RM 294 – Optimization 1

Group 16: Project 1 – Linear Programming

Group Members:

Group Member	**UT EID**
Manvi Goyal	mg65952
Muskan Agrawal	ma64547
Rochan Nehete	rrn479
Rianna Patel	rnp599

——————————————— Note: The code snippets are posted at the end of this document ————————————

Questions:

1) Assume that your company is deciding how to spend a marketing budget of $10M. You work in the marketing department as a data scientist and the chief marketing officer has asked you to write a report recommending how to spread this budget among several marketing mediums. Your department has employed an outside consulting firm to estimate the return on investment (ROI) of each marketing medium under consideration. The results are in the table below, and also in a CSV attached to this assignment:

The total marketing budget is $10M which we have to allocate to 10 different platforms. Our objective here is to maximize the ROI (return on investment) by proposing the best budget allocation. We will use the below values of ROI to allocate the budgets:

Platform	Print	TV	SEO	AdWords	Facebook	LinkedIn	Instagram	Snapchat	Twitter	Email
ROI	0.031	0.049	0.024	0.039	0.016	0.024	0.046	0.026	0.033	0.044

——————————————————— End of Question 1 ———————————————————

2) On top of these ROIs, your boss has decided to constrain your budget as follows: a. The amount invested in print and TV should be no more than the amount spent on Facebook and Email. Surprisingly, email seems to be a great channel for reaching real people. b. The total amount used in social media (Facebook, LinkedIn, Instagram, Snapchat, and Twitter) should be at least twice of SEO and AdWords. c. For each platform, the amount invested should be no more than $3M.

There are total 13 constraints that our objective function will be subjected to:

a) Investment (Print) + Investment (TV) <= Investment (Facebook) + Investment (Email)
b) Investment (Facebook) + Investment (LinkedIn) + Investment (Instagram) + Investment (Snapchat) + Investment (Twitter) >= 2*(Investment (SEO) + Investment (AdWords))
c) Investment for each platform(10 platforms) <= $3M (*Setting upper limit to investment in each platform*)
d) Total investment made on each platform <= $10M *(Total Marketing budget)*

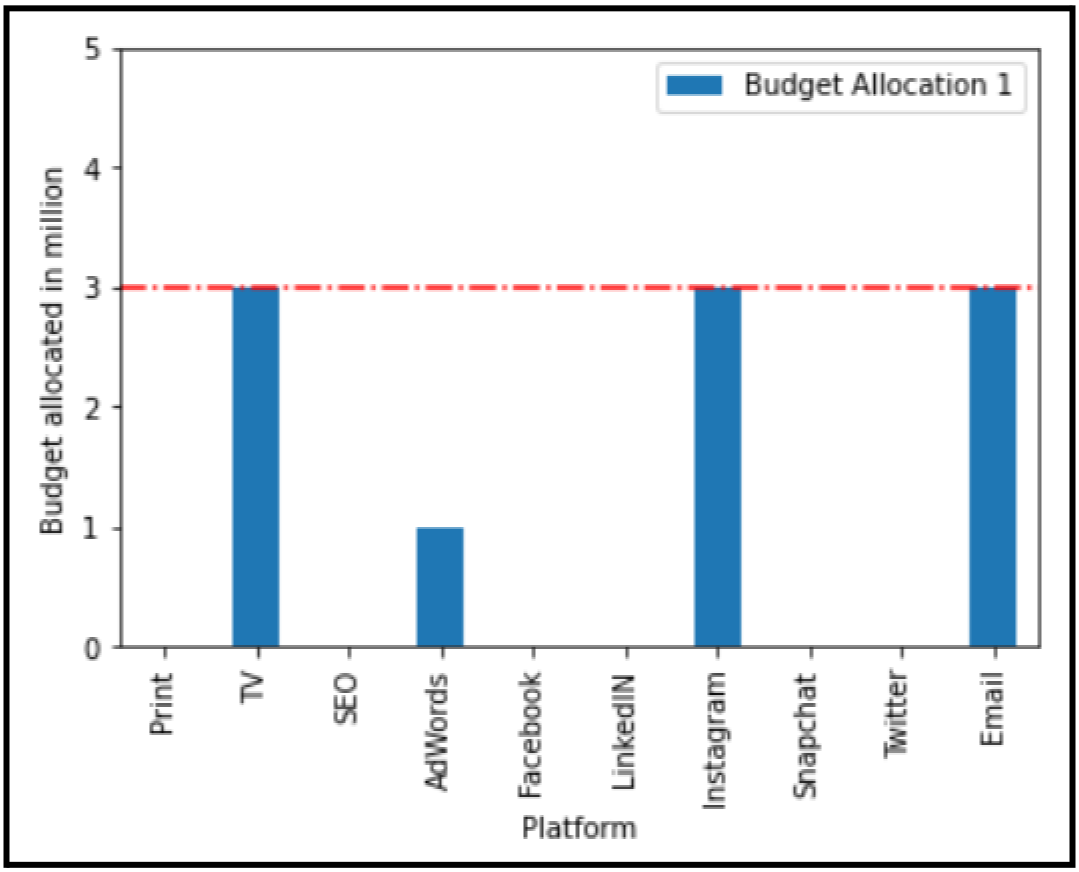
——————————————————— End of Question 2 ———————————————————

3) Formulate the marketing budget allocation problem as a linear program. Use gurobi to find the optimal budget allocation

Using all the above mentioned 13 constraints, we maximized our objective function by formulating our problems as linear equations and using Gurobi to solve these in order to find the optimal budget allocation. As per our results, highlighted below, the Maximum optimal ROI would be $456,000.

```
Maximum ROI on the investment of 10 Million: $ 0.456 M
```

Below is the actual allocation that will be done on the respective platforms in order to achieve the optimal ROI.



— The red line represent the upper limit set for investment in each platform

-- End of Question 3 --

4) Your boss is happy to see the promising results presented by the marketing department. However, your boss is also very concerned because your boss recalls being somewhat disappointed after following such recommendations in the past. To be cautious about the decision, your team has decided to get another opinion about the ROI data and rerun the analysis. The second consulting firm returns the estimates of the ROI data in the table below (also in the CSV file mentioned above). You are asked to compare the two optimal allocations from these two ROI estimates.

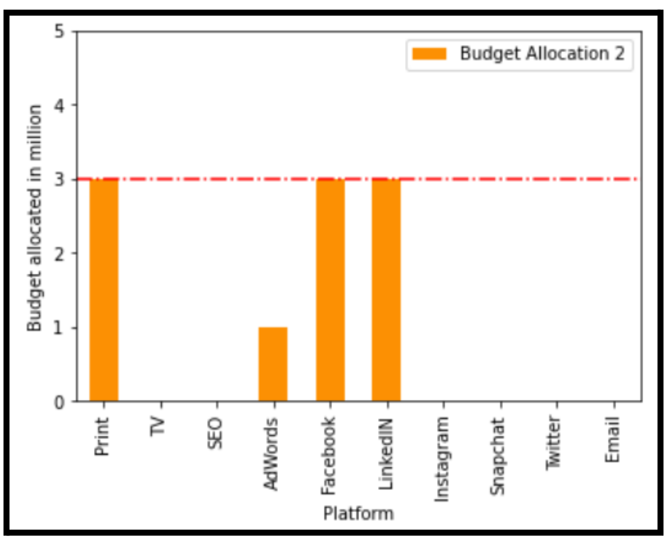
Below are the new values of ROI (Second ROI), that will be now used to find the new optimal value of the budget allocation.

Platform	Print	TV	SEO	AdWords	Facebook	LinkedIn	Instagram	Snapchat	Twitter	Email
Second Firms ROI Estimate	0.049	0.023	0.024	0.039	0.044	0.046	0.026	0.019	0.037	0.026

We again used all the above mentioned 13 constraints *(keeping them unchanged)* and maximized our objective function using Gurobi. As per our results, highlighted below, the Maximum optimal ROI again comes out to be $456,000 that is no change in the optimal value when we change the ROI.

```
Maximum ROI on the investment of 10 Million: $ 0.456 M
```

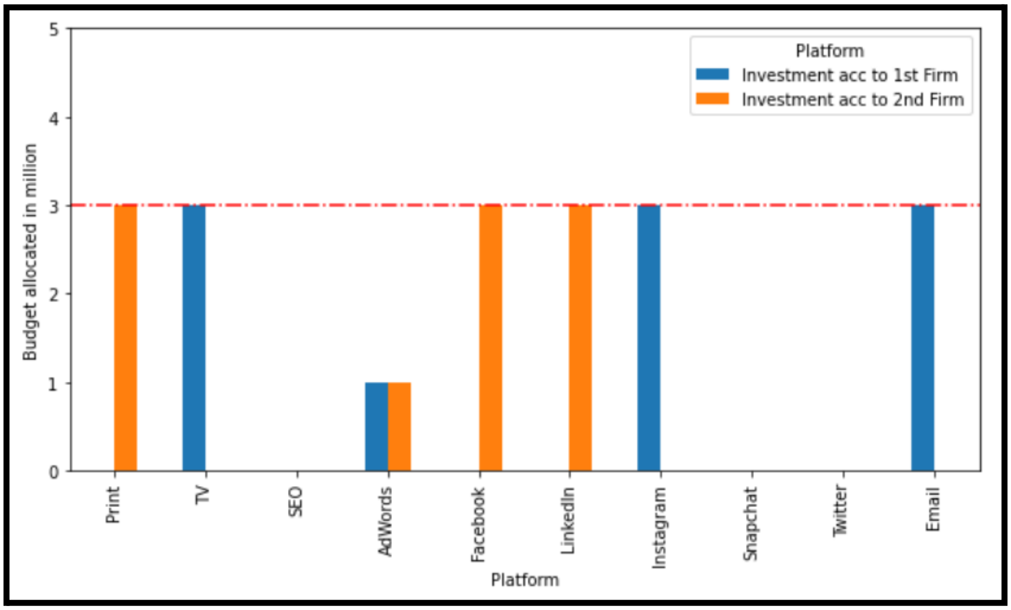
However, the actual budget allocation values changes for the platforms which are highlighted below : (The red line represent the upper limit set for each platform)



-- End of Question 4 --

5) Are the allocations the same? Assuming the first ROI data is correct, if you were to use the second allocation (the allocation that assumed the second ROI data was correct) how much lower would the objective be relative to the optimal objective (the one that uses the first ROI data and the first allocation)? Assuming the second ROI data is correct, if you used the first allocation how much lower would the objective be relative to the optimal objective? Do you think the third constraint above, based on your boss' experience, is useful?

The allocations obtained on the basis of two ROI are not the same, however, the value of optimal ROI is the same(The red line represent the upper limit set for each platform)



Case 1: When first ROI is correct, and we use the second allocation, the value of objective gets lowered by $0.204 Million when compared with optimal value of $0.456 Million

```
In [29]: # If the first ROI was correct and we used the 2nd allocation, we lost how much?
         round(roi_investment_matrix.loc['Return acc to 1st ROI on 2nd allocation']['Total'] - roi_investment_matrix.loc['Return acc to 1s

Out[29]: -0.204
```

Case 2: When second ROI is correct, and we use the first allocation, the value of objective gets lowered by $0.192 Million when compared with optimal value of $0.456 Million

```
In [30]: was correct and we used the 1st allocation, we lost how much?
         _matrix.loc['Return acc to 2nd ROI on 1st allocation']['Total'] - roi_investment_matrix.loc['Return acc to 2nd Firm']['Total'],3)

Out[30]: -0.192
```

Is the third constraint useful?

To answer this question, we tried removing the constraint and calculating the ROI and allocation for the same situation. Following is the ROI obtained using the first and second firm's estimates when we remove the third constraint:

```
Max ROI on the investment of 10 Million if upper limit on investment at each head is removed:  0.465 M
```

The maximum ROI remains unchanged but an interesting observation can be seen in the allocations:

Platform	Print	TV	SEO	AdWords	Facebook	LinkedIn	Instagram	Snapchat	Twitter	Email
ROI	0.031	0.049	0.024	0.039	0.016	0.024	0.046	0.026	0.033	0.044
Second Firms ROI Estimate	0.049	0.023	0.024	0.039	0.044	0.046	0.026	0.019	0.037	0.026
Investment acc to 1st Firm	0.000	3.000	0.000	1.000	0.000	0.000	3.000	0.000	0.000	3.000
Investment acc to 2nd Firm	3.000	0.000	0.000	1.000	3.000	3.000	0.000	0.000	0.000	0.000
Return acc to 1st Firm	0.000	0.147	0.000	0.039	0.000	0.000	0.138	0.000	0.000	0.132
Return acc to 2nd Firm	0.147	0.000	0.000	0.039	0.132	0.138	0.000	0.000	0.000	0.000
Return acc to 1st ROI on 2nd allocation	0.093	0.000	0.000	0.039	0.048	0.072	0.000	0.000	0.000	0.000
Return acc to 2nd ROI on 1st allocation	0.000	0.069	0.000	0.039	0.000	0.000	0.078	0.000	0.000	0.078
Investment acc to 1st Firm but no upper limit	0.000	5.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	5.000
Investment acc to 2nd Firm but no upper limit	5.000	0.000	0.000	0.000	5.000	0.000	0.000	0.000	0.000	0.000

We can see that the optimal allocation is directed towards 2 channels only, TV and email using estimates of the first firm and Print and Facebook using estimates of the second firm. Other channels are not being given any allocation.

The third constraint is advantageous. This constraint helps to diversify your portfolio of channels, and that way you can reduce liability on a limited number of channels. So if the actual ROI is different from the predicted ROI, the loss is low. For instance, if we invest 5M each in TV and Email but the actual ROI is different than assumed ROI, we will lose a lot of money and opportunity.

——————————————————————— End of Question 5 ———————————————————————

6) To explore this further, perform some analysis of how your optimal allocation would change based on changes in the ROI data. Use the first ROI data as your starting point. By how much could each advertising medium's ROI increase or decrease and still result in the same optimal allocation you found in step (3)?

Out[10]:

	Print	TV	SEO	AdWords	Facebook	LinkedIn	Instagram	Snapchat	Twitter	Email
Lower limit on ROI	-inf	0.039	-inf	0.033	-inf	-inf	0.039	-inf	-inf	0.029
Upper limit on ROI	0.049	0.062	0.039	0.046	0.029	0.039	inf	0.039	0.039	inf
ROI estimate by Firm 1	0.031	0.049	0.024	0.039	0.016	0.024	0.046	0.026	0.033	0.044

Each advertising medium's ROI can increase or decrease by the upper and lower limits given by this table, resulting in the same optimal allocation found in step 3. We can see that for Instagram, even if we increase ROI to infi, the allocation will remain the same i.e., 3M since that channel is one of the best ROI giving channels.

——————————————————————— End of Question 6 ———————————————————————

7) Your boss has gained permission to reinvest half of the return. For example, if the marketing obtains a 4% return in January, the budget of February will be $10M + $10M * 4% * 50% = $10.2M. The monthly ROI for next year is given in Project1.Rdata. The three constraints given by your boss are still in place for each month. What is the optimal allocation for each month?

<u>**Approach:**</u> We seek to obtain optimal allocation each month. Hence we changed the objective function for each month according to the given predicted monthly ROI's (table ROI_mat) and found the optimal allocation and actual ROI in the month. Upon doing to we obtain the following allocations :

	Print	TV	SEO	AdWords	Facebook	LinkedIn	Instagram	Snapchat	Twitter	Email
0	3.000000	0.0	0.0	1.333333	0.000000	0.000000	2.666667	0.0	0.000000	3.000000
1	3.000000	0.0	0.0	2.395500	3.000000	0.000000	0.000000	0.0	1.791000	0.000000
2	0.000000	0.0	0.0	3.000000	0.000000	3.000000	1.389648	0.0	3.000000	0.000000
3	0.000000	0.0	0.0	3.000000	0.000000	3.000000	3.000000	0.0	1.596856	0.000000
4	1.804100	0.0	0.0	0.000000	0.000000	0.000000	3.000000	0.0	3.000000	3.000000
5	3.000000	0.0	0.0	0.000000	0.000000	0.000000	3.000000	0.0	2.020172	3.000000
6	1.123777	0.0	0.0	3.000000	1.123777	0.000000	3.000000	0.0	3.000000	0.000000
7	3.000000	0.0	0.0	1.827294	0.000000	0.654588	0.000000	0.0	3.000000	3.000000
8	1.362933	0.0	0.0	3.000000	0.000000	3.000000	0.000000	0.0	3.000000	1.362933
9	0.000000	0.0	0.0	3.000000	0.000000	3.000000	3.000000	0.0	0.000000	2.955475
10	3.000000	0.0	0.0	2.056421	0.000000	1.112842	3.000000	0.0	0.000000	3.000000
11	3.000000	3.0	0.0	0.427951	3.000000	0.000000	0.000000	0.0	0.000000	3.000000

With the 3 constraints given by the boss and new monthly budgets which include 50% of the previous year's return, the optimal allocation moving forward is given in the table above.

—————————————————————— End of Question 7 ——————————————————————

8) A stable budget is defined as a monthly allocation such that for each platform the monthly change in spend is no more than $1M. Is the allocation you found stable? If it isn't, you do not need to solve a new optimization model. Describe how my might model this

In [56]:
```
df_monthly_allocation.diff()
```

Out[56]:

	Print	TV	SEO	AdWords	Facebook	LinkedIn	Instagram	Snapchat	Twitter	Email
0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
1	0.000000	0.0	0.0	1.062167	3.000000	0.000000	-2.666667	0.0	1.791000	-3.000000
2	-3.000000	0.0	0.0	0.604500	-3.000000	3.000000	1.389648	0.0	1.209000	0.000000
3	0.000000	0.0	0.0	0.000000	0.000000	0.000000	1.610352	0.0	-1.403144	0.000000
4	1.804100	0.0	0.0	-3.000000	0.000000	-3.000000	0.000000	0.0	1.403144	3.000000
5	1.195900	0.0	0.0	0.000000	0.000000	0.000000	0.000000	0.0	-0.979828	0.000000
6	-1.876223	0.0	0.0	3.000000	1.123777	0.000000	0.000000	0.0	0.979828	-3.000000
7	1.876223	0.0	0.0	-1.172706	-1.123777	0.654588	-3.000000	0.0	0.000000	3.000000
8	-1.637067	0.0	0.0	1.172706	0.000000	2.345412	0.000000	0.0	0.000000	-1.637067
9	-1.362933	0.0	0.0	0.000000	0.000000	0.000000	3.000000	0.0	-3.000000	1.592543
10	3.000000	0.0	0.0	-0.943579	0.000000	-1.887158	0.000000	0.0	0.000000	0.044525
11	0.000000	3.0	0.0	-1.628470	3.000000	-1.112842	-3.000000	0.0	0.000000	0.000000

After finding the monthly allocation for each medium, we found that each medium, except Snapchat, has at least one period where the monthly allocation difference is greater than $1M. Thus, the budget allocation is not stable and we will add additional constraints to create a stable budget. We would add 10 more constraints for each month to restrict the amount allocated.

In a real world scenario, it would be unethical to invest for one month in a channel and then not invest in it the next month, as this would lead to a bad relationship between the channel clients and the business. Therefore, we believe that there should be a constraint making sure that the investment per month should happen in a serial fashion in order to make the budget stable.

—————————————————————— End of Question 8 ——————————————————————

Code snippets:

Part 3:

In [1]:
```python
import numpy as np
import gurobipy as gp
import pandas as pd
```

In [2]:
```python
roi_data = pd.read_csv('ROI_data.csv', index_col=0)
roi_data
```

Out[2]:

Platform	Print	TV	SEO	AdWords	Facebook	LinkedIn	Instagram	Snapchat	Twitter	Email
ROI	0.031	0.049	0.024	0.039	0.016	0.024	0.046	0.026	0.033	0.044
Second Firms ROI Estimate	0.049	0.023	0.024	0.039	0.044	0.046	0.026	0.019	0.037	0.026

Part 3

In [3]:
```python
obj = roi_data.iloc[0].values #ROI estimates of firm 1
obj
```

Out[3]:
```
array([0.031, 0.049, 0.024, 0.039, 0.016, 0.024, 0.046, 0.026, 0.033,
       0.044])
```

In [4]:
```python
obj = roi_data.iloc[0].values # objective vector
A = np.zeros((13,10)) # initialize constraint matrix
A[0,:] = [1,1,0,0,-1,0,0,0,0,-1] # Print & TV, no more than FB and email
A[1,:] = [0,0,-2,-2,1,1,1,1,1,0] # Social media more than twice of SEO and AdWords
A[2,:] = [1,1,1,1,1,1,1,1,1,1] # Total Amount
A[3:13,0:10] = np.diag(np.ones(10)) #Money spent on any medium be less than or equal to 3 million USD

b = np.array([0,0,10,3,3,3,3,3,3,3,3,3,3]) # limits on production, storage, and demand
sense = np.array(['<','>','<','<','<','<','<','<','<','<','<','<','<']) # all constraints are less than or equal constraints
```

In [5]:
```python
MarkMod = gp.Model()

Markx = MarkMod.addMVar(len(obj)) # tell the model how many variables there are
# must define the variables before adding constraints because variables go into the constraints
MarkModCon = MarkMod.addMConstrs(A, Markx, sense, b) # add the constraints to the model
MarkMod.setMObjective(None,obj,0,sense=gp.GRB.MAXIMIZE) # add the objective to the model...we'll talk about the None and the 0
# None for no quadratic values in our equation, linear is the obj,
MarkMod.Params.OutputFlag = 0 # tell gurobi to shut up!!
MarkMod.optimize()
```

```
Set parameter Username
Academic license - for non-commercial use only - expires 2023-09-14
```

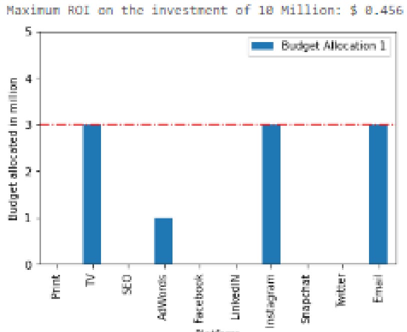
In [6]:
```python
print("Max ROI on the investment of 10 Million: $", round(MarkMod.objVal,5), "M")
```

```
Max ROI on the investment of 10 Million: $ 0.456 M
```

Optimal Budget Allocation and ROI of firm 1

In [7]:
```python
alloc_1 = MarkMod.x
Max_ROI_1 = MarkMod.objVal
alloc_1_df = pd.DataFrame(index=['Print','TV','SEO','AdWords','Facebook','LinkedIN','Instagram','Snapchat','Twitter','Email'])
alloc_1_df['Budget Allocation 1']=MarkMod.x
alloc_1_df.plot(kind='bar', xlabel='Platform',ylabel='Budget allocated in million', ylim = (0,5))
import matplotlib.pyplot as plt
plt.axhline(y=3, color='r', linestyle='-.')
print("Maximum ROI on the investment of 10 Million: $", round(Max_ROI_1,5), "M")
```

```
Maximum ROI on the investment of 10 Million: $ 0.456 M
```



Part 4

In [13]:
```
#Optimize using second ROI estimates
obj2 = roi_data.iloc[1].values # objective vector
A2 = np.zeros((13,10)) # initialize constraint matrix
A2[0,:] = [1,1,0,0,-1,0,0,0,0,-1] # Print & TV, no more than FB and email
A2[1,:] = [0,0,-2,-2,1,1,1,1,1,0] # Social media more than twice of SEO and AdWords
A2[2,:] = [1,1,1,1,1,1,1,1,1,1] # Total Amount
A2[3:13,0:10] = np.diag(np.ones(10)) #Money spent on any medium be less than or equal to 3 million USD

b2 = np.array([0,0,10,3,3,3,3,3,3,3,3,3,3]) # limits on production, storage, and demand
sense = np.array(['<','>','<','<','<','<','<','<','<','<','<','<','<']) # all constraints are less than or equal constraints
```

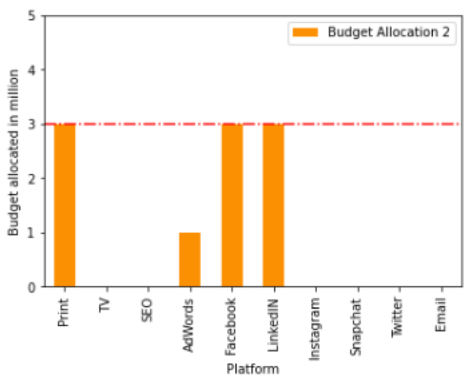
In [14]:
```
MarkMod = gp.Model()

Markx = MarkMod.addMVar(len(obj2)) # tell the model how many variables there are
# must define the variables before adding constraints because variables go into the constraints
MarkModCon = MarkMod.addMConstrs(A2, Markx, sense, b2) # add the constraints to the model
MarkMod.setMObjective(None,obj2,0,sense=gp.GRB.MAXIMIZE) # add the objective to the model...we'll talk about the None and the 0
# None for no quadratic values in our equation, linear is the obj,
MarkMod.Params.OutputFlag = 0 # tell gurobi to shut up!!
MarkMod.optimize()
```

In [15]:
```
print("Max ROI on the investment of 10 Million: ", round(MarkMod.objVal,5), "M")
```

```
Max ROI on the investment of 10 Million:  0.456 M
```

In [16]:
```
Markx.x
alloc_2 = Markx.x
Max_ROI_2 = round(MarkMod.objVal,5)
alloc_2_df = pd.DataFrame(index=['Print','TV','SEO','AdWords','Facebook','LinkedIN','Instagram','Snapchat','Twitter','Email'])
alloc_2_df['Budget Allocation 2']=MarkMod.x
alloc_2_df.plot(kind='bar', xlabel='Platform',ylabel='Budget allocated in million',color='#fc9003',  ylim = (0,5))
plt.axhline(y=3, color='r', linestyle='-.')
print("Maximum ROI on the investment of 10 Million: $", round(Max_ROI_2,5), "M")
```

```
Maximum ROI on the investment of 10 Million: $ 0.456 M
```

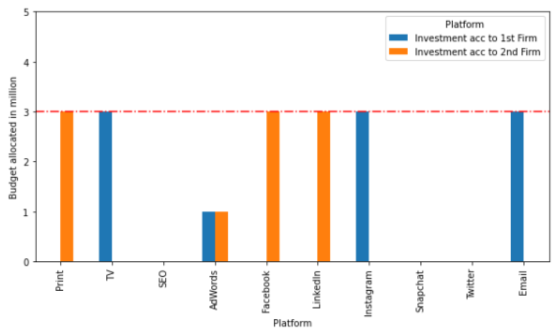


Part 5:

Comparison of optimal allocations from ROI estimates of the firms

In [20]:
```
#Creating a dataframe roi_investment to keep a track of all the given scenarios we would go through in the project
roi_investment[['Investment acc to 1st Firm','Investment acc to 2nd Firm']].plot(kind = 'bar',xlabel='Platform',ylabel='Budget allocated in million', figsize=(10,5), ylim = (0,5))
plt.axhline(y=3, color='r', linestyle='-.')
```

Out[20]:
```
<matplotlib.lines.Line2D at 0x20781362d30>
```



In [26]:
```
# If the first ROI was correct and we used the 2nd allocation, we lost how much?
round(roi_investment_matrix.loc['Return acc to 1st ROI on 2nd allocation']['Total'] - roi_investment_matrix.loc['Return acc to 1st Firm']['Total'],3)
```

Out[26]:
```
-0.204
```

In [27]:
```
# If the second ROI was correct and we used the 1st allocation, we lost how much?
round(roi_investment_matrix.loc['Return acc to 2nd ROI on 1st allocation']['Total'] - roi_investment_matrix.loc['Return acc to 2nd Firm']['Total'],3)
```

Out[27]:
```
-0.192
```

In [28]:
```
#Lets check how much we can make if we remove the 3rd constriant for first firm

obj3 = roi_data.iloc[0].values # objective vector
A3 = np.zeros((3,10)) # initialize constraint matrix
A3[0,:] = [1,1,0,0,-1,0,0,0,0,-1] # Print & TV, no more than FB and email
A3[1,:] = [0,0,-2,-2,1,1,1,1,1,0] # Social media more than twice of SEO and AdWords
A3[2,:] = [1,1,1,1,1,1,1,1,1,1] # Total Amount

b3 = np.array([0,0,10]) # limits on production, storage, and demand
sense = np.array(['<','>','<']) # all constraints are less than or equal constraints
```

In [29]:
```
MarkMod = gp.Model()

Markx = MarkMod.addMVar(len(obj3)) # tell the model how many variables there are
# must define the variables before adding constraints because variables go into the constraints
MarkModCon = MarkMod.addMConstrs(A3, Markx, sense, b3) # add the constraints to the model
MarkMod.setMObjective(None,obj3,0,sense=gp.GRB.MAXIMIZE) # add the objective to the model...we'll talk about the None and the 0
# None for no quadratic values in our equation, linear is the obj,
MarkMod.Params.OutputFlag = 0 # tell gurobi to shut up!!
MarkMod.optimize()
```

In [30]:
```
print("Max ROI on the investment of 10 Million if upper limit on investment at each head is removed: ", round(MarkMod.objVal,5),
```

```
Max ROI on the investment of 10 Million if upper limit on investment at each head is removed:  0.465 M
```

In [32]:
```
#Lets check how much we can make if we remove the 3rd constriant for second firm

obj4 = roi_data.iloc[1].values # objective vector
A4 = np.zeros((3,10)) # initialize constraint matrix
A4[0,:] = [1,1,0,0,-1,0,0,0,0,-1] # Print & TV, no more than FB and email
A4[1,:] = [0,0,-2,-2,1,1,1,1,1,0] # Social media more than twice of SEO and AdWords
A4[2,:] = [1,1,1,1,1,1,1,1,1,1] # Total Amount

b4 = np.array([0,0,10]) # limits on production, storage, and demand
sense = np.array(['<','>','<']) # all constraints are less than or equal constraints
```

In [33]:
```
MarkMod = gp.Model()

Markx = MarkMod.addMVar(len(obj4)) # tell the model how many variables there are
# must define the variables before adding constraints because variables go into the constraints
MarkModCon = MarkMod.addMConstrs(A4, Markx, sense, b4) # add the constraints to the model
MarkMod.setMObjective(None,obj4,0,sense=gp.GRB.MAXIMIZE) # add the objective to the model...we'll talk about the None and the 0
# None for no quadratic values in our equation, linear is the obj,
MarkMod.Params.OutputFlag = 0 # tell gurobi to shut up!!
MarkMod.optimize()
```

In [34]:
```
print("Max ROI on the investment of 10 Million if upper limit on investment at each head is removed: ", round(MarkMod.objVal,5),
```

```
Max ROI on the investment of 10 Million if upper limit on investment at each head is removed:  0.465 M
```

In [40]:
```
#calculating a Total ROI in % as a column
roi_investment_matrix_all_combinations['Total'] = roi_investment_matrix_all_combinations.sum(axis=1)
roi_investment_matrix_all_combinations
```

Out[40]:

Platform	Print	TV	SEO	AdWords	Facebook	LinkedIn	Instagram	Snapchat	Twitter	Email	Total
ROI	0.031	0.049	0.024	0.039	0.016	0.024	0.046	0.026	0.033	0.044	0.332
Second Firms ROI Estimate	0.049	0.023	0.024	0.039	0.044	0.046	0.026	0.019	0.037	0.026	0.333
Investment acc to 1st Firm	0.000	3.000	0.000	1.000	0.000	0.000	3.000	0.000	0.000	3.000	10.000
Investment acc to 2nd Firm	3.000	0.000	0.000	1.000	3.000	3.000	0.000	0.000	0.000	0.000	10.000
Return acc to 1st Firm	0.000	0.147	0.000	0.039	0.000	0.000	0.138	0.000	0.000	0.132	0.456
Return acc to 2nd Firm	0.147	0.000	0.000	0.039	0.132	0.138	0.000	0.000	0.000	0.000	0.456
Return acc to 1st ROI on 2nd allocation	0.093	0.000	0.000	0.039	0.048	0.072	0.000	0.000	0.000	0.000	0.252
Return acc to 2nd ROI on 1st allocation	0.000	0.069	0.000	0.039	0.000	0.000	0.078	0.000	0.000	0.078	0.264
Investment acc to 1st Firm but no upper limit	0.000	5.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	5.000	10.000
Investment acc to 2nd Firm but no upper limit	5.000	0.000	0.000	0.000	5.000	0.000	0.000	0.000	0.000	0.000	10.000
Return acc to 1st Firm but no upper limit	0.000	0.245	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.220	0.465
Return acc to 2nd Firm but no upper limit	0.245	0.000	0.000	0.000	0.220	0.000	0.000	0.000	0.000	0.000	0.465
Return acc to 1st ROI on 2nd allocation without upper limit	0.155	0.000	0.000	0.000	0.080	0.000	0.000	0.000	0.000	0.000	0.235
Return acc to 2nd ROI on 1st allocation without upper limit	0.000	0.115	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.130	0.245

In [41]:
```
# If the first ROI was correct and we used the 2nd allocation, we lost how much if no upper bound at spending on each category?
roi_investment_matrix_all_combinations.loc['Return acc to 1st ROI on 2nd allocation without upper limit']['Total'] - roi_investment_matrix_all_combinations.loc['Return acc to 1st Firm but no upper limit']['Total']
```

Out[41]:
```
-0.22999999999999998
```

In [42]:
```
# If the second ROI was correct and we used the 1st allocation, we lost how much if no upper bound at spending on each category?
roi_investment_matrix_all_combinations.loc['Return acc to 2nd ROI on 1st allocation without upper limit']['Total'] - roi_investment_matrix_all_combinations.loc['Return acc to 2nd Firm but no upper limit']['Total']
```

Out[42]:
```
-0.21999999999999997
```

The third constraint is useful because without that we are spending in only 2 channels which increases the loss amount in case the assumed ROI is wrong.

Part 6:

To explore this further perform some analysis of how your optimal allocation would change based on changes in the ROI data. Use the first ROI data as your sta
step (3)?

In [8]:
```
Markx.SAObjLow
```

Out[8]:
```
array([ -inf, 0.039,  -inf, 0.033,  -inf,  -inf, 0.039,  -inf,  -inf,
        0.029])
```

In [9]:
```
Markx.SAObjUp
```

Out[9]:
```
array([0.049, 0.062, 0.039, 0.046, 0.029, 0.039,   inf, 0.039, 0.039,
         inf])
```

In [10]:
```
df_limits_on_ROI = pd.DataFrame(columns=['Print','TV','SEO','AdWords','Facebook','LinkedIn','Instagram','Snapchat','Twitter','Email'])
df_limits_on_ROI.loc[len(df_limits_on_ROI)] = Markx.SAObjLow
df_limits_on_ROI.loc[len(df_limits_on_ROI)] = Markx.SAObjUp
df_limits_on_ROI.rename(index={0:'Lower limit on ROI'},inplace=True)
df_limits_on_ROI.rename(index={1:'Upper limit on ROI'},inplace=True)
df_limits_on_ROI.loc[len(df_limits_on_ROI)] = roi_data.iloc[0]
df_limits_on_ROI.rename(index={2:'ROI estimate by Firm 1'},inplace=True)
df_limits_on_ROI
```

Out[10]:

	Print	TV	SEO	AdWords	Facebook	LinkedIn	Instagram	Snapchat	Twitter	Email
Lower limit on ROI	-inf	0.039	-inf	0.033	-inf	-inf	0.039	-inf	-inf	0.029
Upper limit on ROI	0.049	0.062	0.039	0.046	0.029	0.039	inf	0.039	0.039	inf
ROI estimate by Firm 1	0.031	0.049	0.024	0.039	0.016	0.024	0.046	0.026	0.033	0.044

Part 7:

Your boss has gained permission to reinvest half of the return. For example, if the marketing obtains a 4% return in Jan
for each month. What is the optimal allocation for each month?

In [43]:
```
# Loading ROI Monthly csv and converting the values to percentage by dividing by 100
roi_monthly = (pd.read_csv('roi_mat.csv', index_col=0))/100
roi_monthly
```

In [44]:
```
# Initialize a Dataframe with incremental budget values

budget = [10] #set initial $10M budget
budget_df = pd.DataFrame(budget, columns=['budget'])

# Initialize a DataFrame to save the ROI values each month

df_monthly_returns = pd.DataFrame(columns=['return'])

# Initialize a Dataframe to store monthly allocations

df_monthly_allocation = pd.DataFrame(columns=['Print','TV','SEO','AdWords','Facebook','LinkedIn','Instagram','Snapchat','Twitter','Email'])

for i in range(len(roi_monthly)):
    obj = roi_monthly.iloc[i].values # objective vector, changes every month
    A = np.zeros((13,10)) # initialize constraint matrix
    A[0,:] = [1,1,0,0,-1,0,0,0,0,-1] # Print & TV, no more than FB and email
    A[1,:] = [0,0,-2,-2,1,1,1,1,1,0] # Social media more than twice of SEO and AdWords
    A[2,:] = [1,1,1,1,1,1,1,1,1,1] # Total Amount
    A[3:13,0:10] = np.diag(np.ones(10)) # Money spent on any medium be less than or equal to 3 million USD

    b = np.array([0,0,budget_df.loc[i]['budget'],3,3,3,3,3,3,3,3,3,3]) # The total budget keeps changing according to previous month's returns so it is dynamic
    sense = np.array(['<','>','<','<','<','<','<','<','<','<','<','<','<'])
    MarkMod = gp.Model()

    Markx = MarkMod.addMVar(len(obj)) # tell the model how many variables there are
    # must define the variables before adding constraints because variables go into the constraints
    MarkModCon = MarkMod.addMConstrs(A, Markx, sense, b) # add the constraints to the model
    MarkMod.setMObjective(None,obj,0,sense=gp.GRB.MAXIMIZE) # add the objective to the model
    # None for no quadratic values in our equation, linear is the obj,
    MarkMod.Params.OutputFlag = 0 # tell gurobi to shut up!!
    MarkMod.optimize()

    # Storing the returns from marketing every month in the data frame df_monthly_returns
    df_monthly_returns.loc[len(df_monthly_returns)] = round(MarkMod.objVal,5)

    # Storing the monthly allocation every month in the data frame df_monthly_allocation
    df_monthly_allocation.loc[len(df_monthly_allocation)] = Markx.x

    # Storing the available budget for next month by adding the 50% of previous month's returns to 10M
    budget_df.loc[len(budget_df)] = (budget_df.loc[i] + 0.5*(MarkMod.objVal))
```

In [45]:
```
df_monthly_allocation #Monthly Allocation of funds in million USD
```

Out[45]:

	Print	TV	SEO	AdWords	Facebook	LinkedIn	Instagram	Snapchat	Twitter	Email
0	3.000000	0.0	0.0	1.333333	0.000000	0.000000	2.666667	0.0	0.000000	3.000000
1	3.000000	0.0	0.0	2.395500	3.000000	0.000000	0.000000	0.0	1.791000	0.000000
2	0.000000	0.0	0.0	3.000000	0.000000	3.000000	1.389648	0.0	3.000000	0.000000
3	0.000000	0.0	0.0	3.000000	0.000000	3.000000	3.000000	0.0	1.596856	0.000000
4	1.804100	0.0	0.0	0.000000	0.000000	0.000000	3.000000	0.0	3.000000	3.000000
5	3.000000	0.0	0.0	0.000000	0.000000	0.000000	3.000000	0.0	2.020172	3.000000
6	1.123777	0.0	0.0	3.000000	1.123777	0.000000	3.000000	0.0	3.000000	0.000000
7	3.000000	0.0	0.0	1.827294	0.000000	0.654588	0.000000	0.0	3.000000	3.000000
8	1.362933	0.0	0.0	3.000000	0.000000	3.000000	0.000000	0.0	3.000000	1.362933
9	0.000000	0.0	0.0	3.000000	0.000000	3.000000	3.000000	0.0	0.000000	2.955475
10	3.000000	0.0	0.0	[illegible]	0.000000	[illegible]	3.000000	0.0	0.000000	3.000000

Part 8

n [51]:

```
df_monthly_allocation_copy.diff() # a better way to calculate the difference, should've
```

ut[51]:

	Print	TV	SEO	AdWords	Facebook	LinkedIn	Instagram	Snapchat	Twitter	Email
0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
1	0.000000	0.0	0.0	1.062167	3.000000	0.000000	-2.666667	0.0	1.791000	-3.000000
2	-3.000000	0.0	0.0	0.604500	-3.000000	3.000000	1.389648	0.0	1.209000	0.000000
3	0.000000	0.0	0.0	0.000000	0.000000	0.000000	1.610352	0.0	-1.403144	0.000000
4	1.804100	0.0	0.0	-3.000000	0.000000	-3.000000	0.000000	0.0	1.403144	3.000000
5	1.195900	0.0	0.0	0.000000	0.000000	0.000000	0.000000	0.0	-0.979828	0.000000
6	-1.876223	0.0	0.0	3.000000	1.123777	0.000000	0.000000	0.0	0.979828	-3.000000
7	1.876223	0.0	0.0	-1.172706	-1.123777	0.654588	-3.000000	0.0	0.000000	3.000000
8	-1.637067	0.0	0.0	1.172706	0.000000	2.345412	0.000000	0.0	0.000000	-1.637067
9	-1.362933	0.0	0.0	0.000000	0.000000	0.000000	3.000000	0.0	-3.000000	1.592543
10	3.000000	0.0	0.0	-0.943579	0.000000	-1.887158	0.000000	0.0	0.000000	0.044525
11	0.000000	3.0	0.0	-1.628470	3.000000	-1.112842	-3.000000	0.0	0.000000	0.000000