

# Introduction to Machine Learning

Kasetsart University & DEPA

# ตัวอย่างเทคโนโลยีของ Machine Learning

## 1. Clustering

- K-mean Clustering
- DBSCAN

## 2. Classification

- k-Nearest Neighbors
- Decision Tree
- Naive Bayes

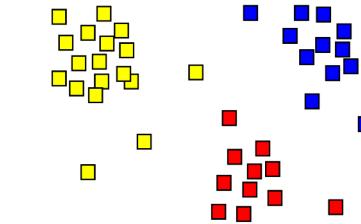
## 3. Regression

# Unsupervised Machine Learning

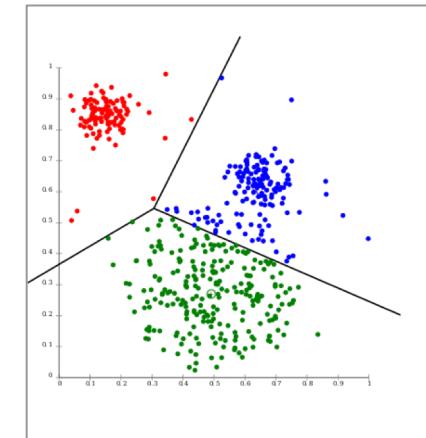
ไม่กล่าวว่า คน จัดกลุ่ม เต่าลง → เนื่องมั่น unsupervised

**Clustering** มีเป้าหมายเพื่อจะจัดกลุ่มตัวอย่างที่มีคุณลักษณะที่ใกล้เคียงกันให้อยู่รวมเป็นกลุ่มเดียวกันโดยคำนวณจากระยะห่างของแต่ละตัวอย่าง

- **Medical patients**
  - Feature values: อายุ, เพศ, อาการที่ 1, อาการที่ 2, ผลการรักษา 1, ผลการรักษา 2
- **Web pages**
  - Feature values: URL domain, length, #images, heading 1, heading 2, ..., heading n
- **Products**
  - Feature values: หมวดหมู่, ชื่อ, ขนาด, น้ำหนัก, ราคา



Source: By Cluster-2.gif: hellispderivative work: Wgabrie (talk) - Cluster-2.gif, Public Domain, <https://commons.wikimedia.org/w/index.php?curid=9442336>



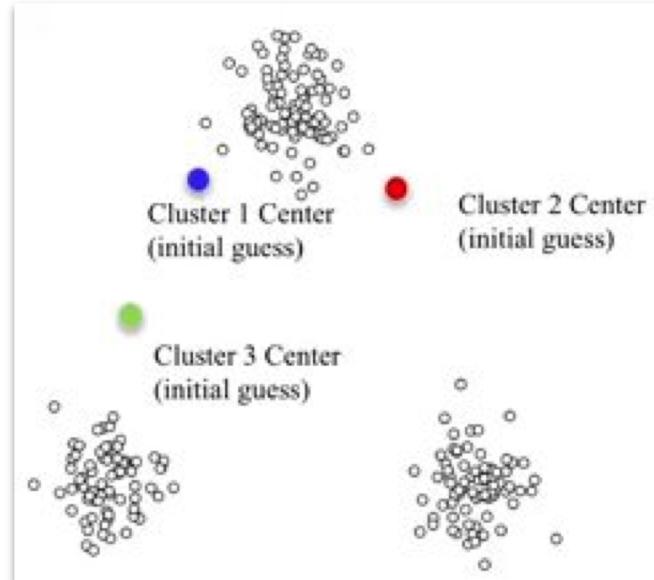
Source: By Chire - Own work, CC BY-SA 3.0, <https://commons.wikimedia.org/w/index.php?curid=1708571>

# Clustering Algorithms

- แยกตัวอย่างออกเป็นกลุ่มๆ ตัวอย่างในแต่ละกลุ่มนี้มีความ “คล้ายกัน” มากกว่ากับตัวอย่างนอกกลุ่ม
  - ไม่มี label ให้ ไม่เดลต้องจัดกลุ่มเอง (เป็น Unsupervised Learning)
- ไม่เดลจะให้รหัสกลุ่มกับทุกตัวอย่างเป็นผลลัพธ์ของไม่เดล
- การจัดกลุ่มแบ่งเป็นสองประเภท
  - จัดกลุ่มแบบยาก: ข้อมูลแต่ละตัวอย่างจะอยู่ในกลุ่มเดียว
  - จัดกลุ่มแบบ fuzzy: ตัวอย่างอาจอยู่ในหลายกลุ่มได้ และจะมี weight หรือความน่าจะเป็นให้ว่าโอกาสที่ตัวอย่างนี้จะอยู่ในแต่ละกลุ่มเป็นเท่าใด

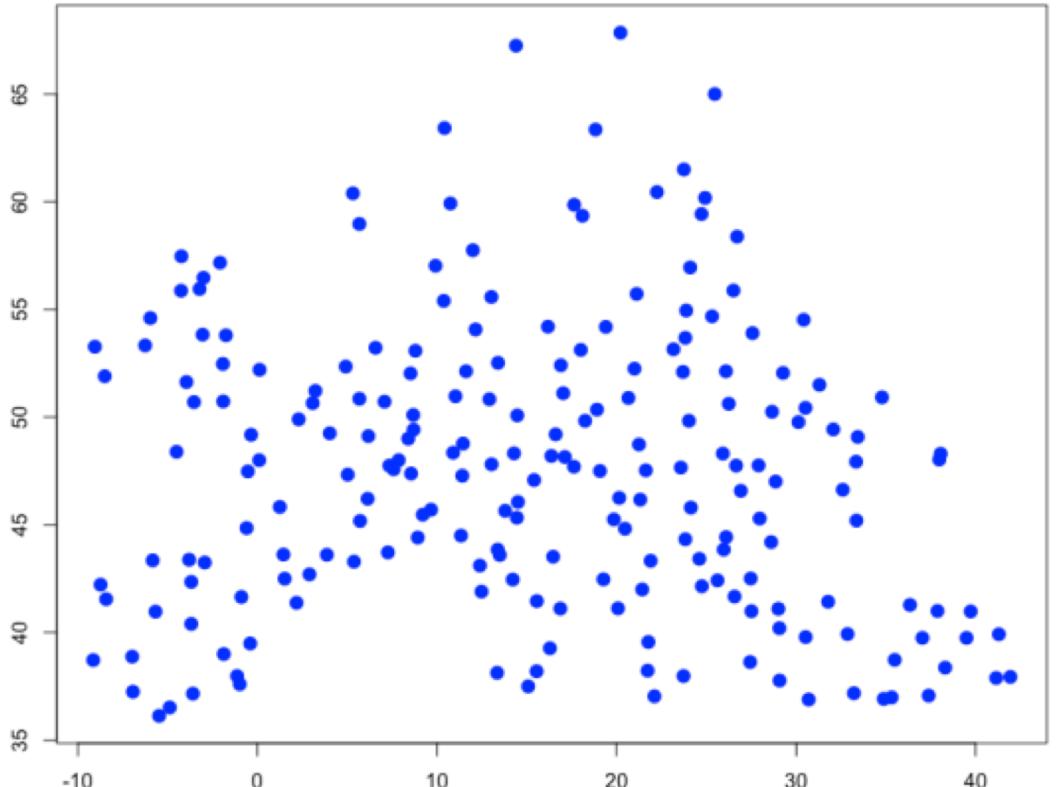
# K-Means Clustering

- วิธีการจัดกลุ่มแบบง่าย
- กำหนดค่า K หมายถึง กำหนดจำนวนของกลุ่มที่ต้องการจัด และสุ่มตำแหน่งจุดกลางของแต่ละกลุ่ม
- คำนวณระยะทางระหว่างแต่ละตัวอย่างกับจุดกลางทั้งหมด และกำหนดให้ตัวอย่างมีน้ำหนักกับกลุ่มที่ใกล้จุดกลางที่สุด  
ศูนย์กลาง = ค่าที่ปั้นขึ้น
- คำนวณจุดกลางของแต่ละกลุ่มใหม่โดยใช้ค่าเฉลี่ยของสมาชิกกลุ่มทุกตัวอย่าง  
หาค่าเฉลี่ยไปทั้งหมด
- ดำเนินการจัดกลุ่มของทุกกลุ่มเข้าสถานะเสถียร



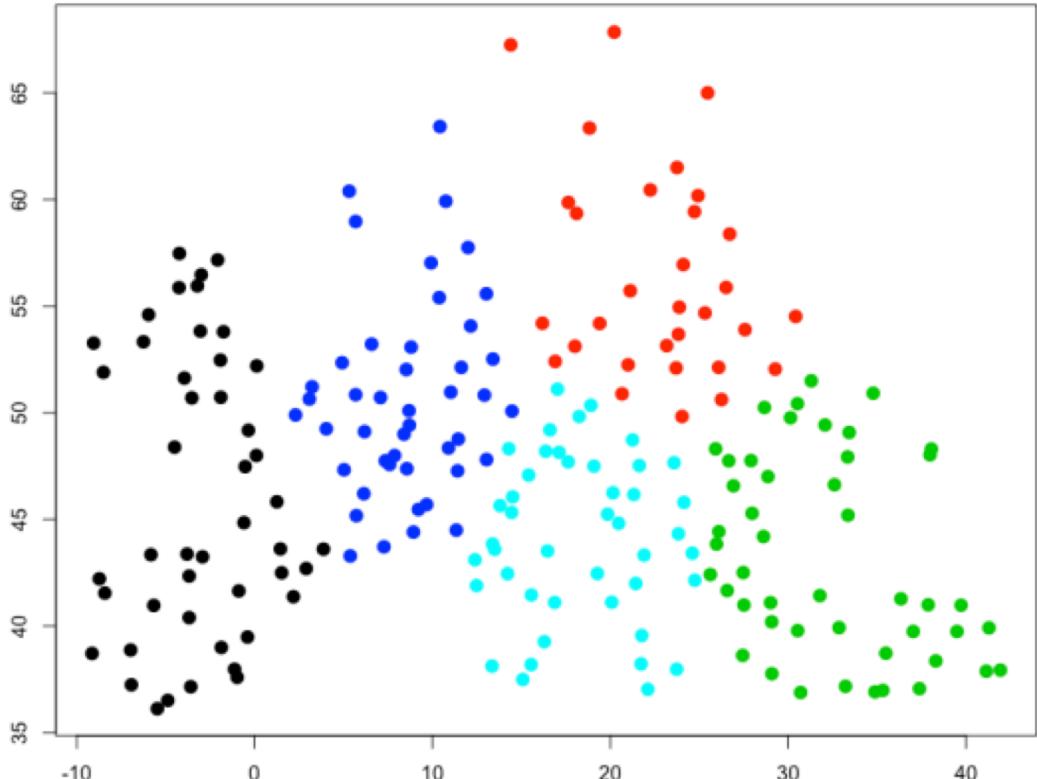
# K-Means Clustering Example

- Clustering European Cities



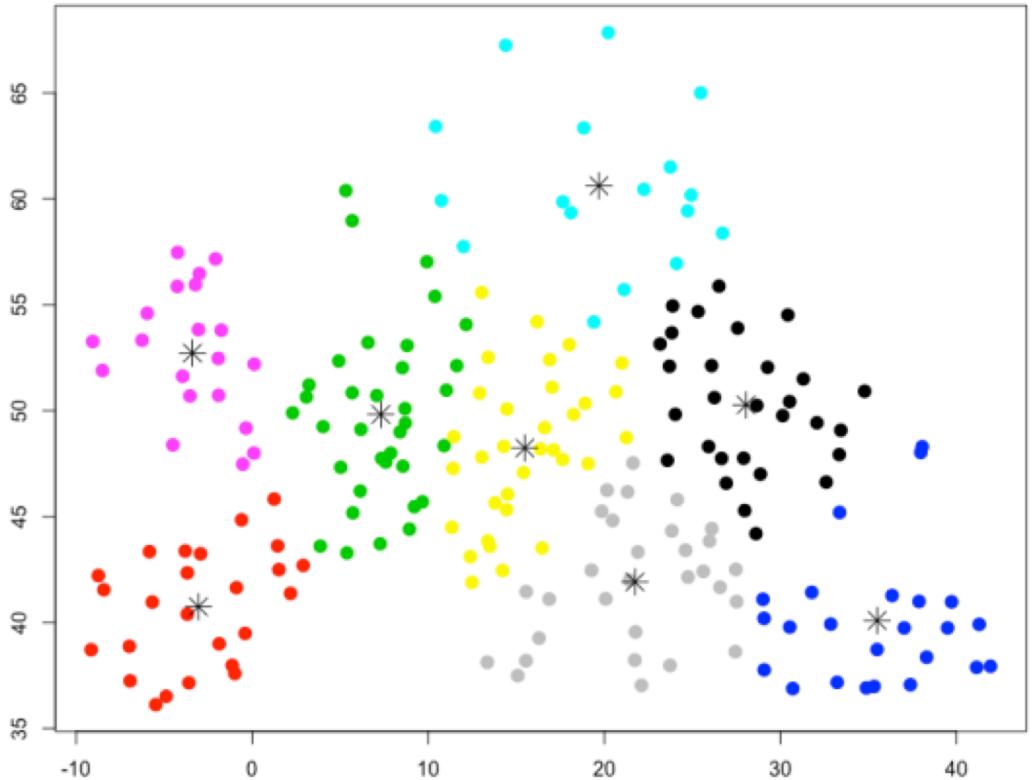
# K-Means Clustering Example (con't)

- K = 5



# K-Means Clustering Example (con't)

- K = 8 with centroid



# ข้อดีข้อเสียของ K-Mean Clustering

## ข้อดี

- เข้าใจง่าย ใช้ง่าย
- เหมาะกับกลุ่มที่มีจำนวนข้อมูลใกล้เคียงกัน และมีรูปร่างเป็นทรงกลม

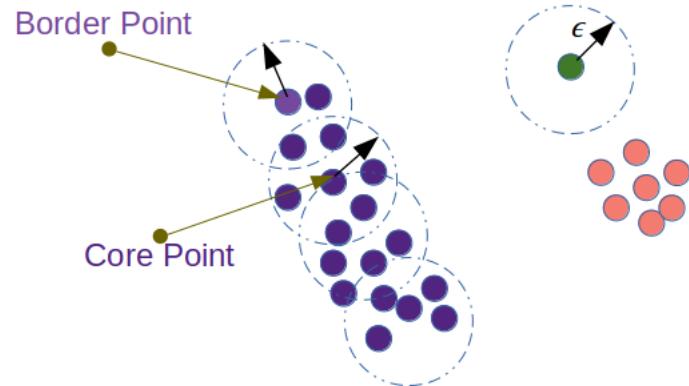
## ข้อเสีย

- เซ็นเซอร์พต่อตำแหน่งเริ่มต้นของจุดกลางของกลุ่ม
- ต้องปรับค่า feature ให้อยู่ในสเกลเดียวกัน *↑ ต้องเป็นตรีกูล!*
- ใช้ไม่ได้กับ feature ที่เป็น category
- ไม่เหมาะกับกลุ่มที่มีลักษณะเป็นวงรียาว หรือจำนวนข้อมูลแตกต่างกันมากในแต่ละกลุ่ม  
*↑ พนธุ์ตัวนั้น มีค่า ที่น่าจะไม่*

# DBSCAN

ໃນຮັບສ່ວນ ນັກອົງຮອນຄຸມນິ້ນອອງໄຫວ່ຂຶ້ນ

- ເຕັກນີກໃນການແບ່ງກລຸ່ມໂດຍດູຕາມຄວາມທະນາແນ່ນຂອງ  
ຂ້ອມູລ
- ກຳນົດພື້ນທີ່ໃນກາຣວິເຄຣະທີ່ເປັນວົງກລມຮັກນີ້  
epsilon ( $\epsilon$ ) ແລະ ຈຳນວນຂ້ອມູລທີ່ນ້ອຍກີ່ສຸດໃນການ  
ນັບເປັນພື້ນທີ່ທະນາແນ່ນ
- ຈຸດທີ່ອູ່ຢູ່ໃນພື້ນທີ່ທີ່ຂ້ອມູລທະນາແນ່ນຈະເຮີຍກວ່າ Core
- ຈຸດທີ່ຕິດກັບພື້ນທີ່ທະນາແນ່ນແຕ່ຈຳນວນຂ້ອມູລໄຟ່ພອຈະ  
ເຮີຍກວ່າ Border Point
- ຈຸດທີ່ໄຟ່ຕິດກັບພື້ນທີ່ທະນາແນ່ນເຮີຍກວ່າ Noise



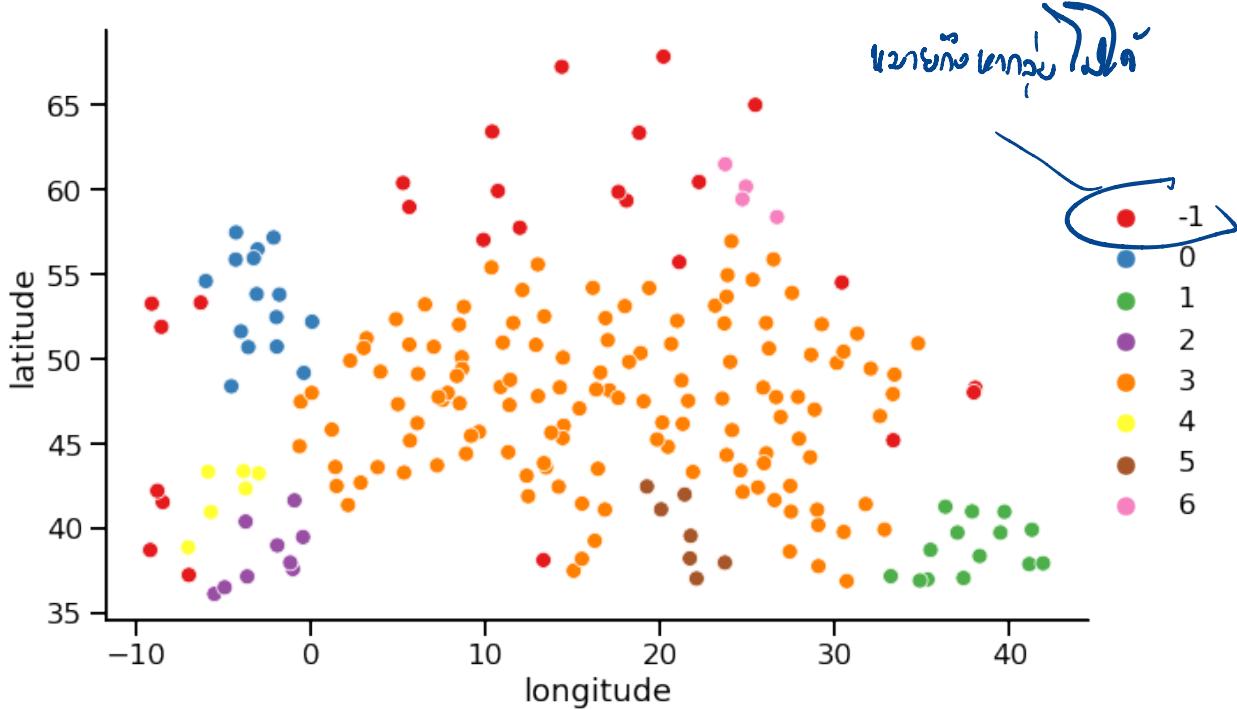
$$N_{Eps}(p) = \{q \in D \text{ such that } dist(p, q) \leq \epsilon\}$$

ກົດທີ່ກັບ  $\epsilon = 1$  unit, MinPts = 7

<https://towardsdatascience.com/dbSCAN-algorithm-complete-guide-and-application-with-python-scikit-learn-d690cbae4c5d>

# DBSCAN Example

- $\varepsilon = 2.6$
- จำนวนจุด = 5



# ข้อดีข้อเสียของ DBSCAN

- ข้อดี
  - จัดกลุ่มตามความหนาแน่นของชุดข้อมูล
  - สามารถจัดกลุ่มที่ขนาดของกลุ่มไม่เท่ากันได้ดี
  - หาข้อมูลที่ไม่เข้ากลุ่มได้
  - ไม่ต้องระบุจำนวนกลุ่มในตอนแรก → ใหม่ DBSCAN ไม่การันตีว่าจะ “จัดได้” ก็ “กอบ”
    - $\varepsilon$ , min point อาจ “หัก” 1 กรณีหนึ่ง
    - รู้ว่า “จะ”
    - แบบ loop 10 แก้ก็ “ปั๊บ”
- ข้อเสีย
  - การจัดกลุ่มขึ้นกับค่า  $\varepsilon$  กับจำนวนข้อมูลน้อยสุดในกลุ่ม อาจต้องปรับจูนค่าหลายรอบ
  - หากกลุ่มที่มีความหนาแน่นต่างกันไม่ได้

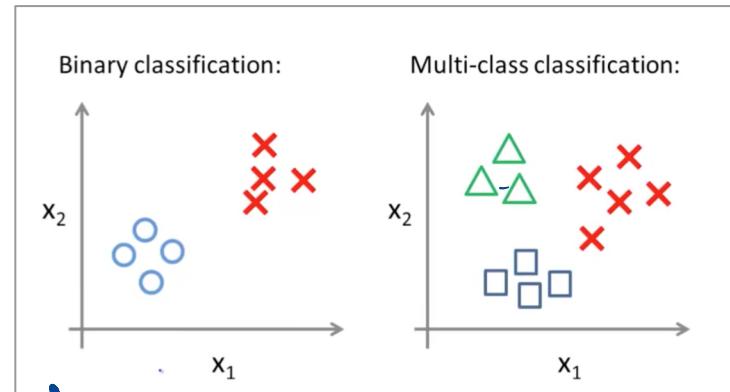
# Classification

## Classification

คือ การจำแนกหมวดหมู่ของข้อมูลตามพอลเบลย์ (label) โดยใช้ไมเดลที่ผ่านการฝึก

จำแนก

- Feature Value = Numeric / Categorical
- LABEL = Categorical Output Value
- Example
  - Features: อายุ, เพศ, รายได้, อาชีพ
  - Label: เป็นผู้ซื้อ, ไม่เป็นผู้ซื้อ
- ประเภทของ Classification
  - Binary Classification Yes, No
  - Multi-class Classification
  - Multi-label Classification

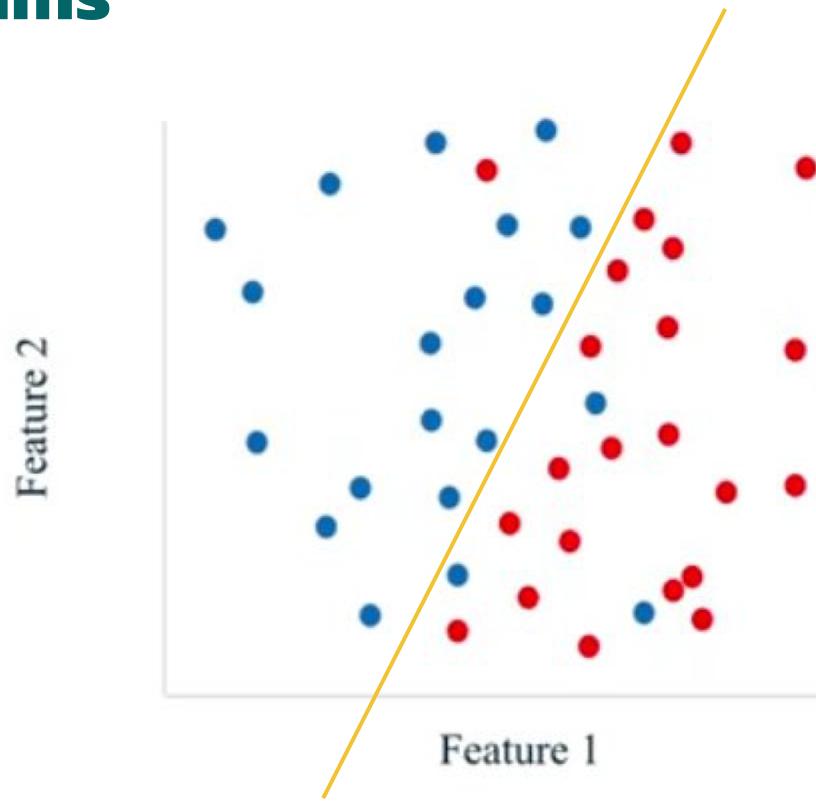


Source: [https://www.datasciencecentral.com/machine-learning-stanford/master-w2\\_logistic\\_regression\\_regularization/multiclass\\_classification.png](https://www.datasciencecentral.com/machine-learning-stanford/master-w2_logistic_regression_regularization/multiclass_classification.png)

ในกราฟ: ให้ class ที่ 1 คือ Taiwan  
ที่ Taiwan is a —  
country 10%  
city 5%  
part of china 1%

# Classification Algorithms

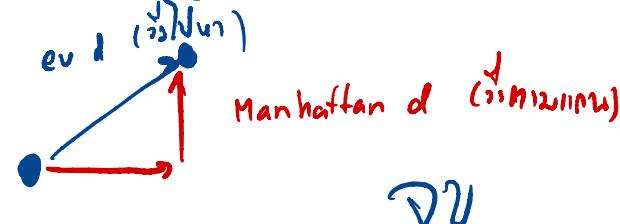
1. K-nearest Neighbors (KNN)
2. Decision Trees
3. Naïve Bayes



Alg. ห้ามคิดครอฟ!

## K-nearest Neighbors (KNN)

- เริ่มต้น Train โดยเดลด้วยการจำข้อมูลของ Training กั้งหมด
- ข้อมูล Train แต่ละตัวอย่างประกอบด้วย feature:  $x_1, x_2, \dots, x_n$  เมื่อ  $n$  คือจำนวน feature กั้งหมด และ  $l$  คือ label (class) ของตัวอย่างนั้น
- เมื่อมีตัวอย่างใหม่ (Test) เข้ามา โดยจะคำนวณหาตัวอย่างเก่าที่ใกล้กับข้อมูล Test ที่สุด  $k$  ตัว ( $k$  คือพารามิเตอร์ที่ผู้สร้างโมเดลปรับได้ ปกติคือ 5)
- วิธีการคำนวณหาระยะห่างของจุดข้อมูล
  - Euclidean distance =  $d(x, y) = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2}$
  - Manhattan distance =  $d(x, y) = |x_1 - y_1| + |x_2 - y_2|$
- จากนั้นดูว่า  $k$  ตัวอย่างเก่าที่ใกล้ที่สุดมี label อะไรมากที่สุด จะเป็น label ของ Test

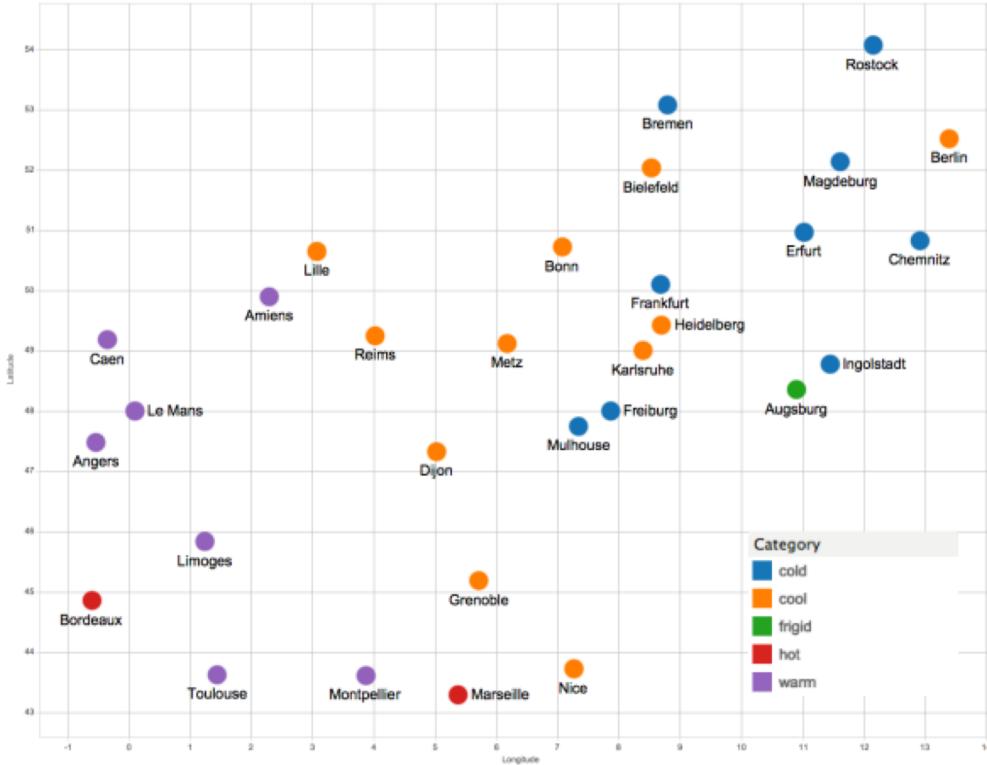


จริง

# K-nearest Neighbors Example

- City Temperatures Prediction – France and Germany
- **Features:** longitude, latitude
- **Distance:** Euclidean method
- **Labels:** frigid, cold, cool, warm, hot
- Training Example
  - Nice (7.27, 43.72) **cool**
  - Toulouse (1.45, 43.62) **warm**
  - Frankfurt (8.68, 50.1) **cold**

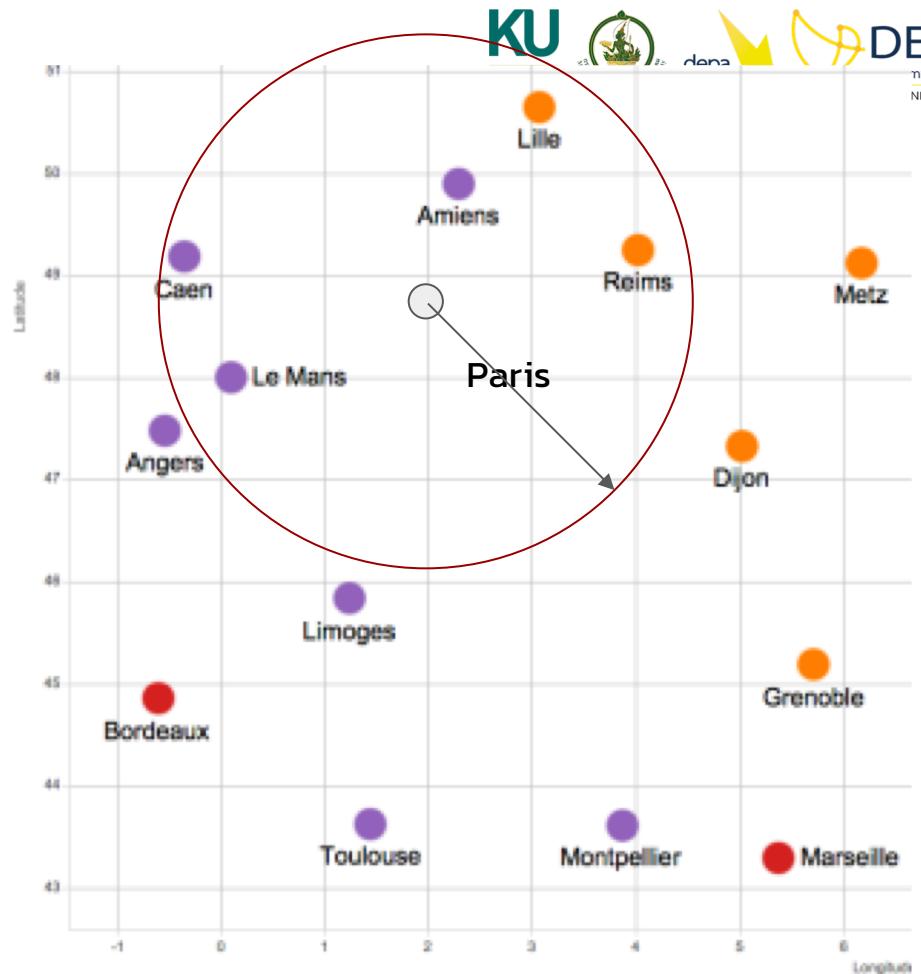
# K-nearest Neighbors Training



# K-nearest Neighbor Test

- New City: Paris Lat: 48, Long: 2
- K = 5
- Closest Cities and Labels

Amiens	Warm
Le Mans	Warm
Caen	Warm
Reims	Cool
Lille	Cool



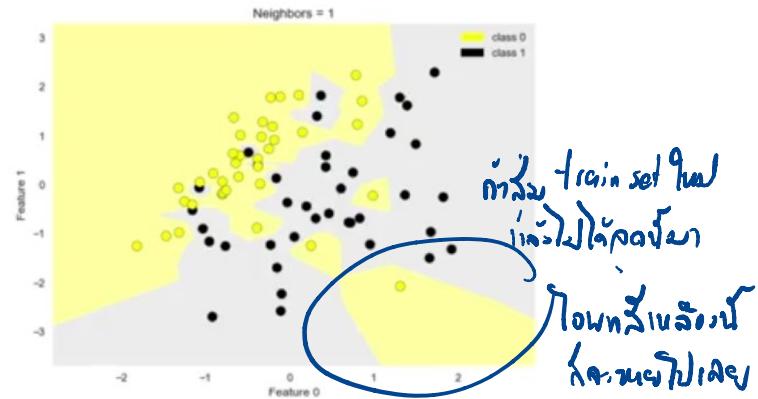
- Paris is labeled Warm

# Nearest neighbors classification ( $k=1$ )

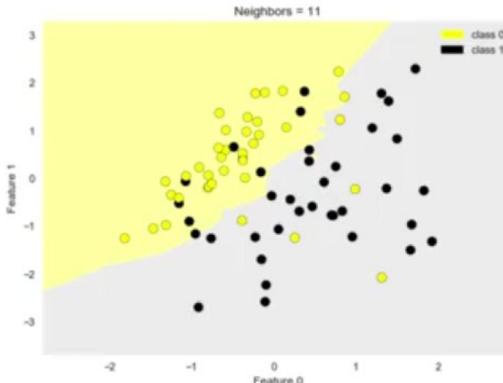
## ค่า $k$

- ค่า  $k$  ที่ต่ำ (เช่น  $k=1$ ) จะทำให้โมเดล overfit
  - เพราะ sensitive ต่อความแปรปรวนของข้อมูล
  - พิจารณาแค่ข้อมูลตัวเดียวที่อยู่ใกล้ ถ้าตัวอย่างใกล้สุดเพ้ออญมี label ไม่เหมือนตัวอื่นรอบๆ จะทำให้โมเดลเข้าใจผิดได้
- ค่า  $k$  ที่สูง (เช่น  $k = 50$ ) จะทำให้โมเดล underfit

overfit!



# Nearest neighbors classification ( $k=11$ )



# ประเด็นอื่นของ k-Nearest Neighbors

▶ หมายเหตุทั่วไป

- อาจต้องทำการ rescale ค่าของ feature ที่มี range ต่างกันมาก
  - เงินเดือน: 6000 - 200000 ความแตกต่างของเงินเดือนที่ 20 บาทนั้นน้อยมาก
  - อายุ: 18-82 ความแตกต่างของอายุที่ 20 ปีนั้นสูงมาก
  - Rescale ให้ทั้งเงินเดือนและอายุอยู่ในช่วง [0, 1] ทั้งคู่
- นำ feature ที่เป็น category มาสร้างโมเดลยาก
  - ต้องเปลี่ยน categorical feature ให้เป็น numerical และให้ค่า distance ด้วย
  - สามารถใช้ one-hot encoding ช่วยได้

หากห้องเยาแทนเมือง 0, 1, 2, 3 ในกราฟนี้มีข้อมูลเท่านั้น  
ซึ่งมีเมือง 0, 1, 2, 3 ในกราฟนี้มีข้อมูลเท่านั้น  
One-hot encoding 0, 1, 2, 3 เมือง! ห้องเยา 0



Color	Red	Yellow	Green
Red	1	0	0
Red	1	0	0
Yellow	0	1	0
Green	0	0	1
Yellow	0	0	1

<https://haadispeaks.wordpress.com/2018/04/09/one-hot-encoding-in-practice/>

# ข้อดีข้อเสียของ kNN

## ข้อดี

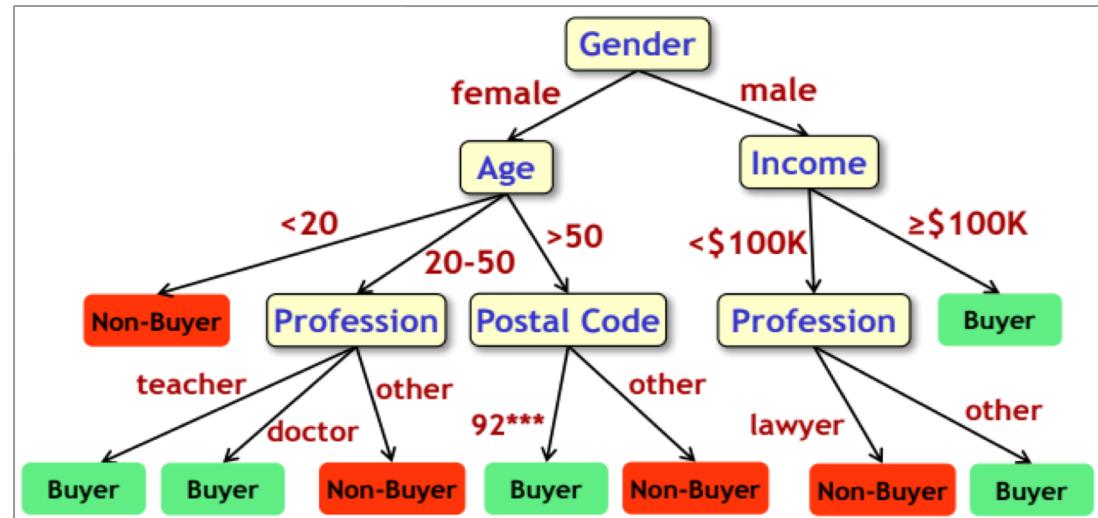
- เข้าใจง่าย
- ปรับใช้กับ Multi-class/Multi-label หรือ Regression ได้ง่าย

## ข้อเสีย

- ถ้า feature เยอะหรือ  $k$  สูง จะใช้เวลาคำนวณนาน
- ไม่เดล米ขihad ใหญ่ เพราะต้องจำข้อมูลชุด train กึ่งหมด

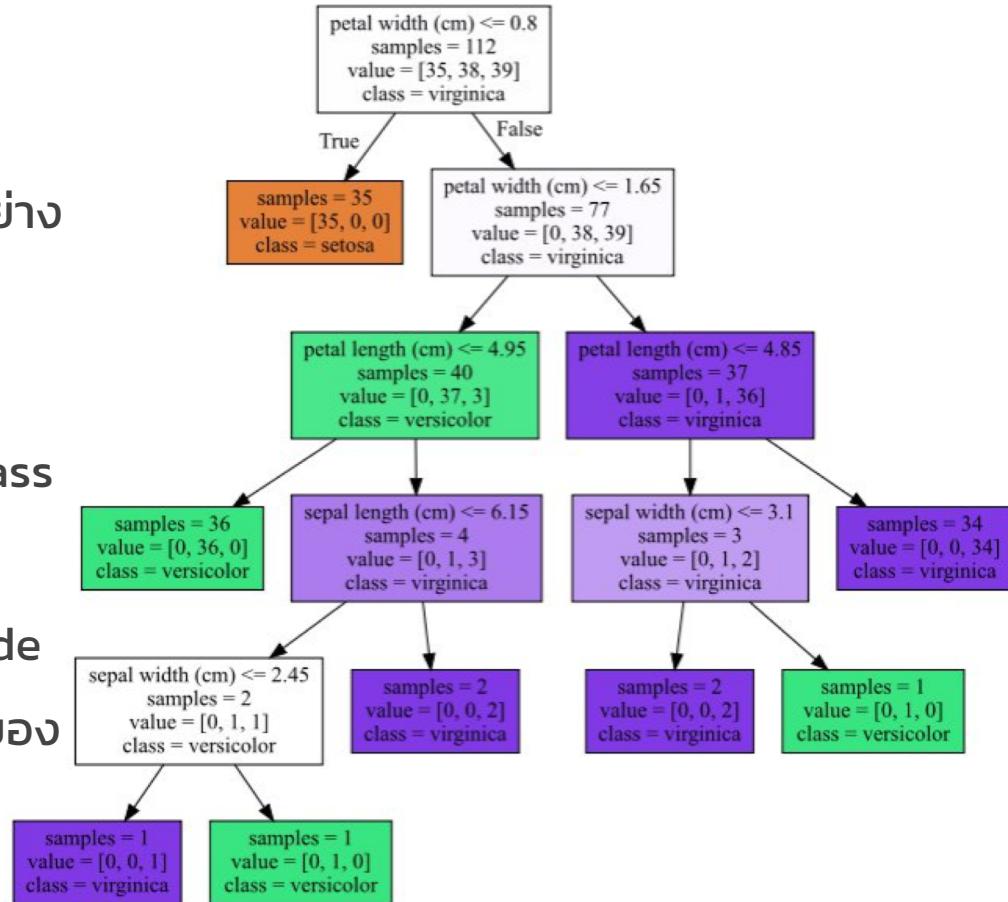
# Decision Trees

- นำข้อมูลมาสร้างต้นไม้การตัดสินใจ
- ใช้ต้นไม้การตัดสินใจจำแนกผลเฉลยของข้อมูลในอนาคต
- Node:** Features
- Edges:** Feature Values
- Leaf:** Classes / Labels
- Node ที่อยู่ด้านบน จะมีอำนาจ  
จำแนกตัวอย่างสูงกว่า (general  
กว่า) Node ด้านล่าง
- สามารถวิเคราะห์ได้ว่า Feature ใด  
มีอำนาจแยกตัวอย่างสูงแค่ไหน



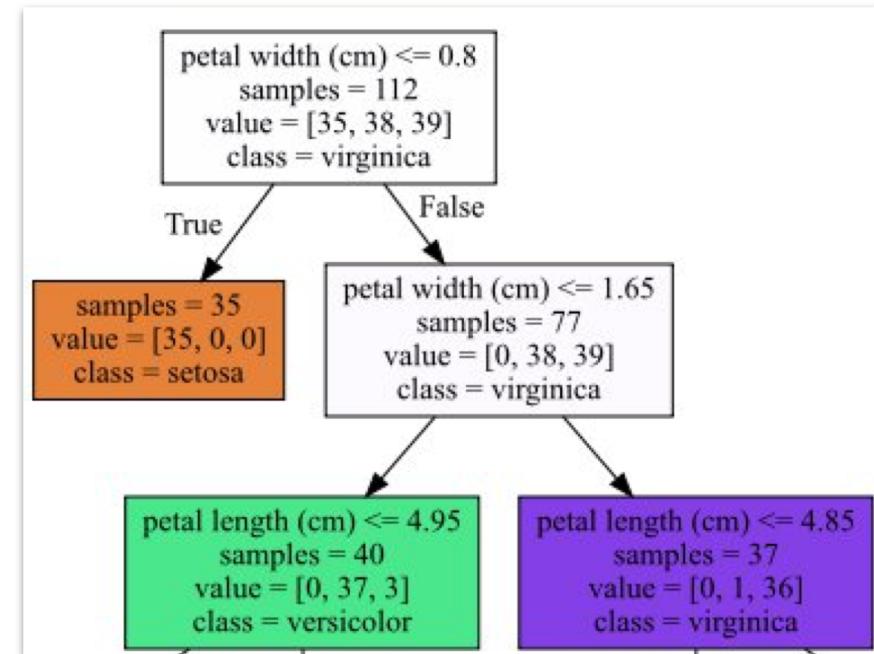
# การสร้าง Decision Tree

1. เริ่มต้นจาก decision node ที่มีทุกตัวอย่าง อยู่ร่วมกัน
2. หาก feature และค่าที่คัดแยกตัวอย่าง ออกเป็น Class ที่สมบูรณ์ที่สุด (ไม่มี Class อื่นปน)
3. ถ้า Class สมบูรณ์แล้วให้สร้าง leaf node
4. ถ้าไม่สมบูรณ์ให้สร้าง decision node ของตัวอย่างที่ยังปนกัน และกำข้อ 2 ใหม่
5. ทำจนไม่มี decision node เหลือ



# การลดการ overfit ของ Decision Tree

- Decision Tree มักจะหา feature และค่าที่จะตัดแบ่งตัวอย่างจนกลายเป็น leaf node จนหมด ซึ่งเป็นการ overfit
- ลดการ overfit ได้โดย
  - จำกัดความลึกของต้นไม้
  - จำกัดจำนวน leaf node ของต้นไม้
    - จำกัดจำนวน sample 9 ใน node , node 0 ก



# ข้อดีข้อเสียของ Decision Tree

## ข้อดี

- เรียนรู้ง่าย ตีความง่าย
- ไม่ต้องคำนึงถึง feature scaling
- ใช้ feature ที่เป็น category กับ numerical ผสมกันได้ง่าย

Model parameter  $\rightarrow$  สูตร เช่น  $y = \alpha x_1^2 + \beta x_1^1$   
 Hyper parameter  $\rightarrow$  ปัจจัยที่ควบคุมการทำงานของ Model  
 เช่น ใน KNN  
 $\text{Max-depth}$  ของ tree

## ข้อเสีย

- มักจะ overfit ถึงแม้จะพยายามลดแล้วก็ตาม
- อาจต้องใช้ต้นไม้หลายต้นเพื่อให้สามารถ generalize ได้ (random forest)

สร้างต้นไม้ 2 ขาตัวหนึ่ง error ของตัว 1  
 3 ~ 2  
 1 ~ 2  
 1 2 3 4 5 6 7 8 9  
 1 2 3 4 5 6 7 8 9

# Naïve Bayes

คำสอนนี้!  
เป็น

- ใช้หลักการทางหลักสถิติเพื่อคำนวณความน่าจะเป็นที่ตัวอย่างใหม่จะอยู่ในแต่ละ Class จากโอกาสที่ feature แต่ละตัวจะปรากฏใน Class นั้น และจึงเลือกผลลัพธ์ที่มีความน่าจะเป็นสูงที่สุด
- สมมติฐาน: feature แต่ละตัวเป็นอิสระต่อกัน (ไม่เกี่ยวข้องกัน)
- คล้ายกับการ “เหมารวม” ของคน คือใช้ “ตัวอย่างเก่า” กับ “ความน่าจะเป็นที่องค์ประกอบจะเกิดขึ้นในตัวอย่างเก่า” เพื่อคาดการตัวอย่างใหม่เมื่อเห็นองค์ประกอบเดิมเกิดขึ้นอีก
  - เช่น เราสังเกตว่าตอนฝนตกก้องฟ้าส่วนมากจะมีดครึ่มและมีฟ้าผ่าบ้าง ในอนาคตถ้าก้องฟ้ามีดครึ่มเราจะคาดการณ์ว่าโอกาสที่ฝนตกนั้นมีสูงกว่าถ้าก้องฟ้าไม่มีดแต่ฟ้าผ่าอย่างเดียว

# ประเภทของ Naive Bayes

## 1. Gaussian Naive Bayes

- ใช้ใบกรณีที่ feature เป็นค่าตัวเลข เช่น ส่วนสูง น้ำหนัก

## 2. Multinomial Naive Bayes

- ใช้ใบกรณีที่ feature เป็นการนับ เช่นจำนวนคำที่ปรากฏในเอกสาร

## 3. Bernoulli Naive Bayes

- ใช้ใบกรณีที่ feature เป็นค่า category เช่นอยู่หรือไม่อยู่ใน EU, มีหรือไม่มีคำนี้ในเอกสาร

Multinomial กับ Bernoulli Naive Bayes มักใช้กับการค้นคืนเอกสาร

# Bernoulli Naïve Bayes Example

- นำรายกลุ่มของอุณหภูมิของแต่ละประเทศโดยพิจารณาจากประเทศนั้นอยู่ใน European Union หรือไม่ และประเทศนั้นติดกับแนวฝั่งทะเลหรือไม่
- **Features**
  - Coastline: [yes, no]
  - EU: [yes, no]
- **Labels**
  - Cold
  - Cool
  - Warm
  - Hot

# Naïve Bayes Example (con't)

- คำนวณความน่าจะเป็นของแต่ละผลเฉลยจากทุกตัวอย่างในชุดข้อมูลฝึก

Cold	0.18	1.0
Cool	0.38	
Warm	0.24	
Hot	0.20	

# Naïve Bayes Example (con't)

2. หลังจากได้ค่าความน่าจะเป็นของแต่ละผลเฉลยแล้วให้คำนวณค่าความน่าจะเป็นของแต่ละคุณลักษณะและแต่ละผล

เฉลย

Cold (0.18)	coastline = yes	0.83
	coastline = no	0.17
	EU = yes	0.67
	EU = no	0.33
Warm (0.24)	coastline = yes	0.50
	coastline = no	0.50
	EU = yes	0.50
	EU = no	0.50
Cool (0.38)	coastline = yes	0.69
	coastline = no	0.31
	EU = yes	0.77
	EU = no	0.23
Hot (0.20)	coastline = yes	1.0
	coastline = no	0.0
	EU = yes	0.71
	EU = no	0.29

# Naïve Bayes Example (con't)

3. คำนวณความน่าจะเป็นของตัวอย่างโดยนำค่าคุณลักษณะมาคูณกันเพื่อให้ได้เป็นค่าความน่าจะเป็นของผลเฉลยนั้น

- New Instance

- France, coastline=yes, EU=yes

Category	Prob.	coastline = yes	EU = yes	Score
Cold	0.18	0.83	0.67	0.10
Cool	0.38	0.69	0.77	0.20
Warm	0.24	0.50	0.50	0.06
Hot	0.20	1.0	0.71	0.14

- New Instance

- Serbia, coastline=no, EU=no

Category	Prob.	coastline = no	EU = no	Score
Cold	0.18	0.17	0.33	0.01
Cool	0.38	0.31	0.23	0.03
Warm	0.24	0.50	0.5	0.06
Hot	0.20	0.0	0.29	0.00

# ข้อดีข้อเสียของ Naive Bayes

## ข้อดี

- คำนวณความน่าจะเป็นของ feature ต่างๆ ได้ง่าย ทำให้ทำงานกับจำนวน feature ที่เยอะได้
- มักใช้เป็น baseline เพื่อเปรียบเทียบกับวิธีอื่นที่ซับซ้อนกว่า

## ข้อเสีย

- สมมติฐานว่า feature แต่ละตัวไม่เกี่ยวข้องกันนั้นยากที่จะเป็นจริงกับทุกคู่ feature
- ประสิทธิภาพในการ generalize มักจะไม่ค่อยดี
- ใช้บัตริกาฟต่อสัดส่วนตัวอย่างของ Class แต่ละ Class ไม่เท่ากัน จึงควรทำให้จำนวนตัวอย่างของแต่ละ Class เท่ากันก่อน
- ถ้า probability ของ feature ใน class ได้เป็น 0 จะทำให้ไม่สามารถคำนวณได้