Lecture B2. Newsvendor 2

Bong Seok Kim

2021-01-03

차 례

Discrete distribution- try out all alternative-Implementation p.3

In the previous newsvendor lecture, following information was given.

- (demand) How many customer? 11, 12, 13, 14, or 15, equally likely
- (retail price) How much do you sell a copy at? $2 per copy
- (material cost) How much do you pay to the wholesaler? $1 per copy
- (salvage value) How much do you sell an unsold copy back to the wholesaler? $0.5per copy

Following code tries the stock level X ∈11,12,13,14,15 in python

```
import numpy as np

np.random.seed(1234)

for X in range(11,16):
    MC_N=10000

    D=np.random.choice(list(range(11,16)),size=MC_N,replace=True) #random discrete uniform

    sales_rev=2*np.minimum(D,X) #vector level minimum

    salvage_rev=0.5*np.maximum(X-D,0) #vector level maximum

    material_cost = 1*X
```

```
    profit=sales_rev+salvage_rev-material_cost

    print("X:",X,", expected profit:", np.mean(profit))
```

```
## X: 11 , expected profit: 11.0
## X: 12 , expected profit: 11.6931
## X: 13 , expected profit: 12.0976
## X: 14 , expected profit: 12.20795
## X: 15 , expected profit: 12.00405
```

Continuous distribution - grid search approach

- Your brother is now selling milk. The customer demands follow $\circledcirc(20, 40)$ gallons. Retail price is \$2 per gallon, material cost is \$1 per gallon, and salvage cost is \$0.5 per gallon. Find optimal stock level and expected profit.
- Notice that there are the marginal difference compared to the previous page's code.

```
import numpy as np
import random
import pandas as pd

np.random.seed(1234)

try_X=np.arange(20, 40, 0.01)
exp_profits=np.array([])


for X in try_X:
    MC_N=10000

    D=np.random.uniform(low=20, high=40, size=MC_N)

    sales_rev=2*np.minimum(D,X) #vector level minimum

    salvage_rev=0.5*np.maximum(X-D,0) #vector level maximum

    material_cost = 1*X

    exp_profit=np.mean(sales_rev+salvage_rev-material_cost)
```

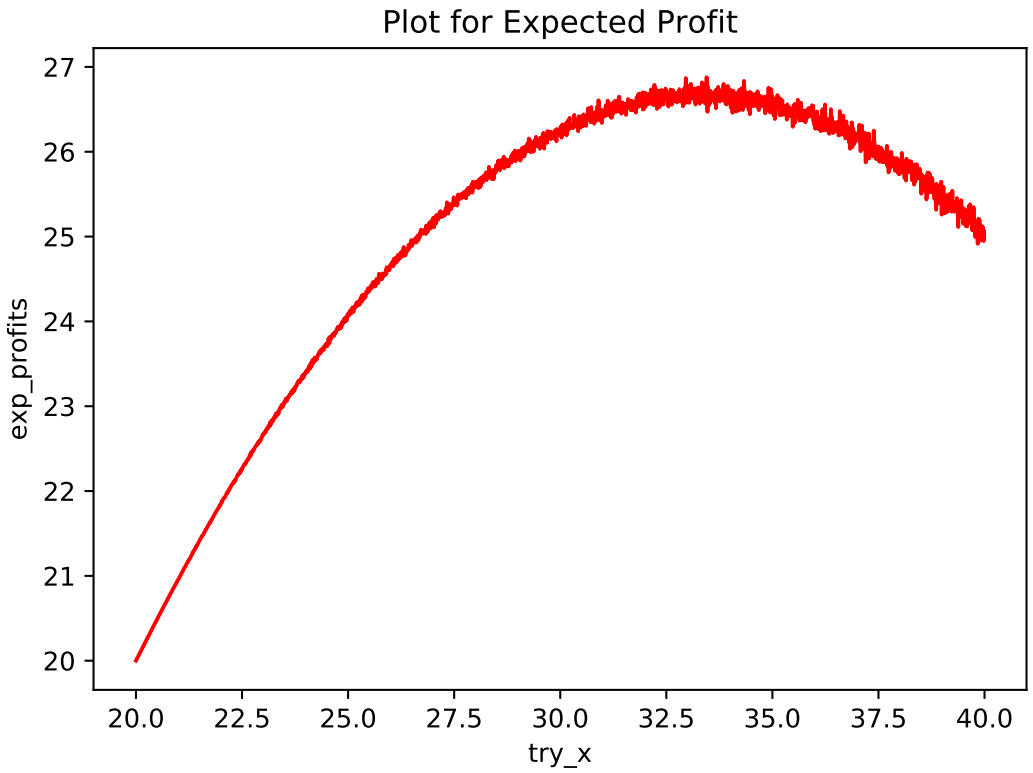
```
    exp_profits=np.append(exp_profits, exp_profit)


result=pd.DataFrame({'try_x':try_X,'exp_profits':exp_profits})
result
```

```
##       try_x  exp_profits
## 0     20.00    20.000000
## 1     20.01    20.009996
## 2     20.02    20.019989
## 3     20.03    20.029953
## 4     20.04    20.039915
## ...     ...          ...
## 1995  39.95    25.121576
## 1996  39.96    25.029005
## 1997  39.97    25.088620
## 1998  39.98    24.947680
## 1999  39.99    25.055350
##
## [2000 rows x 2 columns]
```

```
import matplotlib.pyplot as plt

plt.plot(try_X,exp_profits,color='red')
plt.xlabel("try_x")
plt.ylabel("exp_profits")
plt.title("Plot for Expected Profit")
plt.show()
```




```
idx=np.where(exp_profits==np.max(exp_profits))
print(*try_X[idx]) #this is optimal quantity
```

```
## 33.450000000002106
```

```
print(*exp_profits[idx]) #this is expected optimal profit
```

```
## 26.878008823859624
```

```
"Done, Lecture B2. Newsvendor 2 "
```

```
## [1] "Done, Lecture B2. Newsvendor 2 "
```