20190604 meeting

### User preferences modeling using dirichlet process mixture model for a content-based recommender system (Cami at al., 2019)

* 적용 분야 : **추천 시스템 (recommander system)**
  + 비모수 베이지안을 이용해 많은 모델과 서비스들이 개발됨 (ex-Nexflix, Amazon)
  + 추천 시스템에는 ’무엇을 재료로 삼에 시스템을 구축하는가?’에 따라 크게 다섯 종류가 있지만, 이 논문에서는 내용기반 추천시스템(content-based recommander system)에 집중
  + 추천 시스템은 확률적 모델이 들어가는 지에 따라 두 가지로 나눠볼 수 있다.
    - 사용자들의 특성을 가지고 주요한 feature를 모델링한다음 유사도를 계산해 추천을 하는 ‘structural modeling’
    - 확률 모형을 모델링한다음 사용자의 선택에 영향을 미치는 *assignment parameter*를 추정하는 ’behavioral model’이 존재.
    - 해당 논문은 **비모수 베이지안 기법을 이용해 여태 ’structural modeling’에 속하는 ’content based recommder system’에 확률적 모형을 적용하는 것.**(latent group의 수를 정하기 어렵고, 동적으로 변하는 환경에서 latent allocation을 update하기 어려움)
* 사용 알고리즘
  + DPMM 중 ddCRP (distance dependent CRP)
    - data similiarity matrix를 이용해 clustering.
* 사용 데이터 : **MovieLens 영화 평점**(추천 시스템 연구/개발에 유명한 데이터), news twitter(API로 수집)
  + MovieLens(<https://grouplens.org/>)
    - the University of Minnesota의 컴퓨터 공학과 소속 ’grouplens’에서 운영.(The MovieLens Datasets: History and Context , 2016)
    - 다양한 형태의 데이터셋 제공
      * MovieLens 20M Dataset(2015 Apr, 190MB, 27,000개 영화, 138,000 사용자, 12m scores)
      * MovieLens Latest Datasets(2018, 9)
        + Small (1MB, 9,000개 영화, 600 users, 100,000 ratings)
        + Full (265 MB, 58,000 영화, 280,000 사용자, 27,000,000 ratings)
    - 캐글에 EDA 존재(<https://www.kaggle.com/grouplens>)
    - 예시
* 추가
  + Nonparametric Bayesian Statistical Models in Biomedical Research(노희상 외 3명, 2014)
  + 비모수 베이지안 군집분석(김도현\_박사, 2009)
  + 비모수 베이지안 온라인 학습방법과 변분 게이트 순환 유닛(정구환\_박사, 2019)

### Rstan 설치

* 주요 설치 이슈
  + Rstudio는 1.2.x 이상으로
  + Windows 10에서 R 3.6.0과 Rtools 35의 조합은 추가 조치가 필요함
    - 2번 링크 참고
* stan 파일에서는 주석이 #이 아닌 //

1. <https://github.com/stan-dev/rstan/wiki/RStan-Getting-Started>
2. <https://discourse.mc-stan.org/t/rtools-not-properly-detected-in-r3-6-0-under-windows-10/9075>

* 예제 1 - BDA chpt 5.5 (2003)

library(rstan)

## Loading required package: ggplot2

## Registered S3 methods overwritten by 'ggplot2':  
## method from   
## [.quosures rlang  
## c.quosures rlang  
## print.quosures rlang

## Loading required package: StanHeaders

## rstan (Version 2.18.2, GitRev: 2e1f913d3ca3)

## For execution on a local, multicore CPU with excess RAM we recommend calling  
## options(mc.cores = parallel::detectCores()).  
## To avoid recompilation of unchanged Stan programs, we recommend calling  
## rstan\_options(auto\_write = TRUE)

## For improved execution time, we recommend calling  
## Sys.setenv(LOCAL\_CPPFLAGS = '-march=native')  
## although this causes Stan to throw an error on a few processors.

options(mc.cores = parallel::detectCores())  
rstan\_options(auto\_write = TRUE)  
Sys.setenv(LOCAL\_CPPFLAGS = '-march=native')

## https://github.com/stan-dev/rstan/wiki/RStan-Getting-Started  
## rstan exmaple  
### write the example  
  
schools\_dat <- list(J = 8,   
 y = c(28, 8, -3, 7, -1, 1, 18, 12),  
 sigma = c(15, 10, 16, 11, 9, 11, 10, 18))  
  
fit <- stan(file = '../source/8schools.stan', data = schools\_dat)

## Warning: There were 2 divergent transitions after warmup. Increasing adapt\_delta above 0.8 may help. See  
## http://mc-stan.org/misc/warnings.html#divergent-transitions-after-warmup

## Warning: Examine the pairs() plot to diagnose sampling problems

print(fit)

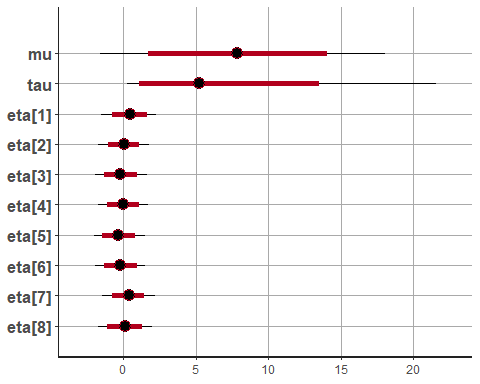
## Inference for Stan model: 8schools.  
## 4 chains, each with iter=2000; warmup=1000; thin=1;   
## post-warmup draws per chain=1000, total post-warmup draws=4000.  
##   
## mean se\_mean sd 2.5% 25% 50% 75% 97.5% n\_eff Rhat  
## mu 7.91 0.12 5.00 -1.59 4.66 7.82 11.04 18.07 1728 1  
## tau 6.61 0.16 5.75 0.29 2.52 5.23 9.12 21.59 1226 1  
## eta[1] 0.42 0.02 0.96 -1.53 -0.19 0.44 1.06 2.28 3799 1  
## eta[2] 0.01 0.01 0.88 -1.76 -0.57 0.02 0.59 1.74 4022 1  
## eta[3] -0.19 0.01 0.90 -1.93 -0.80 -0.21 0.42 1.61 4031 1  
## eta[4] -0.01 0.01 0.86 -1.71 -0.58 -0.03 0.52 1.70 4104 1  
## eta[5] -0.34 0.02 0.87 -2.00 -0.93 -0.37 0.22 1.49 3214 1  
## eta[6] -0.20 0.02 0.90 -1.92 -0.81 -0.20 0.40 1.53 3553 1  
## eta[7] 0.34 0.02 0.89 -1.46 -0.21 0.36 0.91 2.17 3489 1  
## eta[8] 0.09 0.02 0.96 -1.76 -0.56 0.10 0.71 1.96 4016 1  
## theta[1] 11.52 0.17 8.48 -1.85 6.15 10.31 15.62 31.76 2567 1  
## theta[2] 7.91 0.10 6.09 -4.28 4.15 7.92 11.55 20.38 3763 1  
## theta[3] 6.14 0.13 7.50 -10.74 2.11 6.68 10.87 19.67 3327 1  
## theta[4] 7.81 0.10 6.37 -4.75 3.92 7.76 11.60 21.04 4489 1  
## theta[5] 5.10 0.11 6.20 -8.51 1.41 5.55 9.31 16.39 3294 1  
## theta[6] 6.21 0.11 6.72 -8.13 2.21 6.65 10.51 18.96 3863 1  
## theta[7] 10.66 0.12 6.73 -1.20 6.19 9.97 14.44 26.44 2964 1  
## theta[8] 8.72 0.15 8.09 -7.31 4.15 8.34 12.99 27.41 2973 1  
## lp\_\_ -39.49 0.08 2.69 -45.58 -41.10 -39.20 -37.57 -35.00 1200 1  
##   
## Samples were drawn using NUTS(diag\_e) at Fri Jun 07 06:39:04 2019.  
## For each parameter, n\_eff is a crude measure of effective sample size,  
## and Rhat is the potential scale reduction factor on split chains (at   
## convergence, Rhat=1).

plot(fit)

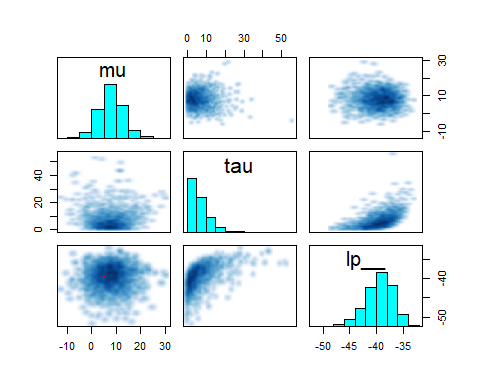
## 'pars' not specified. Showing first 10 parameters by default.

## ci\_level: 0.8 (80% intervals)

## outer\_level: 0.95 (95% intervals)



pairs(fit, pars = c("mu", "tau", "lp\_\_"))



la <- extract(fit, permuted = TRUE) # return a list of arrays   
mu <- la$mu

### return an array of three dimensions: iterations, chains, parameters   
a <- extract(fit, permuted = FALSE)

### use S3 functions on stanfit objects  
a2 <- as.array(fit)  
m <- as.matrix(fit)  
d <- as.data.frame(fit)

* 학회 지원 ㅠㅠㅠ
  + 카카오 : <https://careers.kakao.com/jobs/P-10476?fbclid=IwAR16pEQGzO5ChVkARWcVBlhvUEE1wN8cnWSrx_YxvGuLiqER7uR6tylV4NE>
  + 교내 국제학술대회 참가 지원금 : <http://graduate.cau.ac.kr/2018/board_view.php?no=3514&page=1&part=notice&category=%EC%9E%A5%ED%95%99>