Google ads Hourly Analysis

Date : 19-06-2023

Project Start Date - End Date	• Start Date – 19 -06 -2023 • End Date – 19-06 2023
Objectives	• To analyses how many people who clicked on the advertisement enrolled 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 : Mahesh R. Darade
Mobile: 8459693181
Email: daradem18@gmail.com

Project Abstract

Google ads Hourly Analysis

This is Marketing Analysis for Education technology company.

In this data Impression indicates the visibility of the Advertisement., Clicks indicates the interested persons, Hot leads indicates that the number of person who are ready to purchase of service, Warm leads indicates the number of person who are really interested in service, Cold leads indicates the number of persons who are getting the information about services, CTR shows click through rate which is conversion rate of the customers, CPC shows cost per click. As we are looking for at which preferred time in a day where we can do marketing and we will get sales definitely.

We have analyze using Descriptive and Exploratory Analyses also further we have used Decision tree Classification.

Google ads Hourly Analysis

Importing the libraries

In [56]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

Importing the dataset

In [4]:
```
data = pd.read_excel("C:/Users/91845/Downloads/Google ads hourly analysis 19th june.xlsx")
```

In [13]:
```
data
```

Out[13]:

	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

Processing Dataset

```
In [9]: a = data.drop("Sr_no", axis=1)
```

```
In [10]: a.isnull().sum()
```

```
Out[10]: Impressions     0
         Clicks          0
         Cost            0
         CTR             0
         CPC             0
         Cold Leads      0
         Warm Leads      1
         Hot Leads       0
         dtype: int64
```

```
In [12]: b = a.fillna(a.mean() )
```

Splitting the dataset into the Training set and Test set

```
In [18]: from sklearn.model_selection import train_test_split
```

```
In [19]: x_train, x_test, y_train, y_test = train_test_split( x,y, test_size=0.4, random_state=50)
```

```
In [20]: from sklearn.preprocessing import StandardScaler
         sc = StandardScaler()
```

```
In [21]: x_train = sc.fit_transform(x_train)
```

```
In [22]: x_test = sc.fit_transform(x_test)
```

Feature Scaling

```
In [20]: from sklearn.preprocessing import StandardScaler
         sc = StandardScaler()
```

```
In [21]: x_train = sc.fit_transform(x_train)
```

```
In [22]: x_test = sc.fit_transform(x_test)
```

Training the Decision Tree Classification model on the Training set

```
In [23]: from sklearn.tree import DecisionTreeClassifier

In [24]: DT = DecisionTreeClassifier(criterion="gini",random_state=0)

In [25]: DT.fit(x_train,y_train)

Out[25]: DecisionTreeClassifier
         DecisionTreeClassifier(random_state=0)
```

Predicting the Test set results

```
In [26]: print(DT.predict(sc.transform([[5942,356,594,0.0600,1.666667,11,5.0]])))
         [2]

In [27]: from sklearn.metrics import confusion_matrix, accuracy_score

In [28]: y_prediction = DT.predict(x_test)

In [29]: y_prediction

Out[29]: array([2, 2, 3, 4, 0, 0, 0, 1, 0, 1, 0, 4, 0, 4, 2, 4, 0, 3, 2, 2],
               dtype=int64)
```

Prediction Accuracy

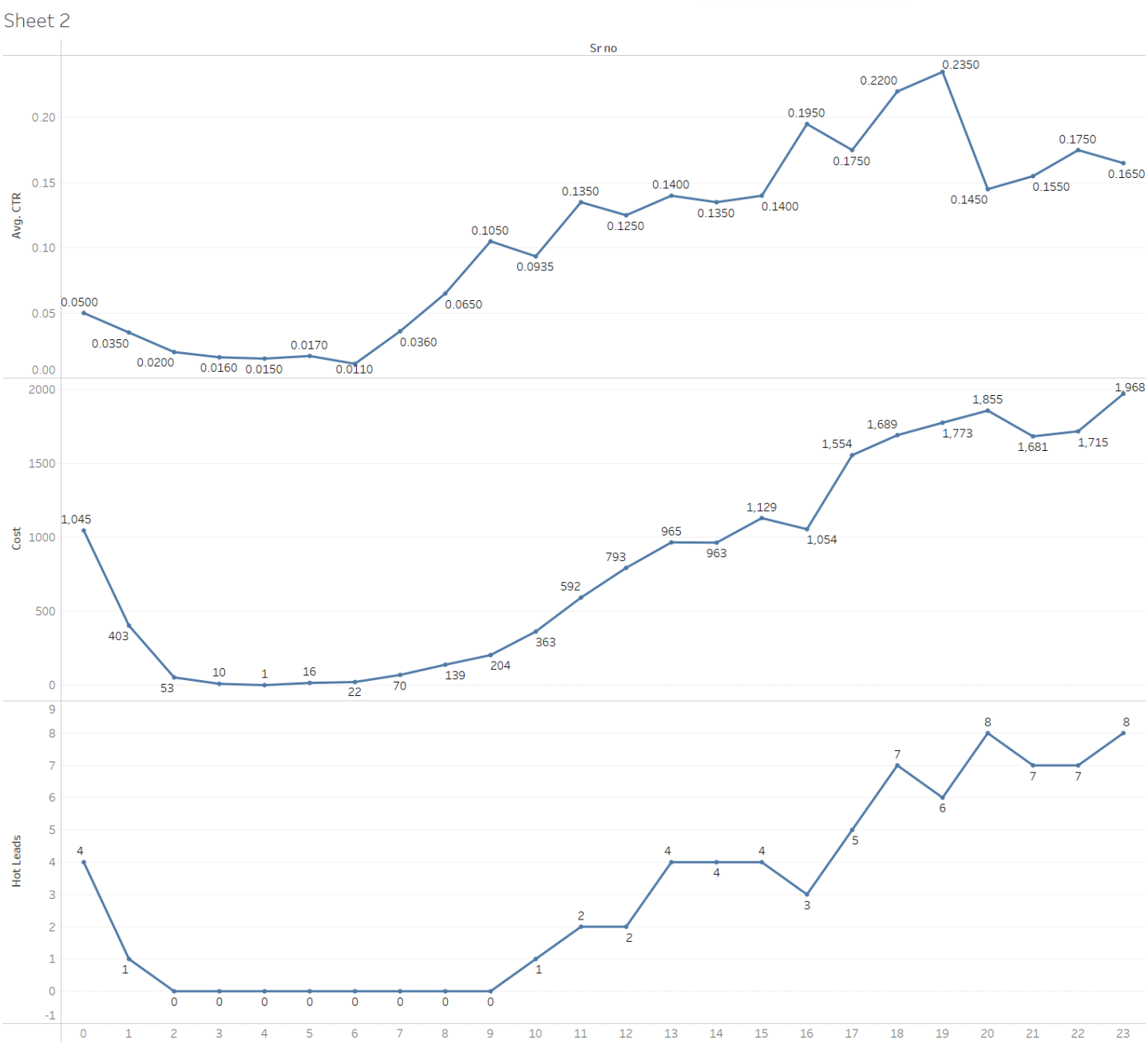
```
In [38]: accuracy_score(y_test,y_prediction)

Out[38]: 0.85
```

Conclusion

From the given data we can conclude that the Decision tree is giving 85% accuracy for advertisement data.

Data Visualization (using Tableau)



#insights

1. On 19th June company had spend 20057 Rupees for getting 73 Hot leads.
2. On this day Lead Acquisition cost is 274.75 Rupees.
3. From 1am-9am in this time Company spend 1838 Rupees which is not helpful, Because in this time there is very less conversion rate that is only 3% and no leads. Company can avoid this time for Marketing
4. From 9am-4pm good time for marketing because in this time conversion rate is 12.84% which is good and also CPC is low which is 0.82 rupee. 9am-4pm Company spend 5497 rupees. 26% of Sell happen in this time.
5. From 4pm-12:30am It is a peak time for marketing because in this time 74% sell happen. Conversion rate also high which is 18.60%. Cost per click (CPC) is also very low that is 0.56 rupee.