B2_Newsvendor2 Python

Jaemin Park

2021-01-04

차 례

Exercises

p.4 Implementation

R code

```
for (X in 11:15){
MC_N <- 10000
D <- sample(11:15, MC_N, replace = T) # random discrete uniform
sales_rev <- 2*pmin(D,X) # vector level minimum
salvage_rev <- 0.5*pmax(X-D,0) # vector level maximum
material_cost <- 1*X
profit <- sales_rev + salvage_rev - material_cost
print(paste0("X: ", X, ", expected profit: ", mean(profit)))
}
```

Python code

```
import numpy as np
from random import *

a_np=np.array([11,12,13,14,15])
b=[]
for X in a_np:
  MC_N = 10000
  rand_list= np.random.rand(MC_N)*4 +11
  D = np.array([X] * MC_N)
  sales_rev = np.minimum(rand_list, D)*2
  salvage_rev = np.maximum(D-rand_list,0)*0.5
  material_cost = D*1
  profit = sales_rev + salvage_rev - material_cost
  mean = np.mean(profit)
  b.append(mean)

b_np = np.array(b)
for i in range(len(b)):
    print('X: %i, expected profit: %f' %(a_np[i], b[i]))
```

```
## X: 11, expected profit: 11.000000
```

```
## X: 12, expected profit: 11.808999
## X: 13, expected profit: 12.264593
## X: 14, expected profit: 12.320600
## X: 15, expected profit: 12.003322
```

p.5 Continuous distribution - grid search approach

R code

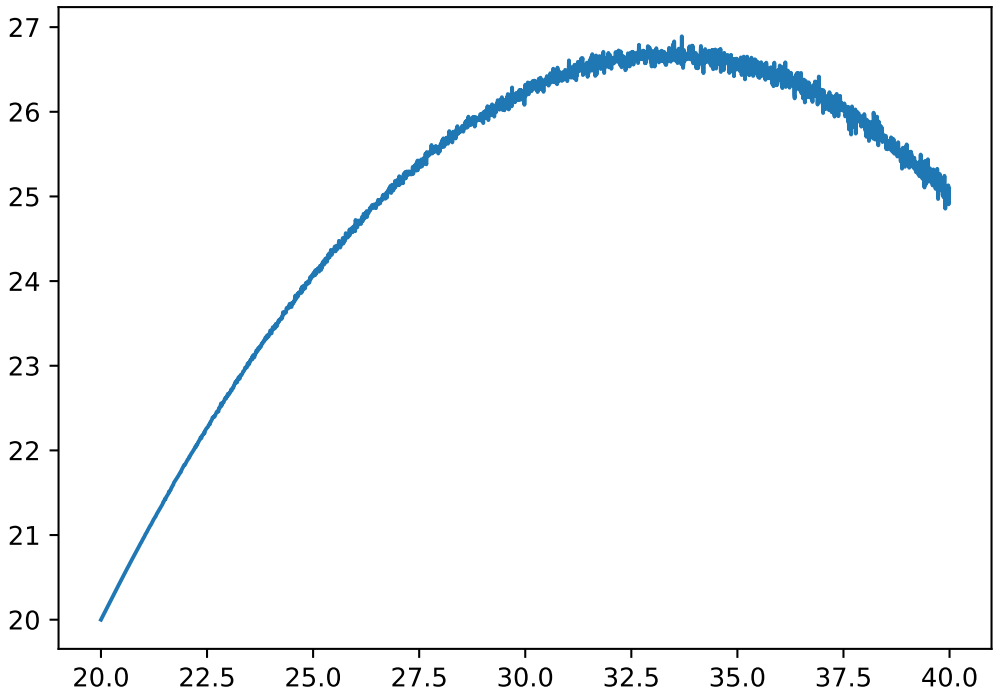
```r
try_X <- seq(from = 20, to = 40, by = 0.01)
exp_profits <- NULL
for (X in try_X){
MC_N <- 10000
D <- runif(MC_N, min = 20, max = 40)
sales_rev <- 2*pmin(D,X) # vector level minimum
salvage_rev <- 0.5*pmax(X-D,0) # vector level maximum
material_cost <- 1*X
exp_profit <- mean(sales_rev + salvage_rev - material_cost)
exp_profits <- c(exp_profits, exp_profit)
}
results <- data.frame(try_X, exp_profits)
```

Python code

```python
import numpy as np
from random import *
import matplotlib.pyplot as plt

try_X = np.arange(20.0,40.0,0.01)
exp_profits = []
for X in try_X:
  MC_N = 10000
  rand_list= np.random.rand(MC_N)*20 +20
  D = np.array([X] * MC_N)
  sales_rev = np.minimum(rand_list, D)*2
  salvage_rev = np.maximum(D-rand_list,0)*0.5
  material_cost = D*1
  profit = sales_rev + salvage_rev - material_cost
  mean = np.mean(profit)
  exp_profits.append(mean)

plt.plot(try_X, exp_profits)
plt.show()
```


27
26
25
24
23
22
21
20
20.0
22.5
25.0
27.5
30.0
32.5
35.0
37.5
40.0

p.8

R code

```
idx <- which(exp_profits==max(exp_profits)) # index for maximum profit
try_X[idx] # this is optimal quantity
```

Python code

```
exp_profits_np = np.array(exp_profits)
idx = np.argmax(exp_profits_np)
print(20+0.01*idx)
```

```
## 33.69
```

R code

```
exp_profits[idx] # this is expected optimal profit
```

Python code

```
print(np.max(exp_profits_np))
```