Google Ads hourly Analysis

Date : 19 June 23

Project Start Date - End Date	• Start Date – 19-06 -2023 • End Date – 20-06 2023
Objectives	• To analyses how many people who clicked on the advertisement and highly interested to enroll in our course. • General exploratory analyses. • General descriptive analyses.
Milestones accomplished the week of Start Date - End Date:	• Descriptive analyses • Exploratory analyses • Classification of data with respect to term.

Contact Information

This project is performed for educational purpose of under the guidance of Siddhivinayak Sir .

Project Manager

Name : Siddhivinayak Phulwadkar
Mobile: 9028965955
Email: siddhivinayakphulwadkar@gmail.com

Student Name

Name : Shubham Kalyan Mane
Mobile: 7972390175
Email: shubhammane656@gmail.com

Project Abstract

This dataset is all about showing Advertisement to clients and to enroll for the course . The main object was to understand at what time students are clicking on our ads and getting enrolled our course or to interested to buy courses. As we are looking for at which preferred time in a day. where we can do marketing and we will get sales definitely. For this dataset we have applied Decision Tree algorithm and performed exploratory and descriptive analysis.

Google Ads Hourly Analysis

#Importing libraries

```
import numpy as np
import matplotlib.pyplot as plt # this library is used to represent data in graphical form
import pandas as pd
```

#Importing dataset

```
data=pd.read_excel("C:/Users/Shubham/Desktop/shubham/Google ads hourly analysis 19th june.xlsx")
```

```
data
```

	Sr no	Impressions	Clicks	Cost	CTR	CPC	Cold Leads	Warm Leads	Hot Leads
0	00:00:00	5941	356	594	0.0600	1.666667	11	5.0	2
1	00:30:00	4511	180	451	0.0400	2.500000	9	4.0	2
2	01:00:00	3378	101	338	0.0300	3.333333	6	3.0	1
3	01:30:00	652	26	65	0.0400	2.500000	1	NaN	0
4	02:00:00	421	8	42	0.0200	5.000000	0	0.0	0
5	02:30:00	110	2	11	0.0200	5.000000	0	0.0	0
6	03:00:00	56	1	6	0.0200	5.000000	0	0.0	0
7	03:30:00	42	1	4	0.0120	8.333333	0	0.0	0
8	04:00:00	3	0	0	0.0110	9.090909	0	0.0	0
9	04:30:00	8	0	1	0.0190	5.263158	0	0.0	0
10	05:00:00	95	1	10	0.0140	7.142857	0	0.0	0
11	05:30:00	64	1	6	0.0200	5.000000	0	0.0	0
12	06:00:00	26	0	3	0.0020	50.000000	0	0.0	0
13	06:30:00	193	4	19	0.0200	5.000000	0	0.0	0
14	07:00:00	236	5	24	0.0220	4.545455	0	0.0	0
15	07:30:00	463	23	46	0.0500	2.000000	1	0.0	0
16	08:00:00	896	54	90	0.0600	1.666667	2	1.0	0
17	08:30:00	486	34	49	0.0700	1.428571	1	0.0	0
18	09:00:00	785	71	79	0.0900	1.111111	1	0.0	0
19	09:30:00	1245	149	125	0.1200	0.833333	2	1.0	0

Preprocessing the dataset

```
file=data.drop(["Sr no","CTR","CPC","Warm Leads"],axis =1)
```

```
file.head()
```

	Impressions	Clicks	Cost	Cold Leads	Hot Leads
0	5941	356	594	11	2
1	4511	180	451	9	2
2	3378	101	338	6	1
3	652	26	65	1	0
4	421	8	42	0	0

```
data.isnull().sum()
```

```
Sr no           0
Impressions     0
Clicks          0
Cost            0
CTR             0
CPC             0
Cold Leads      0
Warm Leads      1
Hot Leads       0
dtype: int64
```

```
dataset=file.fillna({"Warm Leads":0.0})
```

```
dataset.isnull().sum()
```

```
Impressions     0
Clicks          0
Cost            0
Cold Leads      0
Hot Leads       0
dtype: int64
```

```
dataset
```

	Impressions	Clicks	Cost	Cold Leads	Hot Leads
0	5941	356	594	11	2
1	4511	180	451	9	2
2	3378	101	338	6	1
3	652	26	65	1	0
4	421	8	42	0	0
5	110	2	11	0	0
6	56	1	6	0	0
7	42	1	4	0	0
8	3	0	0	0	0
9	8	0	1	0	0
10	95	1	10	0	0

#Descriptive Analysis

```
# Discriptive Analysis
```

```
dataset.sum()
```

```
Impressions     200522
Clicks           31584
Cost             20057
Cold Leads         392
Hot Leads           73
dtype: int64
```

```
# Impressions are indicating total visiblity of Ads on 19th june 23
# Total Impression are 200522
# Clicks are indicating futher intrested audience for our Ads
# total Clicks are 31584
# cost indicates cost per cilck and impression for ads on 19th june 23
# Total cost are 20057
# Cold leads are indicating  little to no interest in your brand.
#Total cold Leads are 392.
# hot leads are indicating those who are highly intrested and ready to buy your brand.
# Total Hot Leads are 73.
```

```
dataset.mean()
```

```
Impressions     4177.541667
Clicks           658.000000
Cost             417.854167
Cold Leads         8.166667
Hot Leads          1.520833
dtype: float64
```

```
# The Average no of Impression are 4177.541667
# The Average no of Clicks are  658.000000
# The Average no of Cost are 417.854167
# The Average no of Cold leads are 8.166667
# The Average no of Hot Leads are 1.520833
```

```
#Individul Ratio

# An Individual Ratio of Cold Leads and Warm Leads is 45.45%
# An Individual Ratio of Cold Leads and Hot Leads is 18.62%
```

```
X= dataset.iloc[:,:-1].values
y= dataset.iloc[:,-1].values
```

```
X
```

```
array([[ 5941,   356,   594,    11],
       [ 4511,   180,   451,     9],
       [ 3378,   101,   338,     6],
       [  652,    26,    65,     1],
       [  421,     8,    42,     0],
       [  110,     2,    11,     0],
       [   56,     1,     6,     0],
       [   42,     1,     4,     0],
       [    3,     0,     0,     0],
       [    8,     0,     1,     0],
       [   95,     1,    10,     0],
       [   64,     1,     6,     0],
       [   26,     0,     3,     0],
       [  193,     4,    19,     0],
       [  236,     5,    24,     0],
       [  463,    23,    46,     1],
       [  896,    54,    90,     2],
       [  486,    34,    49,     1],
       [  785,    71,    79,     1],
       [ 1245,   149,   125,     2],
       [ 1755,   228,   176,     3],
       [ 1865,   106,   187,     4],
       [ 2658,   399,   266,     5],
       [ 3255,   391,   326,     7],
       [ 3284,   427,   328,     6],
       [ 4651,   558,   465,     9],
       [ 4895,   685,   490,    10],
       [ 4752,   665,   475,     9],
       [ 4665,   606,   467,     9],
       [ 4958,   694,   496,    10],
       [ 5320,   745,   532,    10],
       [ 5968,   836,   597,    12],
       [ 5673,   794,   567,    11],
```

```
y
```

```
array([2, 2, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1,
       1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 1, 2, 3, 3, 4, 3, 3, 4, 4, 4, 3,
       3, 4, 4, 4], dtype=int64)
```

#Traning Model

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.20, random_state = 20)
```

```
print(X_train)
```

```
[[ 5941   356   594    11]
 [ 4651   558   465     9]
 [  486    34    49     1]
 [  236     5    24     0]
 [ 5320   745   532    10]
 [  785    71    79     1]
 [   95     1    10     0]
 [  652    26    65     1]
 [ 8968  1883   897    18]
 [  193     4    19     0]
 [ 7521  1203   752    15]
 [ 3378   101   338     6]
 [ 3284   427   328     6]
 [ 8452  1944   845    17]
 [  110     2    11     0]
 [ 9120  1277   912    18]
 [ 4958   694   496    10]
 [ 7863  1179   786    16]
 [   56     1     6     0]
 [  896    54    90     2]
 [ 1245   149   125     2]
 [ 4752   665   475     9]
 [ 4865  1216   487     9]
 [ 8945  1431   895    18]
 [ 1865   106   187     4]
 [ 5673   794   567    11]
```

```
print(y_train)
```

```
[2 1 0 0 2 0 0 0 3 0 3 1 1 4 0 4 2 3 0 0 0 2 1 4 1 2 2 0 1 0 0 0 4 2 2 0 2
 3]
```

```
print(X_test)
```

```
[[ 421    8   42    0]
 [   3    0    0    0]
 [3255  391  326    7]
 [9452 1512  945   19]
 [4511  180  451    9]
 [9632 1830  963   19]
 [8757 2277  876   17]
 [  26    0    3    0]
 [8444 1773  844   17]
 [9425 1414  943   19]]
```

```
print(y_test)
```

```
[0 0 1 4 2 4 3 0 3 4]
```

```
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)
```

```
print(X_train)
```

```
[[ 0.6377908  -0.37717199  0.63719669  0.53252426]
 [ 0.24284746 -0.04328061  0.2422121   0.22822468]
 [-1.03229908 -0.90941469 -1.03153667 -0.98897363]
 [-1.10883849 -0.95734959 -1.10808408 -1.14112342]
 [ 0.44766691  0.26581686  0.44735913  0.38037447]
 [-0.94075795 -0.84825636 -0.93967979 -0.98897363]
 [-1.15200672 -0.9639613  -1.15095062 -1.14112342]
 [-0.98147691 -0.92263811 -0.98254634 -0.98897363]
 [ 1.56452995  2.14684851  1.5649512   1.59757279]
 [-1.12200327 -0.95900252 -1.12339356 -1.14112342]
 [ 1.12151986  1.02285772  1.12097627  1.14112342]
 [-0.14689121 -0.79866854 -0.14664871 -0.22822468]
 [-0.17567003 -0.25981413 -0.17726767 -0.22822468]
 [ 1.40655262  2.2476771   1.40573261  1.445423  ]
 [-1.14741435 -0.96230837 -1.14788872 -1.14112342]
 [ 1.61106591  1.14517437  1.61087964  1.59757279]
 [ 0.33683785  0.18151755  0.33713087  0.38037447]
 [ 1.22622577  0.98318746  1.22508074  1.29327321]
 [-1.16394686 -0.9639613  -1.16319821 -1.14112342]
 [-0.90677445 -0.87635613 -0.90599893 -0.83682384]
 [-0.79992544 -0.71932801 -0.79883257 -0.83682384]
 [ 0.27376938  0.13358265  0.27283106  0.22822468]
 [ 0.30836519  1.04434578  0.30957381  0.22822468]
 [ 1.55748833  1.39972522  1.55882741  1.59757279]
 [-0.61010771 -0.7904039  -0.60899501 -0.53252426]
 [ 0.55574056  0.34681032  0.5545255   0.53252426]
 [ 0.83740558  0.77822443  0.83621993  0.83682384]
 [-1.16823307 -0.9639613  -1.169322   -1.14112342]
```

```
print(X_test)
```

```
[[-1.05219933 -0.95239081 -1.05296995 -1.14112342]
 [-1.18017322 -0.96561423 -1.18156958 -1.14112342]
 [-0.1845486  -0.31931952 -0.18339146 -0.07607489]
 [ 1.71271025  1.53361236  1.71192221  1.74972257]
 [ 0.19998539 -0.66808725  0.19934555  0.22822468]
 [ 1.76781862  2.05924335  1.76703634  1.74972257]
 [ 1.49993069  2.798102    1.50065138  1.445423  ]
 [-1.17313159 -0.96561423 -1.17238389 -1.14112342]
 [ 1.40410336  1.96502647  1.40267071  1.445423  ]
 [ 1.70444399  1.37162545  1.70579842  1.74972257]]
```

Decision Tree

```
from sklearn.tree import DecisionTreeClassifier
classifier = DecisionTreeClassifier(criterion = 'gini', random_state = 0)
classifier.fit(X_train, y_train)
```

```
DecisionTreeClassifier(random_state=0)
```

```
print(classifier.predict(sc.transform([[10230,1739,1023,20]])))
```

```
[4]
```

#Predicting the Test set results

```
y_pred = classifier.predict(X_test)
print(np.concatenate((y_pred.reshape(len(y_pred),1), y_test.reshape(len(y_test),1)),1))
```

```
[[0 0]
 [0 0]
 [1 1]
 [4 4]
 [1 2]
 [4 4]
 [4 3]
 [0 0]
 [4 3]
 [4 4]]
```

#Making the Confusion Matrix

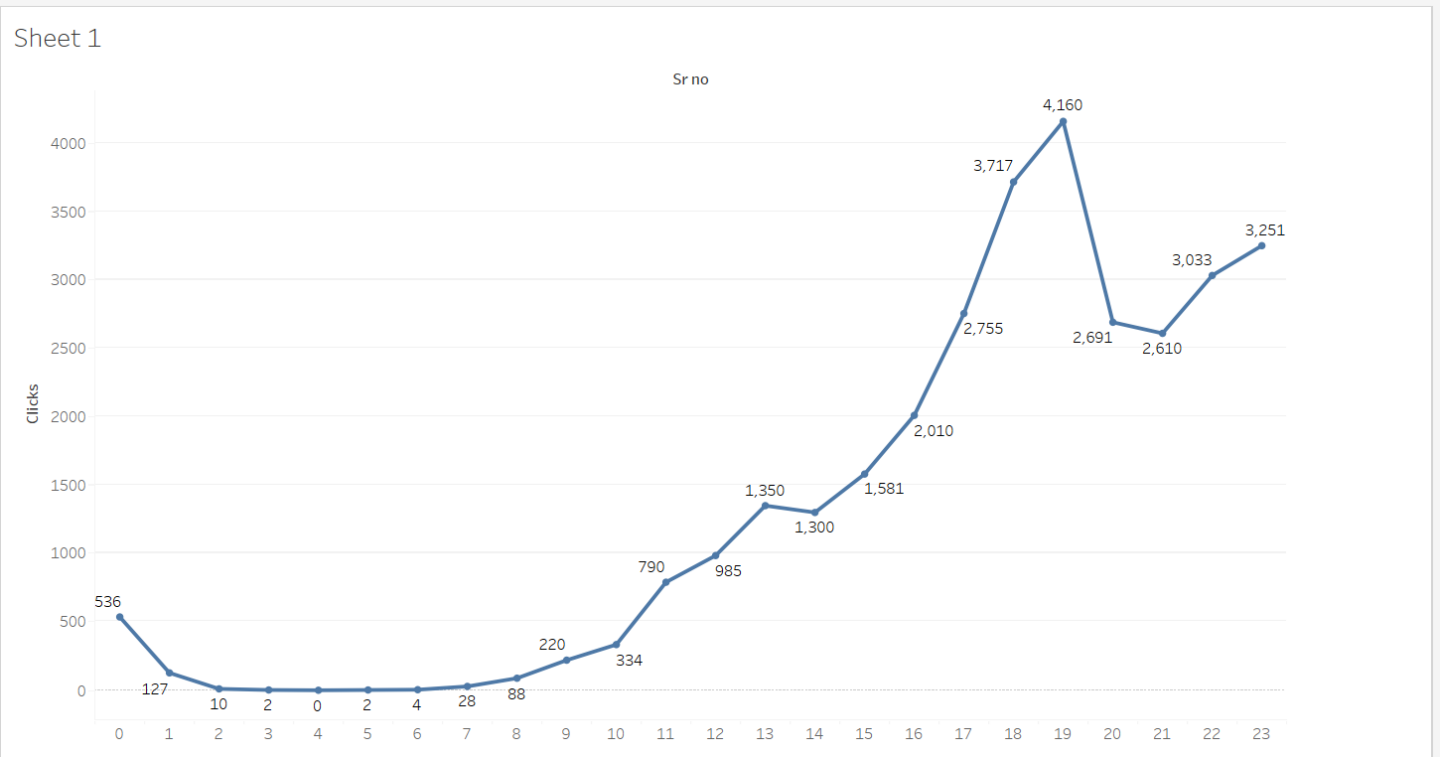
```
from sklearn.metrics import confusion_matrix, accuracy_score
cm = confusion_matrix(y_test, y_pred)
print(cm)
accuracy_score(y_test, y_pred)
```

```
[[3 0 0 0 0]
 [0 1 0 0 0]
 [0 1 0 0 0]
 [0 0 0 0 2]
 [0 0 0 0 3]]

0.7
```

```
#Accuracy is 70%
```

Visualization



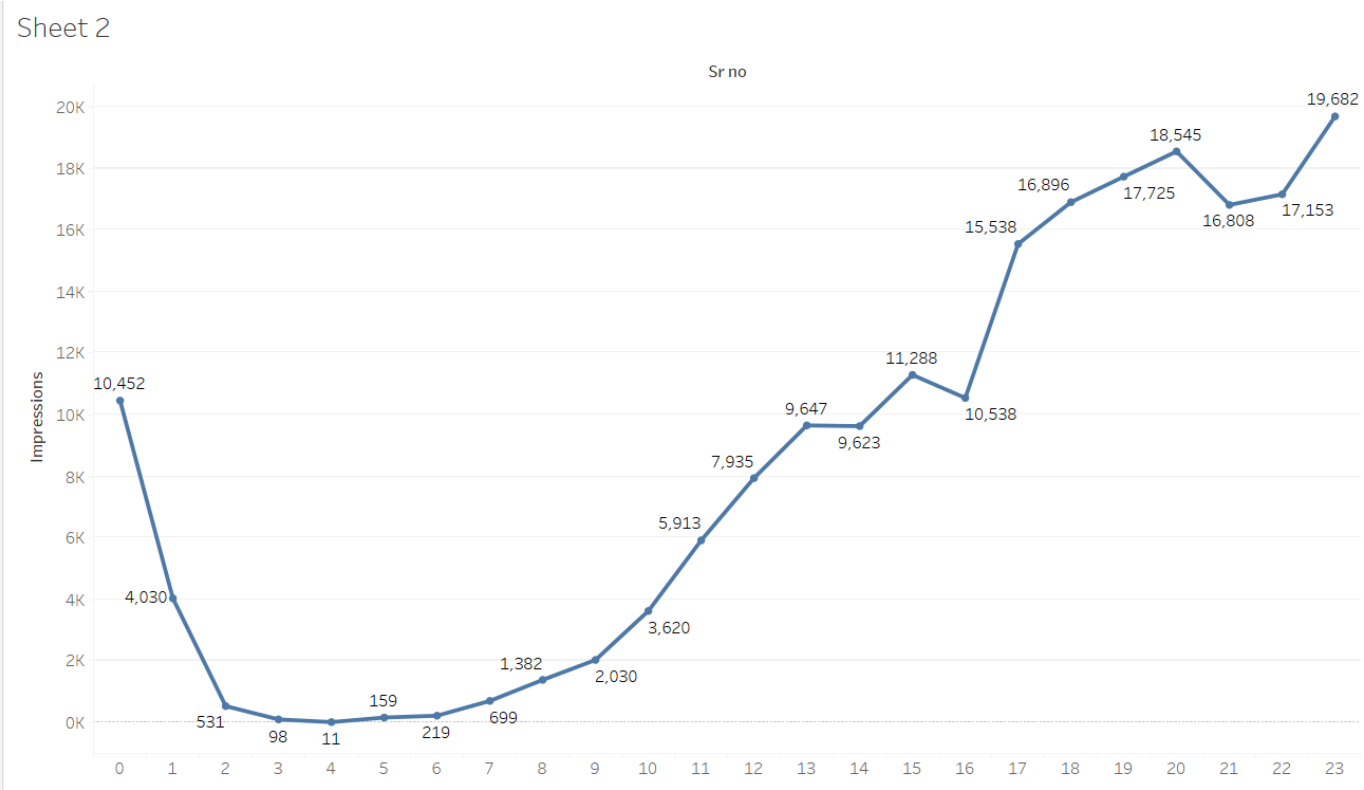
Insights of the Clicks Graph

- Clicks are slightly high from 12:00 am to 1:00 am. because lot of people are searching for new thing and they are online as they are free.

- Then from 1:30 am to 9:30 am clicks are down as most of the people are sleeping

- From 10:00 am to 6:00 pm clicks are getting high than previous time.

- Then clicks are going to increases from 6:00 pm to 11:30 pm as people will have free time to be online.



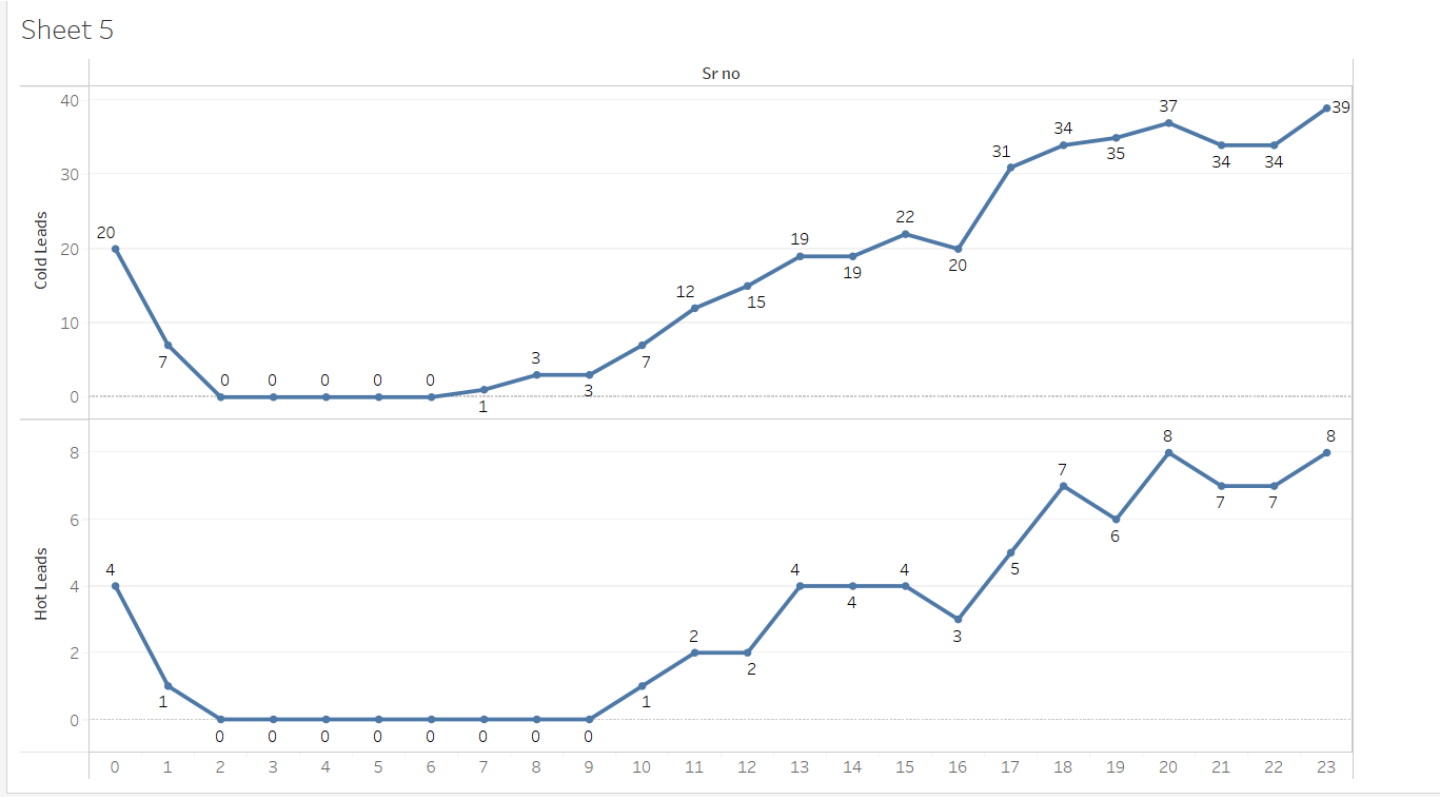
Insights of the Impressions Graph

- Impressions are going very high at the night from 12:00 am to 1:00 am.

- Then Impressions starts falling down after 1:30 am (1:30 am to 6:00 am) because majority of the people are not online.

- 6:30 am to 11:30 am impressions are getting slightly high and starts to increases because on that time lot of people are online on sites.

- Then from 12:00 pm to 11:30 pm impressions are increasing high and from the evening 6:30 pm to 11:30 pm impressions are very high, because more no of people is free and online.



Insights of Cold Lead and Hot Lead Graph

- Very few people are showing interest in that particular Ad from 12:00 am to 1:00 am

- Then after 1:00 am not even one candidate is showing interest in that Ads from 1:30 am to 7:00 am because lot of people are sleeping.

- After 7:30 am only 1, 2 candidates are showings interest in particular time slots from 7:30 am to 10:00 am.

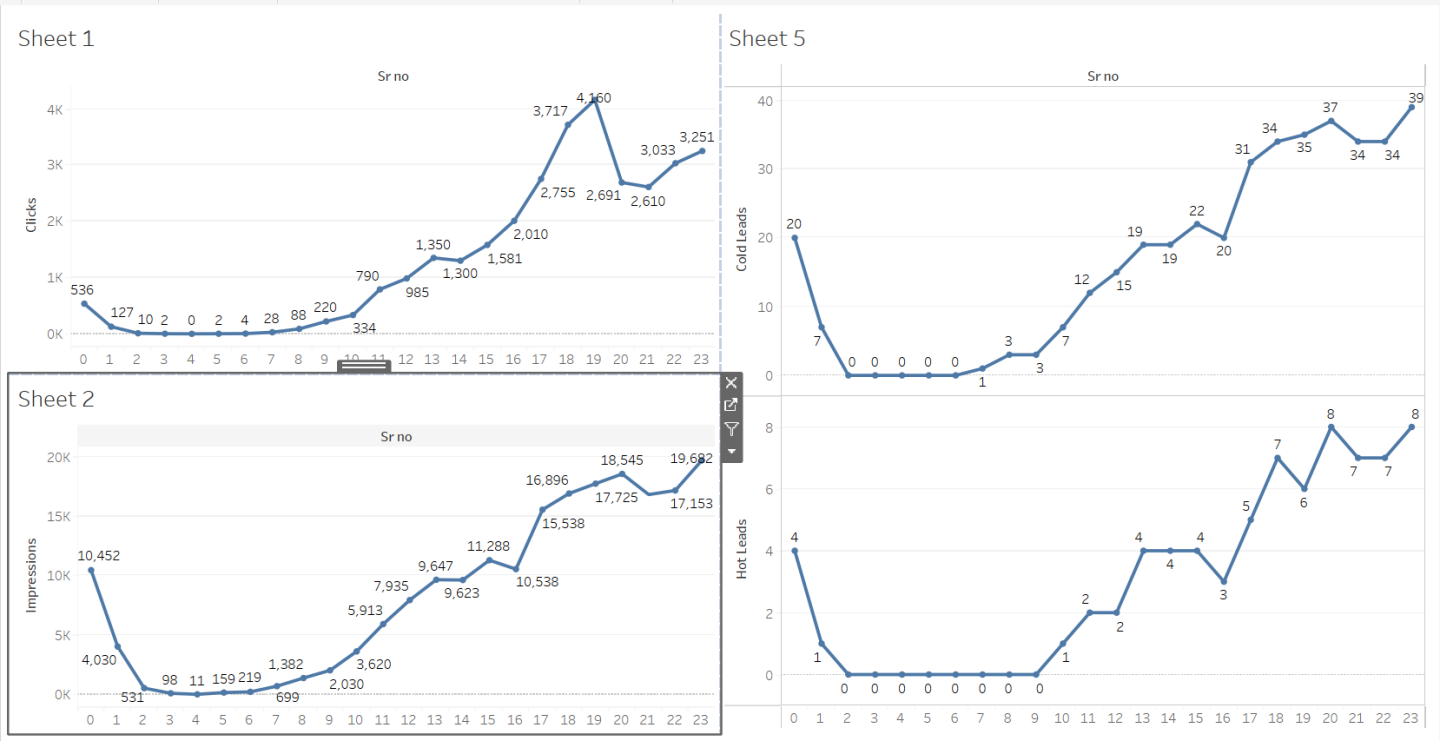
- from 10:30 am to 6:30 pm leads get high than previous time slots of leads

- From 12:00 am to 1:00 am few people are showing interest and they like to purchase that service (they like to enroll in that Ad)

- Then from 1:30 am to 10:00 am no one showing interest in that Ads.

• Then from 10:30 am to 5:00 pm few people like 1,2 people are showing interest and they like to purchase that service (they like to enroll in that Ad)

• From 5:30 pm to 11:30 pm 3,4 people are showing interest and they like to purchase that service.



Conclusion

```
#Conclusion

# There are 30 no preffered time slot of 30 min in entire day where it is preffereable to show the Advertisement
# In which 5:30pm to 11:30pm from evening to night company has generated more number of Hot leads
#so we conclude that we can show Ads to people in that time slot as more no of people use their mobile phones and they are online
# so this is a good time slot for showing our Ads.
# The maxmium cost we used to show our Ads is in 7:30pm to 9:00pm time slot where we get 4 Hot Leads in Each time slot.
# In htis particular time Slot company has get more no of Impression, Click, Cold Leads, Warm Leads and HOt Leads.
# There are 18 numbers of time slot where company dosent get Hot Leads.
# cost spent on 18 numbers Where we dosent get any Hot Leads is 756 rs.
# And cost spent on 30 numbers time slot where we get Hot Leads is 19301 rs.
# The Total number of Cost spend by the company to generate Hot Leads is 20057 rs.
# so 96.24% of cost spent by company make profit by getting Hot Leads.
```

```
# So we suggest that we can reduce the cost by spending in the 1:00am to 10:00am time slot.
```