SIT_718 REAL WORLD ANALYTICS

ASSESMENT-4

1) a Explain why a linear programming model would be suitable for this case study.

Linear programming is used to generate the optimal result from a given set of constraints. The results are evolved into mathematical outputs consisting of the objective function. Since our problem is an apt real time case study and the constraints involved in this scenario are very linear in the problem and moreover, our task is to find out the amount of the composition of products A and B in the final composition for the user or customer. From overall components involving products Mango(M) and Orange (O)

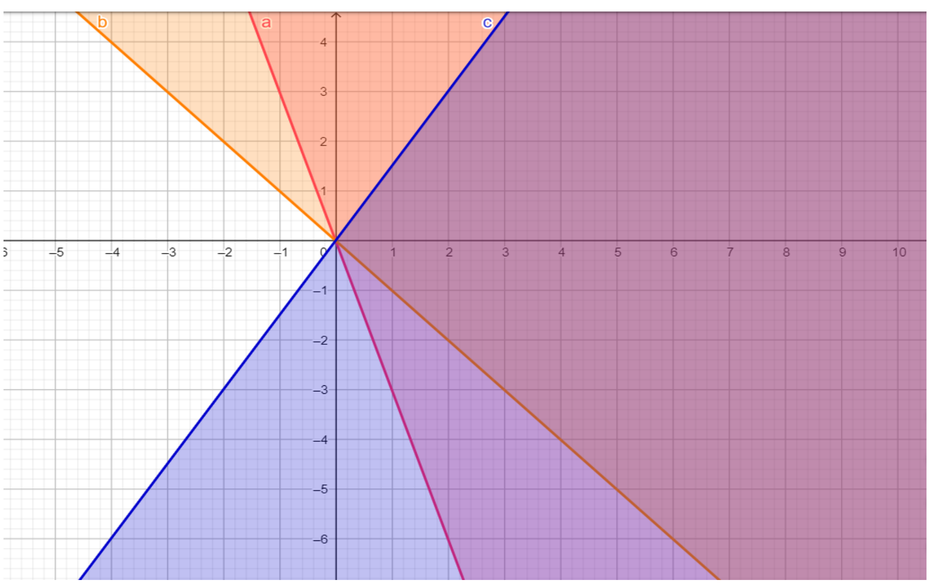
b) (LP) model for the factory that minimizes the total cost of producing the beverage while satisfying all constraints.

In this model we have taken 2 solutions from A and B to make a beverage where the cost of beverage should be less, and it should be more than 70 liters the equations given below has satisfied all the constraints

$$\boxed{\begin{array}{c} 1.5x + 0.5y \geq 0 \\ x + y \geq 0 \\ -3x + 2y \leq 0 \\ x + y \geq 70 \end{array}}$$

c) Graphical method to find the optimal solution:

from this graph we have determined the optimal solution at (28,42) and feasible region also



d) range for the cost ($): There is no other range for the cost of A that can be changed without affecting the optimum solution. From the slopes of the equations we can determine it.

$$\boxed{\begin{gathered}\frac{8x}{7}+\frac{7y}{7}=518/7\\ y\geq -x+70\\ y\geq 3x-20\\ y\leq 3x/2\\ y=\frac{8x}{7}+74\end{gathered}}$$

2. a) Sales of the products

$$\boldsymbol{(x_{e1}+x_{w1}+x_{s1})60+(x_{e2}+x_{w2}+x_{s2})55+(x_{e3}+x_{w3}+x_{s3})60}$$

Production cost: here we have derived an equation for cost of the product

$$\boldsymbol{(x_{e1}+x_{w1}+x_{s1})5+(x_{e2}+x_{w2}+x_{s2})4+(x_{e3}+x_{w3}+x_{s3})5}$$

Purchase Price/cost: here we determined the cost of the material

$$\boldsymbol{(x_{e1}+x_{w1}+x_{e3})30+(x_{w1}+x_{w2}+x_{w3})45+(x_{s1}+x_{s2}+x_{s3})50}$$

Profit (c) here we have derived the equation for total profit

$$\boldsymbol{\begin{aligned}(60x_{e1}-x_{w1}-30x_{s1})+(60x_{w1}-5x_{w1}-30_{e2})+(60x_{s1}-5x_{s1}-30x_{e3})+(55x_{e2}\\ -4x_{e2}-45x_{w1})+(55x_{w2}-4x_{w2}-45_{w2})+(55x_{s2}-4x_{s2}+45x_{w3})\\ +\ (60x_{e3}-5x_{e3}-50x_{s1})+(60x_{w3}-5x_{w3}-50_{s2})\\ +(60x_{s3}-5x_{s3}-45x_{s3})\end{aligned}}$$

Objective function: a function that is desired to be maximum or minimum

$$\boldsymbol{z=25x_{e1}+10x_{w1}+5x_{e1}+21x_{e2}-6x_{w2}+x_{s2}+25x_{e2}+10x_{w3}+5x_{s3}}$$

Constraints

Demand:

$$\boldsymbol{(\ x_{e1}\ +x_{w2}+x_{s1})\leq 4500}$$

$$\boldsymbol{(\ x_{e2}\ +x_{w2}+x_{s3})\leq 4000}$$

$$\boldsymbol{(\ x_{e3}\ +x_{w3}+x_{s3})\leq 4000}$$

Proportion of cotton: here we have created an equation for cotton proportion

$$0.5x_{e1} - 0.5x_{w1} - 0.5x_{s1} \geq 0$$

$$0.4x_{e2} - 0.6x_{w2} - 0.6x_{s2} \geq 0$$

$$0.6x_{e3} - 0.4x_{w3} - 0.4x_{s3} \geq 0$$

Proportion of wool: here we have created an equation for wool proportion

$$-0.3x_{e1} + 0.7x_{w1} - 0.3x_{s1} \geq 0$$

$$0.4x_{e2} - 0.6x_{w2} - 0.4x_{s2} \geq 0$$

$$0.5x_{e3} - 0.5x_{w3} - 0.5x_{s3} \geq 0$$

b) Optimal profit and optimal values of decision variables

Objective function: $222,250

Decision Variables:

$$x_{e1} = 3,150$$

$$x_{w1} = 1,350$$

$$x_{s1} = 0$$

$$x_{e2} = 2,400$$

$$x_{w2} = 3,150$$

$$x_{s2} = 3,150$$

$$x_{e3} = 3,150$$

$$x_{w3} = 0$$

$$x_{s3} = 0$$

2) a) Two- players- zero-sum-game:

In game theory, the game of matching pennies is often cited as an example of a zero-sum game. The game involves two players, A and B, simultaneously placing a penny on the table. The payoff depends on whether the pennies match or not. ... This is a zero-sum game because one player's gain is the other's loss. Zero-sum games are a specific example of constant sum games where the sum of each outcome is always zero. Such games are distributive, not integrative; the pie cannot be enlarged by good negotiation.

- In two player zero sum game when a player tries to maximize his payoff he simultaneously minimizes the payoff of the other player.
- The sum of utilities in each entry of the payoff matrix is zero

Payoff matrix for the game:

b) case:1

HELEN PROBABILITY	DAVID PROBABILITY			
	(1,4)	(2,3)	(3,2)	(4,1)
(1,5)	1	0	0	1
(2,4)	1	1	1	1
(3,3)	0	2	2	0
(4,2)	1	1	1	1
(5,1)	1	0	0	1

case:2

HELEN PROBABILITY	DAVID PROBABILITY					
	(0,5)	(1,4)	(2,3)	(3,2)	(4,1)	(5,0)
(0,6)	1	0	0	0	0	1
(1,5)	1	1	0	0	1	1
(2,4)	0	1	1	1	1	0
(3,3)	0	0	2	2	0	0
(4,2)	0	1	1	1	1	0
(5,1)	1	1	0	0	1	1
(6,0)	1	0	0	0	0	1

C. Saddle point:

A necessary and enough condition for a saddle point to exist is the presence of a payoff matrix element which is both a minimum of its row and a maximum of its column. A game may have more than one saddle point, but all must have the same value.

Given a matrix of n x n size, the task is to find saddle point of the matrix. A saddle point is an element of the matrix such that it is the minimum element in its row and maximum in its column.

A simple solution is to traverse all matrix elements one by one and check if the element is Saddle Point or not.

An efficient solution is based on below steps.
Traverse all rows one by one and do following for every row i.

1. Find the minimum element of current row and store column index of the minimum element.
2. Check if the row minimum element is also maximum in its column. We use the stored column index here.
3. If yes, then saddle point else continue till end of matrix.

In case 1 we don't have any saddle points

In case 2 we have four saddle points (2,4) (1,4) (4,2) (1,4)

(d) Liner Programming Model

Helen's

Max Z = V

$$\text{V}-(\boldsymbol{x_1} + \boldsymbol{x_2} + \boldsymbol{x_4}) \leq \mathbf{0}$$

$$\text{V}-(\boldsymbol{x_2} + \boldsymbol{x_3} + \boldsymbol{x_5} + \boldsymbol{x_6}) \leq \mathbf{0}$$

$$\text{V}-(\boldsymbol{x_3} + \mathbf{2}\boldsymbol{x_4} + \boldsymbol{x_5}) \leq \mathbf{0}$$

$$\text{V}-(\boldsymbol{x_3} + \mathbf{2}\boldsymbol{x_4} + \boldsymbol{x_5}) \leq \mathbf{0}$$

$$\text{V}-(\boldsymbol{x_2} + \boldsymbol{x_3} + \boldsymbol{x_5} + \boldsymbol{x_6}) \leq \mathbf{0}$$

$$\text{V}-(\boldsymbol{x_1} + \boldsymbol{x_2} + \boldsymbol{x_6} + \boldsymbol{x_7}) \leq \mathbf{0}$$

$$(\boldsymbol{x_1} + \boldsymbol{x_2} + \boldsymbol{x_3} + \boldsymbol{x_4} + \boldsymbol{x_5} + \boldsymbol{x_6} + \boldsymbol{x_7}) = 1$$

$$\boldsymbol{x_i} \geq \mathbf{0}$$

V=2/3

$$\boldsymbol{x_1} = \mathbf{0} + \boldsymbol{x_2} = \frac{\mathbf{2}}{\mathbf{3}} + \boldsymbol{x_3} = \mathbf{0} + \boldsymbol{x_4} = \frac{\mathbf{1}}{\mathbf{3}} + \boldsymbol{x_5} = \mathbf{0} + \boldsymbol{x_6} = \mathbf{0} + \boldsymbol{x_7} = \mathbf{0}$$

David's

Max W = V

$$\text{W}-(\boldsymbol{y_1} + \boldsymbol{y_6}) \geq \mathbf{0}$$

$$\text{W}-(\boldsymbol{y_1} + \boldsymbol{y_2} + \boldsymbol{y_5} + \boldsymbol{y_4}) \geq \mathbf{0}$$

$$W-(y_2 + y_3 + y_4 + y_5) \geq 0$$

$$V-(2\ y_3\ + 2y_4) \geq 0$$

$$V-(y_2 + y_3 + y_4 + y_5) \geq 0$$

$$V-(y_1 + y_2 + y_5 + y_6) \geq 0$$

$$y_1 + y_2 + y_3 + y_4 + y_5 + y_6 = 1$$

$$y_1 + y_2 + y_3 + y_4 + y_5 + y_6 = 1$$

$$y_j \geq 0$$

W = 2/3

$$y_1 = \frac{2}{3} + y_2 = 0 + y_3 = \frac{1}{3} + y_4 = 0 + y_5 = 0 + y_6 = 0$$

e) Helen's

```
#HELEN

library(lpSolveAPI)

matrix1 <- make.lp(0,8)

lp.control(matrix1, sense= "maximize")

set.objfn(matrix1,c(0,0,0,0,0,0,0,1))

add.constraint(matrix1,c(-1,-1,0,-1,0,0,-1,-1), "<=", 0)

add.constraint(matrix1,c(0,-1,-1,0,-1,-1,0,1), "<=", 0)

add.constraint(matrix1,c(0,0,-1,-2,-1,0,0,1), "<=", 0)

add.constraint(matrix1,c(-1,-1,0,-1,0,0,-1,-1), "<=", 0)

add.constraint(matrix1,c(0,-1,-1,0,-1,-1,0,-1), "<=", 0)

add.constraint(matrix1,c(-1,-1,0,0,0,-1,-1,-1), "<=", 0)

add.constraint(matrix1,c(1,1,1,1,1,1,1,0), "<=", 0)

solve(matrix1)
```

```
get.objective(matrix1)

get.variables(matrix1)

matrix1
```

f) David's

```
#DAVID

matrix1<-make.lp(0,7)

lp.control(matrix1,sense="maximize")

set.objfn(matrix1, c(0,0,0,0,0,0,1))

add.constraint(matrix1,c(-1,0,0,0,0,-1,1), ">=", 0)

add.constraint(matrix1,c(-1,-1,0,0,-1,-1,1), ">=", 0)

add.constraint(matrix1,c(0,-1,-1,-1,-1,0,1), ">=", 0)

add.constraint(matrix1,c(0,0,-2,-2,0,0,1), ">=", 0)

add.constraint(matrix1,c(-1,-1,0,0,-1,-1,1), ">=", 0)

add.constraint(matrix1,c(-1,0,0,0,0,-1,1), ">=", 0)

add.constraint(matrix1,c(1,1,1,1,1,1,0), "=",1)

solve(matrix1)

get.objective(matrix1)

get.variables(matrix1)

get.constraints(matrix1)

matrix1
```

From this we can interpret that nobody wins the game which justifies two player zero sum game.