

## The Newspaper Seller Problem

This is a classical inventory problem that concerns the purchase and sale of newspapers. Here the assumptions are:

- The paper seller buys the papers for 33 cents each and sells them for 50 cents each.
- The lost profit from excess demand is 17 cents for each paper demanded that could not be provided.
- Newspapers not sold at the end of the day are sold as scrap for 5 cents each. (the salvage value of scrap papers).
- Newspapers can be purchased in bundles of 10. Thus, the paper seller can buy 50, 60, and so on. There are three types of newsdays, “good,” “fair,” and “poor,” with probabilities different probabilities of 0.35, 0.45, and 0.20 respectively.
- The problem is to determine the optimal number of papers the newspaper seller should purchase. /Prepare Simulation table for purchase of 70 Newspapers.
- This will be accomplished by simulating demands and recording profits from sales each day.
- The distribution of papers demanded on each of these days is given table 2.15
- Table 2.16 and 2.17 provide the random digit assignments for the types of News days and the demands for those News days.

### Distribution of newspaper demanded,

Demand	Good	Fair	Poor
40	0.03	0.10	0.44
50	0.05	0.18	0.22
60	0.15	0.40	0.16
70	0.20	0.20	0.12
80	0.35	0.08	0.06
90	0.15	0.04	0.00
100	0.07	0.00	0.00

Table 01: Newspaper distribution by demand

**Random digit assignment for types of Newsday,**

<b>Types of Newsday</b>	<b>Probability</b>	<b>Cumulative probability</b>	<b>Random digit assignment</b>
Good	0.35	0.35	1-35
Fair	0.45	0.80	36-80
Poor	0.20	1.00	81-100

Table 02: Random digit for news days.

**Random digit assignment for newspaper demanded,**

<b>Demand</b>	<b>Good</b>	<b>Fair</b>	<b>Poor</b>	<b>Good</b>	<b>Fair</b>	<b>Poor</b>
40	0.03	0.10	0.44	01-03	01-10	02-44
50	0.05	0.18	0.22	04-08	11-28	45-66
60	0.15	0.40	0.16	09-23	29-68	67-82
70	0.20	0.20	0.12	24-43	69-88	83-94
80	0.35	0.08	0.06	44-78	89-96	95-100
90	0.15	0.04	0.00	79-93	97-100	
100	0.07	0.00	0.00	94-100		

Table 03: Random digit for demand.

## Solution:

### Manual simulation:

### Calculation:

*Profit = [(revenue from sales) - (cost of newspapers) - (lost profit from excess demand) + (salvage from sale of scrap papers)]*

*Revenue = Demand X Selling price*

*Lost Profit = (Demand - Stock) X 17 cents [Demand > Stock]*

*Salvage = (Stock - Demand) X 5 cents [Demand < Stock]*

*Cost of Newspaper = Stock X Buying price*

*Cost of newspaper = 70 \* (33/100) = 23.1*

### Manual simulation table:

Day	RN for TON	TON	RN for Demand	Demand	revenue	lost profit	Salvage	Daily profit	pv of 70
1	56	fair	63	60	30	0	0.5	7.4	23.1
2	50	fair	87	70	35	0	0	11.9	23.1
3	87	poor	53	50	25	0	1	2.9	23.1
4	53	fair	73	70	35	0	0	11.9	23.1
5	11	good	62	80	40	1.7	0	15.2	23.1
6	32	good	41	70	35	0	0	11.9	23.1
7	21	good	2	40	20	0	1.5	-1.6	23.1
8	54	fair	83	70	35	0	0	11.9	23.1
9	24	good	36	70	35	0	0	11.9	23.1
10	77	fair	43	60	30	0	0.5	7.4	23.1
11	65	fair	9	40	20	0	1.5	-1.6	23.1
12	3	good	100	100	50	5.1	0	21.8	23.1
13	24	good	81	90	45	3.4	0	18.5	23.1
14	54	fair	21	50	25	0	1	2.9	23.1
15	11	good	1	40	20	0	1.5	-1.6	23.1
16	84	poor	28	40	20	0	1.5	-1.6	23.1
17	42	fair	14	50	25	0	1	2.9	23.1

18	44	fair	49	60	30	0	0.5	7.4	23.1
19	41	fair	89	80	40	1.7	0	15.2	23.1
20	33	good	86	90	45	3.4	0	18.5	23.1
21	6	good	99	100	50	5.1	0	21.8	23.1
22	90	poor	31	40	20	0	1.5	-1.6	23.1
23	52	fair	21	50	25	0	1	2.9	23.1
24	78	fair	94	80	40	1.7	0	15.2	23.1
25	81	poor	83	70	35	0	0	11.9	23.1
26	29	good	44	80	40	1.7	0	15.2	23.1
27	26	good	37	70	35	0	0	11.9	23.1
28	19	good	65	80	40	1.7	0	15.2	23.1
29	16	good	68	80	40	1.7	0	15.2	23.1
30	77	fair	86	70	35	0	0	11.9	23.1
					1000	27.2	13		693

Table 4: Solution with equations.

## Simulation in excel:

Day	RN for TON	TON	RN for Demand	Demand	revenue	lost profit	Salvage	Daily profit	pv of 70
1	56	fair	63	60	30	0	0.5	7.4	23.1
2	50	fair	87	70	35	0	0	11.9	23.1
3	87	poor	53	50	25	0	1	2.9	23.1
4	53	fair	73	70	35	0	0	11.9	23.1
5	11	good	62	80	40	1.7	0	15.2	23.1
6	32	good	41	70	35	0	0	11.9	23.1
7	21	good	2	40	20	0	1.5	-1.6	23.1
8	54	fair	83	70	35	0	0	11.9	23.1
9	24	good	36	70	35	0	0	11.9	23.1
10	77	fair	43	60	30	0	0.5	7.4	23.1
11	65	fair	9	40	20	0	1.5	-1.6	23.1
12	3	good	100	100	50	5.1	0	21.8	23.1
13	24	good	81	90	45	3.4	0	18.5	23.1
14	54	fair	21	50	25	0	1	2.9	23.1
15	11	good	1	40	20	0	1.5	-1.6	23.1
16	84	poor	28	40	20	0	1.5	-1.6	23.1
17	42	fair	14	50	25	0	1	2.9	23.1
18	44	fair	49	60	30	0	0.5	7.4	23.1
19	41	fair	89	80	40	1.7	0	15.2	23.1
20	33	good	86	90	45	3.4	0	18.5	23.1
21	6	good	99	100	50	5.1	0	21.8	23.1
22	90	poor	31	40	20	0	1.5	-1.6	23.1
23	52	fair	21	50	25	0	1	2.9	23.1
24	78	fair	94	80	40	1.7	0	15.2	23.1
25	81	poor	83	70	35	0	0	11.9	23.1
26	29	good	44	80	40	1.7	0	15.2	23.1
27	26	good	37	70	35	0	0	11.9	23.1
28	19	good	65	80	40	1.7	0	15.2	23.1
29	16	good	68	80	40	1.7	0	15.2	23.1
30	77	fair	86	70	35	0	0	11.9	23.1
					1000	27.2	13		693

Figure 01: Newspaper problem solution in excel.

# Simulation using Python:

## Source Code:

```
from random import randrange
day_by_day = []
all_day = []
rn_ton = 0
ton = 0
rn_demand = 0
demand = 0
revenue = 0
lp = 0
slvg = 0
purchase_value_of_70_np = round((.33 * 70), 2)
dp = 0
```

```
def rn_for_ton(k): #2 function
    for i in range(k):
        global rn_ton, day_by_day
        day_by_day = [] # reset the list to empty
        rn_ton = randrange(100)
        day_by_day.append(i + 1)
        day_by_day.append(rn_ton)
        type_of_news_day() #3 call
```

```
def type_of_news_day(): #3 function
    global ton
    if rn_ton < 36:
        ton = "good"
    elif rn_ton < 81:
        ton = "fair"
    elif rn_ton < 101:
        ton = "poor"
    day_by_day.append(ton)
```

```
rn_for_demand() #4 call
```

```
def rn_for_demand(): #4 function
    global rn_demand
    rn_demand = randrange(100)
    day_by_day.append(rn_demand)
    _demand() #5 call
```

```
def _demand(): #5 function
    global demand
    if ton == "poor" and rn_demand < 45:
        demand = 40
    elif ton == "poor" and rn_demand < 67:
        demand = 50
    elif ton == "poor" and rn_demand < 83:
        demand = 60
```

```

elif ton == "poor" and rn_demand < 95:
    demand = 70
elif ton == "poor" and rn_demand < 101:
    demand = 80
elif ton == "fair" and rn_demand < 11:
    demand = 40
elif ton == "fair" and rn_demand < 29:
    demand = 50
elif ton == "fair" and rn_demand < 69:
    demand = 60
elif ton == "fair" and rn_demand < 89:
    demand = 70
elif ton == "fair" and rn_demand < 97:
    demand = 80
elif ton == "fair" and rn_demand < 101:
    demand = 90
elif ton == "good" and rn_demand < 4:
    demand = 40
elif ton == "good" and rn_demand < 9:
    demand = 50
elif ton == "good" and rn_demand < 24:
    demand = 60
elif ton == "good" and rn_demand < 44:
    demand = 70
elif ton == "good" and rn_demand < 79:
    demand = 80
elif ton == "good" and rn_demand < 94:
    demand = 90
elif ton == "good" and rn_demand < 101:
    demand = 100
else:
    demand = 0

day_by_day.append(demand)
_revenue() #6 call

def _revenue(): #6 function
    global revenue
    revenue = demand * .5
    day_by_day.append(revenue)
    profit_lost() #7 call

def profit_lost(): #7 function
    global lp
    if demand > 70:
        lp = (demand - 70) * .17
    else:
        lp = 0
    day_by_day.append(round(lp, 2))
    salvage() #8 call

```

```

def salvage(): #8 function
    global slvg
    if demand < 70:
        slvg = (70 - demand) * .05
    else:
        slvg = 0
    day_by_day.append(slvg)
    daily_profit() #9 call

def daily_profit(): #9 function
    global dp
    dp = revenue - purchase_value_of_70_np - lp + slvg
    day_by_day.append(round(dp,2))
    day_by_day.append(purchase_value_of_70_np)
    all_day.append(day_by_day)
    #end of one loop initiated in function 2

rn_for_ton(30) #1 #2 call

import pandas as pd
df = pd.DataFrame(all_day, columns
= ['Day', 'RN for TON', 'TON', 'RN for Demand', 'Demand', 'Revenue', 'Lost Profit', 'Salvage

df = df.set_index('Day')
pd.set_option('display.colheader_justify', 'center')

print(df)

```

## Output:

Day	RN for TON	TON	RN for Demand	Demand	Revenue	Lost Profit	Salvage	Daily Profit	Purchase Value of 70
1	2	good	69	80	40.0	1.7	0.0	15.2	23.1
2	15	good	43	70	35.0	0.0	0.0	11.9	23.1
3	29	good	21	60	30.0	0.0	0.5	7.4	23.1
4	25	good	49	80	40.0	1.7	0.0	15.2	23.1
5	87	poor	63	50	25.0	0.0	1.0	2.9	23.1
6	18	good	72	80	40.0	1.7	0.0	15.2	23.1
7	24	good	61	80	40.0	1.7	0.0	15.2	23.1
8	34	good	41	70	35.0	0.0	0.0	11.9	23.1
9	6	good	3	40	20.0	0.0	1.5	-1.6	23.1
10	23	good	68	80	40.0	1.7	0.0	15.2	23.1
11	56	fair	32	60	30.0	0.0	0.5	7.4	23.1
12	2	good	67	80	40.0	1.7	0.0	15.2	23.1
13	99	poor	31	40	20.0	0.0	1.5	-1.6	23.1
14	17	good	80	90	45.0	3.4	0.0	18.5	23.1
15	84	poor	16	40	20.0	0.0	1.5	-1.6	23.1
16	9	good	78	80	40.0	1.7	0.0	15.2	23.1
17	70	fair	59	60	30.0	0.0	0.5	7.4	23.1
18	5	good	31	70	35.0	0.0	0.0	11.9	23.1
19	88	poor	11	40	20.0	0.0	1.5	-1.6	23.1
20	60	fair	67	60	30.0	0.0	0.5	7.4	23.1
21	7	good	2	40	20.0	0.0	1.5	-1.6	23.1
22	74	fair	23	50	25.0	0.0	1.0	2.9	23.1
23	35	good	65	80	40.0	1.7	0.0	15.2	23.1
24	35	good	57	80	40.0	1.7	0.0	15.2	23.1
25	20	good	64	80	40.0	1.7	0.0	15.2	23.1
26	18	good	95	100	50.0	5.1	0.0	21.8	23.1
27	83	poor	79	60	30.0	0.0	0.5	7.4	23.1
28	83	poor	29	40	20.0	0.0	1.5	-1.6	23.1
29	11	good	59	80	40.0	1.7	0.0	15.2	23.1
30	20	good	79	90	45.0	3.4	0.0	18.5	23.1

(dip) PS E:\Study\Fall 2021\Simulation and modeling sessional\Final> █

Figure 02: Simulation in Python.