บทที่ 8

เรื่อง Feature Selection

Feature Selection

เป็นการคัดเลือก Feature ที่ดี เนื่องจากบางครั้งการสกัด Feature จากสิ่งที่เราสนใจ บาง Feature อาจจะไม่จำเป็น บาง Feature อาจจะดีหรือไม่ดี เราจะมีวิธีการเลือกอย่างไร เราจะเลือก Feature ที่ดีและสำคัญออกมา เพื่อนำไปทำการจำแนกได้อย่างถูกต้อง

Feature Selection

Attribute (Feature) Selection

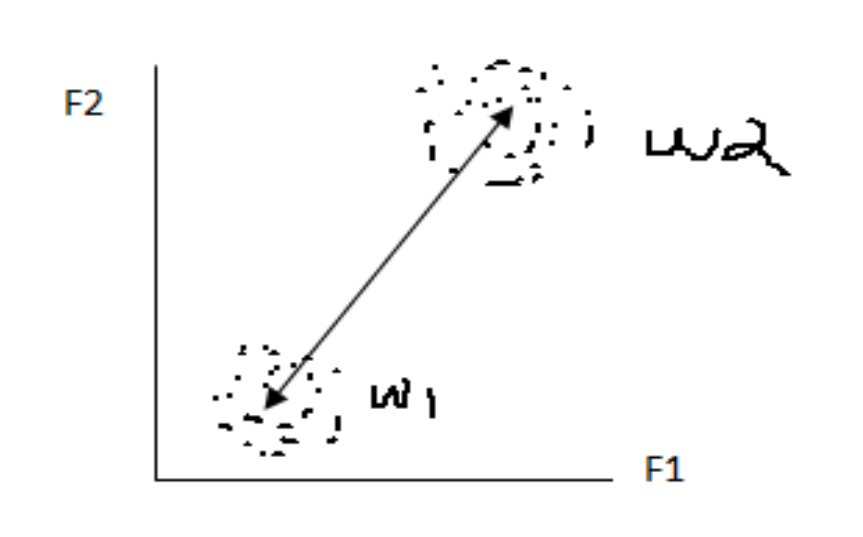
(data)^3
base | warehouse | mining

- ประสิทธิภาพของ Classification ขึ้นอยู่กับ แอตทริบิวต์ หรือ feature ที่นำมาใช้
- attribute selection เป็นวิธีการคัดเลือกแอตทริบิวต์ (หรือ feature) ที่สำคัญในการสร้างโมเดล
- การทำ attribute selection เหมาะกับ
  - ข้อมูลที่มีจำนวนแอตทริบิวต์เป็นจำนวนเยอะ เช่น text mining
  - ใช้เวลาในการสร้างโมเดลนาน

Feature ที่ดีจะต้อง

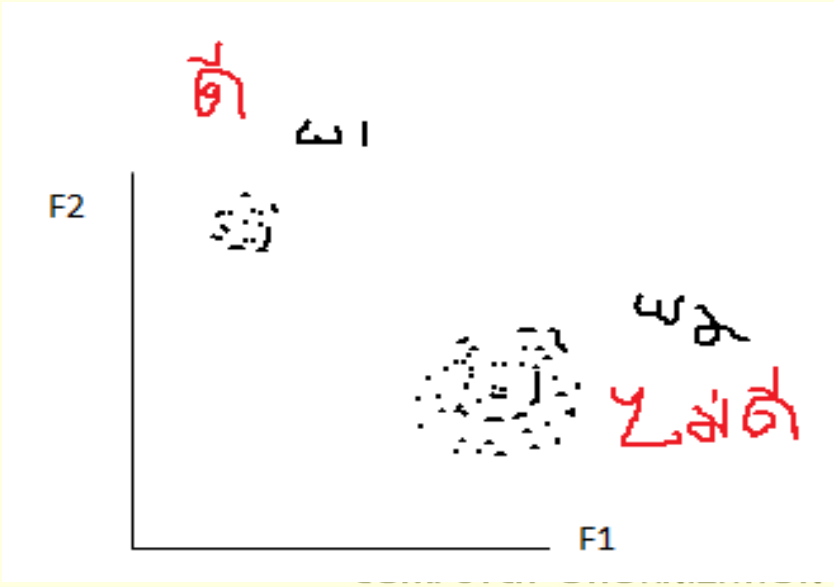
- Large between class distance

ระยะทางระหว่าง class ห่างกันมาก



- Samall within class variance

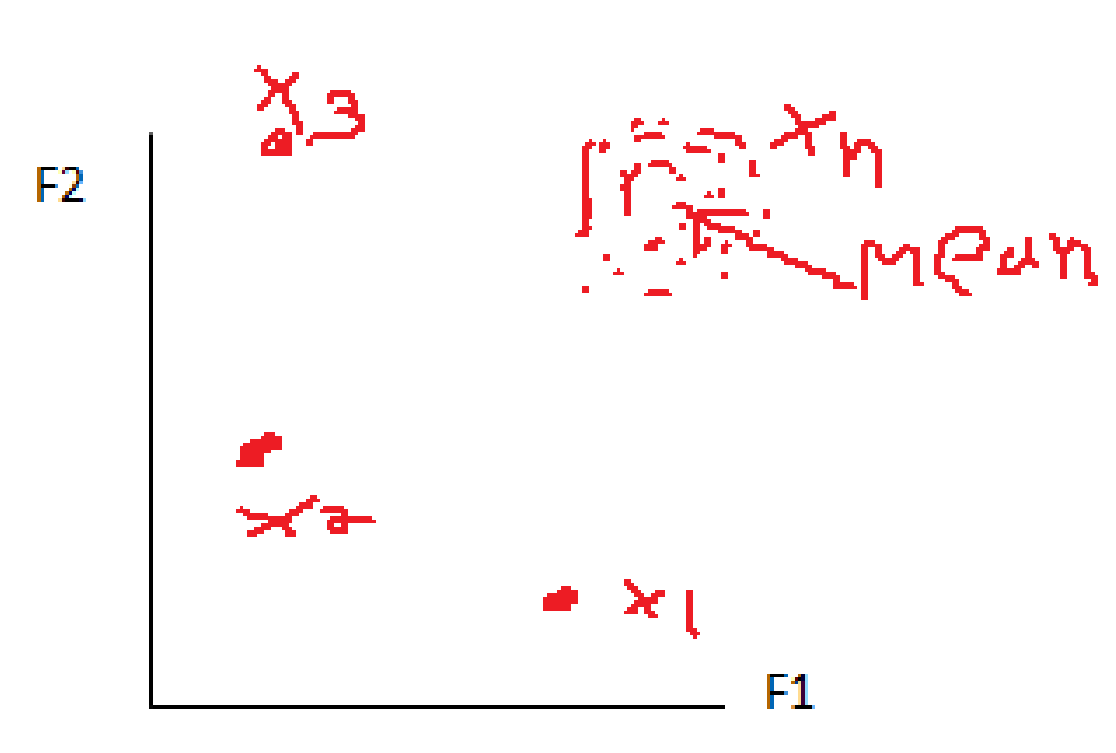
ค่าความแปรปรวนต่ำ ๆ



Preprocessing

จะทำการปรับ Feature ให้เหมาะสมก่อนที่จะนำไปใช้งาน เนื่องจาก Feature บาง Feature จะแตกต่างกันมาก ซึ่งก็จะมีวิธีการหลากหลายวิธีเช่น Outlier Removal และวิธี Data Normalization เป็นต้น

- Outlier removal จะกระทำลบ data บางตัวที่อยู่ไกลจากกลุ่ม (ไกลจากค่า Mean)



30% Robust ไม่ควรตัดข้อมูลเกิน 30% เพราะถ้าเกินมันอาจจะตัด Data ที่ดีทิ้งไป

- Data Normalization ทำการปรับ data ให้อยู่ในรูปแบบที่เหมาะสม

F1	F2	F3	Class
X	X	X	W1
X	X	X	W2
X	X	X	W3

ค่าแตกต่างกันมากเกินไป

$$\frac{\overrightarrow{xi} - Mi}{\delta i}$$

$\overrightarrow{xi}$ คือ ข้อมูลแต่ละตัว

Mi คือ ค่าเฉลี่ย (Mean)

โดย δ คือ std Standard diviation

$$\delta = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(xi - mi)^2}$$

Feature Selection

เป็นกระบวนการเลือก Feature ที่จะนำมาใช้ในระบบจำแนก (Classifier) โดยฟีเจอร์ที่เราสกัดมาอาจจะมีจำนวนหลาย Feature แต่เราจะทราบได้อย่างไรว่า Feature ไหนดี หรือไม่ดีและเหมาะสมกับงานของระบบของเรามากน้อยแค่ไหน ดังนั้นเราควรมีวิธีการวัดหา Feature เพื่อดูว่า Feature ที่เราสกัดมาดีหรือไม่ดี ดังเช่นวิธีการ

1. วัดดูความแตกต่างของ Feature ทีละ Feature แล้วดูระยะทาง

2 วัดดูความแตกต่างของ Feature ทั้งกลุ่ม แล้วดูระยะทาง

3 วัดดูความแตกต่างของ Feature โดยวิธี Wrapper Approach โดยใช้ Model ของ Classification มาใช้

Feature Selection

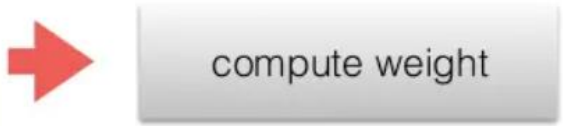
(data)^3
base | warehouse | mining

1 และ 2

Filter approach เป็นการคำนวณค่าน้ำหนัก (หรือค่าความสัมพันธ์) ของแต่ละแอตทริบิวต์และเลือกเฉพาะแอตทริบิวต์ที่สำคัญเก็บไว้

ID	Free	Won	Cash	Call	Service	Type
1	Y	Y	Y	Y	Y	spam
2	N	Y	Y	Y	N	spam

แอตทริบิวต์ทั้งหมดใน training data



Attribute Selection: Filter Approach

ID	Free	Won	Type
1	Y	Y	spam
2	N	Y	spam

แอตทริบิวต์หลังจากการเลือก (selection) แล้ว

3

- Wrapper approach เป็นการคำนวณค่าน้ำหนักโดยใช้โมเดล classification เป็นตัววัดประสิทธิภาพของแอตทริบิวต์

ID	Free	Won	Cash	Call	Service	Type
1	Y	Y	Y	Y	Y	spam
2	N	Y	Y	Y	N	spam

แอตทริบิวต์ทั้งหมดใน training data

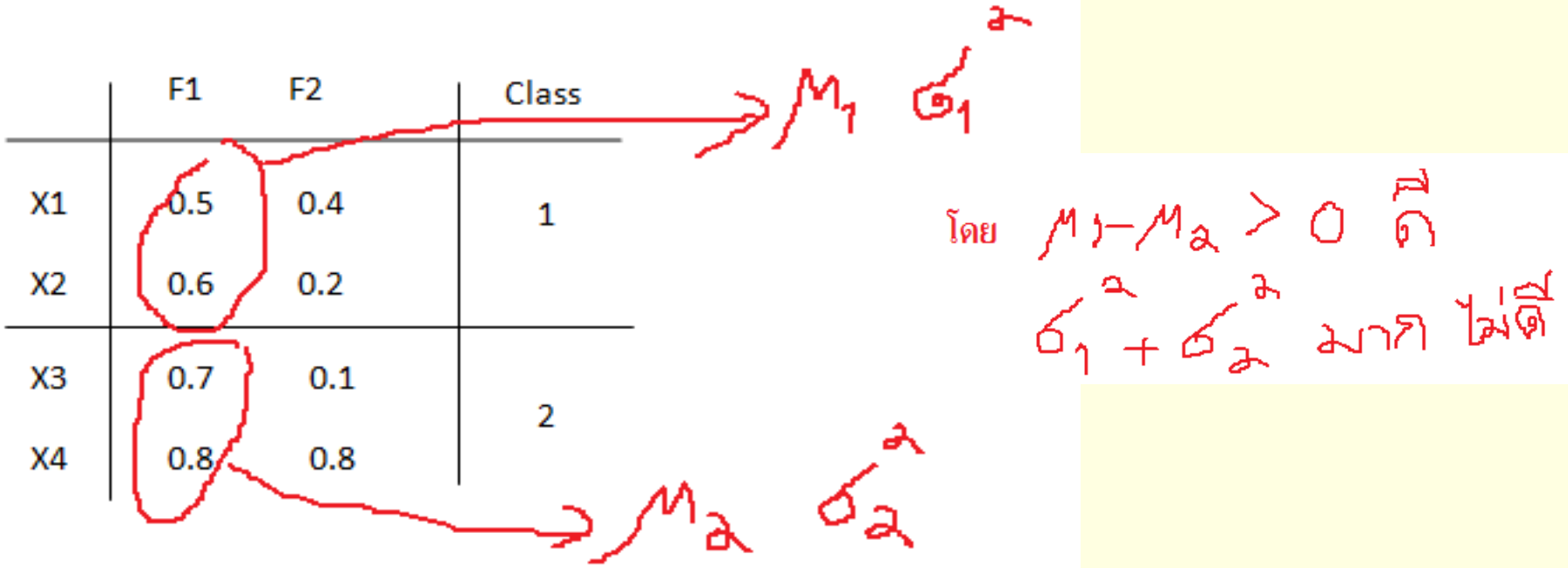
Attribute Selection: Wrapper Approach

ID	Free	Won	Type
1	Y	Y	spam
2	N	Y	spam

แอตทริบิวต์หลังจากการเลือก (selection) แล้ว

วิธีที่ 1. วัดดูความแตกต่างของ Feature ทีละ Feature แล้วดูระยะทาง

	F1	F2	Class
X1	0.5	0.4	1
X2	0.6	0.2	
X3	0.7	0.1	2
X4	0.8	0.8	

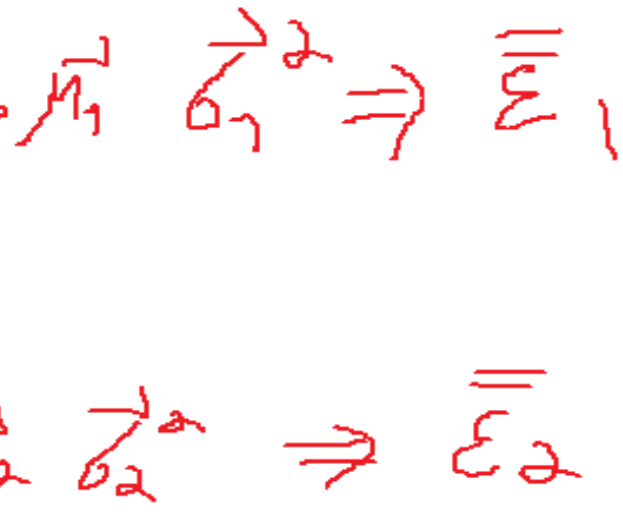


จะดู Feature แต่ละตัว แล้ววัดระยะจากนั้นเรียงลำดับจากมากไปน้อย แล้วค่อยเลือกค่าที่มาก ซึ่งจะทำให้ระบบ AI ของเราสามารถมีประสิทธิภาพสูงขึ้น

วิธีที่ 2. วัดดูความแตกต่างของ Feature ทั้งกลุ่ม แล้วดูระยะทาง

เนื่องจากในบางครั้ง Feature มีความสัมพันธ์กัน

	F1	F2	Class
X1	0.5	0.4	1
X2	0.6	0.2	
X3	0.7	0.1	2
X4	0.8	0.8	



นำค่า μ, σ ไปคำนวณหาค่า $\overline{\overline{\varepsilon}}$ (corvarience) โดยขนาดของ $\overline{\overline{\varepsilon}}$ ขึ้นอยู่กับจำนวน Feature จากนั้นหาระยะทาง d_{ij}

$$dij > 0$$

วิธีการนี้จะเลือก Feature บางตัวแล้วหา d_{ij} ดังเช่นวิธีการ Back ward และ Forward ก็ได้

Class separability measure เป็นการวัดความสัมพันธ์ระหว่าง Class ว่าดีหรือไม่ดี ซึ่งสามารถบอกได้ว่า Feature ไหนดี หรือไม่ดี

- Divergence ความแตกต่าง

$$d_{12} = D_{12} + D_{21}$$

ถ้า d12 ยิ่งมาก แสดงว่า Feature ที่ได้มานั้นดี

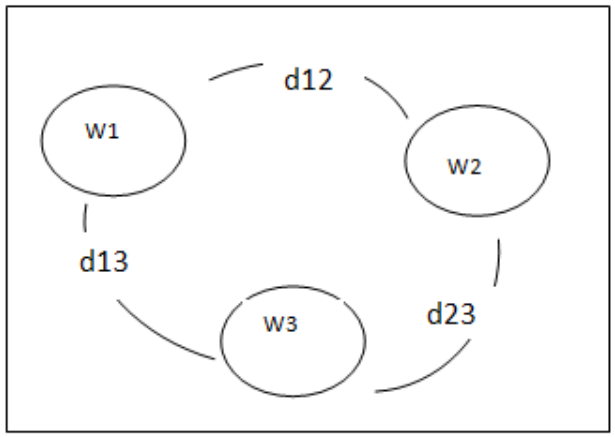
$$D12 = \int_{-\infty}^{\infty} f(\vec{x}/w1) \ln \frac{f(\vec{x}/w1)}{f(\vec{x}/w2)} \, d\vec{x}$$

$$D21 = \int_{-\infty}^{\infty} f(\vec{x}/w2) \ln \frac{f(\vec{x}/w2)}{f(\vec{x}/w1)} \, d\vec{x}$$

ในกรณี > 2 Class ให้จับคู่ Wi และ Wj ก่อน

$$d_{ij} = D_{ij} + D_{ji}$$

$$= \int_{-\infty}^{\infty} f(\vec{x}/wi) - f(\vec{x}/wj) \ln \frac{f(\vec{x}/wi)}{f(\vec{x}/wj)} \, d\vec{x}$$



$$d = (d12 + d13 + d23)/3$$

เมื่อต้องการวัดว่า Feature ดีไหมต้องหาค่าของ Average divergence จากสมการจะเป็นการเอาค่าที่จับคู่ทุกตัวมาทำการหาค่าเฉลี่ย

Average divergence ค่ายิ่งมากยิ่งดี

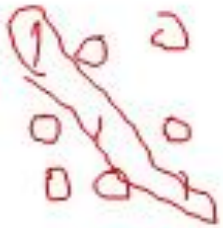
$$D = \sum_{i=1}^{M} \sum_{j=1}^{M} P(wi)P(wj)dij$$

- **<u>กรณีหลาย Dimension</u>** เราสามารถหา dij ความแตกต่างระหว่าง Class แบบกลุ่มจากสมการ

สมมุติเราให้ $N(\bar{\mu}_i, \bar{\bar{\Sigma}}_i), N(\bar{\mu}_j, \bar{\bar{\Sigma}}_j)$ ดังนั้นในกรณีหลาย Dimension คือ

$$d_{ij} = \frac{1}{2} trace\left\{\bar{\bar{\Sigma}}_i^{-1}\bar{\bar{\Sigma}}_j + \bar{\bar{\Sigma}}_j^{-1}\bar{\bar{\Sigma}}_i - 2\bar{\bar{I}}\right\} + \frac{1}{2}(\bar{\mu}_i - \bar{\mu}_j)^T\left(\bar{\bar{\Sigma}}_i^{-1} + \bar{\bar{\Sigma}}_j^{-1}\right)(\bar{\mu}_i - \bar{\mu}_j)$$

Trace คือ sum ของ Eigen value ของ Matrix (ผลบวกของแนวทแยง)

ส่วน $\bar{\bar{I}}$ คือ Matrix เอกลักษณ์ ⇒ 

กรณี ที่เป็น 1 Dimension

$$d_{ij} = \frac{1}{2}\left(\frac{\sigma_j^2}{\sigma_i^2} + \frac{\sigma_i^2}{\sigma_j^2} - 2\right) + \frac{1}{2}(\mu_i - \mu_j)^2\left(\frac{1}{\sigma_i^2} + \frac{1}{\sigma_j^2}\right)$$

นำไปวัดทีละ Feature แล้วทำการ Ranking

Bhattacharyya distance (B)

เป็นวิธีการหา Distance อีกวิธีการหนึ่ง คล้ายกันกับ divergence คือจะเป็น การหา distance ระหว่าง Class เหมือนกัน วิธีการนี้จะให้ประสิทธิภาพ แต่จะใช้เวลานาน โดยคำนวณได้จากสมการ

$$B = \frac{1}{8}(\vec{\mu}_i - \vec{\mu}_j)^T \left(\frac{\overline{\overline{\Sigma}}_i + \overline{\overline{\Sigma}}_j}{2}\right)^{-1} (\vec{\mu}_i - \vec{\mu}_j) + \frac{1}{2}\ln \frac{|(\overline{\overline{\Sigma}}_i + \overline{\overline{\Sigma}}_j)/2|}{\sqrt{|\overline{\overline{\Sigma}}_i||\overline{\overline{\Sigma}}_j|}}$$

```
Bratta_distant(Number_Del,n) = (1/8) * ((m1 - m2)' * inv((Cov1 + Cov2)/2) * (m1 - m2)
+ ((1/2) * log(det((Cov1 + Cov2)/2) / sqrt(detCov1 * detCov2))));
```

```
Divergence(Number_Del,n) = ((1/2) * trace((inv(Cov1) * Cov2) + (inv(Cov2) * Cov1)
- (2 * eye(30-Number_Del,30-Number_Del)))) + ((1/2) * ((m1-m2)' * (inv(Cov1) +
inv(Cov2)) * (m1-m2)));
```

วิธี Sequential Backward

(1) เลือก C คำนวณสำหรับ $\begin{bmatrix} F1 \\ F2 \\ F3 \\ F4 \end{bmatrix}$

(2) ลบไป 1 Feature โดยลบทีละ 1 Feature (เลือกเอาออก 1 Feature จาก Feature ทั้งหมด)

$$\begin{bmatrix} F1 \\ F2 \\ F3 \end{bmatrix}, \begin{bmatrix} F1 \\ F2 \\ F4 \end{bmatrix}, \begin{bmatrix} F1 \\ F3 \\ F4 \end{bmatrix}, \begin{bmatrix} F2 \\ F3 \\ F4 \end{bmatrix}$$

คำนวณหา distance ของทุกกลุ่ม แล้วเลือกกลุ่มที่ดีที่สุดโดยใช้ Divergence หรือ Brattachargga

F1	F2	F3	Class
0.1	0.2	0.2	1
0.3	0.4	0.5	1
1.8	2.1	2.2	2
2.3	2.4	2.5	2

สมมุติกลุ่มที่คำนวณได้ดีสุด คือ $\begin{bmatrix} F1 \\ F2 \\ F3 \end{bmatrix}$ ก็จะทำขั้นตอนต่อไป

(3) ลบอีก 1 Feature แล้วจะได้

$$\begin{bmatrix} F1 \\ F2 \end{bmatrix}, \begin{bmatrix} F1 \\ F3 \end{bmatrix}, \begin{bmatrix} F2 \\ F3 \end{bmatrix}$$

จากนั้นคำนวณหา distance แต่ละกลุ่ม แล้วเลือกกลุ่มที่ดีสุด

(4) ทำไปจนกว่าจะได้ผลลัพท์ที่คาดว่าดีที่สุดจากการทดสอบซึ่งอาจจะเหลือไม่กี่ Feature ถ้ามันทดสอบกับการทดลองจริงแล้วได้ดีที่สุด

วิธี Sequential Forward

เป็นวิธีการเลือก Feature แบบหนึ่งที่มีลักษณะการทำงานตรงข้ามกับแบบ Backward โดยมีขั้นตอนดังนี้

1. หา C ของแต่ละ Feature แล้วเลือก Feature ที่ดีที่สุด สมมุติ เป็น F1 (อาจจะสุ่มเอาหรือใช้วิธีการวัดหาค่าที่ดีที่สุด)

$$[F_1]$$

2. จากนั้นเอา F1 ไปจับคู่ที่เหลือ แล้วหา Distance ของทั้งหมด แล้วเลือกชุดที่ดีที่สุด

$$\begin{bmatrix} F_1 \\ F_2 \end{bmatrix}, \begin{bmatrix} F_1 \\ F_3 \end{bmatrix}, \begin{bmatrix} F_1 \\ F_4 \end{bmatrix}$$

สมมุติ

$$\begin{bmatrix} F_1 \\ F_3 \end{bmatrix}$$ ดีที่สุด

3. จับคู่ที่เหลือ จนกว่าจะครบที่กำหนดหรือได้ผลลัพธ์ที่ดีที่สุด

$$\begin{bmatrix} F_1 \\ F_3 \\ F_2 \end{bmatrix}, \begin{bmatrix} F_1 \\ F_3 \\ F_4 \end{bmatrix}$$

คำนวณค่า Distance ทุกกลุ่ม และให้ทำไปจนกว่าจะได้ Feature ที่ต้องการที่ดีที่สุด

วิธีที่ 3. วัดดูความแตกต่างของ Feature โดยวิธี Wrapper Approach โดยใช้ Model ของ Classification มาใช้

3

- Wrapper approach เป็นการคำนวณค่าน้ำหนักโดยใช้โมเดล classification เป็นตัววัดประสิทธิภาพของแอตทริบิวต์

ID	Free	Won	Cash	Call	Service	Type
1	Y	Y	Y	Y	Y	spam
2	N	Y	Y	Y	N	spam

แอตทริบิวต์ทั้งหมดใน training data

Attribute Selection: Wrapper Approach

ID	Free	Won	Type
1	Y	Y	spam
2	N	Y	spam

แอตทริบิวต์หลังจากการเลือก (selection) แล้ว

Wrapper Approach

(data)^3
base | warehouse | mining

- เป็นวิธีการเลือกแอตทริบิวต์ใส่เข้าไปหรือถอดออกมาเพื่อสร้างโมเดล และเลือก set ของแอตทริบิวต์ที่ดีไว้ใช้
  - ใช้แอตทริบิวต์ Free อย่างเดียว

ID	Free	Won	Cash	Type
1	Y	Y	Y	spam
2	N	Y	Y	spam
3	N	N	N	normal
4	N	N	N	normal
5	Y	N	N	spam
6	Y	N	N	spam
7	N	N	N	normal
8	N	Y	N	spam
9	N	N	N	normal
10	N	N	N	normal

ID	Free	Type
1	Y	spam
2	N	spam
3	N	normal
4	N	normal
5	Y	spam
6	Y	spam
7	N	normal
8	N	spam
9	N	normal
10	N	normal

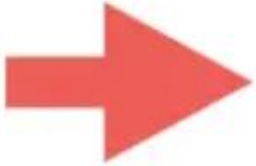
- ใช้แอตทริบิวต์ Won อย่างเดียว

ID	Free	Won	Cash	Type
1	Y	Y	Y	spam
2	N	Y	Y	spam
3	N	N	N	normal
4	N	N	N	normal
5	Y	N	N	spam
6	Y	N	N	spam
7	N	N	N	normal
8	N	Y	N	spam
9	N	N	N	normal
10	N	N	N	normal

ID	Won	Type
1	Y	spam
2	Y	spam
3	N	normal
4	N	normal
5	N	spam
6	N	spam
7	N	normal
8	Y	spam
9	N	normal
10	N	normal

- ใช้แอตทริบิวต์ Cash อย่างเดียว

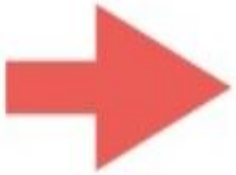
ID	Free	Won	Cash	Type
1	Y	Y	Y	spam
2	N	Y	Y	spam
3	N	N	N	normal
4	N	N	N	normal
5	Y	N	N	spam
6	Y	N	N	spam
7	N	N	N	normal
8	N	Y	N	spam
9	N	N	N	normal
10	N	N	N	normal



ID	Cash	Type
1	Y	spam
2	Y	spam
3	N	normal
4	N	normal
5	N	spam
6	N	spam
7	N	normal
8	N	spam
9	N	normal
10	N	normal

- ใช้แอตทริบิวต์ Free และ Won

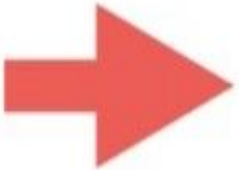
ID	Free	Won	Cash	Type
1	Y	Y	Y	spam
2	N	Y	Y	spam
3	N	N	N	normal
4	N	N	N	normal
5	Y	N	N	spam
6	Y	N	N	spam
7	N	N	N	normal
8	N	Y	N	spam
9	N	N	N	normal
10	N	N	N	normal



ID	Free	Won	Type
1	Y	Y	spam
2	N	Y	spam
3	N	N	normal
4	N	N	normal
5	Y	N	spam
6	Y	N	spam
7	N	N	normal
8	N	Y	spam
9	N	N	normal
10	N	N	normal

- ใช้แอตทริบิวต์ Free และ Cash

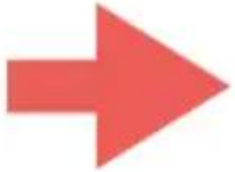
ID	Free	Won	Cash	Type
1	Y	Y	Y	spam
2	N	Y	Y	spam
3	N	N	N	normal
4	N	N	N	normal
5	Y	N	N	spam
6	Y	N	N	spam
7	N	N	N	normal
8	N	Y	N	spam
9	N	N	N	normal
10	N	N	N	normal



ID	Free	Cash	Type
1	Y	Y	spam
2	N	Y	spam
3	N	N	normal
4	N	N	normal
5	Y	N	spam
6	Y	N	spam
7	N	N	normal
8	N	N	spam
9	N	N	normal
10	N	N	normal

- ใช้แอตทริบิวต์ Won และ Cash

ID	Free	Won	Cash	Type
1	Y	Y	Y	spam
2	N	Y	Y	spam
3	N	N	N	normal
4	N	N	N	normal
5	Y	N	N	spam
6	Y	N	N	spam
7	N	N	N	normal
8	N	Y	N	spam
9	N	N	N	normal
10	N	N	N	normal



ID	Won	Cash	Type
1	Y	Y	spam
2	Y	Y	spam
3	N	N	normal
4	N	N	normal
5	N	N	spam
6	N	N	spam
7	N	N	normal
8	Y	N	spam
9	N	N	normal
10	N	N	normal

- ใช้แอตทริบิวต์ Free, Won และ Cash

ID	Free	Won	Cash	Type
1	Y	Y	Y	spam
2	N	Y	Y	spam
3	N	N	N	normal
4	N	N	N	normal
5	Y	N	N	spam
6	Y	N	N	spam
7	N	N	N	normal
8	N	Y	N	spam
9	N	N	N	normal
10	N	N	N	normal

Forward Selection

- เพิ่มแอตทริบิวต์ทีละ 1 แอตทริบิวต์และคัดเลือกเฉพาะแอตทริบิวต์ที่มีความสำคัญเก็บไว้
  - ถ้าแอตทริบิวต์ที่ใส่เพิ่มเข้าไปให้ค่า performance ดีขึ้นก็จะเก็บแอตทริบิวต์นี้ไว้
  - ถ้าแอตทริบิวต์ที่ใส่เพิ่มเข้าไปให้ค่า performance แย่ลงก็จะดึงแอตทริบิวต์นี้ออกมา

Backward Elimination

- เริ่มจากใช้แอตทริบิวต์ทั้งหมดและตัดแอตทริบิวต์ออกไปทีละ 1 ตัว เพื่อคัดเลือกเฉพาะแอตทริบิวต์ที่มีความสำคัญเก็บไว้
  - ถ้าแอตทริบิวต์ที่ตัดออกไปให้ค่า performance ดีขึ้นก็จะตัดแอตทริบิวต์นี้ทิ้ง
  - ถ้าแอตทริบิวต์ที่ตัดออกไปให้ค่า performance แย่ลงก็จะเก็บแอตทริบิวต์นี้ไว้

Forward Selection

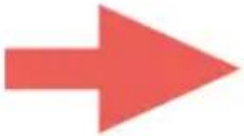
(data)3
base | warehouse | mining

- เพิ่มแอตทริบิวต์ทีละ 1 แอตทริบิวต์และคัดเลือกเฉพาะแอตทริบิวต์ที่มีความสำคัญเก็บไว้
  - ถ้าแอตทริบิวต์ที่ใส่เพิ่มเข้าไปให้ค่า performance ดีขึ้นก็จะเก็บแอตทริบิวต์นี้ไว้
  - ถ้าแอตทริบิวต์ที่ใส่เพิ่มเข้าไปให้ค่า performance แย่ลงก็จะดึงแอตทริบิวต์นี้ออกมา

Forward Selection

- ใช้แอตทริบิวต์ Free อย่างเดียว

ID	Free	Type
1	Y	spam
2	N	spam
3	N	normal
4	N	normal
5	Y	spam
6	Y	spam
7	N	normal
8	N	spam
9	N	normal
10	N	normal



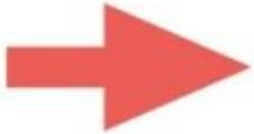
ทดสอบประสิทธิภาพ
ด้วย Cross-validation

accuracy = 80%

Forward Selection

- ใช้แอตทริบิวต์ Won อย่างเดียว

ID	Won	Type
1	Y	spam
2	Y	spam
3	N	normal
4	N	normal
5	N	spam
6	N	spam
7	N	normal
8	Y	spam
9	N	normal
10	N	normal



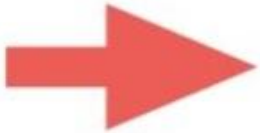
ทดสอบประสิทธิภาพ
ด้วย Cross-validation

accuracy = 80%

Forward Selection

- ใช้แอตทริบิวต์ Cash อย่างเดียว

ID	Cash	Type
1	Y	spam
2	Y	spam
3	N	normal
4	N	normal
5	N	spam
6	N	spam
7	N	normal
8	N	spam
9	N	normal
10	N	normal



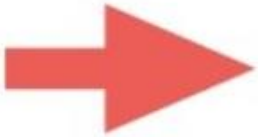
ทดสอบประสิทธิภาพ
ด้วย Cross-validation

accuracy = 50%

Forward Selection

- ใช้แอตทริบิวต์ Free และ Won

ID	Free	Won	Type
1	Y	Y	spam
2	N	Y	spam
3	N	N	normal
4	N	N	normal
5	Y	N	spam
6	Y	N	spam
7	N	N	normal
8	N	Y	spam
9	N	N	normal
10	N	N	normal



ทดสอบประสิทธิภาพ
ด้วย Cross-validation

accuracy = 60%

Forward Selection

- ใช้แอตทริบิวต์ Free และ Won

ID	Free	Won	Type
1	Y		spam
2	N		spam
3	N		normal
4	N		normal
5	Y		spam
6	Y		spam
7	N		normal
8	N		spam
9	N		normal
10	N		normal

ทดสอบประสิทธิภาพ
ด้วย Cross-validation

accuracy = 60%

ตัดแอตทริบิวต์ Cash ทิ้งเนื่องจากให้ค่าความถูกต้องลดลง

Forward Selection

- ใช้แอตทริบิวต์ Free และ Cash

ID	Free	Cash	Type
1	Y	Y	spam
2	N	Y	spam
3	N	N	normal
4	N	N	normal
5	Y	N	spam
6	Y	N	spam
7	N	N	normal
8	N	N	spam
9	N	N	normal
10	N	N	normal

ทดสอบประสิทธิภาพ
ด้วย Cross-validation

accuracy = 80%

Forward Selection

- ใช้แอตทริบิวต์ Free และ Cash

ID	Free	Cash	Type
1	Y		spam
2	N		spam
3	N		normal
4	N		normal
5	Y		spam
6	Y		spam
7	N		normal
8	N		spam
9	N		normal
10	N		normal

ทดสอบประสิทธิภาพ
ด้วย Cross-validation

accuracy = 80%

ตัดแอตทริบิวต์ Cash ทิ้งเนื่องจากไม่ได้ทำให้ค่าความถูกต้องเพิ่มขึ้น

Backward Elimination

- เริ่มจากใช้แอตทริบิวต์ทั้งหมดและตัดแอตทริบิวต์ออกไปทีละ 1 ตัว เพื่อคัดเลือกเฉพาะแอตทริบิวต์ที่มีความสำคัญเก็บไว้
  - ถ้าแอตทริบิวต์ที่ตัดออกไปให้ค่า performance ดีขึ้นก็จะตัดแอตทริบิวต์นี้ทิ้ง
  - ถ้าแอตทริบิวต์ที่ตัดออกไปให้ค่า performance แย่ลงก็จะเก็บแอตทริบิวต์นี้ไว้

Backward Elimination

- ใช้แอตทริบิวต์ Free, Won และ Cash

ID	Free	Won	Cash	Type
1	Y	Y	Y	spam
2	N	Y	Y	spam
3	N	N	N	normal
4	N	N	N	normal
5	Y	N	N	spam
6	Y	N	N	spam
7	N	N	N	normal
8	N	Y	N	spam
9	N	N	N	normal
10	N	N	N	normal

ทดสอบประสิทธิภาพ
ด้วย Cross-validation

accuracy = 60%

Backward Elimination

- ใช้แอตทริบิวต์ Won และ Cash (ตัดแอตทริบิวต์ Free ทิ้ง)

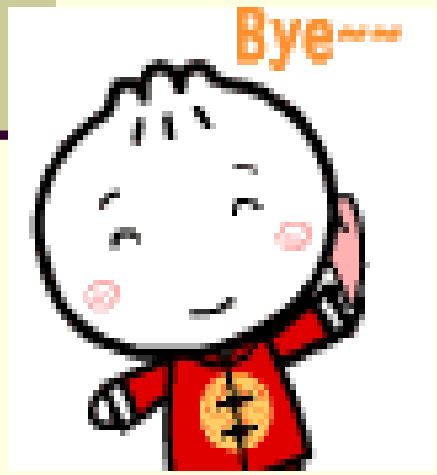
ID	Won	Cash	Type
1	Y	Y	spam
2	Y	Y	spam
3	N	N	normal
4	N	N	normal
5	N	N	spam
6	N	N	spam
7	N	N	normal
8	Y	N	spam
9	N	N	normal
10	N	N	normal

ทดสอบประสิทธิภาพ
ด้วย Cross-validation

accuracy = 80%

ตัดแอตทริบิวต์ Free ทิ้งเนื่องจากทำให้ค่าความถูกต้องเพิ่มขึ้น

THE END



Work

Lab6: จงเขียนโปรแกรมในการคำนวณหาค่า Distance ของข้อมูลในแต่ละ Feature ส่วน Data set ให้หาเพื่อทำการทดสอบเอง เช่น 4 Feature จาก Two classdata ที่ทำใน Assignment 1 การคำนวณ ใช้สมการตัวนี้ แล้วดูว่า Feature ไหนดีที่สุด

$$d_{ij} = \frac{1}{2}\left(\frac{\sigma_j^2}{\sigma_i^2} + \frac{\sigma_i^2}{\sigma_j^2} - 2\right) + \frac{1}{2}(\mu_i - \mu_j)^2\left(\frac{1}{\sigma_i^2} + \frac{1}{\sigma_j^2}\right)$$

Matlab

% 1. โหลดข้อมูล Iris

load fisheriris

X = meas; % ข้อมูล

% 2. แบ่งข้อมูลออกเป็นคลาส Setosa และ Versicolor

setosa_data = X(1:50, :); % ข้อมูลคลาส Setosa

versicolor_data = X(51:100, :); % ข้อมูลคลาส Versicolor

% 3. คำนวณค่าเฉลี่ยและความแปรปรวนของแต่ละ feature สำหรับแต่ละคลาส

mean_setosa = mean(setosa_data); % ค่าเฉลี่ยของคลาส Setosa

mean_versicolor = mean(versicolor_data); % ค่าเฉลี่ยของคลาส Versicolor

variance_setosa = var(setosa_data); % ความแปรปรวนของคลาส Setosa

variance_versicolor = var(versicolor_data); % ความแปรปรวนของคลาส Versicolor

```
% 4. คำนวณ Divergence ระหว่าง feature โดยใช้ค่าความแตกต่างของค่าความเฉลี่ยและความแปรปรวน

n = size(X, 2); % จำนวน feature

divergence_features = zeros(1, n); % เวกเตอร์สำหรับเก็บค่า Divergence ระหว่าง feature

for i = 1:n

    % คำนวณ Divergence ระหว่าง feature i (ให้เปลี่ยนเหมือนโจทย์)

    divergence_features(i) = abs(mean_setosa(i) - mean_versicolor(i)) / sqrt(variance_setosa(i) + variance_versicolor(i));

end
```

```
% 5. แสดงผลลัพธ์

disp('Divergence ระหว่าง feature (attribute):');

disp(divergence_features);
```

python

```
import numpy as np

import pandas as pd

# 1. โหลดข้อมูล Iris

from sklearn.datasets import load_iris

data = load_iris()

X = data.data  # ข้อมูล

# 2. แบ่งข้อมูลออกเป็นคลาส Setosa และ Versicolor

setosa_data = X[:50, :]  # ข้อมูลคลาส Setosa

versicolor_data = X[50:100, :]  # ข้อมูลคลาส Versicolor
```

```
# 3. คำนวณค่าเฉลี่ยและความแปรปรวนของแต่ละ feature สำหรับแต่ละคลาส
mean_setosa = np.mean(setosa_data, axis=0)  # ค่าเฉลี่ยของคลาส Setosa
mean_versicolor = np.mean(versicolor_data, axis=0)  # ค่าเฉลี่ยของคลาส Versicolor
variance_setosa = np.var(setosa_data, axis=0)  # ความแปรปรวนของคลาส Setosa
variance_versicolor = np.var(versicolor_data, axis=0)  # ความแปรปรวนของคลาส Versicolor

# 4. คำนวณ Divergence ระหว่าง feature โดยใช้ค่าความแตกต่างของค่าความเฉลี่ยและความแปรปรวน
n = X.shape[1]  # จำนวน feature
divergence_features = np.abs(mean_setosa - mean_versicolor) / np.sqrt(variance_setosa + variance_versicolor)

# 5. แสดงผลลัพธ์
df = pd.DataFrame({'Feature': data.feature_names, 'Divergence': divergence_features})
df = df.sort_values(by='Divergence', ascending=False)
print(df)
```