DP programming on Price Strategy

Yicheng

16 October, 2019

Contents

Problem sets: **1**

Solution **1**

Price_Stage1 2

Price_Stage2 2

Price_Stage3 3

Inventory_Stage1 3

Inventory_Stage2 4

Inventory_Stage3 5

Problem sets:

My boss invested an ice cream store. I operate the store from 7am to 7pm.

The wholesale price of an ice cream is 1 dollar, the salvage price is 0.5 dollar.

The demand follows a Poisson process with the hourly arrival rate = 10-2*price of the ice cream.

For example, if the price of an ice cream is $1, then the average demand per hour is 8;

if price of an ice cream is $1.5, then the average demand per hour is 7 customers.

The price of the ice cream should not exceed 5$.

The ice cream order needs to be place the night before. Any left-over ice-cream from the day can only be given to the charity (for the salvage value of $0.5).

My boss gives me the freedom to decide on the price – I can change the selling price based on the time of the day and how many ice creams left by this time.
There is only one constraint: one price for morning (7am-12pm), one price for the afternoon (12pm-6pm, etc.) and one price for the evening (6pm-7pm).
So at the three time points, 7am, noon and 6pm, I can make price changes, which will affect your demand rate.

Solution

Set the search range and global parameter

```r
#Search range
start<-2.8
end<-3.2
interval<-0.1

#Global parameter
ps=0.5
qn<-c(2:8)
qa<-c(25:40)
qm<-c(45:65)
```

Dynamic programming, from backwards searching for the optimal order quantity under each price.

Price_Stage1

Get strategy for night

```r
#return price and order qnuantity for night
q1<-myfun1(start,end,interval)
q1
```

```
##   price order_qnuantity  profit
## 1   2.8               6 6.42517
## 2   2.9               6 6.60168
## 3   3.0               6 6.55600
## 4   3.1               5 6.46948
## 5   3.2               5 6.45536
```

```r
#Price for the night
pn<-q1[which.max(q1$profit),1]
```

Using the price we get from previous function to apply to afternoon situation

Price_Stage2

Get strategy for afternoon

```r
#This is the result
q2<-myfun2(start,end,interval)
q2
```

```
##   price order_quantity   profit
## 1   2.8             35 51.89014
## 2   2.9             34 52.00888
## 3   3.0             32 52.22459
## 4   3.1             31 52.18144
## 5   3.2             30 51.74337
```

```r
#Price for the afternoon
pa<-q2[which.max(q2$profit),1]
```

Using the price we get from previous function to apply to morning situation

Price_Stage3

Get strategy for morning

```r
#This is the result
q3<-myfun3(start,end,interval)
q3
```

```
##   price order_quantity   profit
## 1   2.8             56 90.66288
## 2   2.9             55 90.70207
## 3   3.0             54 91.17273
## 4   3.1             53 91.18466
## 5   3.2             52 90.71588
```

```r
#Price for the morning
pm<-q3[which.max(q3$profit),1]
```

Get a price strategy output without considering inventory

```r
#optimal
output<-data.frame(t(q1[which.max(q1$profit),]),t(q2[which.max(q2$profit),]),t(q3[which.max(q3$profit),]
colnames(output)<-c("night","afternoon","morning")
output<-output[,order(-output[3,])]
output
```

```
##                   morning afternoon   night
## price             3.10000   3.00000 2.90000
## order_qnuantity  53.00000  32.00000 6.00000
## profit           91.18466  52.22459 6.60168
```

```r
#search range output
fancy<-data.frame(q1,q2$order_quantity,q2$profit,q3$order_quantity,q3$profit)
colnames(fancy)<-c("price","n_q","n_p","a_q","a_p","m_q","m_p")
```

If we consider price strategy with inventory, we can also use DP to solve the problem

From backwards searching for the optimal price under each inventory level.

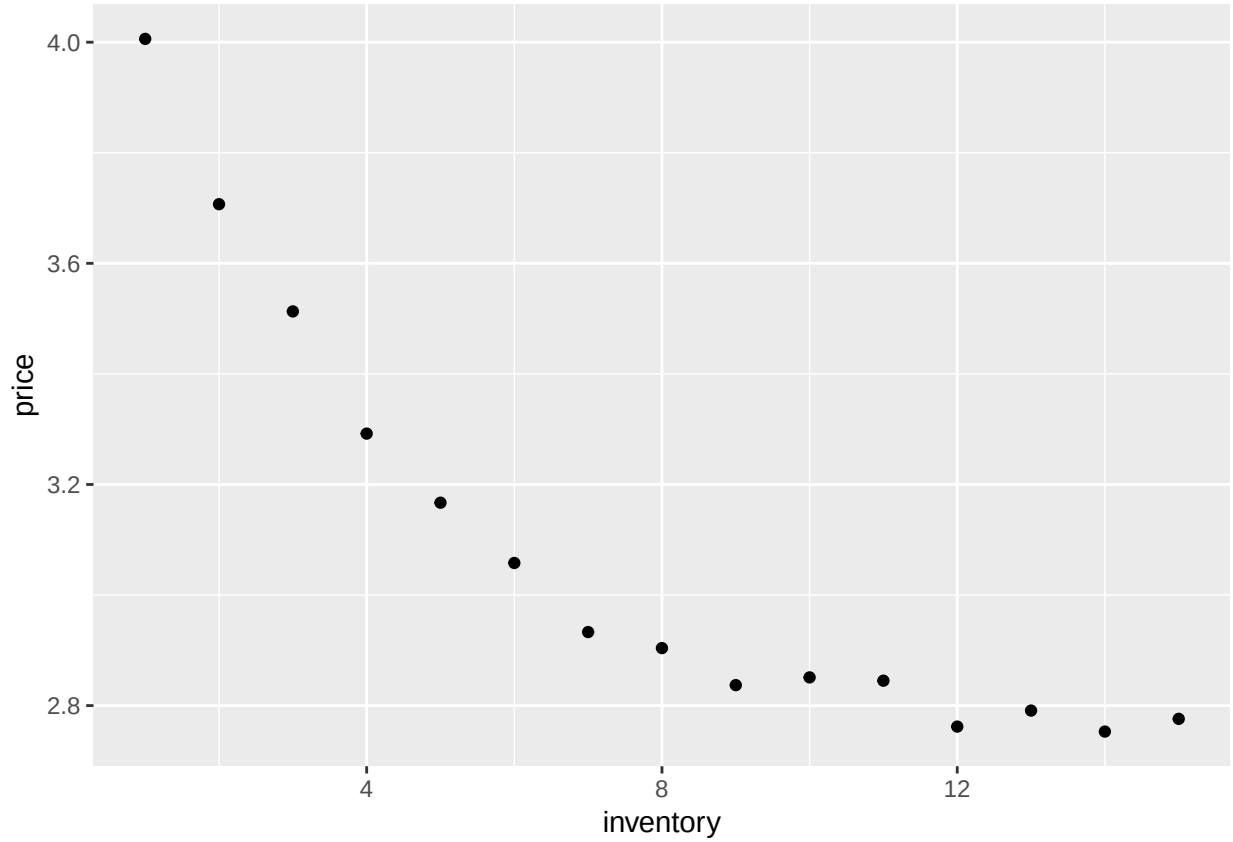
Inventory_Stage1

(take a while based on computer performance)

Once at the stage of night, we would like to adjust price according to inventory left.

For example, if there is only one left, we would like to set a higher price above $4.

```
library(ggplot2)
ggplot(inventory1,aes(x=inventory,y=price))+geom_point()
```



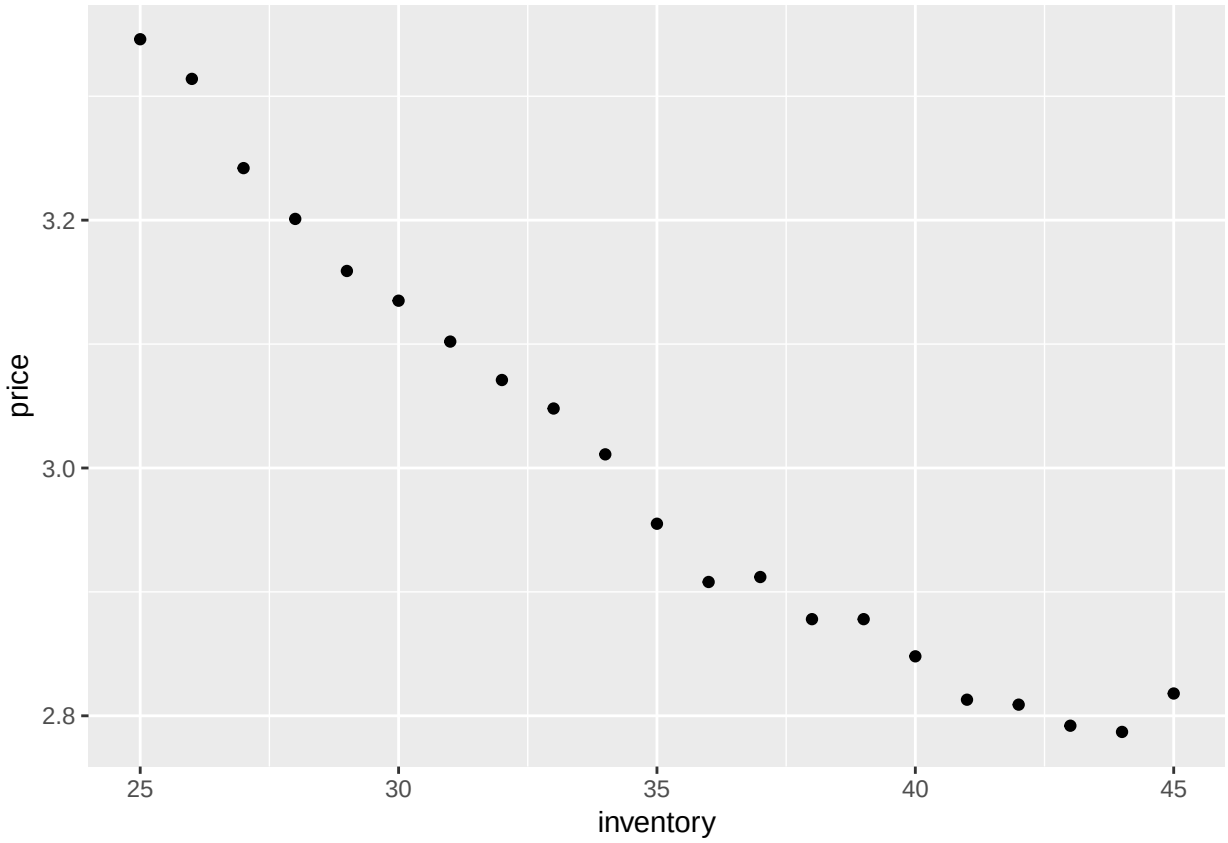
Inventory_Stage2

(take a while based on computer performance)

Once at the stage of afternoon, we would like to adjust price according to inventory left.

For example, if there is 30 icecream left, we would like to set a price approx at $3.1.

```
ggplot(inventory2,aes(x=inventory,y=price))+geom_point()
```



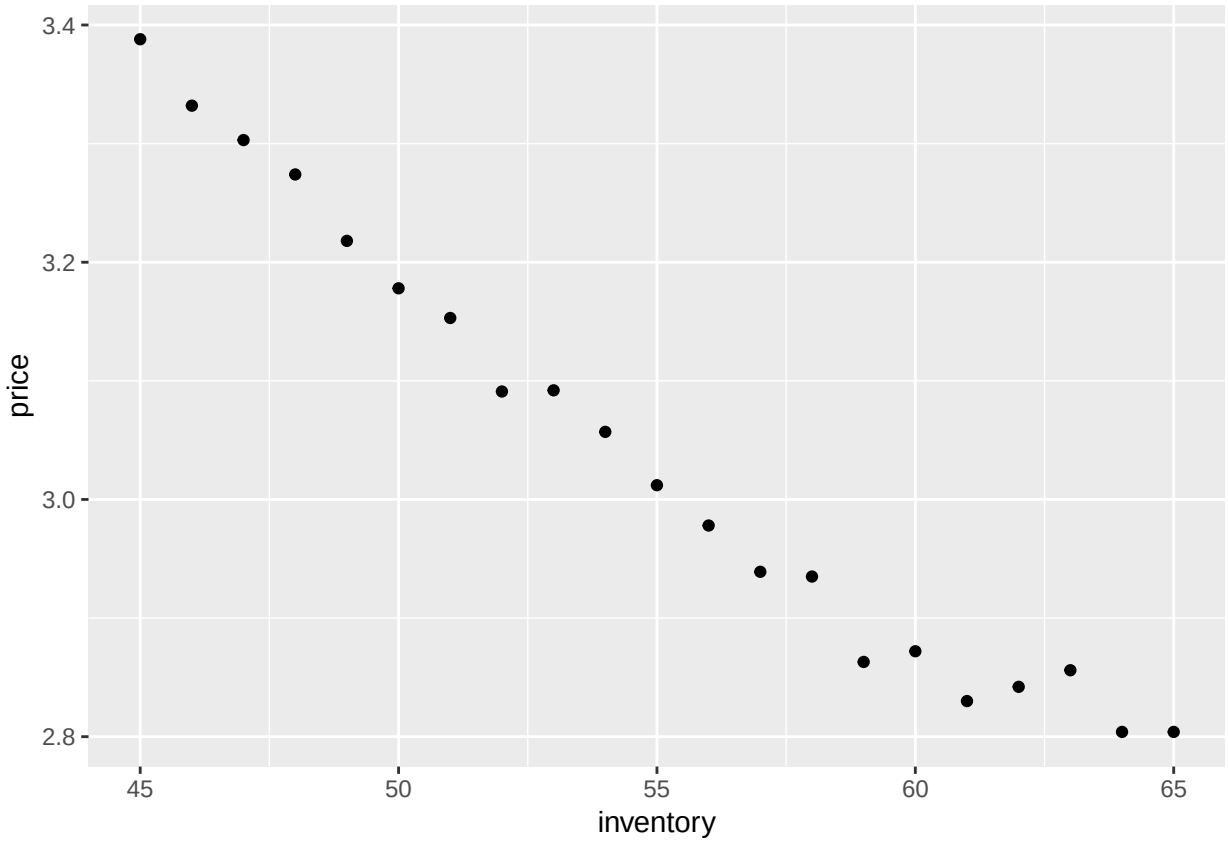
Inventory_Stage3

(take a while based on computer performance)

Once at the stage of morning , we would like to adjust price according to inventory left.

But since we would have a fixed order quantity, and inventory in the morning is the same as order quantity, we will just use the optimal order quantity and the price in accordance

```
ggplot(inventory3,aes(x=inventory,y=price))+geom_point()
```



And that's pretty much it

For further work, we can do more general work (real-time change) to make the price strategy to be more smart

and simulate a more 'real' market environment

Figure 1: My school logo