

CustomerAnalytics_CustomerBehavior

Follow me on [LinkedIn](#) for more:
Steve Nouri

May 29, 2019

<https://www.linkedin.com/in/stevenouri/>

The importance of customer analytics is rising: because access to customer data became easier for many businesses, and also customers now have easier access to data and information on similar products and contents provided by other competitors, it is critical to many businesses to be able to understand and predict what their customers are likely to purchase or view. **The deeper the understanding your company has about its customers, the better competitive power it will have against its competitors.**

```
In [1]: %matplotlib inline
```

```
In [2]: import matplotlib.pyplot as plt
import pandas as pd
```

1 1. Load Data

This data set is one of the publicly available datasets from IBM at the following link: <https://www.ibm.com/communities/analytics/watson-analytics-blog/marketing-customer-value-analysis/>

```
In [3]: df = pd.read_csv('Data/WA_Fn-UseC_-Marketing-Customer-Value-Analysis.csv')
```

```
In [4]: df.shape
```

```
Out[4]: (9134, 24)
```

```
In [5]: df.head()
```

```
Out[5]:
```

| | Customer | State | Customer Lifetime Value | Response | Coverage | Education | \ |
|---|----------|------------|-------------------------|----------|----------|-----------|---|
| 0 | BU79786 | Washington | 2763.519279 | No | Basic | Bachelor | |
| 1 | QZ44356 | Arizona | 6979.535903 | No | Extended | Bachelor | |
| 2 | AI49188 | Nevada | 12887.431650 | No | Premium | Bachelor | |
| 3 | WW63253 | California | 7645.861827 | No | Basic | Bachelor | |
| 4 | HB64268 | Washington | 2813.692575 | No | Basic | Bachelor | |

| | Effective To Date | EmploymentStatus | Gender | Income | ... | \ |
|---|-------------------|------------------|--------|--------|-----|---|
| 0 | 2/24/11 | Employed | F | 56274 | ... | |
| 1 | 1/31/11 | Unemployed | F | 0 | ... | |
| 2 | 2/19/11 | Employed | F | 48767 | ... | |
| 3 | 1/20/11 | Unemployed | M | 0 | ... | |

| | | | | | |
|---|--------|----------|---|-------|-----|
| 4 | 2/3/11 | Employed | M | 43836 | ... |
|---|--------|----------|---|-------|-----|

| | Months Since Policy Inception | Number of Open Complaints | Number of Policies | \ |
|---|-------------------------------|---------------------------|--------------------|---|
| 0 | 5 | 0 | 1 | |
| 1 | 42 | 0 | 8 | |
| 2 | 38 | 0 | 2 | |
| 3 | 65 | 0 | 7 | |
| 4 | 44 | 0 | 1 | |

| | Policy Type | Policy | Renew Offer Type | Sales Channel | \ |
|---|----------------|--------------|------------------|---------------|---|
| 0 | Corporate Auto | Corporate L3 | Offer1 | Agent | |
| 1 | Personal Auto | Personal L3 | Offer3 | Agent | |
| 2 | Personal Auto | Personal L3 | Offer1 | Agent | |
| 3 | Corporate Auto | Corporate L2 | Offer1 | Call Center | |
| 4 | Personal Auto | Personal L1 | Offer1 | Agent | |

| | Total Claim Amount | Vehicle Class | Vehicle Size |
|---|--------------------|---------------|--------------|
| 0 | 384.811147 | Two-Door Car | Medsize |
| 1 | 1131.464935 | Four-Door Car | Medsize |
| 2 | 566.472247 | Two-Door Car | Medsize |
| 3 | 529.881344 | SUV | Medsize |
| 4 | 138.130879 | Four-Door Car | Medsize |

[5 rows x 24 columns]

In [6]: df.columns

Out[6]: Index(['Customer', 'State', 'Customer Lifetime Value', 'Response', 'Coverage',
'Education', 'Effective To Date', 'EmploymentStatus', 'Gender',
'Income', 'Location Code', 'Marital Status', 'Monthly Premium Auto',
'Months Since Last Claim', 'Months Since Policy Inception',
'Number of Open Complaints', 'Number of Policies', 'Policy Type',
'Policy', 'Renew Offer Type', 'Sales Channel', 'Total Claim Amount',
'Vehicle Class', 'Vehicle Size'],
dtype='object')

2 2. Analytics on Engaged Customers

We are going to analyze it to understand how different customers behave and react to different marketing strategies.

2.1 - Overall Engagement Rate

The Response field contains information about whether a customer responded to the marketing efforts.

In [7]: # Get the total number of customers who have responded

```
df.groupby('Response').count()['Customer']
```

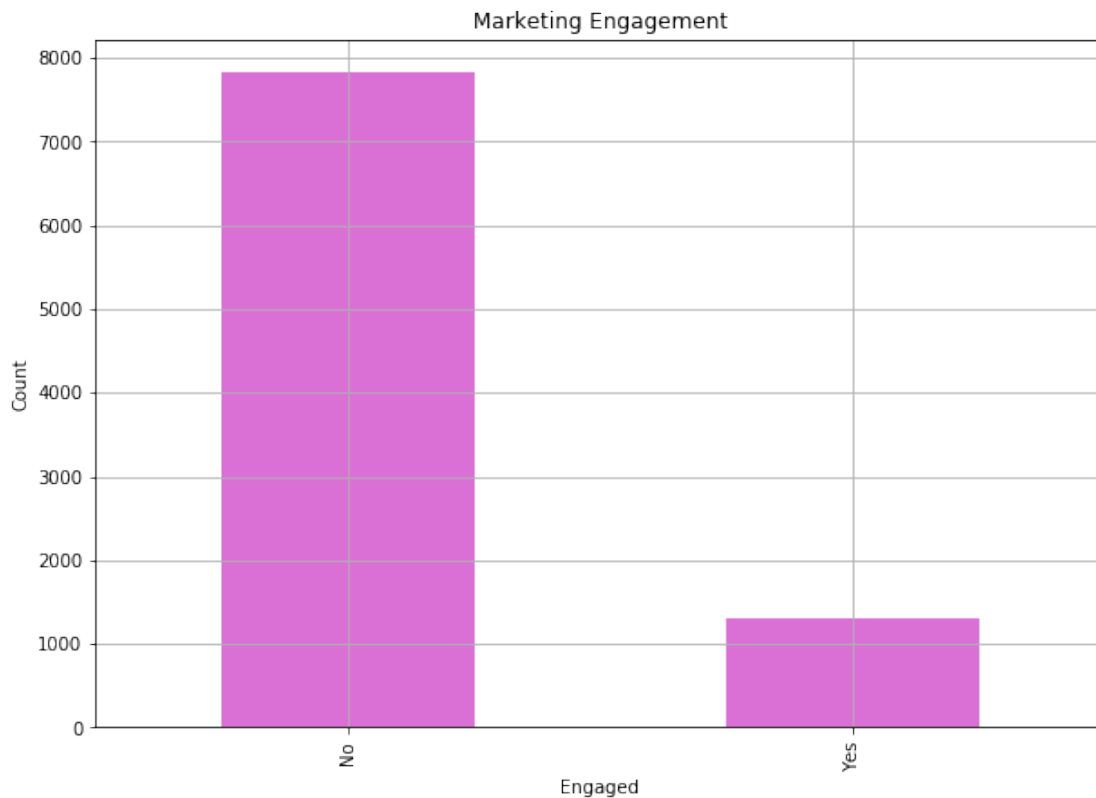
```
Out[7]: Response
No      7826
Yes     1308
Name: Customer, dtype: int64
```

```
In [8]: # Visualize this in a bar plot
```

```
ax = df.groupby('Response').count()['Customer'].plot(
    kind='bar',
    color='orchid',
    grid=True,
    figsize=(10, 7),
    title='Marketing Engagement'
)

ax.set_xlabel('Engaged')
ax.set_ylabel('Count')

plt.show()
```



```
In [9]: # Calculate the percentages of the engaged and non-engaged customers
```

```
df.groupby('Response').count()['Customer']/df.shape[0]
```

```
Out[9]: Response
No      0.856799
Yes     0.143201
Name: Customer, dtype: float64
```

From this output and from the plot, we can see that only about 14% of the customers responded to the marketing calls.

2.2 - Engagement Rates by Offer Type

The Renew Offer Type column in this DataFrame contains the type of the renewal offer presented to the customers. We are going to look into what types of offers worked best for the engaged customers.

```
In [10]: # Get the engagement rates per renewal offer type
```

```
by_offer_type_df = df.loc[
    df['Response'] == 'Yes', # count only engaged customers
].groupby([
    'Renew Offer Type' # engaged customers grouped by renewal offer type
]).count()['Customer'] / df.groupby('Renew Offer Type').count()['Customer']

by_offer_type_df
```

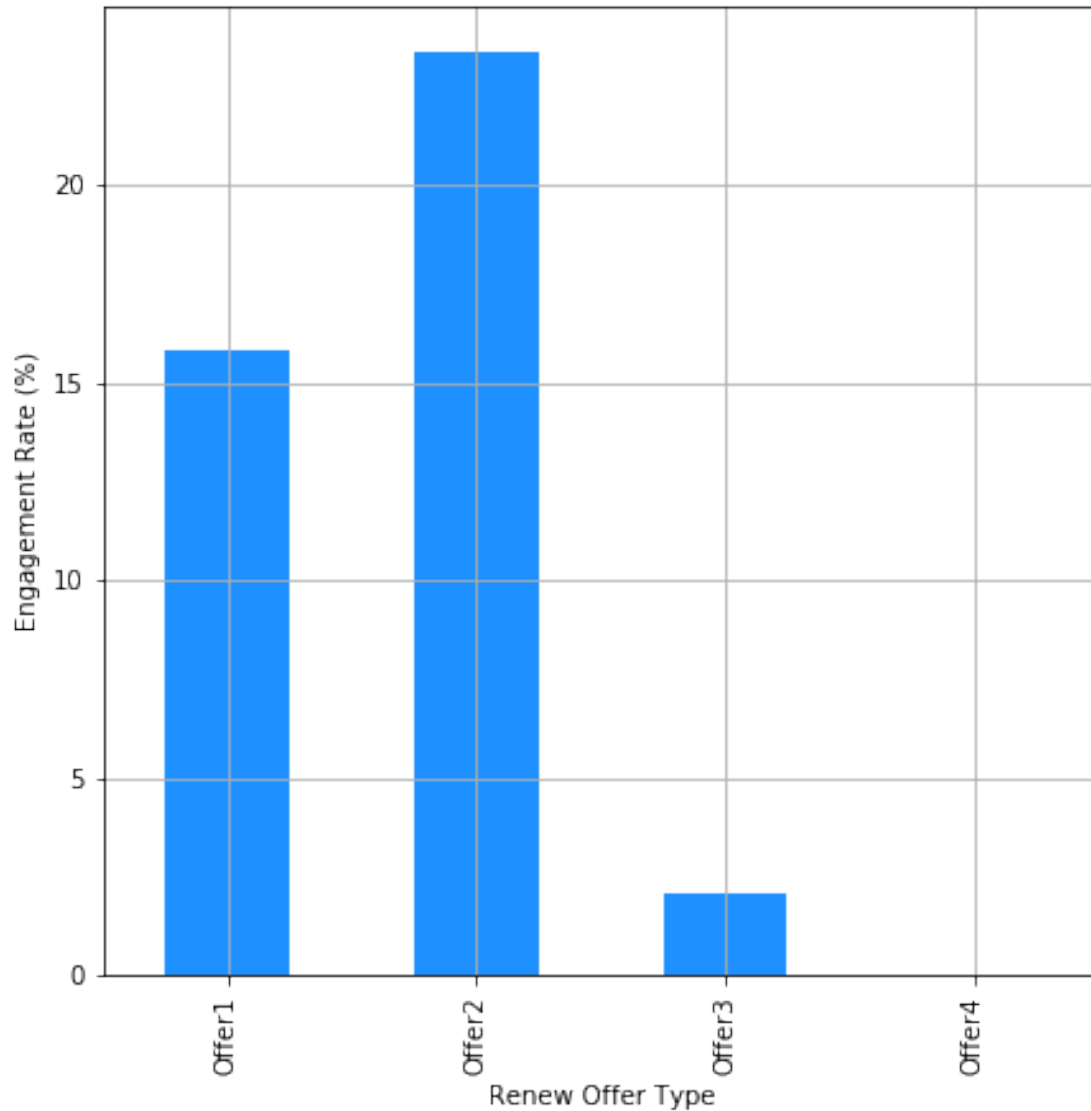
```
Out[10]: Renew Offer Type
Offer1    0.158316
Offer2    0.233766
Offer3    0.020950
Offer4         NaN
Name: Customer, dtype: float64
```

```
In [11]: # Visualize it in a bar plot
```

```
ax = (by_offer_type_df*100.0).plot(
    kind='bar',
    figsize=(7, 7),
    color='dodgerblue',
    grid=True
)

ax.set_ylabel('Engagement Rate (%)')

plt.show()
```



As we can see, **Offer2** had the highest engagement rate among the customers

2.3 - Offer Type & Vehicle Class

We are going to understand how customers with different attributes respond differently to different marketing messages. We start looking at the engagements rates by each offer type and vehicle class.

```
In [12]: by_offer_type_df = df.loc[
          df['Response'] == 'Yes' # engaged customers
        ].groupby([
          'Renew Offer Type', 'Vehicle Class' # grouping the data by these two columns
        ]).count()['Customer'] / df.groupby('Renew Offer Type').count()['Customer'] # rates f
```

```
by_offer_type_df
```

```
Out[12]: Renew Offer Type Vehicle Class
Offer1      Four-Door Car    0.070362
            Luxury Car      0.001599
            Luxury SUV      0.004797
            SUV              0.044776
            Sports Car      0.011194
            Two-Door Car    0.025586
Offer2      Four-Door Car    0.114833
            Luxury Car      0.002051
            Luxury SUV      0.004101
            SUV              0.041012
            Sports Car      0.016405
            Two-Door Car    0.055366
Offer3      Four-Door Car    0.016760
            Two-Door Car    0.004190
Name: Customer, dtype: float64
```

```
In [13]: # Make the previous output more readable using unstack function
         # to pivot the data and extract and transform the inner-level groups to columns
```

```
by_offer_type_df = by_offer_type_df.unstack().fillna(0)
by_offer_type_df
```

```
Out[13]: Vehicle Class      Four-Door Car  Luxury Car  Luxury SUV      SUV  Sports Car  \
Renew Offer Type
Offer1              0.070362    0.001599    0.004797  0.044776    0.011194
Offer2              0.114833    0.002051    0.004101  0.041012    0.016405
Offer3              0.016760    0.000000    0.000000  0.000000    0.000000

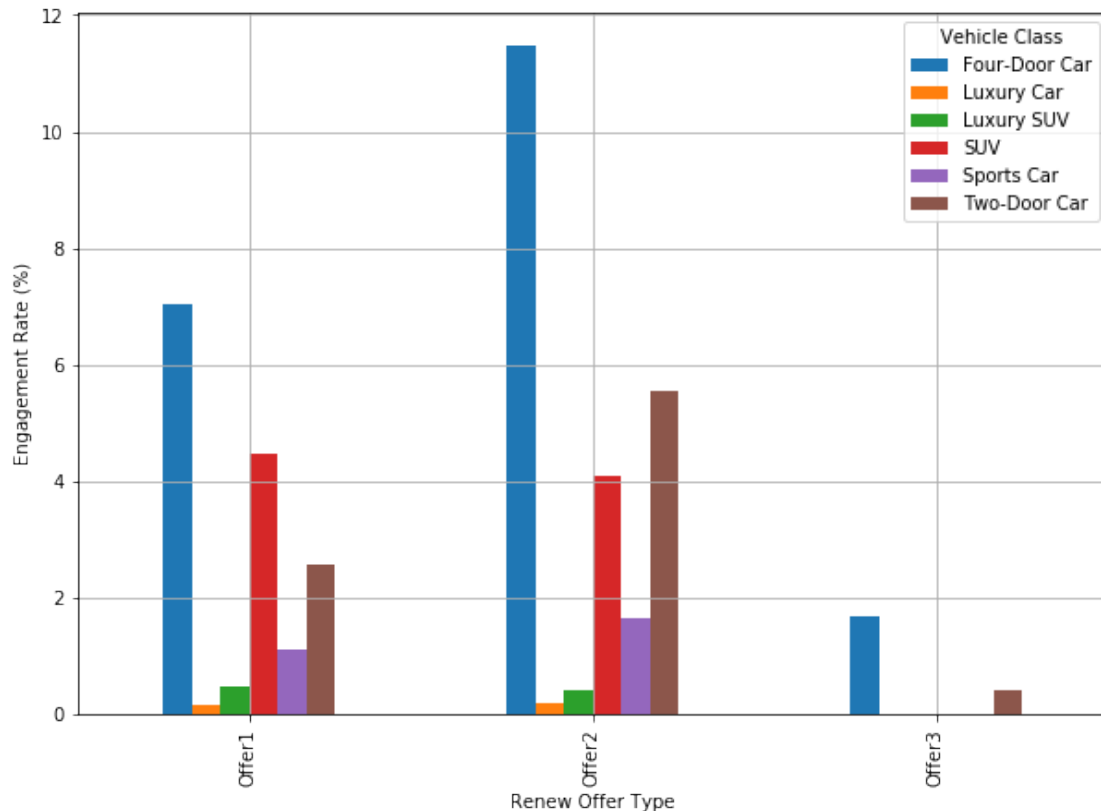
Vehicle Class      Two-Door Car
Renew Offer Type
Offer1              0.025586
Offer2              0.055366
Offer3              0.004190
```

```
In [14]: # Visualize this data in bar plot
```

```
ax = (by_offer_type_df*100.0).plot(
    kind='bar',
    figsize=(10, 7),
    grid=True
)
```

```
ax.set_ylabel('Engagement Rate (%)')
```

```
plt.show()
```



We already knew from the previous section “Engagement Rates by Offer Type” that Offer2 had the highest response rate among customers. Now we can add more insights by having broken down the customer attributes with the category “Vehicle class”: we can notice that customers with Four-Door Car respond more frequently for all offer types and that those with “Luxury SUV” respond with a higher chance to Offer1 than to Offer2. **If we have significantly difference in the response rates among different customer rates, we can fine-tune who to target for different set of offers.**

2.4 - Engagement Rates by Sales Channel

We are going to analyze how engagement rates differ by different sales channels.

```
In [15]: by_sales_channel_df = df.loc[
          df['Response'] == 'Yes'
        ].groupby([
          'Sales Channel'
        ]).count()['Customer']/df.groupby('Sales Channel').count()['Customer']

by_sales_channel_df
```

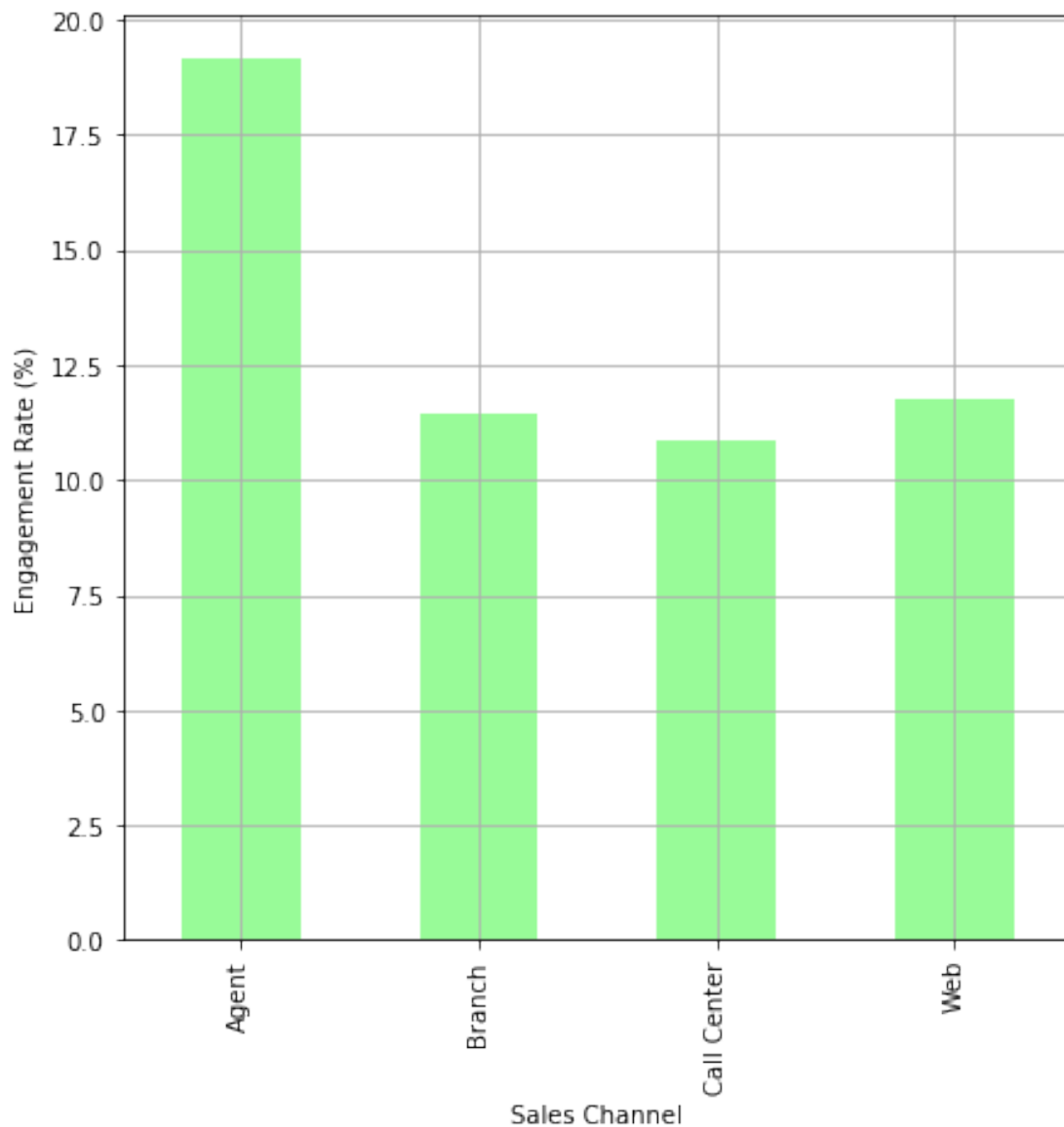
```
Out[15]: Sales Channel
Agent          0.191544
```

```
Branch      0.114531
Call Center  0.108782
Web         0.117736
Name: Customer, dtype: float64
```

```
In [16]: ax = (by_sales_channel_df*100.0).plot(
        kind='bar',
        figsize=(7, 7),
        color='palegreen',
        grid=True
    )
```

```
ax.set_ylabel('Engagement Rate (%)')
```

```
plt.show()
```



As we can notice, Agent works better in term of getting responses from the customers, and then sales through Web works the second best. Let's go ahead in breaking down this result deeper with different customers' attributes.

2.5 - Sales Channel & Vehicle Size

We are going to see whether customers with various vehicle sizes respond differently to different sales channels.

```
In [17]: by_sales_channel_df = df.loc[
        df['Response'] == 'Yes'
    ].groupby([
        'Sales Channel', 'Vehicle Size'
    ]).count()['Customer'] / df.groupby('Sales Channel').count()['Customer']

    by_sales_channel_df
```

```
Out[17]: Sales Channel  Vehicle Size
Agent                Large      0.020708
                  Medsize      0.144953
                  Small        0.025884
Branch              Large      0.021036
                  Medsize      0.074795
                  Small        0.018699
Call Center         Large      0.013598
                  Medsize      0.067989
                  Small        0.027195
Web                 Large      0.013585
                  Medsize      0.095094
                  Small        0.009057
Name: Customer, dtype: float64
```

```
In [18]: # Unstack the data into a more visible format
```

```
by_sales_channel_df = by_sales_channel_df.unstack().fillna(0)
by_sales_channel_df
```

```
Out[18]: Vehicle Size      Large  Medsize  Small
Sales Channel
Agent          0.020708  0.144953  0.025884
Branch         0.021036  0.074795  0.018699
Call Center    0.013598  0.067989  0.027195
Web            0.013585  0.095094  0.009057
```

```
In [19]: ax = (by_sales_channel_df*100.0).plot(
        kind='bar',
        figsize=(10, 7),
```

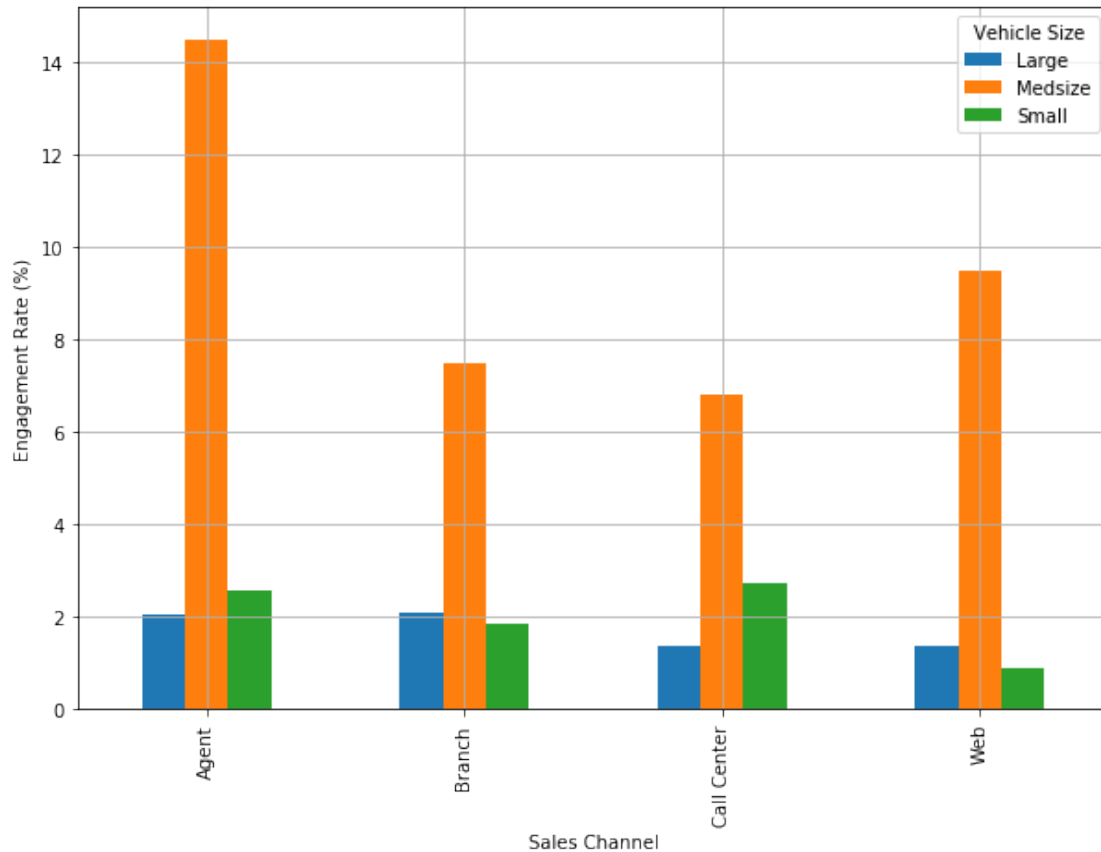
```

        grid=True
    )

    ax.set_ylabel('Engagement Rate (%)')

    plt.show()

```



As we can see, customers with medium size vehicles respond the best to all sales channels whereas the other customers differs slightly in terms of engagement rates across different sales channels.

2.6 - Engagement Rates by Months Since Policy Inception

```

In [20]: by_months_since_inception_df = df.loc[
        df['Response'] == 'Yes'
    ].groupby(
        by='Months Since Policy Inception'
    )['Response'].count() / df.groupby(
        by='Months Since Policy Inception'
    )['Response'].count() * 100.0

```

```
by_months_since_inception_df.fillna(0)
```

Out[20]: Months Since Policy Inception

| | |
|----|-----------|
| 0 | 14.457831 |
| 1 | 14.117647 |
| 2 | 20.224719 |
| 3 | 26.315789 |
| 4 | 19.780220 |
| 5 | 6.896552 |
| 6 | 0.000000 |
| 7 | 7.594937 |
| 8 | 7.407407 |
| 9 | 18.750000 |
| 10 | 15.789474 |
| 11 | 17.307692 |
| 12 | 6.000000 |
| 13 | 14.814815 |
| 14 | 0.000000 |
| 15 | 22.018349 |
| 16 | 0.000000 |
| 17 | 11.881188 |
| 18 | 13.333333 |
| 19 | 16.981132 |
| 20 | 11.650485 |
| 21 | 11.428571 |
| 22 | 12.903226 |
| 23 | 20.454545 |
| 24 | 21.951220 |
| 25 | 13.483146 |
| 26 | 15.000000 |
| 27 | 12.371134 |
| 28 | 17.475728 |
| 29 | 12.244898 |
| | ... |
| 70 | 23.529412 |
| 71 | 12.000000 |
| 72 | 23.762376 |
| 73 | 6.818182 |
| 74 | 19.780220 |
| 75 | 6.122449 |
| 76 | 6.976744 |
| 77 | 18.947368 |
| 78 | 7.317073 |
| 79 | 11.881188 |
| 80 | 16.438356 |
| 81 | 15.789474 |
| 82 | 0.000000 |

```

83      24.000000
84       6.000000
85      14.117647
86       0.000000
87       7.894737
88       7.894737
89      18.556701
90      14.285714
91       8.000000
92      16.216216
93      26.666667
94      25.000000
95      15.584416
96      17.910448
97       0.000000
98       0.000000
99       7.692308
Name: Response, Length: 100, dtype: float64

```

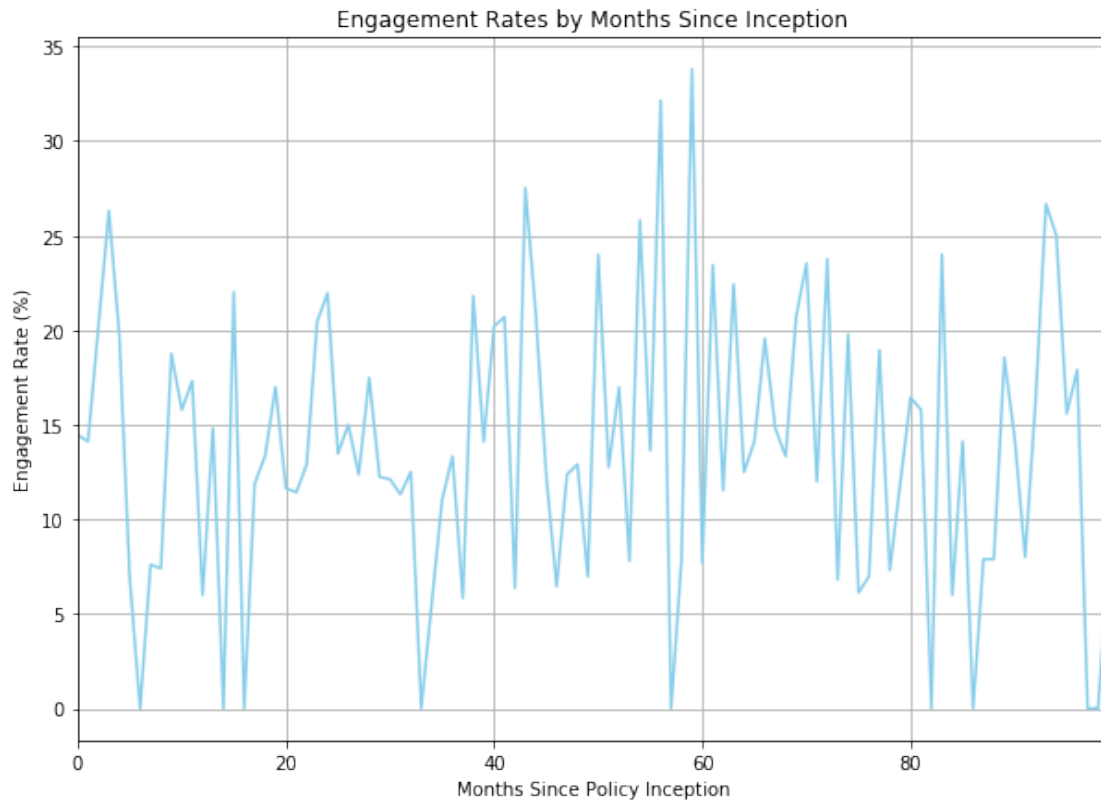
```

In [21]: ax = by_months_since_inception_df.fillna(0).plot(
        figsize=(10, 7),
        title='Engagement Rates by Months Since Inception',
        grid=True,
        color='skyblue'
    )

    ax.set_xlabel('Months Since Policy Inception')
    ax.set_ylabel('Engagement Rate (%)')

    plt.show()

```



3. Customer Segmentation by CLV & Months Since Policy Inception

We are going to segment our customer base by *Customer Lifetime Value* and *Months Since Policy Inception*.

In [22]: # Take a look at the distribution of the CLV

```
df['Customer Lifetime Value'].describe()
```

```
Out[22]: count      9134.000000
         mean      8004.940475
         std      6870.967608
         min      1898.007675
         25%      3994.251794
         50%      5780.182197
         75%      8962.167041
         max      83325.381190
         Name: Customer Lifetime Value, dtype: float64
```

For the previous output, we are going to define those customers with a CLV higher than the median as **high-CLV customers**, and those with a CLV lower than the median as **low-CLV customers**.

```

In [23]: df['CLV Segment'] = df['Customer Lifetime Value'].apply(
        lambda x: 'High' if x > df['Customer Lifetime Value'].median() else 'Low'
    )

In [24]: # Do the same procedure for Months Since Policy Inception

        df['Months Since Policy Inception'].describe()

Out[24]: count    9134.000000
         mean      48.064594
         std       27.905991
         min        0.000000
         25%       24.000000
         50%       48.000000
         75%       71.000000
         max       99.000000
         Name: Months Since Policy Inception, dtype: float64

In [25]: df['Policy Age Segment'] = df['Months Since Policy Inception'].apply(
        lambda x: 'High' if x > df['Months Since Policy Inception'].median() else 'Low'
    )

In [26]: df.head()

Out[26]:   Customer      State  Customer Lifetime Value  Response  Coverage  Education \
0  BU79786  Washington          2763.519279         No      Basic  Bachelor
1  QZ44356   Arizona          6979.535903         No  Extended  Bachelor
2  AI49188   Nevada          12887.431650         No   Premium  Bachelor
3  WW63253  California          7645.861827         No      Basic  Bachelor
4  HB64268  Washington          2813.692575         No      Basic  Bachelor

      Effective To Date  EmploymentStatus  Gender  Income  ...  Number of Policies \
0          2/24/11        Employed         F    56274  ...              1
1          1/31/11        Unemployed        F         0  ...              8
2          2/19/11        Employed         F    48767  ...              2
3          1/20/11        Unemployed        M         0  ...              7
4          2/3/11         Employed         M    43836  ...              1

      Policy Type      Policy  Renew Offer Type  Sales Channel \
0  Corporate Auto  Corporate L3          Offer1          Agent
1  Personal Auto   Personal L3          Offer3          Agent
2  Personal Auto   Personal L3          Offer1          Agent
3  Corporate Auto  Corporate L2          Offer1  Call Center
4  Personal Auto   Personal L1          Offer1          Agent

      Total Claim Amount  Vehicle Class  Vehicle Size  CLV Segment \
0          384.811147    Two-Door Car    Medsize          Low
1         1131.464935    Four-Door Car    Medsize          High
2          566.472247    Two-Door Car    Medsize          High

```

| | | | | |
|---|------------|---------------|---------|------|
| 3 | 529.881344 | SUV | Medsize | High |
| 4 | 138.130879 | Four-Door Car | Medsize | Low |

| | Policy Age Segment |
|---|--------------------|
| 0 | Low |
| 1 | Low |
| 2 | Low |
| 3 | High |
| 4 | Low |

[5 rows x 26 columns]

In [27]: *# Visualize these segments*

```
ax = df.loc[
    (df['CLV Segment'] == 'High') & (df['Policy Age Segment'] == 'High')
].plot.scatter(
    x='Months Since Policy Inception',
    y='Customer Lifetime Value',
    logy=True,
    color='red'
)

df.loc[
    (df['CLV Segment'] == 'Low') & (df['Policy Age Segment'] == 'High')
].plot.scatter(
    ax=ax,
    x='Months Since Policy Inception',
    y='Customer Lifetime Value',
    logy=True,
    color='blue'
)

df.loc[
    (df['CLV Segment'] == 'High') & (df['Policy Age Segment'] == 'Low')
].plot.scatter(
    ax=ax,
    x='Months Since Policy Inception',
    y='Customer Lifetime Value',
    logy=True,
    color='orange'
)

df.loc[
    (df['CLV Segment'] == 'Low') & (df['Policy Age Segment'] == 'Low')
].plot.scatter(
    ax=ax,
    x='Months Since Policy Inception',
```

```

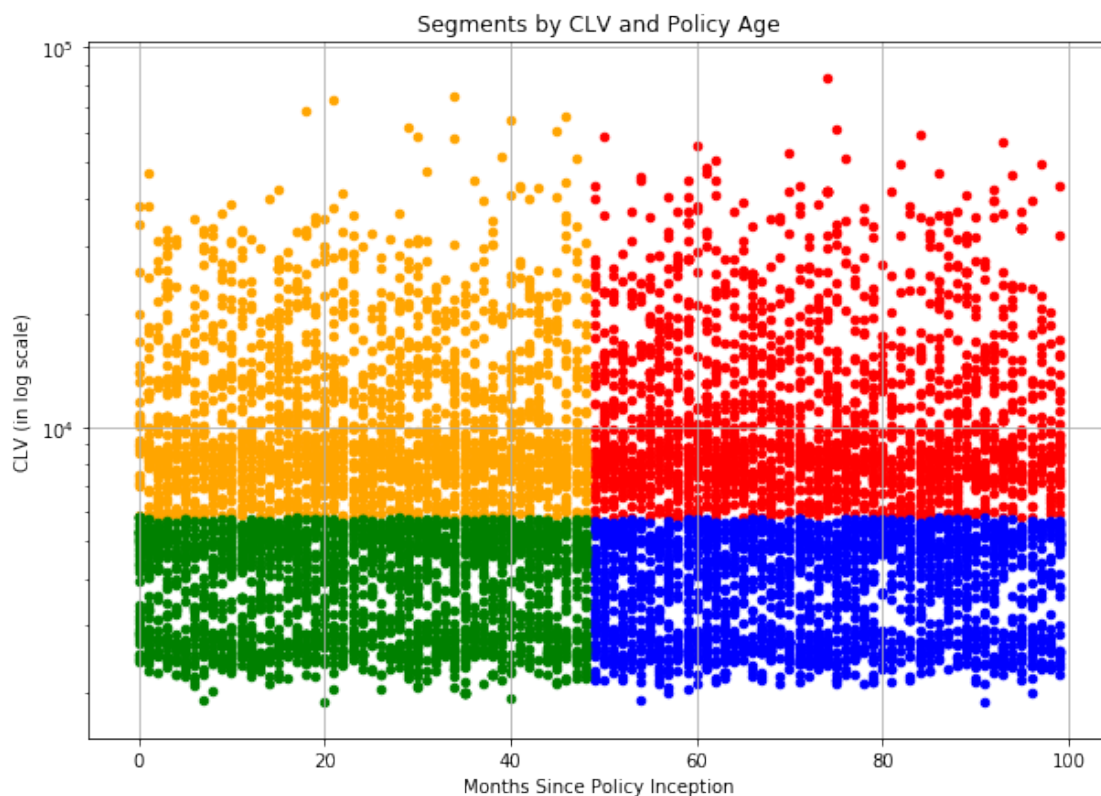
y='Customer Lifetime Value',
logy=True,
color='green',
grid=True,
figsize=(10, 7)
)

ax.set_ylabel('CLV (in log scale)')
ax.set_xlabel('Months Since Policy Inception')

ax.set_title('Segments by CLV and Policy Age')

plt.show()

```



logy=True transform the scale to log scale and it is often used for monetary values as they often have high skewness in their values. We have repeated the code for the `plot.scatter` 4 times because we have created 4 segments.

In [28]: *# See whether there is any noticeable difference in the engagement rates among these*

```

engagement_rates_by_segment_df = df.loc[
    df['Response'] == 'Yes'
].groupby([

```



```

        'CLV Segment', 'Policy Age Segment'
    ]).count()['Customer'] / df.groupby([
        'CLV Segment', 'Policy Age Segment'
    ]).count()['Customer']

```

```
engagement_rates_by_segment_df
```

```

Out[28]: CLV Segment  Policy Age Segment
High             High          0.138728
           Low          0.132067
Low             High          0.162450
           Low          0.139957
Name: Customer, dtype: float64

```

```
In [29]: # Look at these differences in a chart
```

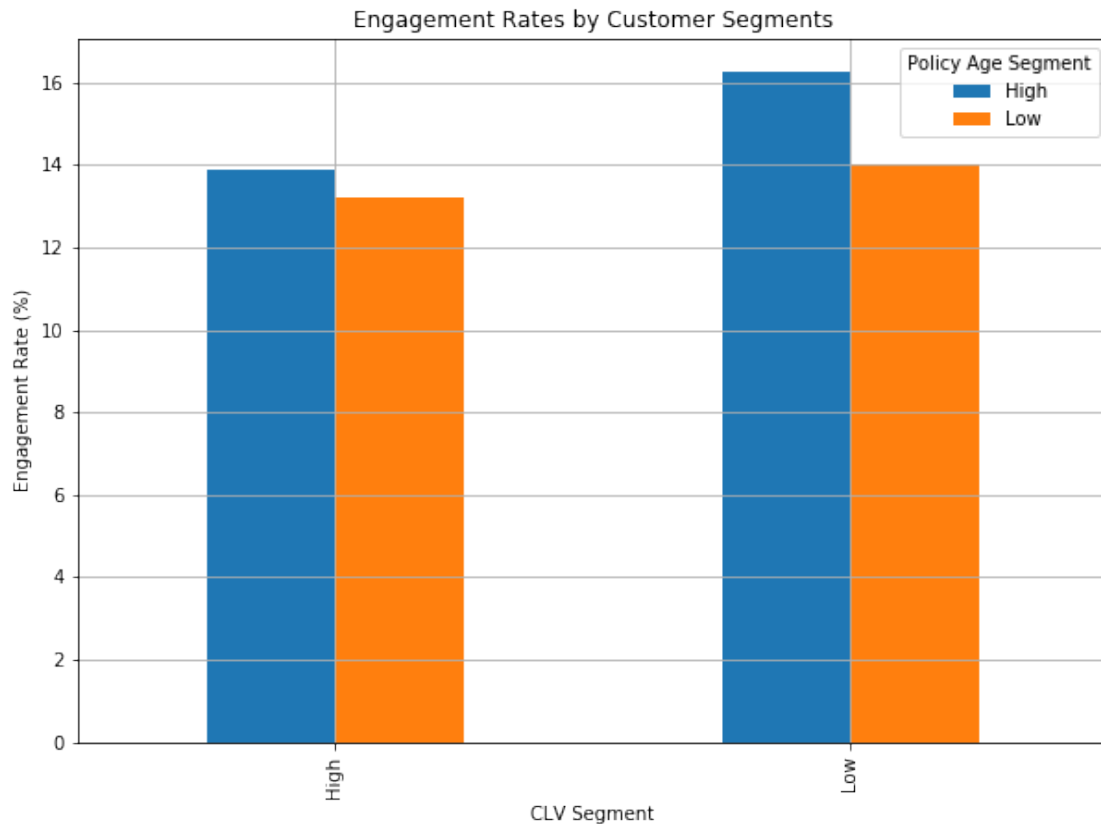
```

ax = (engagement_rates_by_segment_df.unstack()*100.0).plot(
    kind='bar',
    figsize=(10, 7),
    grid=True
)

ax.set_ylabel('Engagement Rate (%)')
ax.set_title('Engagement Rates by Customer Segments')

plt.show()

```



As we can notice, **High Policy Age Segment** has higher engagement than the **Low Policy Age Segment**. *This suggests that those customers who have been insured by this company longer respond better.* Moreover, the **High Policy Age and Low CLV segment** has the highest engagement rate among the four segments.

By creating different customer segments based on customer attributes, we can better understand how different groups of customers behave differently, and consequently, use this information to customize the marketing messages.

Follow me on [LinkedIn](#) for more:
Steve Nouri
<https://www.linkedin.com/in/stevenouri/>