Generate Insights for Marketing Intelligence

Data Analyst, Marketing Intelligence

Task 1 - Marketing Campaigns

My used tools:

my operation system; ubuntu(18.04)

my ide for python; spyder on the anaconda

for data analysis and visualization; python(3.6.4v)

a) You can fnd below my code, graph and my comment about entire market and campaigns. Please look top of the each graphs for my comments.

```
""" IMPORT """
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt


""" GET DATA and FIRST CHECKING for null or duplicate variables"""
data_path="/home/mahmut/Documents/DataScience/trivago_remote_task/marketing_campaigns.csv"
data_campaign = pd.read_csv(data_path)
print(data_campaign)
type(data_campaign)
data_campaign = data_campaign.drop_duplicates()
```

""" DATA STATISTICS EDA (Explore Data Analysis with graph) """

Data Statistics ~ The meaning of the campaign data

```
#frst 5 rows of all data
data_campaign.head(5)
```

Out[13]:

	Week	Campaign	Visits	Revenue	Cost
0	1	Aldebaran	27	2.269511	3.763627
1	2	Aldebaran	64	10.820403	15.322613
2	3	Aldebaran	80	7.132998	10.753533
3	4	Aldebaran	93	11.085813	16.906191
4	5	Aldebaran	120	14.282481	21.446570

```
#data information
data_campaign.info()
<class 'pandas.core.frame.DataFrame'>
Int64Index: 90 entries, 0 to 89
Data columns (total 5 columns):
Week        90 non-null int64
Campaign    90 non-null object
Visits      90 non-null int64
Revenue     90 non-null foat64
Cost        90 non-null foat64
dtypes: foat64(2), int64(2), object(1)
memory usage: 6.7+ KB
```

```
#description of data
data_campaign.describe()
```

Out[15]:

	Week	Visits	Revenue	Cost
count	90.000000	90.000000	90.000000	90.000000
mean	15.500000	214.788889	235.401749	239.888475
std	8.703932	128.437498	134.127862	135.876952
min	1.000000	27.000000	2.269511	3.763627
25%	8.000000	144.000000	120.761712	127.781050
50%	15.500000	158.500000	232.090920	234.700293
75%	23.000000	235.000000	356.154278	346.847241
max	30.000000	613.000000	463.249265	507.521951

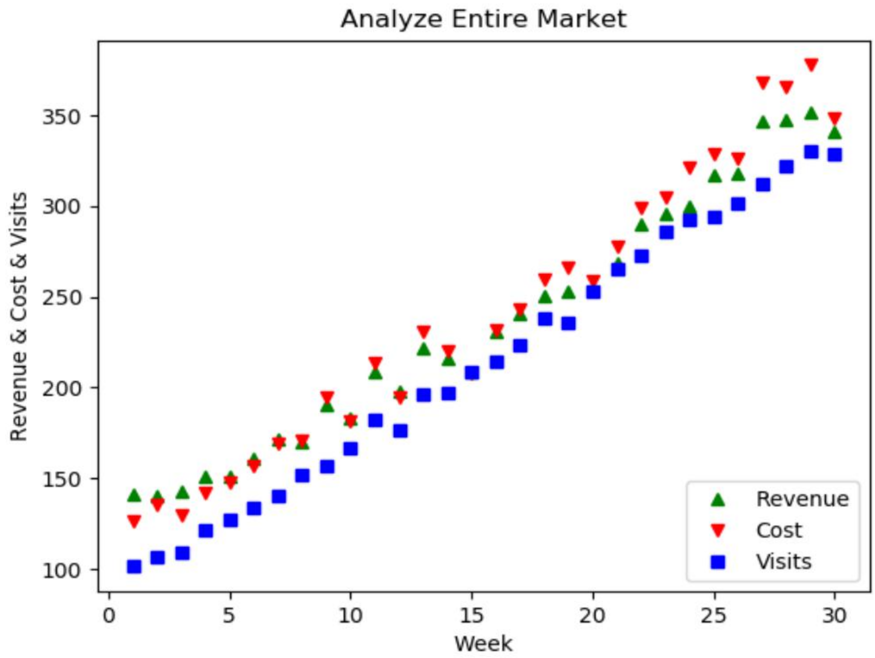
```
#Function for graph
week = data_campaign['Week'].drop_duplicates()

#CDF (Cumulative Distribution Function)
def cdf(cdf_data1, cdf_data2, label1, label2, label3):
    plt.fgure(1)

    if len(cdf_data1) > 0:
        len_data1 = len(cdf_data1)
        data_array1 = np.arange(0,len_data1)
        probability1 = data_array1/len_data1
        plt.plot(np.sort(cdf_data1), probability1, marker='.', linestyle='none')

    if len(cdf_data2) > 0:
        len_data2 = len(cdf_data2)
        data_array2 = np.arange(0,len_data2)
        probability2 = data_array2/len_data2
        plt.plot(np.sort(cdf_data2), probability2, marker='.', linestyle='none')

    plt.legend([label2, label3], loc=4)
    plt.xlabel(label1)
    plt.ylabel('probability')
    plt.title('CDF (Cumulative Distribution Function)')
    plt.show()
```

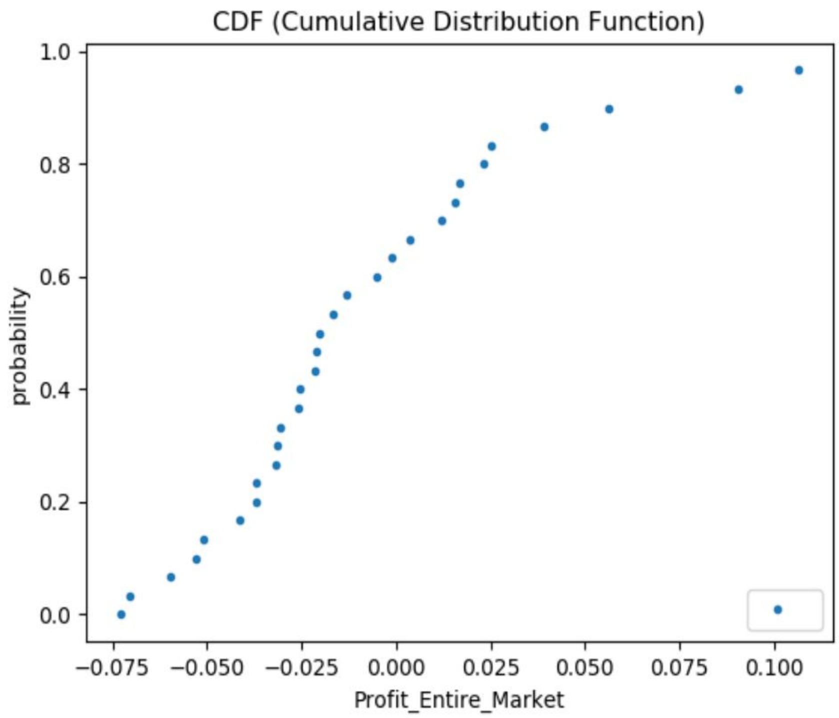
```
#EDA
def DataAnalyze(campaign_name):
    if campaign_name == 'Entire Market':
        campaign = data_campaign
        revenue = campaign.groupby('Week')['Revenue'].mean()
        visits = campaign.groupby('Week')['Visits'].mean()
        cost    = campaign.groupby('Week')['Cost'].mean()
        proft = ((revenue-cost) / revenue)
        cdf(proft, '', 'Proft_Entire_Market', '', '')
    else:
        campaign = data_campaign[(data_campaign.Campaign == campaign_name)]
        revenue = campaign['Revenue']
        visits = campaign['Visits']
        cost    = campaign['Cost']
        proft = ((revenue-cost) / revenue)
        cdf(proft, '', 'Proft_' + campaign_name + '_Campaign', '', '')

    plt.fgure(2)
    plt.plot(week, revenue, 'g^', week, cost, 'rv', week, visits, 'bs')
    plt.legend(['Revenue', 'Cost', 'Visits'], loc=4)
    plt.xlabel('Week')
    plt.ylabel('Revenue & Cost & Visits')
    plt.title('Analyze ' + campaign_name)
    plt.show()

    plt.fgure(3)
    plt.plot(visits, revenue, 'r*')
    plt.xlabel('Visits')
    plt.ylabel('Revenue')
    plt.title('Quality of Trafc for ' + campaign_name)
    plt.show()

    plt.fgure(4)
    plt.plot(cost, revenue, 'g^', cost, visits, 'bs')
    plt.legend(['Revenue', 'Visits'], loc=4)
    plt.xlabel('Cost')
    plt.ylabel('Revenue & Visits')
    plt.title('ROAD MAP ' + campaign_name)
    plt.show()
```

When I look at the 'Entire Market' graph; Generally everything is seems good. Because revenue and visits rise up. But it is not enough that increased the number of visitors and revenue. Because revenue is falling behind cost. If you want to detail please look 'Proft_Entire_Market' graph. This graph tell us that while our are losing about 70% of entire market, our profting just are 30%.

#Call Function for entire market

DataAnalyze('Entire Market')

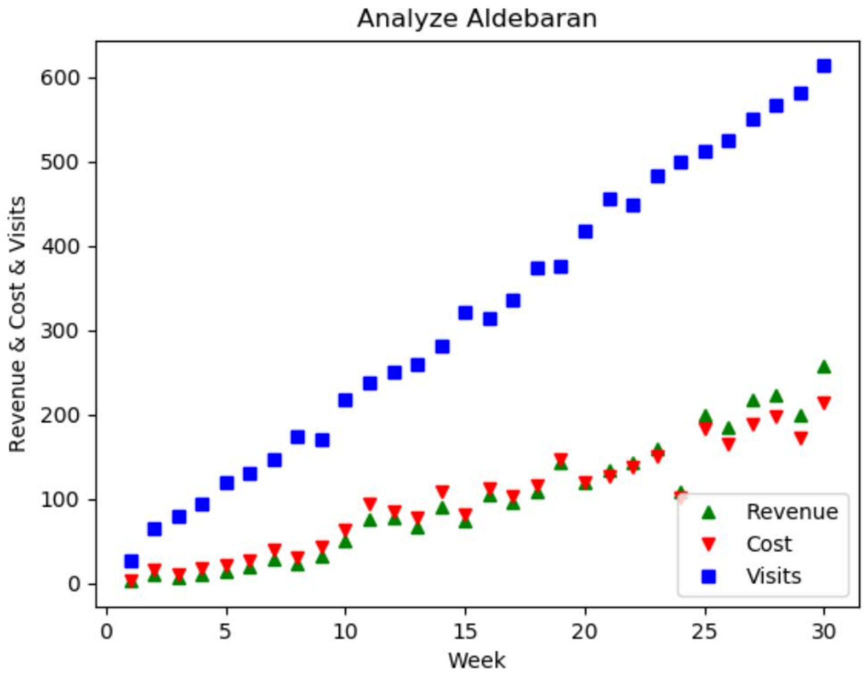


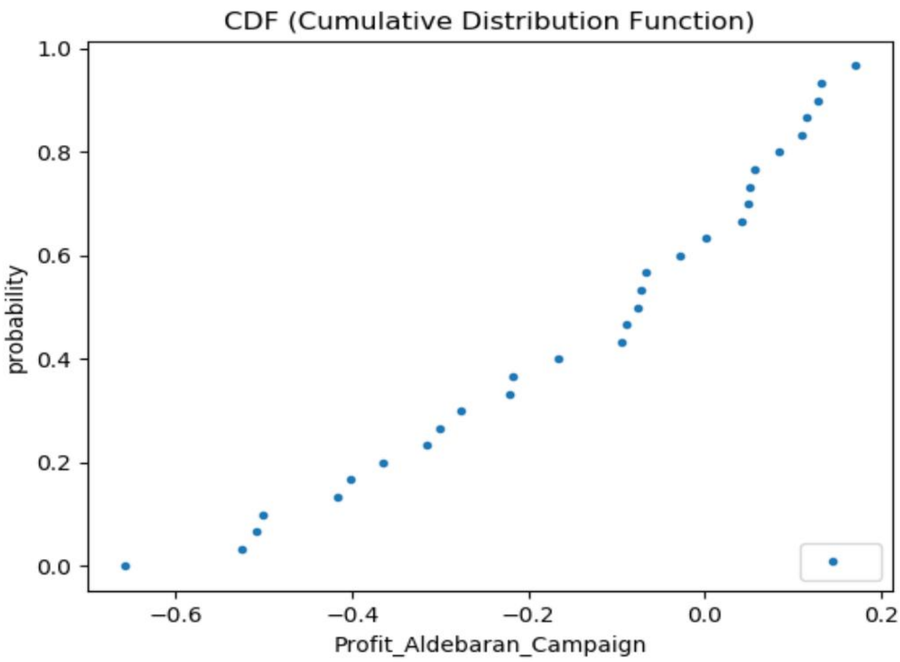


When I look at the ‘Aldebaran’ campaign graph; Generally everything is seems good. Because revenue and visits rise up also you should look carefully at 20th week and after. Because we are starting to make proft after the this week. If you want to detail please look ‘Proft_Aldebaran_Campaign’ graph. This graph tell us that while our are losing about 60% of entire market, our profting are 40%.

#Call Function for Aldebaran campaign

DataAnalyze('Aldebaran')

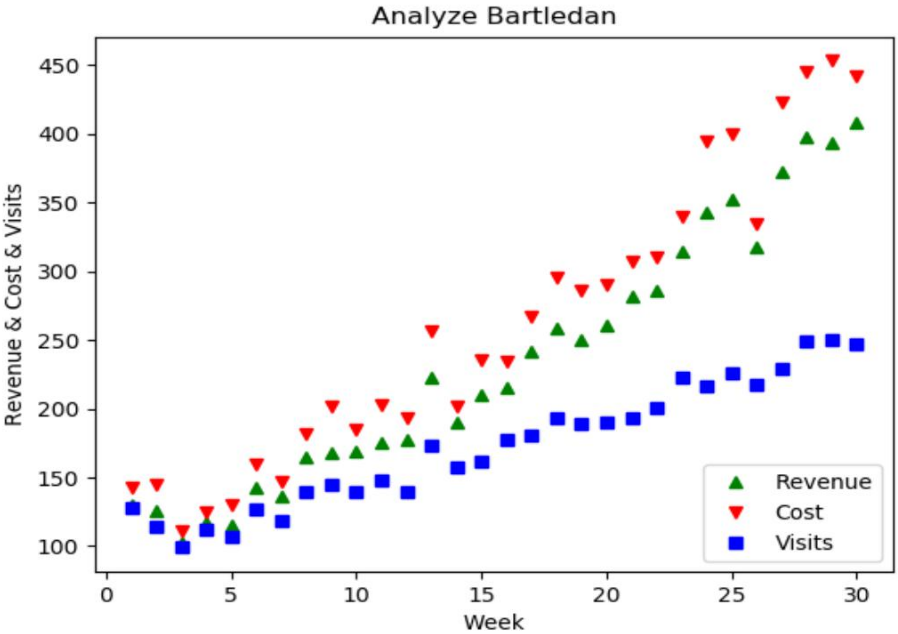


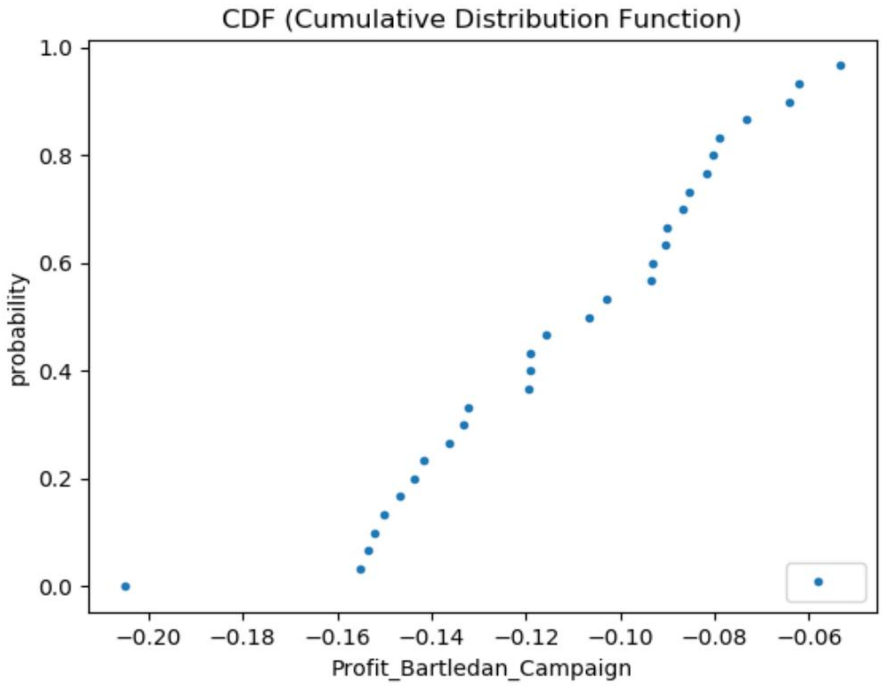


When I look at the ‘ Bartledan Campaign’ graph; Generally everything is seems good. Because revenue and visits rise up. But it is not enough that increased the number of visitors and revenue. Because revenue is falling behind cost. If you want to detail please look ‘Proft_Bartledan_Campaign’ graph. This graph tell us that through this campaign we can't gain any proft. It is clearly seen at the frst graph; all time we have lossed.

#Call Function for Bartledan campaign

DataAnalyze('Bartledan')

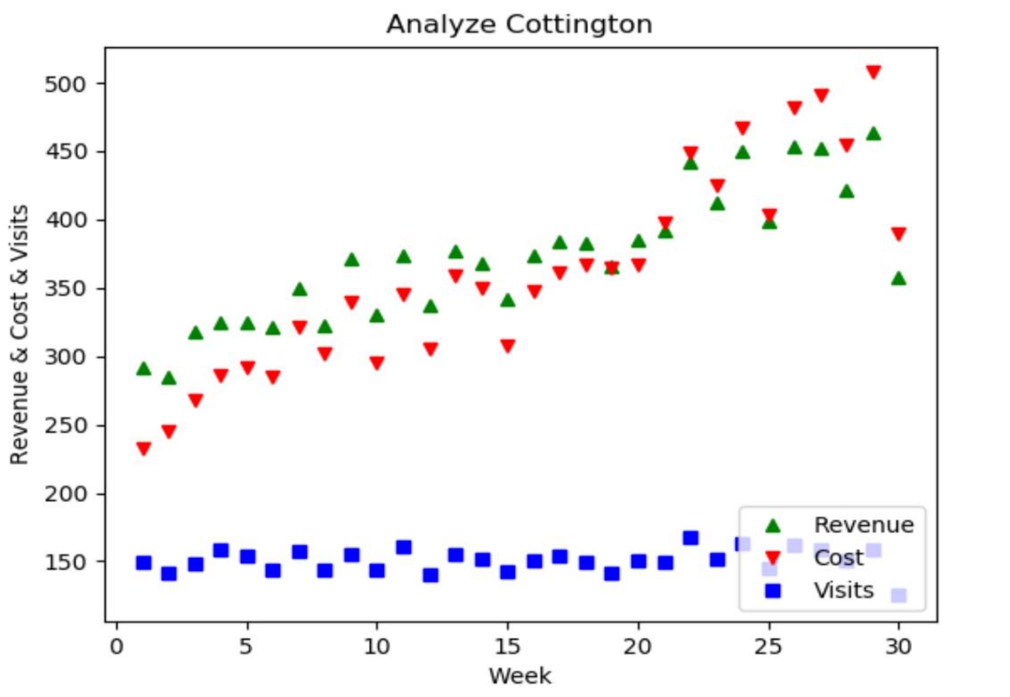


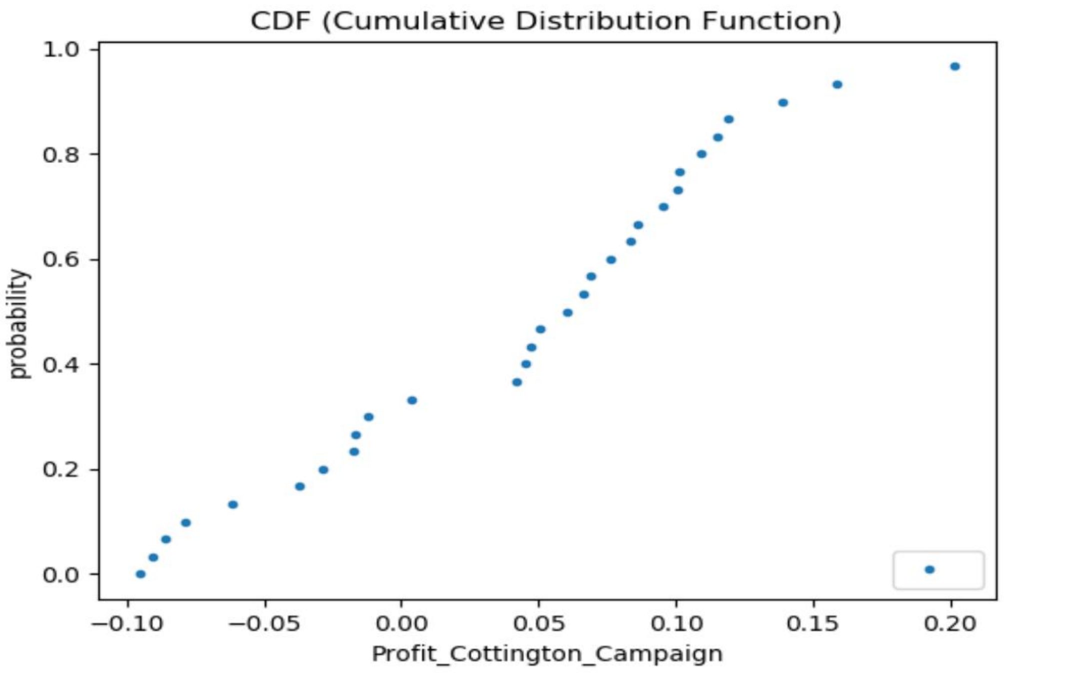


When I look at the ‘ Cottington Campaign’ graph; i see interesting values. Because revenue frst 20 weeks rise up but visits pretty much are same. May be it is good but not enough. Because i am sure you want to rising up visits. After the 20th week revenue is falling behind cost, why? That is meaningless for my data. Because anything did not change. For example number of visits are same. So I don’t understand why change our proft.

#Call Function for Cottington campaign

DataAnalyze('Cottington')





b)

<u>If we sort quality of trafc :</u>

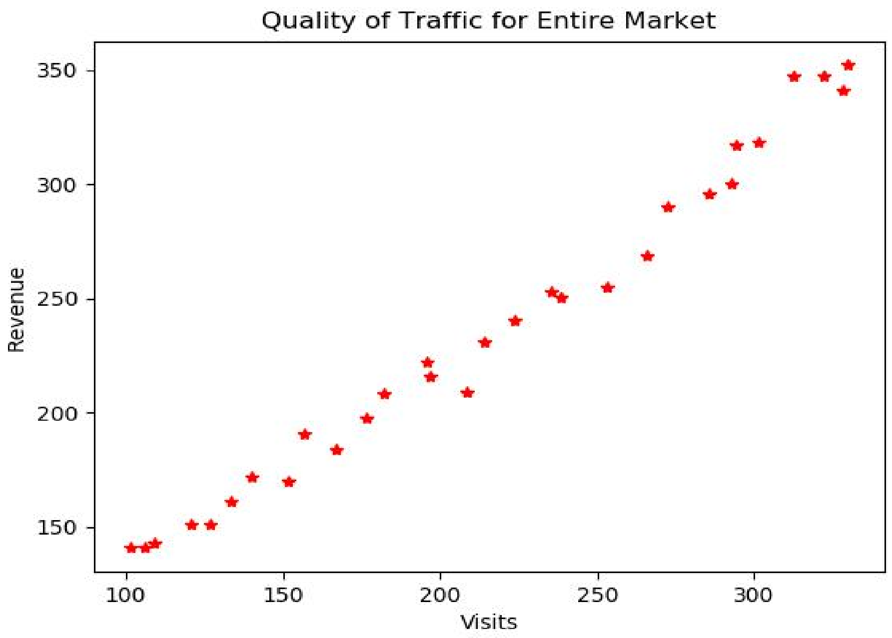
Bartledan is the best.

Aldebaran is better than Cottington and it is worse than Bartleden.

Cottington is the worst.

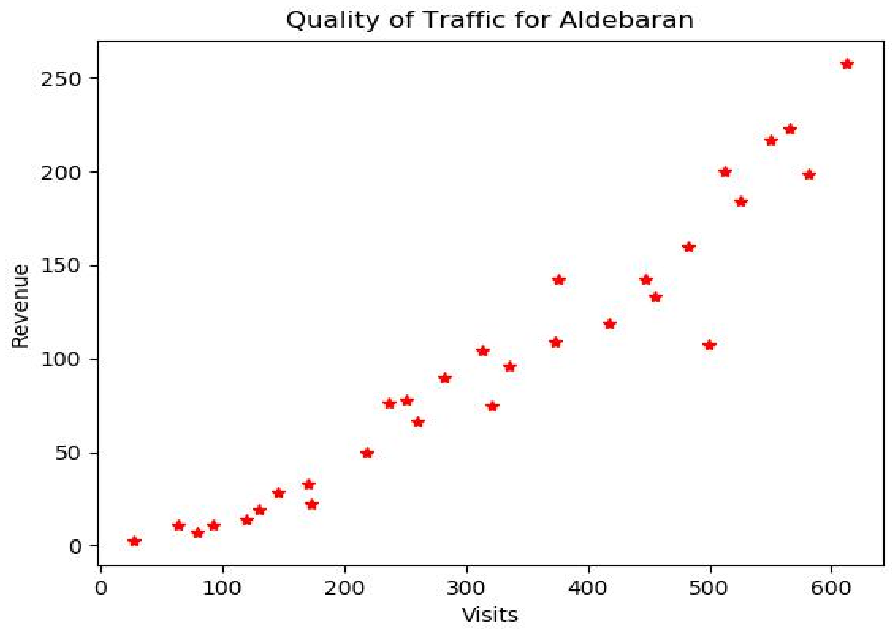
Quality of Trafc for entire market:

For the entire market, that is great. If you rise up visits. you can be make proft.



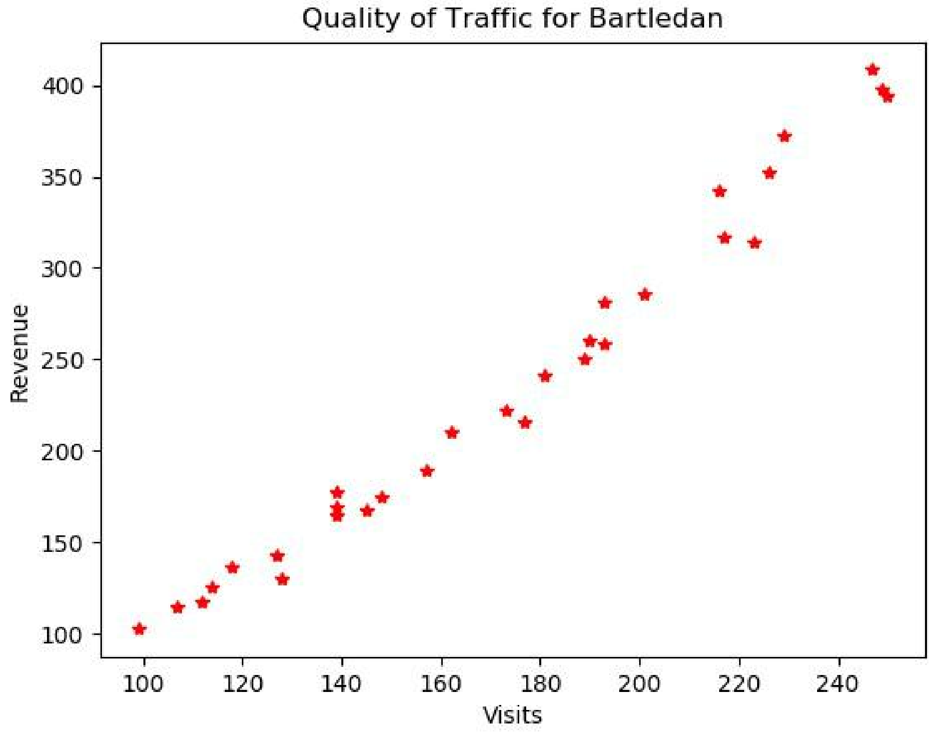
Quality of Trafc for Aldebaran campaign:

That is good. If you rise up visits you can be make proft.



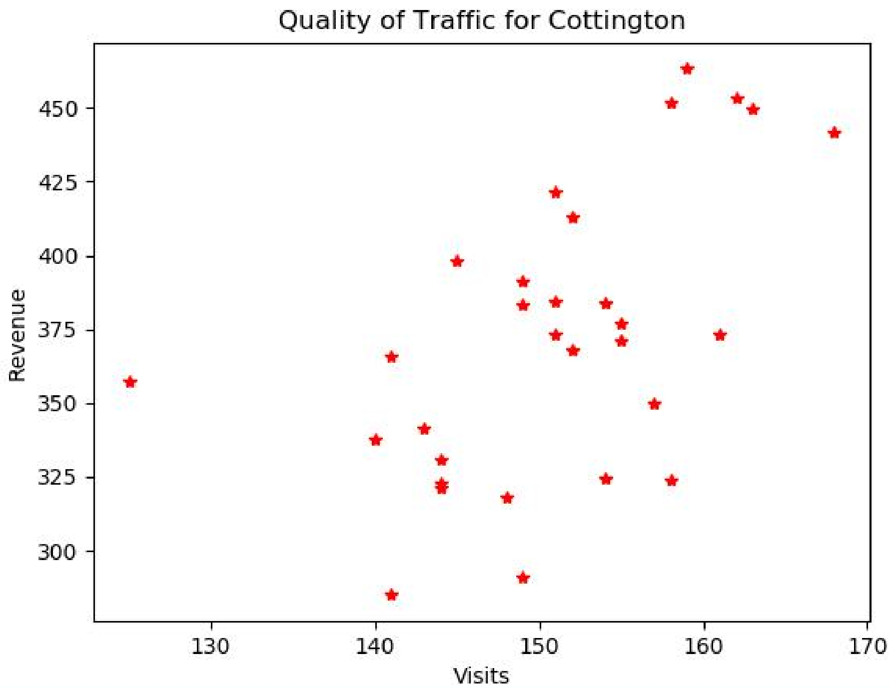
Quality of Trafc for Bartledan campaign:

That is great. If you rise up visits you can be make proft.

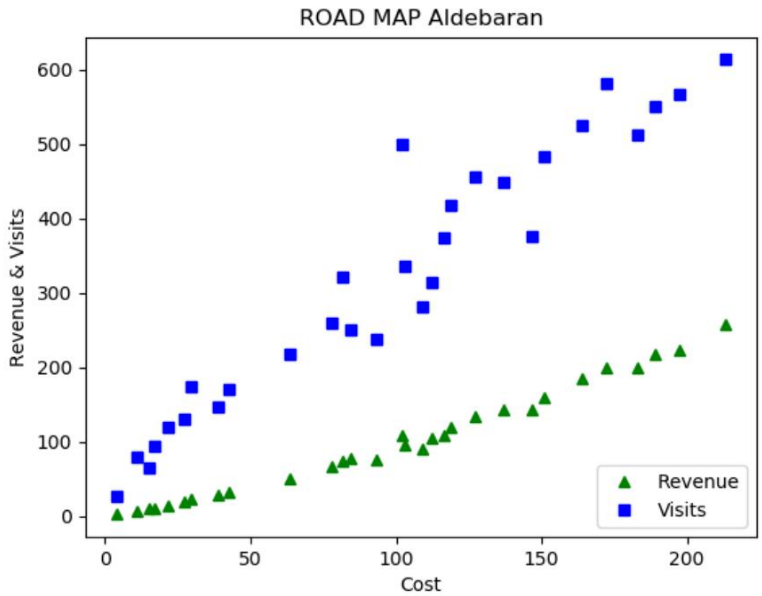


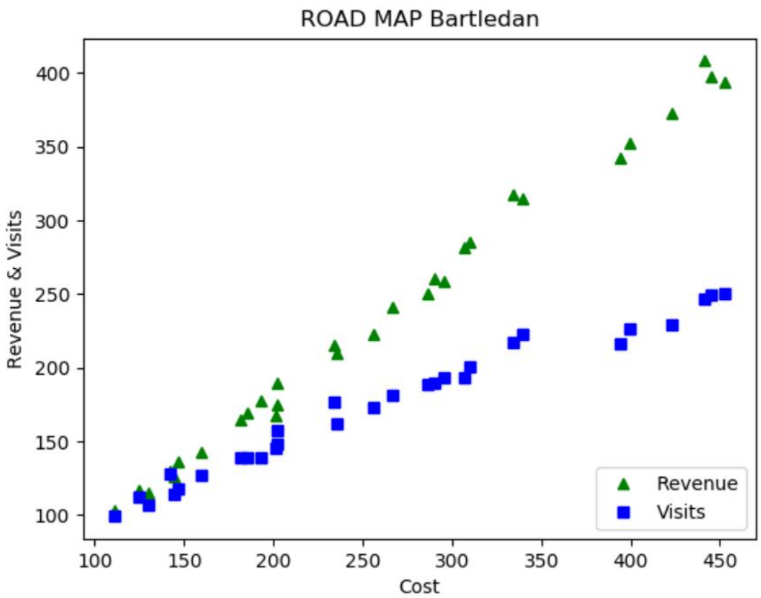
Quality of Trafc for Cottington campaign:

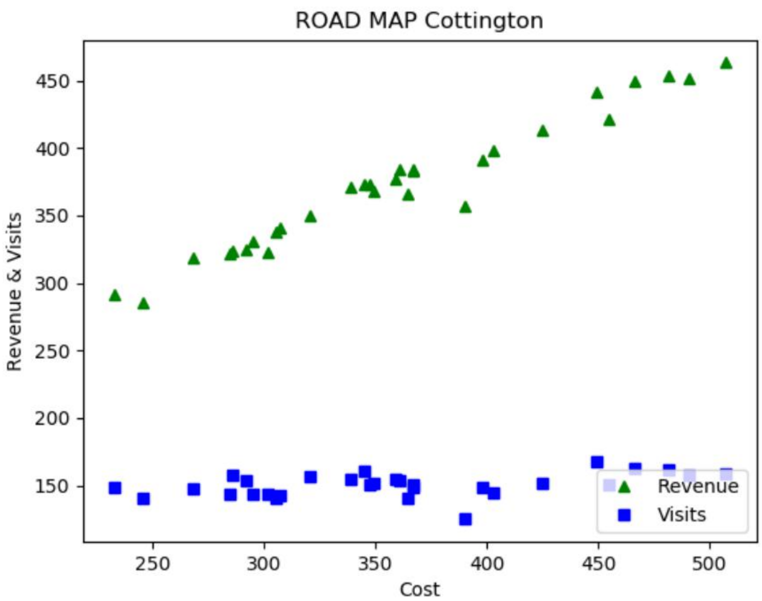
That is bad. I think, relationships are absurd or nonlinear between revenue and visits.



c) First of all we should analyze revenue change and visits change, according to cost. My suggestion is Aldebaran. Because you gain visits and revenue for spend cost. So Aldebaran is best choice. I expect to positive impact for the entire market. Because we will rise up visits and revenue thanks to new invest. Then in the same way this afect impact entire market as positive way.







Task 2 - Session Data

We want to understand what impact your choice. So we analyze relationship between booking and other parameter. At the last we will do hypothesis test.

First of all, we draw some graphs and calculate quantitative analyst for understand data. You can see my code below.

```
""" IMPORT """

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

""" GET DATA and FIRST CHECKING for null or duplicate variables"""

data_path="/home/mahmut/Documents/DataScience/trivago_remote_task/session_data.csv"
data_session = pd.read_csv(data_path)

print(data_session)
type(data_session)
```

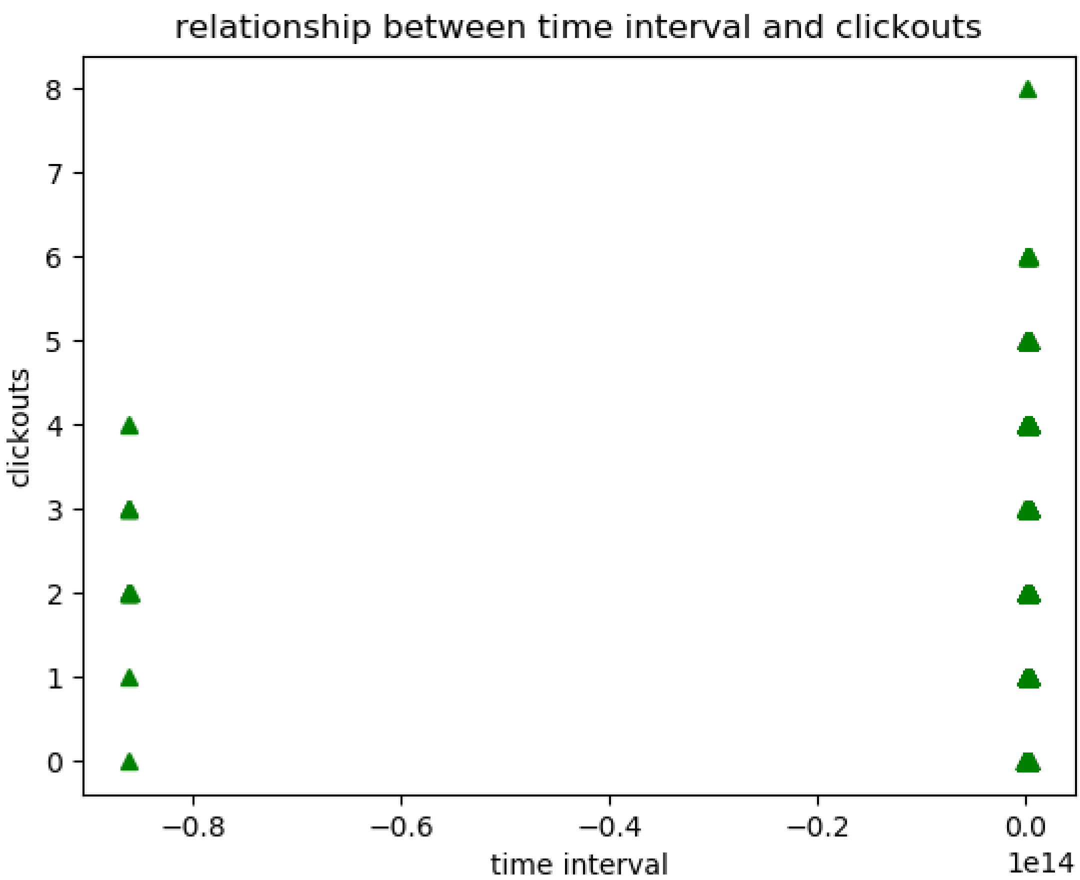
```
""" DATA STATISTICS AND EDA (Explore data analysis with graph) """
# Data Statistics ~ The meaning of the campaign data

data_session.head()
Out[14]:
          session session_start_text session_end_text  clickouts  booking  \
0  20170503000001           06:11:53         06:15:11          3        0
1  20170503000002           21:06:41         21:08:23          3        0
2  20170503000003           12:03:01         12:06:02          3        0
3  20170503000004           05:58:00         06:02:56          0        0
4  20170503000005           09:13:43         09:17:01          1        0

  time_interval
0      00:03:18
1      00:01:42
2      00:03:01
3      00:04:56
4      00:03:18
```

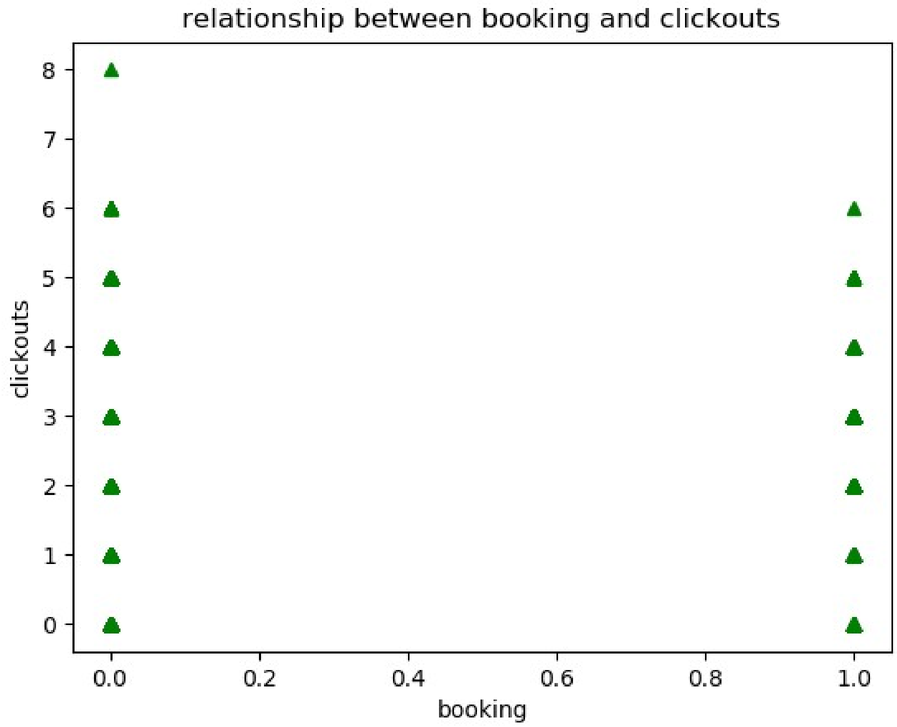
```
data_session.info()
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10000 entries, 0 to 9999
Data columns (total 6 columns):
session               10000 non-null int64
session_start_text    10000 non-null object
session_end_text      10000 non-null object
clickouts             10000 non-null int64
booking               10000 non-null int64
time_interval         10000 non-null timedelta64[ns]
dtypes: int64(3), object(2), timedelta64[ns](1)
memory usage: 468.8+ KB
```

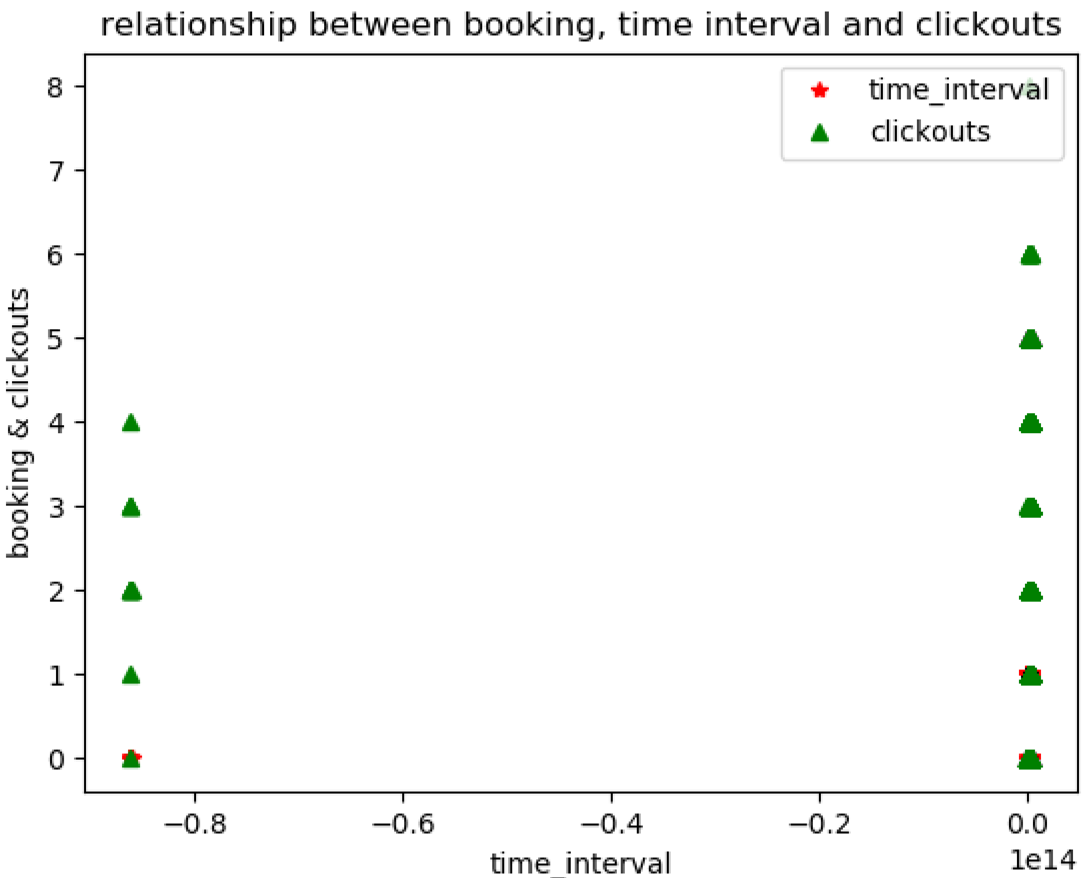
```
data_session.describe()
Out[12]:
          session    clickouts      booking         time_interval
count  1.000000e+04  10000.000000  10000.000000                 10000
mean   2.017050e+13      2.485200      0.096700  0 days 00:01:09.023500
std    2.886896e+03      1.060987      0.295564  0 days 00:51:54.406156
min    2.017050e+13      0.000000      0.000000      -1 days +00:01:39
25%    2.017050e+13      2.000000      0.000000       0 days 00:02:19
50%    2.017050e+13      2.000000      0.000000       0 days 00:03:02
75%    2.017050e+13      3.000000      0.000000       0 days 00:03:43
max    2.017050e+13      8.000000      1.000000       0 days 00:06:18
```

```
data_session['time_interval'] = (pd.to_datetime(data_session.session_end_text) - pd.to_datetime(data_session.session_start_text)) #.astype('timedelta64[s]')
#data_session[data_session.time_interval < 0] = ((24*60*60) + data_session[(data_session['time_interval'] < 0)][['time_interval']])

time_interval = data_session['time_interval']
clickouts = data_session['clickouts']
booking = data_session['booking']

# you spend alot ot time on the web site but you can a little bit click
plt.figure(1)
plt.plot(time_interval, clickouts, 'g^')
plt.xlabel('time interval')
plt.ylabel('clickouts')
plt.title('relationship between time interval and clickouts')
plt.show()

plt.figure(2)
```

```
plt.plot(time_interval, booking, 'r*')
plt.xlabel('time_interval')
plt.ylabel('booking')
plt.title('relationship between time interval and booking')
plt.show()

# you can click much times but you can not booking
plt.figure(3)
plt.plot(booking, clickouts, 'g^')
plt.xlabel('booking')
plt.ylabel('clickouts')
plt.title('relationship between booking and clickouts')
plt.show()

plt.figure(4)
plt.plot(time_interval, booking, 'r*', time_interval, clickouts, 'g^')
plt.legend(['time_interval', 'clickouts'], loc=1)
plt.xlabel('time_interval')
plt.ylabel('booking & clickouts')
plt.title('relationship between booking, time interval and clickouts')
plt.show()
```




relationship between booking and clickouts
clickouts
booking
0
1
2
3
4
5
6
7
8
0.0
0.2
0.4
0.6
0.8
1.0



if we want to know relationship between booking and clickout, we have to run code below. We will see negative correlation. But correlation not strong. May be you can say uncorrelated.

book_click_c = np.corrcoef(booking, clickouts)

	0	1
0	1	-0.0498117
1	-0.0498117	1

if we want to know relationship between booking and time_interval, we have to run code below. We will see positive correlation. But correlation not strong. May be you can say uncorrelated.

```
book_interval_time_c = np.corrcoef(booking, data_session['time_interval'])
```

	0	1
0	1	0.0177646
1	0.0177646	1