🧠 03 - การวิเคราะห์ข้อความเชิงสถิติ

(Statistical Text Analysis)

Chanankorn

School of Informatics, Walailak University

🔍 บทนำเชิงทฤษฎี

การวิเคราะห์ข้อความเชิงสถิติเป็นกระบวนการทาง **Data Science** และ **Computational Linguistics** ที่นำหลักการทาง **คณิตศาสตร์และสถิติ** มาประยุกต์กับ **ข้อมูลข้อความ (Text Data)**

🎯 เป้าหมายหลัก:

- แปลงข้อความให้อยู่ในรูปเชิงปริมาณ
- เพื่อการจำแนกประเภท, วิเคราะห์หัวข้อ, วิเคราะห์อารมณ์ ฯลฯ

1 Text Preprocessing

(การเตรียมข้อมูลข้อความ)

เป็นขั้นตอนพื้นฐานที่สำคัญที่สุด
เพื่อจัดการ **Noise** เช่น คำฟุ่มเฟือยหรือเครื่องหมายวรรคตอน

ขั้นตอนหลักของ Text Preprocessing

1. **Tokenization** – ตัดคำ
2. **Stopword Removal** – ลบคำที่ไม่สำคัญ
3. **Stemming / Lemmatization** – แปลงคำให้อยู่ในรูปมาตรฐาน
4. **Symbol & Number Removal** – ลบตัวเลขและสัญลักษณ์ที่ไม่จำเป็น

✂️ Tokenization – การตัดข้อความเป็นคำ

ตัวอย่าง	ผลลัพธ์หลัง Tokenization
“Data scientists are analyzing large amounts of data in 2024!”	`['Data', 'scientists', 'are', 'analyzing', 'large', ...]`

🟦 แยกข้อความเป็นคำหรือวลี เพื่อให้นำไปวิเคราะห์เชิงสถิติได้

🚫 Stopword Removal – ลบคำที่ไม่สำคัญ

ลบคำทั่วไป เช่น “the”, “of”, “in”

ตัวอย่าง	ผลลัพธ์
“Data scientists are analyzing large amounts of data in 2024!”	`['Data', 'scientists', 'analyzing', 'large', 'data']`

🔤 Lemmatization – ทำให้คำอยู่ในรูปมาตรฐาน

รวมคำที่มีรากเดียวกัน เช่น

“analyzing” → “analyze”, “models” → “model”

🔣 Symbol & Number Removal

ลบตัวเลข เครื่องหมาย และสัญลักษณ์ที่ไม่จำเป็น

📍 “AI model perform exceptionally well experiment”

🧩 สรุปภาพรวมการเตรียมข้อมูล

ขั้นตอน	วัตถุประสงค์	ผลลัพธ์
Tokenization	แยกคำ	รายการคำ
Stopword Removal	ลบคำทั่วไป	ลด Noise
Lemmatization	รวมรูปคำ	คำมาตรฐาน
Symbol Removal	ลบตัวเลข/สัญลักษณ์	ข้อความสะอาด

💻 ตัวอย่างโค้ด Python

```
import nltk
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from nltk.stem import WordNetLemmatizer
import string

nltk.download('punkt')
nltk.download('stopwords')
nltk.download('wordnet')

text = "Statistical text analysis involves processing text data."

tokens = word_tokenize(text.lower())
filtered = [w for w in tokens if w not in stopwords.words('english') and w not in string.punctuation]
lemmatizer = WordNetLemmatizer()
lemmatized = [lemmatizer.lemmatize(word) for word in filtered]

print("After Cleaning:", lemmatized)
```

2 Statistical Representation

(การแทนค่าข้อความเชิงสถิติ)

หลังจากเตรียมข้อมูลแล้ว → ต้องแทนค่าข้อความเป็นเชิงตัวเลข
เพื่อให้วิเคราะห์ทางสถิติได้

🔹 Bag of Words (BoW)

แนวคิด:
นับจำนวนคำที่ปรากฏในแต่ละเอกสารโดยไม่สนใจลำดับคำ

📊 เช่น
“data scientist analyze large amount data” → `[data:2, scientist:1, ...]`

✅ เข้าใจง่าย
⚠️ ไม่สนใจบริบทของคำ

🔷 TF–IDF (Term Frequency – Inverse Document Frequency)

แนวคิด:
ให้ค่าน้ำหนักกับคำที่ปรากฏเฉพาะในบางเอกสาร

🧮 สูตร:

```
TF–IDF(t, d) = TF(t, d) × log(N / n_t)
```

ช่วยให้คำสำคัญโดดเด่นขึ้น เช่น

- “experiment” → ค่าสูง
- “data” → ค่าต่ำ

🔹 N-grams

แนวคิด:
พิจารณาคำต่อเนื่องกัน (sequence of words)

- Unigram → “machine”
- Bigram → “machine learning”
- Trigram → “machine learning helps”

ช่วยให้จับความหมายและบริบทได้ดีกว่า BoW

สรุปเปรียบเทียบเทคนิคการแทนข้อความ

เทคนิค	หลักการ	ข้อดี	ข้อจำกัด
BoW	นับคำ	เข้าใจง่าย	ไม่รู้ลำดับคำ
TF–IDF	น้ำหนักตามความสำคัญ	เน้นคำสำคัญ	ไม่รู้บริบท
N-grams	พิจารณาลำดับคำ	เข้าใจบริบท	ข้อมูลขยายใหญ่

3 Descriptive Statistics

(การวิเคราะห์เชิงพรรณนา)

ใช้เพื่ออธิบายลักษณะของข้อมูลข้อความ เช่น

- นับคำที่พบบ่อย
- สร้าง **Word Cloud**

💻 ตัวอย่างโค้ด

```
from collections import Counter
from wordcloud import WordCloud
import matplotlib.pyplot as plt

words = ["data", "text", "analysis", "data", "machine", "learning"]
freq = Counter(words)
print(freq.most_common(5))

wc = WordCloud(width=800, height=400, background_color='white').generate(" ".join(words))
plt.imshow(wc, interpolation='bilinear')
plt.axis("off")
plt.show()
```

4 Inferential & Predictive Analysis

🔹 Topic Modeling (LDA)

แยกข้อความออกเป็นหัวข้อโดยอัตโนมัติ

```
from sklearn.decomposition import LatentDirichletAllocation
from sklearn.feature_extraction.text import CountVectorizer

docs = ["Machine learning improves analytics.", "Deep learning in AI."]
X = CountVectorizer().fit_transform(docs)
lda = LatentDirichletAllocation(n_components=2).fit(X)
```

🔹 Sentiment Analysis

วิเคราะห์อารมณ์ข้อความ เช่น บวก / กลาง / ลบ

```
from textblob import TextBlob
texts = ["I love this product!", "Worst experience ever."]
for t in texts:
    print(t, TextBlob(t).sentiment.polarity)
```

Clustering / Classification

ใช้ K-means หรือ Naïve Bayes
เพื่อจัดกลุ่มข้อความโดยอัตโนมัติ

```
from sklearn.cluster import KMeans
from sklearn.feature_extraction.text import TfidfVectorizer
texts = ["AI and ML", "Statistical data science"]
X = TfidfVectorizer().fit_transform(texts)
KMeans(n_clusters=2).fit(X)
```

5 แนวคิดเชิงทฤษฎีที่เกี่ยวข้อง

สาขาวิชา	คำอธิบาย
Corpus Linguistics	ศึกษาภาษาโดยใช้คลังข้อมูลขนาดใหญ่
Information Retrieval (IR)	การค้นหาและดึงข้อความที่เกี่ยวข้อง
Natural Language Processing (NLP)	การประมวลผลและเข้าใจภาษามนุษย์

6 ตัวอย่างการประยุกต์

การประยุกต์	คำอธิบาย
วิเคราะห์โพสต์สังคมออนไลน์	ตรวจจับอารมณ์ของผู้ใช้
วิเคราะห์ข่าวเศรษฐกิจ	หาหัวข้อแนวโน้มจากบทความ
วิเคราะห์รีวิวสินค้า	ตรวจสอบความพึงพอใจของลูกค้า
วิเคราะห์บทความวิชาการ	ดึงคำสำคัญด้วย TF-IDF

7 กรณีศึกษา: การตรวจจับ Spam

การจำแนกข้อความว่าเป็น **Spam / Ham**
โดยใช้เทคนิค TF–IDF และ Naïve Bayes

💻 ตัวอย่างโค้ดย่อ

```
from sklearn.naive_bayes import MultinomialNB
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer

X_train, X_test, y_train, y_test = train_test_split(texts, labels)
tfidf = TfidfVectorizer()
X_train_tfidf = tfidf.fit_transform(X_train)

model = MultinomialNB().fit(X_train_tfidf, y_train)
```

🔎 สรุปขั้นตอนของ Spam Detection

ขั้นตอน	รายละเอียด
โหลดข้อมูล	Spam / Ham messages
Preprocessing	ลบ stopwords, lowercase
TF–IDF	แปลงข้อความเป็นตัวเลข
Classification	Naïve Bayes
Evaluation	Accuracy, Confusion Matrix

🧠 การขยายเชิงวิชาการ

- ทดลองโมเดลอื่น: Logistic Regression, SVM, Random Forest
- ใช้ **N-grams** เพื่อเพิ่มบริบท
- วิเคราะห์ **Feature Importance**
- ใช้ชุดข้อมูลจริง เช่น *Kaggle SMS Spam Dataset*

🟦 สรุป

- Text Analysis คือการเปลี่ยนข้อความเป็นข้อมูลเชิงตัวเลข
- ใช้สถิติและ Machine Learning เพื่อเข้าใจเนื้อหา
- มีการประยุกต์ใช้กว้างขวางใน Data Science และ NLP

🙏 ขอบคุณครับ

Chanankorn

School of Informatics, Walailak University