Modern Academy
Since 1993
maadi
مودرن اكاديمى بالمعادى

FINAL LAB REPORT

2025

NEWSPAPER SELLER CASE STUDY

Presented by:

AHMED MOHAMED A. ABBAS

ROKIYA ABDELSATTAR

AHMED IBRAHIM

HASNAA NAGEH

Presented for:

DR: NOHA ALI

Newspaper Seller Case Study

Problem Definition:

The newspaper seller wants to determine how many newspapers he should buy each day in order to **maximize his profits**. His present method of determining the quantity of Newspaper is based upon his best guess or estimate of the daily demand for the day news.

Assumptions and Parameters:

- **Selling Price (P):** $0.50 per newspaper
- **Cost Price (C):** $0.33 per newspaper
- **Scrap Value (S):** $0.05 per unsold newspaper
- **X:** Quantity of newspapers purchased
- **d:** Daily demand
- **Z:** Daily profit

Day Type Probability Distribution:

Type of Newsday	Probability	Cumulative	Random Number Range
Good	0.35	0.35	$0 \leq R < 0.35$
Fair	0.45	0.80	$0.35 \leq R < 0.80$
Poor	0.20	1.00	$0.80 \leq R \leq 1$

Demand Distribution:

(For demand values: 40, 50, 60, 70, 80, 90, 100 newspapers)

You will need to calculate the Cumulative Distribution Function (CDF) for demand for each type of day (Good, Fair, Poor) using your Excel data. You can present each distribution in a table format like:

	Demand Prob. Dist.		
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

Model Formulation:

Key Equations:

1) Revenue Sales = $\min(d, X) \times P$
2) Excess Demand = $\max(X, d) - d$
3) Profit per newspaper sold = $P - C$
4) Lost Profit = $(X < d)\ ?\ (d - X) \times (P - C)$
5) Number of Scrap = $\max(X - d, 0)$
6) Scrap Salvage = Number of Scrap × S
7) Cost of Newspapers = $X \times C$
8) Daily Profit (Z) = Revenue Sales − Cost of Newspapers − Lost Profit + Scrap Salvage

Objective

Use simulation (e.g., random number generation for day type and demand) to estimate the optimal number of newspapers to purchase daily (X) that maximizes the average profit $Z\ (X, d)$.

Implementation

Excel Sheet:

Assume	
Item	Value
P : Paper Sell	50
C : Paper Cost	33
S : Scrap Sale	5
D: Days	40
x: Quantity	70
d Daily Demand	
z: Daily Profit	

Type of Newsday	Probability	Cumlative Probability	Random-Digit Assignment	
			Lower	*Upper*
Good	0.35	0.35	0.00	0.34
Fair	0.45	0.80	0.35	0.79
Poor	0.20	1.00	0.80	1.00

	Demand Prob. Dist.		
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

	Prob dist.			Cumulative			Random-Digit Assignment					
Demand	Good	Fair	Poor	Good	Fair	Poor	Lower	Upper	Lower	Upper	Lower	Upper
40	0.03	0.10	0.44	0.03	0.10	0.44	0.00	0.02	0.00	0.09	0.00	0.43
50	0.05	0.18	0.22	0.08	0.28	0.66	0.03	0.07	0.10	0.27	0.44	0.65
60	0.15	0.40	0.16	0.23	0.68	0.82	0.08	0.22	0.28	0.67	0.66	0.81
70	0.20	0.20	0.12	0.43	0.88	0.94	0.23	0.42	0.68	0.87	0.82	0.93
80	0.35	0.08	0.06	0.78	0.96	1.00	0.43	0.77	0.88	0.95	0.94	1.00
90	0.15	0.04	0.00	0.93	1.00		0.78	0.92	0.96	1.00		
100	0.07	0.00	0.00	1.00			0.93	1.00				

Day	R.D Type	Type	R.D Demand	Demand	Revnue Sales	Excess Demand	lost Profit E-D	Num of Scrap	Salvage from scrap	Daily Profit
1	9	Good	13	60	3000	0	0	10	50	740
2	2	Good	26	70	3500	0	0	0	0	1190
3	32	Good	75	80	3500	10	170	0	0	1020
4	52	Fair	76	70	3500	0	0	0	0	1190
5	10	Good	92	90	3500	20	340	0	0	850
6	46	Fair	71	70	3500	0	0	0	0	1190
7	29	Good	3	50	2500	0	0	20	100	290
8	12	Good	39	70	3500	0	0	0	0	1190
9	75	Fair	21	50	2500	0	0	20	100	290
10	8	Good	82	90	3500	20	340	0	0	850
11	91	Poor	40	40	2000	0	0	30	150	-160
12	7	Good	37	70	3500	0	0	0	0	1190
13	82	Poor	60	50	2500	0	0	20	100	290
14	45	Fair	55	60	3000	0	0	10	50	740
15	65	Fair	32	60	3000	0	0	10	50	740
16	74	Fair	81	70	3500	0	0	0	0	1190
17	78	Fair	82	70	3500	0	0	0	0	1190
18	11	Good	51	80	3500	10	170	0	0	1020
19	25	Good	57	80	3500	10	170	0	0	1020
20	57	Fair	69	70	3500	0	0	0	0	1190
21	36	Fair	90	80	3500	10	170	0	0	1020
.....	.....	.....	.....	.....	.....	.....	.....	.....	.....	.....

C++ Code:

```cpp
#include<bits/stdc++.h>
using namespace std;

constexpr int days          = 30;
constexpr int quantity      = 70;
constexpr int paper_cost    = 33;
constexpr int iterations    = 10;
constexpr float scrap_sale  =  5;
constexpr float paper_sell  = 50;

/// Types of News day
constexpr float GOOD        = 0.35;
constexpr float FAIR        = 0.45;
constexpr float POOR        = 0.20;

float cumulative_prop[3] = {
    GOOD,
    GOOD + FAIR,
    GOOD + FAIR + POOR,
};

struct RD {
    float lower, upper;
} RandDist[3]{
    {0.0, GOOD - 0.01},
    {GOOD, GOOD + FAIR - 0.01},
    {GOOD + FAIR, GOOD + FAIR + POOR}
};

struct DayData {
```

```
    int day;
    double R1;
    string type;
    double R2;
    int demand;
    int revenue_sales;
    int excess_demand;
    int lost_profit_excess;
    int num_scrap;
    int salvage_scrap;
    int daily_profit;
};

int getDemand(const string& dayType, const double R) {
    if (dayType == "Good") {
        if (R < 0.03) return 40;
        if (R < 0.08) return 50;
        if (R < 0.23) return 60;
        if (R < 0.43) return 70;
        if (R < 0.78) return 80;
        if (R < 0.93) return 90;
        return 100;
    }

    if (dayType == "Fair") {
        if (R < 0.10) return 40;
        if (R < 0.28) return 50;
        if (R < 0.68) return 60;
        if (R < 0.88) return 70;
        if (R < 0.96) return 80;
        return 90;
    }

    if (dayType == "Poor") {
        if (R < 0.44) return 40;
        if (R < 0.66) return 50;
        if (R < 0.82) return 60;
        if (R < 0.94) return 70;
        return 80;
    }
    return {};
}

vector<DayData> simulate(int X, int N) {
    vector<DayData> results;
    random_device rd;
    mt19937 gen(rd());
    uniform_real_distribution<> dis(0.0, 1.0);

    for (int day = 1; day <= N; ++day) {
        double R1 = dis(gen);
        string type;
        if (R1 < cumulative_prop[0]) type = "Good";
        else if (R1 < cumulative_prop[1]) type = "Fair";
        else type = "Poor";

        double R2 = dis(gen);
        int demand = getDemand(type, R2);

        int revenue_sales = min(X, demand) * paper_sell;
        int excess_demand = max(demand - X, 0);
        int lost_profit_excess = excess_demand * (paper_sell - paper_cost);
        int num_scrap = max(X - demand, 0);
        int salvage_scrap = num_scrap * scrap_sale;
        int daily_profit = revenue_sales + salvage_scrap - X * paper_cost;

        DayData data = {day, R1, type, R2, demand, revenue_sales,
          excess_demand, lost_profit_excess, num_scrap, salvage_scrap, daily_profit};
        results.push_back(data);
    }
    return results;
}

void writeTableToFile(const vector< DayData > &results, const string filename) {
    ofstream outfile(filename);
    if (!outfile) return void (cerr << "NOOOOOOOOOOOOOOO such File: " << filename << endl);
```

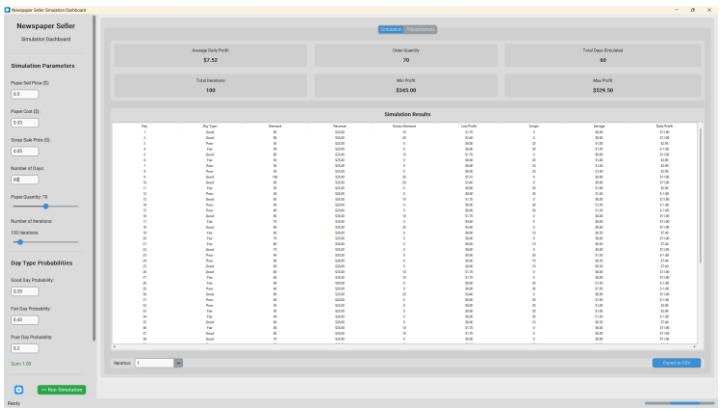
```
    outfile <<
"Day\t|\tR1\t|\tType\t|\tR2\t|\tDemand\t|\tRevenue\t|\tExcessD\t|\tLostProfit\t|\tScrap\t|\tSalvage\t
|\tProfit\n";
    for (const auto &data: results) {
        outfile << data.day << "\t| \t"
                << fixed << setprecision(2) << data.R1 << "\t| \t"
                << data.type << "\t| \t"
                << fixed << setprecision(2) << data.R2 << "\t| \t"
                << data.demand << "\t| \t"
                << data.revenue_sales << "\t| \t"
                << data.excess_demand << "\t| \t"
                << data.lost_profit_excess << "\t| \t"
                << data.num_scrap << "\t| \t"
                << data.salvage_scrap << "\t| \t"
                << data.daily_profit << "\n";
    }
    outfile.close();
}

int main() {
    int N_large = 1000;
    vector< int > X_values = {40, 50, 60, 70, 80, 90, 100};
    double best_avg_profit = -numeric_limits< double >::max();
    int best_X = -1;

    cout << "Let's Simulate...\n";
    for (int X: X_values) {
        auto results = simulate(X, N_large);
        double total_profit = 0;

        for (const auto &data: results) total_profit += data.daily_profit;

        double avg_profit = total_profit / N_large;
        cout << "Quantity X = " << X << ", Average Profit = " << fixed << setprecision(2)
           << avg_profit << " cents\n";
        if (avg_profit > best_avg_profit) best_avg_profit = avg_profit, best_X = X;
    }

    cout << "\nOptimal quantity to buy: " << best_X << " newspapers\n";
    cout << "Expected avg profit: " << fixed << setprecision(2) << best_avg_profit << " cents per
day\n";

    auto table_results = simulate(best_X, days);
    writeTableToFile(table_results, "newspaper_simulation.txt");

    double table_total_profit = 0;
    for (const auto &data: table_results) {
        table_total_profit += data.daily_profit;
    }
    double table_avg_profit = table_total_profit / days;
    cout << "\nAverage profit over " << days << " days with X = " << best_X << ": "
         << fixed << setprecision(2) <<
            table_avg_profit << " cents\n";

    return 0;
}
```

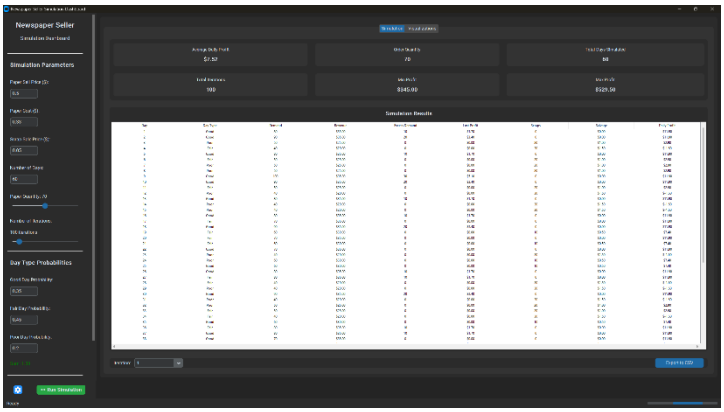
Run

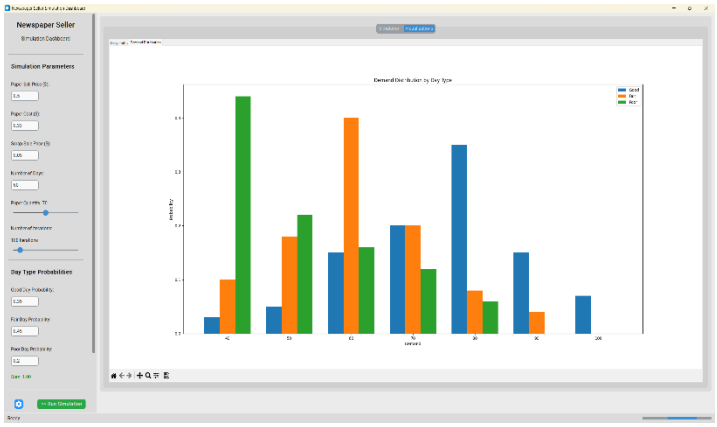
```
Let's Simulate...
Quantity X = 40, Average Profit = 680.00 cents
Quantity X = 50, Average Profit = 790.60 cents
Quantity X = 60, Average Profit = 831.00 cents
Quantity X = 70, Average Profit = 716.60 cents
Quantity X = 80, Average Profit = 557.20 cents
Quantity X = 90, Average Profit = 328.05 cents
Quantity X = 100, Average Profit = 75.95 cents

Optimal quantity to buy: 60 newspapers
Expected avg profit: 831.00 cents per day

Average profit over 30 days with X = 60: 855.00 cents
```

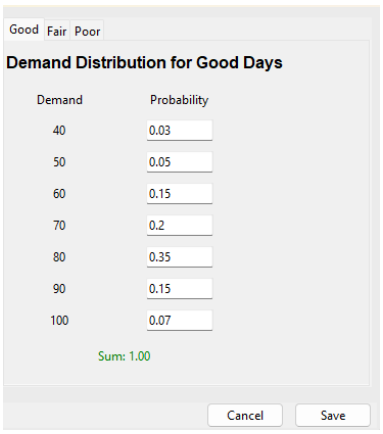
Day	R1	Type	R2	Demand	Revenue	ExcessD	LostProfit	Scrap	Salvage	Profit
1	0.08	Good	0.92	90	3000	30	510	0	0	1020
2	0.72	Fair	0.56	60	3000	0	0	0	0	1020
3	0.65	Fair	0.55	60	3000	0	0	0	0	1020
4	0.35	Fair	0.57	60	3000	0	0	0	0	1020
5	0.59	Fair	0.98	90	3000	30	510	0	0	1020
6	0.00	Good	0.68	80	3000	20	340	0	0	1020
7	0.76	Fair	0.11	50	2500	0	0	10	50	570
8	0.60	Fair	0.28	50	2500	0	0	10	50	570
9	0.01	Good	0.15	60	3000	0	0	0	0	1020
10	0.36	Fair	0.58	60	3000	0	0	0	0	1020
11	0.13	Good	0.84	90	3000	30	510	0	0	1020
12	0.62	Fair	0.68	70	3000	10	170	0	0	1020
13	0.04	Good	0.45	80	3000	20	340	0	0	1020

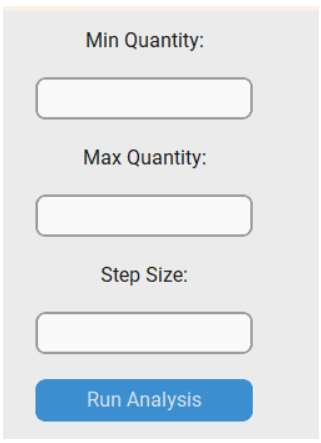
GUI Presentation

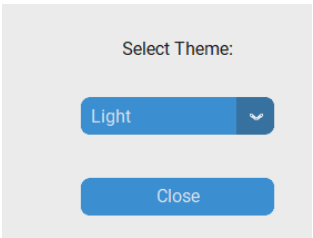






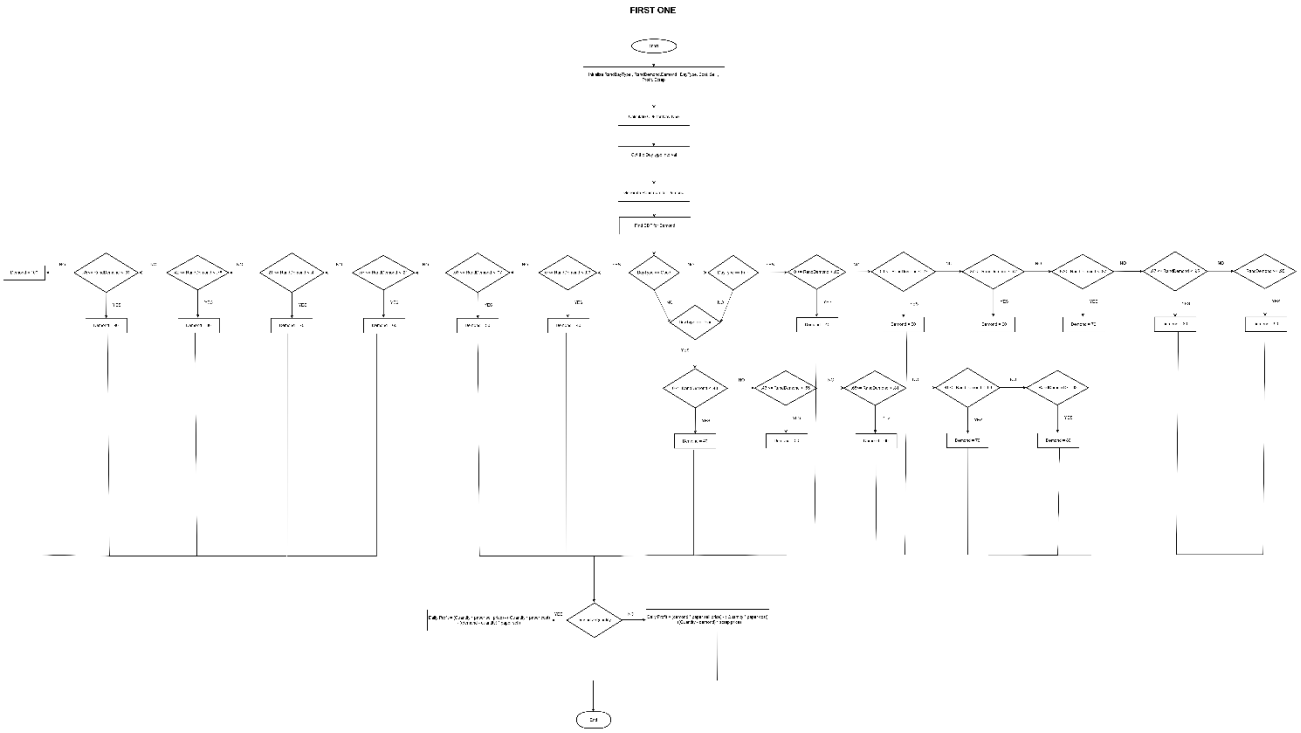




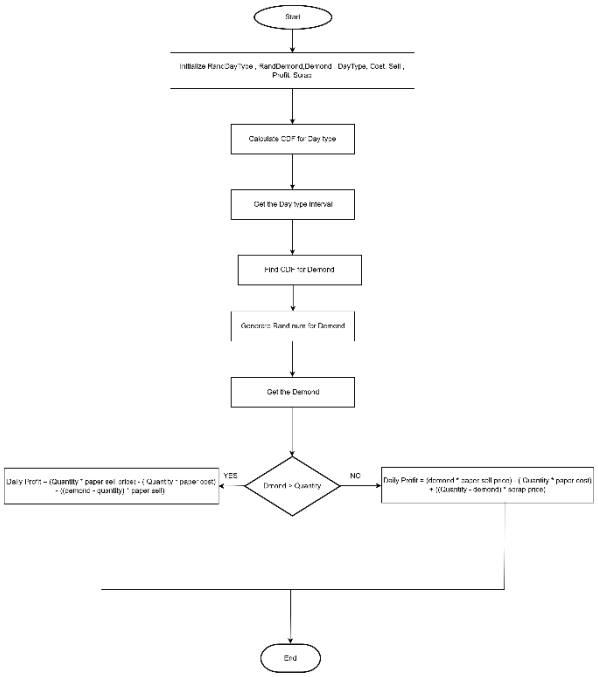


Flowchart

FIRST ONE



SECOND ONE



Results and Conclusion

- Present results in a table/graph: Quantity vs. Average Profit.
- Interpret how demand patterns and day types of influence profitability.