AR Model

Jesse

2019年6月4日

library(tidyverse)

## -- Attaching packages --------------------------------------------------------------------------------- tidyverse 1.2.1 --

## √ ggplot2 3.1.1 √ purrr 0.3.2   
## √ tibble 2.0.1 √ dplyr 0.8.0.1  
## √ tidyr 0.8.2 √ stringr 1.4.0   
## √ readr 1.3.1 √ forcats 0.4.0

## -- Conflicts ------------------------------------------------------------------------------------ tidyverse\_conflicts() --  
## x dplyr::filter() masks stats::filter()  
## x dplyr::lag() masks stats::lag()

library(rethinking)

## Loading required package: rstan

## 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.

##   
## Attaching package: 'rstan'

## The following object is masked from 'package:tidyr':  
##   
## extract

## Loading required package: parallel

## rethinking (Version 1.59)

##   
## Attaching package: 'rethinking'

## The following object is masked from 'package:purrr':  
##   
## map

library(MLmetrics)

##   
## Attaching package: 'MLmetrics'

## The following object is masked from 'package:base':  
##   
## Recall

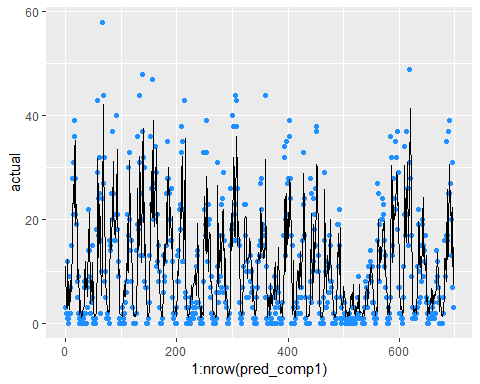
select = dplyr::select  
  
  
ubikedata = read.csv("file:///C:/Users/asus/Desktop/final project/XinyiSquare.csv")  
train = ubikedata[1:(24\*23),]  
test = ubikedata[(24\*23+1):721,]

## Backtest

AR1\_backtest\_model = "  
data {  
int<lower=0> N;  
vector[N] y;  
}  
parameters {  
real alpha;  
real beta;  
real beta2;  
real<lower=0> sigma;  
}  
model {  
for (n in 25:N)  
y[n] ~ normal(alpha + beta \* y[n-1] + beta2 \* y[n-24], sigma);  
alpha ~ normal(0,1);  
beta ~ normal(0,1);  
beta2 ~ normal(0,1);  
sigma ~ lognormal(0,1);  
}  
generated quantities {  
vector [N-24] pred\_y;  
for ( n in 25:N){  
pred\_y [n-24] = normal\_rng(alpha + beta \* y[n-1] + beta2 \* y[n-24], sigma);}  
}  
"  
  
data = list(  
 N = nrow(ubikedata),  
 y = ubikedata$quantity  
)  
  
fit1.1 = stan(  
 model\_code = AR1\_backtest\_model,  
 data = data  
)

##   
## SAMPLING FOR MODEL 'e5acb46e3d49d9a29bce3177931a227d' NOW (CHAIN 1).  
## Chain 1:   
## Chain 1: Gradient evaluation took 0 seconds  
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.  
## Chain 1: Adjust your expectations accordingly!  
## Chain 1:   
## Chain 1:   
## Chain 1: Iteration: 1 / 2000 [ 0%] (Warmup)  
## Chain 1: Iteration: 200 / 2000 [ 10%] (Warmup)  
## Chain 1: Iteration: 400 / 2000 [ 20%] (Warmup)  
## Chain 1: Iteration: 600 / 2000 [ 30%] (Warmup)  
## Chain 1: Iteration: 800 / 2000 [ 40%] (Warmup)  
## Chain 1: Iteration: 1000 / 2000 [ 50%] (Warmup)  
## Chain 1: Iteration: 1001 / 2000 [ 50%] (Sampling)  
## Chain 1: Iteration: 1200 / 2000 [ 60%] (Sampling)  
## Chain 1: Iteration: 1400 / 2000 [ 70%] (Sampling)  
## Chain 1: Iteration: 1600 / 2000 [ 80%] (Sampling)  
## Chain 1: Iteration: 1800 / 2000 [ 90%] (Sampling)  
## Chain 1: Iteration: 2000 / 2000 [100%] (Sampling)  
## Chain 1:   
## Chain 1: Elapsed Time: 1.197 seconds (Warm-up)  
## Chain 1: 1.102 seconds (Sampling)  
## Chain 1: 2.299 seconds (Total)  
## Chain 1:   
##   
## SAMPLING FOR MODEL 'e5acb46e3d49d9a29bce3177931a227d' NOW (CHAIN 2).  
## Chain 2:   
## Chain 2: Gradient evaluation took 0 seconds  
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.  
## Chain 2: Adjust your expectations accordingly!  
## Chain 2:   
## Chain 2:   
## Chain 2: Iteration: 1 / 2000 [ 0%] (Warmup)  
## Chain 2: Iteration: 200 / 2000 [ 10%] (Warmup)  
## Chain 2: Iteration: 400 / 2000 [ 20%] (Warmup)  
## Chain 2: Iteration: 600 / 2000 [ 30%] (Warmup)  
## Chain 2: Iteration: 800 / 2000 [ 40%] (Warmup)  
## Chain 2: Iteration: 1000 / 2000 [ 50%] (Warmup)  
## Chain 2: Iteration: 1001 / 2000 [ 50%] (Sampling)  
## Chain 2: Iteration: 1200 / 2000 [ 60%] (Sampling)  
## Chain 2: Iteration: 1400 / 2000 [ 70%] (Sampling)  
## Chain 2: Iteration: 1600 / 2000 [ 80%] (Sampling)  
## Chain 2: Iteration: 1800 / 2000 [ 90%] (Sampling)  
## Chain 2: Iteration: 2000 / 2000 [100%] (Sampling)  
## Chain 2:   
## Chain 2: Elapsed Time: 1.024 seconds (Warm-up)  
## Chain 2: 1.106 seconds (Sampling)  
## Chain 2: 2.13 seconds (Total)  
## Chain 2:   
##   
## SAMPLING FOR MODEL 'e5acb46e3d49d9a29bce3177931a227d' NOW (CHAIN 3).  
## Chain 3:   
## Chain 3: Gradient evaluation took 0 seconds  
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.  
## Chain 3: Adjust your expectations accordingly!  
## Chain 3:   
## Chain 3:   
## Chain 3: Iteration: 1 / 2000 [ 0%] (Warmup)  
## Chain 3: Iteration: 200 / 2000 [ 10%] (Warmup)  
## Chain 3: Iteration: 400 / 2000 [ 20%] (Warmup)  
## Chain 3: Iteration: 600 / 2000 [ 30%] (Warmup)  
## Chain 3: Iteration: 800 / 2000 [ 40%] (Warmup)  
## Chain 3: Iteration: 1000 / 2000 [ 50%] (Warmup)  
## Chain 3: Iteration: 1001 / 2000 [ 50%] (Sampling)  
## Chain 3: Iteration: 1200 / 2000 [ 60%] (Sampling)  
## Chain 3: Iteration: 1400 / 2000 [ 70%] (Sampling)  
## Chain 3: Iteration: 1600 / 2000 [ 80%] (Sampling)  
## Chain 3: Iteration: 1800 / 2000 [ 90%] (Sampling)  
## Chain 3: Iteration: 2000 / 2000 [100%] (Sampling)  
## Chain 3:   
## Chain 3: Elapsed Time: 1.069 seconds (Warm-up)  
## Chain 3: 0.81 seconds (Sampling)  
## Chain 3: 1.879 seconds (Total)  
## Chain 3:   
##   
## SAMPLING FOR MODEL 'e5acb46e3d49d9a29bce3177931a227d' NOW (CHAIN 4).  
## Chain 4:   
## Chain 4: Gradient evaluation took 0 seconds  
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.  
## Chain 4: Adjust your expectations accordingly!  
## Chain 4:   
## Chain 4:   
## Chain 4: Iteration: 1 / 2000 [ 0%] (Warmup)  
## Chain 4: Iteration: 200 / 2000 [ 10%] (Warmup)  
## Chain 4: Iteration: 400 / 2000 [ 20%] (Warmup)  
## Chain 4: Iteration: 600 / 2000 [ 30%] (Warmup)  
## Chain 4: Iteration: 800 / 2000 [ 40%] (Warmup)  
## Chain 4: Iteration: 1000 / 2000 [ 50%] (Warmup)  
## Chain 4: Iteration: 1001 / 2000 [ 50%] (Sampling)  
## Chain 4: Iteration: 1200 / 2000 [ 60%] (Sampling)  
## Chain 4: Iteration: 1400 / 2000 [ 70%] (Sampling)  
## Chain 4: Iteration: 1600 / 2000 [ 80%] (Sampling)  
## Chain 4: Iteration: 1800 / 2000 [ 90%] (Sampling)  
## Chain 4: Iteration: 2000 / 2000 [100%] (Sampling)  
## Chain 4:   
## Chain 4: Elapsed Time: 1.042 seconds (Warm-up)  
## Chain 4: 0.917 seconds (Sampling)  
## Chain 4: 1.959 seconds (Total)  
## Chain 4:

post1.1 = as.data.frame(fit1.1)  
post1.1 = post1.1 %>% select(contains('pred\_y'))  
  
actual\_data =ubikedata$quantity[25:nrow(ubikedata)]  
pred\_comp1 = data.frame(actual = actual\_data,  
 pred = post1.1 %>% apply(MARGIN = 2, FUN = mean),  
 l\_PI = post1.1 %>% apply(MARGIN = 2, FUN = PI) %>% .[1,],  
 h\_PI = post1.1 %>% apply(MARGIN = 2, FUN = PI) %>% .[2,])  
pred\_comp1 %>% ggplot() +  
 geom\_point(aes(x=1:nrow(pred\_comp1), y=actual),color='dodgerblue') +  
 geom\_line(aes(x=1:nrow(pred\_comp1),y=pred))



RMSE(y\_pred = pred\_comp1$pred, y\_true = pred\_comp1$actual)

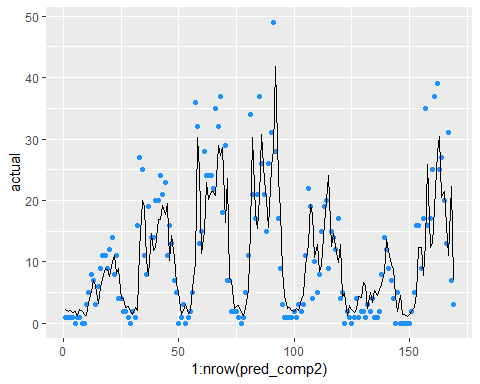
## [1] 7.53769

## Train and Test

AR1\_tandp\_model = "  
data {  
 int<lower=0> N\_train;  
 vector[N\_train] y;  
 int N\_test;  
 int N\_total;  
 vector[N\_total] y2;  
}  
parameters {  
 real alpha;  
 real beta;  
 real beta2;  
 real<lower=0> sigma;  
}  
model {  
 for (n in 25:N\_train)  
 y[n] ~ normal(alpha + beta \* y[n-1] + beta2 \* y[n-24], sigma);  
 alpha ~ normal(0,1);  
 beta ~ normal(0,1);  
 beta2 ~ normal(0,1);  
 sigma ~ lognormal(0,1);  
}  
generated quantities {  
 vector [N\_test] pred\_y;  
for ( n in N\_train+1:N\_total){  
pred\_y [n-N\_train] = normal\_rng(alpha + beta \* y2[n-1] + beta2 \* y2[n-24], sigma);}  
}  
"  
  
data = list(  
 N\_train = nrow(train),  
 y = train$quantity,  
 N\_test = nrow(test),  
 N\_total = nrow(ubikedata),  
 y2 = ubikedata$quantity  
)  
  
fit1.2 = stan(  
 model\_code = AR1\_tandp\_model,  
 data = data  
 )

##   
## SAMPLING FOR MODEL '73e112cc979b6bb668d8fa2c5608c53c' NOW (CHAIN 1).  
## Chain 1:   
## Chain 1: Gradient evaluation took 0 seconds  
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.  
## Chain 1: Adjust your expectations accordingly!  
## Chain 1:   
## Chain 1:   
## Chain 1: Iteration: 1 / 2000 [ 0%] (Warmup)  
## Chain 1: Iteration: 200 / 2000 [ 10%] (Warmup)  
## Chain 1: Iteration: 400 / 2000 [ 20%] (Warmup)  
## Chain 1: Iteration: 600 / 2000 [ 30%] (Warmup)  
## Chain 1: Iteration: 800 / 2000 [ 40%] (Warmup)  
## Chain 1: Iteration: 1000 / 2000 [ 50%] (Warmup)  
## Chain 1: Iteration: 1001 / 2000 [ 50%] (Sampling)  
## Chain 1: Iteration: 1200 / 2000 [ 60%] (Sampling)  
## Chain 1: Iteration: 1400 / 2000 [ 70%] (Sampling)  
## Chain 1: Iteration: 1600 / 2000 [ 80%] (Sampling)  
## Chain 1: Iteration: 1800 / 2000 [ 90%] (Sampling)  
## Chain 1: Iteration: 2000 / 2000 [100%] (Sampling)  
## Chain 1:   
## Chain 1: Elapsed Time: 0.763 seconds (Warm-up)  
## Chain 1: 0.852 seconds (Sampling)  
## Chain 1: 1.615 seconds (Total)  
## Chain 1:   
##   
## SAMPLING FOR MODEL '73e112cc979b6bb668d8fa2c5608c53c' NOW (CHAIN 2).  
## Chain 2:   
## Chain 2: Gradient evaluation took 0 seconds  
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.  
## Chain 2: Adjust your expectations accordingly!  
## Chain 2:   
## Chain 2:   
## Chain 2: Iteration: 1 / 2000 [ 0%] (Warmup)  
## Chain 2: Iteration: 200 / 2000 [ 10%] (Warmup)  
## Chain 2: Iteration: 400 / 2000 [ 20%] (Warmup)  
## Chain 2: Iteration: 600 / 2000 [ 30%] (Warmup)  
## Chain 2: Iteration: 800 / 2000 [ 40%] (Warmup)  
## Chain 2: Iteration: 1000 / 2000 [ 50%] (Warmup)  
## Chain 2: Iteration: 1001 / 2000 [ 50%] (Sampling)  
## Chain 2: Iteration: 1200 / 2000 [ 60%] (Sampling)  
## Chain 2: Iteration: 1400 / 2000 [ 70%] (Sampling)  
## Chain 2: Iteration: 1600 / 2000 [ 80%] (Sampling)  
## Chain 2: Iteration: 1800 / 2000 [ 90%] (Sampling)  
## Chain 2: Iteration: 2000 / 2000 [100%] (Sampling)  
## Chain 2:   
## Chain 2: Elapsed Time: 0.763 seconds (Warm-up)  
## Chain 2: 0.747 seconds (Sampling)  
## Chain 2: 1.51 seconds (Total)  
## Chain 2:   
##   
## SAMPLING FOR MODEL '73e112cc979b6bb668d8fa2c5608c53c' NOW (CHAIN 3).  
## Chain 3:   
## Chain 3: Gradient evaluation took 0 seconds  
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.  
## Chain 3: Adjust your expectations accordingly!  
## Chain 3:   
## Chain 3:   
## Chain 3: Iteration: 1 / 2000 [ 0%] (Warmup)  
## Chain 3: Iteration: 200 / 2000 [ 10%] (Warmup)  
## Chain 3: Iteration: 400 / 2000 [ 20%] (Warmup)  
## Chain 3: Iteration: 600 / 2000 [ 30%] (Warmup)  
## Chain 3: Iteration: 800 / 2000 [ 40%] (Warmup)  
## Chain 3: Iteration: 1000 / 2000 [ 50%] (Warmup)  
## Chain 3: Iteration: 1001 / 2000 [ 50%] (Sampling)  
## Chain 3: Iteration: 1200 / 2000 [ 60%] (Sampling)  
## Chain 3: Iteration: 1400 / 2000 [ 70%] (Sampling)  
## Chain 3: Iteration: 1600 / 2000 [ 80%] (Sampling)  
## Chain 3: Iteration: 1800 / 2000 [ 90%] (Sampling)  
## Chain 3: Iteration: 2000 / 2000 [100%] (Sampling)  
## Chain 3:   
## Chain 3: Elapsed Time: 0.822 seconds (Warm-up)  
## Chain 3: 0.704 seconds (Sampling)  
## Chain 3: 1.526 seconds (Total)  
## Chain 3:   
##   
## SAMPLING FOR MODEL '73e112cc979b6bb668d8fa2c5608c53c' NOW (CHAIN 4).  
## Chain 4:   
## Chain 4: Gradient evaluation took 0 seconds  
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.  
## Chain 4: Adjust your expectations accordingly!  
## Chain 4:   
## Chain 4:   
## Chain 4: Iteration: 1 / 2000 [ 0%] (Warmup)  
## Chain 4: Iteration: 200 / 2000 [ 10%] (Warmup)  
## Chain 4: Iteration: 400 / 2000 [ 20%] (Warmup)  
## Chain 4: Iteration: 600 / 2000 [ 30%] (Warmup)  
## Chain 4: Iteration: 800 / 2000 [ 40%] (Warmup)  
## Chain 4: Iteration: 1000 / 2000 [ 50%] (Warmup)  
## Chain 4: Iteration: 1001 / 2000 [ 50%] (Sampling)  
## Chain 4: Iteration: 1200 / 2000 [ 60%] (Sampling)  
## Chain 4: Iteration: 1400 / 2000 [ 70%] (Sampling)  
## Chain 4: Iteration: 1600 / 2000 [ 80%] (Sampling)  
## Chain 4: Iteration: 1800 / 2000 [ 90%] (Sampling)  
## Chain 4: Iteration: 2000 / 2000 [100%] (Sampling)  
## Chain 4:   
## Chain 4: Elapsed Time: 0.851 seconds (Warm-up)  
## Chain 4: 0.822 seconds (Sampling)  
## Chain 4: 1.673 seconds (Total)  
## Chain 4:

post1.2 = as.data.frame(fit1.2) %>%   
 select(contains('pred\_y'))  
  
actual\_data2 =test$quantity  
pred\_comp2 = data.frame(actual = actual\_data2,  
 pred = post1.2 %>% apply(MARGIN = 2, FUN = mean),  
 l\_PI = post1.2 %>% apply(MARGIN = 2, FUN = PI) %>% .[1,],  
 h\_PI = post1.2 %>% apply(MARGIN = 2, FUN = PI) %>% .[2,])  
pred\_comp2 %>% ggplot() +  
 geom\_point(aes(x=1:nrow(pred\_comp2), y=actual),color='dodgerblue') +  
 geom\_line(aes(x=1:nrow(pred\_comp2),y=pred))



RMSE(y\_pred = pred\_comp2$pred, y\_true = pred\_comp2$actual)

## [1] 6.881255