วิโรจน์ อรุณมานะกุล



R และสถิติพื้นฐาน

ภาษาศาสตร์ จุฬาลงกรณ์มาวิทยาลัย

© ๒๕๖๕ วิโรจน์ อรุณมานะกุล

ภาควิชาภาษาศาสตร์ คณะอักษรศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย

คำนำ

หนังสือเล่มนี้เขียนขึ้นมาเพื่อใช้เป็นคู่มือประกอบการใช้โปรแกรม R ในการทำงานด้านสถิติ โดยมุ่ง เน้นที่การใช้งานทางภาษาศาสตร์เป็นหลัก แต่ก็สามารถนำไปใช้กับงานด้านอื่นได้ หนังสือประกอบ ด้วยสามบท บทแรกอธิบายหลักแนวคิดพื้นฐานทางสถิติและภาพรวมของสถิติแบบต่าง ๆ บทที่สอง เป็นตัวอย่างการใช้งานโปรแกรม R พร้อม script ที่ใช้ในทางสถิติ ส่วนบทที่สามเป็นเรื่องการใช้ R ใน งานที่เป็น data mining

^{บทที่ ๑} สถิติเบื้องต้น

สถิติแยกประเภทใหญ่ได้เป็น descriptive statistics กับ inferential statistics Descriptive statistics เป็นสถิติที่ใช้เพื่ออธิบายข้อมูลทั้งหมดที่รวบรวมมา คือแทนที่จะอธิบาย หรือแจกแจงข้อมูลที่รวบรวมมาทีละตัวๆ เราก็ใช้วิธีสรุปภาพรวมของข้อมูลชุดนั้นออกมาเป็นตัวเลข จำนวนหนึ่ง โดยใช้วิธีการอย่างการนับความถี่ (frequency) เพื่อดูการกระจายตัวของข้อมูลที่พบ ใช้การวัดค่ากลางออกมาเป็นตัวแทนข้อมูล (central tendency) อย่างเช่น การหาค่าเฉลี่ยที่เป็นค่า mean ค่า mode หรือค่า median เป็นต้น ค่าสถิติเหล่านี้จะบ่งบอกถึงลักษณะโดยรวมของข้อมูล ที่รวบรวมมาได้ ส่วน Inferential statistics เป็นการใช้สถิติเพื่อหาข้อสรุปสำหรับข้อมูลที่มีจำนวน มาก เราไม่สามารถเก็บข้อมูลทั้งหมดได้ จึงต้องเลือกสุ่มตัวอย่างข้อมูล (sample) มาเพื่อใช้เป็น ตัวแทน (representative) ของข้อมูลทั้งหมด แล้วสรุปค่าทางสถิติที่ได้จากกลุ่มตัวอย่างนั้นเพื่อใช้ อนุมาน (infer) ถึงลักษณะที่คาดว่าจะเป็นของประชากรทั้งหมด (population)

Descriptive Statistics

สถิติเซิงพรรณาเป็นการใช้สถิติเพื่อหาข้อสรุปเกี่ยวกับข้อมูลทั้งหมดที่รวบรวมมาได้ เช่น กรณีที่เราต้องการเปรียบเทียบผลการเรียนของนักเรียนห้องหนึ่งเทียบกับอีกห้องหนึ่ง เราสามารถ แจกแจงผลการเรียนของนักเรียนแต่ละคนในห้องได้ว่ามีจำนวนนักเรียนได้คะแนนสอบในแต่ละช่วง มากน้อยกว่ากันเพียงใดคือดูจากการ กระจายตัวของข้อมูลคะแนน แต่วิธีที่ช่วยให้เห็นภาพเปรียบ เทียบโดยง่ายคือการสรุปผลออกมาเป็นค่าตัวเลขกลาง เช่น ค่าเฉลี่ยของคะแนนนักเรียนในแต่ละห้อง ก็จะทำให้เห็นว่าคะแนนเฉลี่ยของนักเรียนแต่ละห้องมากน้อยต่างกันอย่างไร แต่การดูเฉพาะค่า คะแนนเฉลี่ยอย่างเดียวก็ยังไม่พอ เพราะนักเรียนสองห้องอาจคำนวณค่าคะแนนเฉลี่ยออกมาเท่ากัน แต่นักเรียนห้องแรกอาจมีคะแนนเกาะกลุ่มกันคือนักเรียนส่วนใหญ่ได้คะแนนใกล้เคียงกับค่าเฉลี่ย แต่ อีกห้องหนึ่งจะมีนักเรียนที่คะแนนต่างกันมากไม่เกาะกลุ่มคือมีทั้งคนที่ได้คะแนนสูงมากและคนที่ได้ คะแนนต่ำมาก แต่เมื่อคำนวณออกมาแล้วได้ค่าเฉลี่ยเท่ากันกับค่าเฉลี่ยของคะแนนจากห้องแรก กรณีแบบนี้ ค่าเบี่ยงเบนมาตรฐาน (standard deviation) ของคะแนนจากแต่ละห้องจะสะท้อนภาพ ที่แตกต่างกันนี้ได้ คือห้องแรกที่คะแนนโดยมากเกาะกลุ่มใกล้ค่าคะแนนเฉลี่ยจะมีค่าเบี่ยงเบน มาตรฐานต่ำในขณะที่ห้องที่สองที่มีทั้งคนที่ได้คะแนนมากและคะแนนน้อยจะมีค่าเบี่ยงเบนมาตรฐาน สูงกว่า เป็นต้น

Inferential Statistics

ในงานวิจัยส่วนใหญ่แล้ว เรามักไม่สามารถเก็บข้อมูลทั้งหมดได้เนื่องจากสิ้นเปลืองเวลามาก หรือบางกรณีก็เป็นไปไม่ได้ที่จะเก็บข้อมูลทั้งหมดมาได้ บางกรณีก็เป็นเรื่องยุ่งยากและมีค่าใช้จ่ายสูง เกินไปในการเก็บข้อมูลมาทั้งหมด ในกรณีที่เราไม่สามารถเก็บข้อมูลทั้งหมดมาได้นี้ เราจะใช้วิธีการ สุ่มตัวอย่าง (sample) มาเพื่อเป็นตัวแทนของประชากรทั้งหมด (population) จากนั้นจะวิเคราะห์ ทางสถิติเพื่อให้ได้ค่าที่จะสามารถนำมาอนุมาน (infer) ว่าเป็นคุณสมบัติของประชากรทั้งหมดนั้นใน ภายหลัง การใช้งานสถิติลักษณะนี้คือที่เรียกว่าสถิติเชิงอนุมานหรือ inferential statistics

วิธีการที่ใช้ใน inferential statistics แยกออกได้เป็นสองกลุ่ม คือ parameter estimation และ hypothesis testing สถิติกลุ่ม parameter estimation เป็นการใช้กลุ่มตัวอย่าง มาเพื่อประมาณค่าที่ควรจะเป็นของประชากรทั้งหมดนั้น และเนื่องจากเป็นการประมาณค่าจึงมี โอกาสที่จะผิดพลาดได้ ในทางสถิติจึงจะต้องพูดถึงระดับความเชื่อมั่น (confidential interval) ไป ด้วย เช่น ถ้าหาค่าเฉลี่ย (mean) จากกลุ่มตัวอย่างมาได้ ก็จะต้องบอกว่า 95% confidence interval หรือระดับความมั่นใจว่าค่าเฉลี่ยที่ถูกต้องอย่างน้อย 95% นั้นจะอยู่ภายในช่วงค่าใดซึ่งก็จะ เป็นตัวเลขค่าเฉลี่ยบวกลบค่าตัวเลขช่วงหนึ่ง

สถิติกลุ่ม hypothesis testing เป็นการใช้วิธีการทางสถิติเพื่อหาความสัมพันธ์ระหว่าง ตัวแปร dependent กับ independent (independent variable หรือตัวแปรต้นเป็นตัวที่เราคิด ว่าเป็นเหตุที่ทำให้มีผลต่อ dependent variable หรือตัวแปรตาม เช่น เราอาจคิดว่า เพศมีผลต่อ ความยาวของประโยคที่พูด เพศก็เป็นตัวแปรต้น ความยาวของประโยคก็จะเป็นตัวแปรตาม) สถิติ กลุ่มนี้ยังแยกออกเป็น parametric testing กับ non-parametric testing

hypothesis testing จะมี null hypothesis ที่ตรงข้ามกับสิ่งที่เราคิด เราทดสอบเพื่อจะ ปฏิเสธ null hypothesis นี้เพื่อที่จะได้ยอมรับ alternative hypothesis ซึ่งเป็นสิ่งที่เราคาดว่าควร จะเป็น คือโดยปกติ เราคาดว่าจะมีความสัมพันธ์ระหว่างตัวแปรที่เราต้องการศึกษาอยู่ แต่เราจะตั้ง null hypothesis ว่าไม่มีความสัมพันธ์ระหว่างตัวแปรดังกล่าวเพื่อที่จะปฏิเสธ null hypothesis นั้น เหตุที่ทางสถิติเราจะตั้งสมมติฐานแบบนี้ ก็เพราะการตั้งสมมติฐานแบบ null hypothesis จะพิสูจน์ ว่าไม่จริงได้ง่ายกว่า ตัวอย่างเช่น สมมติว่าเราตั้งสมมติฐานว่ามีหนูอยู่ในบ้าน การที่เราเดินเข้าไปดูใน บ้านหลาย ๆ ครั้งก็ยังไม่พบหนูในบ้านก็ยังไม่เป็นเหตุเพียงพอที่จะปฏิเสธสมมติฐานนี้ได้ เพราะจริง ๆ หนูอาจจะออกมาในเวลาที่เราหลับหรือไม่ได้เฝ้ามอง แต่ถ้าเราตั้งสมมติฐานว่าไม่มีหนูอยู่ในบ้าน เรา สามารถปฏิเสธสมมติฐานนี้ได้ทันทีหากเราพบเห็นหนูสักตัวหนึ่ง ดังนั้น เวลาที่เราตั้งสมมติฐานว่าไม่มี ความสัมพันธ์ใด ๆ แต่ในข้อมูลมีสิ่งชี้ให้เห็นว่ามีความสัมพันธ์บางอย่างอยู่ เราก็จะปฏิเสธสมมติฐาน ว่าไม่มีความสัมพันธ์ได้ การตั้งสมมติฐานแบบที่สองจึงเหมาะสมกว่า สิ่งนี้สะท้อนให้เห็นหลักการ สำคัญที่ Crawley (2005: Kindle Locations 305-306) กล่าวไว้ "absence of evidence is not evidence of absence"

อย่างไรก็ตาม การทดสอบสมมติฐานนี้เป็นการสรุปจากกลุ่มตัวอย่างที่เราสังเกตุเท่านั้น จึง เป็นไปได้ว่าอาจมีความผิดพลาดได้ ซึ่งความผิดพลาดเป็นได้สองลักษณะ ลักษณะแรกคือเราปฏิเสธ สมมติฐานโดยที่สมมติฐานนั้นเป็นจริง ทางสถิติจะเรียกว่าเป็น Type I error อีกลักษณะหนึ่งคือเรา ควรจะปฏิเสธสมมติฐานนั้นแต่เราไม่ได้ทำ ทางสถิติเรียกว่าเป็น Type II error มีตัวอย่างที่มีคน เปรียบเทียบว่า Type I error เหมือนกับการบอกผู้หญิงท้องว่าไม่ได้ตั้งครรภ์ ส่วน Type II error เหมือนกับการบอกกับผู้ชายว่าเขากำลังตั้งครรภ์ ในเวลาที่เราทดสอบ null hypothesis นี้เราจะดู ค่าความน่าจะเป็นเพื่อบอกถึงความมั่นใจในการปฏิเสธสมมติฐานด้วย โดยทั่วไปจะใช้ค่าความน่าจะ เป็นน้อยกว่า 0.05 ซึ่งบ่งบอกความมั่นใจได้อย่างน้อย 95% ที่จะปฏิเสธสมมติฐานนั้น

เนื่องจากสถิติที่เราจะใช้นั้นมีหลากหลาย การจะเลือกใช้สถิติตัวไหนนั้นขึ้นกับชนิดของ ข้อมูล ข้อมูลที่ใช้ในทางสถิติแบ่งเป็นประเภทต่างๆ ได้แก่ nominal, ordinal, interval, ratio

nominal คือ ข้อมูลที่สามารถจัดเป็นกลุ่มหรือ categorize ได้ว่าเป็นอะไร เช่น คำตอบว่า Yes No nominal เป็นข้อมูลที่แยกประเภทต่างๆชัดเจน เช่น เพศ

ordinal เป็นข้อมูลที่มีการเรียงลำดับจากน้อยไปมาก แต่ตัวเลขไม่ได้มีค่าที่แท้จริงอยู่ เช่น scale
 1-5 อาจใช้เป็น 0-4 ก็ได้ ช่วงห่างระหว่าง 1-2, กับ 2-3 ไม่ได้มีนัยยะว่ามีความแตกต่างเท่ากัน

 interval มีลักษณะของการเป็น scale ที่แต่ละช่วงห่างมีความหมายเท่าๆกัน เพียงแต่ว่าค่า ตัวเลข ที่เป็นศูนย์ไม่ได้มีความหมายเป็นศูนย์แบบสมบูรณ์ (absolute zero) ตัวอย่างเช่น scale การวัด อุณหภูมิ ตัวเลข 30 องศาไม่ได้มีความหมายว่าร้อนเป็นสองเท่าของ 15 องศา

ratio คือค่า scale ของตัวเลขที่มีค่าศูนย์แบบสมบูรณ์ ตัวอย่างเช่น scale ของการวัดอุณหภูมิที่มี
 หน่วยเป็น kelvin คะแนนสอบของนักเรียน เป็นต้น

แม้ว่าในหนังสือสถิติโดยทั่วไปจะแยกข้อมูลออกเป็นสี่ประเภทนี้ แต่ในการพิจารณา เราจะ มองข้อมูเป็นสองกลุ่ม คือกลุ่มที่เป็นเหมือนป้ายหรือ label (nominal, ordinal) ซึ่งสถิติที่ใช้กับ ข้อมูลกลุ่มนี้เรียกว่า non-parametric test เช่น Chi-square, Mann-Whitney U-test, Wilcoxon ranked test, Kruskal–Wallis test กับกลุ่มที่เป็นเหมือนตัวเลขวัด numeric (interval, ratio) ซึ่งสถิติที่ใช้กับข้อมูลกลุ่มนี้เรียกว่า parametric test เช่น t-test, z test, anova

โดยทั่วไปแล้ว การใช้สถิตินั้นเป็นวิธีการใช้เครื่องมือเพื่อช่วยยืนยันความคิดหรือสมมติฐาน บางอย่างที่เราคาดไว้ เช่น เรามองเห็นหรือคาดว่าน่าจะมีความสัมพันธ์ระหว่างตัวแปรต่าง ๆ ซึ่ง ความสัมพันธ์นั้นมีคำอธิบายในเชิงหลักการหรือแนวคิดทฤษฎีบางอย่างได้ เราจึงได้ไช้วิธีการทางสถิติ ที่เหมาะสมกับข้อมูลที่ศึกษาเพื่อช่วยยืนยันความสัมพันธ์ที่มีอยู่นั้นจากข้อมูลที่เก็บมาศึกษา สถิติจึง มักใช้เป็นเครื่องมือเพื่อช่วยยืนยันถึงการมีอยู่ของความสัมพันธ์ที่มีอยู่นั้นจากข้อมูลที่เก็บมาศึกษา สถิติจึง มักใช้เป็นเครื่องมือเพื่อช่วยยืนยันถึงการมีอยู่ของความสัมพันธ์บางอย่างที่เราสามารถให้คำอธิบายได้ เราไม่ควรใช้วิธีการทางสถิติจับดูความสัมพันธ์ใด ๆ แล้วเมื่อเห็นว่าได้ค่าที่มีนัยสำคัญทางสถิติแล้วก็มา สรุปว่าตัวแปรเหล่านั้นมีความสัมพันธ์ต่อกันโดยที่ไม่มีเหตุผลหรือคำอธิบายที่ดีพอว่าทำไมจึงมีความ สัมพันธ์กัน เช่น สมมติว่าเราเก็บข้อมูลความสูงของคนกับความนิยมในการใส่เสื้อสีต่างๆ มาคำนวณ ด้วยวิธีการทางสถิติบางอย่างแล้วพบว่ามีความสัมพันธ์ต่อกัน ก็ไม่ได้หมายความว่า ความสูงของคน จะมีผลต่อการเลือกใส่เสื้อสีต่าง ๆ ตราบเท่าที่เรายังไม่สามารถหาเหตุผลอะไรที่จะนำมาใช้อธิบาย ความสัมพันธ์นี้ได้ เป็นต้น

^{บทที่ ๒} การใช้โปรแกรม R

R เป็นภาษาคอมพิวเตอร์ภาษาหนึ่งที่เป็นที่นิยมในการใช้คำนวณทางสถิติ เพราะเป็นโปร แกมที่ใช้ได้ฟรีบนเครื่องคอมพิวเตอร์แบบต่างๆ ไม่ว่าจะใช้บน Windows, Mac OS, หรือ Linux ภาษา R พัฒนามาจากภาษา S ซึ่งพัฒนาขึ้นมาเพื่อใช้ในงานสถิติและต่อมากลายเป็น S+ แต่ ซอฟต์แวร์นี้มีราคาแพงเกินไปที่จะซื้อมาใช้ในสถาบันการศึกษา ภายหลังนักสถิติสองคนคือ Ross Ihaka และ Robert Gentleman จึงได้ช่วยกันเขียนซอฟต์แวร์ตามแบบ S+ แต่ดึงมาบางส่วนเพื่อให้ เพียงพอสำหรับการสอนสถิติและตั้งชื่อว่า R เพื่อให้มีนัยว่ามาก่อน S โปรแกรม R มีการเผยแพร่ แบบ General Public License ในปี 1995 โปรแกรม R จึงเป็น open source ที่พัฒนามาจาก S+ (ดู Crawley 2005) R เป็นที่นิยมใช้กันในวงวิชาการเพื่อคำนวณด้านสถิติ เพราะมี built-in function ที่เกี่ยวข้องกับการคำนวณทางสถิติมาก และมีความสามารถแสดงผลทางด้านกราฟฟิก แม้โปรแกรม R ดูเหมือนไม่มี interface ให้ใช้ง่ายๆ แบบ SPSS ต้องสั่งงานผ่าน command line แต่หากได้รู้จักและคุ้นเคยกับ R แล้ว จะเห็นว่า R มีประสิทธิภาพและทำงานได้เร็วกว่า และทำได้ มากกว่าการคำนวณสถิติ และที่สำคัญเป็นโปรแกรมฟรี จึงเป็นที่นิยมใช้ของนักวิชาการจำนวนมาก ้นอกจากนี้ ในงานทางด้าน data sciences ซึ่งเป็นที่สนใจอย่างมากในปัจจุบัน R ยังเป็นภาษาที่ ้นิยมใช้กันในงานด้านนี้พอ ๆ กับภาษา Python เพราะ R มีฟังก์ชันทางสถิติที่ใช้ช่วยในการประมวล ้ผลข้อมูลใหญ่ (big data) ได้โดยง่าย แต่ในบทนี้จะกล่าวถึงการใช้งานโปรแกรม R เพื่อคำนวณสถิติ พื้นฐานสำหรับงานวิจัยทางภาษาศาสตร์

การติดตั้งโปรแกรม R

โปรแกรม R สามารถดาวน์โหลดได้ฟรีจาก <u>http://cran.r-project.org</u> แล้วเลือกว่าจะ ใช้บนเครื่องอะไร เมื่อติดตั้งเสร็จเรียกโปรแกรม R ขึ้นมาจะเห็นเป็น console ของมันเอง มี เครื่องหมาย > แสดงว่าพร้อมจะรับคำสั่ง

R Console

```
R version 2.11.1 (2010-05-31)
Copyright (C) 2010 The R Foundation for Statistical Computing
ISBN 3-900051-07-0
R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.
Natural language support but running in an English locale
R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.
Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.
>
```

_ • ×

ให้ทดลองพิมพ์คำสั่งต่างๆ หลัง > R จะประมวลผลแล้วแสดงผลออกมาให้ เช่น > 3+2 จะตอบมาว่า [1] 5 เราสามารถเก็บค่าที่ได้ไว้ในตัวแปรแบบที่ใช้ในภาษาคอมพิวเตอร์ได้ เช่น > x = 2+3 เป็นการสร้างตัวแปร x ซึ่งจะมีค่าเป็น 5 หรือ > x = x + 1 จะสั่งให้เพิ่มค่าใน x อีกหนึ่ง บางคนจะนิยมใช้เครื่องหมาย <- แทน = โดยมีความหมายเดียวกัน เช่น x <- x+1 ก็ หมายถึงให้เพิ่มค่าใน x อีก 1 แล้วเก็บไว้ที่เดิมคือ x

ตัวแปรที่ใช้ใน R ตัวพิมพ์ใหญ่เล็กถือว่ามีความแตกต่างกัน เมื่อต้องการเลิกใช้โปรแกรม R ใช้คำสั่ง >q() หรือ >quit() ก่อนจะปิดโปรแกรมเราสามารถเลือก save workspace เพื่อ โหลดข้อมูลทั้งหมดที่ทำค้างไว้กลับมาทำงานต่อได้ หากมีข้อสงสัยการใช้งานคำสั่งใดให้พิมพ์ >help(....) เช่น >help("getwd")

>getwd() ดูว่า working directory ณ ปัจจุบันคืออะไร

>setwd("c:/temp/RFiles") ใช้กำหนด working directory ให้อยู่ที่ c:/temp/RFiles working directory คือที่ที่โปรแกรมจะถือว่าเป็นที่ตั้งต้นที่จะอ่านหรือเขียนไฟล์ใดใด

การติดตั้ง RStudio

การใช้งาน R ผ่าน R console เป็นการสั่งงานผ่านทาง command line ซึ่งหลายคนอาจ ไม่คุ้นเคย หากต้องการใช้งาน R ในลักษณะที่มีหน้าจอต่างๆ ให้พร้อมสำหรับการแก้ไขชุดคำสั่งต่างๆ หน้าจอแสดงผล หน้าจอแสดงตัวแปรต่างๆ าลา ก็สามารถติดตั้ง RStudio เพิ่มเติมได้ ให้ไปที่เว็บ rstudio.com แล้วดาวน์โหลดโปรแกรม RStudio desktop ที่เป็น open source edition มาติดตั้ง



การใช้ RStudio จะช่วยอำนวยความสะดวกในเวลาที่เราต้องใช้ชุดคำสั่งเดิมหลายๆครั้ง เราสามารถนำชุดคำสั่งทั้งหมดเก็บเป็นไฟล์โปรแกรม R และเรียกมาใช้งานผ่าน RStudio ได้ทีละ หลายคำสั่งได้ ทำให้ไม่ต้องเสียเวลาพิมพ์หรือเรียกคำสั่งใหม่ทีละคำสั่ง

ประเภทข้อมูลใน R

ข้อมูลพื้นฐานที่ใช้ใน R คือ เว็กเตอร์ ซึ่งเป็นข้อมูลมิติเดียว มีข้อมูลตัวเดียวหรือหลายตัว ก็ได้ แต่จะเป็นเป็น object ประเภทเดียวกัน object แบบ atomic ที่มีใช้ใน R ได้แก่ character, numeric, integer, complex number, และ logical (True/False) vector เป็นการมองข้อมูล แบบมีลำดับ เช่น c(1,3,4) จะต่างจาก c(3,4,1)

เราสามารถป้อนข้อมูลเข้าโดยตรงเป็น **vector** ใน R ได้โดยพิมพ์ c และตามด้วยวงเล็บ บอกลำดับข้อมูล (c ย่อมาจาก concatenate เป็นฟังก์ชั่นใน R ให้เอาข้อมูลมาต่อกันเป็น vector) ข้อมูลที่อยู่ใน vector ต้องเป็นข้อมูลแบบเดียวกัน เช่น เป็นตัวเลขทั้งหมดหรือไม่ก็ตัวอักษรทั้งหมด เช่น >x = c(10, 5, 23, 18) เป็นการสร้าง vector ที่มีค่า 4 ค่าตามลำดับคือ 10, 5, 23, 18 ไป เก็บไว้ที่ตัวแปร x เราสามารถอ้างถึงค่าแต่ละตัวใน vector โดยอ้างถึงลำดับที่ เช่น x[2] จะมีค่า เป็น 5 หากต้องการป้อนค่าเข้าไปทางคีย์บอร์ดให้ใช้คำสั่ง scan() เมื่อใช้คำสั่ง x = scan() แล้ว พิมพ์ตัวเลขเข้าไป 20 ตัว โดยเว้นวรรคให้ตัวเลขแต่ละตัว เมื่อบรรทัดต่อไปเคาะ Enter เลย ก็จะ หยุดรับข้อมูล แล้วรายงานว่ามีข้อมูลอ่านเข้าใน vector ที่ตั้งชื่อว่า x 20 ตัว

```
> x=scan()
1: 1 3 2 3 3 3 4 2 1 2 2 1 1 4 3 3 2 1 2 3
21:
Read 20 items
```

หากข้อมูลที่นำเข้าเป็นข้อมูลต่างประเภทกัน เช่น เป็นตัวเลขบ้าง เป็นข้อความบ้าง สามารถนำเข้าได้โดยใช้คำสั่ง list และอ้างถึงข้อมูลเฉพาะตำแหน่งที่ต้องการได้

```
> l <- list('male',34,'single',67.8)
> l[3]
[[1]]
[1] "single"
```

ข้อมูล vector และ list จึงเป็นข้อมูลแบบมิติเดียว ต่างกันที่ vector ไม่มีการปนกันของ ข้อมูลอยู่ภายใน ในกรณีที่ข้อมูลที่เราใช้ในการคำนวณสถิติมีตัวแปรหลายๆตัว ข้อมูลจะมีลักษณะ เป็นตารางโดยที่ข้อมูลแต่ละคอลัมน์แทนตัวแปรแต่ละตัวได้ ข้อมูลหลายมิตินี้เป็นข้อมูลแบบ matrix หรือไม่ก็ data frame เราจะเรียกว่าเป็น matrix ถ้าข้อมูลทุกตัวในตารางนั้นต้องเป็น object ชนิด เดียวกัน ส่วน data frame เป็นตารางที่ข้อมูลที่อยู่ต่างคอลัมน์เป็นข้อมูลต่างชนิดกันได้ ความ ต่างของ matrix และ data frame จึงเหมือนความต่างระหว่าง vector กับ list เพียงแต่ว่า matrix และ data frame เป็นข้อมูลหลายมิติ

ข้อมูล data frame เป็นข้อมูลที่ใช้บ่อยในงานสถิติ R มอง data frame เป็น object หนึ่งที่มี row และ column เหมือนเป็น spreadsheet อันหนึ่ง หลักสำคัญก็คือ ค่าหรือ value ขอตัวแปรเดียวกันจะต้องอยู่ใน column เดียวกัน รูปแรกแสดงการจัดข้อมูลที่ไม่ถูกต้อง เพราะจัด วาง response time ของสามกลุ่มแยกจากกันเป็นสามคอลัมน์ ในการจัด data frame ของข้อมูล นี้ จะต้องจัดแบบรูปสอง คือนำ response time ไว้ในคอลัมน์เดียวกันแล้วให้ค่ากลุ่มต่างๆ เป็นค่า ในคอลัมน์ Treatment

Control	Pre-heated	Pre-chilled	
6.1	6.3	7.1	
5.9	6.2	8.2	
5.8	5.8	7.3	
5.4	6.3	6.9	

Response	Treatment
6.1	Control
5.9	Control
5.8	Control
5.4	Control
6.3	Pre-heated
6.2	Pre-heated
5.8	Pre-heated
6.3	Pre-heated
7.1	Pre-chilled
8.2	Pre-chilled
7.3	Pre-chilled
6.9	Pre-chilled

รูปจาก Crawley, Michael J. (2005). Statistics: An Introduction using R

เราสามารถรวมข้อมูลหลายๆ vector เป็น dataframe ได้ โดยมองแต่ละ vector เป็น ข้อมูลแต่ละคอลัมน์ เช่น ในตัวอย่างข้างล่างที่รวมเว็กเตอร์ gend, age, pref เข้าด้วยกันด้วยคำสั่ง data.frame แล้วนำ vector ที่ต้องการมาประกอบกันโดยหนึ่ง vector จะแทนหนึ่งคอลัมน์ใน data frame ที่สร้างขึ้น

ข้อมูลใน data frame สามารถอ้างถึงโดยการเลือกระบุคอลัมน์ที่ต้องการได้ เช่น x\$age หมายถึงนำเฉพาะข้อมูลในคอลัมน์ age ของ data frame x มาใช้ หรืออ้างโดยใช้ x[['age']] แทน ก็ได้ หากต้องการอ้างถึงข้อมูลแต่ละตัวดีอ้างผ่าน subscription ต่อ เช่น x\$gend[4] หรือ x[['gend']][4] มีค่าเป็น 'male' ตามตัวอย่างข้อมูลข้างล่างนี้

```
> gend = c('male','female','female','female','female','female','male','male','female')
> age = c('8-15','16-35','36-50','16-35','16-35','36-50','36-50','36-50','8-15')
> pref = c('red','green','blue','yellow','red','blue','green','blue','red')
> x = data.frame(gend,age,pref)
> X
      gend
                age
                          pref
      male 8-15
1
                           red
2 female 16-35 green
3 female 36-50
                        blue
      male 16-35 yellow
4
5 female 16-35
                            red
6 female 36-50
                          blue
7
      male 36-50 green
8
      male 36-50
                          blue
9 female 8-15
                            red
```

การอ่านข้อมูลจากไฟล์

เพื่อความสะดวก เราสามารถเตรียมข้อมูลโดยพิมพ์เข้าใน Excel ก่อนได้ จากนั้น save ให้ อยู่ในรูปที่สามารถนำเข้าในโปรแกรม R ได้ เช่น เก็บไฟล์แบบ tab delimited จากนั้นจึงอ่าน ข้อมูลเข้ามาเป็น data frame โดยใช้คำสั่ง read.table

```
>x1 <- read.table("c:/temp/data.txt")
```

ในกรณีที่ไฟล์ข้อมูลไม่ได้แยกข้อมูลด้วย tab แต่ใช้สัญลักษณ์อื่นแทน เช่น : ให้เติม option sep=":" ด้วย เช่น

```
>x1 <- read.table("c:/temp/data.txt", sep=":")
```

หากข้อมูลที่เตรียมไว้มี ข้อมูลบรรทัดแรกเป็น header ที่บอกชื่อตัวแปรแต่ละคอลัมน์ เวลาอ่าน ข้อมูลเข้าใน data frame ก็ต้องใส่ option header=TRUE ด้วย หรือหากเก็บข้อมูลเป็นแบบ csv ก็ให้ใช้คำสั่ง read.csv แทน

คำสั่ง read.table หรือ read.csv จึงเหมาะกับการอ่านข้อมูลที่จับเก็บเป็นตารางอยู่แล้ว เมื่ออ่านเข้ามาก็จะเป็น data frame ที่แต่ละคอลัมน์คือข้อมูลของตัวแปรเดียวกัน เช่น เพศ คะแนน อายุ ส่วนข้อมูลแต่ละแถวจะเป็นข้อมูลรายการแต่ละรายการ

กรณีที่ข้อมูลเป็นไฟล์ข้อความหรือ text สามารถใช้คำสั่ง readLines เพื่ออ่านข้อความเข้า มาทีละย่อหน้าเข้ามาเป็น string ข้อมูลทั้งหมดจะเป็น vector ของ string แต่ละตัวคือหนึ่งย่อหน้า encoding เป็น option สำหรับระบุ character encoding ของไฟล์นั้น

>text <- readLines("c:/temp/DH.txt",encoding="UTF-8")

text[i] จะเป็นข้อความย่อหน้าที่ i นอกจากคำลั่ง readLines ยังมีคำสั่ง scan อ่านข้อมูลจากไฟล์ >text <- scan(file="c:/temp/ DHl.txt", what="character", sep="\n")

อ่านข้อมูลลักษณะเดียวกัน ดู \n หรือการขึ้นย่อหน้าใหม่เป็นตัวแยกข้อมูล อ่านข้อมูลตัวอักษรไป เก็บทีละย่อหน้า ผลที่อาจต่างจาก readLines คือ กรณีที่ย่อหน้านั้นไม่มีข้อความอะไร คือมีการกด Enter มากกว่าหนึ่งหน คำสั่งหลังจะไม่อ่าน NULL เข้ามา

กรณีที่ต้องการอ่านข้อมูลเข้ามาทีละประโยค ก็สามารถใช้คำสั่งเดิม แต่ให้ sep="." แทน สิ่งที่ได้คือเว็กเตอร์ของประโยคที่ระบุด้วย full stop ทั้งหมดในไฟล์นั้น แต่ถ้าต้องการอ่านเข้ามาที ละคำ ก็ให้ละ option "sep" ออกไป

```
>text <- scan(file="c:/temp/DHl.txt", what="character", sep=".")
```

```
>text <- scan(file="c:/temp/DHl.txt", what="character")
```

ถ้าต้องการอ่านไฟล์มากหนึ่งไฟล์ ให้อ่านชื่อไฟล์ทั้งหมดออกมาก่อน จากนั้นวนอ่านไฟล์ ไปจนหมด list.files จะหาชื่อไฟล์ตามที่ระบุใน pattern เก็บชื่อรายการไฟล์ for เป็นคำสั่งวน รอบจาก 1 ถึง จำนวนไฟล์ทั้งหมด แล้วเอาชื่อไฟล์แต่ละไฟล์มาอ่านข้อมูลไปเก็บไว้ที่ list.data >list.filenames<-list.files(pattern=".csv\$")

>list.data<-list()

>for (i in 1:length(list.filenames))

{ list.data[[i]]<-read.csv(list.filenames[i]) }

ข้อมูลแต่ละไฟล์จะเก็บที่ list.data[i] ถ้าต้องการเข้าถึงข้อมูลแต่ละแถวในไฟล์นั้น สามารถเรียกผ่าน list.data[[i]][j] เช่น list.data[[2]][5] คือไฟล์ที่สองบรรทัดที่ห้า

นอกจากการสั่งวนรอบให้อ่านไฟล์ เราสามารถใช้อีกวิธีหนึ่งคือ lapply เป็นคำสั่งให้ apply ฟังก์ชั่นที่สร้างขึ้นกับ list ที่ให้ รูปแบบคำสั่งคือ lapply(list, function(x) รายละเอียดที่ ต้องการให้ทำ) คำสั่งข้างล่างนี้จึงได้ผลเหมือนกับ for loop ข้างบน

list.data <- lapply(list.filenames, function(x) scan(file=x, what="character", sep="\n"))

การเลือกใช้ข้อมูลบางส่วน

เมื่อเรานำข้อมูลเข้าเป็น data frame และต้องการใช้ข้อมูลเฉพาะส่วน เช่น ใช้แถวที่ ข้อมูลเฉพาะตัวอย่างจากเพศชาย เราสามารถกำหนด subset เฉพาะที่ต้องการได้ โดยใช้คำสั่ง subset ในตัวอย่างข้างล่าง เป็นการนำข้อมูล csv จากไฟล์ Data-scoring.csv และพิมพ์ข้อมูล 20 รายการแรกออกมาดู จากนั้นดึงเฉพาะข้อมูลที่ได้จากกลุ่มตัวอย่างเพศหญิง โดยใช้คำสั่ง subset(data, Sex == 'หญิง') คือดึงรายการจาก Data เฉพาะแถวที่ในคอลัมน์ Sex มีค่าเป็น 'หญิง' เก็บในตัวแปรชื่อ data.sub1 จากนั้นพิมพ์ 10 รายการแรกออกมา ก็จะเห็นว่า data.sub1 เป็นอย่างที่ต้องการ (การใช้ข้อมูลเข้าภาษาไทย ให้เก็บไฟล์เข้ารหัสแบบ utf8) > data = read.csv("/Users/macbook/Cloud/Dropbox/wrk/research_tech/Datascoring.csv",header=TRUE,row.names=1) > data[120]

-	uucu[1.20,]										
	Sex Age	Edu Sco	oreExp Re	eadScore Rea	ad Wr	iteScore Wri	te Li	stenScore Li	sten Sp	eakScore Spe	ak
1	ทญิง 36-50	ปริญญาโท	52	47.33333	C2	45.83333	C1	47.666667	C2	48.00000	C2
2	ชาย 36-50	ปริญญาโท	46	45.20000	C1	47.55000	C2	45.000000	C1	44.00000	C1
3	ชาย 26-35	ปริญญาโท	45	43.80000	C1	44.60000	C1	45.333333	C1	45.50000	C1
4	ทญิง 36-50	ปริญญาโท	39	46.66667	C2	41.45000	C1	39.166667	C1	30.83333	B2
5	ทญิง 26-35	ปริญญาตรี	25	39.65000	C1	39.40000	C1	23.833333	B2	13.41667	B1
6	ทญิง 36-50	ปริญญาโท	59	46.60000	C2	45.66667	C1	45.666667	C1	43.00000	C1
7	ทญิง 36-50	ปริญญาตรี	2	2 5.15000	A1	0.00000	C1	6.166667	AZ	0.00000	C1
8	ทญิง 26-35	ปริญญาโท	42	43.58333	C1	40.66667	C1	43.000000	C1	6.00000	A2
9	ทญิง 36-50	ปริญญาโท	56	48.00000	C2	48.00000	C2	46.333333	C2	45.83333	C1
10)ชาย 26-35	ปริญญาโท	43	44.25000	C1	46.00000	C2	45.166667	C1	46.75000	C2
11	1 ชาย 26-35	ปริญญาตรี	48	44.86667	C1	44.35000	C1	42.000000	C1	43.50000	C1
12	2 ชาย 26-35	ปริญญาตรี	48	44.86667	C1	44.35000	C1	42.000000	C1	43.50000	C1
13	3 ทญิง 51-65	ปริญญาตรี	22	2 20.85000	B1	12.93333	A2	19.333333	B1	15.75000	B1
14	4 ทญิง 26-35	ปริญญาตรี	16	6 40.33333	C1	15.00000	B1	11.000000	A2	11.75000	A2
15	5 ทญิง 36-50	ปริญญาตรี	41	L 44.08333	C1	30.86667	B2	41.000000	C1	19.66667	B1
16	ธิทญิง 36-50	ปริญญาโท	45	45.66667	C1	43.70000	C1	41.833333	C1	29.33333	B2
17	7 ทญิง 36-50	ปริญญาตรี	16	5 45.33333	C1	48.00000	C2	16.166667	B1	39.25000	C1
18	8 ชาย 36-50 เ	ปริญญาเอก	63	48.00000	C2	48.00000	C2	48.000000	C2	48.00000	C2
19	9 ทญิง 36-50	ปริญญาโท	35	46.60000	C2	48.00000	C2	44.333333	C1	47.00000	C2
20	0 ทญิง 36-50	ปริญญาโท	28	43.60000	C1	41.20000	C1	29.833333	B2	15.75000	B1

> data.sub1 <- subset(data,Sex == 'หญิง') > data.sub1[1:10,] Sex Aae Edu ScoreExp ReadScore Read WriteScore Write ListenScore Listen SpeakScore Speak 1 หญิง 36-50 ปริญญาโท 52 47.33333 C2 45.83333 C1 47.666667 C2 48,00000 C2 4 หญิง 36-50 ปริญญาโท 39 46.66667 C2 41.45000 C1 39.166667 C1 30.83333 B2 5 หญิง 26-35 ปริญญาตรี 25 39.65000 C1 39.40000 C1 23.833333 R2 13.41667 B1 6 หญิง 36-50 ปริญญาโท 59 46.60000 C2 45.66667 C1 45.666667 C1 43.00000 C1 7 หญิง 36-50 ปริญญาตรี 2 5.15000 A1 0.00000 C1 6.166667 AZ 0.00000 C1 8 หญิง 26-35 ปริญญาโท 42 43.58333 C1 40.66667 C1 43.000000 C1 6.00000 A2 9 หญิง 36-50 ปริญญาโท 56 48,00000 C2 48.00000 C2 46.333333 C2 45.83333 C1 13 หญิง 51-65 ปริญญาตรี 22 20.85000 B1 12.93333 AZ 19.333333 B1 15.75000 B1 14 หญิง 26-35 ปริญญาตรี 16 40.33333 C1 15.00000 **R1** 11.000000 A2 11.75000 AZ 15 หญิง 36-50 ปริญญาตรี 41 44.08333 C1 30.86667 B2 41.000000 C1 19,66667 **B1**

การคำนวณสถิติพื้นฐาน

์ โปรแกรม R มีฟังก์ชั่นพื้นฐานสำหรับคำนวณสถิติ ดังนี้

- mean(x) หาค่าเฉลี่ยในข้อมูลที่เก็บไว้ใน vector x
- median(x) หาค่า median ในข้อมูลที่เก็บไว้ใน vector x
- sd(x) หาค่า SD ในข้อมูลที่เก็บไว้ใน vector x
- var(x) หาค่า variance ในข้อมูลที่เก็บไว้ใน vector x

ตัวอย่างเช่น

```
> x=c(1,2,4,3,5,9,12,11,10,7)
> mean(x)
[1] 6.4
> median(x)
[1] 6
> sd(x)
[1] 3.949684
> var(x)
[1] 15.6
```

ใน R ไม่มี function mode แต่เราสามารถใช้คำสั่ง which.max ได้ เช่น >which.max(table(x)) คือแปลงข้อมูลเป็นตารางและนับความถี่ของแต่ละหมวด หาตัวที่มีค่าสูงสุดออกมา

```
> x=c(1,2,1,2,3,2,1,4,5,2,3,4,5,2)
> table(x)
x
1 2 3 4 5
3 5 2 2 2
> which.max(table(x))
2
2
>
```

คำสั่ง summary ใช้กับ vector เพื่อแสดงค่าสรุปข้อมูลซึ่งจะเห็นค่า min, max, median, mean, 1st quarter, และ 3rd quarter

> summary(x)
Min. 1st Qu. Median Mean 3rd Qu. Max.
1.000 2.000 2.000 2.643 3.750 5.000

หากต้องการดูข้อมูลในรูป box graph ให้ใช้คำสั่ง boxplot บางครั้งเรียกว่า "hinge plot" หรือ "box and whiskers plot" ซึ่งจะแสดงกล่องบอก 1st Quartile, Median, และ 3rd Quartile (เส้นหนาคือ median) และมีเส้น whisker บน ล่าง ที่บอกขอบเขตบน ล่าง ที่ข้อมูลยังเกาะกลุ่ม อยู่ ตัวอย่างเช่น

```
> data = read.csv("/Users/macbook/data/survey-scoring.csv", header=TRUE,
row.names=1)
```

อ่านข้อมูลตอบแบบสอบถามจากไฟล์ csv โดยมี header row

```
> data.ba = subset(data,Edu == 'ปริญญาตรี')
```

เลือกเฉพาะข้อมูลส่วนที่คนตอบระบุว่าเป็น ปริญญาตรี

> summary(data.ba\$ReadScore)

Min. 1st Qu. Median Mean 3rd Qu. Max.

1.00 35.94 42.42 38.35 45.28 48.00

แสดงผลสรุปข้อมูลคะแนนส่วนการเขียนของคนจบปริญญาตรี

```
>boxplot(data.ba$ReadScore,data.ba$WriteScore,data.ba$ListenScore,data.ba$SpeakS
core, names=c('read','write','listen','speak'),xlab='BA Degree')
```

สั่งสร้าง boxplot ข้อมูลคะแนนอ่าน เขียน ฟัง พูด ของคนจบปริญญาตรี names ใช้กำหนดชื่อ ข้อมูลที่ให้แสดง xlab คือ label หรือป้ายข้อความในแกนนอน เช่น ขีดบนขีดล่าง คือ ค่า max และ min ทั้งนี้ไม่รวมส่วนที่ถูกมองว่าเป็น outliner ที่แสดงเป็นจุดวงกลม เส้นขีดหนาตรงกลาง คือ ค่า median ส่วนด้านบนกล่องและด้านล่างกล่องคือข้อมูลที่ตำแหน่ง 3/4 และ 1/4 ตามลำดับ



BA Degree

การสุ่มข้อมูล

เราสามารถสุ่มข้อมูลจากเวกเตอร์ได้ด้วยคำสั่ง sample() ในตัวอย่างข้างล่างนี้ เป็นการสั่ง ให้สุ่มข้อมูลมาสามตัวจากเวกเตอร์ x ที่พิมพ์ข้อมูลเข้าไปเอง 16 ตัว หากไม่ระบุจำนวนจะหมายถึง ให้สุ่มมาจนกว่าจะหมด เช่น sample(x) จะได้ข้อมูลทั้งหมดออกมาแบบสุ่ม sample(x,3) ครั้งที่ สองจะเห็นว่าได้ข้อมูลคนละตัวกับครั้งแรก

```
> x = scan()
1: 1 3 5 2 7 9 23 12 34 76 45 34 23 12 34 55
17:
Read 16 items
> x
[1] 1 3 5 2 7 9 23 12 34 76 45 34 23 12 34 55
> sample(x, 3)
[1] 76 34 34
> sample(x)
[1] 7 2 76 55 12 23 12 9 5 1 3 34 23 34 34 45
> sample(x,3)
[1] 7 3 9
```

การติดตั้ง package

เราสามารถติดตั้ง package เพิ่มเติมในโปรแกรม R ได้ โดยที่ package เป็นชุดคำสั่ง หรือฟังก์ชั่นต่างๆ ที่มีผู้เขียนเพิ่มเติมและต้องการแบ่งปันให้ผู้อื่นได้ใช้ด้วย สามารถเข้าไปดูที่ <u>http://</u> <u>cran.r-project.org/web/views/</u> หรือค้นหา package ด้วยคำค้นที่ต้องการใน <u>http://rseek.org/</u> ในหนังสือ Analyzing Linguistic Data (Baayan 2008) ผู้เขียนได้สร้าง package ชื่อ languageR สำหรับใช้ประกอบการอธิบายในหนังสือ เราสามารถติดตั้ง package ของหนังสือเล่มนี้โดยพิมพ์คำ สั่ง install.packages ตามตัวอย่าง

 install.packages(c("rpart", "chron", "Hmisc", "Design","Matrix", "lme4", "coda", "e1071", "zipfR", "ape", "languageR"), repos = "<u>http://cran.r-project.org</u>")

โปรแกรมจะเชื่อมต่ออินเทอร์เน็ตไปที่ cran.r-project.org เพื่อดาวน์โหลด package ต่างๆ ที่ระบุ เมื่อเสร็จสิ้นจะเห็นรายงานผลพร้อมทั้งบอกว่า package นั้นดาวน์โหลดมาไว้ที่ folder ไหน ส่วนตัว package ต่างๆที่ติดตั้งนั้น จะอยู่ที่ folder c:\Program Files\R\R-2.11.1\library

```
R Console
                                                                      Content type 'application/zip' length 953062 bytes (930 Kb)
opened URL
downloaded 930 Kb
trying URL 'http://cran.r-project.org/bin/windows/contrib/2.11/languageR 1.0.zi$
Content type 'application/zip' length 2379127 bytes (2.3 Mb)
opened URL
downloaded 2.3 Mb
package 'gee' successfully unpacked and MD5 sums checked
package 'rpart' successfully unpacked and MD5 sums checked
package 'chron' successfully unpacked and MD5 sums checked
package 'Hmisc' successfully unpacked and MD5 sums checked
package 'Design' successfully unpacked and MD5 sums checked
package 'Matrix' successfully unpacked and MD5 sums checked
package 'lme4' successfully unpacked and MD5 sums checked
package 'coda' successfully unpacked and MD5 sums checked
package 'e1071' successfully unpacked and MD5 sums checked
package 'zipfR' successfully unpacked and MD5 sums checked
package 'ape' successfully unpacked and MD5 sums checked
package 'languageR' successfully unpacked and MD5 sums checked
The downloaded packages are in
        C:\Users\Fujitsu\AppData\Local\Temp\RtmpvVrkpD\downloaded packages
>
```

อีกวิธีการหนึ่งคือการ install package ที่มีใน CRAN ผ่านเมนูของโปรแกรม R ให้เรียก "Package & Data" - "package Installer" แล้วค้นชื่อ package ที่ต้องการ เมื่อพบ package นั้นก็เลือก package ที่ต้องการ และกด "Install Selected"

Packages Repository			
CRAN (binaries)	٥		
Get List	Binary Format Packages	Q som	8

Package	Installed Version	Repository Version
fastSOM		0.9
glassomix		1.2
popsom		2.3
Rsomoclu		1.4.1
SensoMineR		1.20
som	0.3-5	0.3-5
soma		1.1.1
someKfwer		1.2
someMTP		1.4.1
somplot		1.6.4
testthatsomemore		0.1
Install Location		

At System Level (in R framework)	Install Selected
At User Level In Other Location (Will Be Asked Upon Installation)	Install Dependencies
As defined by .libPaths()	Update All

เมื่อติดตั้ง package LanguageR เรียบร้อยแล้ว เราสามารถ load package โดยคำสั่ง library(languageR) โดยที่ languageR คือซื่อของ package ที่ต้องการโหลด

```
> library(languageR)
Loading required package: lattice
Loading required package: zipfR
Loading required package: Matrix
Attaching package: 'Matrix'
The following object(s) are masked from 'package:base':
    det
Loading required package: lme4
Attaching package: 'lme4'
The following object(s) are masked from 'package:stats':
    AIC
> |
```

หลังจากติดตั้ง package languageR แล้วโหลดเข้ามาแล้ว จะมีข้อมูลตัวอย่างส่วนหนึ่ง load เข้าไป เช่น ข้อมูล dative alternation in English ของ Bresnan et al. (2007) เมื่อสั่ง ให้แจง รายการ verb ในข้อมูลออกมา 10 รายการ

>	head(verbs, n=10)				
	RealizationOfRec	Verb	AnimacyOfRec	AnimacyOfTheme	LengthOfTheme
1	NP	feed	animate	inanimate	2.6390573
2	NP	give	animate	inanimate	1.0986123
3	NP	give	animate	inanimate	2.5649494
4	NP	give	animate	inanimate	1.6094379
5	NP	offer	animate	inanimate	1.0986123
6	NP	give	animate	inanimate	1.3862944
7	NP	pay	animate	inanimate	1.3862944
8	NP	bring	animate	inanimate	0.000000
9	NP	teach	animate	inanimate	2.3978953
10) NP	give	animate	inanimate	0.6931472

คอลัมน์แรกบอกว่า ผู้รับ (recipient) อยู่ในรูปอะไร เป็น NP (Mary gave *John* a book) หรือ PP (Mary gave a book *to John*) คอลัมน์สุดท้ายบอกความยาวประโยคเป็นเลข log ฐาน e log(14) = 2.639 ประโยคแรกจึงมีความยาว 14 คำ

การ save ข้อมูลลงไฟล์

เราสามารถเก็บข้อมูลที่เป็น data frame ที่มีอยู่โดย save เก็บเป็นไฟล์ได้โดยใช้คำสั่ง write.table

>write.table(verbs, "c:/dativeS.txt") สั่งให้เก็บ data frame "verbs" เป็นไฟล์ชื่อ dativeS.txt >write.table(verbs, "/Users/macbook/Documents/dativeS.txt") [ตัวอย่างบนเครื่อง Mac]

ข้อมูลในไฟล์ที่ได้ เป็นแต่ละบรรทัดแทนแต่ละรายการ มี header และข้อความอยู่ใน เครื่องหมายคำพูด

"RealizationOfRec" "Verb" "AnimacyOfRec" "AnimacyOfTheme" "LengthOfTheme"

"1" "NP" "feed" "animate" "inanimate" 2.6390573

"2" "NP" "give" "animate" "inanimate" 1.0986123

"3" "NP" "give" "animate" "inanimate" 2.5649494

"4" "NP" "give" "animate" "inanimate" 1.6094379

"5" "NP" "offer" "animate" "inanimate" 1.0986123

.....

เมื่อมีข้อมูลเป็น text ไฟล์ (แบบที่ได้จากการ save ข้างบน) เราสามารถนำข้อมูลนั้นเข้า มาใน R ได้

>verbs = read.table("c:/dativeS.txt", header=TRUE) สั่งให้อ่านตารางข้อมูลในไฟล์ datives.txt ซึ่งเป็นไฟล์ที่มี header บรรทัดแรกสุด

ถ้าต้องการ save ข้อมูลลงไฟล์รูปแบบ csv ใช้คำสั่ง write.csv >write.csv(data, "d:/temp/xxx.csv")

การสร้าง contingency table จากข้อมูล data frame ที่มี เช่น ในตัวอย่าง data frame "verbs" ข้อมูล RealizationOfRec มี NP, PP และข้อมูล AnimacyOfRec มี animate กับ inanimate เราสามารถสร้างตารางนับความถี่ของสองตัวแปรนี้ได้ด้วยคำสั่ง xtabs แบบข้าง ล่าง เครื่องหมาย ~ หมายถึง "is a function of" ซึ่งในตัวอย่างนี้มีสองตัวแปร ส่วน data บอกว่า ใช้ข้อมูลจาก data frame ไหน จากข้อมูลนับได้ว่า NP ที่เป็น animate มี 521 เป็น inanimate มี 34 เป็นต้น

ถ้าอยากรู้ว่าในตัวแปรนั้นมีค่าอะไรอยู่บ้าง สามารถใช้คำสั่ง levels

```
> levels(verbs$RealizationOfRec)
[1] "NP" "PP"
```

verbs.xtabs = xtabs(~ AnimacyOfRec + RealizationOfRec, data = verbs, subset = AnimacyOfTheme != "animate") เป็นการสร้าง 2x2 contingency table ที่สลับตัวแปรกับ ตัวอย่างข้างบน และเพิ่มเงื่อนไขว่าใน data verbs ให้เอาเฉพาะส่วนหรือ subset ที่ AnimacyOfTheme ไม่ใช่ animate แล้วเก็บตารางที่ได้ลงใน verbs.xtabs เครื่องหมาย != หมายถึง not equal (หากพิมพ์ enter ก่อนจบคำสั่ง เราจะเห็นเครื่องหมาย + ในบรรทัดสอง แสดงว่าเป็นการขึ้นบรรทัดใหมโดยที่ยังเป็นคำสั่งต่อเนื่องจากบรรทัดบน)

> sum(verbs.xtabs) สั่งให้หาค่าผลรวมในตาราง verbs.xtabs ได้ค่าเป็น 897 เมื่อสั่ง ให้คูณ 100 หารด้วยค่าผลรวมนี้ ก็ได้เป็นเปอร์เซ็นต์ของแต่ละช่อง (รวมกันทั้งสี่ช่องเป็น 100%)

> mean(verbs[verbs\$AnimacyOfRec == "animate",]\$LengthOfTheme) หาค่า mean เฉพาะข้อมูลส่วนที่เป็น LengthOfTheme ใน verbs แถวที่มีตัวแปร AnimacyOfRec เป็น "animate" จะได้คำตอบเป็น 1.540278

```
> mean(verbs[verbs$AnimacyOfRec == "animate", ]$LengthOfTheme)
[1] 1.540278
```

> mean(verbs[verbs\$AnimacyOfRec != "animate",]\$LengthOfTheme) หาค่า mean ข้อมูลเดียวกันที่มีตัวแปร AnimacyOfRec เป็น "inanimate" ได้คำตอบ 1.071130

```
> mean(verbs[verbs$AnimacyOfRec != "animate", ]$LengthOfTheme)
[1] 1.071130
```

กรณีข้างบนนี้ ทำรวดเดียวได้ด้วยคำสั่ง tapply คือการสั่งให้ใช้ฟังก์ชั่นที่กำหนดกับข้อมูลที่เรา ต้องการแยกเป็นกลุ่มๆ ได้ เช่น

```
> tapply(verbs$LengthOfTheme, verbs$AnimacyOfRec, mean)
```

argument แรกเป็นข้อมูลที่ต้องการหาค่า argument สองเป็นค่าหรือ index ที่ใช้จำแนกกลุ่มข้อมูล ซึ่งในที่นี้มีสองค่า คือ "animate" กับ "inanimate" ส่วน argument สุดท้ายเป็น function ที่ ต้องการ apply ในที่นี้คือหาค่า mean ของ LengthOfTheme แยกตามกลุ่ม "animate" กับ "inanimate"

```
> tapply(verbs$LengthOfTheme, verbs$AnimacyOfRec, mean)
animate inanimate
1.540278 1.071130
```

การแสดงผลด้วยกราฟ

เราสามารถสั่งวาดกราฟแท่งโดยแจกแจงจำนวนข้อมูลที่มีในแต่ละกลุ่มได้ เช่น สมมติ เรา ป้อนข้อมูลเกรด a,b,c,d ของนักเรียนจำนวน 20 คนเก็บไว้ในตัวแปร x ผ่านทางแป้นพิมพ์ด้วยคำ สั่ง >x = scan(what="character") จากนั้นใช้คำสั่ง >barplot(table(x)) โปรแกรมจะแสดง กราฟแท่งของข้อมูลที่จัดลงตาราง a,b,c,d ให้เราได้

```
> x=scan(what="character")
1: a b b c a b c c c d
11: a d d c c b b b c c
21:
Read 20 items
> barplot(table(x))
```



หากต้องการแสดงสีที่ต่างกัน สามารถเติม option col=c("red", "green", "white", "blue") ดังนี้ >barplot(table(x), col=c("red", "green", "white", "blue"))



นอกจากคำสั่ง barplot เราสามาถใช้คำสั่ง plot สำหรับวาดกราฟแสดงจุดตำแหน่งที่มีค่าแกน x, แกน y ของข้อมูลที่มีสองตัวแปร >plot(x,y) โดยที่ x และ y เป็น vector ที่มีจำนวน element เท่ากันพร้อมทั้งระบุ ชื่อกราฟและชื่อแกน x,y ด้วยคำสั่ง

>plot(dataX, dataY, title="The title", xlab="X-axis label", ylab= "Y-axis label") แต่หากต้องการแสดงเป็นกราฟเส้น ให้เติม option type="l"

หากต้องการแสดงเป็นกราฟวงกลม ให้ใช้

```
>pie(table(x), col=rainbow(length(table(x))))
```

คำสั่ง hist() ใช้สร้างภาพ histogram ของข้อมูล เช่น

>hist(data.ba\$ReadScore)



Histogram of data.ba\$ReadScore

หากต้องการดูการกระจายเป็น curve มากกว่า histogram ให้ใช้ Kernal density plot >plot(density(data.ba\$ReadScore))





การสร้างกราฟวงกลม สามารถทำได้ด้วยคำสั่ง pie ซึ่งแสดงเป็นสัดส่วนร้อยละของข้อมูล ส่วนต่าง ๆ ดังตัวอย่างนี้ ข้อมูลรายการเฉพาะส่วนการศึกษา (คอลัมน์ชื่อ Edu) จะถูกนำมา แจกแจงตารางความถี่

>group <- table(data\$Edu)

>pie(group, col=brewer.pal(4,"Dark2"), radius=3)



ข้อมูลที่เป็น table สามารถดูว่ามีหมวดอะไรบ้างด้วยคำสั่ง dimnames() และอ้างถึงแต่ละตัวได้ ผ่าน subscription ของชื่อหมวดนั้น เช่น

>dimnames(group) [[1]] [1] "Bachelor" "Certificate" "Master" "PhD"

>group[['Master']]

[1] 136

library(RColorBrewer) เป็น package ที่ช่วยในการทำกราฟสีต่าง ๆ ได้ โดยระบุใน option col="....." แทนที่จะระบุสีพื้นอย่าง col=c("red","blue","green") ให้ใช้ col=brewer.pal(n, "setname") โดยที่ n เป็นจำนวนสีที่ต้องการใช้แสดง set-name เป็นชื่อ set ที่จะใช้ได้ เช่น Set1, Set2, Set3, Pastel1, Pastel2, Dark2, Greens, Greys, Blues, BuGn, BuPu, GnBu, Oranges, OrRd, PuBu, PuBuGn, PuRd, Purples, RdPu, Reds, YlGn, YlGnBu, YlOrBr, YlOrRd เป็นต้น

ถ้าต้องการใช้ font อื่นเช่น font สำหรับแสดงภาษาไทย ให้ load libary 'extrafont' จากนั้นใช้ option family = "TH Sarabun New" ในคำสั่งสร้างกราฟ font TH Sarabun New ก็จะถูกนำมาใช้

```
>pie(group, col=brewer.pal(4,"Dark2"), family="TH Sarabun New")
```

ถ้าต้องการสร้างกราฟแท่งแบบซ้อนหรือแบบเรียงสำหรับข้อมูลหลายชุดเพื่อดูเปรียบเทียบ ก็สามารถทำได้โดยระบุชุดข้อมูลที่ต้องใช้ให้ครบ ตัวอย่างเช่น

> counts <- table(data\$Speak, data\$Edu)</pre>

> counts

ต่ำกว่าปริญญาตรี ปริญญาตรี ปริญญาโท ปริญญาเอก

A1	4	23	6	1
A2	1	12	6	0
B1	1	18	7	1

B2	0	7	9	1
C1	3	55	65	10
C2	1	19	43	19

แปลงข้อมูลเป็นตารางนับคนที่มีระดับ skill ของการพูดต่างๆ (A1-C2) กับระดับการศึกษา เพื่อจะ ได้นำไปสร้างกราฟแท่งได้โดยใช้คำสั่ง barplot ข้อมูลตาราง counts นี้ เราต้องกำหนด label แกน นอนเอง เพราะข้อมูลชื่อคอลัมน์เป็นภาษาไทยไม่สามารถแสดงผลในกราฟได้ จึงกำหนด names.arg เป็นชื่อ degree ภาษาอังกฤษแทน

> barplot(counts, main="Speaking skill", xlab="Degree of Study", legend = rownames(counts), names.arg=c("Certificate", "Bachelor", "Master", "phD"))



Speaking skill

หากต้องการเป็นกราฟแท่งแบบเรียงต่อกัน ไม่ใช่แบบซ้อนทับกัน (stacked) ก็ให้เติมพารามิเตอร์ beside=TRUE > barplot(counts, main="Speaking skill", xlab="Degree of Study", legend = rownames(counts), names.arg=c("Certificate", "Bachelor", "Master", "phD"), beside=TRUE)



หากต้องการสร้างกราฟเส้น ให้ใช้คำสั่ง plot(VectorData, type="l") สำหรับกราฟเส้น และ plot(VectorData, type="0") สำหรับกราฟเส้นที่มีจุดกำกับค่าที่ plot ด้วย เช่น

> plot(sort(data\$SpeakScore),type="l")



สำหรับกราฟความสัมพันธ์ระหว่างตัวแปรสองตัว (x,y) ให้ใช้คำสั่ง plot เช่นกันโดยมีแหล่ง ข้อมูลสองส่วน plot(DataX, DataY) เช่น

> plot(data\$ScoreInt+data\$ScoreExp+data\$ScoreExpW, data\$SpeakScore) แกนนอนเป็นคะแนนที่ได้จากสามส่วนรวมกัน ScoreInt, ScoreExp, ScoreExpW แกนตั้งเป็น คะแนนส่วนการพูด SpeakScore



data\$ScoreInt + data\$ScoreExp + data\$ScoreExpW

นอกจากคำสั่งพื้นฐานการสร้างกราฟตามที่กล่าวมา ยังมี package ggplot2 ที่ใช้สำหรับ ช่วยทำกราฟ โดยคำสั่งที่ใช้จะเขียนตามหลักไวยากรณ์ของกราฟฟิก (grammar of graphics) ggplot สามารถทำกราฟพื้นฐานแบบที่กล่าวมาได้ แต่กำหนดองค์ประกอบอย่างเป็นระบบชัดเจน เช่น

```
>hist(Data$SpeakScore)
```

เขียนใหมเป็น

```
>ggplot(data, aes(x = SpeakScore)) + geom_histogram()
```

ggplot กำหนดข้อมูลที่ใช้คือจ้อมูลใน data ส่วน aes กำหนดแกนข้อมูล แล้วเติมองค์ประกอบ (+) ด้วยรูปกราฟแบบ histogram

ด้วยการกำหนดองค์ประกอบเพิ่มเติมได้นี้ คือการเติม + กราฟหลาย ๆ อย่าง ทำให้เราสา มาถสร้างกราฟหลายอย่างซ้อนกันได้ เช่น

> p<-ggplot(data, aes(x = SpeakScore, y = ListenScore))</pre>

```
> p <- p + geom_point(aes(color = SpeakScore)) + geom_smooth(model=lm)</pre>
```

> p

กำหนดข้อมูลแกน x และแกน y ให้ทำกราฟจุดแสดงความสัมพันธ์ของ x,y ที่มีสีตามความเข้มของ คะแนน SpeakScore และวาดเส้นโค้งแสดงความสัมพันธ์เชิงเส้นของข้อมูล



หากต้องการสร้างกราฟแท่ง ให้ระบุข้อมูลหลักที่จำแนกใน × ข้อมูลย่อยที่จำแนกใน fill ด้วยอะไร และเติมรูป geom_bar โดยให้ใช้ stat เป็น count คือนับจำนวนในข้อมูลนั้น (ทำให้ไม่



หากต้องการ font อื่น ๆ เช่น แสดงภาษาไทย ให้เติม theme และระบุ font ที่ต้องการ

p <- p + theme(text=element_text(size=16, family="TH Sarabun New")) ถ้าต้องการกราฟแท่งแบบที่ข้อมูลย่อยเรียงคู่กัน ก็ให้เติม position_dodge() เข้าไป >p <- ggplot(data, aes(x=Edu, fill=Speak))</pre>

>p <- p + geom bar(stat="count", position=position dodge())



หากต้องการแสดงผลคะแนนเป็น boxplot แยกตามกลุ่มก็ทำได้ในลักษณะเดียวกัน ดังนี้ >p <- ggplot(data, aes(x=Edu, y=SpeakScore))



ถ้าต้องการสร้างกราฟหลายอันมาเทียบกันตามกลุ่ม ก็สามารถทำได้โดยใช้ facet_grid แล้วระบุตัวแปรที่ต้องการใช้แยกรูปกราฟโดยระบุข้อมูล row ~ column ตัวอย่าง Edu ~ . จะ แสดงกราฟตาม row แยกตามวุฒิการศึกษา แต่หากเป็น . ~ Edu จะแสดงกราฟกลับกันเป็นสี่รูป ตามคอลัมน์ ถ้าต้องการใช้สองตัวแปรแยกกลุ่ม ก็ระบุทั้ง row ~ column ตามตัวอย่างข้างล่างนี้

>p<-ggplot(data,aes(x = Sex, y=WriteScore+SpeakScore))

>p<-p+geom boxplot()+facet grid(. ~ Edu)

>p



>p <- ggplot(data,aes(SpeakScore, ListenScore)) >p <- p + geom_point(aes(color = SpeakScore)) + geom_smooth(model=lm) +



ถ้าต้องการเติมข้อความบรรยายชื่อกราฟ ชื่อแกนข้อมูล ก็สามารถเติม +ggtitle("...") หรือ +xlab("...") หรือ ylab("...") หรือ labs(title="...", xlab="...", ylab="...")

การคำนวณค่า Z-score

ในการเก็บข้อมูลจำนวนหนึ่งมาแจกแจง หากเราเก็บข้อมูลไปเรื่อยๆ เป็นจำนวนมาก หาก ข้อมูลนั้นมีการกระจายตัวอย่างสมดุลหรือที่เรียกว่ามี normal distribution เมื่อเราคำนวณค่าเฉลี่ย จากข้อมูลนั้น ค่าเฉลี่ยจะเป็นค่าที่อยู่ตรงกลางของการกระจายตัว ลักษณะนี้ทางสถิติเรียกว่า central limit theory เมื่อวาดเป็นกราฟก็จะได้เป็นรูประฆังคว่ำ และจากกราฟนี้ เมื่อคำนวณ พื้นที่ที่อยู่ระหว่างตำแหน่งที่ -1.96*sd กับ 1.96*sd พื้นที่นั้นคิดเป็นปริมาณ 95% คือเหลือปลาย ซ้ายและปลายขวาข้างละ 2.5%


Crawley, Michael J. (2005: Kindle Location 1124)

การคำนวณค่า Z-score อาศัยหลักการที่กล่าวมาข้างต้น ปรับค่าตัวเลขหรือ normalize โดยใช้สูตร z = (y - y_bar) / sd หากค่า y มีค่ามากกว่าค่าเฉลี่ยจะมีค่าเป็นบวก หากน้อยกว่าค่าเฉลี่ยจะมีค่า เป็นลบ ค่า z-score ที่ได้นี้จะบอกถึง probability ที่ตำแหน่งนั้นได้ เช่น ในตัวอย่างของ Crawley (2005) ที่เก็บข้อมูลความสูงของกลุ่มตัวอย่างมีค่าเฉลี่ยเป็น 170 และมีค่า sd เป็น 8 หากเรา ต้องการรู้ว่าจำนวนคนที่มีความสูงน้อยกว่า 160 คำนวณได้จากค่า z = (160-170)/8 = -1.25 เมื่อ ใช้คำสั่ง >pnorm(-1.25) ได้ค่า 0.1056498 ก็ตีความได้ว่ามีประมาณ 10% หากอยากรู้ว่าคนที่มี ความสูงมากกว่า 185 มีเท่าไร ก็คำนวณจากค่า z = (185-170)/8 = 1.875 และสั่งหาค่า pnorm(1.875) ได้เท่ากับ 0.9696036 ตีความได้ว่าคนที่สูงกว่า 185 มีจำนวน (1-0.9696036)*100 คือประมาณ 4%

ใน R เราสามารถแปลงค่าใน vector เป็น z-score ได้ด้วยคำสั่ง >scale() ดังในตัวอย่าง ที่แสดงข้างล่างนี้ ค่าที่ได้คือ z-score ของข้อมูลแต่ละตัวตามลำดับ หากต้องการรู้ค่า probability ของแต่ละค่าของ z-score ก็ให้ใช้คำสั่ง >pnorm() ดังตัวอย่างล่างที่ค่าชุดแรกเป็น z-score ของ แต่ละข้อมูลที่แปลงได้ และค่าชุดที่สองเป็นค่า probability ของแต่ละข้อมูล

```
> x=c(165,179,168,178,180,155,153,154,167,171,165,155,168,172,174,173,175)
> mean(x)
[1] 167.7647
> sd(x)
[1] 8.93399
> scale(x)
             [,1]
 [1,] -0.30945926
 [2,] 1.25758974
 [3,] 0.02633696
 [4,] 1.14565767
 [5,] 1.36952181
 [6,] -1.42877997
 [7,] -1.65264411
 [8,] -1.54071204
 [9,] -0.08559511
[10,] 0.36213317
[11,] -0.30945926
[12,] -1.42877997
[13,]
      0.02633696
[14,] 0.47406524
[15,] 0.69792939
[16,] 0.58599731
[17,] 0.80986146
attr(,"scaled:center")
[1] 167.7647
attr(,"scaled:scale")
[1] 8.93399
> y=scale(x)
> pnorm(y)
            [,1]
 [1,] 0.37848610
 [2,] 0.89572992
 [3,] 0.51050571
 [4,] 0.87403159
 [5,] 0.91458189
 [6,] 0.07653374
 [7,] 0.04920166
 [8,] 0.06169344
 [9,] 0.46589414
[10,] 0.64137374
[11,] 0.37848610
[12,] 0.07653374
[13,] 0.51050571
[14,] 0.68227331
[15,] 0.75738932
[16,] 0.72106134
[17,] 0.79099010
attr(,"scaled:center")
[1] 167.7647
attr(,"scaled:scale")
```

```
[1] 8.93399
```

การคำนวณค่า t-test

T-test เป็นสถิติพื้นฐานตัวหนึ่งที่ใช้กันแพร่หลาย สามารถใช้เปรียบเทียบค่าเฉลี่ยของกลุ่ม ตัวอย่างกับค่าเฉลี่ยของประชากรเพื่อตัดสินว่ากลุ่มตัวอย่างนั้นมาจากประชากรนั้นหรือไม่ หรือ เปรียบเทียบค่าเฉลี่ยของกลุ่มตัวอย่างสองกลุ่มว่ามาจากประชากรเดียวกันหรือไม่ t-test เป็นที่นิยม ใช้ในกรณีที่กลุ่มตัวอย่างมีจำนวนไม่มาก เช่น น้อยกว่า 30 ทำให้การคำนวณโดยใช้ z-test ไม่ สามารถใช้ได้ โปรแกรม R มีฟังก์ชั่นพื้นฐานสำหรับคำนวณสถิติ t.test ดังตัวอย่างต่อไปนี้

ในตัวอย่างข้างล่าง เป็นการหา t-test ของข้อมูล DurationPrefixNasal ว่ามี mean ที่ ต่างจาก mean การศึกษาครั้งก่อนที่มีค่า 0.053 หรือไม่ ได้ค่า mean ข้อมูลชุดนี้เท่ากับ 0.04981508 95% confidence interval อยู่ระหว่าง 0.04551370 ถึง 0.05401646 มี pvalue 0.1358 จึงไม่มีนัยสำคัญที่จะ reject null hypothesis ว่า mean ทั้งสองนั้นเท่ากันหรือ มาจาก population เดียวกัน

โดย default R จะคำนวณแบบ two-tailed ถ้าต้องการคำนวณแบบ one-tailed ต้อง ใส่ option alternative="less" หรือ alternative="greater" เนื่องจาก mean เป็น 0.04981508 ซึ่งน้อยกว่า 0.053 จึงเลือกคำนวณ one-tailed แบบน้อยกว่าหรือ less ได้ pvalue ดีขึ้นแต่ก็ยังไม่มีนัยสำคัญทางสถิติ

อย่างไรก็ดี t-test เหมาะกับข้อมูลที่มีการกระจายตัวแบบ normal distribution ใน กรณีที่ข้อมูลเป็น skewed distribution ควรใช้ Wilcoxon Test ซึ่งจะเห็นว่าค่าที่ได้ดีขึ้น แต่ก็ยัง ไม่ถึง 0.05

```
> wilcox.test(durationsOnt$DurationPrefixPlosive, mu = 0.044)
Wilcoxon signed rank test with continuity correction
data: durationsOnt$DurationPrefixPlosive
V = 1871, p-value = 0.01151
alternative hypothesis: true location is not equal to 0.044
```

กรณีที่มีข้อมูลสองชุดเปรียบเทียบหรือสองเวกเตอร์ สามารถใช้ t.test นี้ได้ แต่ในกรณี เป็น paired sample ให้คำนวณแบบ paired sample เพราะจะได้ผลที่ดีกว่าการเปรียบเทียบค่า mean ปกติ เช่น การวัด vowel length ของผู้พูด 10 คน ในบริบทที่มีเสียงพยัญชนะที่ควบคุม ต่างกัน ในบริบท x วัด vowel length ผู้พูดคนแรกได้ 22 บริบท y วัดของผู้พูดคนแรกได้ 26, ผู้พูดคนที่สองวัดในบริบท x ได้ 18 ในบริบท y วัดได้ 22 ... เราสามารถป้อนข้อมูลโดยตรงเป็น 2 เวกเตอร์ x, y แล้วหา t.test โดยกำหนด option paired=TRUE ได้ดังนี้

ค่า p ที่ได้น้อยกว่า 0.05 จึงสรุปว่ามีความแตกต่างระหว่าง mean หรือสรุปว่าความต่างของบริบท พยัญชนะมีผลต่อความยาวสระ

ถ้าข้อมูลไม่ใช่ paired sample เช่น เก็บจำนวนข้อผิดหรือคะแนนนักเรียนสองห้อง ห้อง หนึ่งเป็นห้องทดลอง อีกห้องเป็นห้องควบคุม ก็ใช้ t.test เทียบ 2 vector เหมือนกัน แต่ไม่ต้องใส่ paired=TRUE

การนำข้อมูลเข้าคำนวณ t-test เราสามารถพิมพ์เข้าโดยตรงทีละกลุ่ม 2 ครั้งแบบข้างบน หรือสร้างตารางข้อมูลใน Excel แล้ว save เป็นไฟล์ .csv

	А	В	
1	group1	group2	
2	26	22	
3	22	18	
4	27	26	
5	15	17	
6	24	19	
7	27	23	
8	17	15	
9	20	16	
10	17	19	
11	30	25	
12			

จากนั้นอ่านไฟล์ .csv โดยใช้คำสั่ง read.csv มาเก็บในตัวแปรที่กำหนด แล้วเรียก t.test กับ ข้อมูลนี้โดยอ้างถึงแต่ละกลุ่มเปรียบเทียบแบบ paired sample

```
> dat = read.csv(file="c:/data.csv")
> dat
   group1 group2
       26
              22
1
2
       22
              18
3
       27
              2.6
4
      15
              17
5
      24
              19
6
      27
              23
7
      17
              1.5
8
      20
              16
              19
9
      17
10
       30
              2.5
                                                       > t.test(dat$group1,dat$group2,paired=TRUE)
        Paired t-test
data: dat$group1 and dat$group2
t = 2.9531, df = 9, p-value = 0.01614
alternative hypothesis: true difference in means is not equal to O
95 percent confidence interval:
 0.5849444 4.4150556
sample estimates:
mean of the differences
                    2.5
```

ได้ค่า p น้อยกว่า 0.05 จึงสามารถ reject null hypothesis และสรุปว่ามีความแตกต่างกันระหว่าง สองกลุ่มได้ ในกรณีของ paired sample แต่ไม่สามารถ assume normal distribution ได้ ก็ สามารถใช้ Wilcoxon test ได้ โดยใช้

```
>wilcox.test(x, y, paired=TRUE)
```

การคำนวณค่า chi-square

โปรแกรม R มีฟังก์ชั่นพื้นฐานสำหรับคำนวณค่า chi-square ซึ่งเป็นสถิติแบบที่เป็น nonparametric test ใช้กับข้อมูลที่เป็น category และเป็นจำนวนนับ(ความถี่) ให้ลองทดลองสร้าง ตารางข้อมูล 2 ตัวแปรจากข้อมูล auxiliaries ที่อยู่ใน package languageR ด้วยคำสั่ง xtabs

```
> xt = xtabs(~ Aux + Regularity, data = auxiliaries)
```

> xt = :	xtabs(~ Aux -	F Regularity,	data =	auxiliaries)
> xt				
	Regularity			
Aux	irregular	regular		
hebbei	n 94	118		
zijn	12	3		
zijnhe	≘b 36	22		

xt เก็บตารางของตัวแปร Aux กับ Regularity ซึ่งมีจำนวน category เป็น 3 และ 2 ตามลำดับ ได้เป็นตารางที่เห็นข้างบน (หากต้องการดูข้อมูล auxiliaries จะเห็นว่าเป็นข้อมูลแจกแจงสี่คอลัมน์ Verb, Aux, VerbalSynsets, และ Regularity คำสั่งนี้จะเอาค่าจากสอง column คือ Aux กับ Regularity มานับแจกแจงความถี่เป็นตาราง)

```
> prop.table(xt)
         Regularity
Aux
          irregular
                        regular
 hebben 0.32982456 0.41403509
  zijn
         0.04210526 0.01052632
  zijnheb 0.12631579 0.07719298
> prop.table(xt,1)
        Regularity
         irregular
                    regular
Aux |
 hebben 0.4433962 0.5566038
  zijn
         0.8000000 0.2000000
  zijnheb 0.6206897 0.3793103
> prop.table(xt,2)
        Regularity
                        regular
           irregular
Aux.
 hebben 0.66197183 0.82517483
  ziin
       0.08450704 0.02097902
  zijnheb 0.25352113 0.15384615
```

prop.table เป็นการสั่งให้คำนวณค่า probability ของตาราง ถ้าใส่เฉพาะชื่อตารางแบบแรก จะ คำนวณภาพรวมทั้งหมดเป็น 1 แต่ถ้ามี argument ที่สองเป็น 1 (prop.table(xt,1)) จะคำนวณ แต่ละแถว(row)ให้มีค่ารวมเป็น 1 ถ้า argument ที่สองเป็น 2 จะคำนวณแต่ละคอลัมน์ให้มีค่ารวม เป็น 1

คำสั่ง mosiacplot(xt) จะวาดภาพแสดงให้เห็นสัดส่วนต่างๆ



AW

Chisquare test ใช้คำสั่ง chisq.test กับตัวแปรที่เก็บตารางความถี่ที่ต้องการวัด (หลักของ Chisquare test คือการเปรียบเทียบว่าค่าที่พบ (observed) นี้แตกต่างจากค่าที่ควรจะเป็น (expected) อย่างผิดปกติไหม null hypothesis ค่าที่พบนี้ไม่ได้แตกต่างไปจากค่าที่ควรจะเป็นมาก นัก นั่นคือตัวแปรต้นไม่ได้มีผลอะไรต่อตัวแปรตาม สำหรับผู้สนใจรายละเอียด สามารถหาอ่านสูตร การคำนวณได้จากหนังสือสถิติทั่วไป)

```
> chisq.test(xt)
            Pearson's Chi-squared test
data: xt
X-squared = 11.4929, df = 2, p-value = 0.003194
```

ผลที่ได้มีค่า p < 0.05 จึงมีนัยสำคัญทางสถิติที่จะ reject null hypothesis ที่ว่าสองตัวแปรนี้ไม่มี ความสัมพันธ์กัน ดังนั้น จึงสรุปได้ว่าสองตัวแปรที่พิจารณานี้มีความสัมพันธ์กัน

ในกรณีที่ตารางมีค่าความถี่น้อย ค่า expected value น้อยกว่า 5 จึงไม่ควรใช้ chisquare test ให้ไปใช้ fisher test แทนโดยใช้คำสั่ง fisher.test(xt)

ในการเตรียมข้อมูลเพื่อใช้กับโปรแกรม R เราสามารถเตรียมข้อมูลเป็นตารางโดยแจกแจง ข้อมูลแต่ละรายการที่เก็บมา ไม่ต้องนับความถี่ให้

	А	В	С	D
1	Verb	Aux	VerbalSynsets	Regularity
2	blijken	zijn	1	irregular
3	gloeien	hebben	3	regular
4	glimmen	zijnheb	2	irregular
5	rijzen	zijn	4	irregular
6	werpen	hebben	3	irregular
7	delven	hebben	2	irregular
8	snikken	hebben	1	regular
9	laten	hebben	12	irregular
10	snijden	zijnheb	10	irregular
11	schrikken	zijnheb	3	irregular
12	zetten	hebben	4	regular
13	blazen	hebben	4	irregular
14	spelen	hebben	11	regular
15	spreken	hebben	6	irregular
16	rijgen	hebben	3	irregular
17	krabben	hebben	1	regular
18	binden	hebben	10	irregular
19	worden	zijn	2	irregular

อ่านข้อมูลที่ save เป็น csv ด้วยคำสั่ง >VAR = read.csv("FILENAME") แล้วสร้างตารางนับ ความถี่ด้วย xtab แบบตัวอย่างข้างบนก่อนจะเรียกใช้ chisq.test แต่ถ้าเราเตรียมข้อมูลมาเป็น ตารางความถี่ ก็สามารถป้อนตารางความถี่ลงใน Excel แล้วให้ save เป็น .csv format

	А	В	С
1		irregular	regular
2	hebben	94	118
3	zijn	12	3
4	zijnheb	36	22

แล้วอ่านผ่าน read.csv เช่น >xxt = read.csv("c:/test.csv", header = TRUE, row.names =

1) จะได้ตาราง xxt ที่นำไปใช้หา chisq.test ได้

```
> xxt
        irregular regular
hebben
                94
                       118
ziin
                12
                         3
zijnheb
                36
                        22
> chisq.test(xxt)
        Pearson's Chi-squared test
data: xxt
X-squared = 11.4929, df = 2, p-value = 0.003194
หรือพิมพ์เข้าโดยตรงโดยสร้าง matrix ขึ้นมา
> dat = matrix(c(94,12,36,118,3,22), nrow=3)
> dat
      [,1] [,2]
 [1,]
        94
           118
              3
 [2,]
        12
 [3,1
        36
             22
> chisq.test(dat)
         Pearson's Chi-squared test
 data: dat
X-squared = 11.4929, df = 2, p-value = 0.003194
```

ถ้าถนัดพิมพ์ไล่ตามแถวมากกว่า และอยากใส่ label ของ row และ column ก็ทำได้ ให้ใส่ byrow=TRUE, และใส่ dimnames

```
> dat = matrix(c(94,118,12,3,36,22), nrow = 3, ncol=2, byrow=TRUE,
+ dimnames = list(c("hebben", "zijn","zijnheb"), c("irregular", "regular")))
> dat
        irregular regular
               94
                      118
hebben
zijn
               12
                        3
                       22
zijnheb
               36
> chisq.test(dat)
        Pearson's Chi-squared test
data: dat
X-squared = 11.4929, df = 2, p-value = 0.003194
```

การคำนวณค่า anova

ในกรณีเปรียบเทียบ 2 กลุ่ม เราสามารถใช้ t-test ได้ แต่ถ้ามีมากกว่า 2 กลุ่มที่ต้องการ เปรียบเทียบ เราไม่สามารถใช้วิธีเปรียบเทียบทีละคูโดยใช้ t-test ได้เพราะจะมีค่าความผิดพลาดสูง เราต้องใช้ one-way anova แทน แต่ในกรณีที่มี dependent variable หลายตัวต้องใช้ manova แทน (Multivariate analysis of variance) เช่น ดูผลสัมฤทธิ์ของนักเรียนกลุ่มต่างๆ หลังจากได้รับ วิธีการสอนต่างกัน แบบนี้ใช้ anova ได้ แต่ถ้าวัดทั้งผลสัมฤทธิ์และทัศนคติของนักเรียนกลุ่มต่างๆ คือมี dependent variable มากกว่าหนึ่งตัว แบบนี้ใช้ anova ไม่ได้ต้องใช้ manova แทน ส่วนใน กรณีที่จำนวน independent variable มีมากขึ้น ก็จะเป็น two-way, three-way เช่น ใช้วิธีการ สอนต่างกันและดูปัจจัยทางเพศด้วย ก็จะเป็น two-way เป็นต้น

One-way anova มี null hypothesis ว่า mean ของแต่ละกลุ่มเหมือนกัน หาก reject หมายความว่ามีอย่างน้อยหนึ่งกลุ่มที่ไม่เหมือนกลุ่มอื่นๆ

 $H_0: \mu_1 = \mu_2 = \cdots = \mu_p, \quad H_A:$ at least one is not equal.

ตัวอย่างสมมติมีการให้คะแนนจากผู้ตรวจ 3 คน x,y,z สำหรับใบสมัครจำนวนหนึ่ง อยาก รู้ว่าค่า mean ของผู้ตรวจทั้งสามนั้นมาจากประชากรเดียวกันหรือไม่ คือต้องการดูว่าทั้งสามคนมี เกณฑ์การตรวจเหมือนกัน ให้นำคะแนนจากผู้ตรวจแต่ละคนนำเข้าในรูป vector เก็บไว้ที่ตัวแปร x, y, และ z จากนั้นให้คำสั่ง data.frame() เพื่อนำข้อมูล(ที่ต้องมีจำนวนเท่ากัน)มาสร้างเป็น data frame ตามที่เห็นเก็บไว้ในตัวแปร scores ซึ่งจะเห็นเทียบแต่ละรายการได้ว่าผู้ตรวจทั้งสามคนให้ คะแนนผู้สมัครแต่ละรายอย่างไร

```
> x=c(4,3,4,5,2,3,4,5)
> y=c(4,4,5,5,4,5,4,4)
> z=c(3,4,2,4,5,5,4,4)
> scores = data.frame(x,y,z)
> scores
    x y z
1 4 4 3
2 3 4 4
3 4 5 2
4 5 5 4
5 2 4 5
6 3 5 5
7 4 4 4
8 5 4 4
```

แต่ก่อนจะใช้คำสั่งหา anova ได้ เราจะต้องทำการ stack ข้อมูลให้เป็นแถวเดียวที่บอกว่าคะแนนนั้น มาจากคนไหนก่อน ดังรูปข้างล่าง วิธีการคือใช้คำสั่ง stack() เมื่อ stack แล้วจะได้ข้อมูล 24 รายการที่มี values และ ind

```
> scores=stack(scores)
> scores
  values ind
       4
1
           x
2
       3
           x
3
       4
           x
4
       5 x
5
       2
           х
6
       3 x
7
       4 x
       5
8
         x
       4 y
9
10
       4
         У
       5
11
         У
12
       5
           v
13
       4
         v
14
       5
          У
       4
15
         y.
16
       4
         У
       3
17
           z
       4 z
18
       2 z
19
       4 z
20
21
       5 z
22
       5
         z
       4 z
23
24
       4
           z
```

ชื่อตัวแปร values กับ ind จะถูกสร้างให้โดยอัตโนมัติจากการ stack ข้อมูล เราอาจเตรียมข้อมูล เองในรูปแบบนี้และใช้ชื่อตัวแปรเอง จากนั้นหาค่า oneway-anova ซึ่งทำได้สองแบบ ใช้ คำสั่ง oneway.test() หรือ anova() แต่หากจะใช้แบบหลังจะต้องสั่งให้หา lm() (linear modeling) ก่อน ตามตัวอย่างข้างล่าง

ตัวอย่างข้อมูล auxiliaries ใน package languageR แสดงให้เห็นการใช้ anova (Baayan 2008: 104-108)

```
> auxiliaries.lm = lm(VerbalSynsets ~ Aux, data = auxiliaries)
```

สร้าง linear model ระหว่าง VerbalSynsets กับ Aux จากข้อมูล auxiliaries แล้วหาค่า oneway anova ซึ่งพบว่ามีนัยสำคัญ หมายความว่ามีความต่างอย่างน้อย ในหนึ่งตัวแปรที่อยู่ใน Aux ที่ มีผลต่อ variation ที่พบใน VerbalSynsets

เมื่อแจกแจง Aux ในข้อมูลพบว่า มี 3 ตัว แต่เราไม่รู้ว่า mean ของตัวไหนที่แตกต่างจากกลุ่ม

```
> levels(auxiliaries$Aux)
[1] "hebben" "zijn" "zijnheb"
> xtabs(~auxiliaries$Aux)
auxiliaries$Aux
hebben zijn zijnheb
_ 212 15 58
```

ถ้าจะดูว่าเป็น mean ตัวไหนที่มีผลแตกต่างจากตัวอื่น เราต้องใช้ function aov() สำหรับคำนวณ anova แทน แล้วเรียกใช้ TukeyHSD แต่สถิติตัวนี้จะใช้ได้ในกรณีที่ข้อมูลแต่ละกลุ่มมีจำนวนเท่า กัน ในตัวอย่างบนเนื่องจากจำนวนข้อมูล Aux แต่ละตัวไม่เท่ากันจึงใช้การคำนวณวิธีนี้ไม่ได้ ในที่นี้ จะใช้ข้อมูล warpbreaks เป็นตัวอย่างแทนเพราะมีจำนวนข้อมูล tension แต่ละกลุ่มเท่ากัน (H M L) และใช้ function aov() ซึ่งเมื่อสั่ง summary ค่าออกมาจะเห็นผลเหมือน anova)

จากนั้น ใช้ TukeyHSD (Tukey's honestly significant difference) เพื่อดู

คอลัมน์แรกแสดงข้อมูลความต่างของค่าเฉลี่ยแต่ละคู่ จากผลที่แสดงจะเห็นว่า H-L เป็นคู่ที่สร้าง ความต่างในกลุ่มมากสุด รองมาเป็น M-L และ H-M ส่วนคอลัมน์ที่สองและสามเป็นค่า lower bound และ upper bound ที่ 95% confidence interval หากต้องการดูความต่างเป็นกราฟ ก็ สามารถใช้คำสั่ง >plot(TukeyHSD()) ได้ ซึ่งจะเห็นคู่ H-L อยู่ไกลจากค่าศูนย์มากที่สุด จาก ข้อมูลที่ได้ จึงเห็นว่าความต่างของค่าเฉลี่ยระหว่าง H-M ไม่มีนัยสำคัญทางสถิติ แต่ความต่างของค่า เฉลี่ยระหว่าง H-L และ M-L มีนัยสำคัญทางสถิติ ทำให้สรุปได้ว่า L เป็นชุดข้อมูลที่ต่างไปจากกลุ่ม



95% family-wise confidence level

Figure 4.15. Family-wise 95% confidence intervals for Tukey's honestly significant difference for the warpbreaks data. The significant differences are those for which the confidence intervals do not intersect the dashed zero line.

Baayen (2008: 108)

การคำนวณค่า correlation

correlation เป็นการดูความสัมพันธ์เชิงเส้นของสองตัวแปรที่เป็นข้อมูลตัวเลข (y = a + x*b) R สามารถ plot กราฟแสดงให้เห็นได้ด้วย

```
> plot(ratings$meanWeightRating, ratings$meanSizeRating,
+ xlab = "mean weight rating", ylab = "mean size rating")
```

สั่งให้วาดกราฟแสดงความสัมพันธ์ข้อมูล meanWeightRating กับ meanSizeRating จาก data frame ratings





linear regression model จะเป็นการหาเส้นที่ดีที่สุดที่จะ match ข้อมูลทั้งหมด ซึ่งใช้ หลักการของ least square (ให้ผลรวมค่าความต่างระหว่างค่า observe กับค่าที่ทำนายจากเส้น model ยกกำลังสองแล้วมีค่าน้อยสุด) ถ้าต้องการหาค่า coefficient of correlation ให้ใช้คำสั่ง cor.test

ค่า coefficient of correlation คือค่าที่รายงานในบรรทัดสุดท้าย จะแสดงถึงความสัมพันธ์เชิงเส้น
 ของสองตัวแปรว่าแปรตามกันอย่างไร ส่วนค่า p-value จะบอกถึงความสัมพันธ์นั้นว่ามีนัยสำคัญ
 ทางสถิติหรือไม่ ค่า correlation ยิ่งใกล้หนึ่งแสดงว่ามีความสัมพันธ์เชิงเส้นระหว่างสองตัวแปรนั้น
 ถ้าเข้าใกล้ศูนย์จะไม่มีความสัมพันธ์ต่อกัน ถ้าเป็นค่าติดลบแสดงว่ามีความสัมพันธ์แบบผกผันกัน รูป
 ด้านล่างตัวอย่างกราฟแสดง correlation ต่างๆ



Figure 4.10. Scatterplots for four paired standard normal random variables with different

Baayen (2008: 88)

การคำนวณ correlation หากต้องการใช้ spearman correlation ไม่ใช่ pearson correlation ก็สามารถทำได้โดยการเติม option method="spearman" เข้าไป default ของ cor.test คือ method="pearson"

สำหรับการใช้ spearman correlation นั้นใช้กับข้อมูลแบบ ordinal ได้ด้วยไม่จำเป็น ต้องเป็นข้อมูลแบบ continuous เหมือน pearson และ spearman correlation เป็นการมอง ความสัมพันธ์แบบ rank-order ไม่ได้เป็นแบบ linear model เหมือน pearson ค่าที่ได้จึงต่างกันได้ ดังตัวอย่างนี้ที่มีค่า pearson = 0.851, spearman = 1



(http://support.minitab.com/en-us/minitab-express/1/help-and-how-to/modelingstatistics/regression/supporting-topics/basics/a-comparison-of-the-pearson-andspearman-correlation-methods/)

Linear model assume ค่าต่างๆ มีการกระจายตัวแบบปกติ ค่าที่สุดโต่งจำนวนเล็กน้อย ก็สามารถเบี่ยงเบนผลไปได้. ปัญหาของการใช้ linear modeling ถ้าไม่ดูข้อมูลให้ดี จะได้ค่าความ สัมพันธ์ที่ไม่ได้เป็นแบบนั้น ดังตัวอย่างนี้

เป็นการหา linear modeling ของสองตัวแปร meanSizeRating กับ meanFamiliarity ซึ่งดู เหมือนมีความสัมพันธ์เชิงลบอย่างมีนัยสำคัญ (-0.2 p < .05) แต่หาก plot กราฟมาดูจะเห็นจริงๆ ข้อมูลควรมองแยกเป็นสองกลุ่มมากกว่า (รูปบนขวาแสดง linear modeling) ส่วนรูปด้านล่าง แสดงการมองแยกข้อมูลเดียวกันเป็นสองกลุ่ม กรณีแบบนี้ ความสัมพันธ์แบบ linear อาจไม่เหมาะ (y = a + b*x) อาจใช้วิธีการมองแบบ nonlinear มาช่วย (y = a + b*x + c*x^2) จะได้เส้นแบบพาราโบลา





multiple regression model

โปรแกรม R มีฟังก์ชั่นพื้นฐานสำหรับคำนวณสถิติ ดังนี้ Anova และ linear regression model ใช้กับกรณี dependent variable เป็นข้อมูลตัวเลขต่อเนื่อง โดย anova ใช้กรณีตัวแปร independent เป็นแบบ categorical ส่วน linear regression ใช้เมื่อตัวแปร independent เป็น ข้อมูลตัวเลขต่อเนื่อง

Log-linear และ logistic regression model ใช้กับกรณี dependent variable เป็น ข้อมูลแบบ categorical ทั้งสองแบบนี้อยู่ภายใต้ statistical model ที่เรียกว่า generalized linear model (GLM) Logistic regression model ใช้ในกรณีข้อมูลของ dependent variable เป็น 2 ตัวเลือก ที่มองเป็น 0 กับ 1 ได้ เช่น Yes/No Male/Female แต่ในกรณีที่เป็น category หลายตัวเลือก จะต้องใช้ log-linear model

Linear regression model เป็นการมองความสัมพันธ์ของ independent variable หนึ่ง ตัว (y = a+b*x) แต่ถ้ามีตัวแปร independent variable หลายตัว model จะเป็นแบบ multiple regression model (y = a + b*x1 +c*x2 +d*x3 ...) และสามารถใช้คำสั่ง lm คำนวณ model ที่ดีที่สุดออกมาได้ แต่กระบวนการวิเคราะห์จะไม่เป็นแบบง่ายๆ เราไม่สามารถดูความ สัมพันธ์แบบง่ายๆ โดยเอาตัวแปรทีละคู่มาดู correlation เพราะตัวแปรแต่ละแปรอาจมีผลต่อกัน ไม่ได้เป็นอิสระโดยตัวมันเอง เราจึงต้องดูผลที่ได้จากตัวแปรต่างๆ ไปพร้อมกันในคราวเดียว หากตัว ไหนไม่ significant คือไม่มีผลต่อ dependent variable ก็จะค่อยๆ ตัดออกไป จนเหลือเฉพาะ ตัวแปรที่เกี่ยวข้องจริง

ใน simple linear regression คือการมองความสัมพันธ์ที่ตัวแปร independent มีตัว เดียว มองในรูปสมการได้เป็น y = a + b*x1 เมื่อหา >lm(y ~ x1, data=dfrm) ก็จะได้ค่า coefficients ออกมา ในกรณี multiple linear regression คือมีตัวแปร independent มากกว่า หนึ่งตัว เช่น มองในรูปสมการได้เป็น y = a + b*x1 + c*x2 +d*x3 เมื่อหา >lm(y ~ x1 + x2 + x3, data=dfrm) ก็จะได้ค่า coefficients ออกมา การใช้ lm() เป็นการสร้าง model object ที่ เหมาะกับข้อมูล แต่จะแสดงผลเพียงค่า coefficient ระหว่างตัวแปรเท่านั้น หากต้องการดูค่าทาง สถิติอื่นๆ เราต้องใช้คำสั่ง summary อีกที ในตัวอย่างข้างล่าง >m = lm(y ~ u + v + w) จะ สร้าง object model "m" ขึ้นมา เมื่อใช้คำสั่ง >summary(m) ค่ารายละเอียดทางสถิติต่างๆ จะ แจกแจงออกมา (ดู recipe 11.1-11.4 ใน R Cookbook)

```
> lm(y ~ u + v + w)
Call:
lm(formula = y ~ u + v + w)
Coefficients:
(Intercept) u v w
1.4222 1.0359 0.9217 0.7261
```

> summary(m) Call: $lm(formula = y \sim u + v + w)$ Residuals: Min 10 Median 30 Max -3.3965 -0.9472 -0.4708 1.3730 3.1283 Coefficients: Estimate Std. Error t value Pr(>|t|) (Intercept) 1.4222 1.4036 1.013 0.32029 0.2811 1.0359 3.685 0.00106 ** u v 0.9217 0.3787 2.434 0.02211 * 0.7261 0.3652 1.988 0.05744 . w Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 · · · 0.1 · · 1 Residual standard error: 1.625 on 26 degrees of freedom Multiple R-squared: 0.4981, Adjusted R-squared: 0.4402 F-statistic: 8.603 on 3 and 26 DF, p-value: 0.0003915

ค่า residuals ซึ่งเป็นค่าระยะห่างจาก mean จะแสดงให้เห็นว่า distribution มีลักษณะ normal ไหม ในสถานการณ์ที่ normal distribution ค่า mean ควรใกล้ศูนย์ และค่า quartile ที่ 1 และ 3 ควรมีขนาดพอๆกัน หากมีลักษณะที่ skew ข้างไหน ก็จะอ่านจากค่า residuals ได้ ใน ตัวอย่างนี้ จะเห็น 3Q (1.3730) มากกว่า 1Q (0.9472) แสดงว่า skew ไปทางขวา

ค่า coefficient ส่วนของคอลัมน์ Estimate บอกค่าประมาณของ coefficient (หรือค่า สัมประสิทธิ์ในสมการเชิงเส้น b, c, d) หากมีค่าใกล้ศูนย์ก็แสดงว่าตัวแปรนั้นไม่ได้มีบทบาทสำคัญ คอลัมน์ p-value บอกค่านัยสำคัญทางสถิติ ค่ายิ่งน้อยยิ่งแสดงว่าตัวแปรนั้นมีผลต่อตัวแปรตาม โดยจะเห็นได้จากจำนวนเครื่องหมาย * ที่แสดง

ค่า R2 หรือ coefficient of determination บอกถึงคุณภาพของ model หากมีค่ามาก จะดี ค่า F บอกว่า model มีนัยสำคัญทางสถิติไหม ซึ่งก็ดูได้จากค่า p-value ที่แสดง เวลาดู ผลควรดูที่ค่า F ก่อน หากไม่มีนัยสำคัญก็ไม่จำเป็นต้องพิจารณาต่อ

ปัจจุบันงานวิจัยทางสังคมศาสตร์มีงานที่ใช้สถิติในกลุ่ม regression นี้มาก เพราะข้อมูลที่ เก็บมีหลายตัวแปร ซึ่งเราไม่รู้ว่าตัวแปรไหนมีผลต่อตัวแปร dependent และไม่รู้ว่าตัวแปร independent variable แต่ละตัวมีผลต่อกันหรือไม่ด้วย จึงต้องใช้สถิติแนวนี้หา model ที่ fit กับ ข้อมูลที่รวบรวมมาได้ดีที่สุด ใน R เราสามารถสร้าง lm เป็น full model ขึ้นมาก่อน จากนั้นจึง ลดตัวแปรที่ไม่เกี่ยวข้องลง เช่น

>full.model = lm(y ~ x1 + x2 + x3 + x4)

```
>reduced.model = step(full.model, direction= "back-ward")
```

ในตัวอย่างข้างล่าง (recipe 11.7 R Cookbook) จะเห็นในเบื้องต้นว่า x1, x3 มีค่า coefficient ที่มีนัยสำคัญ เมื่อลอง step backward เราจะเห็นการลดตัวแปรลงทีละขั้น ขั้นแรกลด x2 ออก ต่อมาลด x4 ออกจนได้ model สุดท้ายที่มีแต่ y ~ x1 + x3

```
> full.model <- lm(y ~ x1 + x2 + x3 + x4)
> summary(full.model)
Call:
lm(formula = y ~ x1 + x2 + x3 + x4)
Residuals:
    Min     10 Median     30 Max
-34.405 -5.153     2.025     6.525     19.186
Coefficients:
```

Estimate Std. Error t value Pr(>|t|) (Intercept) 10.274 4.573 2.247 0.0337 * x1 -102.29847.558 -2.151 0.0413 * x2 4.362 40.237 0.108 0.9145 75.115 x3 34.236 2.194 0.0377 * x4 26.286 42.239 0.622 0.5394 0 '***' 0.001 '**' 0.01 '*' 0.05 Signif. codes: ··· 0.1 · · 1 Residual standard error: 12.19 on 25 degrees of freedom Multiple R-squared: 0.8848, Adjusted R-squared: 0.8664 48 on 4 and 25 DF, p-value: F-statistic: 2.235e-11 57.58 3774.3 153.04 - x4 1 3716.7 154.58 <none> - x1 687.89 4404.6 157.68 1 - x3 1 715.67 4432.4 157.87 Step: AIC=152.6 $y \sim x1 + x3 + x4$ Df Sum of Sq RSS AIC 77.82 3796.3 151.22 - x4 1 3718.5 152.60 <none> - x3 737.39 4455.9 156.02 1 - x1 1 787.96 4506.4 156.36 Step: AIC=151.22 $y \sim x1 + x3$ Df Sum of Sq RSS AIC 3796.3 151.22 <none> 884.44 4680.7 155.50 - x1 1 - x3 1 964.24 4760.5 156.01

นอกจากวิธีการ backward เราอาจทำแบบ forward คือค่อยๆเพิ่มตัวแปรที่เกี่ยวข้อง หรือ อาจใช้วิธีการอื่นๆ อีกที่ Baayen (2008) เขียนไว้ในหนังสือ แต่การที่จะใช้สถิติแบบนี้คำนวณต้อง ศึกษาให้เข้าใจหลักการและวิธีการตีความค่าตัวเลขต่างๆ ที่เกี่ยวข้อง ตลอดจนวิธีการปรับเปลี่ยน model เพื่อให้ได้ผลที่ดีที่สุด เช่น ต้องสร้าง model แบบที่ independent variable ไม่มีผลต่อกัน และสร้าง model แบบที่คิดว่า independent variable มีผลต่อกัน จากนั้นเปรียบเทียบ anova ของทั้งสองแบบว่ามีความแตกต่างอย่างมีนัยสำคัญหรือไม่ และในบรรดา independent variable ก็ต้องปรับ model เพื่อกันตัวแปรที่ไม่มีผลต่อ dependent variable ออกไป การใช้สถิติแบบนี้ จึงไม่สามารถทำการคำนวณได้ง่ายๆ แบบ chi-square หรือ anova ผู้ที่จำเป็นต้องใช้สถิติแบบนี้ จึงต้องศึกษาเพิ่มเติมอย่างมากเพื่อทำความเข้าใจวิธีการใช้และวิเคราะห์ model ต่างๆ เมื่อเข้าใจ ดีแล้ว ก็สามารถใช้คำสั่งในโปรแกรม R เพื่อช่วยคำนวณได้ ผู้สนใจสามารถหาอ่านเพิ่มเติมได้ที่

Paolillo, J. C. 2002. Analyzing linguistic variation : statistical models and methods. Stanford, Calif.: CSLI Publications, Center for the Study of Language and Information.

_{บทที่ 3} การทำเหมืองข้อมูล

การนำข้อมูลเข้า

การนำข้อมูล text เข้ามาเพื่อวิเคราะห์สามารถใช้คำสั่งพื้นฐาน scan หรือ readLines ตาม ที่ได้อธิบายไว้ในบทที่แล้วได้

```
>text <- scan(file="c:/temp/DHl.txt", what="character", sep="\n")
```

```
>text <- readLines("c:/temp/DH.txt",encoding="UTF-8")
```

และหากต้องการอ่านข้อมูลมากกว่าหนึ่งไฟล์ ก็สามารถอ่านไฟล์ทั้งหมดใน folder มาเก็บเป็น list แล้ว apply function scan() กับไฟล์ใน list นั้น

```
>list.filenames<-list.files(pattern=".txt$")
```

```
>list.data<-list()
```

```
>list.data <- lapply(list.filenames, function(x) scan(file=x, what="character", sep="\n") )
```

หรือใช้คำสั่งใน package tm (text mining) ที่มีคำสั่งให้อ่านข้อมูลแหล่งต่างๆ มาสร้างเป็นคลัง ข้อมูลได้ นอกจาก package tm อีก package ที่เป็นประโยชน์คือ wordcloud ที่สามารถช่วย สร้างกราฟฟิกของคำตามขนาดความถี่ของคำได้ ก่อนอื่น จะต้องติดตั้ง package ที่จำเป็นต้องใช้ ก่อน ได้แก่

```
>install.packages("tm")
```

```
>install.packages("wordcloud")
```

ในระหว่างการติดตั้ง tm กับ wordcloud package อื่นๆที่จำเป็นต้องใช้ หากยังไม่ได้ติด ตั้งก็จะถูกติดตั้งไปโดยอัตโนมัติด้วย เมื่อติดตั้งเรียบร้อยก็เรียกใช้ package ทั้งสองผ่านคำสั่ง library จากนั้นให้โหลดไฟล์ข้อมูล ในที่นี้ใช้ plain text เก็บไว้ใน directory เดียวกัน คือไฟล์นิยาย 8 เรื่อง ของ Jane Austen โดยใช้คำสั่ง Corpus(DirSource("..."))

```
> library(tm)
```

> library(wordcloud)

> source <- DirSource("/Users/macbook/Cloud/Dropbox/course/Corpuslg/Corpus/ Jane\ austen")

```
> j.corpus <- Corpus(source, readerControl = list(reader=readPlain))
```

DirSource ใช้บอกแหล่งและประเภทข้อมูลที่จะอ่านเข้ามาว่าเป็นไฟล์ทั้งหมดใน directory ที่ระบุ จาก DirSource. ใน tm ให้เรากำหนดแหล่งข้อมูลประเภทอื่น ๆ ได้อีก เช่น DataframeSource, URISource, VectorSource ส่วน readerControl ใช้กำหนดว่าอ่านข้อมูลประเภทไหน โดย deafult เป็น plaintext จึงสามารถละได้ แต่หากต้องการอ่านข้อมูลประเภทอื่น ต้องใช้ reader=readDOC หรือ readPDF หรือ readXML เป็นต้น

เมื่อโหลดข้อมูลเรียบร้อยแล้ว เรียกดู j.corpus จะเห็นว่ามี 8 documents ที่โหลดมาใน corpus นี้

> j.corpus

<<VCorpus>>

Metadata: corpus specific: 0, document level (indexed): 0

Content: documents: 8

ถ้าต้องการดูเฉพาะไฟล์ใน j.corpus ได้ซึ่งจะบอกความยาวของตัวบทนั้น ถ้าต้องการดูเนื้อความก็ให้ ใช้ j.corpus[[1]]\$content

> j.corpus[[1]]

<<PlainTextDocument>>

Metadata: 7

Content: chars: 885172

การสร้าง word cloud

เราสามาถใช้โมดูล wordcloud สร้างภาพรายการคำโดนจำกัดไว้ที่ 100 คำที่มีความถี่สูงสุด

> wordcloud(j.corpus,max.words=100)



ซึ่งจะเห็นว่ามีรายการคำที่เป็นคำไวยากรณ์ปรากฏมามากสุด หากต้องการลบคำเหล่านี้ (เรียกว่า stop-words) รวมทั้งลบเครื่องหมายวรรคตอน ก็สามารถทำได้โดยใช้คำสั่งนี้

- > j.corpus <- tm_map(j.corpus, removeWords, stopwords('english'))
- > j.corpus <- tm_map(j.corpus, removePunctuation)

เมื่อใช้คำสั่ง wordcloud แบบเดิมจะได้



การ remove คำที่ไม่ต้องการออกจาก corpus ยังสามารถใช้คำสั่งเดิมแต่เพิ่ม option อื่น ๆ ได้แก่ removeNumbers ลบตัวเลขออก removeWords, c("department", "email") ลบคำที่ ระบุในรายการออก stripWhitespace ลบช่องไฟที่เกิน หรือ stemDocument ถอดหน่วยคำเติม ออกให้เหลือรากคำ

หากต้องการให้แสดงเป็นภาพสีก็สามารถทำได้โดยการเพิ่ม option colors ในคำสั่ง wordcloud ดังนี้

> wordcloud(j.corpus, max.words=100, colors=brewer.pal(8, "Dark2"))



การเพิ่ม font เพื่อใช้ใน wordcloud

สำหรับกรณีข้อมูลภาษาไทย จะไม่สามารถแสดงผลได้ทันที ต้องติดตั้งโมดูล extrafont

```
> install.packages("extrafont")
```

แล้วเรียกใช้โมดูลนี้พร้อมติดตั้ง font ที่ต้องการใน R จากนั้นจึง load font ที่ติดตั้งเพิ่มนี้เข้ามาก่อน จะเพิ่ม option ในคำสั่ง wordcloud ให้ใช้ font ภาษาไทยได้ โดยใช้คำสั่งต่อไปนี้> library(extrafont)

```
> font_import(pattern="THSarabunNew")
```

> loadfonts()

<pre>> library(extrafont) Registering fonts with R > font_import(pattern="THSarabunNew") Importing fonts may take a few minutes, depending on the number of fonts and the speed of the system. Continue? [y/n] y Scanning ttf files in /Library/Fonts/, /System/Library/Fonts, ~/Library/Fonts/ Extracting .afm files from .ttf files /Users/macbook/Library/Fonts/THSarabunNew Bold.ttf => /Library/Frameworks/R.framework/Versions/3.3/Resources/Library/</pre>
<pre>vetrafontdb/metrics/THSarabunNew Bold /Users/macbook/Library/Fonts/THSarabunNew BoldItalic.ttf => /Library/Frameworks/R.framework/Versions/3.3/Resources/ library/extrafontdb/metrics/THSarabunNew Italic.ttf => /Library/Frameworks/R.framework/Versions/3.3/Resources/ library/extrafontdb/metrics/THSarabunNew Italic.ttf => /Library/Frameworks/R.framework/Versions/3.3/Resources/ library/extrafontdb/metrics/THSarabunNew Italic</pre>
Found FontName for 4 fonts. Scanning afm files in /Library/Frameworks/R.framework/Versions/3.3/Resources/library/extrafontdb/metrics Writing font table in /Library/Frameworks/R.framework/Versions/3.3/Resources/library/extrafontdb/fontmap/ fonttable.csv Writing Fontmap to /Library/Frameworks/R.framework/Versions/3.3/Resources/library/extrafontdb/fontmap/Fontmap > loadfonts() Reaisterina font with R usina pdfFonts(): TH Sarabun New

เมื่อโหลดฟอนต์ภาษาไทยให้ R รู้จักแล้วจึงสามารถใช้คำสั่ง wordcloud สร้างภาพคำไทยมาได้เช่น กันดังตัวอย่างนี้ โดยที่ thai.corpus เป็นข้อมูลที่สร้างจากการใช้คำสั่ง Corpus เหมือนตัวอย่างข้าง ต้นที่ใช้สร้าง corpus ของ Jane Austen option random.order=FALSE ทำให้คำที่ความถี่สูง กว่าจะปรากฏอยู่กลางภาพ rot.per กำหนดจำนวนเปอร์เซ็นต์ที่คำถูกจัดเรียงตามแนวดิ่ง

> wordcloud(t.corpus, max.words=100, family="TH Sarabun New", font=2, random.order=FALSE, rot.per=0.35, colors=brewer.pal(8, "Dark2"))



การค้นหาใน text

คำสั่งพื้นฐาน grep เป็นคำสั่งให้ค้นหาสายอักขระที่ระบุในข้อความได้ ในตัวอย่างล่าง province.names เป็นเว็กเตอร์ของชื่อจังหวัด คำสั่ง grep ปกติจะให้ index ของข้อมูลที่มีสาย อักขระที่กำหนด หากต้องการข้อความออกมาแทนที่จะเป็น index ให้เติม option value-TRUE

>grep("Nakohn", province.names)

[1] 19 20 24 34 35 39 60

>grep("Nakohn", province.names, value=TRUE)

- [1] "Nakhon Phanom" "Nakhon Ratchasima "
- [3] "Sakon Nakhon " "Nakhon Nayok "

[5] "Nakhon Sawan " "Phra Nakhon Si Ayutthaya "

[7] "Nakhon Si Thammarat "

pattern ที่ใช้ในการหานั้นสามารถเขียนโดยใช้ regular expression ได้ เช่น .+[tT]hani หมายถึง หาคำที่มี thani หรือ Thani โดยที่ข้างหน้าเป็นตัวอักษรอะไรกี่ตัวก็ได้

```
    > grep(".+[tT]hani",province.names, value=TRUE)
    [1] "Ubon Ratchathani" "Udon Thani" "Uthai Thani" "Surat Thani"
```

การแทนที่ข้อความ

สามารถใช้คำสั่ง sub หรือ gsub รูปแบบคือ gsub(pattern, replace, text) โดยการค้น สามารถใช้ regular expression ได้ ตัวอย่างข้างล่างเป็นการเขียน function สำหรับลบ white space ที่อยู่หน้าและท้ายข้อความ คือ หา [:space:] ที่อยู่ต้นข้อความ (^) มากกว่าหนึ่ง

```
trim <- function( x ) {
  gsub("(^[[:space:]]+|[[:space:]]+$)", "", x)
}</pre>
```

การหา concordance

>pos<-grep(" ambiguous ",text) >conx<-sub(" ambiguous "," **ambiguous** ",text[pos])

การสร้างรายการคำและความถึ่

กรณีที่ถ้าเราอ่านข้อมูลเข้ามาเป็นเว็กเตอร์ของคำแล้ว

```
> DH.text<-scan("DH-All.txt","character",sep="")
```

เราสามารถสร้างรายการคำพร้อมนับความถี่ได้ทันทีโดยใช้คำสั่งแจกแจงตารางหรือ table แ้วจัด เรียงตามความถี่จากมากไปน้อยด้วยคำสั่ง sort ดังนี้

```
> freq<-table(DH.text)
```

```
> freql <-sort(freq, decreasing=TRUE)
```

> freql[1:10]

DH.text

the of and to a in is that for as 13517 10452 8285 5959 5049 4651 2971 2812 2644 2162

จากข้อมูลใน corpus ที่มี เราสามารถดูการปรากฏของคำในแต่ละเอกสารได้โดยใช้คำสั่ง สร้างตารางคำกับเอกสาร TermDocumentMatrix ผลที่ได้จะเห็นว่าทีไฟล์เอกสาร 8 มีจำนวน ศัพท์ทั้งหมด 17523 คำ

> dtm <- DocumentTermMatrix(j.corpus)

> dtm

<<DocumentTermMatrix (documents: 8, terms: 17523)>>

Non-/sparse entries: 51203/88981

Sparsity : 63%

Maximal term length: 32

Weighting : term frequency (tf)

```
จากข้อมูลที่สร้างเป็นตาราง document-term เราสามารถ explore ข้อมูลในรูปแบบต่าง ๆ ได้
เช่น
```

> freq <- colSums(as.matrix(dtm))

แจกแจงรายการคำศัพท์ทั้งหมดพร้อมความถี่โดยการรวมความถี่ที่พบในแต่ละ document รวมเข้า ด้วยกัน

> findFreqTerms(dtm, lowfreq=1000)

[1] "but" "can" "every" "first" "good" "great" "know" "lady" "little" "may""might"

[12] "miss" "mrs" "much" "must" "never" "nothing" "now" "one" "said" "say" "see"

[23] "she" "soon" "the" "think" "though" "time" "well" "will" "without"

```
"you"
แจงเฉพาะคำที่มีความถื่อย่างน้อย 1000
```

> findAssocs(dtm, "friendship", corlimit=0.9) หาคำที่มักปรากฏร่วมกับคำว่า friendship ด้วยระดับ correlation 0.9 ขึ้นไป

```
> dtms <- removeSparseTerms(dtm, 0.01)
```

> dim(dtms)

[1] 8 1759

removeSparseTerms ใช้ลดขนาดตารางลงโดยนำคำที่ sparse หรือไม่ได้ปรากฏทั่วไปออก ส่วน จะเอาคำใดออกก็ขึ้นกับค่าที่ใส่ ถ้าใกล้หนึ่ง เช่น .99 ก็จะแทบไม่มีคำไหนออก ถ้าต่ำลงเช่น .01 ก็ จะเหลือคำที่ปรากฏอยู่ในทุก document เก็บไว้ซึ่งในที่นี้จะเหลือ 1759 คำจากเดิม 17,523 คำ

กรณีที่ข้อมูลไม่ใช่ text แต่อยู่ในรูปแบบอื่น เช่น xml ให้ติดตั้ง package XML โดยใช้คำ สั่ง install.packages("XML") เราสามารถใช้คำสั่ง xmlToDataFrame ได้โดยระบุ url หรือชื่อไฟล์ xml นั้น เช่น

```
> data <- xmlToDataFrame("000237.xml")</pre>
```

หากต้องการอ่านข้อมูลไฟล์ xml ทั้งหมดที่กำหนดไว้ใน working directory แล้วก็สามารถใช้ชุดคำ สั่งนี้ได้ list.files ใช้แจกแจงชื่อไฟล์ทั้งหมดออกมา xmlToDataFrame ใช้แปลงข้อมูลไฟล์ xml เป็น dataframe ซึ่งในไฟล์นี้ข้อมูลที่ต้องการอยู่ในแท็ก body ซึ่งถูกแปลงมาเป็นคอลัมน์ภายใต้ ตัวแปร body ด้วย จึงนำข้อมูลนี้ของแต่ละไฟล์มาต่อข้อความกันแล้วเก็บไว้ใน txt จากนั้นจึงสร้าง corpus จากข้อมูลที่ได้เก็บไว้ใน txt

```
> files <- list.files()
```

```
> txt <- NULL
```

```
> for (i in files) { doc <- xmlToDataFrame(i)
```

```
+txt <- c(txt, doc$body) }
```

```
> dhq.corpus <- Corpus(VectorSource(txt))
```

```
> dhq.corpus
```

```
<<VCorpus>>
Metadata: corpus specific: 0, document level (indexed): 0
Content: documents: 106
```

Web Crawler

เราสามารถใช้ package ใน R เพื่อช่วยในการสกัดข้อมูลที่ต้องการจากเว็บเพื่อมาใช้ในงาน ที่ต้องการได้ package หนึ่งที่เป็นประโยชน์เพื่อการนี้คือ rvest ตัวอย่างข้างล่างดัดแปลงจาก stat4701.github.io/edav/2015/04/02/rvest_tutorial/

```
> install.packages("rvest")
```

```
>library(rvest)
```

load ข้อมูลจากหน้าเว็บ review ภาพยนตร์ Arrival มาเก็บไว้

```
> arrival_movie <- html("http://www.imdb.com/title/tt2543164/")</pre>
```

ดึงข้อมูล rating ออกมาเก็บไว้ที่ตัวแปร rating ข้อมูลนี้อยู่ในแท็ก strong ซึ่งภายใต้มีแท็ก span อยู่ แล้ว สกัด text ในนั้นออกมา แปลงเป็นตัวเลข เครื่องหมาย %>% เป็นการ pipe หรือส่งผ่าน ข้อมูลต่อ ซึ่งจะเห็นว่าในคำสั่งให้เอาข้อมูลเว็บที่เก็บใน arrival_movie มาสกัด node ที่มีแท็ก strong แล้วก็ span เอาข้อมูลที่ได้ส่งต่อให้ html_text() ดึง text ออกมาส่งต่อให้มาแปลงป็น ตัวเลข

```
> rating <- arrival_movie %>%
```

- + html_nodes("strong span") %>%
- + html_text() %>%
- + as.numeric()
- > rating

หากดูภายในเว็บจะเห็นข้อมูลการให้คะแนนอยู่ในแท็กนี้

<strong title="8.5 based on 15,578 user ratings">8.5</ span>



คำสั่งต่อมาสั่งให้หาโหนดที่มี attribute value เป็น titleCast (มีเครื่องหมาย # นำหน้า) แล้วดูต่อ หา attribute ชื่อ itemprop (มีเครื่องหมาย . นำหน้า) แล้วต่อไปจนพบแท็ก span จึงส่งข้อมูลต่อ ให้ดึง text ออกมาซึ่งจะเป็นชื่อนักแสดง และเนื่องจากพบข้อมูลแบบนี้มากกว่าหนึ่ง จึงดึงทั้งหมด ออกมาเป็นรายการชื่อทั้งหมดได้

```
> cast <- arrival_movie %>%
```

```
+ html_nodes("#titleCast .itemprop span") %>%
```

```
+ html_text()
```

```
> cast
```

[1] "Amy Adams" "Jeremy Renner" "Michael Stuhlbarg" "Forest Whitaker" "Sangita Patel"

```
[6] "Mark O'Brien" "Abigail Pniowsky" "Tzi Ma" "Nathaly Thibault" "Ruth Chiang"
```

[11] "Jadyn Malone" "Julia Scarlett Dan" "Russell Yuen" "Anana Rydvald" "Leisa Reid"



จะเห็นว่าการใช้ rvest ช่วยให้เราสามารถดึงข้อมูลส่วนที่ต้องการในหน้าเว็บมาใช้ได้เลย เพียงแต่เรา ต้องรู้ว่าข้อมูลนั้นอยู่ที่ไหน ภายใต้แท็กอะไร เพื่อจะได้เขียนคำสั่งสกัดให้ถูกตำแหน่งได้
Topic modeling

Topic modeling เป็นการใช้วิธีการทางสถิติเพื่อหาว่าในเอกสารต่างๆที่มีนั้นมีหัวข้อหรือ topic อะไรอยู่บ้าง อาศัยหลักการสร้างเมทริกซ์ดูการกระจายตัวของคำต่างๆในเอกสารสร้าง Document-Term matrix ออกมา สมติว่ามีเอกสาร d ชิ้น และมีจำนวนรูปศัพท์ในเอกสาร ทั้งหมด n ศัพท์ ก็จะสามารถสร้างเมทริกซ์ขนาด d x n ได้ ซึ่งเมทริกซ์ d x n นี้ สามารถมองได้ ว่าเป็นผลมาจากการรวมกันของเทริกซ์ขนาด d x k และ k x n ได้ ซึ่งถ้าเรามองว่า k นี้คือจำนวน topic ทั้งหมดที่มีในข้อมูล เราก็สามารถหาคำตอบนี้ได้ เพราะจะได้ผลที่บอกว่า คำที่เกี่ยวข้องกับ topic ต่างๆ (1..k) มีคำอะไรบ้าง เกี่ยวข้องมากน้อยเพียงใด ส่วนผลอีกตารางจะบอกว่าเอกสาร แต่ละซิ้นมีความเกี่ยวข้องกับ topic ต่างๆ กี่เรื่องและแก่ยวข้องมากน้อยแค่ไหน อัลกอลิทึมพื้นฐาน ที่มักใช้กันในการทำ topic modeling คือ latent Dirichlet allocation (LDA; Blei, Ng, and Jordan 2003)



R มีโมดูลที่ช่วยคำนวณ topic modeling package ที่ต้องติดตั้ง คือ topicmodels และ tm เมื่อติดตั้งแล้วก็สามารโหลดโมดูล tm กับ topicmodels เข้ามาได้ ตัวอย่างข้างล่างดัดแปลง มาจาก "A gentle introduction to topic modeling using R"

```
> install.packages("topicmodels")
```

```
> install.packages("tm")
```

> library(tm)

```
> library(topicmodels)
```

กำหนด folder ที่เก็บไฟล์ข้อมูลไว้ ในที่นี้มีหนังสือที่โหลดมาจาก project Guthenberg จำนวน 8 เล่ม

> setwd("/Users/macbook/Cloud/Dropbox/course/Corpuslg/Corpus/Plato")

```
อ่านชื่อไฟล์ทั้งหมดใน working directory ที่เป็น txt
```

```
> filenames <- list.files(getwd(),pattern="*.txt")</pre>
```

อ่านข้อมูลทุกบรรทัดจากไฟล์มาเก็บไว้ที่ files ทีละไฟล์ แล้วแปลงเป็นเว็กเตอร์ด้วยคำสั่ง VectorSource แล้วรวมเข้าไว้ใน corpus ด้วยคำสั่ง Corpus เก็บไว้ที่ docs เมื่อเรียกดู docs จะเห็นรายงานว่าเป็น VCorpus หมายถึง Volatile Corpus คือเก็บไว้ในหน่วยความจำหากปิด โปรแกรม R คลังข้อมูลนี้ก็จะหายไป

```
> files <- lapply(filenames,readLines)
```

```
> docs <- Corpus(VectorSource(files))
```

```
> docs
```

```
<<VCorpus>>
```

Metadata: corpus specific: 0, document level (indexed): 0

Content: documents: 8

จากนั้นจึงเตรียมข้อมูลโดยทำเป็นตัวพิมพ์เล็กหมด ลบเครื่องหมายวรรคตอน ลบตัวเลข ลบ whitespace และลบคำที่เป็น stop word เสร็จแล้วแปลงคำเป็นรูปพื้นฐาน (stemming process)

> docs <-tm_map(docs,content_transformer(tolower))</pre>

```
> docs <- tm_map(docs, removePunctuation)
```

> docs <- tm_map(docs, removeNumbers)</pre>

> docs <- tm_map(docs, removeWords, stopwords("english"))

> docs <- tm_map(docs, stripWhitespace)

> docs <- tm_map(docs,stemDocument)</pre>

จาก document vectors ที่มีแปลงเป็นเมทริกซ์ของ document กับ term

> dtm <- DocumentTermMatrix(docs)

> dtm

<<DocumentTermMatrix (documents: 8, terms: 11858)>>

Non-/sparse entries: 29785/65079

Sparsity : 69%

Maximal term length: 32

Weighting : term frequency (tf)

dtm เก็บเมทริกซ์ของ Document-Term ซึ่งในที่นี้มี 8 documents และ 11,858 terms ค่าที่อยู่ในเมทริกซ์เป็นความถี่ของคำที่พบในแต่ละเอกสาร (term frequency)

> rownames(dtm) <- filenames

```
> freq <- colSums(as.matrix(dtm))
```

```
> length(freq)
```

[1] 11858

```
> write.csv(freq[ord],"word_freq.csv")
```

```
> ord <- order(freq,decreasing=TRUE)
```

นำชื่อไฟล์มาใส่เป็นชื่อ row ในเมทริกซ์ dtm และรวมค่าตัวเลขทุกคอลัมน์หรือทุก document ไว้ เป็นค่าความถี่รวมของแต่ละคำเก็บไว้ใน freq ใน freq จึงมีรายการคำพร้อมความถี่รวม แล้วเก็บ เป็นไฟล์ชื่อ word_freq.csv

```
> burnin <- 4000
```

> iter <- 2000

> thin <- 500

```
> seed <- list(2003,5,63,2456,765)
```

> nstart <- 5

> best <- TRUE

> k<-20

> ldaOut <-LDA(dtm,k, method="Gibbs", control=list(nstart=nstart, seed = seed, best=best, burnin = burnin, iter = iter, thin=thin))

LDA คือคำสั่ง Latent Dirichlet Allocation สำหรับการทำ topic modeling ซึ่งมีค่าที่ กำหนดให้ใช้กับ algorithm นี้ parameter แรกคือข้อมูลที่เป็นเมทริกซ์ของ Document-Term method ที่ใช้ sampling คือ Gibbs burnin เป็นการกำหนดรอบเริ่มต้นที่ไม่นำผลมาใช้ เนื่องจากช่วงแรกๆ เป็นการสุ่มกระจายคำไปตาม topic ต่างๆ แบบมั่วๆ จนวนรูปไปแล้ว burnin ครั้ง การกระจายคำตาม topic ต่าง ๆ จึงเริ่ม fit กับข้อมูลจริงมากขึ้น iter คือรอบหลังจากนั้นที่ จะให้ทำในที่นี้คือ 2000 รอบ k เป็นจำนวน topic ที่กำหนดว่าน่าจะจัดออกมาเป็น k topic nstart คือให้ลองเลือกจุดเริ่มต้นคำนวณ model ที่ต่างกันในที่นี้คือ 5 ซึ่งตำแหน่งจุดเริ่มต้นจะ กำหนดไว้ใน seed

หลังจากคำนวณหา model ที่ดีสุดมาได้แล้ว ก็เก็บผลที่ได้ไว้ใน ladOut แล้วจึงเก็บผลที่ ได้ลงไฟล์ตามตัวอย่างข้างล่างนี้

- > ldaOut.topics <- as.matrix(topics(ldaOut))
- > write.csv(ldaOut.topics,file=paste("LDAGibbs",k, "DocsToTopics.csv"))
- > ldaOut.terms <- as.matrix(terms(ldaOut,30))
- > write.csv(ldaOut.terms,file=paste("LDAGibbs",k, "TopicsToTerms.csv"))
- > topicProbabilities <- as.data.frame(ldaOut@gamma)
- > write.csv(topicProbabilities,file=paste("LDAGibbs",k, "TopicProbabilities.csv"))
- > topic1ToTopic2 <- lapply(1:nrow(dtm),function(x)
- + sort(topicProbabilities[x,])[k]/sort(topicProbabilities[x,]) [k-1])
- > topic2ToTopic3 <- lapply(1:nrow(dtm),function(x)</pre>
- + sort(topicProbabilities[x,])[k-1]/sort(topicProbabilities[x,]) [k-2])

> write.csv(topic1ToTopic2,file=paste("LDAGibbs",k, "Topic1ToTopic2.csv"))

> write.csv(topic2ToTopic3,file=paste("LDAGibbs",k, "Topic2ToTopic3.csv"))

References

- Baayen, R. H. 2008. Analyzing Linguistic Data: A Practical Introduction to Statistics Using R. Cambridge: Cambridge University Press.
- Blei DM, Ng AY, Jordan MI 2003. Latent Dirichlet Allocation. Journal of Machine Learning Research, 3, 993–1022
- Crawley, Michael J. 2005. Statistics: An Introduction Using R. 1st ed. Wiley.
- Paolillo, J. C. 2002. Analyzing linguistic variation : statistical models and methods. Stanford, Calif.: CSLI Publications, Center for the Study of Language and Information.
- Teetor, Paul. 2011. R Cookbook. 1st ed. O'Reilly Media.
- Woods, A., Fletcher, P. J. and Hughes, A. 1986. Statistics in language studies. Cambridge; New York: Cambridge University Press.
- Steyvers, Mark and Griffiths, Tom. 2007. Probabilistic Topic Models. in T. Landauer, D McNamara, S. Dennis, and W. Kintsch (eds), Latent Semantic Analysis: A Road to Meaning. Laurence Erlbaum (<u>http://psiexp.ss.uci.edu/research/papers/SteyversGriffithsLSABookFormatted.pdf</u>)

Additional

- R website : http://cran.r-project.org/
- R Tutorial : <u>http://www.r-bloggers.com/r-tutorial-series-r-beginners-guide-and-r-bloggers-updates/</u>)
- How I used R to create a word cloud, step by step <u>https://</u> georeferenced.wordpress.com/2013/01/15/rwordcloud/

- Building Wordclouds in R https://www.r-bloggers.com/building-wordclouds-in-r/
- How to use your favorite fonts in R charts <u>http://blog.revolutionanalytics.com/</u> 2012/09/how-to-use-your-favorite-fonts-in-r-charts.html
- A gentle introduction to topic modeling using R <u>https://</u> eight2late.wordpress.com/2015/09/29/a-gentle-introduction-to-topic-modelingusing-r/
- Hands-On Data Science with R Text Mining <u>http://onepager.togaware.com/</u> <u>TextMiningO.pdf</u>
- Ggplot2 cheat sheet : <u>https://www.rstudio.com/wp-content/uploads/2015/03/</u> ggplot2-cheatsheet.pdf