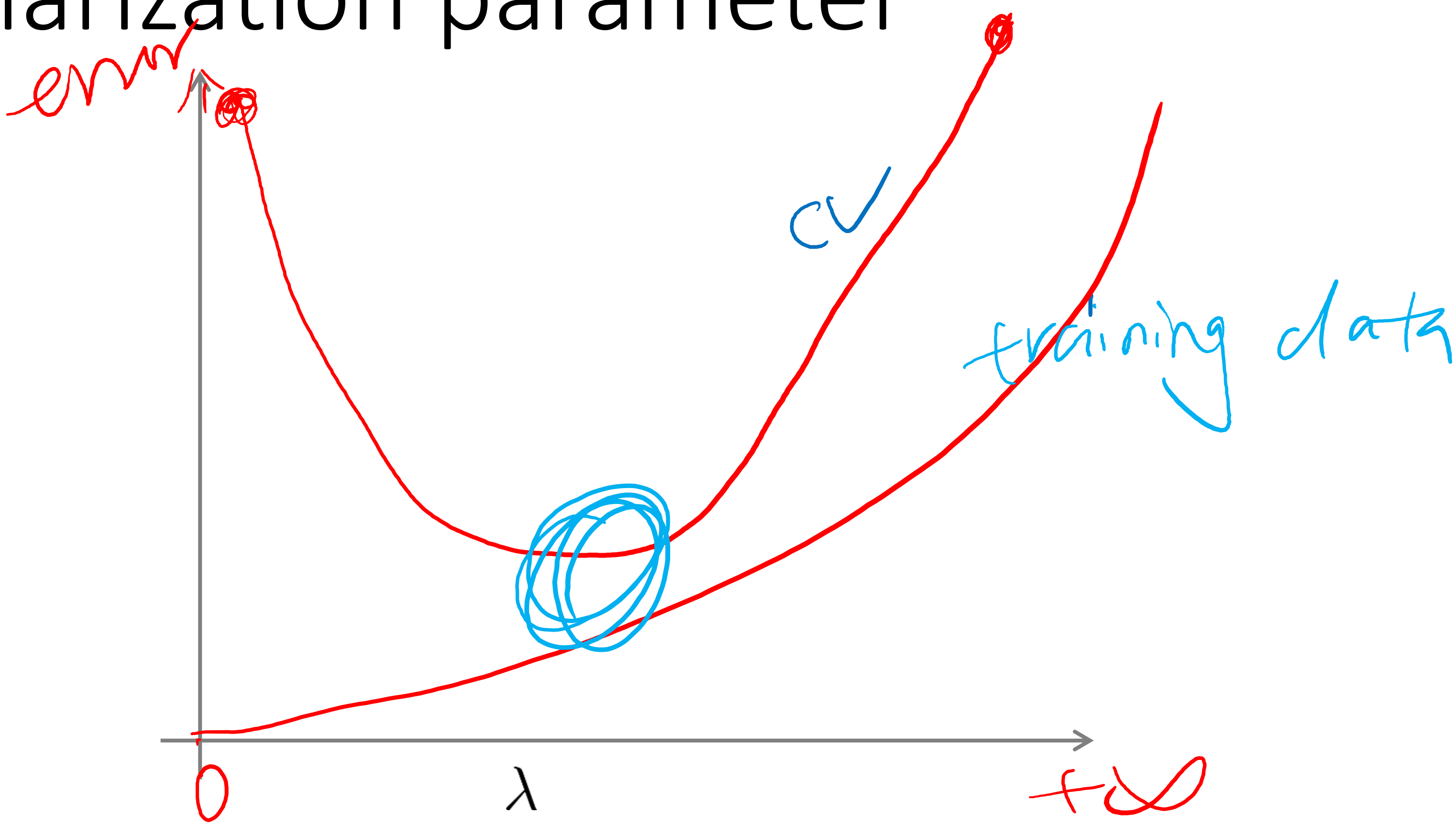


N = 9 M = 10 with three different regularization parameters

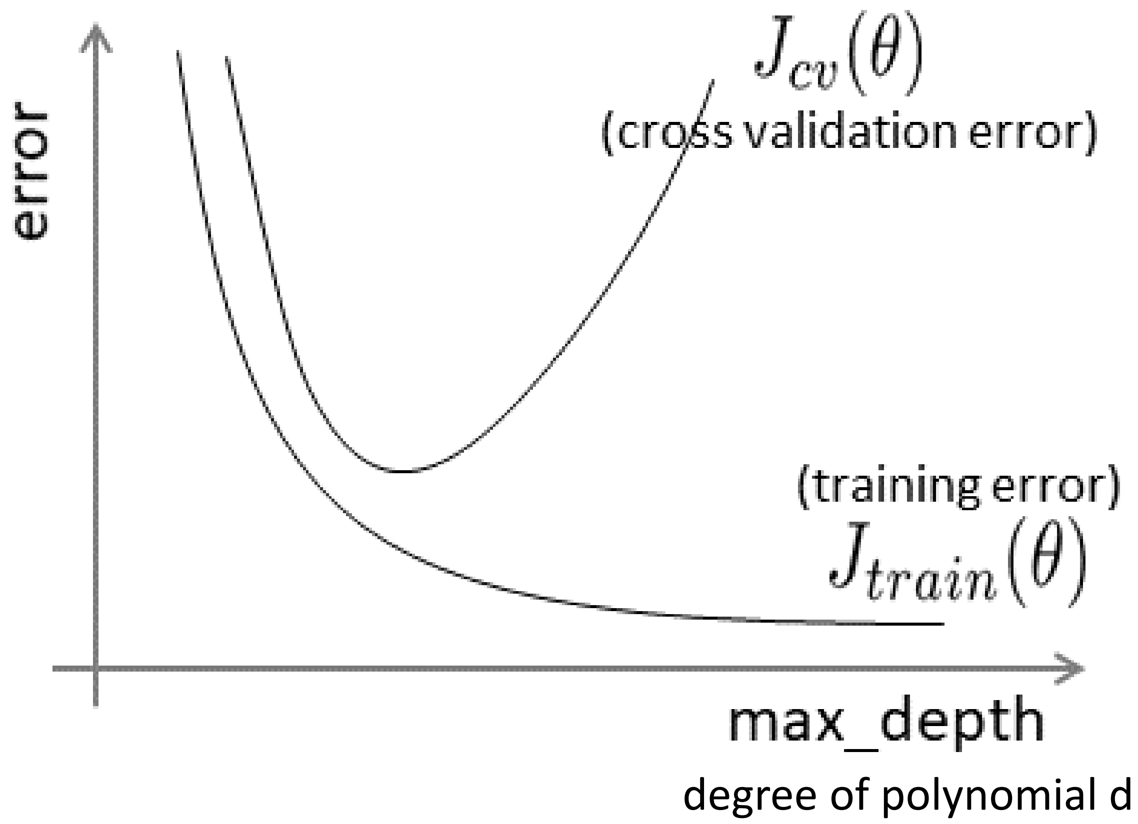
How to select regularization parameter?

- Split the whole data set into training and validation data sets
- Carry out machine learning with different values for regularization parameters
- Calculate the errors for both training and validation data sets for each regularization parameter

Bias/variance as a function of regularization parameter

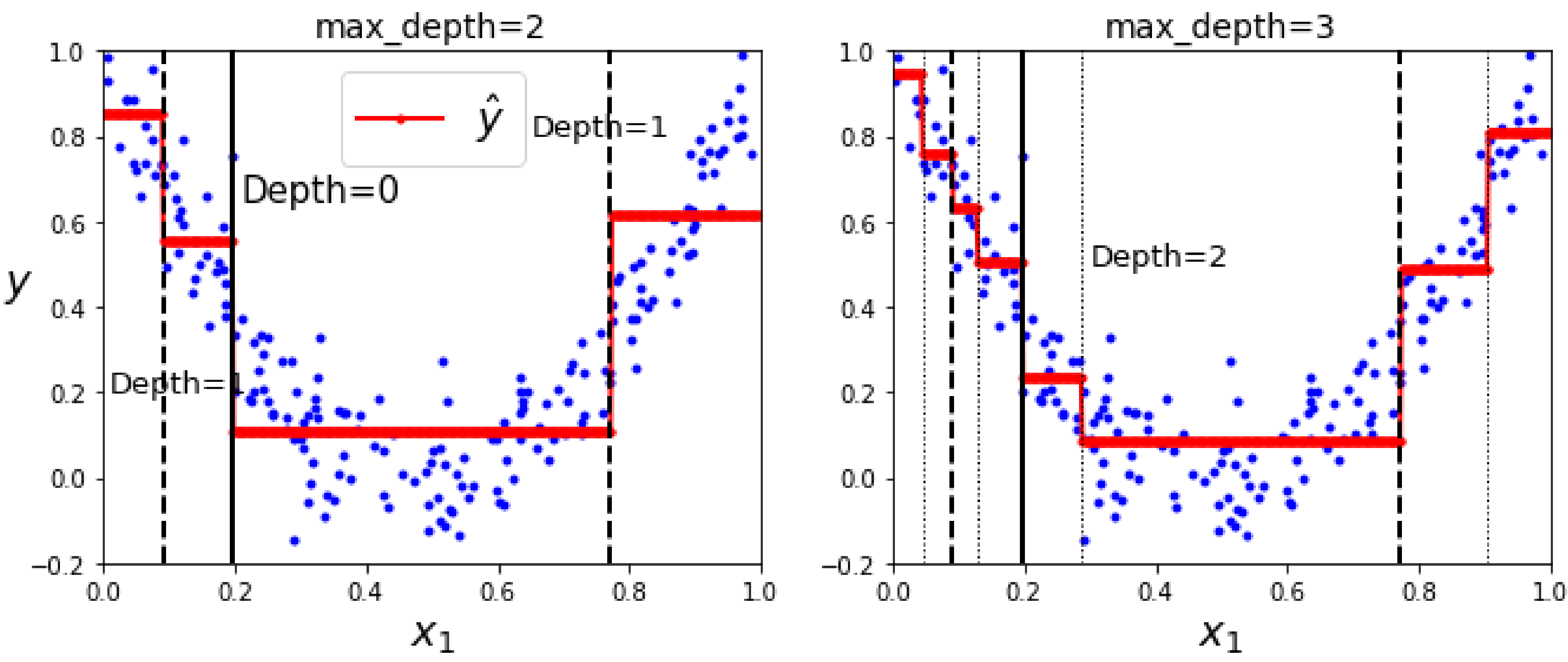


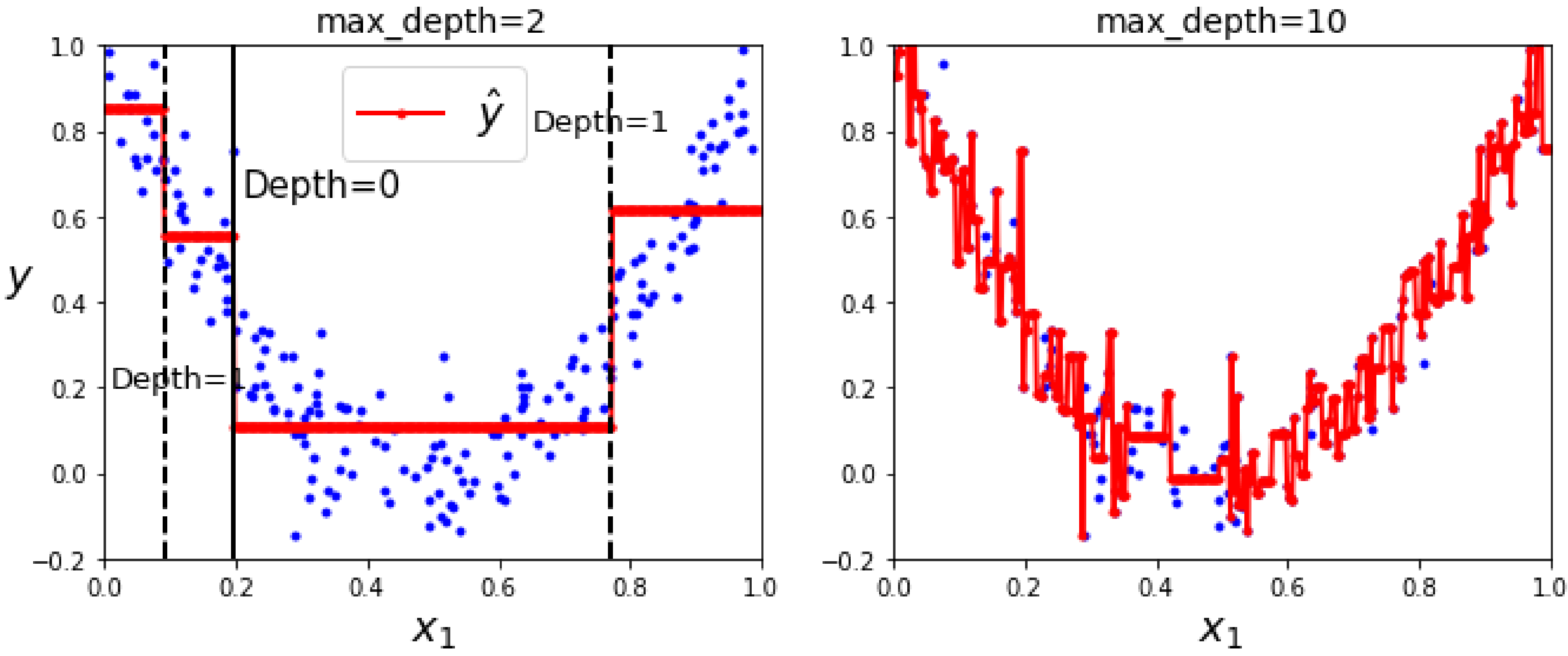
Diagnosing bias vs. variance



Regularization parameters

- Decision trees: max_depth


max_depth=2
max_depth=3
$\hat{y}$
Depth=0
Depth=1
Depth=1
Depth=2
y
x_1
x_1
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
0.0
0.2
0.4
0.6
0.8
1.0


max_depth=2
max_depth=10
$\hat{y}$
Depth=0
Depth=1
Depth=1
y
x_1
x_1
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
0.0
0.2
0.4
0.6
0.8
1.0