Google Ads hourly Analysis

Date : **21th June 23**

Project Start Date - End Date	● Start Date – 21-06 -2023 ● End Date – 22-06- 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

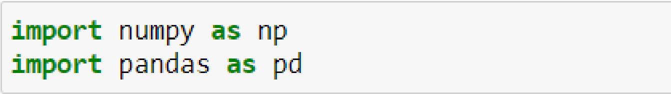
Student Name
Name : Shubham Kalyan Mane
Mobile: 7972390175
Email: shubhammane656@gmail.com

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.

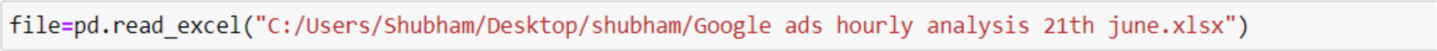
Google Ads Hourly Analysis

Importing libraries

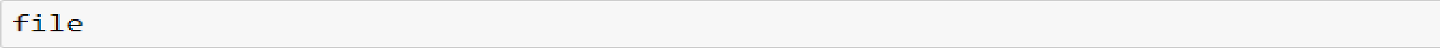


```
import numpy as np
import pandas as pd
```

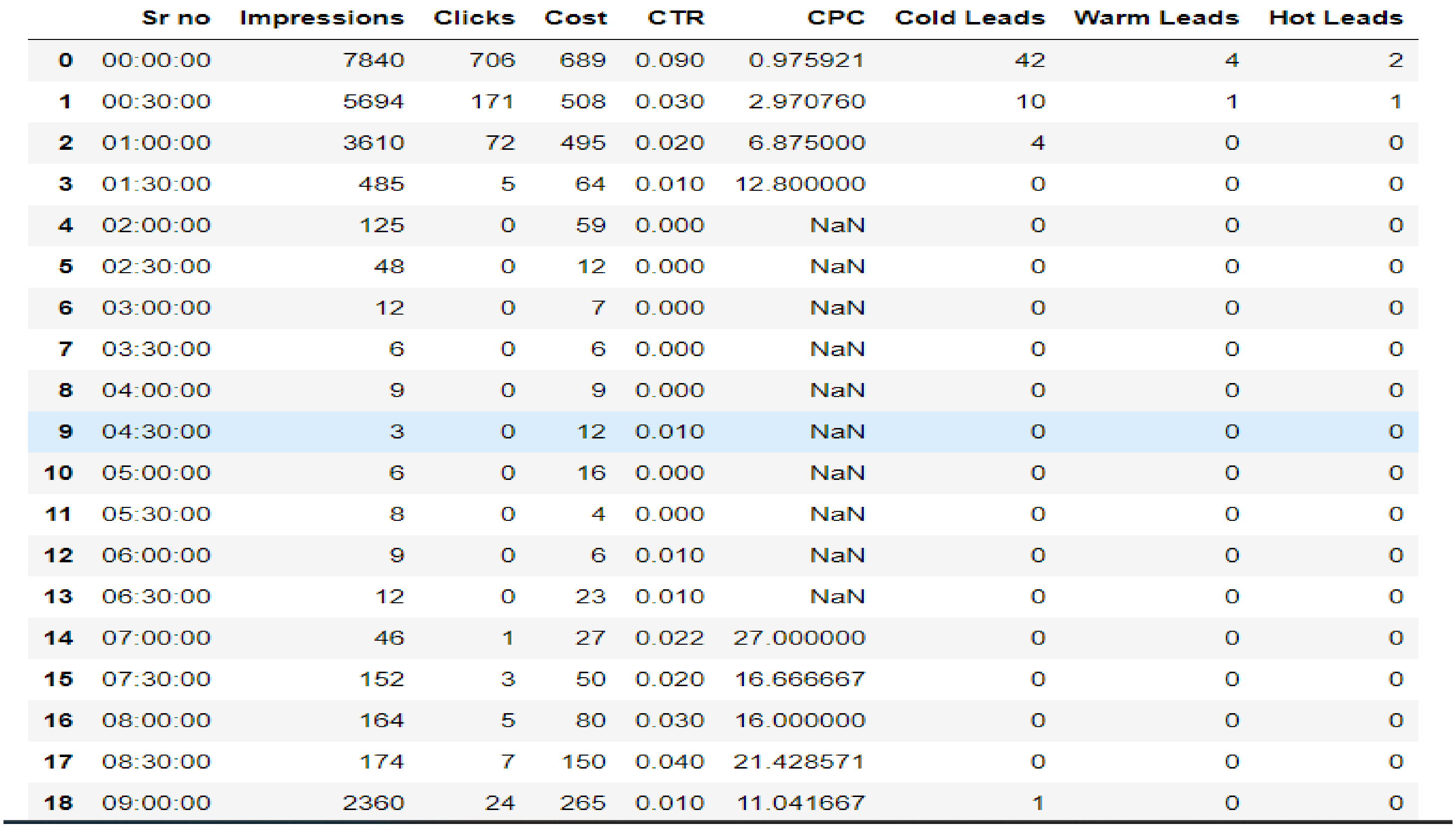
Importing dataset



```
file=pd.read_excel("C:/Users/Shubham/Desktop/shubham/Google ads hourly analysis 21th june.xlsx")
```

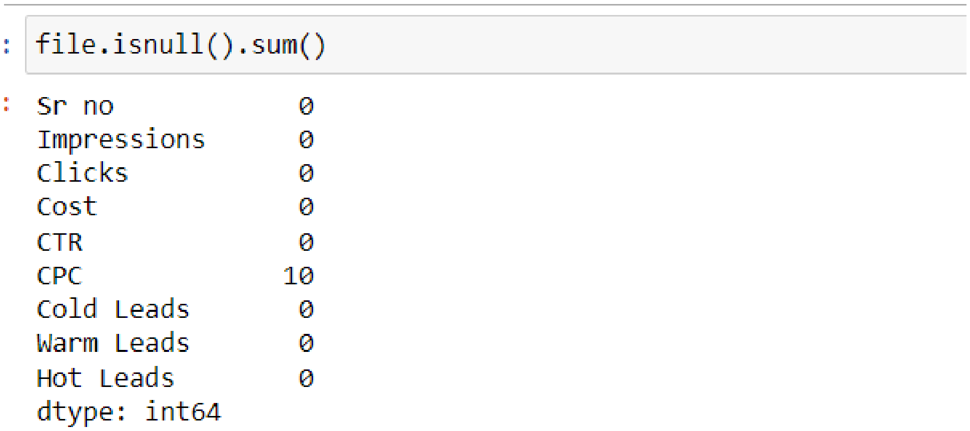


```
file
```



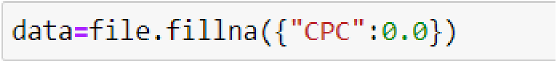
	Sr no	Impressions	Clicks	Cost	CTR	CPC	Cold Leads	Warm Leads	Hot Leads
0	00:00:00	7840	706	689	0.090	0.975921	42	4	2
1	00:30:00	5694	171	508	0.030	2.970760	10	1	1
2	01:00:00	3610	72	495	0.020	6.875000	4	0	0
3	01:30:00	485	5	64	0.010	12.800000	0	0	0
4	02:00:00	125	0	59	0.000	NaN	0	0	0
5	02:30:00	48	0	12	0.000	NaN	0	0	0
6	03:00:00	12	0	7	0.000	NaN	0	0	0
7	03:30:00	6	0	6	0.000	NaN	0	0	0
8	04:00:00	9	0	9	0.000	NaN	0	0	0
9	04:30:00	3	0	12	0.010	NaN	0	0	0
10	05:00:00	6	0	16	0.000	NaN	0	0	0
11	05:30:00	8	0	4	0.000	NaN	0	0	0
12	06:00:00	9	0	6	0.010	NaN	0	0	0
13	06:30:00	12	0	23	0.010	NaN	0	0	0
14	07:00:00	46	1	27	0.022	27.000000	0	0	0
15	07:30:00	152	3	50	0.020	16.666667	0	0	0
16	08:00:00	164	5	80	0.030	16.000000	0	0	0
17	08:30:00	174	7	150	0.040	21.428571	0	0	0
18	09:00:00	2360	24	265	0.010	11.041667	1	0	0

# Preprocessing the dataset

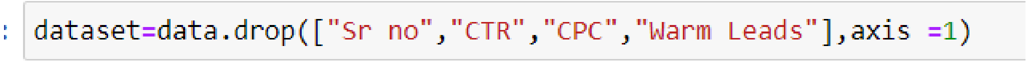


```
file.isnull().sum()
```

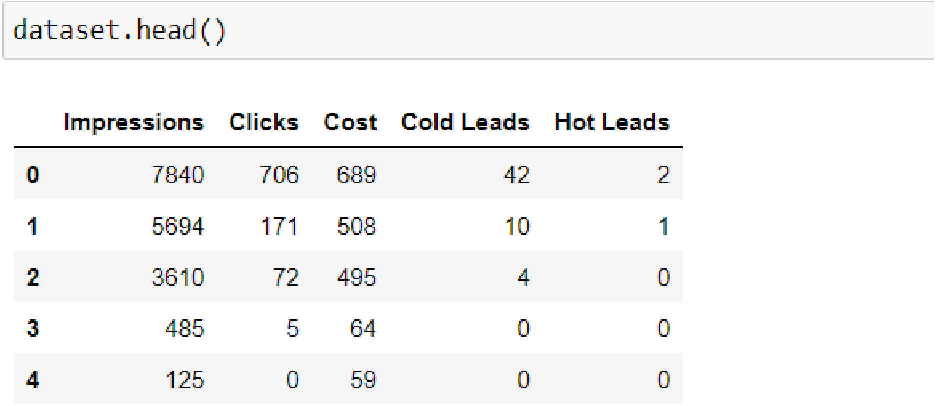
```
Sr no           0
Impressions     0
Clicks          0
Cost            0
CTR             0
CPC            10
Cold Leads      0
Warm Leads      0
Hot Leads       0
dtype: int64
```



```
data=file.fillna({"CPC":0.0})
```

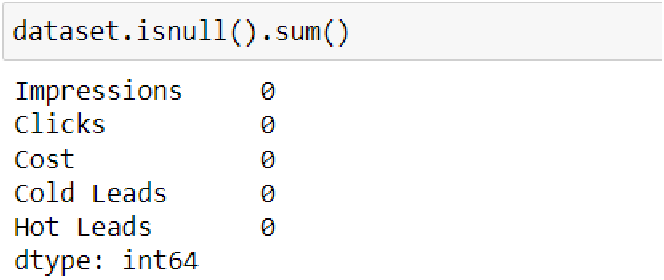


```
dataset=data.drop(["Sr no","CTR","CPC","Warm Leads"],axis =1)
```



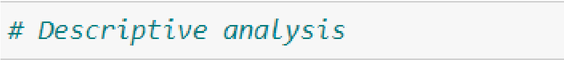
```
dataset.head()
```

	Impressions	Clicks	Cost	Cold Leads	Hot Leads
0	7840	706	689	42	2
1	5694	171	508	10	1
2	3610	72	495	4	0
3	485	5	64	0	0
4	125	0	59	0	0

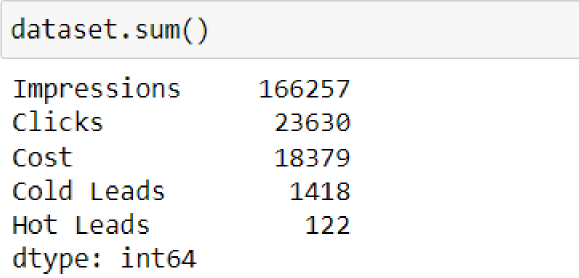


```
dataset.isnull().sum()
```

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

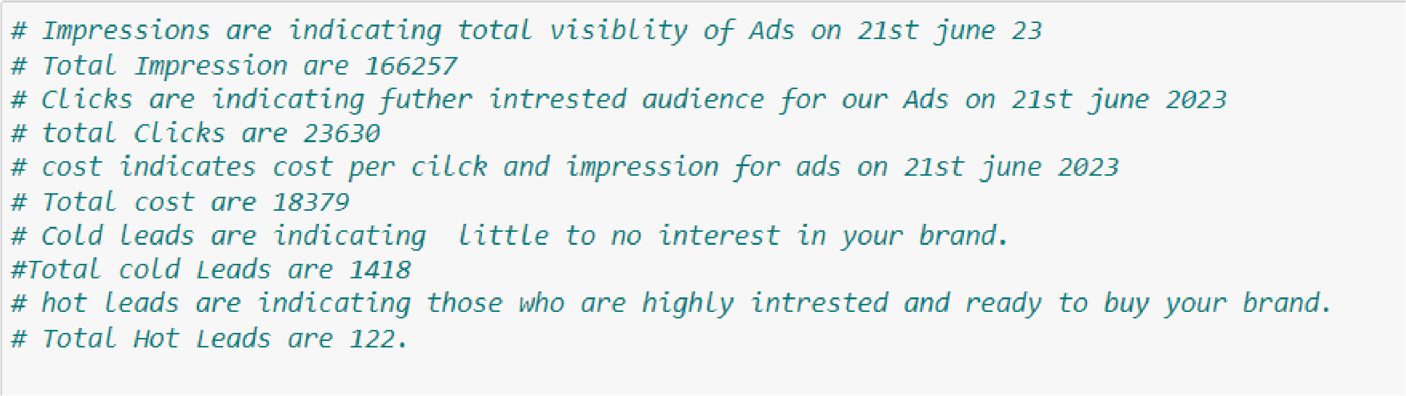


```
# Descriptive analysis
```

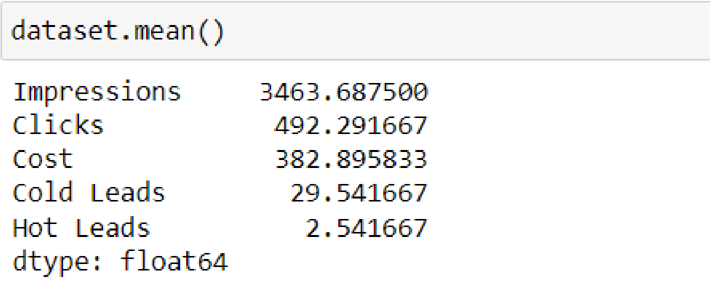


```
dataset.sum()
```

```
Impressions     166257
Clicks           23630
Cost             18379
Cold Leads        1418
Hot Leads          122
dtype: int64
```

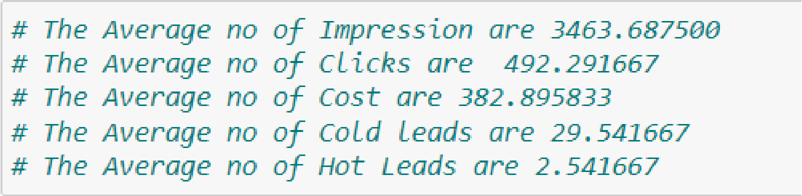


```
# Impressions are indicating total visiblity of Ads on 21st june 23
# Total Impression are 166257
# Clicks are indicating futher intrested audience for our Ads on 21st june 2023
# total Clicks are 23630
# cost indicates cost per cilck and impression for ads on 21st june 2023
# Total cost are 18379
# Cold Leads are indicating  little to no interest in your brand.
#Total cold Leads are 1418
# hot Leads are indicating those who are highly intrested and ready to buy your brand.
# Total Hot Leads are 122.
```

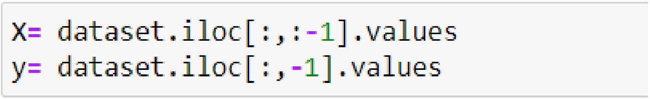


```
dataset.mean()
```

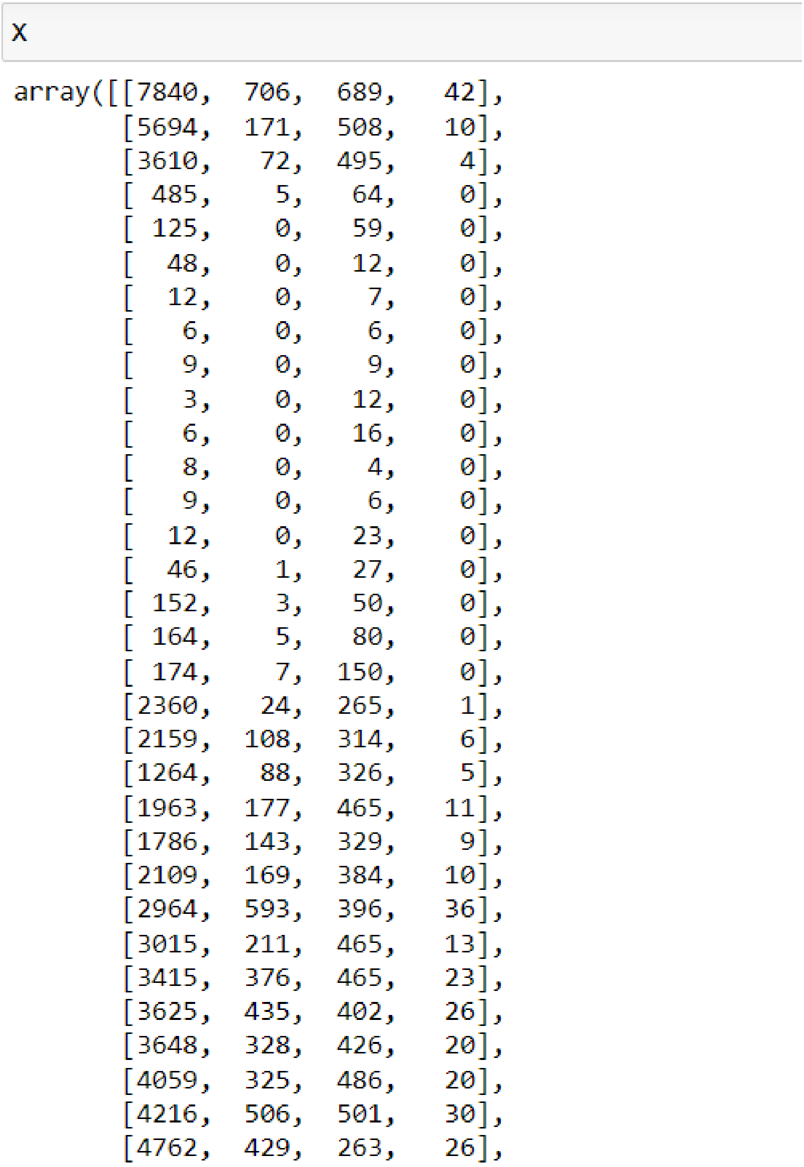
```
Impressions     3463.687500
Clicks           492.291667
Cost             382.895833
Cold Leads        29.541667
Hot Leads          2.541667
dtype: float64
```



```
# The Average no of Impression are 3463.687500
# The Average no of Clicks are  492.291667
# The Average no of Cost are 382.895833
# The Average no of Cold leads are 29.541667
# The Average no of Hot Leads are 2.541667
```

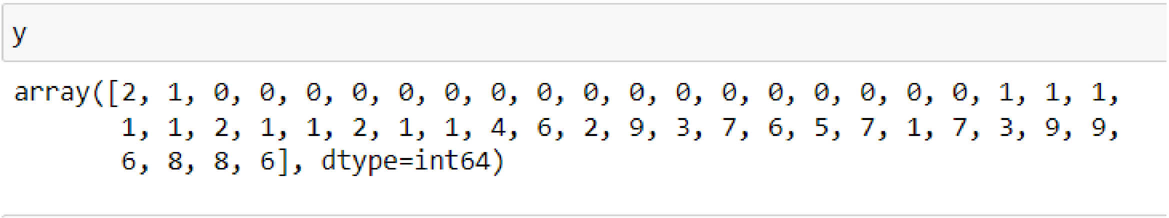


```
X= dataset.iloc[:,:-1].values
y= dataset.iloc[:,-1].values
```



```
X
```

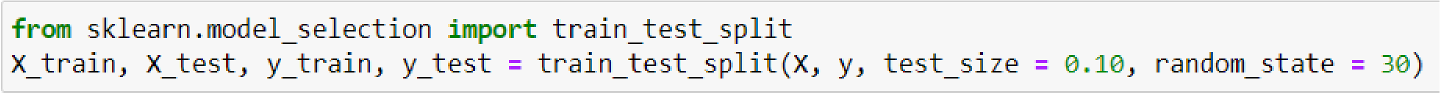
```
array([[7840,  706,  689,   42],
       [5694,  171,  508,   10],
       [3610,   72,  495,    4],
       [ 485,    5,   64,    0],
       [ 125,    0,   59,    0],
       [  48,    0,   12,    0],
       [  12,    0,    7,    0],
       [   6,    0,    6,    0],
       [   9,    0,    9,    0],
       [   3,    0,   12,    0],
       [   6,    0,   16,    0],
       [   8,    0,    4,    0],
       [   9,    0,    6,    0],
       [  12,    0,   23,    0],
       [  46,    1,   27,    0],
       [ 152,    3,   50,    0],
       [ 164,    5,   80,    0],
       [ 174,    7,  150,    0],
       [2360,   24,  265,    1],
       [2159,  108,  314,    6],
       [1264,   88,  326,    5],
       [1963,  177,  465,   11],
       [1786,  143,  329,    9],
       [2109,  169,  384,   10],
       [2964,  593,  396,   36],
       [3015,  211,  465,   13],
       [3415,  376,  465,   23],
       [3625,  435,  402,   26],
       [3648,  328,  426,   20],
       [4059,  325,  486,   20],
       [4216,  506,  501,   30],
       [4762,  429,  263,   26],
```



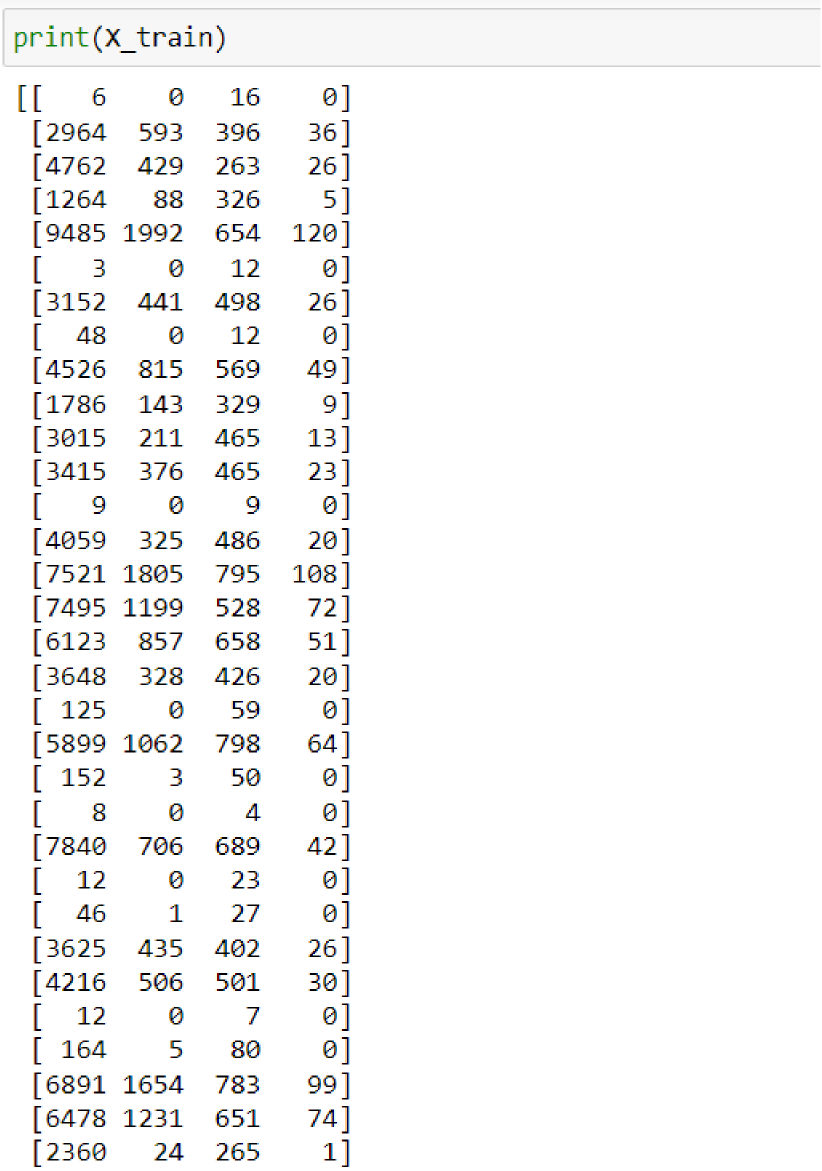
```
y
```

```
array([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, 1, 1, 2, 1, 1, 4, 6, 2, 9, 3, 7, 6, 5, 7, 1, 7, 3, 9, 9,
       6, 8, 8, 6], dtype=int64)
```

#Traning Model

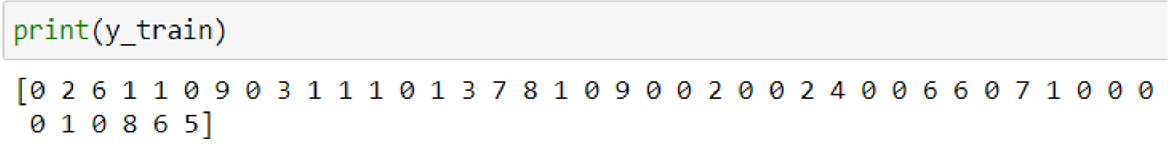


```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.10, random_state = 30)
```



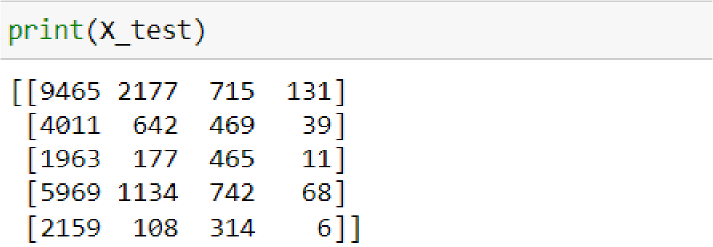
```
print(X_train)
```

```
[[   6    0   16    0]
 [2964  593  396   36]
 [4762  429  263   26]
 [1264   88  326    5]
 [9485 1992  654  120]
 [   3    0   12    0]
 [3152  441  498   26]
 [  48    0   12    0]
 [4526  815  569   49]
 [1786  143  329    9]
 [3015  211  465   13]
 [3415  376  465   23]
 [   9    0    9    0]
 [4059  325  486   20]
 [7521 1805  795  108]
 [7495 1199  528   72]
 [6123  857  658   51]
 [3648  328  426   20]
 [ 125    0   59    0]
 [5899 1062  798   64]
 [ 152    3   50    0]
 [   8    0    4    0]
 [7840  706  689   42]
 [  12    0   23    0]
 [  46    1   27    0]
 [3625  435  402   26]
 [4216  506  501   30]
 [  12    0    7    0]
 [ 164    5   80    0]
 [6891 1654  783   99]
 [6478 1231  651   74]
 [2360   24  265    1]
```



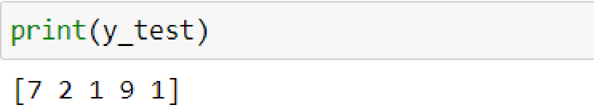
```
print(y_train)
```

```
[0 2 6 1 1 0 9 0 3 1 1 1 0 1 3 7 8 1 0 9 0 0 2 0 0 2 4 0 0 6 6 0 7 1 0 0 0
 0 1 0 8 6 5]
```



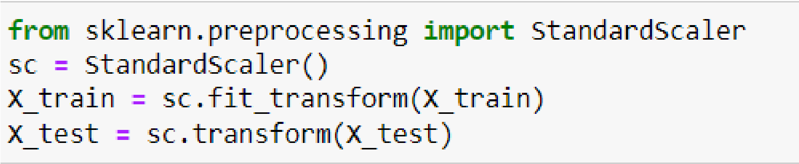
```
print(X_test)
```

```
[[9465 2177  715  131]
 [4011  642  469   39]
 [1963  177  465   11]
 [5969 1134  742   68]
 [2159  108  314    6]]
```

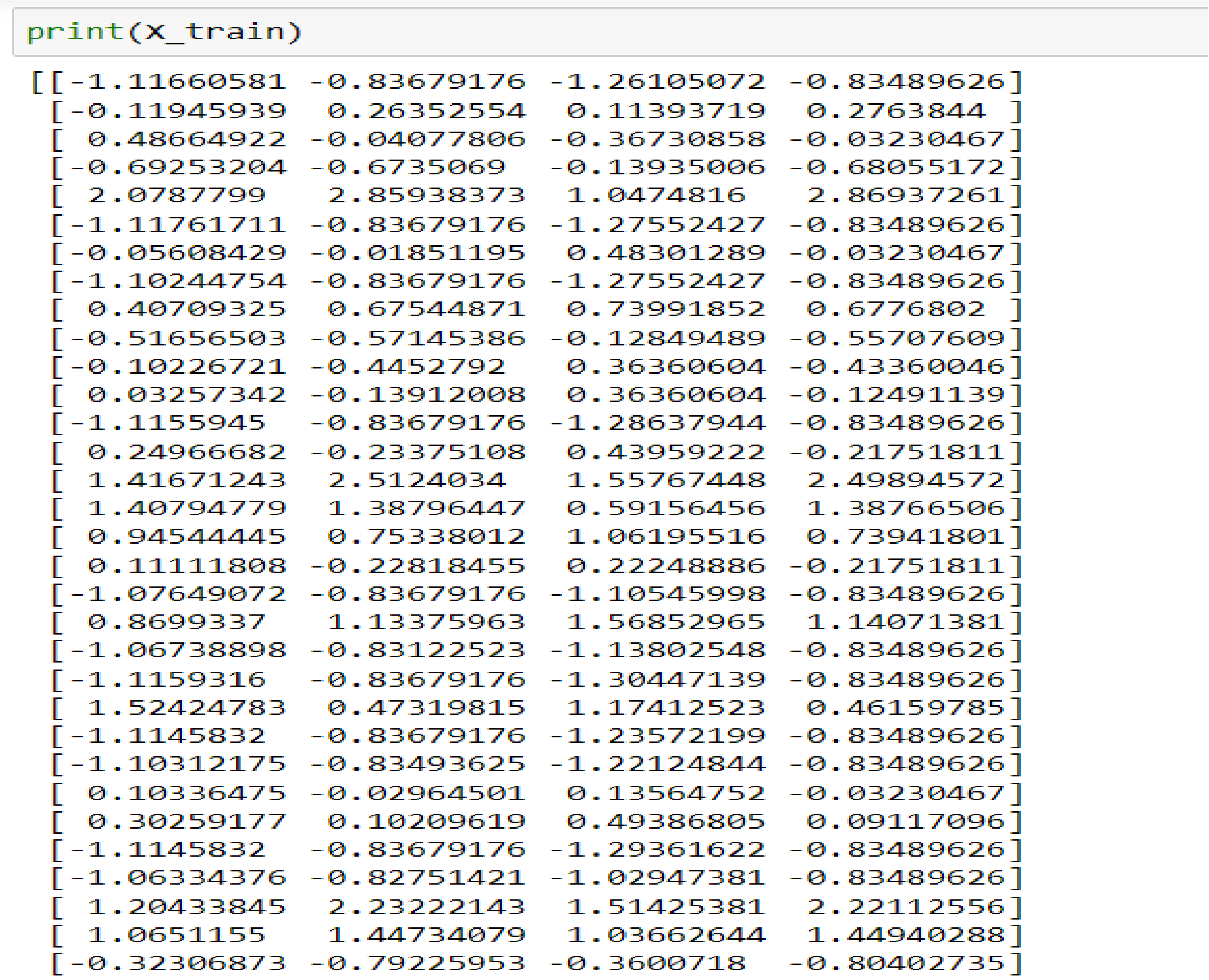


```
print(y_test)
```

```
[7 2 1 9 1]
```

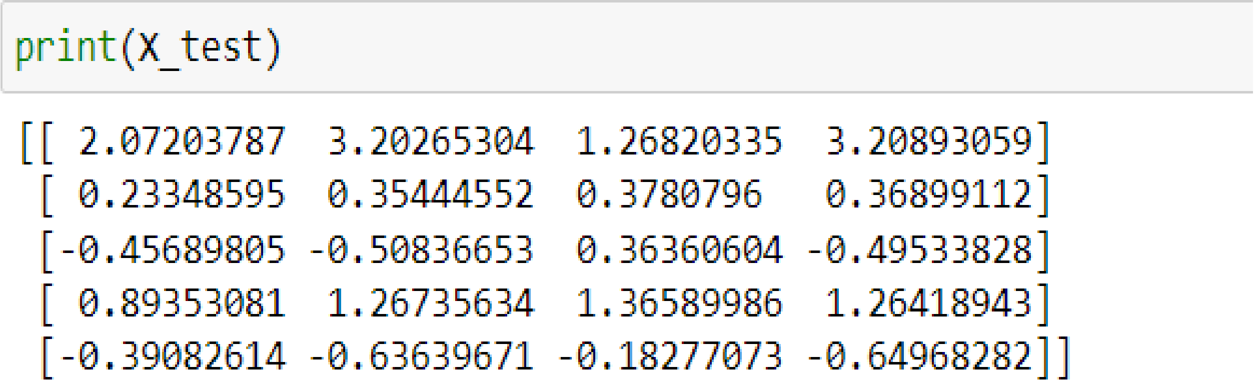


```
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)
```



```
print(X_train)
```

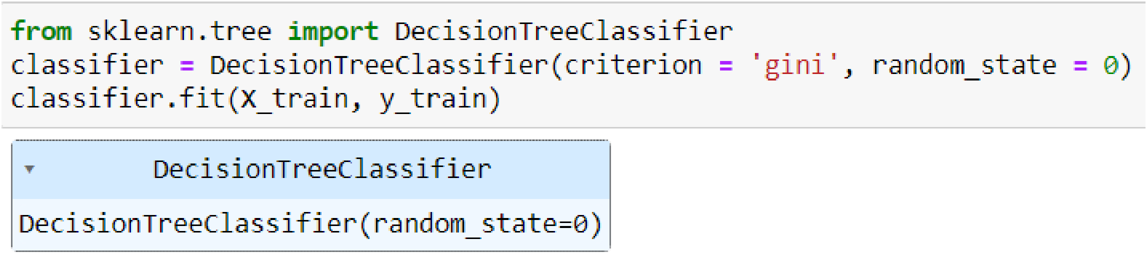
```
[[-1.11660581 -0.83679176 -1.26105072 -0.83489626]
 [-0.11945939  0.26352554  0.11393719  0.2763844 ]
 [ 0.48664922 -0.04077806 -0.36730858 -0.03230467]
 [-0.69253204 -0.6735069  -0.13935006 -0.68055172]
 [ 2.0787799   2.85938373  1.0474816   2.86937261]
 [-1.11761711 -0.83679176 -1.27552427 -0.83489626]
 [-0.05608429 -0.01851195  0.48301289 -0.03230467]
 [-1.10244754 -0.83679176 -1.27552427 -0.83489626]
 [ 0.40709325  0.67544871  0.73991852  0.6776802 ]
 [-0.51656503 -0.57145386 -0.12849489 -0.55707609]
 [-0.10226721 -0.4452792   0.36360604 -0.43360046]
 [ 0.03257342 -0.13912008  0.36360604 -0.12491139]
 [-1.1155945  -0.83679176 -1.28637944 -0.83489626]
 [ 0.24966682 -0.23375108  0.43959222 -0.21751811]
 [ 1.41671243  2.5124034   1.55767448  2.49894572]
 [ 1.40794779  1.38796447  0.59156456  1.38766506]
 [ 0.94544445  0.75338012  1.06195516  0.73941801]
 [ 0.11111808 -0.22818455  0.22248886 -0.21751811]
 [-1.07649072 -0.83679176 -1.10545998 -0.83489626]
 [ 0.8699337   1.13375963  1.56852965  1.14071381]
 [-1.06738898 -0.83122523 -1.13802548 -0.83489626]
 [-1.1159316  -0.83679176 -1.30447139 -0.83489626]
 [ 1.52424783  0.47319815  1.17412523  0.46159785]
 [-1.1145832  -0.83679176 -1.23572199 -0.83489626]
 [-1.10312175 -0.83493625 -1.22124844 -0.83489626]
 [ 0.10336475 -0.02964501  0.13564752 -0.03230467]
 [ 0.30259177  0.10209619  0.49386805  0.09117096]
 [-1.1145832  -0.83679176 -1.29361622 -0.83489626]
 [-1.06334376 -0.82751421 -1.02947381 -0.83489626]
 [ 1.20433845  2.23222143  1.51425381  2.22112556]
 [ 1.0651155   1.44734079  1.03662644  1.44940288]
 [-0.32306873 -0.79225953 -0.3600718  -0.80402735]
```



```
print(X_test)
```

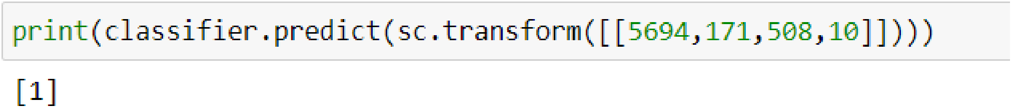
```
[[ 2.07203787  3.20265304  1.26820335  3.20893059]
 [ 0.23348595  0.35444552  0.3780796   0.36899112]
 [-0.45689805 -0.50836653  0.36360604 -0.49533828]
 [ 0.89353081  1.26735634  1.36589986  1.26418943]
 [-0.39082614 -0.63639671 -0.18277073 -0.64968282]]
```

# Decision Tree



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

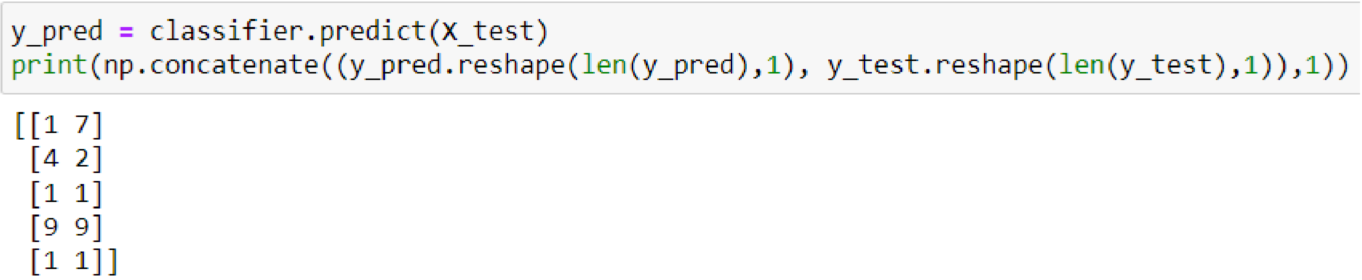
```
▾        DecisionTreeClassifier
DecisionTreeClassifier(random_state=0)
```



```
print(classifier.predict(sc.transform([[5694,171,508,10]])))
```

```
[1]
```

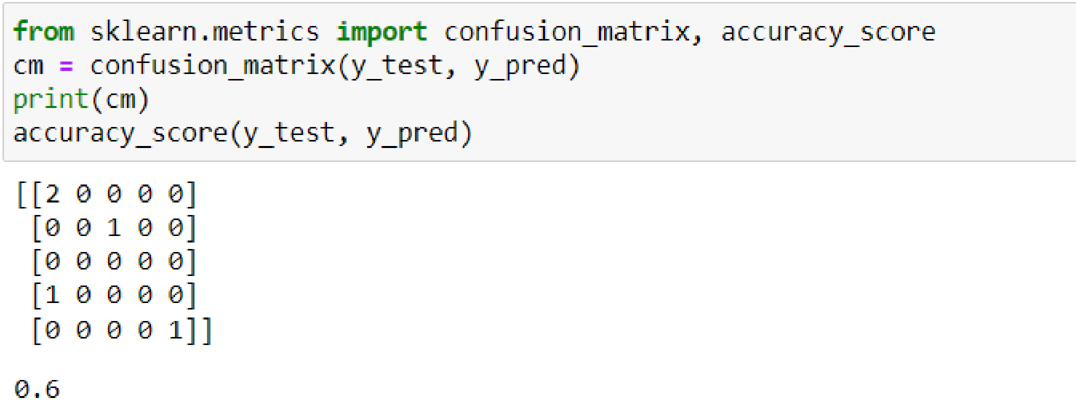
#Predicting the Test set results



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

```
[[1 7]
 [4 2]
 [1 1]
 [9 9]
 [1 1]]
```

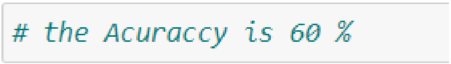
#Making the Confusion Matrix



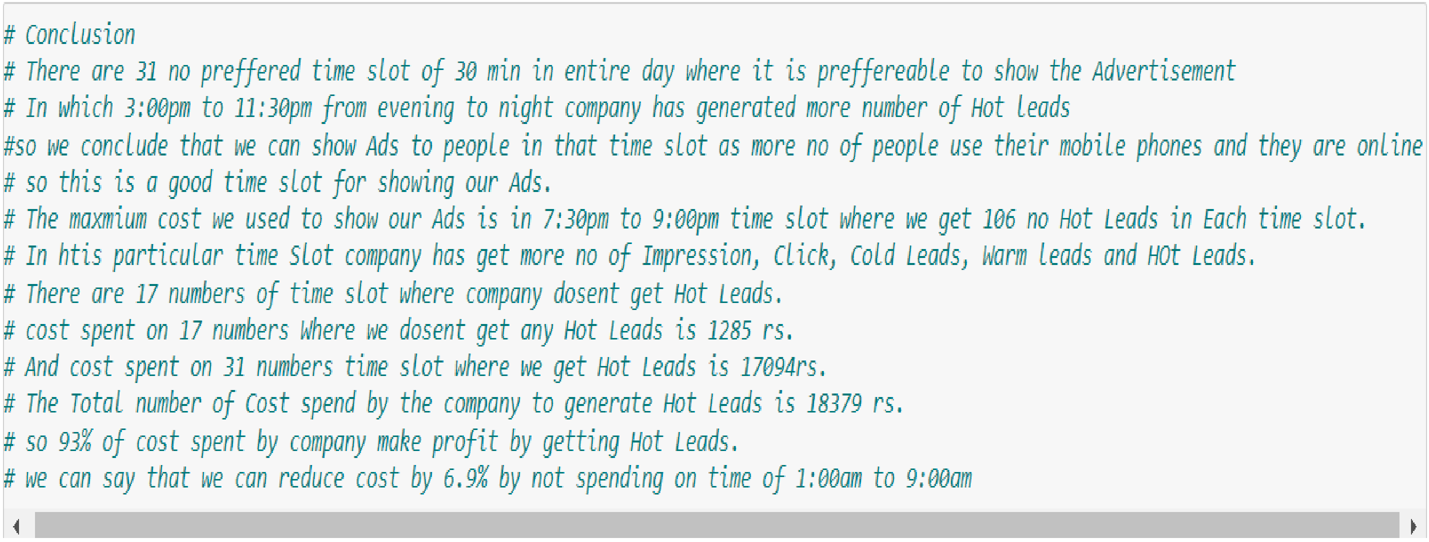
```
from sklearn.metrics import confusion_matrix, accuracy_score
cm = confusion_matrix(y_test, y_pred)
print(cm)
accuracy_score(y_test, y_pred)
```

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

```
0.6
```



```
# the Acuraccy is 60 %
```



```
# Conclusion
# There are 31 no preffered time slot of 30 min in entire day where it is preffereable to show the Advertisement
# In which 3:00pm 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 106 no 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 17 numbers of time slot where company dosent get Hot Leads.
# cost spent on 17 numbers Where we dosent get any Hot Leads is 1285 rs.
# And cost spent on 31 numbers time slot where we get Hot Leads is 17094rs.
# The Total number of Cost spend by the company to generate Hot Leads is 18379 rs.
# so 93% of cost spent by company make profit by getting Hot Leads.
# we can say that we can reduce cost by 6.9% by not spending on time of 1:00am to 9:00am
```

# comparative Analysis of 19th ,20th & 21st June 2023

I have started working on this live project from 19th June to 21st June. I am analyzing impressions, clicks, cost and leads for this project.

The data is in hourly format and then divided into 30 min of slots. The dataset is about showing ads and generate leads in time slots. These time slots are seems to be effective for showing advertisements and also shows a good time to focus on displaying ads to reach maximum leads and engage the leads.

19th June

The time slot from 5:00 pm to 12:00 am have generated hot leads in this slots of time. The total no of hot lead generated on 19th June are **73**. Company spent total amount of **20057** for advertisements. **Company has spent 264rs for each hot lead.** The ratio of converting cold lead **to hot lead is 18.62%.**and on 19th June 2023, we make profit of **96%** from cost spent on hot leads. **CTR for 19th June is 5.21%.**

20th June

Data we observed on 20th June 2023 is bit similar to 19th June 2023 as at on same time slots of 5:00 pm to 12:00 am we get a higher no of hot leads in that slots. The total no of hot lead generated on 20th June are **135**. The ratio of converting cold lead to hot lead is **10.67%.**and on 20th June the profit we make is of **95.94%** from cost that spent on hot leads. Company spent total amount of **16405** for showing all Advertisement . And **116rs of cost is spent for each hot lead. CTR for 20th June is 5.21%.**

21st June

By **comparing 19th and 20th June data** we can see that on 21st June time slot is been **changed to 3:00pm to 12:00 am**, were the company get total **122 Hot Leads on 21st June** . And the ratio of converting Cold Leads to **Hot Leads is 8.60%** , And as well on 21st June Company make a profit of 93% from the total cost spent to get Leads. Company has **spent Total 18379rs** of amount for Advertisement . **CTR** of 21st June **is 4.33%** and company has **spent 140rs on each Hot Lead.**