Introduction of Machine Learning & Deep Learning for Geoinformatics

Attawut Nardkulpat

Faculty of Geoinformatics

Burapha University

14/09/2020

attawut@buu.ac.th

Outline

- **Week 1**
  - Overview AI, Machine Learning & Deep Learning
  - Regression
- **Week 2**
  - Logistic Classification & Image Classification
- **Week 3**
  - Ensemble Learning
- **Week 4**
  - Deep Learning

AI & Machine Learning

Artificial Intelligence

- “ปัญญาประดิษฐ์”
- เป็นแนวคิดในการพัฒนาด้านคอมพิวเตอร์ให้สามารถในการคิด ตัดสินใจและเรียนรู้ได้ด้วยตนเองที่ใกล้เคียงกับมนุษย์ ส่งผลให้เครื่องคอมพิวเตอร์มีความฉลาดมากขึ้น สามารถทำงานในระบบที่มีความซับซ้อนได้อย่างมีประสิทธิภาพ โดยไม่ต้องอาศัยแรงงานจากมนุษย์
- ใช้ครั้งแรก ปี 1956 ในงานสัมมนาที่เกี่ยวข้องกับ Automata, Neural Network
- สามารถแบ่งระดับตามฉลาดและความสามารถของปัญญาประดิษฐ์ ได้ออกเป็น 3 ระดับ คือ

Artificial Narrow Intelligence (ANI)

- ในระดับแรกคือ Artificial Narrow Intelligence (ANI) หรือบางครั้งเรียกว่า “Weak AI” ซึ่งในระดับนี้ AI จะมีความสามารถและความเชี่ยวชาญในเฉพาะด้าน ตัวอย่างที่เห็นได้ชัดก็คือ AlphaGo ที่สามารถเอาชนะเซียนโกะมือหนึ่งของโลก

Artificial General Intelligence (AGI)

- “Strong AI” หรือ “Human-Level AI” ในระดับนี้ AI จะมีความสามารถและความฉลาดเทียบเท่ามนุษย์ ทั้งความสามารถการคิดเชิงเหตุผล การวางแผนและแก้ปัญหา การคิดในเชิงซับซ้อน และสามารถเรียนรู้ได้จากประสบการณ์
- การพัฒนา AI ให้มีความสามารถในระดับนี้ยากกว่าการพัฒนาในระดับ ANI เป็นอย่างมากและแน่นอนว่า AI ในตอนนี้ยังห่างไกลจากความสามารถในการเชื่อมโยงความรู้หลายสาขาหรือทำงานได้เกือบทุกอย่างเฉกเช่นเดียวกับมนุษย์ ไม่ได้จำกัดอยู่ในเรื่องใดเรื่องหนึ่งเพียงอย่างเดียว

Artificial Superintelligence (ASI)

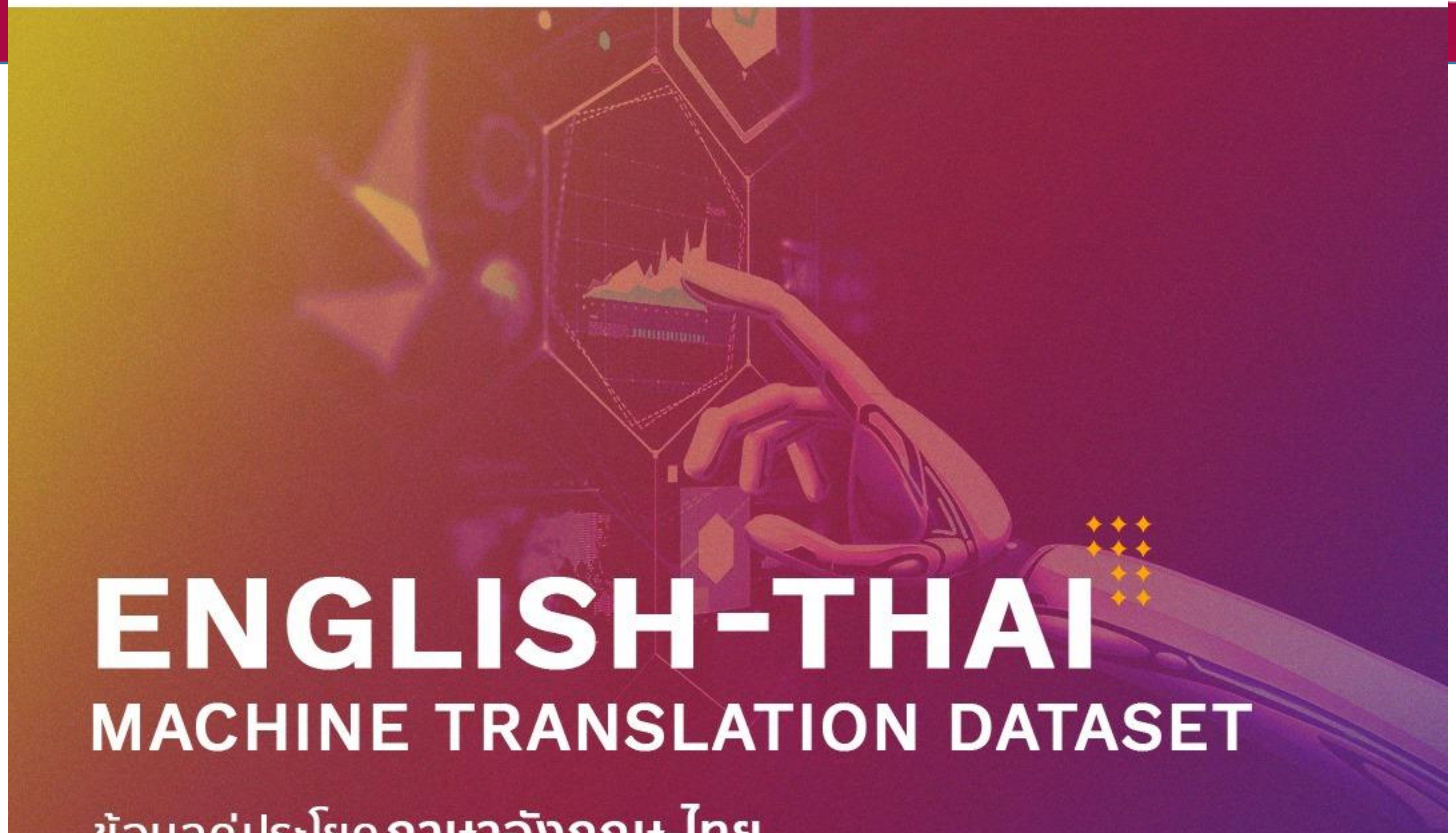
- Nick Bostrom นักปรัชญาและนักคิดชั้นนำด้าน AI ได้ให้นิยามของ “Superintelligence” ไว้ว่า “เครื่องจักรที่มีสติปัญหาและความสามารถเหนือกว่าสมองมนุษย์ที่ฉลาดที่สุดในเกือบทุกสาขา รวมถึงความคิดเชิงสร้างสรรค์ทางวิทยาศาสตร์ ความรู้เชิงภูมิปัญญา และทักษะทางสังคม” เขาเรียกมันว่า “เครื่องจักรทรงภูมิปัญญา (Machine Superintelligence)”

Natural Language Processing

- “การประมวลผลภาษาธรรมชาติ”
- เกี่ยวข้องกับภาษาที่มนุษย์ใช้ ไม่ว่าจะเป็น ข้อความ ตัวอักษร เสียงพูด
- ตัวอย่าง
  - Speech Recognition : Siri (Apple) , Google Assistant (Google), Alexa (Amazon)
  - Machine Translation : Google Translation
  - Text Classification : Gramma checker, Sentiment Analysis, Named Entity Recognition
  - Chat Bot

VISTEC-depa THAILAND AI RESEARCH INSTITUTE
VISTEC VIDYASIRIMEDHI INSTITUTE OF SCIENCE AND TECHNOLOGY
depa
SCB ไทยพาณิชย์

าสตร์
University



ENGLISH-THAI
MACHINE TRANSLATION DATASET

ข้อมูลคู่ประโยคภาษาอังกฤษ-ไทย
จำนวนกว่า *1 ล้านคู่ประโยค*

ดาวน์โหลดฟรี!
23 มิถุนายน 63

TOM SIMONITE BUSINESS 07.22.2020 07:00 AM

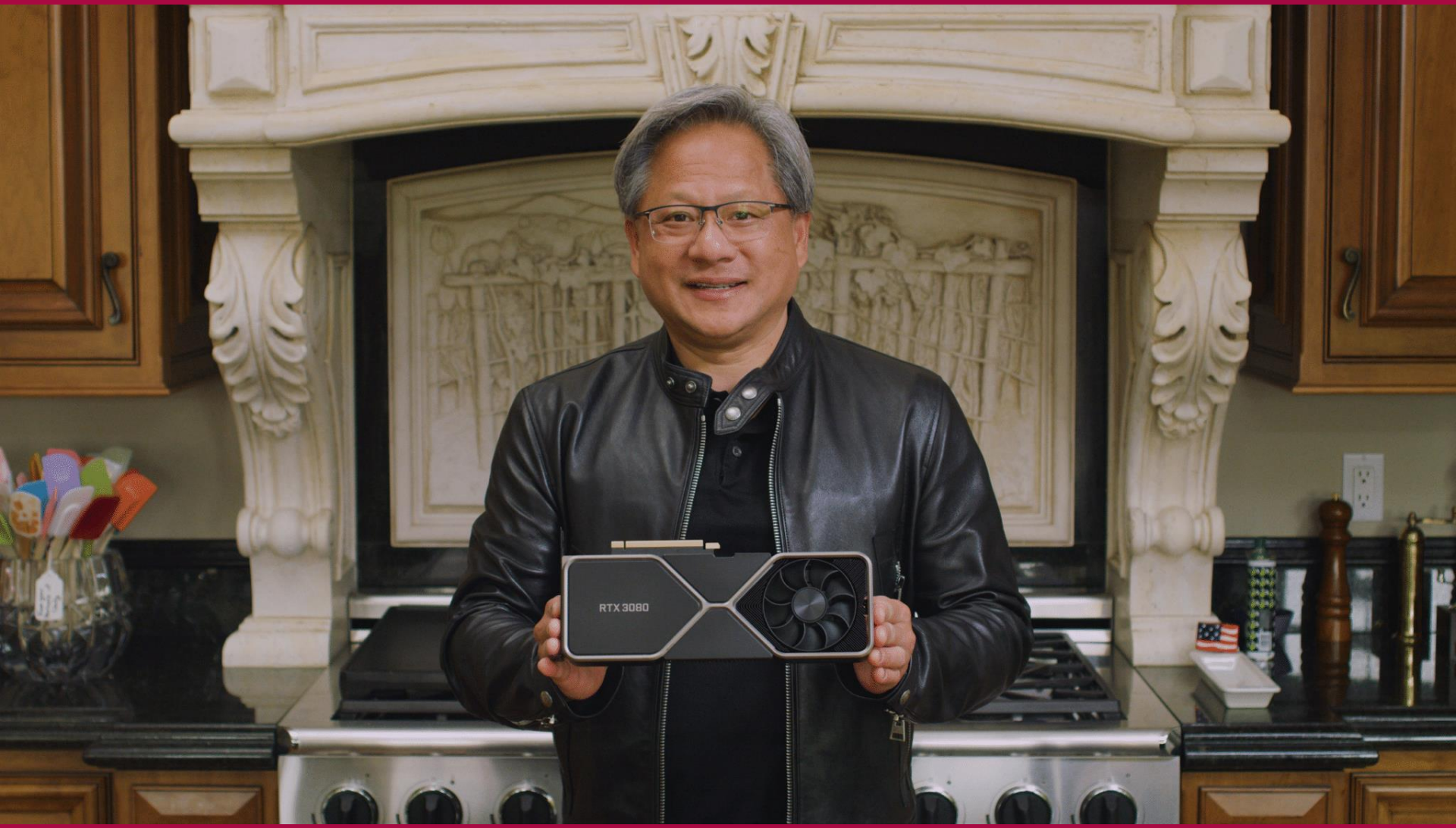
Did a Person Write This Headline, or a Machine?

GPT-3, a new text-generating program from OpenAI, shows how far the field has come—and how far it has to go.

- Human brain: 100 trillion* synapses
  * Could be 1,000+ trillion. ANN ≠ BNN.
- GPT-3: 175 billion parameters
  - Compute: 3.14E+23 flops [1]
  - Cost: $4.6 million [2]

- GPT-3: 175 billion parameters
  - Cost (2020): $4.6 million

- GPT-4 (Human Brain): 100 trillion parameters
  - Cost (2020): $2.6 billion
  - Cost (2024): $325 million
  - Cost (2028): $40 million
  - Cost (2032): $5 million


RTX 3080

		GEFORCE RTX 3090	GEFORCE RTX 3080	GEFORCE RTX 3070
GPU Engine Specs:	NVIDIA CUDA® Cores	10496	8704	5888
	Boost Clock (GHz)	1.70	1.71	1.73
Memory Specs:	Standard Memory Config	24 GB GDDR6X	10 GB GDDR6X	8 GB GDDR6
	Memory Interface Width	384-bit	320-bit	256-bit
Technology Support:	Ray Tracing Cores	2nd Generation	2nd Generation	2nd Generation
	Tensor Cores	3rd Generation	3rd Generation	3rd Generation
	NVIDIA Architecture	Ampere	Ampere	Ampere
	Microsoft DirectX® 12 Ultimate	Yes	Yes	Yes
	NVIDIA DLSS	Yes	Yes	Yes
	PCI Express Gen 4	Yes	Yes	Yes
	NVIDIA® GeForce Experience™	Yes	Yes	Yes
	NVIDIA Ansel	Yes	Yes	Yes

3090 ~ $1,499
~46,873.73 Bath

3080 ~ $699
~21857.73 Bath

3070 ~ $499
~15603.73 Bath

	RTX 2080 Ti	RTX 2080	RTX 2070
FP32 FPUs ("CUDA Cores")	4352	2944	2304
Streaming Multiprocessors	68	46	36
Core Clock / Boost Clock	1350/1545 FE: 1635MHz	1515/1710 FE: 1800MHz	1410/1620 FE: 1710MHz
Memory Interface	352-bit	256-bit	256-bit
Memory Capacity	11GB	8GB	8GB
GDDR6 Speed	14Gbps	14Gbps	14Gbps
Memory Bandwidth	616GB/s	448GB/s	448GB/s
SLI	NVLink 2-Way	NVLink 2-Way	TBD
TDP	~265~285W	~250-260W	175-185W
Price	$1,200 OR $1000*	$800 OR $700*	$600 OR $500*
Release Date	Sept. 20, 2018	Sept. 20, 2018	TBD

*Source for prices: NVIDIA's website. **NOTE:** We have also heard that the prices (perhaps for non-FE cards? Or is there just a miscommunication within NVIDIA?) could also be $500 for the 1070, $700 for the 2080, and $1000 for the 2080 Ti. We think this might be FE vs. Reference, but it could also be miscommunication by NVIDIA's teams. Not clear at the moment.

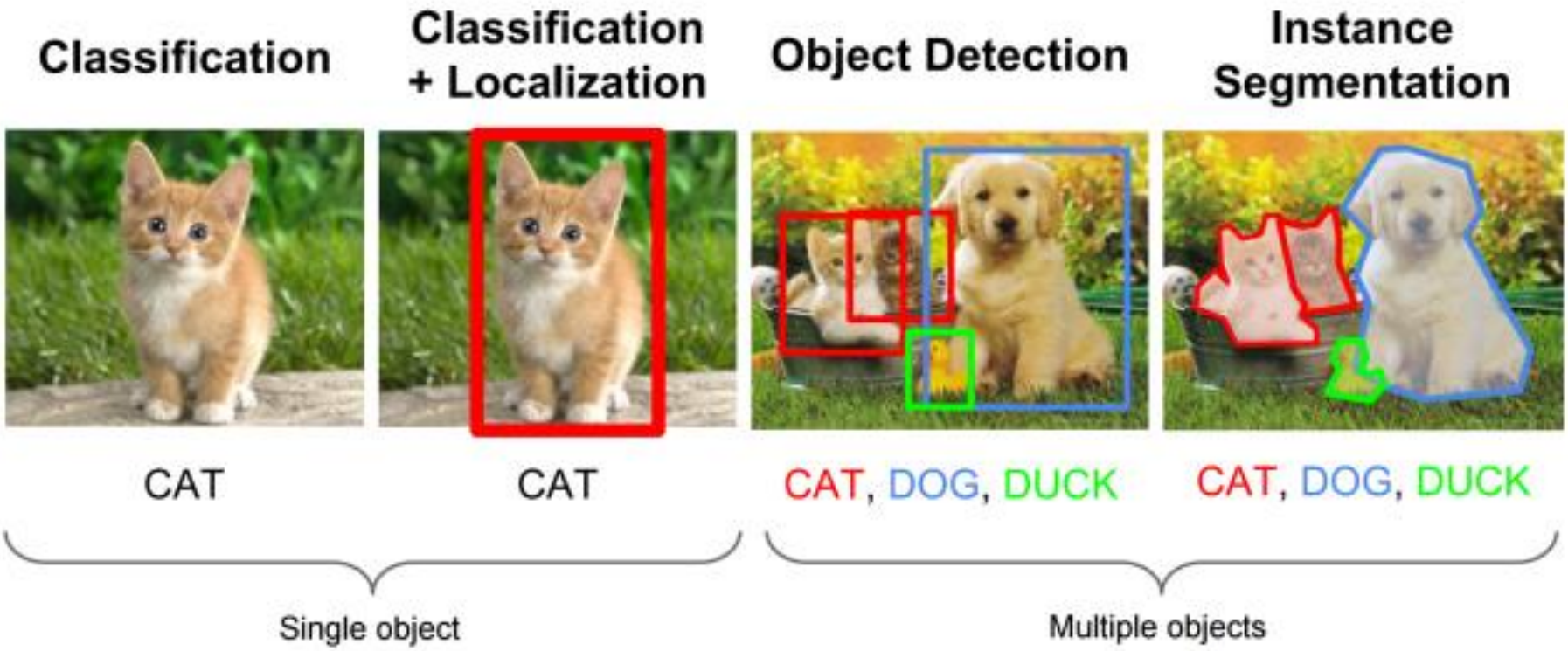
Gigabyte RTX 2080 Ti & 2080 Pricing

Scroll right for more ->

Model Name	Model#	MSRP
GeForce RTX 2080 Ti Gaming OC 11G	GV-N208TGAMING OC-11GC	$1,199.99
GeForce RTX 2080 Ti Windforce OC 11G	GV-N208TWF3OC-11GC	$1,169.99
GeForce RTX 2080 Gaming OC 8G	GV-N2080GAMING OC-8GC	$829.99
GeForce RTX 2080 Windforce OC 8G	GV-N2080WF3OC-8GC	$789.99

Computer Vision

- “คอมพิวเตอร์วิทัศน์”
- เกี่ยวข้องกับการมองเห็นของคอมพิวเตอร์ ทั้งภาพนิ่งและภาพเคลื่อนไหว
- ตัวอย่าง
  - Image Classification : LU/LC Classification
  - Object Detection : Face Recognition
  - Sematic Segmentation : Self-Driving Cars
  - Neural Style Transfer


Classification
Classification + Localization
Object Detection
Instance Segmentation
CAT
CAT
CAT, DOG, DUCK
CAT, DOG, DUCK
Single object
Multiple objects


Bread


Dairy products


Dessert


Egg


Fried food


Meat

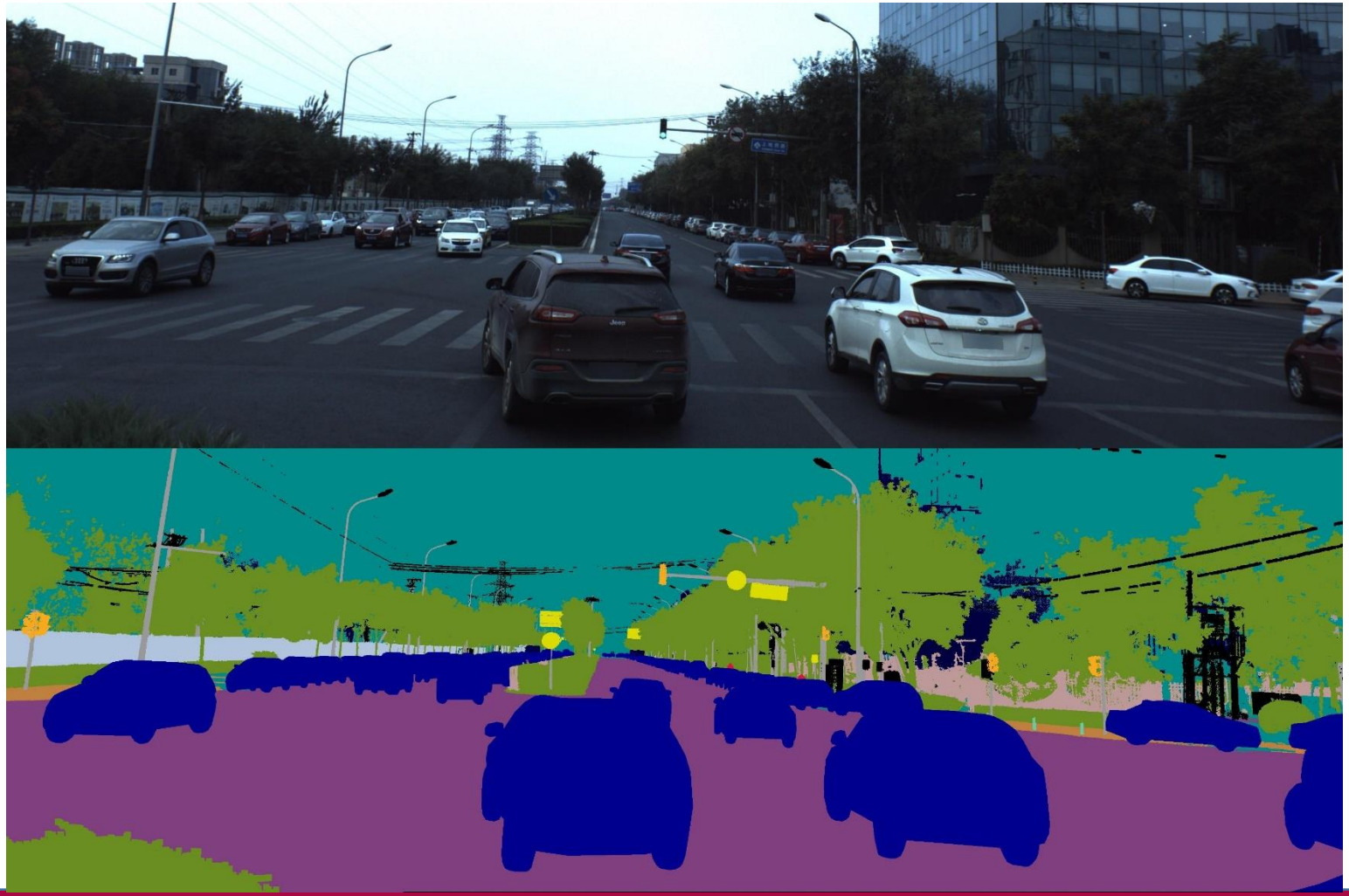

Noodles & Pasta

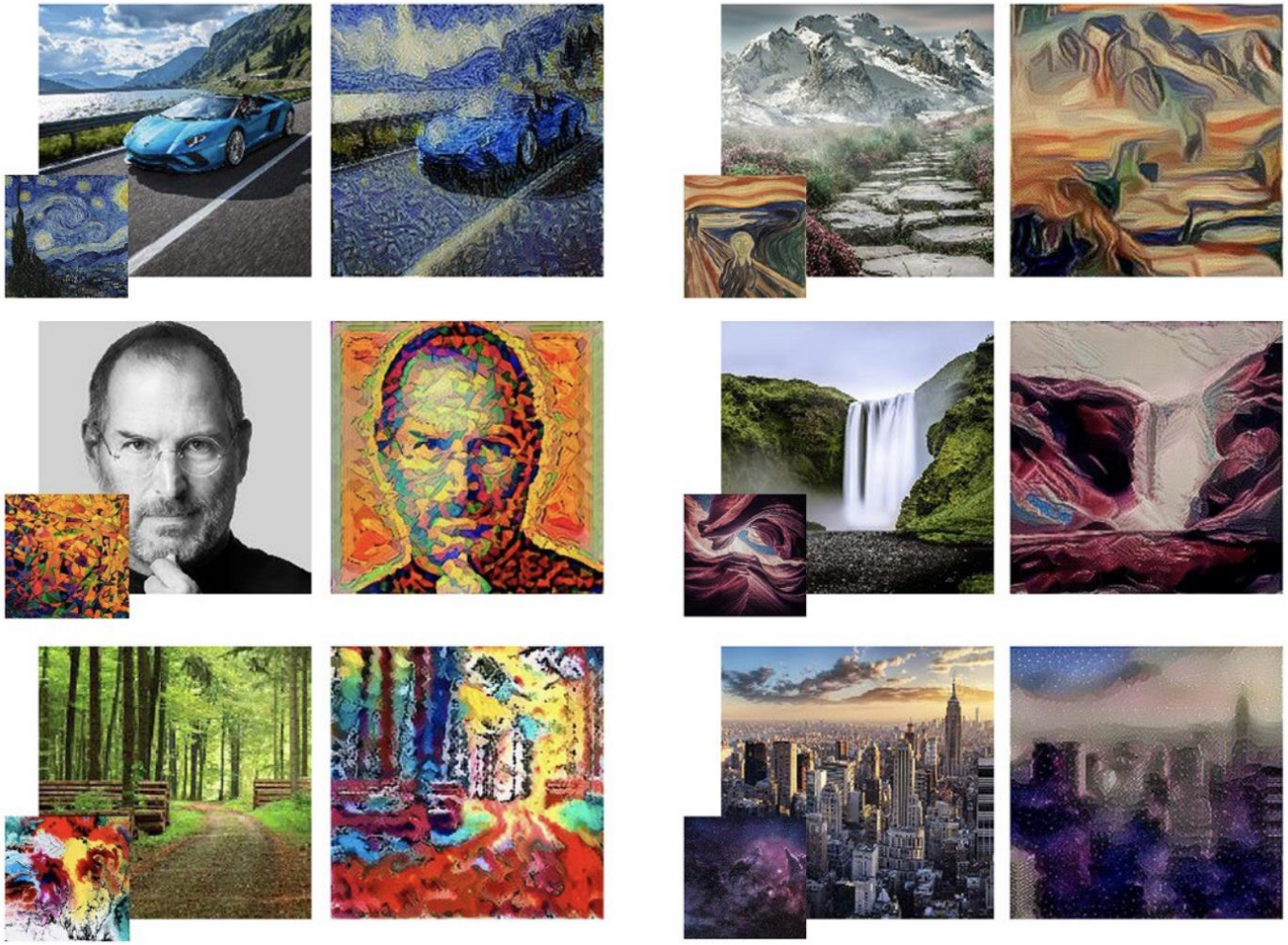

Rice


Seafood

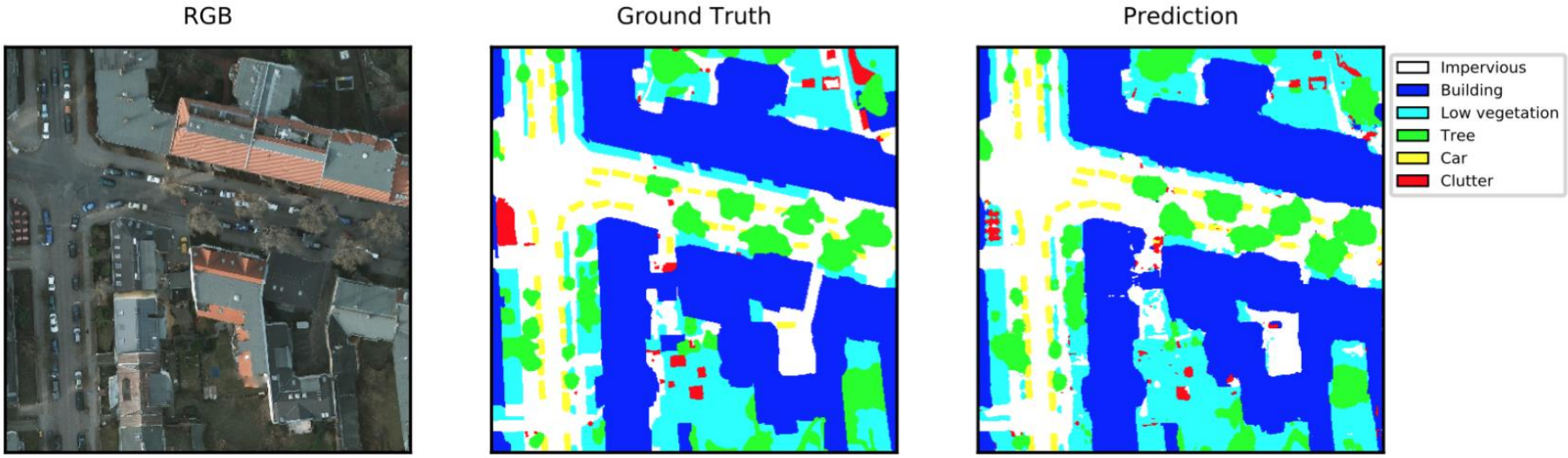

Soup


Vegetables & Fruits

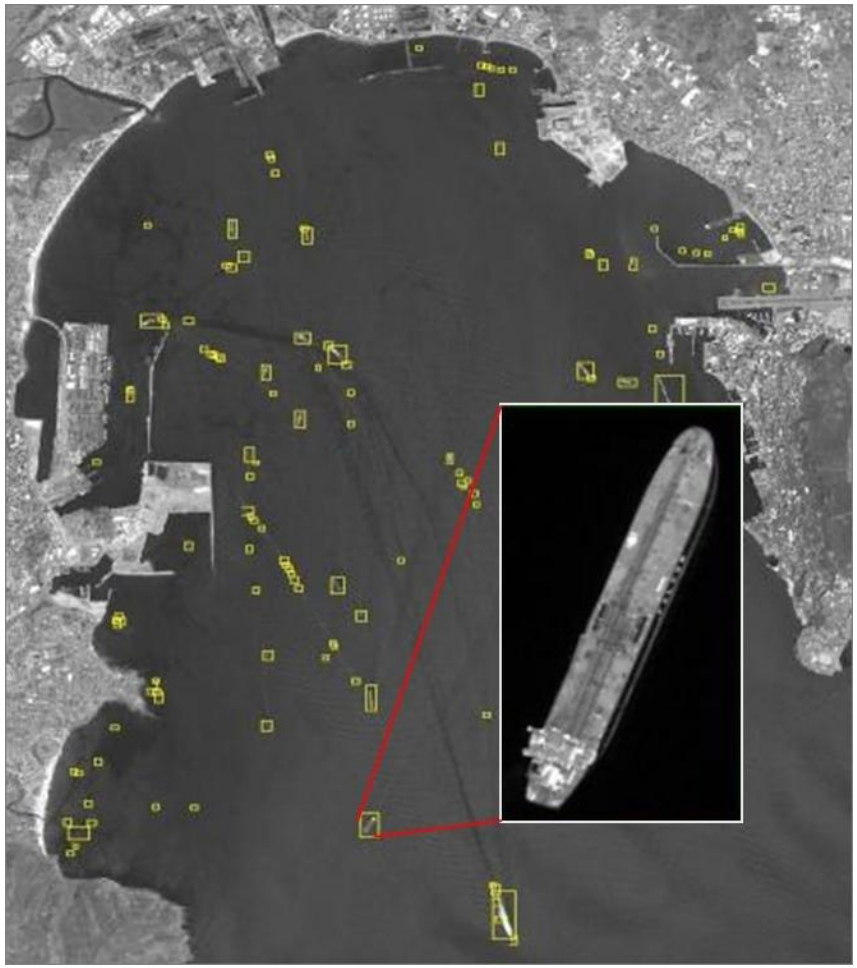




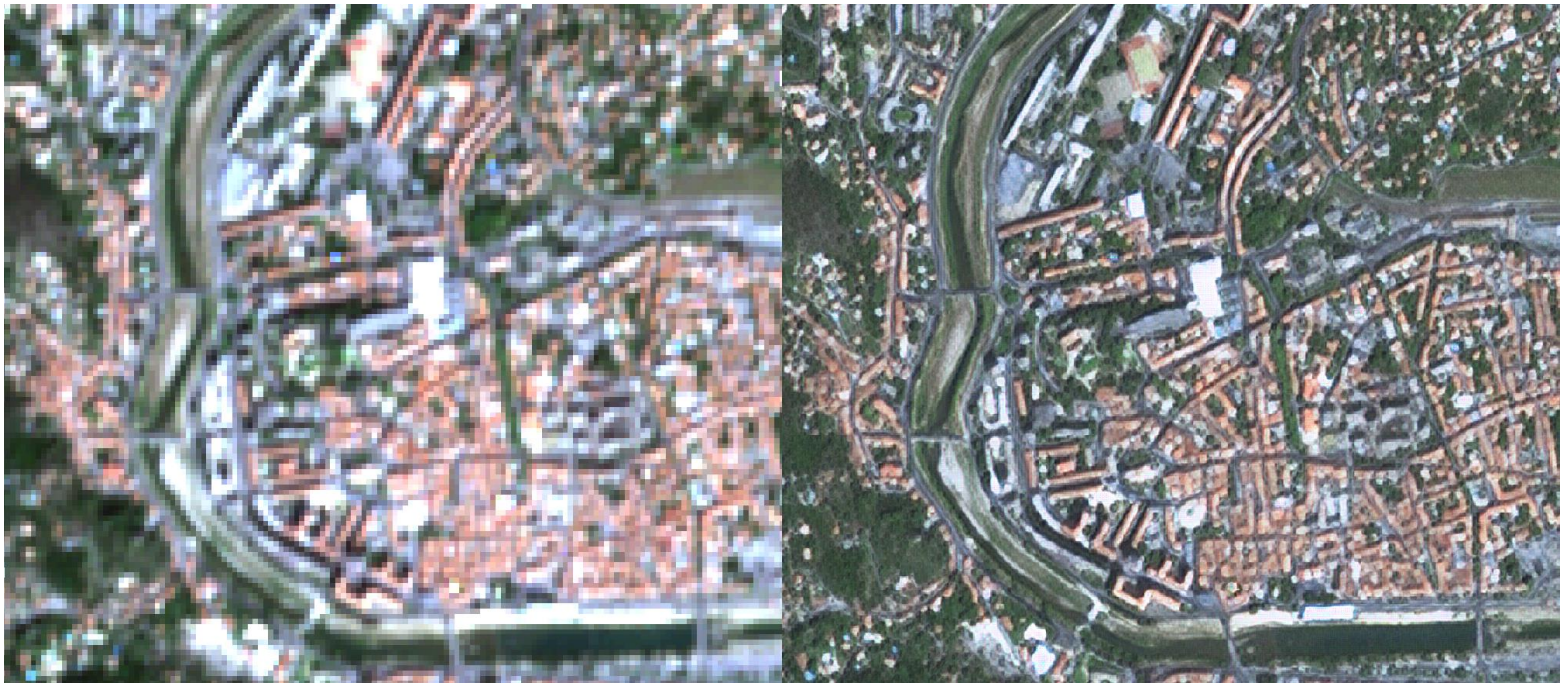
Geoinformatics



Credit : Lewis Fishgold and Rob Emanuele on May 30th, 2017

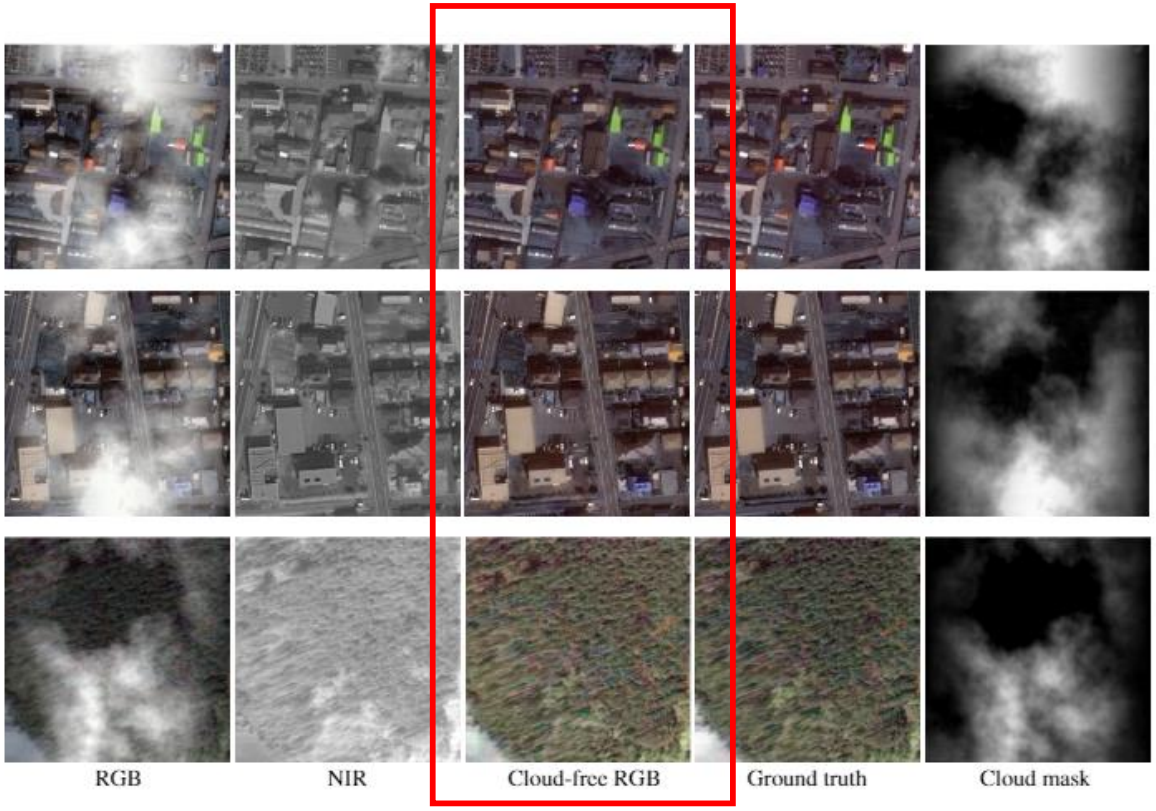


Super Resolution



From 10 m to 1.5 m

Cloud removal



Robotics

- “หุ่นยนต์”
- การสร้างเครื่องกลให้ทำงานในลักษณะเฉพาะอย่าง
- จุดหมายสำคัญ “Reinforce Learning”
- ตัวอย่าง
  - Boston Dynamic
  - SpaceX : Rocket parking

Platforms

Boston Dynamics

SpotMini

Spot

Atlas

Handle


AirSource

Expert System

- “ระบบผู้เชี่ยวชาญ”
- การจำลองผู้เชี่ยวชาญในงานนั้นๆเป็นชุดโปรแกรม ที่สามารถตัดสินใจแทนคนได้
- ตัวอย่าง
  - Recommendation System : Netflix, YouTube, Facebook, Lazada, Shopee
  - Trader System
  - Medical AI : Watson

Everything is a Recommendation

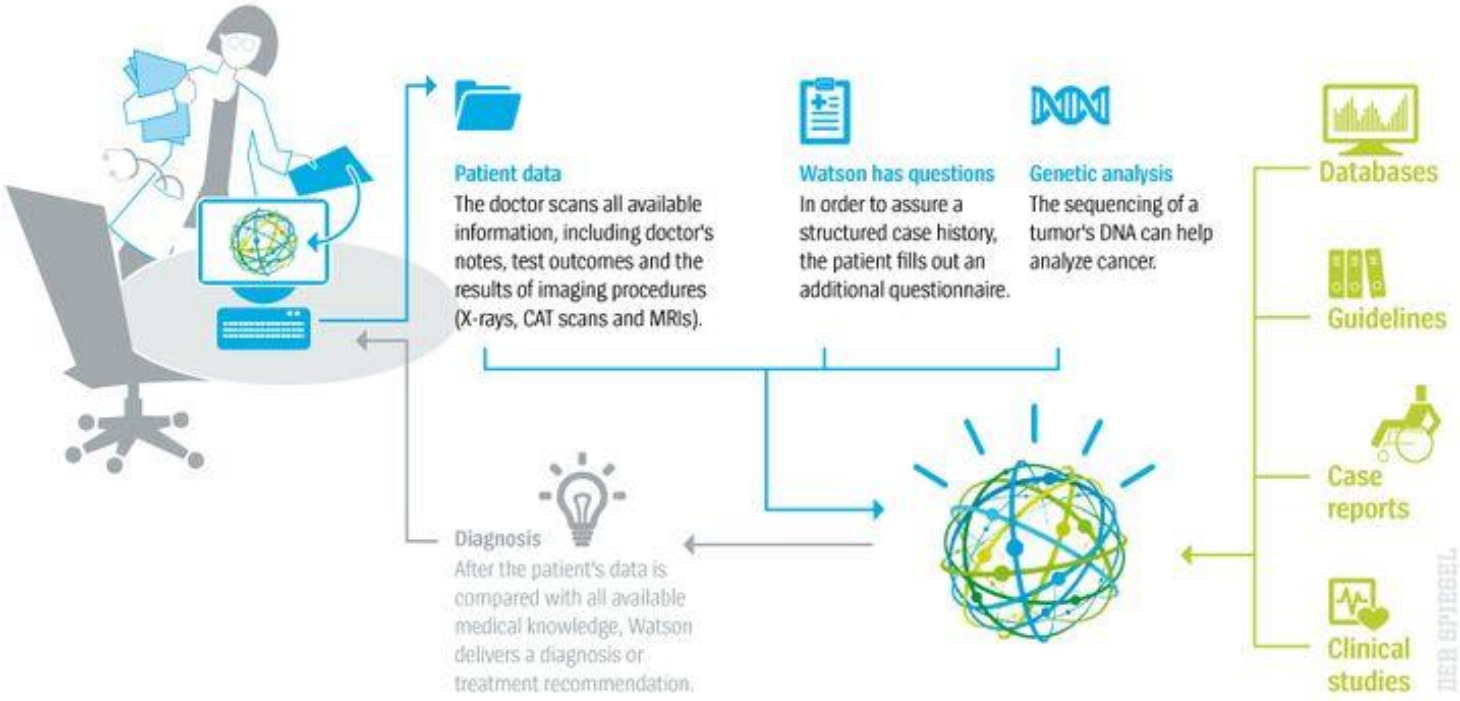
Over 80% of what people watch comes from our recommendations

Recommendations are driven by **Machine Learning**

NETFLIX

How Watson Works

The ways IBM's system is used in medicine



อย่างไรก็ตามเครื่องมือที่สาขาต่างๆเหล่านี้ ใช้แก้ปัญหาส่วนใหญ่ในปัจจบันคือ “การเรียนรู้ของเครื่อง (Machine Learning)”

การเรียนรู้ของเครื่อง (Machine Learning)

- Machine Learning เป็นศาสตร์หนึ่งที่มุ่งเน้นศึกษาการทำให้คอมพิวเตอร์มีความสามารถที่จะเรียนรู้ได้โดยไม่ต้องเขียนโปรแกรมกำกับไว้อย่างชัดเจน
  — Arthur Samuel, 1959
- เราจะเรียกคอมพิวเตอร์โปรแกรมว่าได้เรียนรู้จากประสบการณ์ E เพื่อทำงาน T ได้โดยมีประสิทธิผล P เมื่อโปรแกรมนั้นสามารถทำงาน T ที่วัดผลด้วย P แล้วพัฒนาขึ้นจากประสบการณ์ E
- — Tom Mitchell, 1977

ตัวอย่าง

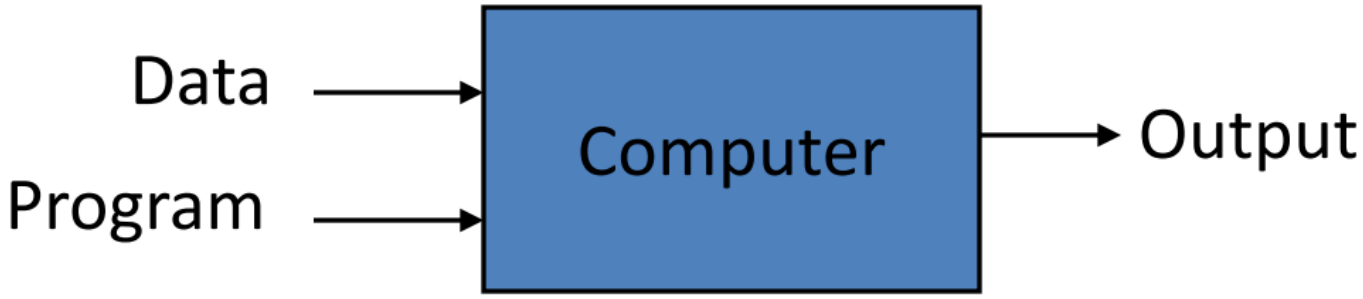
- ระบบป้องกันสแปมเมล
- งาน T คือการทำเครื่องหมายว่าอีเมลที่เข้ามานั้น เป็นสแปมหรือไม่เป็นสแปม
- โดยที่ประสบการณ์ E คือข้อมูลสำหรับใช้สอน (training data)
- ตัววัดประสิทธิภาพ P คือผลการแยกแยะได้อย่างถูกต้องว่าอีเมลอันไหนเป็นสแปมหรือไม่เป็นสแปม
- ในที่นี้คือ “accuracy หรือ ค่าความถูกต้องแม่นยำ”

E * T = P

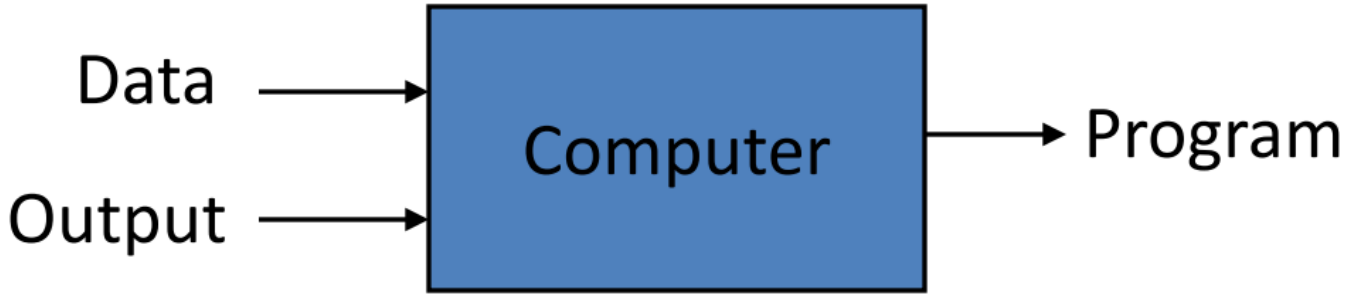
Experience	*	Task	=	Performance
Input Data: • Housing prices • Customer transactions • Clickstream data • Images		**Task:** • Predict prices • Segment customers • Optimize user flows • Categorize images		**Performance:** • Accurate prices • Coherent groupings • KPI lifts • Correctly sorted images

Experience (E) placed against **Task** (T) is measured by **Performance** (P) by ZLD

Traditional Programming



Machine Learning



เราอยู่ในยุคที่องค์ประกอบ 3 อย่างบรรจบกัน

1. Hardware

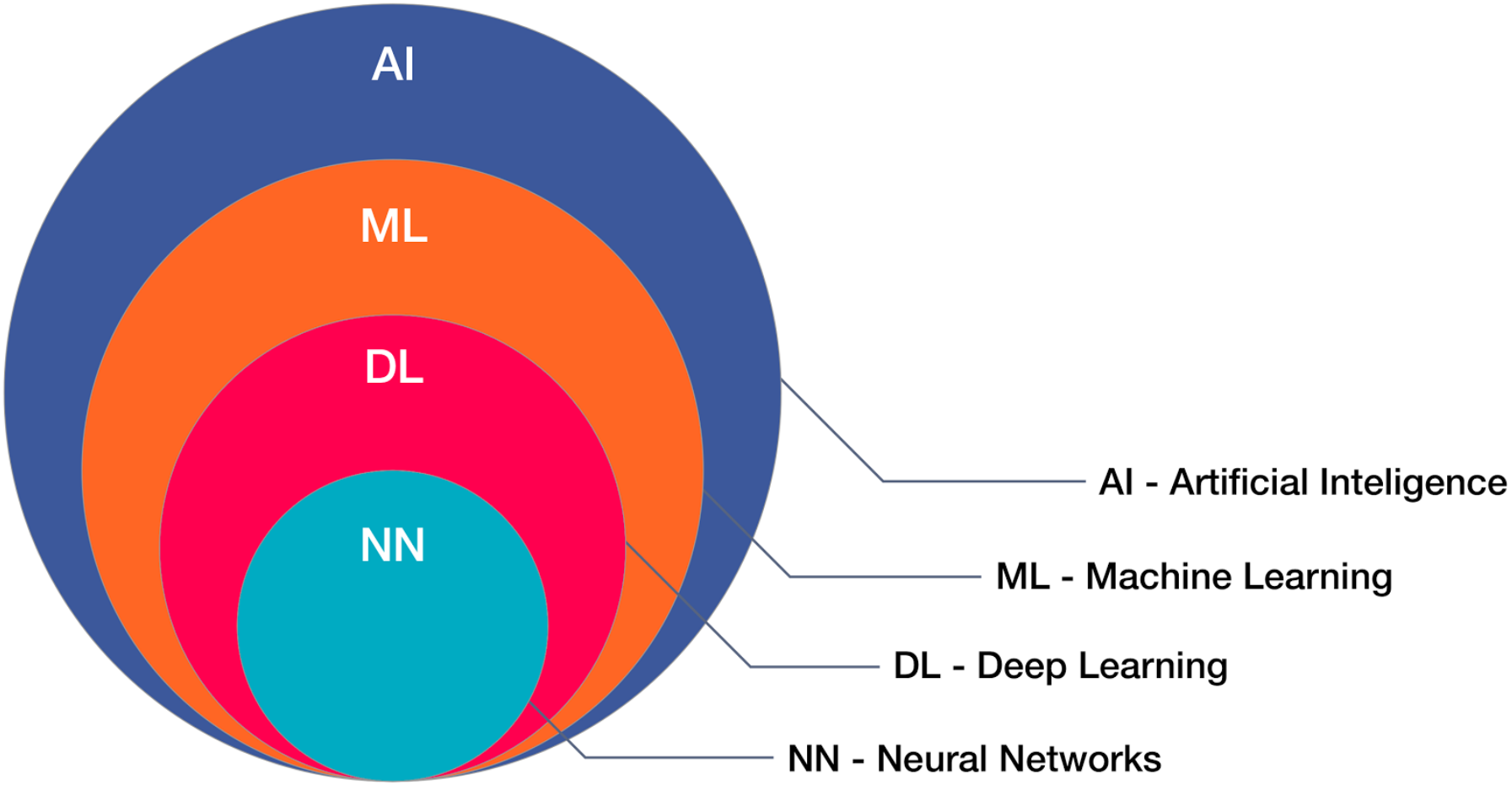
    : GPU!!!, Nano Technology!!!

2. Data

    : Big Data!!!, 4G/5G!!!

3. Knowledge

    : Research!!!, Consumer Product!!!


AI
ML
DL
NN
AI - Artificial Inteligence
ML - Machine Learning
DL - Deep Learning
NN - Neural Networks

Regression

Regression

- คือ **การวิเคราะห์ความสัมพันธ์ระหว่างตัวแปร โดยตัวแปรที่เราทราบค่า เรียกว่าตัวแปรอิสระ(X) และตัวแปรที่เราจะประมาณค่า(ต้องการที่จะรู้)เรียกว่าตัวแปรตาม(Y)** ซึ่งในการวิเคราะห์แบบ Regression นี้อาจจะมีตัวแปรอิสระ(X)มากกว่า 1 ตัวก็ได้

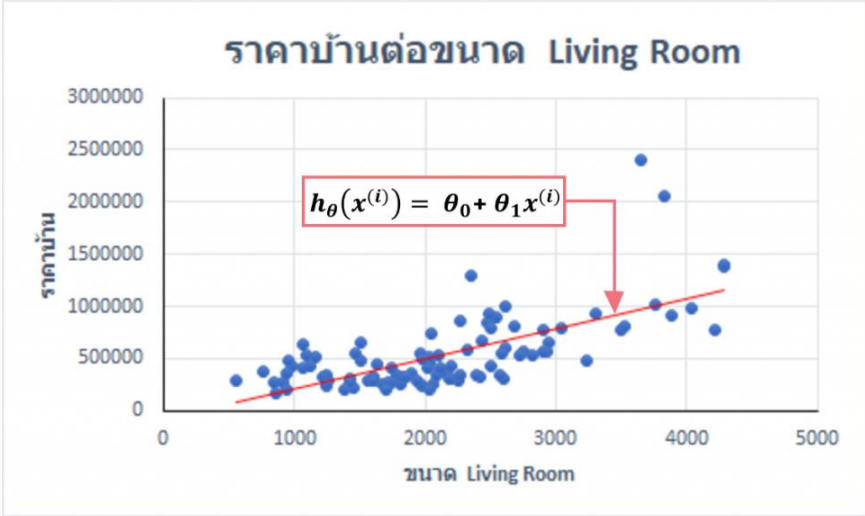
Training Set

ขนาด living (X)	ราคา (Y)
1430	310000
2950	650000
1710	233000
2320	580500
1090	535000
2620	605000
4220	775000
2250	292500
.....	.....

m = จำนวนชุดข้อมูลที่ใช้(100ชุด)

$X^{(i)}$ = ขนาด Living room(i)

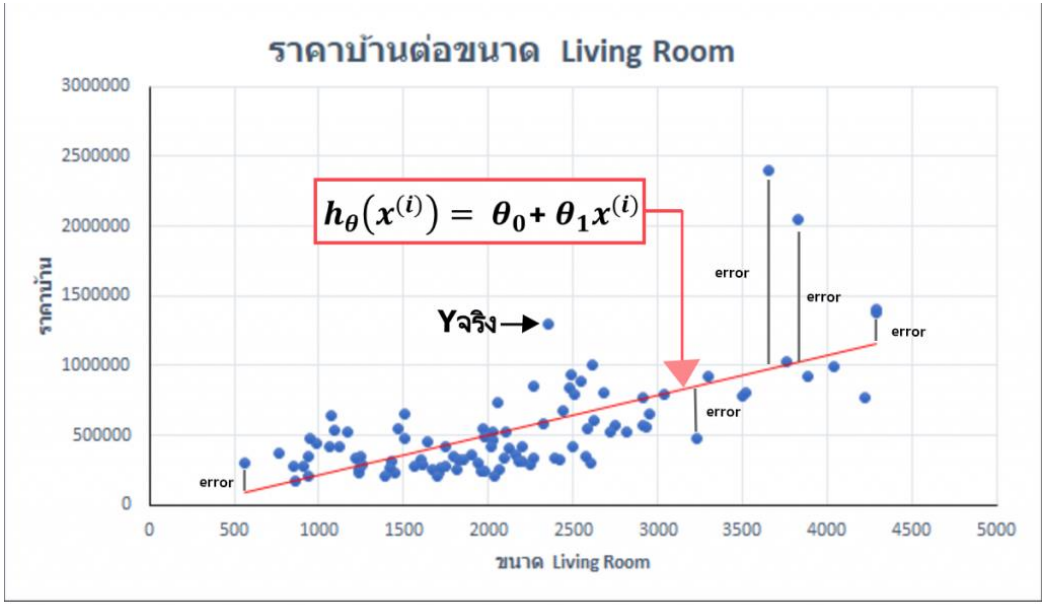
$Y^{(i)}$ = ราคาบ้านที่(i)



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

จากการพล็อตข้อมูลดังรูป จะเห็นได้ว่าข้อมูลมีการกระจายตัวแบบกราฟเส้นตรง ซึ่งสมการเส้นตรงโดยทั่วไปจะเขียนได้ว่า Y= aX +b แต่ใน ML เราใช้สมการ $h_\theta(x^{(i)}) = \theta_0 + \theta_1 x^{(i)}$ แทน โดยที่ค่าของ b,a = θ_0, θ_1 ตามลำดับ

Cost Function



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

จากรูปกำหนดให้

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

(Yจริง) = ราคาบ้านที่แท้จริง

$$J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2$$

Cost Function

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

(สมการเส้นตรงที่ได้จากการประมาณการ)

$y^{(i)}$ → Yจริง

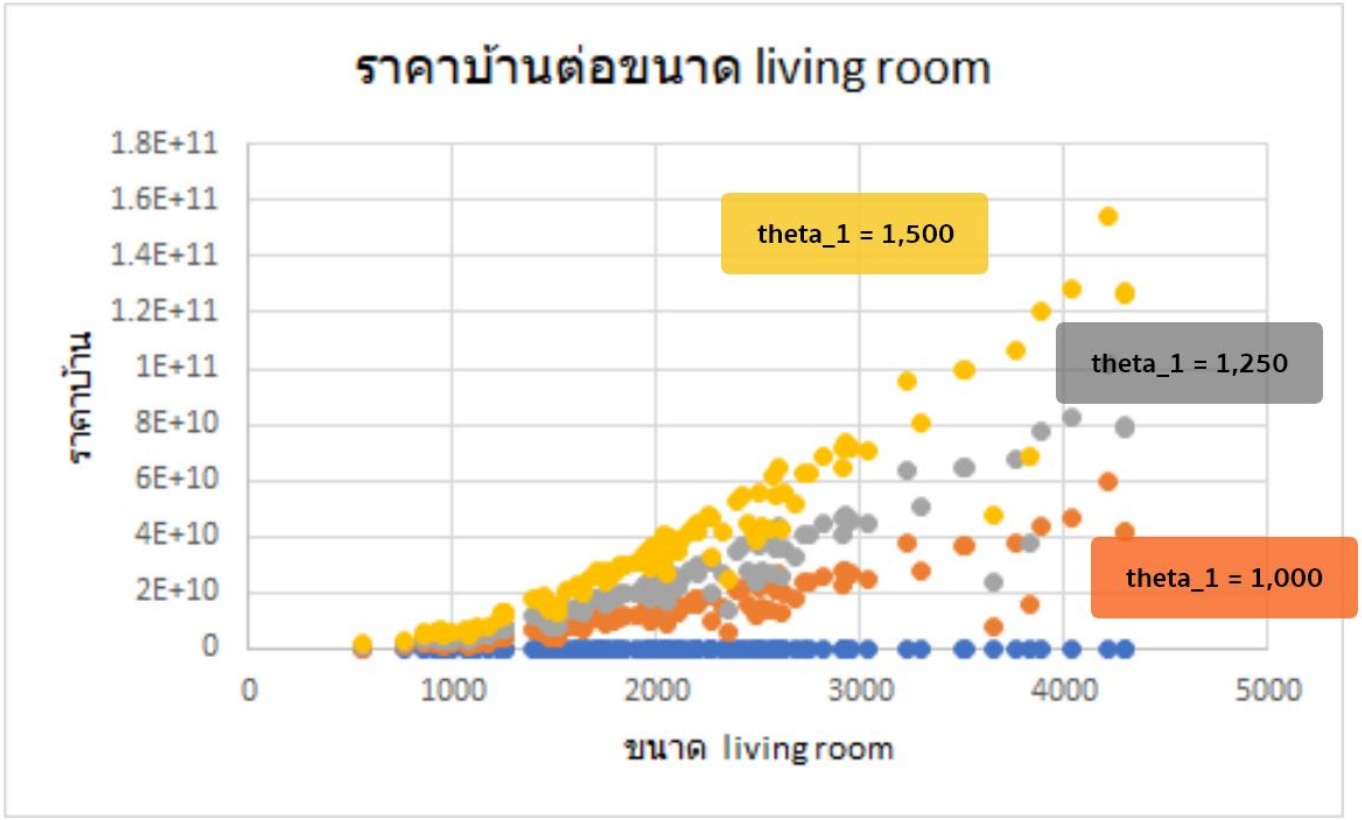
สิ่งที่เรามีตอนนี้คือ

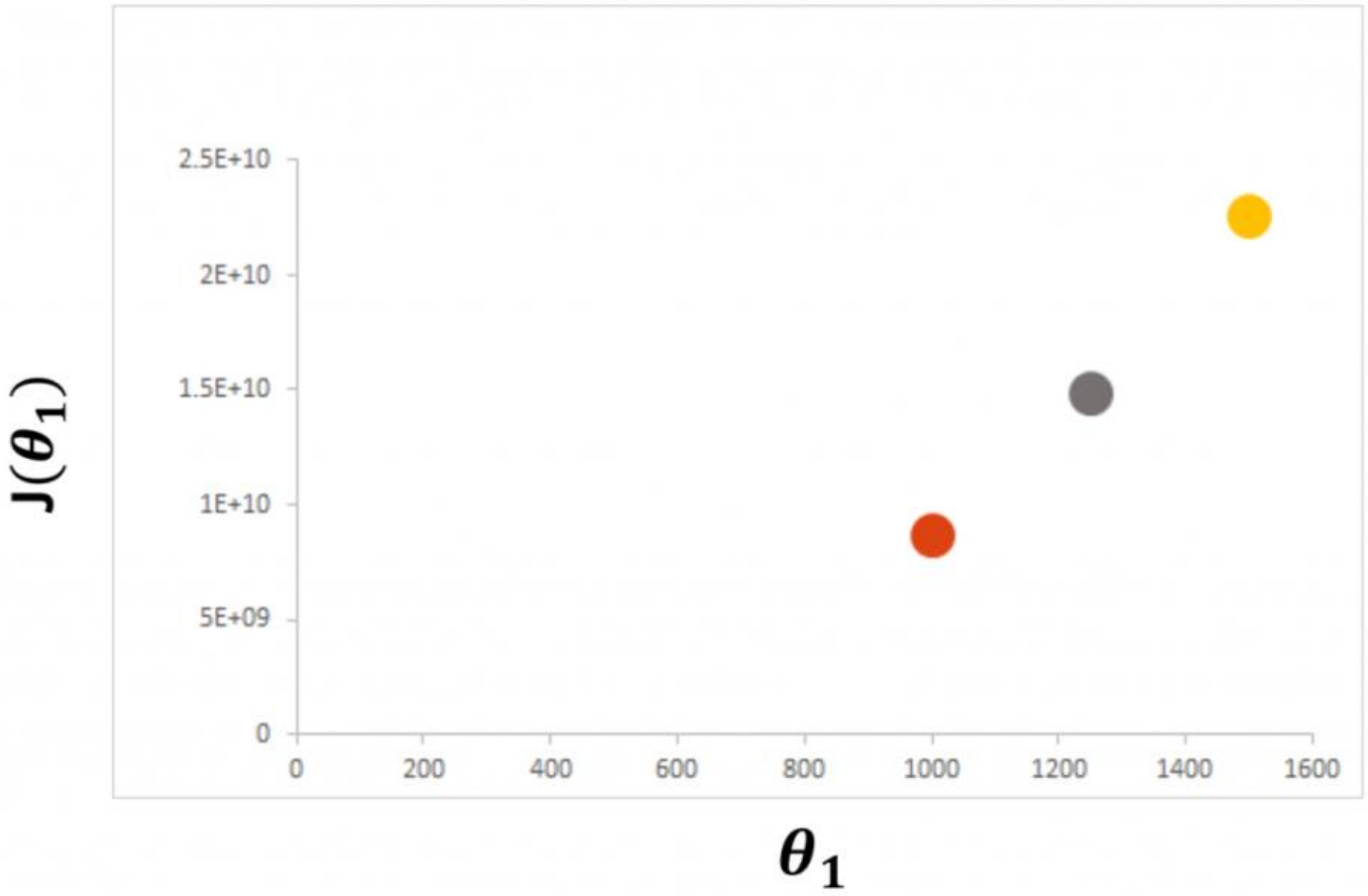
Hypothesis : $h_\theta(x^{(i)}) = \theta_0 + \theta_1 x^{(i)}$

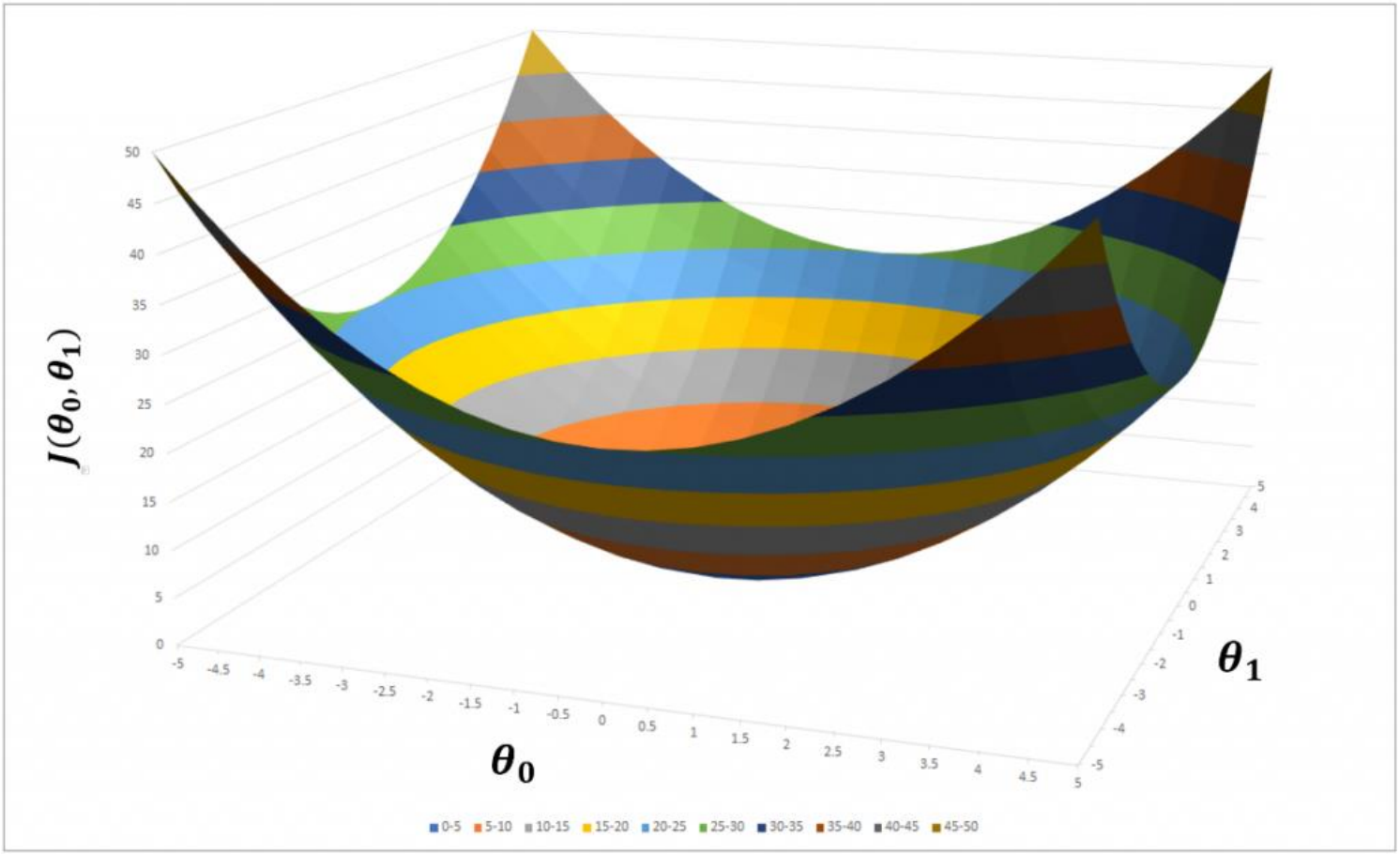
Parameter : θ_0, θ_1

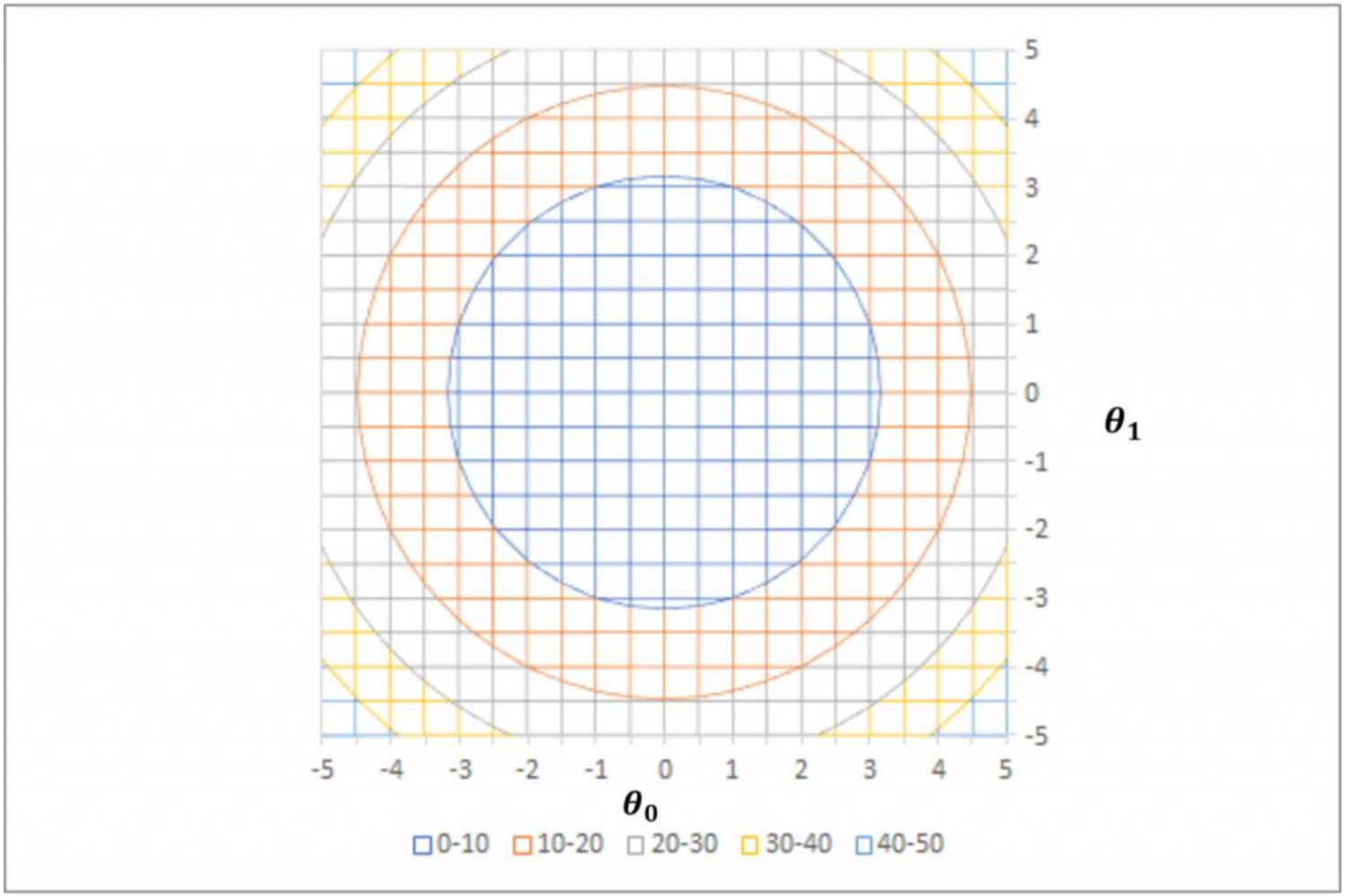
Cost Function : $J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2$

Goal : minimize $J(\theta_0, \theta_1)$

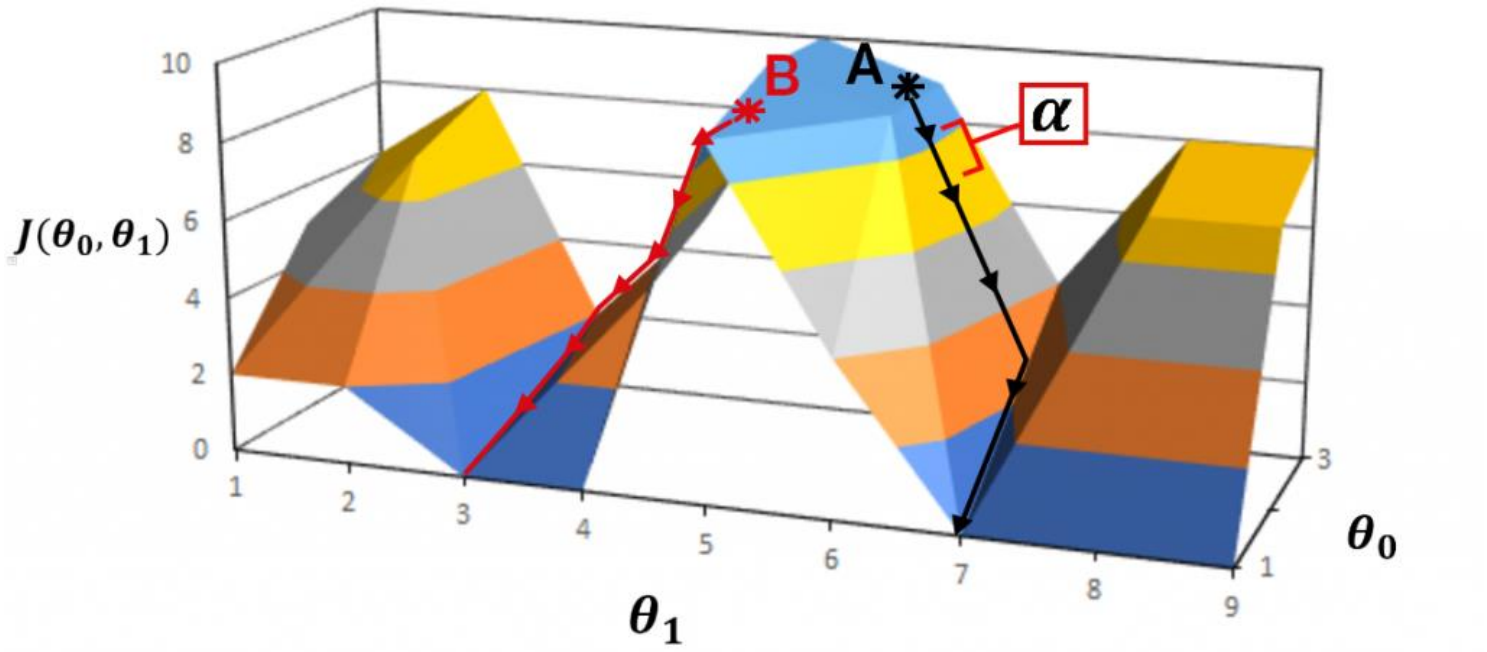

ราคาบ้านต่อขนาด living room
1.8E+11
1.6E+11
1.4E+11
1.2E+11
1E+11
8E+10
6E+10
4E+10
2E+10
0
ราคาบ้าน
0
1000
2000
3000
4000
5000
ขนาด living room
theta_1 = 1,500
theta_1 = 1,250
theta_1 = 1,000


2.5E+10
2E+10
1.5E+10
1E+10
5E+09
0
0
200
400
600
800
1000
1200
1400
1600
J(θ1)
θ1

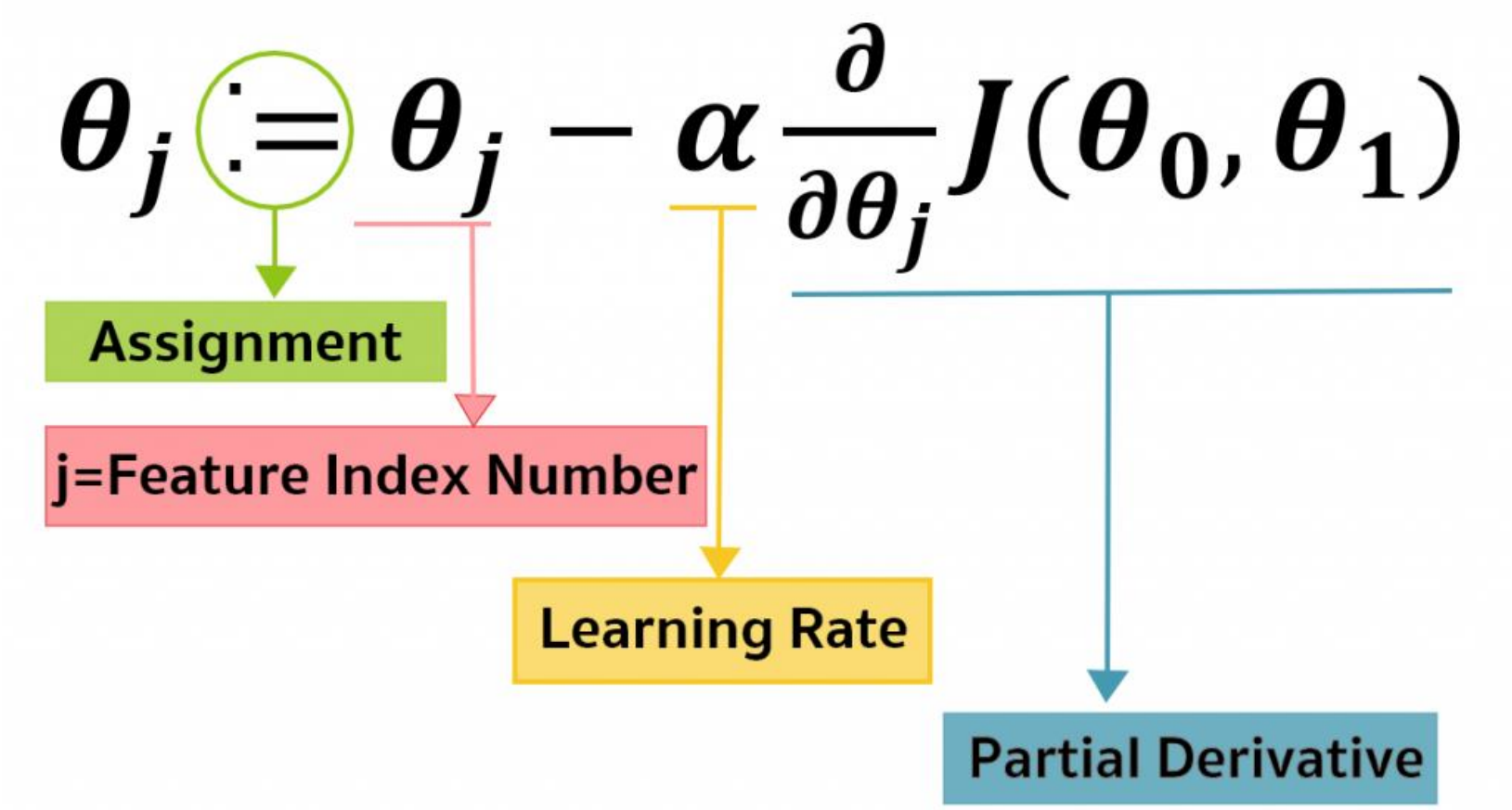

$J(\theta_0, \theta_1)$
50
45
40
35
30
25
20
15
10
5
0
-5 -4.5 -4 -3.5 -3 -2.5 -2 -1.5 -1 -0.5 0 0.5 1 1.5 2 2.5 3 3.5 4 4.5 5
θ_0
5 4 3 2 1 0 -1 -2 -3 -4 -5
θ_1
0-5 5-10 10-15 15-20 20-25 25-30 30-35 35-40 40-45 45-50


5
4
3
2
1
0
-1
-2
-3
-4
-5
θ_1
-5 -4 -3 -2 -1 0 1 2 3 4 5
θ_0
0-10
10-20
20-30
30-40
40-50

Gradient Descent



$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1)$$

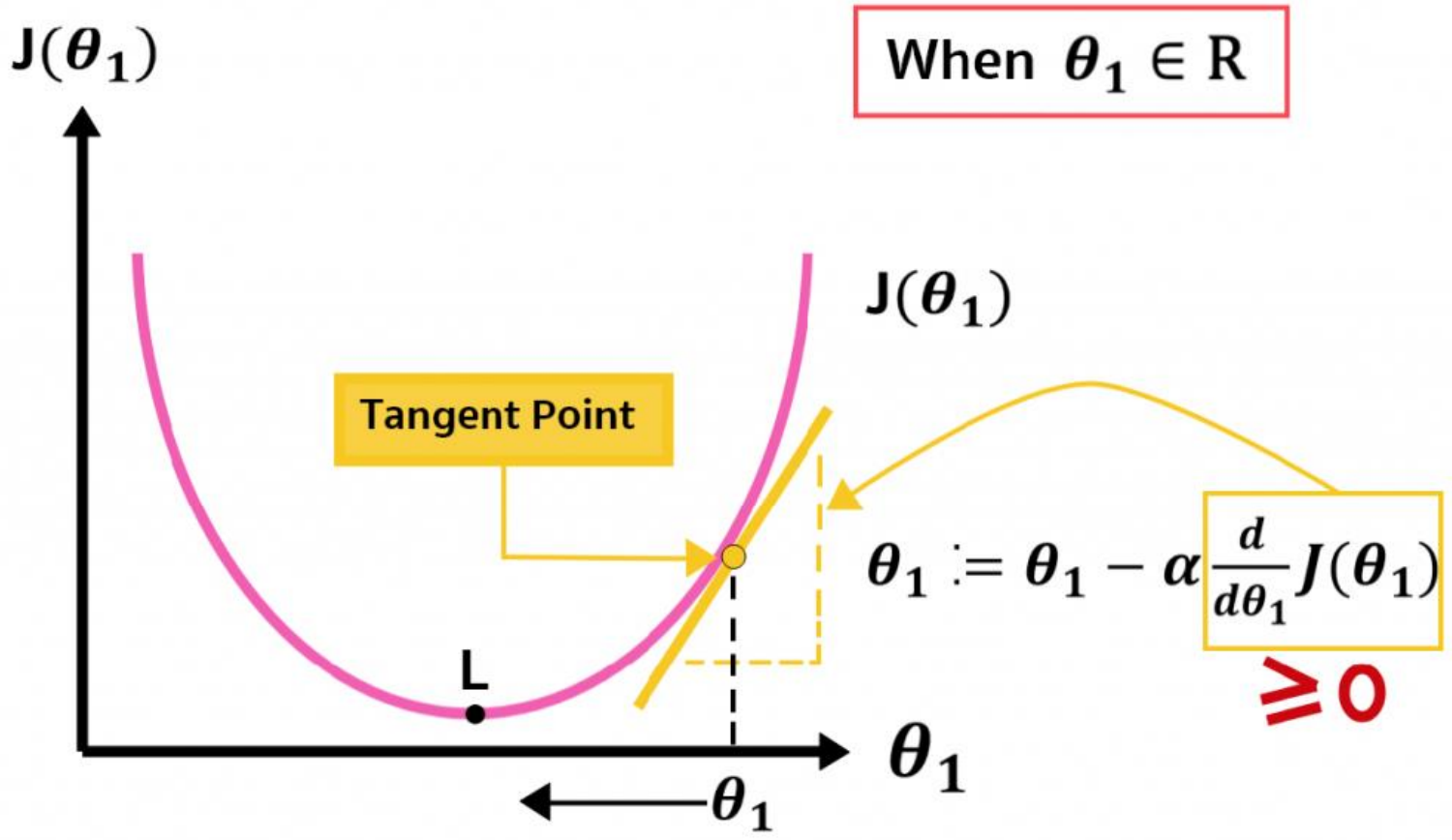

$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1)$
Assignment
j=Feature Index Number
Learning Rate
Partial Derivative

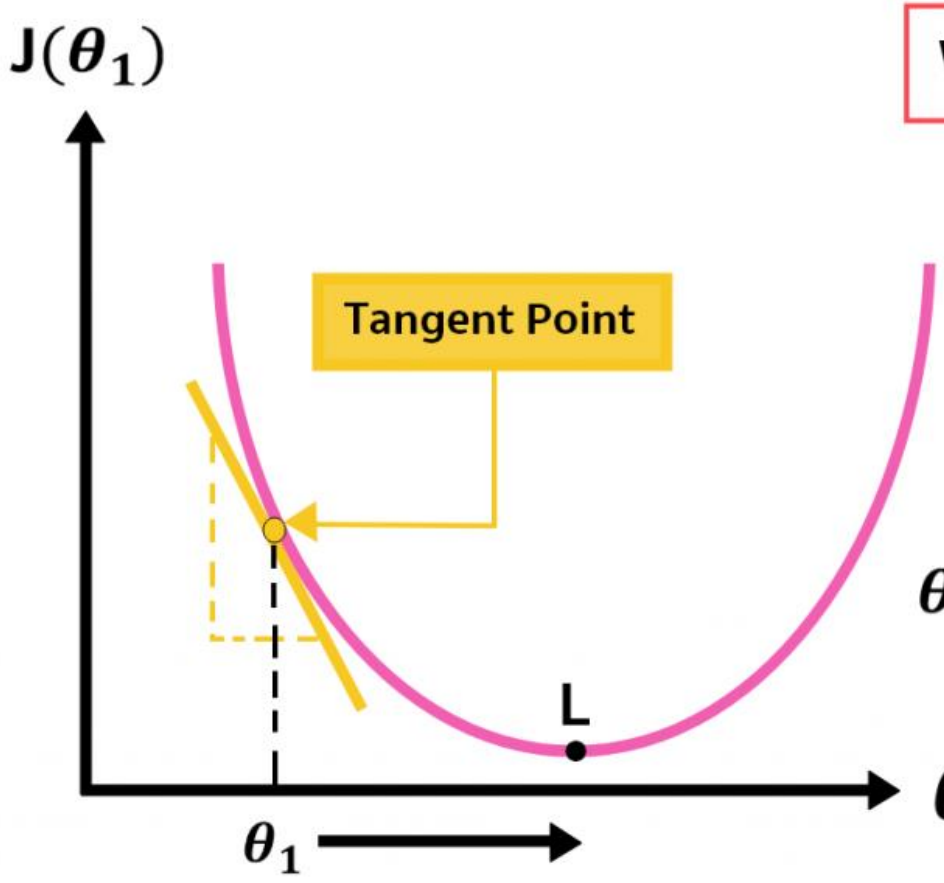
Derivative

- คือ ความชันของเส้นตรงที่สัมผัสจุดใดๆบนเส้นโค้ง (Tangent Point) โดยทั่วไป ถ้าหากสมการเป็นเส้นตรง เราสามารถหาความชันเส้นตรงได้โดยใช้สูตร

$$\text{Slope} = \frac{Y_2 - Y_1}{X_2 - X_1}$$

- แต่ในกรณีนี้ เราต้องการหาความชันของเส้นโค้ง ดังนั้นเราต้องใช้การ Derivative หรือ การ Diff


J(θ1)
When θ1 ∈ R
J(θ1)
Tangent Point
θ1 := θ1 − α d/dθ1 J(θ1)
≥ 0
L
θ1
θ1



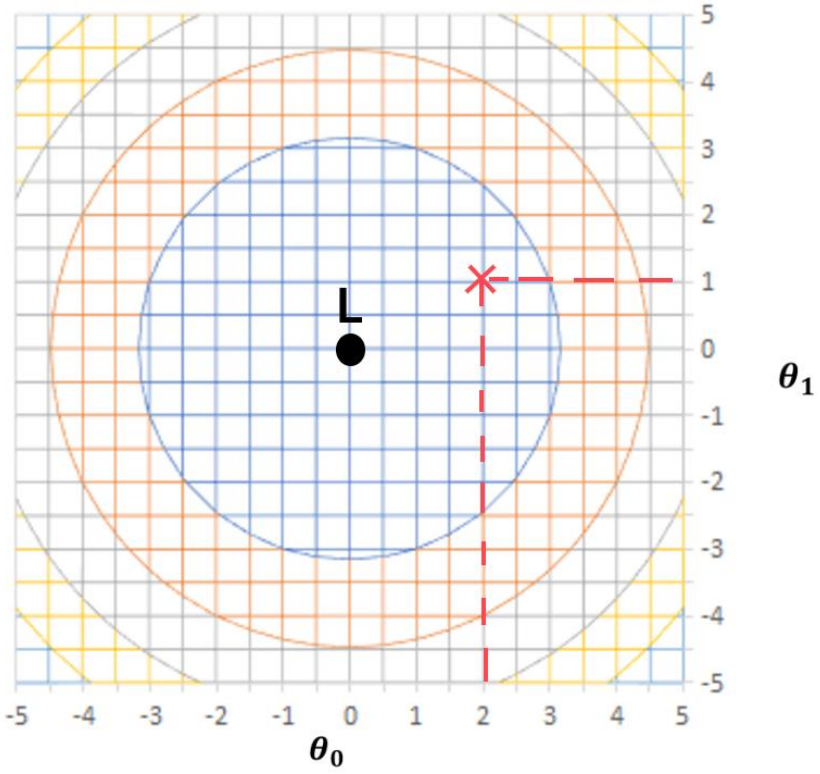
When $\theta_1 \in R$

$J(\theta_1)$

$$\theta_1 := \theta_1 - \alpha \frac{d}{d\theta_1} J(\theta_1)$$

≤ 0

Partial Derivative

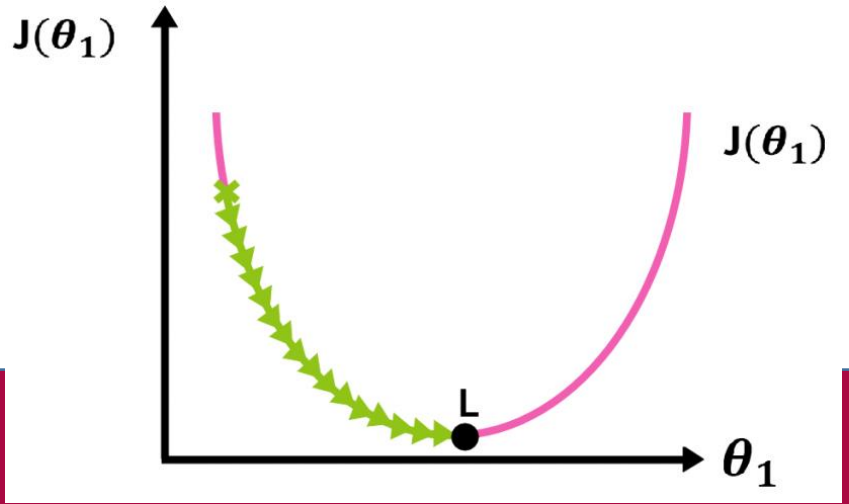


*จะหยุดเมื่อ Derivative=0 (Slope = 0) หรือ ถึงจุดที่ค่า Derivative มันเล็กมากๆ จนแทบไม่เปลี่ยนแปลงเราก็สามารถหยุดได้ ถึงแม้ว่า Derivative มันจะไม่เท่ากับ 0

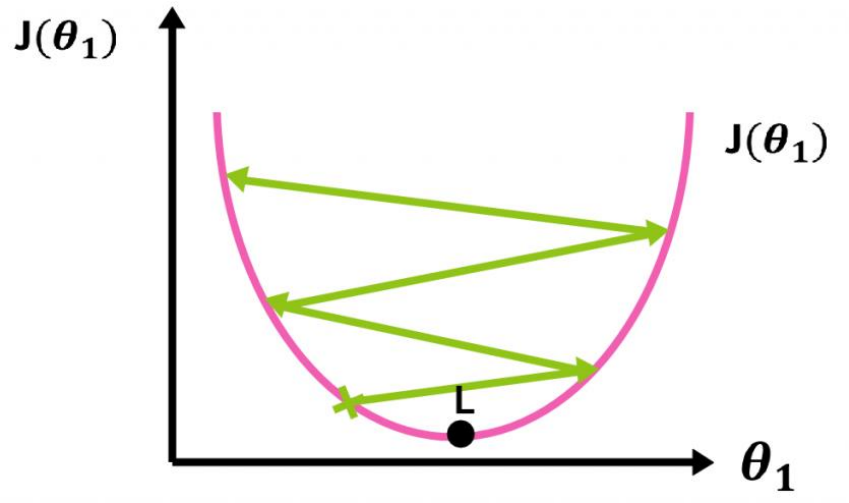
Learning Rate

- Learning Rate (α) คือ ขนาดของการก้าว ซึ่งเป็นค่าคงที่ (Fixed Learning Rate) เมื่อเราได้มีการกำหนดค่า(สมมุติ) j=0, j=1 แล้ว (Simultaneous Update)

Learning Rate ขนาดเล็ก -> ช้า



Learning Rate ขนาดใหญ่ -> Divergence



สรุป

สิ่งที่เรามี

Gradient Descent	Linear Regression
Repeat until convergence $\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1)$ when j=0 and j=1	$h_\theta(x^{(i)}) = \theta_0 + \theta_1 x^{(i)}$ $J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2$ แทนค่า

1. คำนวณตามสูตร Gradient โดยคำนวณค่า **Y predict** จาก ค่า Slope ($\widehat{\theta_1}$) และ Intercept ($\widehat{\theta_0}$) ปัจจุบัน **(Y predict = $\theta_0 + \theta_1 x^i$)**
2. คำนวณ Prediction errors (ความห่างระหว่างค่าที่ได้จากสมการประมาณการกับค่า Y จริง) จาก (**Y prediction** – Y)

3. Update the Intercept ($\widehat{\theta_0}$) :

   - หาค่า Derivative of Intercept ($\widehat{\theta_0}$) ได้จาก sum(errors) :

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

   - หาค่าที่เปลี่ยนแปลงไป (Adjustment) : **Adjustment = Learning rate * Derivative of intercept ($\widehat{\theta_0}$)**
   - หาจุดตัดใหม่ (New Intercept) : **New Intercept ($\widehat{\theta_0}$) = intercept เดิม – ค่า Adjustment**

4. Update the slope ($\widehat{\theta_1}$) :

- หาค่า Derivative of slope ($\widehat{\theta_1}$) ได้จาก sum(errors*input) :

  $$\sum_{i=1}^{m}(((\theta_0 + \theta_1 x^{(i)}) - y^{(i)}) * x^i)$$

- หาค่าที่เปลี่ยนแปลงไป (Adjustment) : **Adjustment = Learning rate * Derivative of Slope**
- หาความชันใหม่(New Slope) : **New Slope ($\widehat{\theta_1}$) = Slope เดิม – ค่า Adjustment**

สรุป (ต่อ)

- ทำจนกว่า Derivative จะเท่ากับที่ตั้งค่าไว้ เช่น 0
- แต่บางครั้งมันเสียเวลา เราเลยกำหนด จุด toleranceที่ใกล้ 0 มาก ๆ แทน ดังนั้น เมื่อเราทำ Derivative ไปเรื่อยๆจนค่า Magnitude มากกว่า ค่า Tolerance แสดงว่า เรา Convergence แล้ว วิธีนี้เรียกว่า Gradient Descent Approach
- โดยใช้ Magnitude คือ ขนาดของการ converge เข้าสู่จุด Local Minimum ซึ่ง บางครั้งมันก็มีทั้งค่าที่เป็นบวกและเป็นลบ แต่ว่าเราไม่สนใจว่ามันจะเป็น -0.1 หรือ 0.1 เพราะมัน converge ทั้งคู่

4. Update the slope ($\widehat{\theta_1}$) :

- หาค่า Derivative of slope ($\widehat{\theta_1}$) ได้จาก sum(errors*input) :
  $\sum_{i=1}^{m}(((\theta_0 + \theta_1 x^{(i)}) - y^{(i)}) * x^i)$
- หาค่าที่เปลี่ยนแปลงไป (Adjustment) : **Adjustment = Learning rate * Derivative of Slope**
- หาความชันใหม่(New Slope) : **New Slope ($\widehat{\theta_1}$) = Slope เดิม – ค่า Adjustment**

5. คำนวณหา Magnitude of the gradient

$$\sqrt{(Derivative\ of\ slope)^2 + (Derivative\ of\ intracept)^2}$$

6. ตรวจสอบว่ามันลู่เข้าแล้วหรือไม่ : ถ้า Magnitude < Tolerance = Convergence!