วิชาเหมืองข้อมูล (Data Mining)

บทที่ 3 การเตรียมข้อมูลสำหรับการทำเหมืองข้อมูล

2/2568

Asst. Prof. Dr. Nattapong Songneam

**Department of Computer Science,
Faculty of Science and Technology,
Phranakhon Rajabhat University**

http://www.siam2dev.com

บทที่ 3 การเตรียมข้อมูลสำหรับการทำเหมืองข้อมูล

Agenda

ทบทวน

- CRISP-DM 6 ขั้นตอน
- ข้อมูล
- การเตรียมข้อมูล

บทที่ 3 การเตรียมข้อมูลสำหรับการทำเหมืองข้อมูล

Agenda

3.1 ความหมายและความสำคัญของเตรียมข้อมูลสำหรับการทำเหมืองข้อมูล

- 3.1.1 ความหมายของข้อมูล
- 3.1.2 ประเภทของข้อมูล
- 3.1.3 แหล่งที่มาของข้อมูลสำหรับเหมืองข้อมูล
- 3.1.4 ลักษณะของข้อมูลที่เหมาะสมสำหรับเหมืองข้อมูล
- 3.1.5 การเตรียมข้อมูลสำหรับเหมืองข้อมูล
- 3.1.6 ตัวอย่างของข้อมูล (Data)

บทที่ 3 การเตรียมข้อมูลสำหรับการทำเหมืองข้อมูล

Agenda

3.2 การนำเข้าข้อมูล (Data Import)

- 3.2.1 แนะนำวิธีการนำเข้าข้อมูลจากแหล่งต่างๆ ด้วย Python
- 3.2.2 การนำเข้าจากไฟล์ CSV
- 3.2.3 การนำเข้าจากไฟล์ Excel
- 3.2.4 การเชื่อมต่อและนำเข้าข้อมูลจากฐานข้อมูล SQL
- 3.2.5 การอ่านข้อมูลจาก APIs หรือ Web Scraping (ถ้าจำเป็น)

บทที่ 3 การเตรียมข้อมูลสำหรับการทำเหมืองข้อมูล

Agenda

3.3 การสำรวจข้อมูล (Data Exploration)

3.3.1 การตรวจสอบข้อมูลเบื้องต้น

- การดูตัวอย่างข้อมูล (head)
- การตรวจสอบขนาดของข้อมูล (shape)
- การดูสถิติพื้นฐาน (describe)

3.3.2 การตรวจสอบค่าที่ขาดหาย (missing values)

3.3.3 การตรวจสอบข้อมูลซ้ำซ้อน (duplicate data)

บทที่ 3 การเตรียมข้อมูลสำหรับการทำเหมืองข้อมูล

Agenda

3.4 การทำความสะอาดข้อมูล (Data Cleaning)

3.4.1 การจัดการค่าที่ขาดหาย (Missing Data):

- การเติมค่าที่ขาดหายด้วยค่าเฉลี่ย, มัธยฐาน, หรือค่าที่เหมาะสม
- การลบแถวหรือคอลัมน์ที่มีข้อมูลหายไปมากเกินไป

3.4.2 การจัดการค่าผิดปกติ (Outliers):

- การค้นหาและกำจัดค่าผิดปกติ
- การใช้เทคนิค IQR (Interquartile Range) ในการตรวจจับ outliers

บทที่ 3 การเตรียมข้อมูลสำหรับการทำเหมืองข้อมูล

Agenda

3.5 การแปลงข้อมูล (Data Transformation)

3.5.1 การแปลงประเภทข้อมูล:

- การแปลงข้อมูลตัวเลขเป็นวันที่ (datetime)
- การแปลงคอลัมน์ที่เป็นข้อความเป็นตัวเลข (categorical to numerical)

3.5.2 การสร้างฟีเจอร์ใหม่ (Feature Engineering):

- การรวมฟีเจอร์หรือการคำนวณฟีเจอร์ใหม่

3.5.3 การปรับขนาดข้อมูล (Scaling):

- การปกติค่า (Normalization)
- การปรับสเกล (Standardization)

บทที่ 3 การเตรียมข้อมูลสำหรับการทำเหมืองข้อมูล

Agenda

3.6 การเลือกฟีเจอร์ (Feature Selection)

3.6.1 การเลือกฟีเจอร์ตามความสำคัญ (Feature Importance):

- การใช้เทคนิคการวัดความสำคัญ เช่น Random Forest

3.6.2 การเลือกฟีเจอร์โดยใช้สถิติ:

- การใช้ SelectKBest หรือ chi-square เพื่อเลือกฟีเจอร์ที่ดีที่สุด
- การลบฟีเจอร์ที่ไม่มีความสัมพันธ์กับเป้าหมาย

บทที่ 3 การเตรียมข้อมูลสำหรับการทำเหมืองข้อมูล

Agenda

3.7 การจัดการข้อมูลที่ไม่สมดุล (Handling Imbalanced Data)

3.7.1 การใช้เทคนิค oversampling:

- การใช้ SMOTE เพื่อเพิ่มข้อมูลในกลุ่มที่มีจำนวนน้อย

3.7.2 การใช้เทคนิค undersampling:

- การใช้ RandomUnderSampler เพื่อลดข้อมูลในกลุ่มที่มีจำนวนมาก

บทที่ 3 การเตรียมข้อมูลสำหรับการทำเหมืองข้อมูล

Agenda

3.8 การแบ่งข้อมูล (Data Splitting)

3.8.1 การแบ่งข้อมูลเป็นชุดฝึก (Training Set) และชุดทดสอบ (Test Set)

- การใช้ train_test_split จาก Scikit-learn
- การตั้งค่าขนาดของชุดทดสอบและค่าการสุ่ม (random_state)

3.9 เครื่องมือและไลบรารีที่ใช้ในการเตรียมข้อมูล

3.9.1 Pandas สำหรับการจัดการข้อมูลเชิงโครงสร้าง

3.9.2 NumPy สำหรับการจัดการข้อมูลที่เป็นอาเรย์และตัวเลข

3.9.3 Scikit-learn สำหรับการแปลงข้อมูลและการคัดเลือกฟีเจอร์

3.9.4 Imbalanced-learn สำหรับการแก้ปัญหาข้อมูลที่ไม่สมดุล

บทที่ 3 การเตรียมข้อมูลสำหรับการทำเหมืองข้อมูล

Agenda

3.10 สรุป

3.10.1 สรุปความสำคัญของการเตรียมข้อมูลที่ดีและความสัมพันธ์กับความแม่นยำในการทำเหมืองข้อมูล

3.10.2 ย้ำถึงความยืดหยุ่นของ Python และไลบรารีที่ช่วยในการเตรียมข้อมูล

ทบทวน

CRISP-DM 6 ขั้นตอน

- CRISP-DM ย่อมาจาก Cross-industry standard process for data mining
- ซึ่งหมายถึง กระบวนการมาตรฐานที่ใช้สำหรับการทำเหมืองข้อมูล เพื่อทำการวิเคราะห์และนำไปใช้ประโยชน์ทางธุรกิจ ประกอบไปด้วย 6 ขั้นตอน ดังภาพ

- พัฒนาโดย 3 บริษัท
  - บริษัท SPSS
  - บริษัท Daimler Chrysler
  - บริษัท NCR

ทบทวน

CRISP-DM 6 ขั้นตอน

1. เข้าใจธุรกิจ (Business Understand)

เป็นขั้นตอนแรกมุ่งไปที่การทำความเข้าใจธุรกิจ ปัญหาและวัตถุประสงค์ของโครงการจากมุมมองทางธุรกิจ จากนั้นแปลงปัญหาให้อยู่ในรูปของโจทย์สำหรับการวิเคราะห์ข้อมูล และวางแผนการดำเนินงานเบื้องต้น

1. ทำความเข้าใจปัญหา หรือโอกาสเชิงธุรกิจ
2. ระบุ output หรือเป้าหมายที่ต้องการได้จากการวิเคราะห์ข้อมูลด้วย Data Mining

- ตัวอย่างเช่น
  - ทำอย่างไรถึงเพิ่มยอดขายให้กับสินค้าชนิดต่าง ๆ ได้
  - ต้องการแบ่งกลุ่มนักศึกษาออกตามความสนใจ / ถนัด
  - ทำอย่างไรให้ลูกค้ากลับมาซื้อสินค้าอีก
  - อยากทำนายปริมาณน้ำฝนที่ตกในอีก 2 วันข้างหน้า
  - อยากรู้ว่าลูกค้าคนใหนมีโอกาสป่วยเป็นโรคมะเร็ง

CRISP-DM 6 ขั้นตอน

ทบทวน

ตัวอย่างของการเข้าใจธุรกิจ

ตัวอย่างร้าน H&W CAFÉ ร้าน เบเกอรรี่ มีอาหารและเครื่องดื่ม มีระบบขายสินค้าด้วยคอมพิวเตอร์ มีการบันทึกลงฐานในฐานข้อมูล ...สิ่งที่เราสนใจหรือโจทย์ก็คือ ผู้ซื้อสินค้าหรือลูกค้า แบ่งออกเป็นออกเป็น 3 ประเภท 1. ทานที่ร้าน 2. สั่งกลับบ้าน 3. สั่งผ่านแอป/เดริเวอรี่

เมื่อทราบลูกค้าแบ่งออกเป็นประเภท ร้านค้าก็สามารถสร้างโปรโมชันสินค้าให้ตรงกับความต้องการของลูกค้าแต่ชะกลุ่ม

CRISP-DM 6 ขั้นตอน

ทบทวน

2. ความเข้าใจข้อมูล (Data Understanding)

ขั้นตอนนี้เริ่มต้นด้วยการรวบรวมข้อมูล จากนั้นทำความเข้าใจ ตรวจสอบคุณภาพของข้อมูล และเลือกข้อมูลที่เก็บรวบรวมมาว่าจะใช้ข้อมูลใดบ้างในการวิเคราะห์

โดยในขั้นตอนนี้เป็นการ รวบรวมข้อมูลที่เกี่ยวข้อง ข้อมูลถูกต้องมีน่าเชื่อถือ ข้อมูลที่ได้ต้องมีปริมาณมากเพียงพอ ข้อมูลที่ได้ต้องมีความเหมาะสม และมีรายละเอียดเพียงพอต่อการนำไปวิเคราะห์ ตัวอย่างเช่น

1. ข้อมูลการซื้อสินค้าของแต่ละคนทั้งปี พ.ศ. 2567
2. ข้อมูลผลการเรียนและการลงทะเบียนเรียนของนักศึกษาตลอดหลักสูตร 4 ปี เป็นต้น

ขั้นตอนที่ 1 และ 2 สามารถทำกลับไปมาได้ เนื่องจากการทำความเข้าใจธุรกิจทำให้เราเข้าใจข้อมูลมากขึ้น และการเข้าใจข้อมูลก็ทำให้เราเข้าใจธุรกิจมากขึ้นเช่นกัน

ทบทวน

CRISP-DM 6 ขั้นตอน

ตัวอย่าง การทำความเข้าใจข้อมูล

จาก ตย. งานวิจัย ได้มาจากฐานข้อมูลการสั่งอาหารของร้าน ตั้งแต่ ปี 2557-2567 (จำนวนหรือ ขนาดข้อมูลมีความสำคัญ) จากตัวอย่างงานวิจัย ใช้ข้อมูล 37020 รายการ แยกเป็น 3 ชุด คือ ชุดที่ 1 นั่งทานที่ร้าน ชุดที่ 2 สั่งกลับไปทานที่บ้าน และชุดที่ 3 สั่งผ่านเดลิเวอรี่

ขั้นตอนที่ 1 และ 2 สามารถทำกลับไปมาได้ เนื่องจากการทำความเข้าใจธุรกิจทำให้เราเข้าใจข้อมูลมากขึ้น และการเข้าใจข้อมูลก็ทำให้เราเข้าใจธุรกิจมากขึ้นเช่นกัน

ทบทวน

CRISP-DM 6 ขั้นตอน

3. การเตรียมข้อมูล (Data preparation)

ขั้นตอนการเตรียมข้อมูล หมายถึง ขั้นตอนทั้งหมดที่จะทำเพื่อให้ข้อมูลดิบที่เรารวบรวมมากลายเป็นข้อมูลสมบูรณ์ที่พร้อมจะเข้าสู่โมเดลในขั้นตอนที่ 4 ขั้นตอนนี้จึงเป็นขั้นตอนที่ต้องใช้เวลานานที่สุด เนื่องจากโมเดลที่ได้จากการทำดาต้าไมนิ่งจะให้ผลลัพธ์ที่ถูกต้องหรือไม่นั้นขึ้นอยู่กับคุณภาพของข้อมูลที่ใช้ เช่น การสร้างตาราง การลบข้อมูลที่ไม่ต้องการออก การแปลงข้อมูลให้อยู่ในรูปแบบที่ต้องการ เป็นต้น โดยประกอบไปด้วย 3 ขั้นตอนดังต่อไปนี้

1) การคัดเลือกข้อมูล (Select Data) เลือกตาราง เลือกแอตทริบิวต์

จากตัวอย่างงานวิจัย เลือกข้อมูลที่เกี่ยวข้องกับการสั่งอาหาร ข้อมูลพนักงานไม่ได้เลือก

2) การทำความสะอาดข้อมูล (Data Cleaning)

3) การแปลงข้อมูล (Data Transformation)

ทบทวน

CRISP-DM 6 ขั้นตอน

4. การสร้างโมเดล (Modeling)

ในขั้นตอนนี้ เราจะเลือกและทดสองสร้างโมเดลหลาย ๆ แบบที่น่าจะสามารถแก้ไขปัญหาที่ต้องการได้ จากนั้นค่อย ๆ ปรับค่าพารามิเตอร์ในแต่ละโมเดล เพื่อให้ได้โมเดลที่เหมาะสมที่สุดมาใช้ในการแก้ไขปัญหา

หมายเหตุ หากยังไม่ได้โมเดลที่พอใจ เราสามารถกลับไปเตรียมข้อมูลให้พร้อมมากกว่านี้ได้ เนื่องจากข้อมูลที่ดี ก็จะทำให้ได้ผลลัพธ์ที่ดีเช่นกัน ดังคำกล่าวที่ว่า
“Garbage in, Garbage out” นั่นเอง

CRISP-DM 6 ขั้นตอน

ทบทวน

5. การวัดประสิทธิภาพของโมเดล (Evaluation)

เราจะทำการวัดประสิทธิภาพของโมเดลที่ได้จากขั้นตอนที่ 4 เพื่อวัดว่าโมเดลมีประสิทธิภาพเพียงพอต่อการนำไปใช้งานแล้วหรือไม่ ซึ่งโมเดลแต่ละประเภทก็จะมีตัววัดประสิทธิภาพที่แตกต่างกันออกไป

ทบทวน

CRISP-DM 6 ขั้นตอน

6. การนำโมเดลไปใช้งานจริง (Deployment)

เป็นการนำโมเดลที่เหมาะสมที่สุดไปใช้งานจริง เพื่อวิเคราะห์และแก้ปัญหาที่ต้องการ

***** ในขั้นตอนการสร้างโมเดล อาจจะต้องทำการสร้างโมเดลหลาย ๆ ตัว เพื่อที่จะเลือกเอาผลลัพธ์หรือโมเดลที่เหมาะสม(ค่าความถูกต้อง + ความเร็ว) ที่สุดต่อการนำไปใช้งาน**

บทที่ 3

บทที่ 3 การเตรียมข้อมูลสำหรับทำเหมืองข้อมูล (Data Pre-Processing)

3.1 ความหมายและความสำคัญของเตรียมข้อมูลสำหรับการทำเหมืองข้อมูล

3.1.1 ความหมายของข้อมูล

3.1.2 ประเภทของข้อมูล

3.1.3 แหล่งที่มาของข้อมูลสำหรับเหมืองข้อมูล

3.1.4 ลักษณะของข้อมูลที่เหมาะสมสำหรับเหมืองข้อมูล

3.1.5 การเตรียมข้อมูลสำหรับเหมืองข้อมูล

3.1.6 ตัวอย่างของข้อมูล (Data)

3.1.1 ความหมายของข้อมูล

ข้อมูล หมายถึง ข่าวสาร เอกสาร ข้อเท็จจริงเกี่ยวกับบุคคล สิ่งของหรือเหตุการณ์ที่มีอยู่ในรูปของตัวเลข ภาษา ภาพ สัญลักษณ์ต่างๆ ที่มีความหมายเฉพาะตัว ซึ่งยังไม่มีการประมวลไม่เกี่ยวกับการนำไปใช้ได้อย่างมีประสิทธิภาพ (ไพโรจน์ คชชา, 2542) พจนานุกรมฉบับราชบัณฑิตยสถาน (2525) ให้ความหมายของข้อมูล(Data) หมายถึง ข้อเท็จจริงหรือสิ่งที่ถือหรือยอมรับว่าเป็นข้อเท็จจริง สำหรับใช้เป็นหลักอนุมานหาความจริงหรือ การคำนวณ
จากข้อความนี้อธิบายความหมายของ "ข้อมูล" โดยอ้างอิงจาก 2 แหล่ง:

1. ไพโรจน์ คชชา (2542) ให้ความหมายว่าเป็นข้อมูลดิบที่ยังไม่ผ่านการประมวลผล ซึ่งอาจอยู่ในรูปแบบต่างๆ เช่น:
   - ตัวเลข
   - ภาษา
   - ภาพ
   - สัญลักษณ์ต่างๆ
2. พจนานุกรมฉบับราชบัณฑิตยสถาน (2525) ให้ความหมายว่าเป็นข้อเท็จจริงที่สามารถนำไปใช้:
   - เป็นหลักในการหาความจริง
   - ใช้ในการคำนวณ

3.1.2 ลักษณะของข้อมูล

นับจำนวนนักศึกษาที่ขึ้น – ลิฟต์ในตึก 8

แหล่งข้อมูลอื่น ๆ เช่น งานวิจัย

1. **ข้อมูลจากแหล่งปฐมภูมิ (Primary source)** การเก็บรวบรวมข้อมูลจากแหล่งนี้ ผู้รวบรวมต้องไปเก็บจากต้นตอจริง ๆ ซึ่งอาจได้จากการสัมภาษณ์วิธี นับสังเกต ทำการทดลอง หรือสังเกตแบบสอบถามทางโปรยงหน่วย เป็นต้นข้อมูลที่รวบรวมได้จากแหล่งนี้เรียกว่า ข้อมูลปฐมภูมิ (Primary data)
2. **ข้อมูลจากแหล่งทุติยภูมิ (Secondary source)** การเก็บรวบรวม

3.1.2 ลักษณะของข้อมูล

ข้อมูลที่ต่อเนื่อง (Continuous Data) คือข้อมูลที่สามารถมีค่าในช่วงหรือความกว้างที่ไม่สิ้นสุดและสามารถที่จะมีค่าทศนิยมได้ เป็นต้นไป ต่อไปนี้คือตัวอย่างข้อมูลที่ต่อเนื่อง

1. ข้อมูลน้ำหนักของนักวิ่ง

plaintext

รอบการวิ่ง	น้ำหนัก (กิโลกรัม)
1	70.5
2	71.2
3	69.8
4	72.0
5	70.2

3. ข้อมูลอุณหภูมิในห้อง

plaintext Copy code

เวลา	อุณหภูมิ (องศาเซลเซียส)
08:00 น.	25.5
12:00 น.	28.0
16:00 น.	26.8
20:00 น.	24.3
00:00 น.	22.1

3.1.2 ลักษณะของข้อมูล

ลักษณะของข้อมูล จำแนกตามคุณภาพหรือปริมาณ

1. Qualitative or Categorical Data
   - คุณภาพสินค้า ข้อมูลที่วัดได้คือ ดี หรือ เสีย
   - การมีพันธบัตรครอบครอง ข้อมูลที่วัดได้คือ มี หรือ ไม่มี
   - กลุ่มเลือด ข้อมูลที่วัดได้คือ A, B, O, AB
2. Quantitative or Numerical Data
   - Discrete data
   - จำนวนนิสิตที่เข้ามาใช้บริการห้องสมุดในแต่ละวัน
   - จำนวนหนังสือที่ยืมในห้องสมุด
3. Continuous data
   - ความสูง
   - ค่าใช้จ่าย น้ำหนัก อายุ

3.1.3 ประเภทของข้อมูลสำหรับเหมืองข้อมูล

ประเภทของข้อมูลสำหรับเหมืองข้อมูล

1.ข้อมูลเชิงโครงสร้าง (Structured Data)
   1. ข้อมูลที่ถูกจัดเก็บในรูปแบบที่มีโครงสร้างชัดเจน เช่น ตารางฐานข้อมูล (Databases) หรือแผ่นงาน Excel
   2. ตัวอย่าง: ข้อมูลยอดขายสินค้า, ข้อมูลลูกค้า, ข้อมูลสินค้าคงคลัง

2.ข้อมูลกึ่งโครงสร้าง (Semi-Structured Data)
   1. ข้อมูลที่มีบางส่วนเป็นโครงสร้างแต่ไม่สมบูรณ์ เช่น XML, JSON
   2. ตัวอย่าง: ข้อมูลจากไฟล์ JSON ของ API, ข้อมูลที่ดึงมาจากเว็บ

3.ข้อมูลไม่มีโครงสร้าง (Unstructured Data)
   1. ข้อมูลที่ไม่มีโครงสร้างชัดเจน
   2. ตัวอย่าง: ข้อความในโซเชียลมีเดีย, อีเมล, รูปภาพ, วิดีโอ

4.ข้อมูลตามเวลา (Temporal Data)
   1. ข้อมูลที่เปลี่ยนแปลงตามเวลา
   2. ตัวอย่าง: ข้อมูลสภาพอากาศรายวัน, ข้อมูลหุ้น

5.ข้อมูลเชิงพื้นที่ (Spatial Data)
   1. ข้อมูลที่เชื่อมโยงกับสถานที่หรือพื้นที่
   2. ตัวอย่าง: ข้อมูล GPS, แผนที่

3.1.3 ประเภทของข้อมูลสำหรับเหมืองข้อมูล

ประเภทของข้อมูลตามลักษณะการเก็บ ดังนี้

ประเภทของข้อมูลตามลักษณะการเก็บ ดังนี้

1. ได้จากการนับ (Counting data or Enumeration data) ซึ่งข้อมูลจะมีลักษณะไม่ต่อเนื่อง (Discrete data)
2. ได้จากการชั่ง ตวง วัด (Measurement data)ซึ่งข้อมูลจะมีลักษณะต่อเนื่อง (Continuous data)

3.1.3 ประเภทของข้อมูลสำหรับเหมืองข้อมูล

ประเภทของข้อมูลตามลักษณะการแสดงข้อเท็จจริงได้ดังนี้

ประเภทของข้อมูลตามลักษณะการแสดงข้อเท็จจริงได้ดังนี้

1. **ข้อมูลเชิงคุณภาพ (Qualitative data หรือ Categorical data) เป็นข้อมูลที่แสดงคุณลักษณะหรือลักษณะเฉพาะ ไม่สามารถวัดเป็นตัวเลขได้ เช่น เพศ สีผม สถานภาพสมรส เป็นต้น**
2. **ข้อมูลเชิงปริมาณ (Quantitative data หรือ Numerical data) เป็นข้อมูลที่แสดงค่าเป็นตัวเลข แบ่งออกเป็น**

- ข้อมูลไม่ต่อเนื่อง (Discrete data) เป็นข้อมูลที่มีค่าเฉพาะจำนวนเต็ม เช่น จำนวนเด็ก จำนวนรถยนต์ เป็นต้น

- ข้อมูลต่อเนื่อง (Continuous data) เป็นข้อมูลที่สามารถมีค่าระหว่างตัวเลขได้อย่างต่อเนื่อง เช่น น้ำหนัก ความสูง อุณหภูมิ เป็นต้น

3.1.4 แหล่งที่มาของข้อมูลสำหรับเหมืองข้อมูล

แหล่งที่มาของข้อมูลสำหรับ เหมืองข้อมูล (Data Mining) สามารถมาจากหลายที่ขึ้นอยู่กับประเภทของข้อมูลและวัตถุประสงค์ของการทำเหมืองข้อมูล ตัวอย่างแหล่งที่มาได้แก่

1. ฐานข้อมูล (Databases):
   - ฐานข้อมูลเชิงสัมพันธ์ (Relational Databases)
   - ฐานข้อมูล NoSQL เช่น MongoDB
2. คลังข้อมูล (Data Warehouses):
   - ใช้รวบรวมข้อมูลจากแหล่งต่าง ๆ เพื่อการวิเคราะห์
3. ข้อมูลบนเว็บ (Web Data):
   - ข้อมูลที่รวบรวมจากเว็บไซต์ เช่น ข้อมูลจากโซเชียลมีเดีย
4. ข้อมูลจากเซ็นเซอร์ (Sensor Data):
   - ข้อมูลที่ได้รับจากอุปกรณ์ IoT หรือเซ็นเซอร์ต่าง ๆ
5. ข้อมูลธุรกรรม (Transaction Data):
   - ข้อมูลที่เกิดขึ้นจากการทำธุรกรรม เช่น การซื้อขายสินค้า

3.1.4 แหล่งที่มาของข้อมูลสำหรับเหมืองข้อมูล

ลักษณะของข้อมูลที่เหมาะสมสำหรับเหมืองข้อมูล ต้องมีลักษณะดังนี้

1. มีปริมาณเพียงพอ (Sufficient Volume):
   - ข้อมูลต้องมีขนาดเพียงพอที่จะช่วยให้สามารถวิเคราะห์และหาแบบแผนได้
2. มีความหลากหลาย (Variety):
   - ข้อมูลที่หลากหลายช่วยให้ได้ผลลัพธ์ที่ครอบคลุม
3. มีคุณภาพ (Data Quality):
   - ข้อมูลต้องครบถ้วน ไม่มีค่าที่ขาดหาย (Missing Values) และไม่มีความผิดพลาด (Errors)
4. ข้อมูลต้องเป็นปัจจุบัน (Timeliness):
   - ข้อมูลควรทันสมัยและสะท้อนสถานการณ์ปัจจุบัน
5. มีการเก็บข้อมูลอย่างต่อเนื่อง (Continuity):
   - ข้อมูลที่ถูกบันทึกต่อเนื่องช่วยให้วิเคราะห์แนวโน้มได้

3.1.5 การเตรียมข้อมูลสำหรับเหมืองข้อมูล

การเตรียมข้อมูล (Data Preparation) ซึ่งเป็นกระบวนการหนึ่งที่จำเป็นต้องทำก่อนที่จะนำข้อมูลดิบ (Raw Data) ไปใช้งานหรือวิเคราะห์ต่อไป โดยมีรายละเอียดดังนี้เราอาจพิจารณาการทำเตรียมข้อมูลเป็นระบบอย่างหนึ่ง ที่มี input เป็นข้อมูลดิบ และมี output เป็นข้อมูลที่อยู่ในรูปแบบที่พร้อมนำไปใช้งานต่อไปได้ทันที (tidy data: ความหมาย) โดยมากแล้วการนำข้อมูลไปใช้งานต่อมักจะเป็นการนำไปใหลดเข้าจานข้อมูล หรือนำไปวิเคราะห์หาคำตอบอย่างใดอย่างหนึ่งภาพยังแสดงให้เห็นกระบวนการเปลี่ยนแปลงจาก Raw Data ผ่านขั้นตอน Data Preparation จนกลายเป็น Tidy Data ที่พร้อมนำไปใช้งานต่อไป

ไฟล์ .csv , ไฟล์ .xlsx , ไฟล์ ฐานข้อมูล ฯลฯ

3.1.5 การเตรียมข้อมูลสำหรับเหมืองข้อมูล

1) ความสำคัญของการเตรียมข้อมูลสำหรับเหมืองข้อมูล

การเตรียมข้อมูลถือเป็นกระบวนการที่สำคัญมากอย่างหนึ่งในการทำเหมืองข้อมูล หากการเตรียมข้อมูลทำได้ไม่ดี มีโอกาสสูงที่จะก่อให้เกิดความเสียหายในขั้นตอนอื่น ๆ อาจส่งผลให้ผลการวิเคราะห์ หรือการตีความจากการนำข้อมูลไปใช้ ผิดเพี้ยนไป

ในบทนี้จะช่วยให้นิสิตสามารถทำการเตรียมข้อมูลได้อย่างมีประสิทธิภาพมากขึ้น ลดงานในอนาคต และได้ประโยชน์สูงสุดจากการเตรียมข้อมูลสรุปได้ว่า เนื้อหาดังกล่าวเน้นย้ำถึงความสำคัญของการเตรียมข้อมูลให้ถูกต้องเหมาะสม เพื่อผลลัพธ์ที่น่าเชื่อถือและมีประสิทธิภาพมากขึ้น

3.1.5 การเตรียมข้อมูลสำหรับเหมืองข้อมูล

1) ความสำคัญของการเตรียมข้อมูลสำหรับเหมืองข้อมูล (ต่อ...)

ปัญหาต่าง ๆ ที่อาจพบในข้อมูล ซึ่งถือเป็นขั้นตอนก่อนทำการหาข้อมูล ประกอบด้วย

1. ข้อมูลไม่สมบูรณ์ (incomplete data) เช่น ค่าของคุณลักษณะบางตัวขาดหายไป (miksing value) บางคุณลักษณะทั้งลักษณะหรือบางค่าขาด รวมถึงค่าของข้อมูล
2. ข้อมูลรบกวน (noisy data) เช่น ข้อมูลมีค่าผิดพลาด (error) หรือมีค่าผิดปกติ (Outliers)
3. ข้อมูลไม่สอดคล้อง (Inconsistent data) เช่น ข้อมูลเดียวกัน แต่ต่างชุดกัน หรือใช้ค่าแทน ซึ่งปัญหาเหล่านี้ต้องได้รับการแก้ไขก่อนนำข้อมูลไปใช้งาน เพื่อให้การวิเคราะห์ข้อมูลมีความถูกต้องและน่าเชื่อถือมากขึ้น

3.1.5 การเตรียมข้อมูลสำหรับเหมืองข้อมูล

2) ขั้นตอนการเตรียมข้อมูลสำหรับเหมืองข้อมูล มีขั้นตอนดังนี้

1. การรวบรวมข้อมูล (Data Collection):
   - Data Integration เป็นขั้นตอนการรวมแหล่งข้อมูล ซึ่งมีข้อมูล หลายแหล่งมารวมไว้ที่เดียวกัน นำข้อมูลจากแหล่งต่าง ๆ มาเก็บรวมไว้
2. การทำความสะอาดข้อมูล (Data Cleaning):
   - Data Cleaning เป็นขั้นตอนสำหรับการคัดข้อมูลที่เป็นส่วนรบกวน หรือข้อมูลที่ไม่เกี่ยวข้องออกไป จัดการข้อมูลที่ซ้ำซ้อน หรือข้อมูลที่ขาดหาย
3. การเปลี่ยนรูปข้อมูล (Data Transformation):
   - Data Transformation เป็นขั้นตอนการแปลงข้อมูลในขั้นตอนการ คัดเลือก ให้เหมาะสำหรับขั้นตอนการที่เหมืองข้อมูลData Reduction เป็นขั้นตอนการลดมิติข้อมูล , แปลงข้อมูลให้อยู่ในรูปแบบที่เหมาะสม เช่น Normalization
4. การคัดเลือกคุณลักษณะ (Feature Selection):
   - เลือกเฉพาะคุณลักษณะสำคัญที่ใช้ในกระบวนการเหมืองข้อมูล

3.1.5 การเตรียมข้อมูลสำหรับเหมืองข้อมูล

การเก็บรวบรวมข้อมูล

เป็นกระบวนการที่จะให้ได้มาซึ่งข้อมูลที่จะนำไปวิเคราะห์ต่อไป- ข้อมูลที่ได้มาในขั้นตอนนี้จะเรียกว่า ข้อมูลดิบ (Raw Data) และจากที่ทราบว่า การเก็บรวบรวมข้อมูลเป็นขั้นตอนแรกและขั้นตอนที่สำคัญของการวิเคราะห์ข้อมูล ดังนั้นการดำเนินงานในขั้นตอนนี้ต้องมีการวางแผนการเก็บรวบรวมข้อมูลอย่างรอบคอบ และต้องทราบด้วยว่าข้อมูลที่ต้องการนั้นเป็นข้อมูลชนิดใด มาจากแหล่งใด เพื่อให้ได้ข้อมูลที่ถูกต้องตามจุดประสงค์มากที่สุด

3.1.6 ตัวอย่างของข้อมูล (Data Examples)

- ข้อมูลการเกิดอุบัติเหตุทางจราจรของ กทม. ปี 2566
- ข้อมูลการลงทะเบียนของนักศึกษาสาขาวิชาวิทยาการคอมพิวเตอร์
- ข้อมูลการสั่งซื้อสินค้าออนไลน์
- ข้อมูลประวัติพนักงาน

รหัสการสั่ง	วันที่สั่ง	ลูกค้า	สินค้า	จำนวน	ราคาต่อหน่วย
ORD001	2023-01-15	ลูกค้า A	โน๊ตบุ๊ค	2	20,000 บาท
ORD002	2023-01-16	ลูกค้า B	สมาร์ทโฟน	1	15,000 บาท
ORD003	2023-01-18	ลูกค้า A	ทีวี	1	30,000 บาท
ORD004	2023-01-20	ลูกค้า C	แท็บเล็ต	3	8,000 บาท

3.1.6 ตัวอย่างของข้อมูล (Data Examples)

ตัวอย่างข้อมูล : ข้อมูลการสั่งซื้อสินค้า

รหัสการสั่ง	วันที่สั่ง	ลูกค้า	สินค้า	จำนวน	ราคาต่อหน่วย
ORD001	2023-01-15	ลูกค้า A	โน๊ตบุ๊ค	2	20,000 บาท
ORD002	2023-01-16	ลูกค้า B	สมาร์ทโฟน	1	15,000 บาท
ORD003	2023-01-18	ลูกค้า A	ทีวี	1	30,000 บาท
ORD004	2023-01-20	ลูกค้า C	แท็บเล็ต	3	8,000 บาท

3.1.6 ตัวอย่างของข้อมูล (Data Examples)

ตัวอย่างข้อมูล : ข้อมูลลูกค้า

รหัสลูกค้า	ชื่อ-สกุล	เพศ	อายุ	ที่อยู่	อีเมล
CUS001	นายสมชาย	ชาย	35	123 ถนนสุขสวัสดิ์ แขวงรัชดา	somchai@example.com
CUS002	นางสมหญิง	หญิง	28	456 หมู่บ้านพัฒนา ตำบลลาดพร้าว	somying@example.com
CUS003	นายวรรธนะ	ชาย	45	789 ถนนรามคำแหง แขวงหัวหมาก	worathanon@example.com

3.1.6 ตัวอย่างของข้อมูล (Data Examples)

ตัวอย่างข้อมูล : ข้อมูลสินค้าคงเหลือ

รหัสสินค้า	ชื่อสินค้า	ประเภท	ราคา	จำนวนคงเหลือ
PROD001	โน๊ตบุ๊ค	อิเล็กทรอนิกส์	25,000 บาท	50
PROD002	สมาร์ทโฟน	อิเล็กทรอนิกส์	20,000 บาท	30
PROD003	ทีวี	อิเล็กทรอนิกส์	40,000 บาท	15
PROD004	แท็บเล็ต	อิเล็กทรอนิกส์	10,000 บาท	40

3.2 การนำเข้าข้อมูล (Data Import)

3.2 การนำเข้าข้อมูล (Data Import)

3.2.1 แนะนำวิธีการนำเข้าข้อมูลจากแหล่งต่าง ๆ ด้วย Python

การนำเข้าข้อมูลด้วย Python สามารถทำได้จากแหล่งต่างๆ เช่น ไฟล์ csv, excel, Text, ฐานข้อมูล, API, หรือแม้แต่เว็บเพจ โดยใช้ไลบรารีที่เหมาะสมกับแหล่งข้อมูลนั้น ๆ ต่อไปนี้เป็นตัวอย่างสำหรับการนำเข้าข้อมูลจากแหล่งข้อมูลที่พบบ่อย

บทที่ 3

3.2.2 การนำเข้าจากไฟล์ CSV

1. ไฟล์ (CSV, Excel, JSON, Text)

ไฟล์ CSV

```
import pandas as pd

# อ่านไฟล์ CSV
df = pd.read_csv("data.csv")
print(df.head())
```

บทที่ 3

3.2.3 การนำเข้าจากไฟล์ Excel

1. ไฟล์ (CSV, Excel, JSON, Text)

ไฟล์ Excel

```
import pandas as pd

# อ่านไฟล์ Excel
df = pd.read_excel("data.xlsx", sheet_name="Sheet1")
print(df.head())
```

3.2.3 การนำเข้าจากไฟล์ CSV

1. ไฟล์ (CSV, Excel, JSON, Text)

ไฟล์ JSON

```
import pandas as pd

# อ่านไฟล์ JSON
df = pd.read_json("data.json")
print(df.head())
```

บทที่ 3

3.2.3 การนำเข้าจากไฟล์ Text

1. ไฟล์ (CSV, Excel, JSON, Text)

ไฟล์ Text

```
# อ่านไฟล์ข้อความ
with open("data.txt", "r") as file:
    data = file.readlines()
    print(data)
```

3.2.4 การเชื่อมต่อและนำเข้าข้อมูลจากฐานข้อมูล (MySQL, SQLite, PostgreSQL)

ฐานข้อมูล MySQL

```
import pymysql
import pandas as pd
# การเชื่อมต่อ MySQL
connection = pymysql.connect(host="localhost", user="root", password="", database="your_database")
query = "SELECT * FROM your_table"
# ใช้ Pandas เพื่อดึงข้อมูล
df = pd.read_sql(query, connection)
print(df.head())
connection.close()
```

บทที่ 3

3.2.4 การเชื่อมต่อและนำเข้าข้อมูลจากฐานข้อมูล (MySQL, SQLite, PostgreSQL)

ฐานข้อมูล SQLite

```
import sqlite3
import pandas as pd
# การเชื่อมต่อ SQLite
connection = sqlite3.connect("database.db")
query = "SELECT * FROM your_table"
# ใช้ Pandas เพื่อดึงข้อมูล
df = pd.read_sql(query, connection)
print(df.head())
connection.close()
```

3.2.5 การอ่านข้อมูลจาก APIs หรือ Web Scraping

เว็บเพจ (Web Scraping)

```
import requests
from bs4 import BeautifulSoup
# ดึง HTML จากเว็บเพจ
url = "https://example.com"
response = requests.get(url)
soup = BeautifulSoup(response.text, "html.parser")
# ดึงข้อมูลตามโครงสร้าง HTML
data = soup.find_all("p")  # ตัวอย่าง: ดึงข้อมูลจากแท็ก <p>
for item in data:
    print(item.text)
```

3.2.5 การอ่านข้อมูลจาก APIs หรือ Web Scraping

API

```
import requests
import pandas as pd
# ดึงข้อมูลจาก API
url = "https://api.example.com/data"
response = requests.get(url)
# แปลงข้อมูล JSON เป็น Pandas DataFrame
data = response.json()
df = pd.DataFrame(data)
print(df.head())
```

3.3 การสำรวจข้อมูล (Data Exploration)

3.3 การสำรวจข้อมูล (Data Exploration)

3.3.1 การตรวจสอบข้อมูลเบื้องต้น

- การดูตัวอย่างข้อมูล (head)
- การตรวจสอบขนาดของข้อมูล (shape)
- การดูสถิติพื้นฐาน (describe)

3.3.2 การตรวจสอบค่าที่ขาดหาย (missing values)

3.3.3 การตรวจสอบข้อมูลซ้ำซ้อน (duplicate data)

3.3.1 การตรวจสอบข้อมูลเบื้องต้น

3.3.1 การตรวจสอบข้อมูลเบื้องต้น

- การดูตัวอย่างข้อมูล
- การตรวจสอบขนาดของข้อมูล (shape)
- การดูสถิติพื้นฐาน (describe)

บทที่ 3

3.3.1 การตรวจสอบข้อมูลเบื้องต้น : การดูตัวอย่างข้อมูล

รหัสสินค้า	ชื่อสินค้า	ประเภท	ราคา
PROD001	โน๊ตบุ๊ค	อิเล็กทรอนิกส์	25,000 บา
PROD002	สมาร์ทโฟน	อิเล็กทรอนิกส์	20,000 บา
PROD003	ทีวี	อิเล็กทรอนิกส์	40,000 บา
PROD004	แท็บเล็ต	อิเล็กทรอนิกส์	10,000 บาท

```
# File : dm_prepareData01.py
# Developer : Asst. Prof. Dr. Nattapong Songneam
# Create date : 15.01.2025
# Last Modified : 15.01.2025
```

Lec03_prepareData01_printData.py

การเตรียมข้อมูลด้วยไลบรารี Pandas

```
pip install pandas
```

```
import pandas as pd
data = pd.DataFrame({
    "รหัสสินค้า": ["PROD001", "PROD002", "PROD003", "PROD004"],
    "ชื่อสินค้า": ["โน๊ตบุ๊ค", "สมาร์ทโฟน", "ทีวี", "แท็บเล็ต"],
    "ประเภท": ["อิเล็กทรอนิกส์", "อิเล็กทรอนิกส์", "อิเล็กทรอนิกส์",
"อิเล็กทรอนิกส์"],
    "ราคา": [25000, 20000, 40000, 10000],
    "จำนวนคงเหลือ": [50, 30, 15, 40]
})
print(data)
```

บทที่ 3

3.3.1 การตรวจสอบข้อมูลเบื้องต้น : การดูตัวอย่างข้อมูล

Lec03_prepareData01_printData.py

dm_prepareData01.py > ...

```python
import pandas as pd
data = pd.DataFrame({
  "รหัสสินค้า": ["PROD001", "PROD002", "PROD003", "PROD004"],
 "ชื่อสินค้า": ["โน๊ตบุ๊ค", "สมาร์ทโฟน", "ทีวี", "แท็บเล็ต"],
  "ประเภท": ["อิเล็กทรอนิกส์", "อิเล็กทรอนิกส์", "อิเล็กทรอนิกส์", "อิเล็กทรอนิกส์"],
  "ราคา": [25000, 20000, 40000, 10000],
  "จำนวนคงเหลือ": [50, 30, 15, 40]
})
print(data)
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS SERIAL MONITOR

```
PS C:\AppServ\www\Python_DataMining> & C:/Users/User/AppData/Local/Microsoft/WindowsApps/python3.10.exe c:/AppServ/www/Python_DataMining/dm_prepareData01.py
  รหัสสินค้า ชื่อสินค้า          ประเภท   ราคา  จำนวนคงเหลือ
0    PROD001   โน๊ตบุ๊ค  อิเล็กทรอนิกส์  25000            50
1    PROD002  สมาร์ทโฟน  อิเล็กทรอนิกส์  20000            30
2    PROD003       ทีวี  อิเล็กทรอนิกส์  40000            15
3    PROD004   แท็บเล็ต  อิเล็กทรอนิกส์  10000            40
PS C:\AppServ\www\Python_DataMining>
```

บทที่ 3

3.3.1 การตรวจสอบข้อมูลเบื้องต้น : การดูตัวอย่างข้อมูล

รหัสสินค้า	ชื่อสินค้า	ประเภท	ราคา	จำนวนคงเหลือ
PROD001	โน๊ตบุ๊ค	อิเล็กทรอนิกส์	25,000 บาท	50
PROD002	สมาร์ทโฟน	อิเล็กทรอนิกส์	20,000	
PROD003	ทีวี	อิเล็กทรอนิกส์	40	
PROD004	แท็บเล็ต	อิเล็กทรอนิกส์	10,	

Lec03_prepareData02_printData.py

อย่าลืม ติดตั้งสำหรับ xlsx

pip install openpyxl

```python
import pandas as pd
data = pd.DataFrame({
    "รหัสสินค้า": ["PROD001", "PROD002", "PROD003", "PROD004"],
    "ชื่อสินค้า": ["โน๊ตบุ๊ค", "สมาร์ทโฟน", "ทีวี", "แท็บเล็ต"],
    "ประเภท": ["อิเล็กทรอนิกส์", "อิเล็กทรอนิกส์", "อิเล็กทรอนิกส์", "อิเล็กทรอนิกส์"],
    "ราคา": [25000, 20000, 40000, 10000],
    "จำนวนคงเหลือ": [50, 30, 15, 40]
})
# ระบุชื่อไฟล์ Excel ที่ต้องการบันทึก
excel_file_path = 'C:\AppServ\www\Python_DataMining\data01.xlsx'
# บันทึก DataFrame ลงในไฟล์ Excel
data.to_excel(excel_file_path, index=False)
print(f"บันทึกข้อมูลลงในไฟล์ Excel สำเร็จ: {excel_file_path}")
```

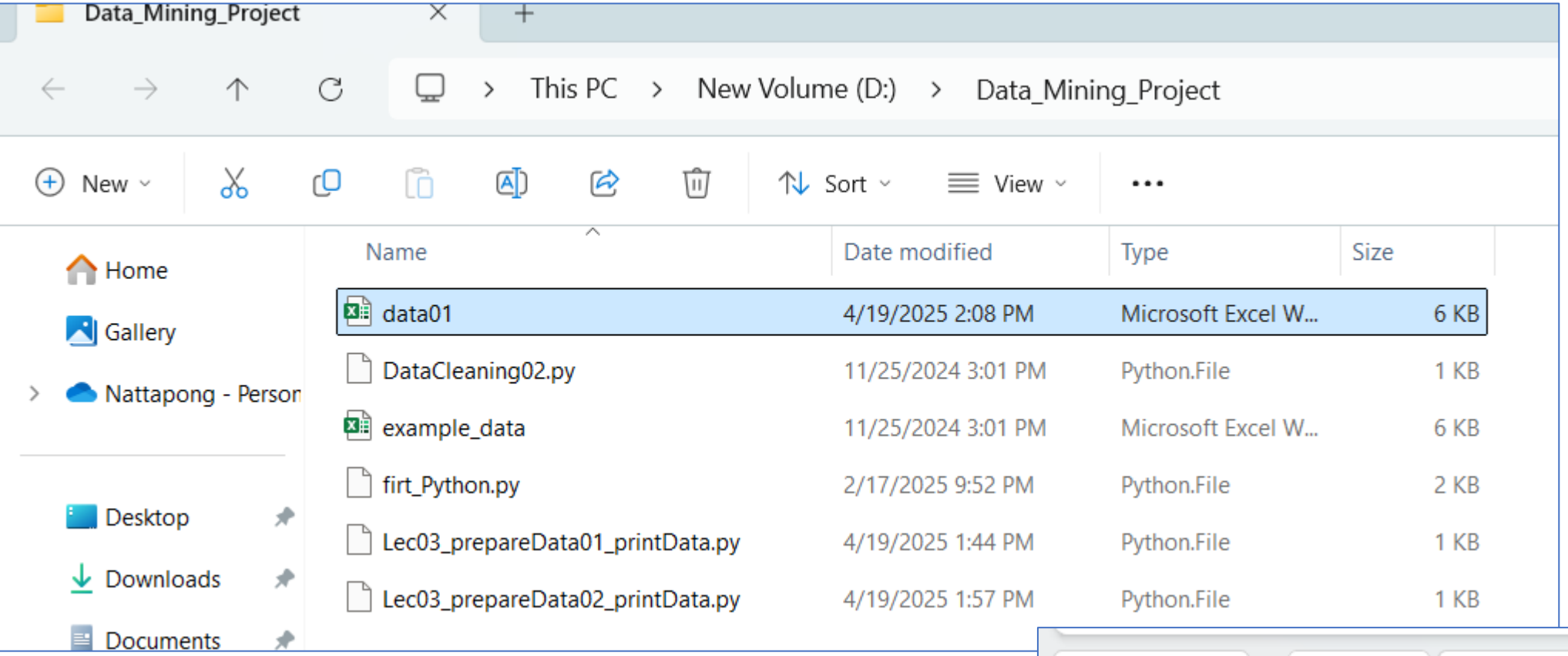
pip install openpyxl

✅ ให้ติดตั้ง `openpyxl` ใหม่ใน Python ตัวที่รันสคริปต์:

powershell Copy Edit

```
& C:/Users/ASUS/AppData/Local/Microsoft/WindowsApps/python3.11.exe -m pip install openpyxl
```

```
& C:/Users/ASUS/AppData/Local/Microsoft/WindowsApps/python3.11.exe
d:/Data_Mining_Project/Lec03_prepareData02_printData.py
```



A1 รหัสสินค้า

	A	B	C	D	E
1	รหัสสินค้า	ชื่อสินค้า	ประเภท	ราคา	จำนวนคงเหลือ
2	PROD001	โน๊ตบุ๊ค	อิเล็กทรอนิ	25000	50
3	PROD002	สมาร์ทโฟน	อิเล็กทรอนิ	20000	30
4	PROD003	ทีวี	อิเล็กทรอนิ	40000	15
5	PROD004	แท็บเล็ต	อิเล็กทรอนิ	10000	40
6					

บทที่ 3

3.3.1 การตรวจสอบข้อมูลเบื้องต้น : การดูตัวอย่างข้อมูล

Lec03_prepareData02_printData.py

```python
import pandas as pd
data = pd.DataFrame({
    "รหัสสินค้า": ["PROD001", "PROD002", "PROD003", "PROD004"],
    "ชื่อสินค้า": ["โบ๊ตบุ๊ค", "สมาร์ทโฟน", "ทีวี", "แท็บเล็ต"],
    "ประเภท": ["อิเล็กทรอนิกส์", "อิเล็กทรอนิกส์", "อิเล็กทรอนิกส์", "อิเล็กทรอนิกส์"],
    "ราคา": [25000, 20000, 40000, 10000],
    "จำนวนคงเหลือ": [50, 30, 15, 40]
})
# ระบุชื่อไฟล์ Excel ที่ต้องการบันทึก
excel_file_path = 'C:\AppServ\www\Python_DataMining\data01.xlsx'
# บันทึก DataFrame ลงในไฟล์ Excel
data.to_excel(excel_file_path, index=False)
print(f"บันทึกข้อมูลลงในไฟล์ Excel สำเร็จ: {excel_file_path}")
```

data01.xlsx

	A	B	C	D	E	F
1	รหัสสินค้า	ชื่อสินค้า	ประเภท	ราคา	นวนคงเหลือ	
2	PROD001	โบ๊ตบุ๊ค	อิเล็กทรอนิ	25000	50	
3	PROD002	สมาร์ทโฟน	อิเล็กทรอนิ	20000	30	
4	PROD003	ทีวี	อิเล็กทรอนิ	40000	15	
5	PROD004	แท็บเล็ต	อิเล็กทรอนิ	10000	40	
6						

3.3.1 การตรวจสอบข้อมูลเบื้องต้น : การดูตัวอย่างข้อมูล

ตัวอย่างที่ 3.3 : จงเตรียมข้อมูลการสั่งซื้อสินค้าด้วย Python

1. ข้อมูลการสั่งซื้อสินค้า

1.1 ตัวอย่างข้อมูล

รหัสการสั่ง	วันที่สั่ง	ลูกค้า	สินค้า	จำนวน	ราคาต่อหน่วย
ORD001	2023-01-15	ลูกค้า A	โน๊ตบุ๊ค	2	20,000 บาท
ORD002	2023-01-16	ลูกค้า B	สมาร์ทโฟน	1	15,000 บาท
ORD003	2023-01-18	ลูกค้า A	ทีวี	1	30,000 บาท
ORD004	2023-01-20	ลูกค้า C	แท็บเล็ต	3	8,000 บาท

บทที่ 3

3.3.1 การตรวจสอบข้อมูลเบื้องต้น : การดูตัวอย่างข้อมูล

ตัวอย่างที่ 3.4 : จงเตรียมข้อมูลรายชื่อลูกค้าด้วย Python

Lec03_prepareData03_printData.py

```
import pandas as pd
data = pd.DataFrame({
    "วันที่": pd.to_datetime(["2023-07-20", "2023-07-21", "2023-07-22", "2023-07-23"]),
    "ลูกค้า": ["นายสมชาย", "นางสมหญิง", "นายสมชาย", "นางสมหญิง"],
    "รายการ": ["ซื้อคอมพิวเตอร์", "ซื้อโทรศัพท์มือถือ", "ซื้อทีวี", "ซื้อแท็บเล็ต"],
    "ยอดเงิน": [15000, 20000, 40000, 10000]
})
print(data)
```

รหัสลูกค้า	ชื่อ-สกุล	เพศ	อายุ	ที่อยู่	อีเมล
CUS001	นายสมชาย	ชาย	35	123 ถนนสุขสวัสดิ์ แขวงรัชดา	somchai@example.com
CUS002	นางสมหญิง	หญิง	28	456 หมู่บ้านพัฒนา ตำบลลาดพร้าว	somying@example.com
CUS003	นายวรรธนะ	ชาย	45	789 ถนนรามคำแหง แขวงหัวหมาก	worathanon@example.com

3.3.1 การตรวจสอบข้อมูลเบื้องต้น : การดูตัวอย่างข้อมูล

ตัวอย่างที่ 3.4 : จงเตรียมข้อมูลรายชื่อลูกค้าด้วย Python

ตัวอย่างข้อมูล : ข้อมูลรายชื่อลูกค้า

รหัส	ชื่อ-สกุล	เพศ	อายุ	ที่อยู่	อีเมล
CUS001	นายสมชาย	ชาย	35	123 ถนนสุขสวัสดิ์ แขวงรัชดา	somchai@example.com
CUS002	นางสมหญิง	หญิง	28	456 หมู่บ้านพัฒนา ตำบลลาดพร้าว	somying@example.com
CUS003	นายวรรธนะ	ชาย	45	789 ถนนรามคำแหง แขวงหัวหมาก	worathanon@example.com

บทที่ 3

3.3.1 การตรวจสอบข้อมูลเบื้องต้น : การดูตัวอย่างข้อมูล

ตัวอย่างที่ 3.4 : จงเตรียมข้อมูลรายชื่อลูกค้าด้วย Python

Lec03_prepareData03_printData.py

dm_prepareData03.py > ...

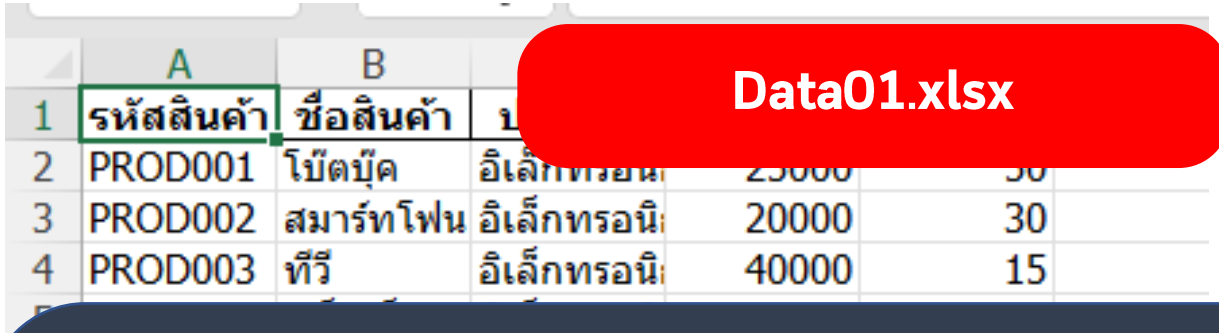
```python
import pandas as pd
data = pd.DataFrame({
"วันที่": pd.to_datetime(["2023-07-20", "2023-07-21", "2023-07-22", "2023-07-23"]),
 "ลูกค้า": ["นายสมชาย", "นางสมหญิง", "นายสมชาย", "นางสมหญิง"],
 "รายการ": ["ซื้อคอมพิวเตอร์", "ซื้อโทรศัพท์มือถือ", "ซื้อทีวี", "ซื้อแท็บเล็ต"],
 "ยอดเงิน": [15000, 20000, 40000, 10000]
})
print(data)

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS SERIAL MONITOR

```
PS C:\AppServ\www\Python_DataMining> & C:/Users/User/AppData/Local/Microsoft/WindowsApps/python3.10.exe c:/AppServ/www/Python_DataMining/dm_prepareData03.py
      วันที่     ลูกค้า          รายการ  ยอดเงิน
0 2023-07-20  นายสมชาย  ซื้อคอมพิวเตอร์     15000
1 2023-07-21  นางสมหญิง  ซื้อโทรศัพท์มือถือ     20000
2 2023-07-22  นายสมชาย          ซื้อทีวี     40000
3 2023-07-23  นางสมหญิง        ซื้อแท็บเล็ต     10000
PS C:\AppServ\www\Python_DataMining>
```

3.3.1 การตรวจสอบข้อมูลเบื้องต้น : การดูตัวอย่างข้อมูล

ตัวอย่างที่ 3.5 การอ่านข้อมูลจากไฟล์ excel ด้วย Python



	A	B			
1	รหัสสินค้า	ชื่อสินค้า	ป		
2	PROD001	โบ๊ตบุ๊ค	อิเล็กทรอนิ	25000	30
3	PROD002	สมาร์ทโฟน	อิเล็กทรอนิ	20000	30
4	PROD003	ทีวี	อิเล็กทรอนิ	40000	15

Data01.xlsx

```
# File : dm_prepareData04.py
# Developer : Asst. Prof. Dr. Nattapong Songneam
# Create date : 15.01.2025
# Last Modified : 15.01.2025
```

Lec03_prepareData04_readData.py

```
import pandas as pd

# ระบุชื่อไฟล์ Excel ที่ต้องการอ่าน
excel_file_path = 'C:\AppServ\www\Python_DataMining\data01.xlsx'
# อ่านข้อมูลจากไฟล์ Excel เข้าสู่ DataFrame
data = pd.read_excel(excel_file_path)
# แสดงข้อมูล
print(data)
```

ตัวอย่างที่ 3.5 การอ่านข้อมูลจากไฟล์ excel ด้วย Python

```
# File : dm_prepareData04.py
# Developer : Asst. Prof. Dr. Nattapong Songneam
# Create date : 15.01.2025
# Last Modified : 15.01.2025
```

Lec03_prepareData04_readData.py

dm_prepareData04.py > ...

```python
import pandas as pd

# ระบุชื่อไฟล์ Excel ที่ต้องการอ่าน
excel_file_path = 'C:\AppServ\www\Python_DataMining\data01.xlsx'

# อ่านข้อมูลจากไฟล์ Excel เข้าสู่ DataFrame
data = pd.read_excel(excel_file_path)

# แสดงข้อมูล
print(data)

```

3.3.1 การตรวจสอบข้อมูลเบื้องต้น : การตรวจสอบขนาดของข้อมูล (shape

บทที่ 3

ตัวอย่างที่ 3.6 การนับจำนวนแถวของข้อมูลด้วย Python

	A	B	C	D	E
1	รหัสสินค้า	ชื่อสินค้า	ประ		
2	PROD001	โบ๊ตบุ๊ค	อิเล็		
3	PROD002	สมาร์ทโฟน	อิเล็กทรอนิ	20000	30
4	PROD003	ทีวี	อิเล็กทรอนิ	40000	15
5	PROD004	แท็บเล็ต	อิเล็กทรอนิ	10000	40
6					

Data01.xlsx

เมื่ออ่านข้อมูลจากไฟล์ Excel ด้วย Pandas แล้วสามารถใช้ shape เพื่อดูขนาดของ DataFrame ซึ่งรวมถึงจำนวนแถวและคอลัมน์

Lec03_prepareData05_countData.py

```python
import pandas as pd

# ระบุชื่อไฟล์ Excel ที่ต้องการอ่าน
excel_file_path = 'C:\AppServ\www\Python_DataMining\data01.xlsx'
# อ่านข้อมูลจากไฟล์ Excel เข้าสู่ DataFrame
data = pd.read_excel(excel_file_path)
# แสดงจำนวนแถวและคอลัมน์
print("จำนวนแถวและคอลัมน์:")
print(data.shape)
# แสดงข้อมูล
print("\nข้อมูล:")
print(data)
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS  SERIAL MO
PS C:\AppServ\www\Python_DataMining> & C:/Users/User/AppData/L
จำนวนแถวและคอลัมน์:
(4, 5)

ข้อมูล:
  รหัสสินค้า ชื่อสินค้า          ประเภท   ราคา  จำนวนคงเหลือ
0    PROD001   โบ๊ตบุ๊ค  อิเล็กทรอนิกส์  25000            50
1    PROD002  สมาร์ทโฟน  อิเล็กทรอนิกส์  20000            30
2    PROD003       ทีวี  อิเล็กทรอนิกส์  40000            15
3    PROD004   แท็บเล็ต  อิเล็กทรอนิกส์  10000            40
PS C:\AppServ\www\Python_DataMining>
```

3.3.1 การตรวจสอบข้อมูลเบื้องต้น : การตรวจสอบขนาดของข้อมูล (shape

บทที่ 3

ตัวอย่างที่ 3.7 แสดงจำนวนแถวของข้อมูลในตัวแปร row, col ด้วย Python

	รหัสสินค้า	ชื่อสินค้า	ประเภท	ราคา	นวนคงเหลือ
2	PROD001				
3	PROD002				
4	PROD003				
5	PROD004	แท็บเล็ต	อิเล็กทรอนิ	10000	40
6					

Data01.xlsx

DataFrame และเก็บในตัวแปร row และ col ตามลำดับ. จากนั้นเราแสดงผลลัพธ์ออกทางหน้าจอ. คุณสามารถใช้ตัวแปร row และ col ในการนำไปใช้ต่อได้ตามความต้องการของคุณ.

Lec03_prepareData06_countData.py

```python
import pandas as pd
# ระบุชื่อไฟล์ Excel ที่ต้องการอ่าน
excel_file_path = 'C:\AppServ\www\Python_DataMining\data01.xlsx'
# อ่านข้อมูลจากไฟล์ Excel เข้าสู่ DataFrame
data = pd.read_excel(excel_file_path)
# เก็บข้อมูลแถวและคอลัมน์
row = data.shape[0]  # จำนวนแถว
col = data.shape[1]  # จำนวนคอลัมน์
# แสดงข้อมูลแถวและคอลัมน์
print(f"จำนวนแถว: {row}")
print(f"จำนวนคอลัมน์: {col}")
# แสดงข้อมูล
print("\nข้อมูล:")
print(data)
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS  SERIAL MON

PS C:\AppServ\www\Python_DataMining> & C:/Users/User/AppData/Loc
จำนวนแถว: 4
จำนวนคอลัมน์: 5

ข้อมูล:
  รหัสสินค้า ชื่อสินค้า          ประเภท   ราคา  จำนวนคงเหลือ
0    PROD001    โน๊ตบุ๊ค  อิเล็กทรอนิกส์  25000            50
1    PROD002  สมาร์ทโฟน  อิเล็กทรอนิกส์  20000            30
2    PROD003        ทีวี  อิเล็กทรอนิกส์  40000            15
3    PROD004    แท็บเล็ต  อิเล็กทรอนิกส์  10000            40
PS C:\AppServ\www\Python_DataMining>
```

3.3.1 การตรวจสอบข้อมูลเบื้องต้น : การดูสถิติพื้นฐาน (describe)

อย่าลืม ติดตั้งสำหรับ xlsx

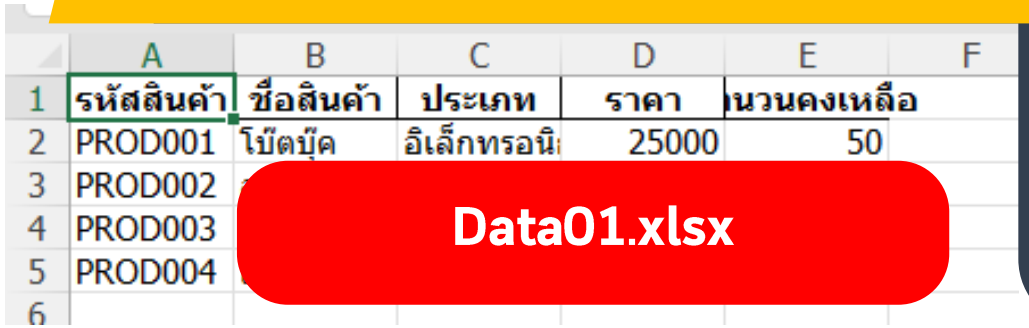
pip install openpyxl

```
    formatter.write(
PS D:\023> &
C:/Users/COM_IT/AppData/Local/Programs/Python/Python311/pytho
n.exe d:/023/klk.py
Traceback (most recent call last):
  File "d:\023\klk.py", line 10, in <module>
    data.to_excel(excel_file_path,index=False)
  File
"C:\Users\COM_IT\AppData\Local\Programs\Python\Python311\Lib\sit
e-packages\pandas\util\_decorators.py", line 333, in wrapper
    return func(*args, **kwargs)
           ^^^^^^^^^^^^^^^^^^^^^
  File
"C:\Users\COM_IT\AppData\Local\Programs\Python\Python311\Lib\sit
e-packages\pandas\core\generic.py", line 2417, in to_excel
    formatter.write(
  File
"C:\Users\COM_IT\AppData\Local\Programs\Python\Python311\Lib\sit
e-packages\pandas\io\formats\excel.py", line 943, in write
    writer = ExcelWriter(
             ^^^^^^^^^^^^
  File
"C:\Users\COM_IT\AppData\Local\Programs\Python\Python311\Lib\sit
e-packages\pandas\io\excel\_openpyxl.py", line 57, in __init__
    from openpyxl.workbook import Workbook
ModuleNotFoundError: No module named 'openpyxl'
PS D:\023>
Q: แก้ไข
```

3.3.1 การตรวจสอบข้อมูลเบื้องต้น : การดูสถิติพื้นฐาน (describe)

บทที่ 3

ตัวอย่างที่ 3.8 การหาค่าเฉลี่ย ค่าสูงสุด ต่ำสุดของข้อมูลใน excel ด้วย Python



	A	B	C	D	E	F
1	รหัสสินค้า	ชื่อสินค้า	ประเภท	ราคา	นวนคงเหลือ	
2	PROD001	โน๊ตบุ๊ค	อิเล็กทรอนิ	25000	50	
3	PROD002					
4	PROD003					
5	PROD004					
6						

Data01.xlsx

(minimum) ของราคาสินค้าจาก DataFrame ที่มีข้อมูลเกี่ยวกับสินค้า คุณสามารถใช้ฟังก์ชันทางสถิติของ Pandas ได้ดังนี้:

ในตัวอย่างนี้, เราใช้ฟังก์ชัน mean, max, และ min ของ Pandas เพื่อหาค่าเฉลี่ย, ค่าสูงสุด, และค่าต่ำสุดของคอลัมน์ 'ราคา' ใน DataFrame. ผลลัพธ์จะถูกแสดงออกทางหน้าจอ.

Lec03_prepareData07_Statistic.py

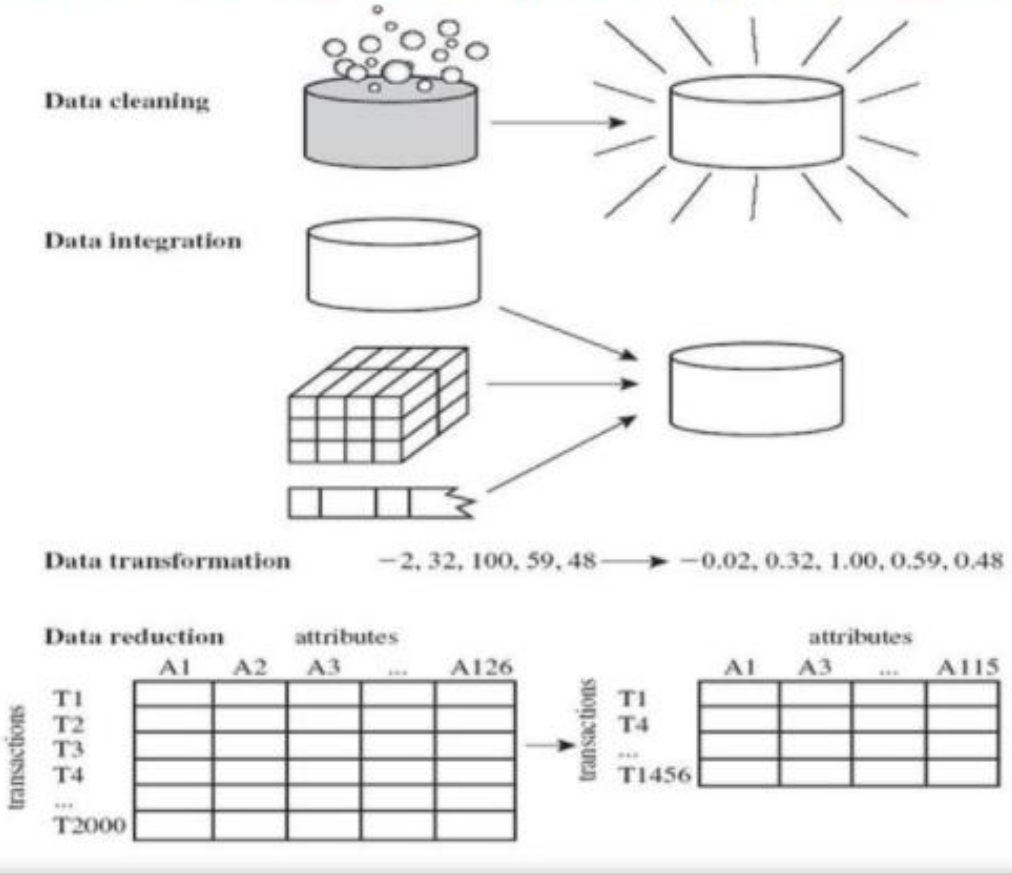
```python
import pandas as pd
# ระบุชื่อไฟล์ Excel ที่ต้องการอ่าน
excel_file_path = 'C:\AppServ\www\Python_DataMining\data01.xlsx'
# อ่านข้อมูลจากไฟล์ Excel เข้าสู่ DataFrame
data = pd.read_excel(excel_file_path)
# หาค่าเฉลี่ย, ค่าสูงสุด, และค่าต่ำสุดของราคาสินค้า
mean_price = data['ราคา'].mean()
max_price = data['ราคา'].max()
min_price = data['ราคา'].min()
# แสดงผลลัพธ์
print(f"ค่าเฉลี่ยราคา: {mean_price:.2f}")
print(f"ค่าสูงสุดของราคา: {max_price}")
print(f"ค่าต่ำสุดของราคา: {min_price}")
```

ผลลัพธ์

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS  S
PS C:\AppServ\www\Python_DataMining> & C:/Users/User/App
ค่าเฉลี่ยราคา: 23750.00
ค่าสูงสุดของราคา: 40000
ค่าต่ำสุดของราคา: 10000
PS C:\AppServ\www\Python_DataMining>
```

Forms of data pre-processing

Forms of data pre-processing



3.3.2 การตรวจสอบค่าที่ขาดหาย (missing values)

Replace missing value

การแทนที่ค่าข้อมูลที่หายไปหรือไม่ถูกต้อง โดยมีรายละเอียดดังนี้ Replace Value

- ข้อมูลจากหลาย ๆ ฐานข้อมูลมักจะเกิดปัญหาข้อมูลไม่ตรงกัน
- ข้อมูลทั้งตัวหนังสือ กรุงเทพมหานคร, กรุงเทพ, กรุงเทพฯ หรือ ก.ท.ม.
- สายการศึกษา สายอาชีพ, อาชีพ, สาขา, นักเรียนปราย หรือ ปวช.

ต้องทำการปรับเปลี่ยนข้อมูลให้ตรงกันก่อนนำไปใช้งานต่อการปรับเปลี่ยนรูปแบบให้ข้อมูลตรงกันจะช่วยให้ข้อมูลมีความสม่ำเสมอและถูกต้องมากขึ้น สามารถนำไปใช้วิเคราะห์หรือประมวลผลได้อย่างมีประสิทธิภาพ

3.3.2 การตรวจสอบค่าที่ขาดหาย (missing values)

Replace missing value

การแทนค่า (Replace Value) เป็นกระบวนการที่ใช้เพื่อแทนค่าที่มีอยู่ในชุดข้อมูลด้วยค่าอื่น ๆ ที่เราต้องการ การแทนค่าสามารถทำได้เพื่อแก้ไขข้อมูลผิดพลาด, การทำความสะอาดข้อมูล, หรือปรับแต่งข้อมูลให้เหมาะสมกับการวิเคราะห์. ต่อไปนี้คือตัวอย่างการแทนค่าใน Python โดยใช้ไลบรารี pandas:

3.3.2 การตรวจสอบค่าที่ขาดหาย (missing values)

บทที่ 3

ตัวอย่างที่ 3.9 Replace missing value

```
import pandas as pd
# สร้าง DataFrame ตัวอย่าง
data = {'ชื่อ': ['John', 'Alice', 'Bob', 'Charlie', 'David'],
        'อายุ': [25, 28, None, 35, 30],
        'เงินเดือน': [50000, 60000, 45000, 70000, None]}
df = pd.DataFrame(data)
# แสดง DataFrame ก่อนการแทนค่า
print("DataFrame ก่อนการแทนค่า:")
print(df)
# แทนค่าของคอลัมน์ 'อายุ' ที่มีค่าเป็น None ด้วยค่าเฉลี่ย
df['อายุ'] = df['อายุ'].fillna(df['อายุ'].mean())
# แทนค่าของคอลัมน์ 'เงินเดือน' ที่มีค่าเป็น None ด้วยค่ามัธยฐาน
df['เงินเดือน'] = df['เงินเดือน'].fillna(df['เงินเดือน'].median())
# แสดง DataFrame หลังการแทนค่า
print("\nDataFrame หลังการแทนค่า:")
print(df)
```

Lec03_prepareData08_ReplaceValues.py

ผลลัพธ์

```
PS C:\AppServ\w
DataFrame ก่อนก
      ชื่อ  อายุ  เงินเดือน
0    John  25.0    50000.0
1   Alice  28.0    60000.0
2     Bob   NaN    45000.0
                      .0
                     NaN

DataFrame หลังการแทนค่า:
      ชื่อ  อายุ  เงินเดือน
0    John  25.0    50000.0
1   Alice  28.0    60000.0
2     Bob  29.5    45000.0
3  Charlie  35.0    70000.0
4   David  30.0    55000.0
PS C:\AppServ\www\Python_DataMining>
```

3.3.2 การตรวจสอบค่าที่ขาดหาย (missing values)

บทที่ 3

ตัวอย่างที่ 3.9 Replace missing value

วิเคราะห์ผลลัพธ์

- เราใช้ fillna() เพื่อแทนค่าที่เป็น None หรือ NaN ด้วยค่าเฉลี่ย (สำหรับคอลัมน์ 'อายุ') และค่ามัธยฐาน (สำหรับคอลัมน์ 'เงินเดือน').
- mean() ใช้เพื่อคำนวณค่าเฉลี่ยของคอลัมน์ 'อายุ'.
- median() ใช้เพื่อคำนวณค่ามัธยฐานของคอลัมน์ 'เงินเดือน'.

```
PS C:\AppServ\www\Python_DataMining> & C:/Users/Use
DataFrame ก่อนการแทนค่า:
      ชื่อ  อายุ  เงินเดือน
0    John  25.0    50000.0
1   Alice  28.0    60000.0
2     Bob   NaN    45000.0
3  Charlie  35.0    70000.0
4   David  30.0        NaN

DataFrame หลังการแทนค่า:
      ชื่อ  อายุ  เงินเดือน
0    John  25.0    50000.0
1   Alice  28.0    60000.0
2     Bob  29.5    45000.0
3  Charlie  35.0    70000.0
4   David  30.0    55000.0
PS C:\AppServ\www\Python_DataMining>
```

การแทนค่ามีประโยชน์มากในการจัดการข้อมูลที่ขาดหาย, ข้อมูลผิดพลาด, หรือข้อมูลที่ไม่เหมาะสมสำหรับการวิเคราะห์.

3.4 การทำความสะอาดข้อมูล (Data Cleaning)

3.4.1 การจัดการค่าที่ขาดหาย (Missing Data):

- การเติมค่าที่ขาดหายด้วยค่าเฉลี่ย, มัธยฐาน, หรือค่าที่เหมาะสม
- การลบแถวหรือคอลัมน์ที่มีข้อมูลหายไปมากเกินไป

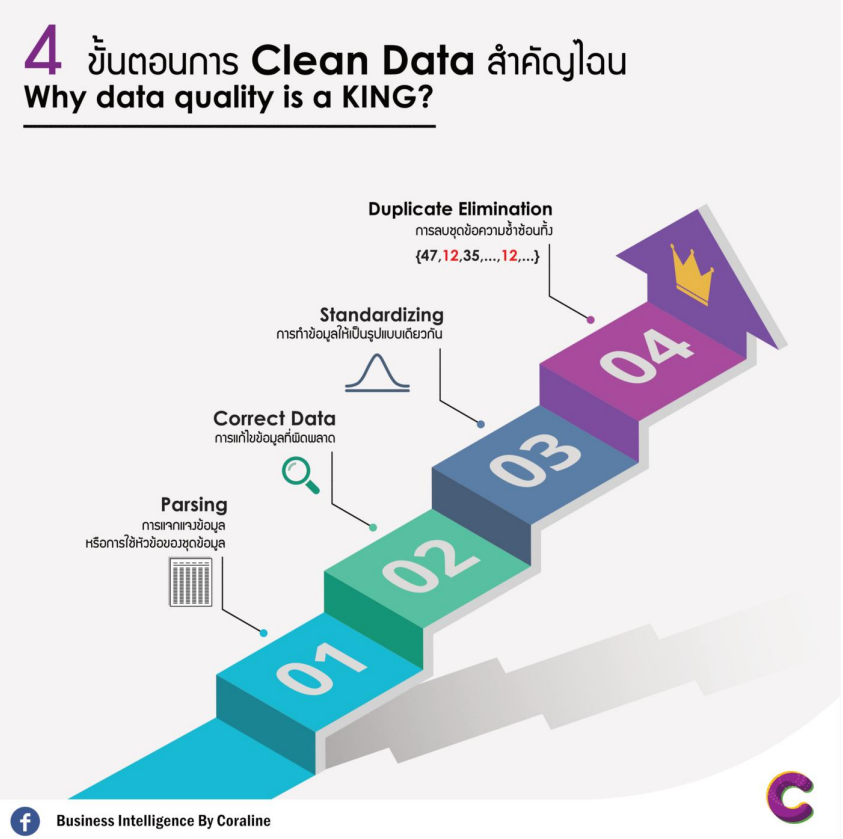
3.4.2 การจัดการค่าผิดปกติ (Outliers):

- การค้นหาและกำจัดค่าผิดปกติ
- การใช้เทคนิค IQR (Interquartile Range) ในการตรวจจับ outliers

3.4 การทำความสะอาดข้อมูล (Data Cleaning)

การทำความสะอาดข้อมูล (Data Cleaning)

นักวิทยาศาสตร์ข้อมูล (Data Scientist) ที่ทำงานจริงส่วนใหญ่ใช้เวลากว่า 80% ในการ Clean ข้อมูล และใช้อีก 20% ที่เหลือในการสร้างโมเดล การ Clean ข้อมูลนี้ เปรียบเหมือนการทำอาหาร เมื่อเรามีวัตถุดิบ นอกจากการคัดสรรวัตถุดิบอย่างพิถีพิถันแล้ว เรายังต้องนำวัตถุดิบนั้นมาทำความสะอาด ปลอกเปลือก ตัดแต่งส่วนที่เน่าเสียออก หั่นให้เป็นรูปร่างที่พร้อมปรุง และอีกหลากหลายขั้นตอน เพื่อให้อาหารจานนั้นถูกปรุงออกมาอย่างดีที่สุด

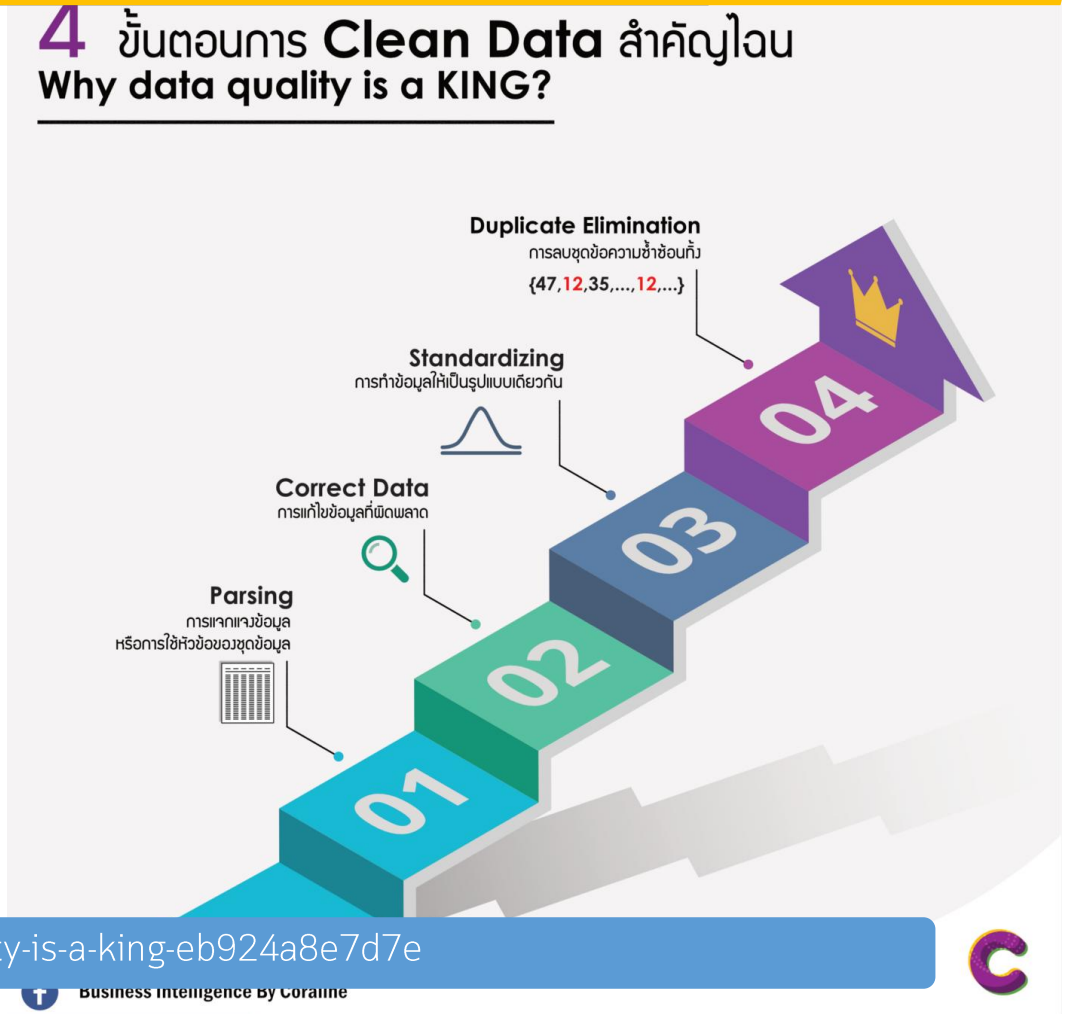


https://coralineltd.medium.com/4-ขั้นตอนการ-clean-data-สำคัญไฉน-why-data-quality-is-a-king-eb924a8e7d7e

3.4 การทำความสะอาดข้อมูล (Data Cleaning)

การทำความสะอาดข้อมูล (Data Cleaning)

เหตุผลที่ข้อมูลไม่สะอาดนั้น มีที่มาได้จากหลากหลายสาเหตุ ตั้งแต่การพิมพ์ผิด พิมพ์ตก เครื่องมือเกิด Error หรือ ข้อมูลตัวเลขที่มีความเป็นไปได้น้อยมาก เช่น คนอายุ 120 ปี หรือ ส่วนสูง 230 ซม. เป็นต้น ซึ่งในทางเทคนิคจะเรียกว่าข้อมูลที่อยู่นอกกลุ่มว่า “Outlier” ดังนั้น Data Scientist ที่ดีนั้นจำเป็นต้องแสดงให้เจ้าของข้อมูลนั้นเข้าใจว่า การ Clean Data นั้นเป็นขั้นตอนที่ยาก สำคัญ และใช้เวลานาน เพราะนอกจากการหาคำจำกัดความของ “ความไม่สะอาด” ของข้อมูลแล้ว เรายังต้องหาวิธีจัดการกับข้อมูลที่ตกหล่นหายไป หรือที่เราเรียกว่า “Missing value” อีกด้วย ทั้งหมดนี้ Data Scientist จำเป็นต้องใช้กลไกทางความคิดเพื่อออกแบบเป็นโมเดลสำหรับการ Clean Data โดยเฉพาะ เนื่องจากเรากำลังพูดถึงข้อมูลขนาดใหญ่ หรือ Big Data ที่ตาเปล่าและสองมือไม่สามารถจัดการได้ไหว



https://coralineltd.medium.com/4-ขั้นตอนการ-clean-data-สำคัญไฉน-why-data-quality-is-a-king-eb924a8e7d7e

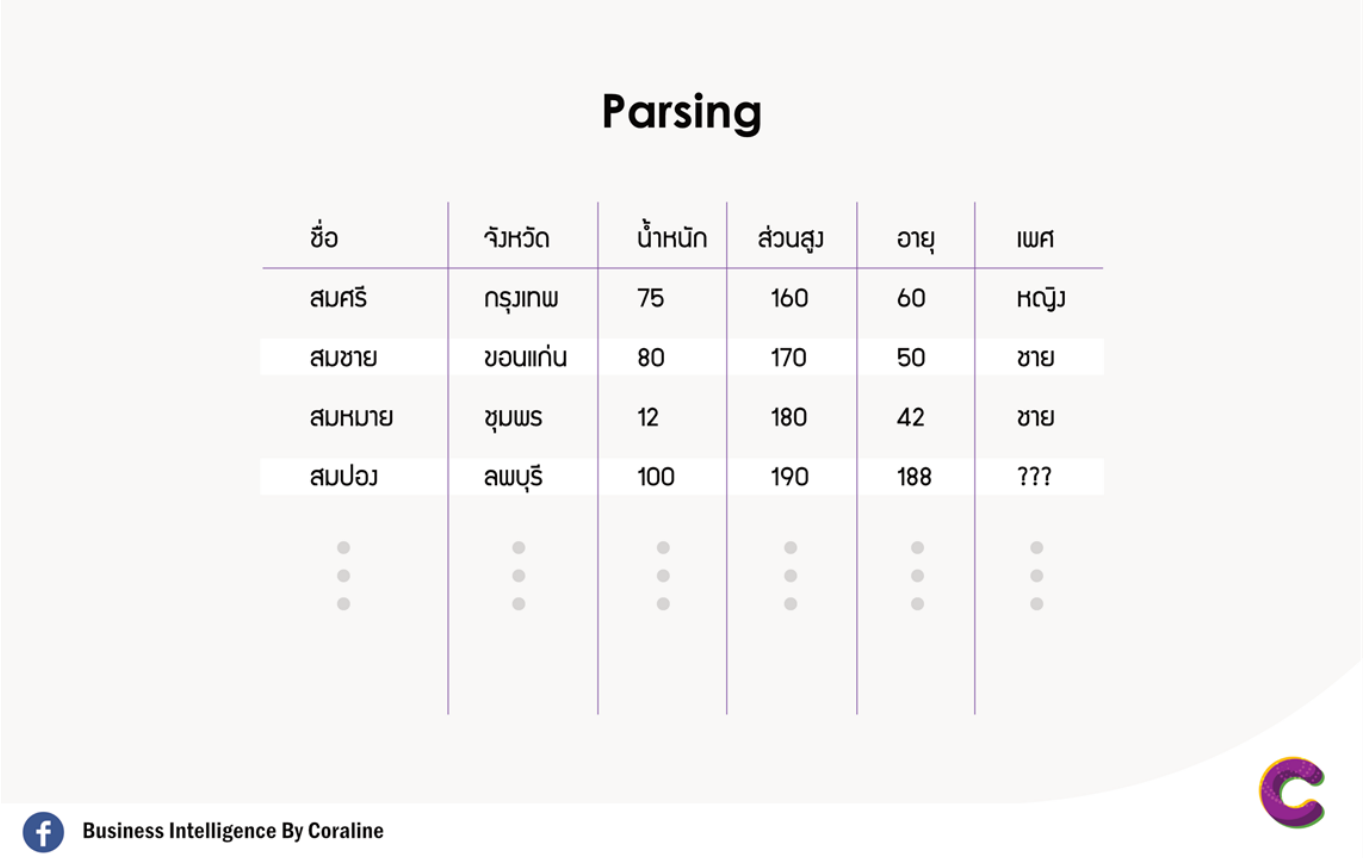
3.4 การทำความสะอาดข้อมูล (Data Cleaning)

การทำความสะอาดข้อมูล (Data Cleaning)

ขั้นตอนเบื้องต้นในการ Clean ข้อมูล 4 ขั้นตอน ได้แก่

1. Parsing คือ การแจกแจงข้อมูล หรือการใช้หัวข้อของชุดข้อมูล เช่น ชื่อ: สมศรี, จังหวัด: กรุงเทพ, น้ำหนัก: 75, ส่วนสูง: 160, อายุ: 60, เพศ: หญิง

ความสำคัญของขั้นตอนนี้ไม่ใช่แค่การใช้ Head ของข้อมูล แต่เป็นการทำความเข้าใจว่าคำจำกัดความของชุดข้อมูลนั้น ๆ คืออะไร รวมไปถึงเข้าใจค่า และความหมายของมัน เช่น มีค่าสูงสุด หรือ ต่ำสุดเท่าไหร่ เป็นต้น

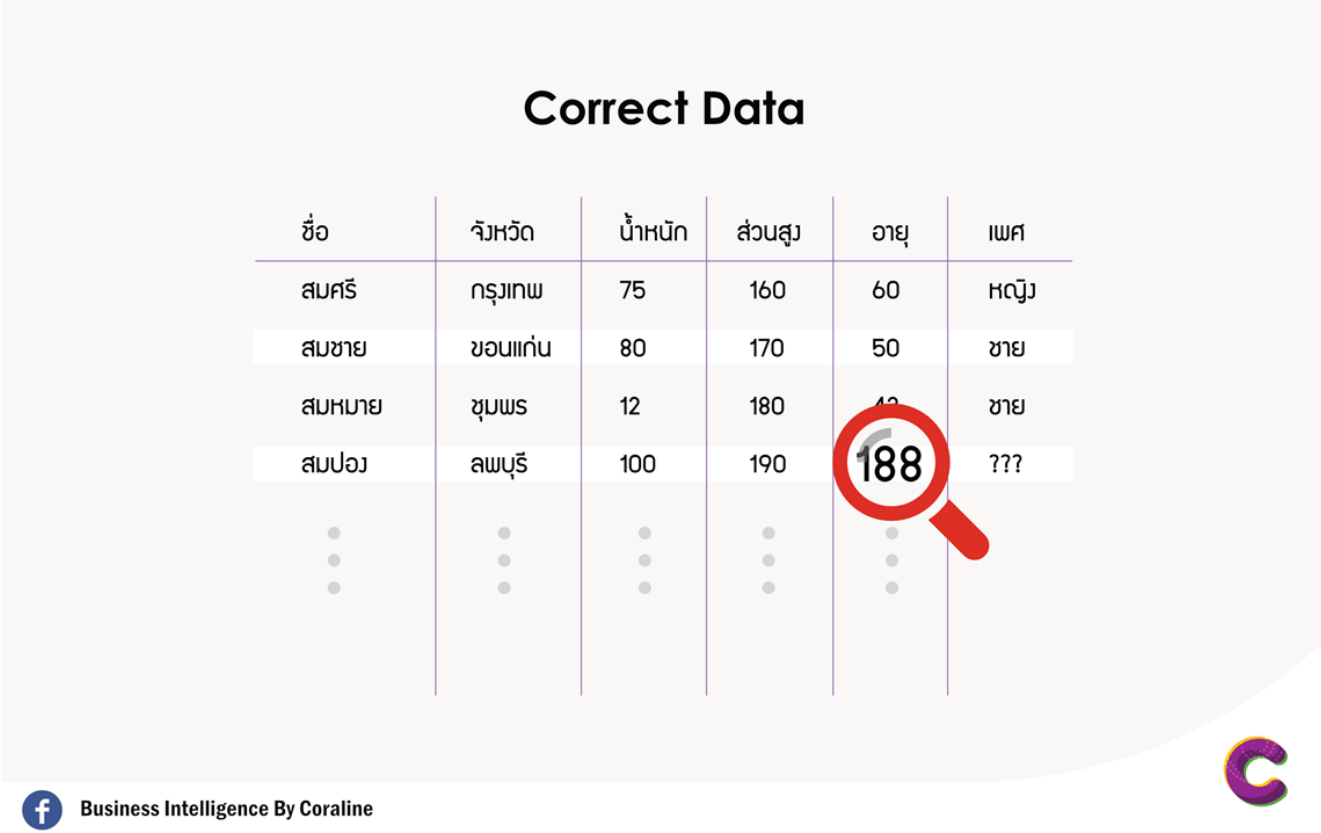


https://coralineltd.medium.com/4-ขั้นตอนการ-clean-data-สำคัญไฉน-why-data-quality-is-a-king-eb924a8e7d7e

3.4 การทำความสะอาดข้อมูล (Data Cleaning)

การทำความสะอาดข้อมูล (Data Cleaning)

2. Correcting คือ การแก้ไขข้อมูลที่ผิดพลาด เช่น ในช่องเพศ มีการใส่ตัวเลข หรือแม้กระทั่งตัวเลขที่ผิดปกติไปเนื่องจากมี 0 เกินมา ก็เป็นได้

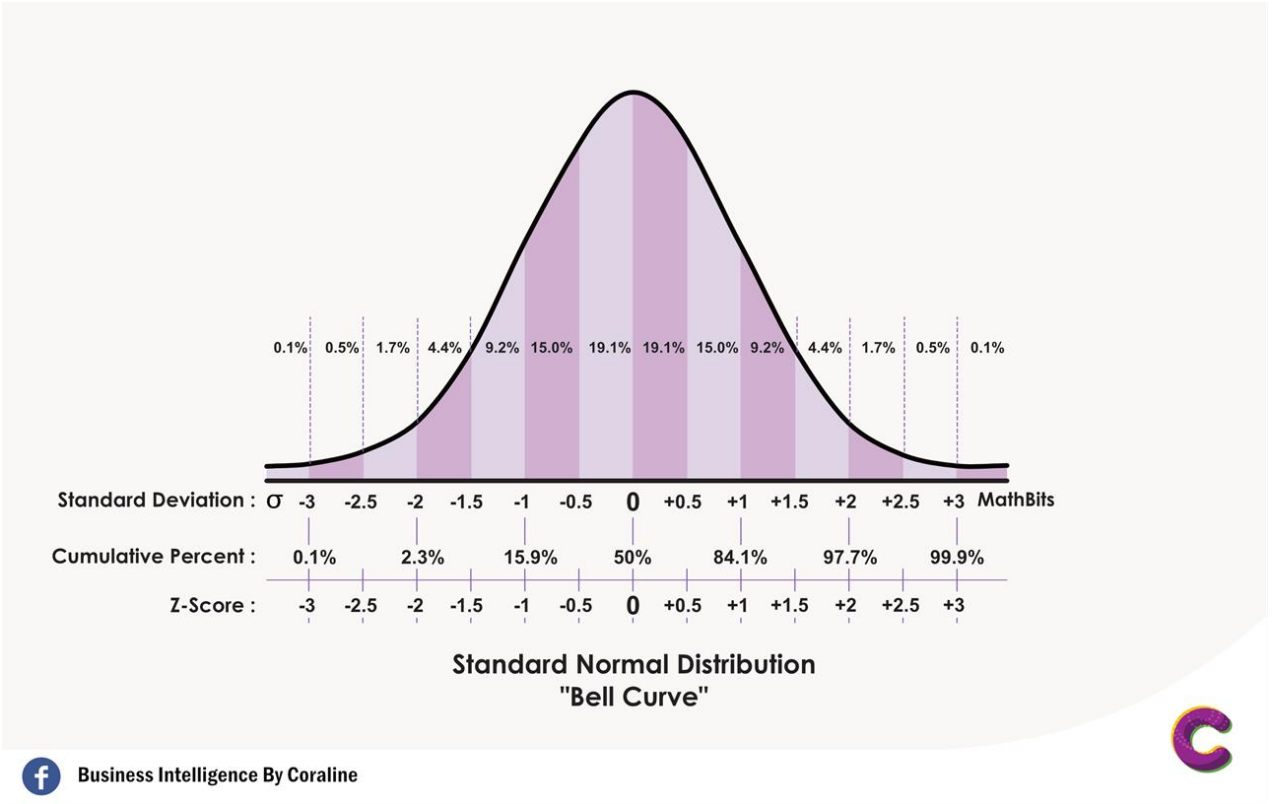


https://coralineltd.medium.com/4-ขั้นตอนการ-clean-data-สำคัญไฉน-why-data-quality-is-a-king-eb924a8e7d7e

3.4 การทำความสะอาดข้อมูล (Data Cleaning)

การทำความสะอาดข้อมูล (Data Cleaning)

วิธีการ Correct data นี้ต้องใช้กลยุทธ์ทางสถิติกันหน่อย ไม่ว่าจะเป็นการหาค่าเฉลี่ย ค่าเบี่ยงเบียนมาตรฐาน หรือ standard deviation หรือแม้กระทั่งการใช้ Clustering algorithm ก็ช่วยได้ หลังจากนั้นก็ต้องมาพิจารณากันต่อว่า ในช่องที่มีข้อมูลผิดพลาดนั้น เราจะมีการลบทิ้งทั้งแถวไปเลย หรือจะแก้ไขข้อมูลที่ผิดนั้นด้วยการแทนที่ด้วยตัวใดตัวหนึ่ง ถ้านึกอะไรไม่ออกก็ให้นึกถึง หลักการสถิติเข้าไว้ก่อน ดังกราฟนี้

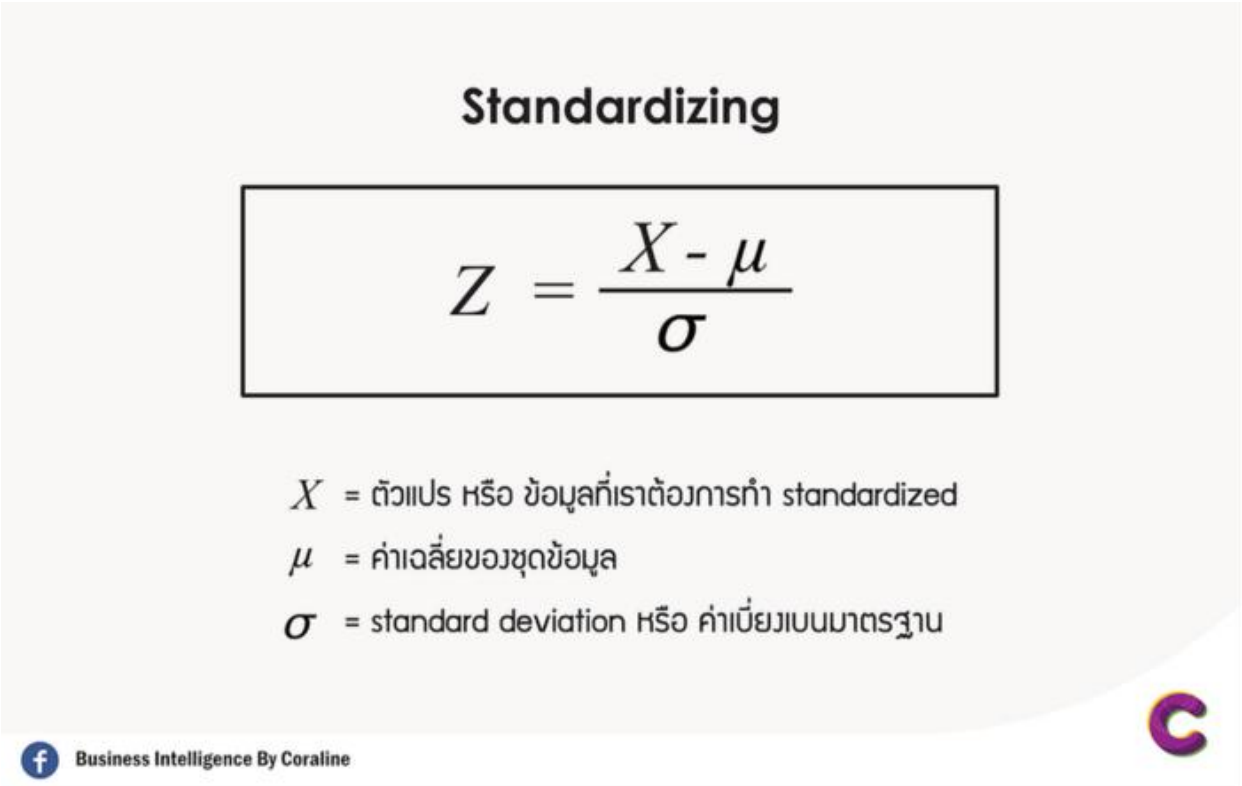


https://coralineltd.medium.com/4-ขั้นตอนการ-clean-data-สำคัญไฉน-why-data-quality-is-a-king-eb924a8e7d7e

3.4 การทำความสะอาดข้อมูล (Data Cleaning)

การทำความสะอาดข้อมูล (Data Cleaning)

3. Standardizing คือ การทำข้อมูลให้เป็นรูปแบบเดียวกัน ตัวอย่างเช่น จังหวัด กรุงเทพฯ ที่มีรูปแบบ กทม. กรุงเทพฯ และ กรุงเทพมหานคร ซึ่งคอมพิวเตอร์ไม่สามารถทราบได้เองว่ามันคือจังหวัดเดียวกัน ส่วนข้อมูลที่เป็นตัวเลขนั้น ในกรณีที่ต้องการแก้ปัญหาเรื่องหน่วย หรือความกว้างของข้อมูลที่ไม่เหมือนกัน สามารถวิธี Standard Normal Distribution ได้ ซึ่งวิธีนี้เป็นการจัดเรียงข้อมูลให้อยู่ในรูป Normalization หรือ ระฆังคว่ำที่เราคุ้นเคยกันดี สูตรการทำ Standardization คือ



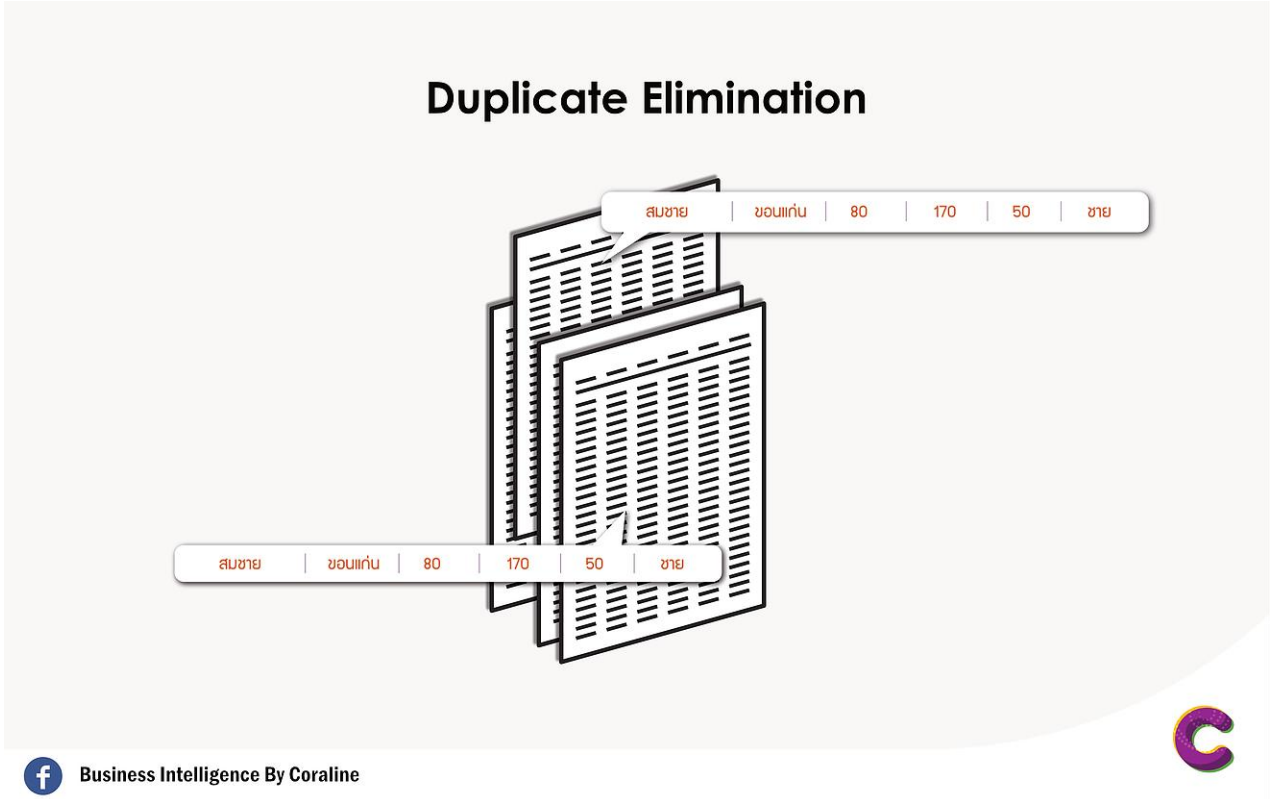
https://coralineltd.medium.com/4-ขั้นตอนการ-clean-data-สำคัญไฉน-why-data-quality-is-a-king-eb924a8e7d7e

3.4 การทำความสะอาดข้อมูล (Data Cleaning)

การทำความสะอาดข้อมูล (Data Cleaning)

4. Duplicate Elimination คือ การลบชุดข้อความซ้ำซ้อนทิ้ง ซึ่งอาจต้องใช้การเขียน Algorithm เพื่อระบุชุดข้อมูลที่ซ้ำซ้อน

ด้วยความยาก และใช้เวลานานกว่าจะได้ข้อมูลที่พร้อมนำไปสร้าง Model ตอนนี้หลายๆ องค์กรที่มีโครงการทำ Big Data System จึงให้ Data Scientist เป็นผู้ออกแบบวิธีการเก็บข้อมูลควบคู่ไปกับ Data Engineer ด้วย ซึ่งจะเห็นได้ว่า การ Clean Data นั้น ต้องอาศัยความรู้ด้านสถิติ และความคิดสร้างสรรค์ในการออกแบบ Algorithm หรือการเขียนโปรแกรมมาประกอบกัน การเป็น Data Scientist ที่ดี ควรให้ความสำคัญกับการ Clean ข้อมูลไม่แพ้การสร้าง Model เพราะหากเมื่อวัตถุดิบที่มีไม่สะอาดสมบูรณ์ ก็ยากนักที่จะได้ผลลัพธ์ออกมาสวยงามแบบได้ตามที่ต้องการ



https://coralineltd.medium.com/4-ขั้นตอนการ-clean-data-สำคัญไฉน-why-data-quality-is-a-king-eb924a8e7d7e

3.4 การทำความสะอาดข้อมูล (Data Cleaning)

การทำความสะอาดข้อมูล (Data Cleaning)

id	name	Gender	Tax	Income		
001	somchai	Female	Y	80000		
002	somying	Female	N	7500.75		
003	Somsri	M	Yes	12000		
004	Somsak	F	N	12,455		
005	Somporn	Male	No	100111		

F , M
Female ,Male

3.4 การทำความสะอาดข้อมูล (Data Cleaning)

บทที่ 3

ตัวอย่าง_01 : ลบค่าข้อมูลที่เป็นค่าว่างหรือค่าผิดปกติ เช่น ค่าว่าง ค่า NaN ค่าอักขระพิเศษ เป็นต้น

Part_01 File : Lec03_data_cleaning01.py

```
import pandas as pd
import numpy as np
# สร้าง DataFrame ตัวอย่าง
data = {'Name': ['John', 'Anna', 'Peter', 'Linda', 'Bob'],
        'Age': [28, 23, np.nan, 45, 32],
        'Salary': [50000, 60000, 75000, np.nan, 80000],
        'Gender': ['Male', 'Female', 'Male', 'Female', np.nan]}
df = pd.DataFrame(data)
print("Original DataFrame:")
print(df)
```

```
Original DataFrame:
    Name   Age   Salary  Gender
0   John  28.0  50000.0    Male
1   Anna  23.0  60000.0  Female
2  Peter   NaN  75000.0    Male
3  Linda  45.0      NaN  Female
4    Bob  32.0  80000.0     NaN

DataFrame after data cleaning:
   Name   Age   Salary  Gender
0  John  28.0  50000.0    Male
1  Anna  23.0  60000.0  Female
```

3.4 การทำความสะอาดข้อมูล (Data Cleaning)

บทที่ 3

ตัวอย่าง_01 : ลบค่าข้อมูลที่เป็นค่าว่างหรือค่าผิดปกติ เช่น ค่าว่าง ค่า NaN ค่าอักขระพิเศษ เป็นต้น

Part_02 File : Lec03_data_cleaning01.py

```
# ตรวจสอบและจัดการข้อมูลที่หายไปหรือข้อมูลผิดพลาด
df_cleaned = df.dropna().copy()  # ทำการคัดลอก DataFrame ที่เป็น subset
# แทนที่ข้อมูลที่หายไปด้วยค่าเฉลี่ย
mean_salary = df_cleaned['Salary'].mean()
df_cleaned['Salary'].fillna(mean_salary, inplace=True)
# แทนที่ข้อมูลที่หายไปด้วยค่า Mode ของคอลัมน์ 'Gender'
mode_gender = df_cleaned['Gender'].mode()[0]
df_cleaned['Gender'].fillna(mode_gender, inplace=True)
print("\nDataFrame after data cleaning:")
print(df_cleaned)
```

```
Original DataFrame:
    Name   Age   Salary  Gender
0   John  28.0  50000.0    Male
1   Anna  23.0  60000.0  Female
2  Peter   NaN  75000.0    Male
3  Linda  45.0      NaN  Female
4    Bob  32.0  80000.0     NaN

DataFrame after data cleaning:
   Name   Age   Salary  Gender
0  John  28.0  50000.0    Male
1  Anna  23.0  60000.0  Female
```

จากข้อความเตือน (`FutureWarning`) ที่แสดงในขณะรันสคริปต์ `Lec03_data_cleaning01.py` บอกว่า:

> คุณกำลังใช้ `.fillna(..., inplace=True)` บน `df_cleaned['column']` ซึ่งอาจจะไม่ได้แก้ไขที่ต้นฉบับจริง แต่แก้ไขที่สำเนา (copy) ของคอลัมน์นั้น

ซึ่งใน **Pandas 3.0 ขึ้นไป** พฤติกรรมนี้จะไม่ทำงานแล้ว ❌

✅ วิธีแก้ที่แนะนำ (ห

ให้ใช้วิธีนี้แทน:

python

```
df_cleaned['Salary']
df_cleaned['Gender']
```

🔍 ตัวอย่างโค้ดเดิม (ที่ทำให้เกิด warning):

python Copy Edit

```
df_cleaned['Salary'].fillna(mean_salary, inplace=True)  # ❌ อาจทำงานกับสำเนา
df_cleaned['Gender'].fillna(mode_gender, inplace=True)  # ❌
```

🔁 แก้เป็นแบบนี้:

python Copy Edit

```
df_cleaned['Salary'] = df_cleaned['Salary'].fillna(mean_salary)  # ✅ ปลอดภัย
df_cleaned['Gender'] = df_cleaned['Gender'].fillna(mode_gender)  # ✅
```

```
'Gender': ['Male', 'Female', 'Male', 'Female', np.nan]}
df = pd.DataFrame(data)
print("Original DataFrame:")
print(df)
# ตรวจสอบและจัดการข้อมูลที่หายไปหรือข้อมูลผิดพลาด
df_cleaned = df.dropna().copy() # ทำการคัดลอก DataFrame ที่เป็น subset
# แทนที่ข้อมูลที่หายไปด้วยค่าเฉลี่ย


mean_salary = df_cleaned['Salary'].mean()
df_cleaned['Salary'] = df_cleaned['Salary'].fillna(mean_salary)  # ✅ ปลอดภัย

mode_gender = df_cleaned['Gender'].mode()[0]
df_cleaned['Gender'] = df_cleaned['Gender'].fillna(mode_gender)  # ✅

print("\nDataFrame after data cleaning:")
print(df_cleaned)
```

```
16  mean_salary = df_cleaned['Salary'].mean()
17  df_cleaned['Salary'] = df_cleaned['Salary'].fillna(mean_salary)  # ✅ ปลอดภัย
18
19  mode_gender = df_cleaned['Gender'].mode()[0]
20  df_cleaned['Gender'] = df_cleaned['Gender'].fillna(mode_gender)  # ✅
21
22  print("\nDataFrame after data cleaning:")
23  print(df_cleaned)
24
```

```python
import pandas as pd
import numpy as np
# สร้าง DataFrame ตัวอย่าง
data = {'Name': ['John', 'Anna', 'Peter', 'Linda', 'Bob'],
'Age': [28, 23, np.nan, 45, 32],
'Salary': [50000, 60000, 75000, np.nan, 80000],
'Gender': ['Male', 'Female', 'Male', 'Female', np.nan]}
df = pd.DataFrame(data)
print("Original DataFrame:")
print(df)
# ตรวจสอบและจัดการข้อมูลที่หายไปหรือข้อมูลผิดพลาด
df_cleaned = df.dropna().copy() # ทำการคัดลอก DataFrame ที่เป็น subset
# แทนที่ข้อมูลที่หายไปด้วยค่าเฉลี่ย


mean_salary = df_cleaned['Salary'].mean()
df_cleaned['Salary'] = df_cleaned['Salary'].fillna(mean_salary)  # ✅

mode_gender = df_cleaned['Gender'].mode()[0]
df_cleaned['Gender'] = df_cleaned['Gender'].fillna(mode_gender)  #

print("\nDataFrame after data cleaning:")
print(df_cleaned)
```

ชื่อไฟล์ File : Lec03_data_cleaning01.py

```
PS D:\Data_Mining_Project> & C:/Users/ASUS/AppData/Local/Micr
Original DataFrame:
    Name   Age   Salary  Gender
0   John  28.0  50000.0    Male
1   Anna  23.0  60000.0  Female
2  Peter   NaN  75000.0    Male
3  Linda  45.0      NaN  Female
4    Bob  32.0  80000.0     NaN

DataFrame after data cleaning:
   Name   Age   Salary  Gender
0  John  28.0  50000.0    Male
1  Anna  23.0  60000.0  Female
PS D:\Data_Mining_Project>
```

3.4 การทำความสะอาดข้อมูล (Data Cleaning)

บทที่ 3

ตัวอย่าง_01 : ลบค่าข้อมูลที่เป็นค่าว่างหรือค่าผิดปกติ เช่น ค่าว่าง ค่า NaN ค่าอักขระพิเศษ เป็นต้น

ในตัวอย่างนี้:

อภิปรายผล File : Lec03_data_cleaning08.py

คำสั่ง dropna() ถูกใช้เพื่อลบแถวที่มีค่าหายไป (NaN).

ใช้ astype(int) เพื่อแปลงชนิดข้อมูลของคอลัมน์ 'Age' เป็น integer.

ใช้ mean() เพื่อคำนวณค่าเฉลี่ยของคอลัมน์ 'Salary' และ fillna() เพื่อแทนที่ค่าที่หายไปด้วยค่าเฉลี่ย.

ใช้ mode()[0] เพื่อหาค่า Mode ของคอลัมน์ 'Gender' และ fillna() เพื่อแทนที่ค่าที่หายไปด้วย Mode.

ผลลัพธ์ที่พิมพ์ออกมาจะแสดง DataFrame หลังจากการทำความสะอาดข้อมูล. การทำความสะอาดข้อมูลเป็นขั้นตอนที่สำคัญในการปรับปรุงคุณภาพข้อมูลและให้ข้อมูลพร้อมใช้งานในการวิเคราะห์หรือการเรียนรู้ของเครื่อง.

3.4 การทำความสะอาดข้อมูล (Data Cleaning)

3.4.1 การจัดการค่าที่ขาดหาย (Missing Value):

วิธีการแทนที่ค่าข้อมูลที่หายไป (Replace missing value) โดยมีรายละเอียดดังนี้

- **ข้อมูลอาจจะขาดหายไป เนื่องจาก**
- **ไม่มีข้อมูล**
- **กรอกข้อมูลผิดพลาด**
- **การแทนที่ข้อมูลที่ขาดหายไป**
  - **แทนที่ด้วยค่า เช่น N/A หรือ none หรือ null**
  - **แทนที่ด้วยค่าเฉลี่ย ในกรณีที่แอตทริบิวต์เป็นตัวเลข (numeric)**
  - **แทนที่ด้วยค่ามากที่สุด**
- **ในกรณีที่แอตทริบิวต์เป็นตัวเลข (numeric)**
  - **แทนที่ด้วยค่าเฉลี่ย (mean/average)**
  - **ในกรณีที่แอตทริบิวต์เป็นตัวเลข (numeric) การแทนที่ค่าข้อมูลที่หายไปด้วยวิธีการเหล่านี้ จะช่วยให้ชุดข้อมูลมีความสมบูรณ์มากขึ้น และสามารถนำไปวิเคราะห์หรือประมวลผลได้อย่างมีประสิทธิภาพมากขึ้น**

3.4 การทำความสะอาดข้อมูล (Data Cleaning)

3.4.2 ค่าผิดปกติ (Outlier)

Outliers

เป็นค่าคาดเคลื่อนที่มีลักษณะแตกต่างจากกลุ่มหลักค่อนข้างมาก ซึ่งมีรายละเอียดดังนี้

- **ค่าคาดเคลื่อน (Outliers) เป็นข้อมูลที่ผิดแผกออกจากกลุ่มหรือดิดแตกต่างจากข้อมูลค่าอื่นๆ**
- **ตัวอย่างของค่าคาดเคลื่อนได้แก่ IQ ของเด็กได้ 195, น้ำหนักของคน 220 กิโลกรัม, ความสูงของคน 210 ซม.**
- **ซึ่งค่าคาดเคลื่อนนี้อาจเกิดขึ้นได้จากสาเหตุหลัก 2 ประการคือ1) การจดบันทึกหรือเก็บข้อมูลผิดพลาดเคลื่อน หรือ2) กลุ่มตัวอย่างที่เก็บรวบรวมข้อมูลมา มีความแตกต่างไปจากกลุ่มวิจัย**

3.4 การทำความสะอาดข้อมูล (Data Cleaning)

บทที่ 3

3.4.2 ค่าผิดปกติ (Outlier)

การจัดการข้อมูล Outlier (ข้อมูลที่อยู่นอกเหนือจากแบบแผนที่มีค่ามากหรือน้อยเกินไป) เป็นกระบวนการที่สำคัญในการทำความสะอาดข้อมูลเพื่อให้ข้อมูลมีความถูกต้องและเป็นประโยชน์สูงสุด ต่อไปนี้คือตัวอย่างการจัดการข้อมูล Outlier ใน Python โดยใช้ไลบรารี pandas:

File : Lec03_data_cleaning_outlier09.py

```
import pandas as pd
# สร้าง DataFrame ตัวอย่าง
data = {'คะแนน': [85, 88, 92, 78, 94, 120, 88, 90, 89, 91]}
df = pd.DataFrame(data)
# แสดง DataFrame ก่อนการจัดการ Outlier
print("DataFrame ก่อนการจัดการ Outlier:")
print(df)
# ตรวจสอบและจัดการ Outlier โดยใช้ค่าส่วนกลาง (median)
median_value = df['คะแนน'].median()
df['คะแนน'] = df['คะแนน'].apply(lambda x: median_value if x > 100 else x)
# แสดง DataFrame หลังการจัดการ Outlier
print("\nDataFrame หลังการจัดการ Outlier:")
print(df)
```

การหาค่ามัธยฐาน (Median)

ทบทวน

การหาค่า **Median (มัธยฐาน)** คือการหาค่ากลางของชุดข้อมูลที่เรียงลำดับจากน้อยไปมาก โดยมีขั้นตอนดังนี้:

ข้อมูล:

[85, 88, 92, 78, 94, 120, 88, 90, 89, 91]

ขั้นตอน:

1. **เรียงลำดับข้อมูลจากน้อยไปมาก**:
   [78, 85, 88, 88, 89, 90, 91, 92, 94, 120]
2. **นับจำนวนข้อมูล**:
   มีข้อมูลทั้งหมด 10 ค่า (เลขคู่)
3. **หาค่ากลาง**:
   - เนื่องจากจำนวนข้อมูลเป็นเลขคู่ ให้หาค่าเฉลี่ยของค่ากลาง 2 ค่า คือ **อันดับที่ 5** และ **อันดับที่ 6**
   - ค่าในอันดับที่ 5 = 89
   - ค่าในอันดับที่ 6 = 90
4. **คำนวณค่าเฉลี่ยของค่ากลาง 2 ค่า**:

$$\text{Median} = \frac{89 + 90}{2} = \frac{179}{2} = 89.5$$

คำตอบ:

ค่า **Median** ของชุดข้อมูลคือ **89.5**

3.4 การทำความสะอาดข้อมูล (Data Cleaning)

บทที่ 3

3.4.2 ค่าผิดปกติ (Outlier)

การจัดการข้อมูล Outlier (ข้อมูลที่อยู่นอกเหนือจากแบบแผนที่มีค่ามากหรือน้อยเกินไป) เป็นกระบวนการที่สำคัญในการทำความสะอาดข้อมูลเพื่อให้ข้อมูลมีความถูกต้องและเป็นประโยชน์สูงสุด. ต่อไปนี้คือตัวอย่างการจัดการข้อมูล Outlier ใน Python โดยใช้ไลบรารี pandas:

File : Lec03_data_cleaning_outlier09.py

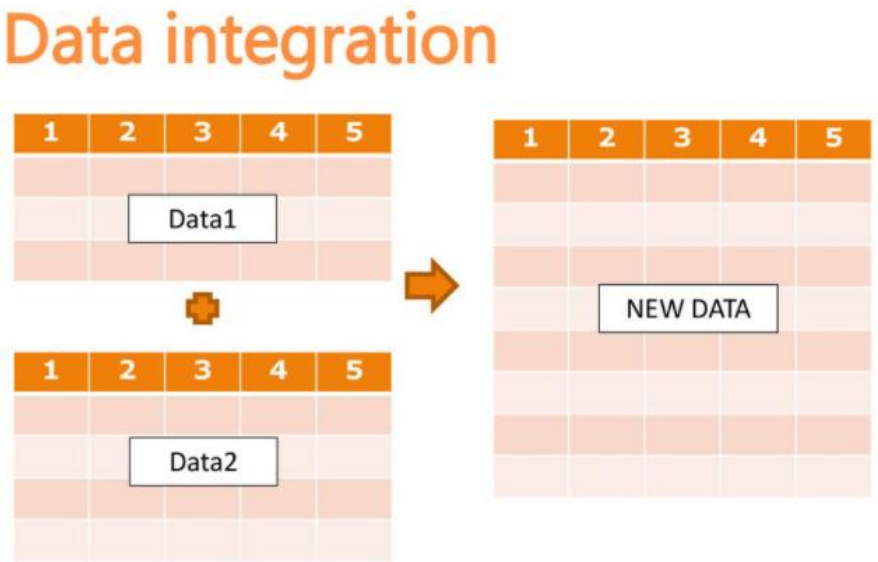
Replace_value_Outlier.py > ...

```python
import pandas as pd
# สร้าง DataFrame ตัวอย่าง
data = {'คะแนน': [85, 88, 92, 78, 94, 120, 88, 90, 89, 91]}
df = pd.DataFrame(data)
# แสดง DataFrame ก่อนการจัดการ Outlier
print("DataFrame ก่อนการจัดการ Outlier:")
print(df)
# ตรวจสอบและจัดการ Outlier โดยใช้ค่าส่วนกลาง (median)
median_value = df['คะแนน'].median()
df['คะแนน'] = df['คะแนน'].apply(lambda x: median_value if x > 100 else x)
# แสดง DataFrame หลังการจัดการ Outlier
print("\nDataFrame หลังการจัดการ Outlier:")
print(df)
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL

5   89.5
6   88.0
7   90.0
8   89.0
9   91.0
PS C:\AppServ\www\Python_DataMining>
```

ทบทวน

ความหมายของ Data integration

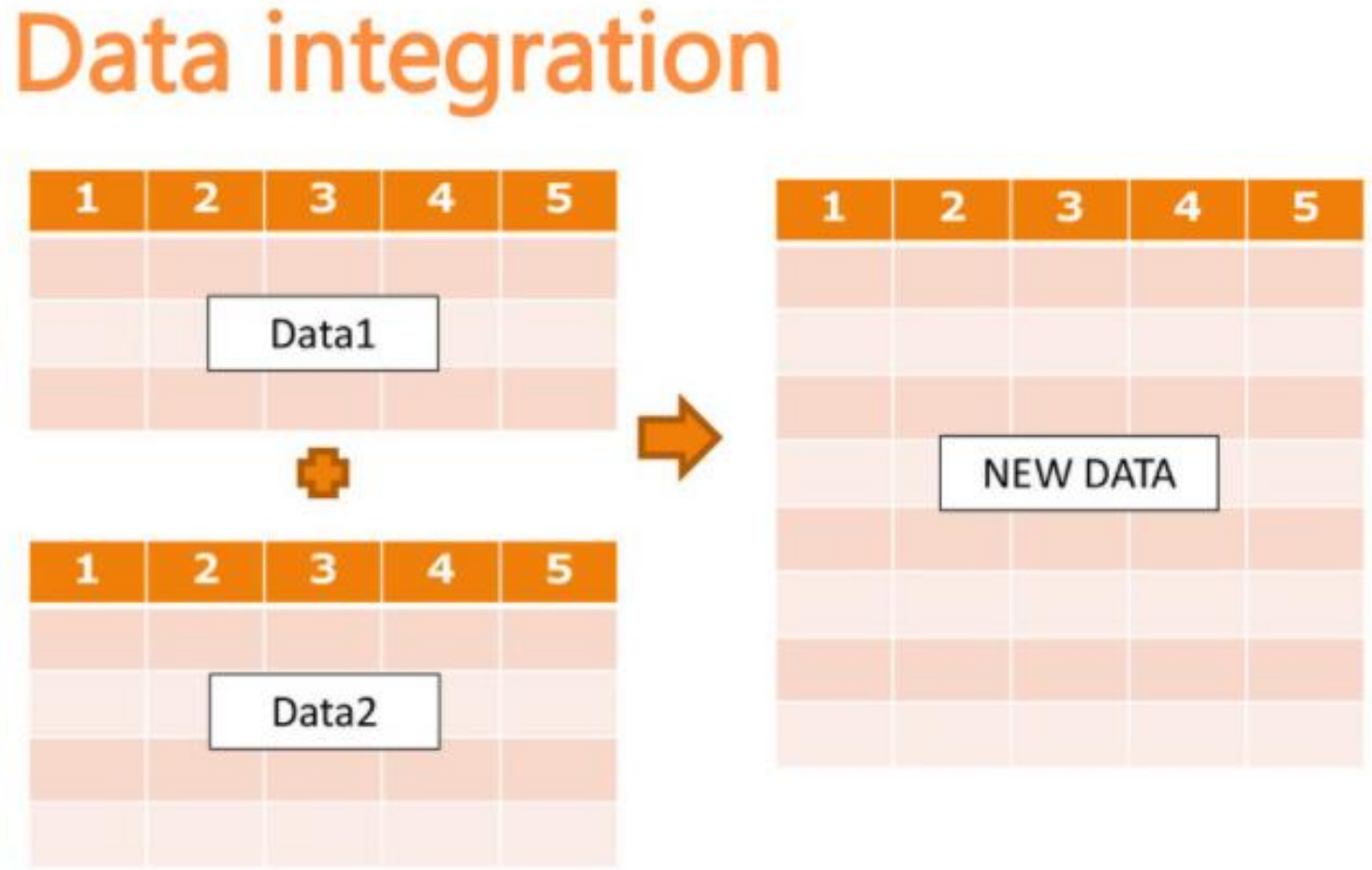


การรวมข้อมูล (Data Integration) หมายถึง กระบวนการรวมข้อมูลจากหลายแหล่ง (Data Sources) ให้กลายเป็นชุดข้อมูลเดียวที่มีความสอดคล้องและพร้อมสำหรับการนำไปใช้งานหรือวิเคราะห์ต่อไป การรวมข้อมูลช่วยให้ข้อมูลจากแหล่งที่มาที่แตกต่างกันสามารถเชื่อมโยงกันได้และสร้างมุมมองแบบรวม (Unified View) เพื่อใช้ในงานวิเคราะห์ เช่น การทำเหมืองข้อมูล การสร้างโมเดล หรือการรายงานผล

ต่อไปนี้คือตัวอย่างการทำ Data Integration ใน Python โดยใช้ไลบรารี pandas

ทบทวน

Data Integration คืออะไร

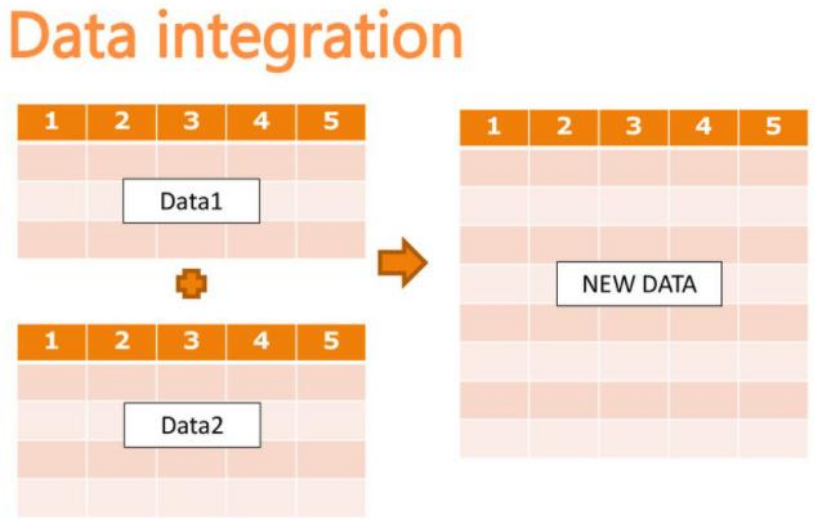


ไฟล์ .csv หรือ .xlsx

ไฟล์ .mdb หรือ SQL

ทบทวน

ตัวอย่างของ Data Integration



ตัวอย่างของ Data Integration

- การรวมข้อมูลลูกค้า: ข้อมูลลูกค้าอาจมาจากระบบ CRM (Customer Relationship Management) และระบบการขาย (Sales System) ที่ต่างกัน การรวมข้อมูลจะช่วยสร้างมุมมองแบบรวมเกี่ยวกับลูกค้า เช่น ประวัติการซื้อขายและข้อมูลการติดต่อ
- การรวมข้อมูลทางธุรกิจ: ข้อมูลจากสาขาต่าง ๆ ของบริษัทในหลายประเทศถูกนำมารวมในคลังข้อมูล (Data Warehouse) เพื่อช่วยวิเคราะห์ภาพรวมของธุรกิจ

ทบทวน

องค์ประกอบของ Data Integration

1.Data Sources
- แหล่งข้อมูลอาจมาจากฐานข้อมูลหลายแบบ เช่น Relational Databases, NoSQL Databases, ไฟล์ Excel, APIs, หรือแหล่งข้อมูลอื่น ๆ

2.การเชื่อมโยงข้อมูล (Data Linking)
- การเชื่อมโยงระหว่างข้อมูลจากแหล่งต่าง ๆ โดยใช้ตัวระบุที่เหมือนกัน (Identifiers) หรือคีย์ร่วมกัน เช่น รหัสลูกค้า, ชื่อผลิตภัณฑ์

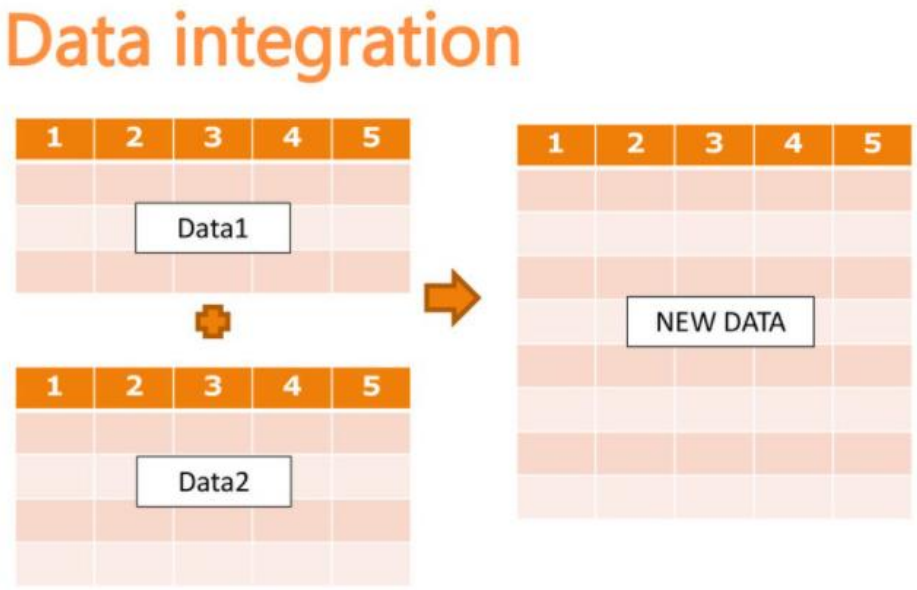
3.การรวมข้อมูล (Data Merging)
- การนำข้อมูลที่เชื่อมโยงมาอยู่ในโครงสร้างเดียวกัน เช่น การรวมตารางหรือไฟล์ข้อมูล

4.การแก้ไขความขัดแย้งของข้อมูล (Data Conflict Resolution)
- การจัดการข้อมูลที่มีความขัดแย้ง เช่น ความไม่สอดคล้องของรูปแบบ (Formats), หน่วย (Units), หรือค่าซ้ำซ้อน (Duplicates)

5.Data Transformation
- การแปลงข้อมูลให้มีรูปแบบเดียวกัน เช่น การปรับหน่วย, การกำหนดค่าเริ่มต้นสำหรับข้อมูลที่ขาดหาย, หรือการจัดโครงสร้างใหม่

ตัวอย่างการทำ Data integration

ทบทวน



เริ่มต้นด้วยสร้างสอง DataFrame ที่จะนำเข้ามารวมกัน:

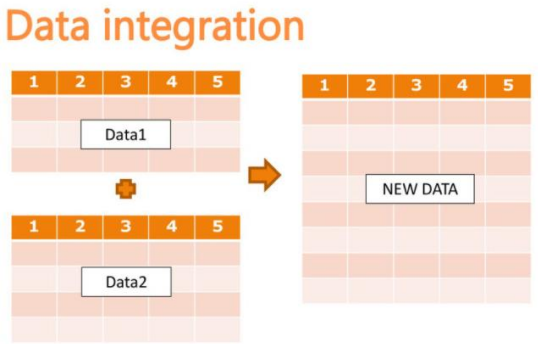
Dm_prepareData10.py

Part01

```
import pandas as pd
# สร้าง DataFrame ตัวอย่างที่ 1
data1 = {'ID': [1, 2, 3, 4],
        'ชื่อ': ['John', 'Alice', 'Bob', 'Charlie'],
        'เงินเดือน': [50000, 60000, 45000, 70000]}
df1 = pd.DataFrame(data1)
# สร้าง DataFrame ตัวอย่างที่ 2
data2 = {'ID': [3, 4, 5, 6],
        'อายุ': [28, 35, 22, 40],
        'ที่อยู่': ['Street A', 'Street B', 'Street C', 'Street D']}
df2 = pd.DataFrame(data2)
# แสดง DataFrame ทั้ง 2
print("DataFrame ที่ 1:")
print(df1)
print("\nDataFrame ที่ 2:")
print(df2)
```

ตัวอย่างการทำ Data integration

ทบทวน



เมื่อมี DataFrame ทั้งสอง, เราสามารถรวมข้อมูลด้วยคำสั่ง merge() โดยให้ระบุคอลัมน์ที่ต้องการให้เป็นหลักการรวมข้อมูล:

```
# รวม DataFrame ทั้ง 2 ด้วยคอลัมน์ 'ID'
merged_df = pd.merge(df1, df2, on='ID', how='inner')

# แสดง DataFrame หลังการรวม
print("\nDataFrame หลังการรวม:")
print(merged_df)
```

Part02

Dm_prepareData10.py

```
PS C:\AppServ\www\Python_DataMining> & C:/Users/Us
DataFrame ที่ 1:
   ID     ชื่อ  เงินเดือน
0   1    John      50000
1   2   Alice      60000
2   3     Bob      45000
3   4 Charlie      70000

DataFrame ที่ 2:
   ID  อายุ   ที่อยู่
0   3    28  Street A
1   4    35  Street B
2   5    22  Street C
3   6    40  Street D

DataFrame หลังการรวม:
   ID     ชื่อ  เงินเดือน  อายุ   ที่อยู่
0   3     Bob      45000    28  Street A
1   4 Charlie      70000    35  Street B
PS C:\AppServ\www\Python_DataMining>
```

ทบทวน

ตัวอย่างการทำ Data integration

ในตัวอย่างนี้:

- **เราใช้ pd.merge() เพื่อรวม DataFrame df1 และ df2 ด้วยคอลัมน์ 'ID'.**
- **on='ID' คือการระบุให้คอลัมน์ 'ID' เป็นคอลัมน์หลักในการรวม.**
- **how='inner' คือการระบุว่าจะให้เลือกข้อมูลที่มีค่า 'ID' ที่เหมือนกันทั้งหมด.**

ผลลัพธ์จะเป็น DataFrame ที่รวมข้อมูลจากทั้งสอง DataFrame ตามคอลัมน์ 'ID'. การทำ Data Integration ช่วยให้เราสามารถรวมข้อมูลจากแหล่งต่าง ๆ เข้าด้วยกันเพื่อให้ได้ข้อมูลที่ครอบคลุมมากขึ้น.

```
PS C:\AppServ\www\Python_DataMining> & C:/Users/Us
DataFrame ที่ 1:
   ID     ชื่อ  เงินเดือน
0   1     John      50000
1   2    Alice      60000
2   3      Bob      45000
3   4  Charlie      70000

DataFrame ที่ 2:
   ID  อายุ   ที่อยู่
0   3    28  Street A
1   4    35  Street B
2   5    22  Street C
3   6    40  Street D

DataFrame หลังการรวม:
   ID     ชื่อ  เงินเดือน  อายุ   ที่อยู่
0   3      Bob      45000    28  Street A
1   4  Charlie      70000    35  Street B
PS C:\AppServ\www\Python_DataMining>
```

ทบทวน

Discretization คืออะไร

Discretization หรือ การทำข้อมูลให้อยู่ในรูปแบบกลุ่มช่วงค่า หมายถึงกระบวนการแปลงข้อมูลเชิงปริมาณ (Numerical Data) หรือข้อมูลแบบต่อเนื่อง (Continuous Data) ให้กลายเป็นข้อมูลที่มีลักษณะเชิงคุณภาพ (Categorical Data) โดยแบ่งข้อมูลออกเป็นช่วง (Intervals หรือ Bins) และกำหนดป้ายชื่อหรือกลุ่มให้แต่ละช่วงนั้น

ตัวอย่าง
หากมีข้อมูลอายุของคนในรูปแบบตัวเลข เช่น 18, 22, 35, 45, 67
การทำ Discretization อาจแบ่งอายุออกเป็นกลุ่ม:
- **0–20 ปี (วัยเด็ก)**
- **21–40 ปี (วัยผู้ใหญ่ต้น)**
- **41–60 ปี (วัยกลางคน)**
- **61+ ปี (วัยสูงอายุ)**

ตัวเลข 18 จะถูกจัดอยู่ในกลุ่ม "วัยเด็ก" และตัวเลข 45 จะถูกจัดอยู่ในกลุ่ม "วัยกลางคน"

ทบทวน

ข้อดี/ข้อเสียของ Discretization

ข้อดีของ Discretization

- **ลดความซับซ้อน: ทำให้การวิเคราะห์ข้อมูลเข้าใจง่ายขึ้น**
- **เพิ่มความสามารถในการตีความ: ข้อมูลแบบกลุ่มช่วยให้เข้าใจความหมายในเชิงธุรกิจได้ดีกว่า**
- **ปรับปรุงประสิทธิภาพของโมเดล: อัลกอริทึมบางชนิดทำงานได้ดีขึ้นกับข้อมูลเชิงกลุ่ม**

ข้อเสีย

- **การสูญเสียรายละเอียด: การแปลงข้อมูลต่อเนื่องอาจทำให้สูญเสียความแตกต่างของค่าจริง**
- **อาจมีอคติจากการกำหนดช่วง: การเลือกช่วงที่ไม่เหมาะสมอาจทำให้ผลลัพธ์ผิดเพี้ยน**

ทบทวน

Discretization คืออะไร

การประยุกต์ใช้

- **การวิเคราะห์ข้อมูล: ทำให้ข้อมูลง่ายต่อการตีความ เช่น การวิเคราะห์ตามกลุ่มอายุ**
- **การทำเหมืองข้อมูล: ลดความซับซ้อนของข้อมูล และช่วยปรับปรุงผลลัพธ์ของอัลกอริทึม เช่น Decision Trees**
- **การสร้างโมเดล: ช่วยลดปัญหาข้อมูลเชิงปริมาณที่มีขนาดใหญ่เกินไป**

สรุป:

Discretization เป็นกระบวนการที่มีประโยชน์ในการปรับข้อมูลให้เข้าใจง่ายขึ้น และช่วยในการสร้างโมเดลวิเคราะห์ข้อมูล แต่ควรเลือกวิธีการแบ่งช่วงที่เหมาะสมเพื่อหลีกเลี่ยงการสูญเสียข้อมูลสำคัญ.

ทบทวน

แบ่งตามเงื่อนไขที่กำหนด

แบ่งตามเงื่อนไขที่กำหนด

No	GPA
1	2.50
2	2.75
3	3.00
4	3.00
5	3.10
6	3.25
7	3.40
8	3.50
9	3.75

เงื่อนไข

- <= 3.00 เกรดน้อย
- >3.00 และ <=3.50 เกรดปานกลาง
- >3.50 และ <=3.75 เกรดดี
- >3.75 เกรดดีมาก

การทำ Data Discretization เป็นกระบวนการแปลงข้อมูลที่เป็นตัวเลข (continuous data) เป็นข้อมูลที่เป็นกลุ่มหรือช่วง (discrete data). กระบวนการนี้มักถูกใช้เพื่อลดความซับซ้อนของข้อมูลหรือเพื่อเตรียมข้อมูลให้เหมาะสมสำหรับการนำเข้าไปใช้ในแบบจำลองทางสถิติหรือการวิเคราะห์. ต่อไปนี้คือตัวอย่างการทำ Data Discretization ใน Python โดยใช้ไลบรารี pandas:

ทบทวน

แบ่งตามเงื่อนไขที่กำหนด

```python
import pandas as pd
# สร้าง DataFrame ตัวอย่าง
data = {'ชื่อ': ['John', 'Alice', 'Bob', 'Charlie', 'David'],
        'อายุ': [25, 28, 35, 45, 32],
        'เงินเดือน': [50000, 60000, 70000, 80000, 55000]}
df = pd.DataFrame(data)
# แสดง DataFrame ก่อนการทำ Data Discretization
print("DataFrame ก่อนการทำ Data Discretization:")
print(df)
# ใช้ cut() เพื่อแบ่งช่วงของคอลัมน์ 'อายุ' และ 'เงินเดือน'
age_bins = [20, 30, 40, 50]
salary_bins = [50000, 60000, 70000, 80000, 90000]
df['กลุ่มอายุ'] = pd.cut(df['อายุ'], bins=age_bins, labels=['20-30', '30-40', '40-50'])
df['กลุ่มเงินเดือน'] = pd.cut(df['เงินเดือน'], bins=salary_bins, labels=['50k-60k', '60k-70k', '70k-80k', '80k-90k'])

# แสดง DataFrame หลังการทำ Data Discretization
print("\nDataFrame หลังการทำ Data Discretization:")
print(df[['ชื่อ', 'อายุ', 'เงินเดือน', 'กลุ่มอายุ', 'กลุ่มเงินเดือน']])
```

```
PS C:\AppServ\www\Python_DataMining> & C:/Users/User/AppDat
DataFrame ก่อนการทำ Data Discretization:
      ชื่อ  อายุ  เงินเดือน
0    John    25      50000
1   Alice    28      60000
2     Bob    35      70000
3 Charlie    45      80000
4   David    32      55000

DataFrame หลังการทำ Data Discretization:
      ชื่อ  อายุ  เงินเดือน กลุ่มอายุ กลุ่มเงินเดือน
0    John    25      50000    20-30          NaN
1   Alice    28      60000    20-30       50k-60k
2     Bob    35      70000    30-40       60k-70k
3 Charlie    45      80000    40-50       70k-80k
4   David    32      55000    30-40       50k-60k
PS C:\AppServ\www\Python_DataMining>
```

Dm_prepareData11.py

ทบทวน

File : Data_Discretization_2

Daa_Dicretization.py > ...

```python
import pandas as pd
# สร้าง DataFrame ตัวอย่าง
data = {'ชื่อ': ['John', 'Alice', 'Bob', 'Charlie', 'David'],
        'อายุ': [25, 28, 35, 45, 32],
        'เงินเดือน': [50000, 60000, 70000, 80000, 55000]}
df = pd.DataFrame(data)
# แสดง DataFrame ก่อนการทำ Data Discretization
print("DataFrame ก่อนการทำ Data Discretization:")
print(df)
# ใช้ cut() เพื่อแบ่งช่วงของคอลัมน์ 'อายุ' และ 'เงินเดือน'
age_bins = [20, 30, 40, 50]
salary_bins = [50000, 60000, 70000, 80000, 90000]
df['กลุ่มอายุ'] = pd.cut(df['อายุ'], bins=age_bins, labels=['20-30', '30-40', '40-50'])
df['กลุ่มเงินเดือน'] = pd.cut(df['เงินเดือน'], bins=salary_bins, labels=['50k-60k',
                        '60k-70k', '70k-80k', '80k-90k'])
# แสดง DataFrame หลังการทำ Data Discretization
print("\nDataFrame หลังการทำ Data Discretization:")
print(df[['ชื่อ', 'อายุ', 'เงินเดือน', 'กลุ่มอายุ', 'กลุ่มเงินเดือน']])
```

ทบทวน

แบ่งตามเงื่อนไขที่กำหนด

```
PS C:\AppServ\www\Python_DataMining> & C:/Users/User/AppDat
DataFrame ก่อนการทำ Data Discretization:
      ชื่อ  อายุ  เงินเดือน
0    John    25      50000
1   Alice    28      60000
2     Bob    35      70000
3 Charlie    45      80000
4   David    32      55000

DataFrame หลังการทำ Data Discretization:
      ชื่อ  อายุ  เงินเดือน กลุ่มอายุ กลุ่มเงินเดือน
0    John    25      50000    20-30           NaN
1   Alice    28      60000    20-30        50k-60k
2     Bob    35      70000    30-40        60k-70k
3 Charlie    45      80000    40-50        70k-80k
4   David    32      55000    30-40        50k-60k
PS C:\AppServ\www\Python_DataMining>
```

ในตัวอย่างนี้:

เราใช้ pd.cut() เพื่อแบ่งช่วงของคอลัมน์ 'อายุ' และ 'เงินเดือน' โดยกำหนดขอบเขตของช่วง (bins) ที่เราต้องการ.

สร้างคอลัมน์ใหม่ 'กลุ่มอายุ' และ 'กลุ่มเงินเดือน' เพื่อเก็บข้อมูลที่ถูกแบ่งช่วง.

ผลลัพธ์จะเป็น DataFrame ที่มีข้อมูลที่ถูกแบ่งช่วงตามกลุ่มที่กำหนด. การทำ Data Discretization ช่วยให้เราสามารถทำนายหรือวิเคราะห์ข้อมูลที่มีลักษณะที่เป็นกลุ่มได้ง่ายขึ้น.

ทบทวน

การแบ่งช่วงของข้อมูลที่เท่ากัน

No	GPA
1	2.50
2	2.75
3	3.00
4	3.00
5	3.10
6	3.25
7	3.40
8	3.50
9	3.75

แบ่งเป็น 4 ช่วง คำนวณจาก

$$\text{ช่วงข้อมูล} = \frac{\text{ค่ามากสุด–ค่าน้อ}}{\text{จำนวนช่วง}}$$

$$= (3.75-2.50)/4$$

$$= 0.313$$

ช่วง	GPA
1	<= 2.81
2	(2.81-3.12)
3	(3.12-3.43)
4	>3.43

ทบทวน

การแบ่งช่วงของข้อมูลที่เท่ากัน

ลำดับ	ชื่อ-นามสกุล	เงินเดือน
1	A	15000
2	B	21000
3	C	23110
4	D	24000
5	E	30100
6	F	28100
7	G	29100
8	H	21000
9	I	22000
10	J	34000

จงแบ่งช่วงของข้อมูลที่เท่ากัน เป็น 5 ช่วง

ทบทวน

การแบ่งช่วงของข้อมูลที่เท่ากัน

Normalization

เป็นการปรับเปลี่ยนช่วงข้อมูลให้มี scale เดียวกันเพื่อจะเปรียบเทียบกัน

$$\text{Z- transformation} = \frac{x - \text{ค่าเฉลี่ย}}{\text{ส่วนเบี่ยงเบนมาตรฐาน}}$$

3.5 การแปลงข้อมูล (Data Transformation)

3.5.1 การแปลงประเภทข้อมูล:

- การแปลงข้อมูลตัวเลขเป็นวันที่ (datetime)
- การแปลงคอลัมน์ที่เป็นข้อความเป็นตัวเลข (categorical to numerical)

3.5.2 การสร้างฟีเจอร์ใหม่ (Feature Engineering):

- การรวมฟีเจอร์หรือการคำนวณฟีเจอร์ใหม่

3.5.3 การปรับขนาดข้อมูล (Scaling):

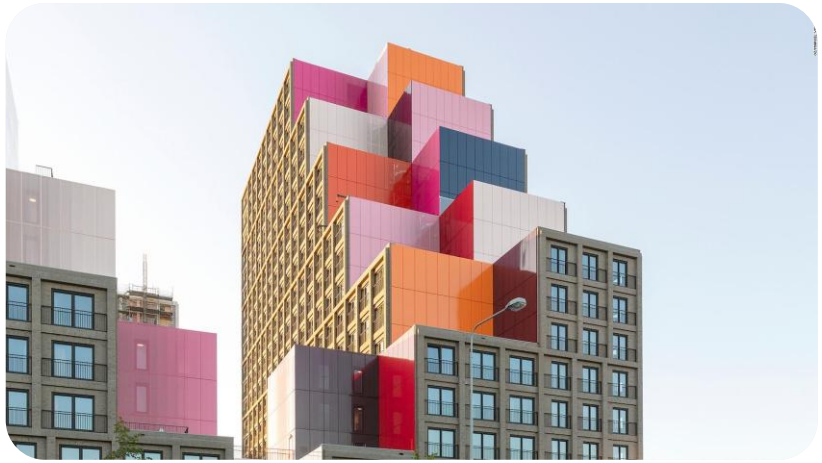
- การปกติค่า (Normalization)
- การปรับสเกล (Standardization)

3.5 การแปลงข้อมูล (Data Transformation)

บทที่ 3

3.5.1 การแปลงประเภทข้อมูล

- เป็นขั้นตอนการปรับเปลี่ยนรูปแบบข้อมูลและแปลงข้อมูลเพื่อให้ข้อมูลเหมาะสมกับเทคนิคเหมืองข้อมูลที่นำมาวิเคราะห์ได้
- การแปลงข้อมูล (Data Transformation) เป็นกระบวนการเปลี่ยนรูปแบบข้อมูลจากรูปแบบหนึ่งไปเป็นอีกรูปแบบหนึ่ง เพื่อให้ข้อมูลอยู่ในภาพที่พร้อมสำหรับการวิเคราะห์หรือนำไปใช้ต่อไป การแปลงข้อมูลสามารถทำได้หลายวิธี ขึ้นอยู่กับรูปแบบของข้อมูลเดิมและรูปแบบที่ต้องการแปลง



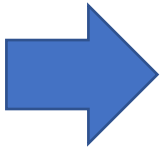
ID	สีแดง	สีเขียว	สีน้ำเงิน
1	89	102	210
2			
3			

3.5 การแปลงข้อมูล (Data Transformation)

3.5.1 การแปลงประเภทข้อมูล

- เป็นขั้นตอนการปรับเปลี่ยนรูปแบบข้อมูลและแปลงข้อมูลเพื่อให้ข้อมูลเหมาะสมกับเทคนิคเหมืองข้อมูลที่นำมาวิเคราะห์ได้

ลำดับที่	รหัสสินค้า	คำอธิบาย	จำนวน
1	b1301	เค้กมะพร้าว	1
1	b1302	เค้กเนยสด	2
2	...	เค้กกล้วยหอม	4
2	b2105	เค้กมะพร้าว	3
2		เค้กเนยสด	2
3		เค้กมะพร้าว	2
3		เค้กเนยสด	1



ID	เค้กกล้วยหอม	เค้กมะพร้าว	เค้กเนยสด
1	FALSE	TRUE	TRUE
2	TRUE	TRUE	TRUE
3	FALSE	TRUE	TRUE

3.5 การแปลงข้อมูล (Data Transformation)

3.5.1 การแปลงประเภทข้อมูล

- เมื่อวันที่ 4 สิงหาคม 2564 เฟสบุ๊กเปิดตัวหน้าเพจใหม่ชื่อ Facebook A look back เมื่อผู้ใช้คลิกไปยังหน้านี้ก็จะแสดงคลิปวิดีโอที่บอกถึงเรื่องราวของผู้ใช้คนนั้น เช่น เริ่มใช้งานเมื่อไหร่ โพสแรกบนเฟสบุ๊ก รูปภาพที่กดไลท์มากที่สุด รูปภาพที่ถูกแชร์มากที่สุด และ 20 อันดับเรื่องราวต่างๆ ที่เกิดในเฟสบุ๊ก ก็จะแสดงและรวบรวมไว้ในคลิปวิดีโอนี้

ID	เฟสบุ๊ก	ไลท์	แชร์	รูปภาพ	คลิปวิดีโอ
1	4	1	1	2	2
2					

ตารางแสดงความถี่ของคำที่พบบ่อย

3.5 การแปลงข้อมูล (Data Transformation)

3.5.1 การแปลงประเภทข้อมูล

Order_ID	b1301	b1302	b1303	...	B2105
T0000001	Y	Y	Y	?	?
T0000002	?	Y	Y	?	Y

Operator ใน Rapid Miner

- Filter
- Select attribute
- Set role

3.5 การแปลงข้อมูล (Data Transformation)

บทที่ 3

3.5.1 การแปลงประเภทข้อมูล

ตัวอย่าง_01 : เป็นการแปลงข้อมูลแบบต่อเนื่องให้เป็นข้อมูลแบบแบ่งกลุ่ม

File : Lec03_data_ transformation10.py

```
# นำเข้าไลบรารี pandas
import pandas as pd
# สร้าง DataFrame
df = pd.DataFrame({
    "อายุ": [18, 25, 35, 45, 55]
})
# แปลงข้อมูลแบบต่อเนื่องให้เป็นข้อมูลแบบแบ่งกลุ่ม
df["ช่วงอายุ"] = df["อายุ"].apply(lambda x: "อายุต่ำกว่า 25 ปี" if x < 25 else "อายุ 25-35 ปี" if x < 35 else "อายุ 35-45 ปี" if x < 45 else "อายุ 45-55 ปี" if x < 55 else "อายุ 55 ปีขึ้นไป")
# แสดงผล DataFrame
print(df)
```

```
   อายุ ช่วงอายุ
0  18   อายุต่ำกว่า 25 ปี
1  25   อายุ 25-35 ปี
2  35   อายุ 35-45 ปี
3  45   อายุ 45-55 ปี
4  55   อายุ 55 ปีขึ้นไป
```

3.5 การแปลงข้อมูล (Data Transformation)

บทที่ 3

3.5.1 การแปลงประเภทข้อมูล

ตัวอย่าง_02: เป็นการแปลงข้อมูลแบบนามธรรมให้เป็นข้อมูลแบบตัวเลข

File : Lec03_data_ transformation11.py

```
# นำเข้าไลบรารี pandas
import pandas as pd
# สร้าง DataFrame
df = pd.DataFrame({
    "ความคิดเห็น": ["ดี", "ปานกลาง", "แย่"]
})
# แปลงข้อมูลแบบนามธรรมให้เป็นข้อมูลแบบตัวเลข
df["คะแนนความคิดเห็น"] = df["ความคิดเห็น"].apply(lambda x: 5 if x == "ดี" else 3 if x == "ปานกลาง" else 1)
# แสดงผล DataFrame
print(df)
```

	ความคิดเห็น	คะแนนความคิดเห็น
0	ดี	5
1	ปานกลาง	3
2	แย่	1

3.5 การแปลงข้อมูล (Data Transformation)

บทที่ 3

3.5.1 การแปลงประเภทข้อมูล

ตัวอย่าง_03: เป็นการแปลงข้อมูลแบบข้อความให้เป็นข้อมูลตัวเลข

File : Lec03_data_ transformation12.py

```
import pandas as pd
# Create DataFrame
df = pd.DataFrame({
    "ที่อยู่": ["123/45 สุขุมวิท เขตวัฒนา กรุงเทพมหานคร 10110"]
})
# Remove non-numeric characters
df["ที่อยู่"] = df["ที่อยู่"].str.replace("[^\d]", "", regex=True)
# Convert to integer
df["ที่อยู่ตัวเลข"] = pd.to_numeric(df["ที่อยู่"])
# Display DataFrame
print(df)
```

```
      ที่อยู่  ที่อยู่ตัวเลข
0  1234510110     1234510110
```

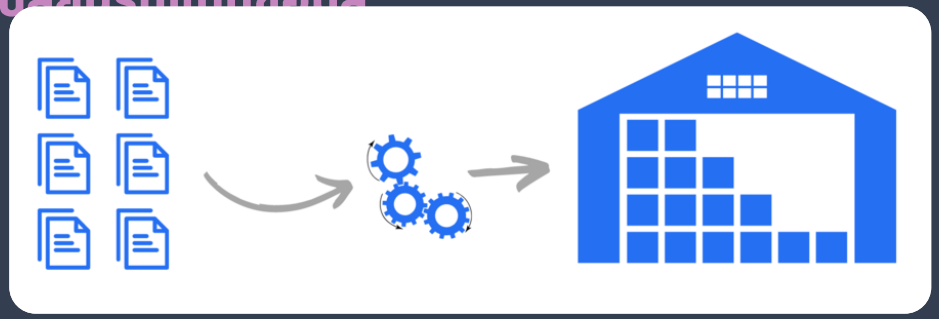
3.5 การแปลงข้อมูล (Data Transformation)

3.5.1 การแปลงประเภทข้อมูล

ตัวอย่าง_04: เป็นการแปลงข้อมูลแบบข้อความให้เป็นข้อมูลตัวเลข

การแปลงข้อมูล (Data Transformation) เป็นกระบวนการที่สำคัญในการเตรียมข้อมูลก่อนที่จะนำเข้าสู่กระบวนการการเรียนรู้ของเครื่องหรือการวิเคราะห์ข้อมูลอื่น ๆ โดยมักนำเข้าไลบรารี pandas เพื่อทำการจัดการข้อมูลในรูปแบบตาราง (DataFrame) และดำเนินการแปลงข้อมูลต่าง ๆ ตามความต้องการของงาน.

ตัวอย่างนี้แสดงการใช้งาน pandas เพื่อทำการแปลงข้อมูลใน DataFrame โดยใช้ฟังก์ชัน apply เพื่อปรับเปลี่ยนรูปแบบข้อมูล:



3.5 การแปลงข้อมูล (Data Transformation)

3.5.1 การแปลงประเภทข้อมูล

ตัวอย่าง_04: เป็นการแปลงข้อมูลแบบข้อความให้เป็นข้อมูลตัวเลข

ในตัวอย่างนี้:

ฟังก์ชัน transform_city ถูกสร้างขึ้นเพื่อแปลงชื่อเมืองในคอลัมน์ 'City' เป็นประเทศที่เกี่ยวข้อง.

คำสั่ง df['Country'] = df['City'].apply(transform_city) นำเข้าฟังก์ชันนี้เพื่อสร้างคอลัมน์ใหม่ 'Country' ที่มีข้อมูลที่ถูกแปลงจาก 'City'.

ผลลัพธ์ที่พิมพ์ออกมาจะแสดง DataFrame ก่อนและหลังการแปลงข้อมูล.

นั่นคือตัวอย่างการใช้งาน pandas เพื่อทำการแปลงข้อมูลใน DataFrame โดยใช้ฟังก์ชันที่กำหนดเอง. การแปลงข้อมูลที่ถูกต้องและเหมาะสมสำคัญในการเตรียมข้อมูลสำหรับการวิเคราะห์หรือการเรียนรู้ของเครื่อง.

3.5 การแปลงข้อมูล (Data Transformation)

บทที่ 3

3.5.1 การแปลงประเภทข้อมูล

ตัวอย่าง_04: การแปลงข้อมูลแบบข้อความให้เป็นข้อมูลตัวเลข

Part01 Lec03_data_ transformation13.py

```
import pandas as pd
# สร้าง DataFrame ตัวอย่าง
data = {'Name': ['John', 'Anna', 'Peter', 'Linda'],
        'Age': [28, 23, 32, 45],
        'City': ['New York', 'Paris', 'Berlin', 'London']}
df = pd.DataFrame(data)
print("Original DataFrame:")
print(df)
```

```
Original DataFrame:
    Name  Age      City
0   John   28  New York
1   Anna   23     Paris
2  Peter   32    Berlin
3  Linda   45    London

DataFrame after data transformation:
    Name  Age      City  Country
0   John   28  New York      USA
1   Anna   23     Paris   France
2  Peter   32    Berlin  Germany
3  Linda   45    London  Unknown
```

3.5 การแปลงข้อมูล (Data Transformation)

บทที่ 3

3.5.1 การแปลงประเภทข้อมูล

ตัวอย่าง_04: เป็นการแปลงข้อมูลแบบข้อความให้เป็นข้อมูลตัวเลข

Part02 Lec03_data_ transformation13.py

```
# สร้างฟังก์ชันสำหรับแปลงข้อมูล
def transform_city(city):
    if city == 'New York':
        return 'USA'
    elif city == 'Paris':
        return 'France'
    elif city == 'Berlin':
        return 'Germany'
    else:
        return 'Unknown'
```

```
Original DataFrame:
    Name  Age      City
0   John   28  New York
1   Anna   23     Paris
2  Peter   32    Berlin
3  Linda   45    London

DataFrame after data transformation:
    Name  Age      City  Country
0   John   28  New York      USA
1   Anna   23     Paris   France
2  Peter   32    Berlin  Germany
3  Linda   45    London  Unknown
```

3.5 การแปลงข้อมูล (Data Transformation)

บทที่ 3

3.5.1 การแปลงประเภทข้อมูล

ตัวอย่าง_04: เป็นการแปลงข้อมูลแบบข้อความให้เป็นข้อมูลตัวเลข

Part03 Lec03_data_ transformation13.py

```
# ใช้ apply เพื่อปรับเปลี่ยนข้อมูลในคอลัมน์ 'City'
df['Country'] = df['City'].apply(transform_city)
print("\nDataFrame after data transformation:")
print(df)
```

นั่นคือตัวอย่างการใช้งาน pandas เพื่อทำการแปลงข้อมูลใน DataFrame โดยใช้ฟังก์ชันที่กำหนดเอง. การแปลงข้อมูลที่ถูกต้องและเหมาะสมสำคัญในการเตรียมข้อมูลสำหรับการวิเคราะห์หรือการเรียนรู้ของเครื่อง.

```
Original DataFrame:
    Name  Age      City
0   John   28  New York
1   Anna   23     Paris
2  Peter   32    Berlin
3  Linda   45    London

DataFrame after data transformation:
    Name  Age      City  Country
0   John   28  New York      USA
1   Anna   23     Paris   France
2  Peter   32    Berlin  Germany
3  Linda   45    London  Unknown
```

ตัวอย่างหัวข้อโครงงาน

1. ด้านการศึกษา

- การวิเคราะห์พฤติกรรมนักเรียนเพื่อคาดการณ์ความเสี่ยงของการลาออกจากโรงเรียน
- การค้นหารูปแบบการเรียนรู้ของนักเรียนเพื่อเพิ่มประสิทธิภาพการสอน
- การคาดการณ์คะแนนสอบของนักเรียนโดยใช้เทคนิคเหมืองข้อมูล

2. ด้านธุรกิจ

- การวิเคราะห์พฤติกรรมลูกค้าเพื่อเพิ่มยอดขายในธุรกิจค้าปลีก
- การจัดกลุ่มลูกค้าเพื่อเสนอโปรโมชั่นที่เหมาะสม
- การคาดการณ์ความต้องการสินค้าในฤดูกาลต่าง ๆ

3. ด้านสุขภาพ

- การคาดการณ์ความเสี่ยงในการเกิดโรคเบาหวานจากพฤติกรรมสุขภาพ
- การวิเคราะห์แนวโน้มการระบาดของโรคในพื้นที่โดยใช้เหมืองข้อมูล
- การคัดกรองผู้ป่วยที่มีความเสี่ยงสูงด้วยข้อมูลจากเวชระเบียน

ตัวอย่างหัวข้อโครงงาน

4. ด้านสิ่งแวดล้อม

- การวิเคราะห์แนวโน้มมลพิษในอากาศโดยใช้ข้อมูล IoT
- การคาดการณ์การเปลี่ยนแปลงสภาพภูมิอากาศด้วยเหมืองข้อมูล
- การวิเคราะห์รูปแบบการใช้พลังงานในครัวเรือนเพื่อการจัดการที่ยั่งยืน

5. ด้านสังคมและพฤติกรรม

- การวิเคราะห์ความคิดเห็นจากโซเชียลมีเดียเพื่อคาดการณ์แนวโน้มทางสังคม
- การศึกษาพฤติกรรมการเดินทางของผู้โดยสารเพื่อปรับปรุงระบบขนส่งสาธารณะ
- การจัดกลุ่มผู้ใช้บริการสตรีมมิ่งเพื่อแนะนำเนื้อหาที่เหมาะสม

6. ด้านการเงิน

- การวิเคราะห์ความเสี่ยงในการให้สินเชื่อโดยใช้เหมืองข้อมูล
- การตรวจจับการทุจริตทางการเงินด้วยเทคนิคเหมืองข้อมูล
- การคาดการณ์ราคาหุ้นโดยใช้เทคนิคเหมืองข้อมูล

7. ด้านการเกษตร

- การวิเคราะห์ข้อมูลสภาพอากาศเพื่อเพิ่มผลผลิตทางการเกษตร
- การคาดการณ์โรคพืชและศัตรูพืชในไร่โดยใช้เหมืองข้อมูล
- การค้นหารูปแบบการเจริญเติบโตของพืชเพื่อการวางแผนเพาะปลูก
- หากต้องการเจาะจงไปในด้านใด หรือเทคนิคที่ใช้เพิ่มเติม สามารถสอบถามได้

อ้างอิง

1. Aggarwal, C. C. (2015). Data mining: The textbook. Springer.

2. Autosoft. (n.d.). Forecasting with the Microsoft time series data mining algorithm. Retrieved December 24, 2024, from http://www.autosoft.in.th/data-science/forecasting-with-the-microsoft-time-series-data-mining-algorithm

3. Data Mining Trend. (2014). Association rules. Retrieved December 24, 2024, from http://dataminingtrend.com/2014/association-rules

4. Han, J., Kamber, M., & Pei, J. (2011). Data mining: Concepts and techniques (3rd ed.). Elsevier.

5. IMP Attani. (2011, June 26). การทำเหมืองข้อมูล. Retrieved December 24, 2024, from https://impattani.wordpress.com/2011/06/26/การทำเหมืองข้อมูล

6. Kotu, V., & Deshpande, B. (2019). Data science: Concepts and practice. Morgan Kaufmann.

7. NMC Xpress. (2012, March 11). การทำเหมืองข้อมูล (Data mining). Retrieved December 24, 2024, from https://nmcxpress.wordpress.com/2012/03/11/การทำเหมืองข้อมูล-data-mining/

8. Sajee GM301. (2015, November). Data mining. Retrieved December 24, 2024, from http://sajeegm301.blogspot.com/2015/11/data-mining.html

9. Witten, I. H., Frank, E., Hall, M. A., & Pal, C. J. (2016). Data mining: Practical machine learning tools and techniques (4th ed.). Morgan Kaufmann.