Pricing Analytics

```r
library("dummies")

library("AER")

library("plotly")

library('RColorBrewer')
library("data.table")
library("mlogit")

library("gmnl")



#load data
data=fread("kiwi_bubbles_P2.csv",stringsAsFactors = F)
demo=fread("demo_P2.csv",stringsAsFactors = F)

#Drop observations with stockout
data=data[!(data$price.KB==99),]
data=data[!(data$price.KR==99),]
data=data[!(data$price.MB==99),]

#pass data to the algorithm
mlogitdata=mlogit.data(data,id="id",varying=4:7,choice="choice",shape="wide")
#MLE
mle= gmnl(choice ~  price, data = mlogitdata)
summary(mle)

##
## Model estimated on: Mon Mar 09 14:02:41 2020
##
## Call:
## gmnl(formula = choice ~ price, data = mlogitdata, method = "nr")
##
## Frequencies of categories:
##
##       0      KB      KR      MB
## 0.41564 0.18035 0.20039 0.20362
##
## The estimation took: 0h:0m:0s
##
## Coefficients:
##                Estimate Std. Error z-value  Pr(>|z|)
## KB:(intercept)  4.25316    0.32821  12.959 < 2.2e-16 ***
## KR:(intercept)  4.36240    0.32945  13.241 < 2.2e-16 ***
## MB:(intercept)  4.20440    0.31331  13.419 < 2.2e-16 ***
## price          -3.73793    0.23671 -15.791 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Optimization of log-likelihood by Newton-Raphson maximisation
## Log Likelihood: -1909
## Number of observations: 1547
## Number of iterations: 4
## Exit of MLE: gradient close to zero
```

```
# average price for each product
avgKB = mean(data$price.KB)
avgMB = mean(data$price.MB)
avgKR = mean(data$price.KR)

# set parameter as coefficients
para = as.numeric(mle$coefficients)

#define probability that a consumer chooses each product (demand):
Multinomial logit model
demand_KB=function(priceKB,priceKR,priceMB,para){

prob=exp(para[1]+para[4]*priceKB)/(1+exp(para[1]+para[4]*priceKB)+exp(para[2]
+para[4]*priceKR)+exp(para[3]+para[4]*priceMB))
  return(prob)
}

demand_KR=function(priceKB,priceKR,priceMB,para){
  prob=exp(para[2]+para[4]*priceKR)/

(1+exp(para[1]+para[4]*priceKB)+exp(para[2]+para[4]*priceKR)+exp(para[3]+para
[4]*priceMB))
  return(prob)
}

demand_MB=function(priceKB,priceKR,priceMB,para){
  prob=exp(para[3]+para[4]*priceMB)/

(1+exp(para[1]+para[4]*priceKB)+exp(para[2]+para[4]*priceKR)+exp(para[3]+para
[4]*priceMB))
  return(prob)
}

#KB
probKB = demand_KB(avgKB,avgKR,avgMB,para)
probKB
```

```
## [1] 0.1753666
```

```
#KR
probKR = demand_KR(avgKB,avgKR,avgMB,para)
probKR

## [1] 0.1979974

#MB
probMB = demand_MB(avgKB,avgKR,avgMB,para)
probMB

## [1] 0.1908972

OwnKB = -para[4]*avgKB*(1-probKB)
CrossKB = -para[4]*avgKB*probKB

OwnKR = -para[4]*avgKR*(1-probKR)
CrossKR = -para[4]*avgKR*probKR

OwnMB = -para[4]*avgMB*(1-probMB)
CrossMB = -para[4]*avgMB*probMB

#define demand function when own two products: KB and KR
demand=function(priceKB,priceKR,priceMB,para){
  probKB=exp(para[1]+para[4]*priceKB)/

(1+exp(para[1]+para[4]*priceKB)+exp(para[2]+para[4]*priceKR)+exp(para[3]+para
[4]*priceMB))
  probKR=exp(para[2]+para[4]*priceKR)/

(1+exp(para[1]+para[4]*priceKB)+exp(para[2]+para[4]*priceKR)+exp(para[3]+para
[4]*priceMB))
  return(cbind(probKB,probKR))
}

#cost
uc=0.5

#define profit function
profit=function(priceKB,priceKR,priceMB,para){
  profitKB=demand(priceKB,priceKR,priceMB,para)[,1]*(priceKB-uc)
  profitKR=demand(priceKB,priceKR,priceMB,para)[,2]*(priceKR-uc)
  return(cbind(profitKB,profitKR))
}

#price for two products
aux=seq(1,3,0.01)
pricespace=expand.grid(aux,aux)

#total profit for two products
profitmat=matrix(0L,nrow(pricespace),1)
```

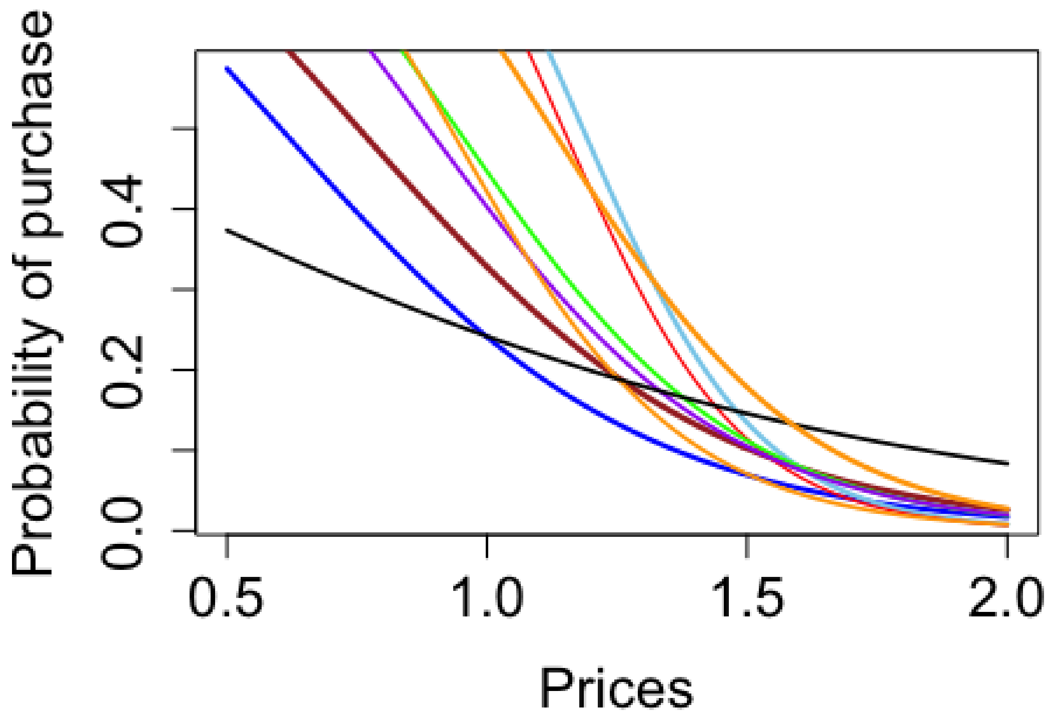
```
for (i in 1:nrow(pricespace)){
  profitmat[i]=sum(profit(pricespace[i,1],pricespace[i,2],1.43,para))*1000
}

#Draw figure
xaxis=list(title="P^{KB}")
yaxis=list(autorange = "reversed",title="P^{KR}")
zaxis=list(title="Profit")
p=plot_ly(x=pricespace[,1],y=pricespace[,2],z=as.numeric(profitmat),
          type="scatter3d",mode="markers",
          marker = list(color = as.numeric(profitmat), colorscale =
c('#FFE1A1', '#683531'), showscale = TRUE))%>%
  layout(scene=list(xaxis=xaxis,yaxis=yaxis,zaxis=zaxis))%>%
  config(mathjax = 'cdn')
p
```

WebGL is not supported by your browser - visit https://get.webgl.org for more info

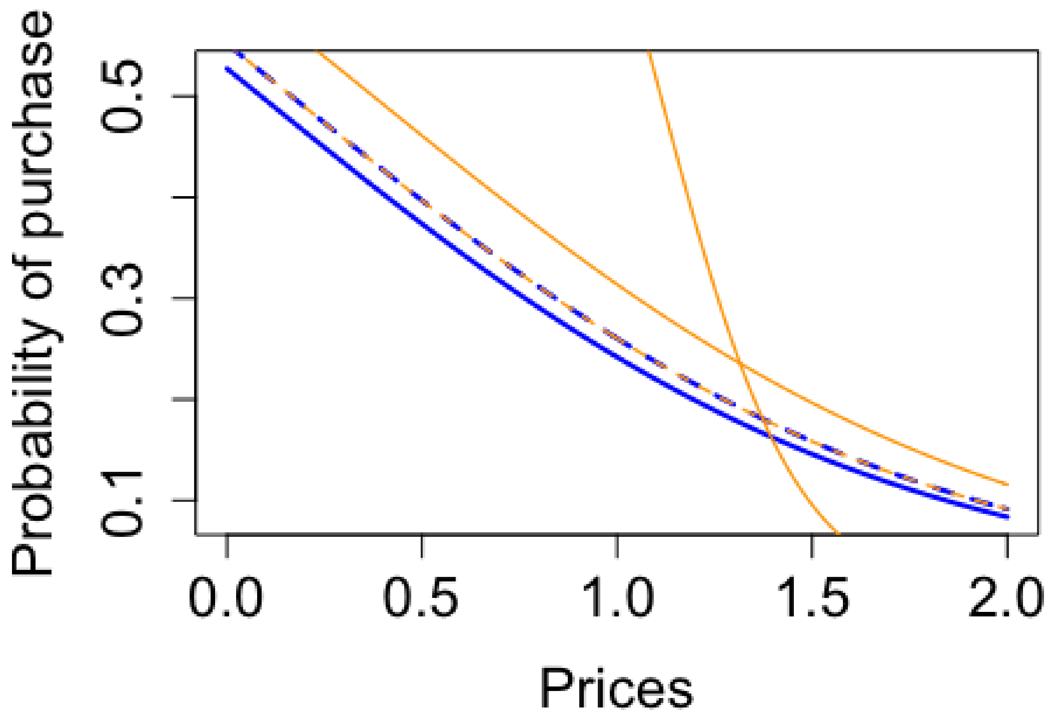
```
#optimal price
optPrice = pricespace[profitmat==max(profitmat)]
optPrice

## [1] 1.16 1.16

profitmat[profitmat==max(profitmat)]
```

```
## [1] 393.4082

#Number of individuals
N = 359
#kmeans
set.seed(0)
demo_cluster = kmeans(x=demo[, 2:18], centers = 8, nstart = 1000)
#merge data
cluster_id = data.frame(id = demo$id)
cluster_id$cluster = demo_cluster$cluster
data = merge(data, cluster_id, by = "id", all.x = T)
data$cluster[is.na(data$cluster)] = 9
seg.share = c( table(demo_cluster$cluster),N -
sum(table(demo_cluster$cluster))) / N
coef.est = data.frame(segment = 1:8, intercept.KB = NA, intercept.KR = NA,
                      intercept.MB = NA, price.coef = NA)
for (seg in 1:9) {
  data.sub = subset(data, cluster == seg)

mlogitdata=mlogit.data(data.sub,id="id",varying=4:7,choice="choice",shape="wi
de")
  mle= gmnl(choice ~  price, data = mlogitdata)
  mle
  coef.est[seg, 2:5] = mle$coefficients
}

#aggregate demand
agg_choice=function(demand,priceKB,priceKR,priceMB) {


agg_choice=seg.share[1]*demand(priceKB,priceKR,priceMB,as.numeric(coef.est[1,
2:5]))+
    seg.share[2]*demand(priceKB,priceKR,priceMB,as.numeric(coef.est[2,2:5]))+
    seg.share[3]*demand(priceKB,priceKR,priceMB,as.numeric(coef.est[3,2:5]))+
    seg.share[4]*demand(priceKB,priceKR,priceMB,as.numeric(coef.est[4,2:5]))+
    seg.share[5]*demand(priceKB,priceKR,priceMB,as.numeric(coef.est[5,2:5]))+
    seg.share[6]*demand(priceKB,priceKR,priceMB,as.numeric(coef.est[6,2:5]))+
    seg.share[7]*demand(priceKB,priceKR,priceMB,as.numeric(coef.est[7,2:5]))+
    seg.share[8]*demand(priceKB,priceKR,priceMB,as.numeric(coef.est[8,2:5]))+
    seg.share[9]*demand(priceKB,priceKR,priceMB,as.numeric(coef.est[9,2:5]))

  return(agg_choice)
}

aggKB = agg_choice(demand_KB,avgKB,avgKR,avgMB)
aggKB

##         1
## 0.1779914
```

```
aggKR = agg_choice(demand_KR,avgKB,avgKR,avgMB)
aggKR

##       1
## 0.20038

aggMB = agg_choice(demand_MB,avgKB,avgKR,avgMB)
aggMB

##         1
## 0.1890433

#Plot results
pricespace=seq(0.5,2,0.01)
plot(pricespace,demand_KB(pricespace,avgKR,avgMB,as.numeric(coef.est[1,2:5]))
,type='l',xlab='Prices',
     ylab='Probability of purchase',col="blue",lwd=20*seg.share[1],
     cex=2,cex.lab=1.5, cex.axis=1.5, cex.main=1.5, cex.sub=1.5)
lines(pricespace,demand_KB(pricespace,avgKR,avgMB,as.numeric(coef.est[2,2:5])
),col="brown",lwd=20*seg.share[2])
lines(pricespace,demand_KB(pricespace,avgKR,avgMB,as.numeric(coef.est[3,2:5])
),col="sky blue",lwd=20*seg.share[3])
lines(pricespace,demand_KB(pricespace,avgKR,avgMB,as.numeric(coef.est[4,2:5])
),col="red",lwd=20*seg.share[4])
lines(pricespace,demand_KB(pricespace,avgKR,avgMB,as.numeric(coef.est[5,2:5])
),col="green",lwd=20*seg.share[5])
lines(pricespace,demand_KB(pricespace,avgKR,avgMB,as.numeric(coef.est[6,2:5])
),col="purple",lwd=20*seg.share[5])
lines(pricespace,demand_KB(pricespace,avgKR,avgMB,as.numeric(coef.est[7,2:5])
),col="orange",lwd=20*seg.share[5])
lines(pricespace,demand_KB(pricespace,avgKR,avgMB,as.numeric(coef.est[8,2:5])
),col="black",lwd=20*seg.share[5])
lines(pricespace,demand_KB(pricespace,avgKR,avgMB,as.numeric(coef.est[9,2:5])
),col="orange",lwd=20*seg.share[6])
```



```
pricespace=seq(0,2,0.01)
plot(pricespace,demand_KB(pricespace,avgKR,avgMB,as.numeric(coef.est[8,2:5]))
,type='l',xlab='Prices',
     ylab='Probability of purchase',col="blue",lwd=20*seg.share[3]
     ,cex=2,cex.lab=1.5, cex.axis=1.5, cex.main=1.5, cex.sub=1.5)
lines(pricespace,demand_KB(pricespace,avgKR+1,avgMB,as.numeric(coef.est[8,2:5
])),col="orange",lwd=20*seg.share[4])
lines(pricespace,demand_KB(pricespace,avgKR,avgMB+1,as.numeric(coef.est[8,2:5
])),col="blue",lwd=20*seg.share[3],lty=2)

lines(pricespace,demand_KB(pricespace,avgKR,avgMB+1,as.numeric(coef.est[8,2:5
])),col="orange",lwd=20*seg.share[4],lty=2)
lines(pricespace,demand_KR(avgKB,pricespace,avgMB,as.numeric(coef.est[4,2:5])
),col="orange",lwd=20*seg.share[4])
```



```
#own-elasticity
elaFun = function(aggData,demand,selfPrice){
  ela = -(selfPrice/aggData)*sum(seg.share[1]*coef.est[1,5]*

demand(avgKB,avgKR,avgMB,as.numeric(coef.est[1,2:5]))*
                                   (1-
demand(avgKB,avgKR,avgMB,as.numeric(coef.est[1,2:5]))),
                                 seg.share[2]*coef.est[2,5]*

demand(avgKB,avgKR,avgMB,as.numeric(coef.est[2,2:5]))*
                                   (1-
demand(avgKB,avgKR,avgMB,as.numeric(coef.est[2,2:5]))),
                                 seg.share[3]*coef.est[3,5]*

demand(avgKB,avgKR,avgMB,as.numeric(coef.est[3,2:5]))*
                                   (1-
demand(avgKB,avgKR,avgMB,as.numeric(coef.est[3,2:5]))),
                                 seg.share[4]*coef.est[4,5]*

demand(avgKB,avgKR,avgMB,as.numeric(coef.est[4,2:5]))*
                                   (1-
demand(avgKB,avgKR,avgMB,as.numeric(coef.est[4,2:5]))),
                                 seg.share[5]*coef.est[5,5]*
```

```
demand(avgKB,avgKR,avgMB,as.numeric(coef.est[5,2:5]))*
                                   (1-
demand(avgKB,avgKR,avgMB,as.numeric(coef.est[5,2:5]))),
                                 seg.share[6]*coef.est[6,5]*

demand(avgKB,avgKR,avgMB,as.numeric(coef.est[6,2:5]))*
                                   (1-
demand(avgKB,avgKR,avgMB,as.numeric(coef.est[6,2:5]))),
                                 seg.share[7]*coef.est[7,5]*

demand(avgKB,avgKR,avgMB,as.numeric(coef.est[7,2:5]))*
                                   (1-
demand(avgKB,avgKR,avgMB,as.numeric(coef.est[7,2:5]))),
                                 seg.share[8]*coef.est[8,5]*

demand(avgKB,avgKR,avgMB,as.numeric(coef.est[8,2:5]))*
                                   (1-
demand(avgKB,avgKR,avgMB,as.numeric(coef.est[8,2:5]))),
                                 seg.share[9]*coef.est[9,5]*

demand(avgKB,avgKR,avgMB,as.numeric(coef.est[9,2:5]))*
                                   (1-
demand(avgKB,avgKR,avgMB,as.numeric(coef.est[9,2:5]))))
  return(ela)
}

KBElas = elaFun(aggKB,demand_KB,avgKB)
KRElas = elaFun(aggKR,demand_KR,avgKR)
MBElas = elaFun(aggMB,demand_MB,avgMB)


#cross-elasticity
crosselaFun = function(aggData,competitorPrice,demandSelf,demandCompetitor){
  ela = -(competitorPrice/aggData)*sum(seg.share[1]*coef.est[1,5]*

demandSelf(avgKB,avgKR,avgMB,as.numeric(coef.est[1,2:5]))*

demandCompetitor(avgKB,avgKR,avgMB,as.numeric(coef.est[1,2:5])),
                                       seg.share[2]*coef.est[2,5]*

demandSelf(avgKB,avgKR,avgMB,as.numeric(coef.est[2,2:5]))*

demandCompetitor(avgKB,avgKR,avgMB,as.numeric(coef.est[2,2:5])),
                                       seg.share[3]*coef.est[3,5]*

demandSelf(avgKB,avgKR,avgMB,as.numeric(coef.est[3,2:5]))*

demandCompetitor(avgKB,avgKR,avgMB,as.numeric(coef.est[3,2:5])),
                                       seg.share[4]*coef.est[4,5]*
```

```
demandSelf(avgKB,avgKR,avgMB,as.numeric(coef.est[4,2:5]))*

demandCompetitor(avgKB,avgKR,avgMB,as.numeric(coef.est[4,2:5])),
                                       seg.share[5]*coef.est[5,5]*

demandSelf(avgKB,avgKR,avgMB,as.numeric(coef.est[5,2:5]))*

demandCompetitor(avgKB,avgKR,avgMB,as.numeric(coef.est[5,2:5])),
                                       seg.share[6]*coef.est[6,5]*

demandSelf(avgKB,avgKR,avgMB,as.numeric(coef.est[6,2:5]))*

demandCompetitor(avgKB,avgKR,avgMB,as.numeric(coef.est[6,2:5])),
                                       seg.share[7]*coef.est[7,5]*

demandSelf(avgKB,avgKR,avgMB,as.numeric(coef.est[7,2:5]))*

demandCompetitor(avgKB,avgKR,avgMB,as.numeric(coef.est[7,2:5])),
                                       seg.share[8]*coef.est[8,5]*

demandSelf(avgKB,avgKR,avgMB,as.numeric(coef.est[8,2:5]))*

demandCompetitor(avgKB,avgKR,avgMB,as.numeric(coef.est[8,2:5])),
                                       seg.share[9]*coef.est[9,5]*

demandSelf(avgKB,avgKR,avgMB,as.numeric(coef.est[9,2:5]))*

demandCompetitor(avgKB,avgKR,avgMB,as.numeric(coef.est[9,2:5])))
  return(ela)
}

CrossKB_MB=crosselaFun(aggKB,avgMB,demand_KB,demand_MB)
CrossKB_KR=crosselaFun(aggKB,avgKR,demand_KB,demand_KR)
CrossMB_KB=crosselaFun(aggMB,avgKB,demand_MB,demand_KB)
CrossMB_KR=crosselaFun(aggMB,avgKR,demand_MB,demand_KR)
CrossKR_KB=crosselaFun(aggKR,avgKB,demand_KR,demand_KB)
CrossKR_MB=crosselaFun(aggKR,avgMB,demand_KR,demand_MB)
CrossKB_MB

##        1
## 1.075592

CrossKB_KR

##         1
## 0.9130573

CrossMB_KB
```

```
##        1
## 1.039614
```

```
CrossMB_KR
```

```
##         1
## 0.8991862
```

```
CrossKR_KB
```

```
##         1
## 0.8129505
```

```
CrossKR_MB
```

```
##         1
## 0.8283068
```

```
#not launch KB
demand_new=function(priceKR,priceMB,para){

prob=exp(para[1]+para[3]*priceKR)/(1+exp(para[1]+para[3]*priceKR)+exp(para[2]
+para[3]*priceMB))
  return(prob)
}

uc=.5
Aux = as.matrix(aux)
profitSum=function(priceKR,para){
profitSum=1000*(seg.share[1]*demand_new(priceKR,1.43,as.numeric(coef.est[1,3:
5]))*(priceKR-uc)+

seg.share[2]*demand_new(priceKR,1.43,as.numeric(coef.est[2,3:5]))*(priceKR-
uc)+

seg.share[3]*demand_new(priceKR,1.43,as.numeric(coef.est[3,3:5]))*(priceKR-
uc)+

seg.share[4]*demand_new(priceKR,1.43,as.numeric(coef.est[4,3:5]))*(priceKR-
uc)+

seg.share[5]*demand_new(priceKR,1.43,as.numeric(coef.est[5,3:5]))*(priceKR-
uc)+

seg.share[6]*demand_new(priceKR,1.43,as.numeric(coef.est[6,3:5]))*(priceKR-
uc)+

seg.share[7]*demand_new(priceKR,1.43,as.numeric(coef.est[7,3:5]))*(priceKR-
uc)+

seg.share[8]*demand_new(priceKR,1.43,as.numeric(coef.est[8,3:5]))*(priceKR-
uc)+

seg.share[9]*demand_new(priceKR,1.43,as.numeric(coef.est[9,3:5]))*(priceKR-
uc))

return(profitSum)
}

Aux=seq(0,3,0.01)
profitsum = profitSum(Aux,para)
optPrice = Aux[profitsum==max(profitsum)]

#launch KB
demandkb=function(priceKB,priceKR,priceMB,para){

prob=exp(para[1]+para[4]*priceKB)/(1+exp(para[1]+para[4]*priceKB)+exp(para[2]
+para[4]*priceKR)+exp(para[3]+para[4]*priceMB))
  return(prob)
}
demandkr=function(priceKB,priceKR,priceMB,para){

prob=exp(para[2]+para[4]*priceKR)/(1+exp(para[1]+para[4]*priceKB)+exp(para[2]
+para[4]*priceKR)+exp(para[3]+para[4]*priceMB))
  return(prob)
}

#Write profit as a function of prices we set and model parameters
profit=function(priceKB,priceKR,priceMB,para){
  profitKB=1000*sum(

seg.share[1]*demandkb(priceKB,priceKR,priceMB,as.numeric(coef.est[1,2:5]))*(p
riceKB-uc),

seg.share[2]*demandkb(priceKB,priceKR,priceMB,as.numeric(coef.est[2,2:5]))*(p
riceKB-uc),

seg.share[3]*demandkb(priceKB,priceKR,priceMB,as.numeric(coef.est[3,2:5]))*(p
riceKB-uc),

seg.share[4]*demandkb(priceKB,priceKR,priceMB,as.numeric(coef.est[4,2:5]))*(p
riceKB-uc),

seg.share[5]*demandkb(priceKB,priceKR,priceMB,as.numeric(coef.est[5,2:5]))*(p
riceKB-uc),

seg.share[6]*demandkb(priceKB,priceKR,priceMB,as.numeric(coef.est[6,2:5]))*(p
riceKB-uc),

seg.share[7]*demandkb(priceKB,priceKR,priceMB,as.numeric(coef.est[7,2:5]))*(p
riceKB-uc),

seg.share[8]*demandkb(priceKB,priceKR,priceMB,as.numeric(coef.est[8,2:5]))*(p
riceKB-uc),

seg.share[9]*demandkb(priceKB,priceKR,priceMB,as.numeric(coef.est[9,2:5]))*(p
riceKB-uc))

  profitKR=1000*sum(

seg.share[1]*demandkr(priceKB,priceKR,priceMB,as.numeric(coef.est[1,2:5]))*(p
riceKR-uc)+

seg.share[2]*demandkr(priceKB,priceKR,priceMB,as.numeric(coef.est[2,2:5]))*(p
riceKR-uc)+

seg.share[3]*demandkr(priceKB,priceKR,priceMB,as.numeric(coef.est[3,2:5]))*(p
riceKR-uc)+

seg.share[4]*demandkr(priceKB,priceKR,priceMB,as.numeric(coef.est[4,2:5]))*(p
riceKR-uc)+

seg.share[5]*demandkr(priceKB,priceKR,priceMB,as.numeric(coef.est[5,2:5]))*(p
riceKR-uc)+

seg.share[6]*demandkr(priceKB,priceKR,priceMB,as.numeric(coef.est[6,2:5]))*(p
riceKR-uc)+

seg.share[7]*demandkr(priceKB,priceKR,priceMB,as.numeric(coef.est[7,2:5]))*(p
riceKR-uc)+

seg.share[8]*demandkr(priceKB,priceKR,priceMB,as.numeric(coef.est[8,2:5]))*(p
riceKR-uc)+

seg.share[9]*demandkr(priceKB,priceKR,priceMB,as.numeric(coef.est[9,2:5]))*(p
riceKR-uc))

  return(cbind(profitKB,profitKR))
}



pricespace=expand.grid(aux,aux)
profitmatnew=matrix(0L,nrow(pricespace),1)
for (i in 1:nrow(pricespace)){
  profitmatnew[i]=sum(profit(pricespace[i,1],pricespace[i,2],1.43,para))
}

#optimal price
pricespace[which.max(profitmatnew),]
```

```
##      Var1 Var2
## 3835 1.15 1.19
```

```
#maximize profit
max(profitmatnew)
```

```
## [1] 395.3924
```

```
#1.Mango's optimal price
priceKB1 = 1.15
priceKR1 = 1.19

demandmb=function(priceKB,priceKR,priceMB,para){

prob=exp(para[3]+para[4]*priceMB)/(1+exp(para[1]+para[4]*priceKB)+exp(para[2]
+para[4]*priceKR)+exp(para[3]+para[4]*priceMB))
  return(prob)
}

profitMB=function(priceKB,priceKR,priceMB,para){

profitMB=1000*(seg.share[1]*demandmb(priceKB,priceKR,priceMB,as.numeric(coef.
est[1,2:5]))*(priceMB-uc)+

seg.share[2]*demandmb(priceKB,priceKR,priceMB,as.numeric(coef.est[2,2:5]))*(p
riceMB-uc)+

seg.share[3]*demandmb(priceKB,priceKR,priceMB,as.numeric(coef.est[3,2:5]))*(p
riceMB-uc)+

seg.share[4]*demandmb(priceKB,priceKR,priceMB,as.numeric(coef.est[4,2:5]))*(p
riceMB-uc)+

seg.share[5]*demandmb(priceKB,priceKR,priceMB,as.numeric(coef.est[5,2:5]))*(p
riceMB-uc)+

seg.share[6]*demandmb(priceKB,priceKR,priceMB,as.numeric(coef.est[6,2:5]))*(p
riceMB-uc)+

seg.share[7]*demandmb(priceKB,priceKR,priceMB,as.numeric(coef.est[7,2:5]))*(p
riceMB-uc)+

seg.share[8]*demandmb(priceKB,priceKR,priceMB,as.numeric(coef.est[8,2:5]))*(p
riceMB-uc)+

seg.share[9]*demandmb(priceKB,priceKR,priceMB,as.numeric(coef.est[9,2:5]))*(p
riceMB-uc))

return(profitMB)
}
```

```
Aux=seq(0,3,0.01)
profitmb1 = profitMB(priceKB1,priceKR1,Aux,para)
optPriceMB1 = Aux[profitmb1==max(profitmb1)]
optPriceMB1

## [1] 0.96

#2.KB and KR respond to the price of MB1
profitmat1=matrix(0L,nrow(pricespace),1)
for (i in 1:nrow(pricespace)){
  profitmat1[i]=sum(profit(pricespace[i,1],pricespace[i,2],optPriceMB1,para))
}
#optimal price of KB and KR
pricespace[which.max(profitmat1),]   #KB=1.02  KR=1.08

##      Var1 Var2
## 1611 1.02 1.08

#3.MB's respond to price of KB & KR
priceKB2 = 1.02
priceKR2 = 1.08
profitmb2 =profitMB(priceKB2,priceKR2,Aux,para)
optPriceMB2 = Aux[profitmb2==max(profitmb2)]    #optPriceMB_1=0.92
max(profitmb_1)=147.7467
optPriceMB2

## [1] 0.92

#4.KB & KR respond to the new price of MB2
profitmat2=matrix(0L,nrow(pricespace),1)
for (i in 1:nrow(pricespace)){
  profitmat2[i]=sum(profit(pricespace[i,1],pricespace[i,2],optPriceMB2,para))
}
pricespace[which.max(profitmat2),]  # KB=1.01,KR=1.07

##      Var1 Var2
## 1409 1.01 1.07

#5. MB's respond to price of KB & KR
priceKB3 = 1.01
priceKR3 = 1.07
profitmb3 =profitMB(priceKB3,priceKR3,Aux,para)
optPriceMB3 = Aux[profitmb3==max(profitmb3)]
optPriceMB3

## [1] 0.91

#6.KB & KR respond to the new price of MB3
profitmat3=matrix(0L,nrow(pricespace),1)
for (i in 1:nrow(pricespace)){
  profitmat3[i]=sum(profit(pricespace[i,1],pricespace[i,2],optPriceMB3,para))
}
pricespace[which.max(profitmat3),]    #KB=1.00 KR=1.07

##      Var1 Var2
## 1408    1 1.07

#7. MB's respond to price of KB & KR
priceKB4 = 1.00
priceKR4 = 1.07
profitmb4 =profitMB(priceKB4,priceKR4,Aux,para)
optPriceMB4 = Aux[profitmb4==max(profitmb4)]
optPriceMB4     #0.91

## [1] 0.91

#8. KB & KR respond to the new price of MB3
profitmat4=matrix(0L,nrow(pricespace),1)
for (i in 1:nrow(pricespace)){
  profitmat4[i]=sum(profit(pricespace[i,1],pricespace[i,2],optPriceMB4,para))
}
pricespace[which.max(profitmat4),]    #KB=1.00 KR=1.07

##      Var1 Var2
## 1408    1 1.07

#7. MB's respond to price of KB & KR
priceKB5 = 1.00
priceKR5 = 1.07
profitmb5 =profitMB(priceKB5,priceKR5,Aux,para)
optPriceMB5 = Aux[profitmb5==max(profitmb5)]
optPriceMB5     #0.91

## [1] 0.91
```

install.packages("webshot") webshot::install_phantomjs()