Customer Personality Analysis

Presented by:

Kalyani Ambulkar, Brian Frank, Sandra Joseph, Amelia Mazer, Shaheen Nazar

Agenda

- Overview of Dataset
- Clean-Up Techniques
- Interesting Finding #1
- Interesting Finding #2
- Interesting Finding #3
- Questions?

Overview of Dataset

Customer Personality Analysis Dataset by Akash Patel

Customer Personality Analysis Dataset by Akash Patel

https://www.kaggle.com/imakash3011/customer-personality-analysis

Customer Personality Analysis is a detailed analysis of a company's ideal customers.

Benefits:

1. Helps a business to better understand its customers
2. Makes it easier for them to modify products according to the specific needs, behaviors and concerns of different types of customers.

Dataset

```
df.head()
```

ID	Year_Birth	Education	Marital_Status	Income	Kidhome	Teenhome	Dt_Customer	Recency	MntWines	MntFruits	MntMeatProducts	MntFishProducts	Mr
5524	1957	Graduation	Single	58138.0	0	0	04-09-2012	58	635	88	546	172	
2174	1954	Graduation	Single	46344.0	1	1	08-03-2014	38	11	1	6	2	
4141	1965	Graduation	Together	71613.0	0	0	21-08-2013	26	426	49	127	111	
6182	1984	Graduation	Together	26646.0	1	0	10-02-2014	26	11	4	20	10	
5324	1981	PhD	Married	58293.0	1	0	19-01-2014	94	173	43	118	46	

Dataset

```
df.head()
```

MntSweetProducts	MntGoldProds	NumDealsPurchases	NumWebPurchases	NumCatalogPurchases	NumStorePurchases	NumWebVisitsMonth	AcceptedCmp3	Ac
88	88	3	8	10	4	7	0	
1	6	2	1	1	2	5	0	
21	42	1	8	2	10	4	0	
3	5	2	2	0	4	6	0	
27	15	5	5	3	6	5	0	

Dataset

```
df.head()
```

urchases	NumWebVisitsMonth	AcceptedCmp3	AcceptedCmp4	AcceptedCmp5	AcceptedCmp1	AcceptedCmp2	Complain	Z_CostContact	Z_Revenue	Response
4	7	0	0	0	0	0	0	3	11	1
2	5	0	0	0	0	0	0	3	11	0
10	4	0	0	0	0	0	0	3	11	0
4	6	0	0	0	0	0	0	3	11	0
6	5	0	0	0	0	0	0	3	11	0

Clean-Up Techniques

Drop Columns and Rows

```
#drop the income rows that are null, as there are only 24, which represents 1% of the data
df2.dropna(how = 'any', subset =['Income'], inplace = True)
```

```
df2.isna().sum().sum()
```

```
0
```

```
#drop attributes that do not provide value to the analysis
df2.drop(['ID','Dt_Customer','Z_CostContact','Z_Revenue','Complain'], axis=1,  inplace = True)
```

```
df2.columns
```

```
Index(['Year_Birth', 'Education', 'Marital_Status', 'Income', 'Kidhome',
       'Teenhome', 'Recency', 'MntWines', 'MntFruits', 'MntMeatProducts',
       'MntFishProducts', 'MntSweetProducts', 'MntGoldProds',
       'NumDealsPurchases', 'NumWebPurchases', 'NumCatalogPurchases',
       'NumStorePurchases', 'NumWebVisitsMonth', 'AcceptedCmp3',
       'AcceptedCmp4', 'AcceptedCmp5', 'AcceptedCmp1', 'AcceptedCmp2',
       'Response'],
      dtype='object')
```

Replace Values

```
1 df2.groupby('Marital_Status')['Marital_Status'].size()
```

```
Marital_Status
Absurd         2
Alone          3
Divorced     232
Married      857
Single       471
Together     573
Widow         76
YOLO           2
Name: Marital_Status, dtype: int64
```

```
1 #Yolo, Absurd, and Alone are a small section of  the marital status column.
2 #replace those values to Single. All categories signify Single
3 df2.Marital_Status.replace(['YOLO','Absurd','Alone'], value = 'Single', inplace = True)
```

```
1 df2.Marital_Status.unique()
```

```
array(['Single', 'Together', 'Married', 'Divorced', 'Widow'], dtype=object)
```

Replace Values Continued

```
1 df2.groupby('Education')['Education'].size()
```

```
Education
2n Cycle       200
Basic           54
Graduation    1116
Master         365
PhD            481
Name: Education, dtype: int64
```

```
1 #2n Cycle is another name for master's, so replace 2n cycle with master
2 df2.Education.replace(to_replace ='2n Cycle', value = 'Master', inplace = True)
```

```
1 df2.Education.unique()
```

```
array(['Graduation', 'PhD', 'Master', 'Basic'], dtype=object)
```

Use get_dummies

```
#make the categorial data into numerical data
df2 = pd.get_dummies(df2,columns=['Marital_Status','Education'])
```

```
df2.head()
```

Marital_Status_Divorced	Marital_Status_Married	Marital_Status_Single	Marital_Status_Together	Marital_Status_Widow	Education_Basic	Education_Graduation
0	0	1	0	0	0	1
0	0	1	0	0	0	1
0	0	0	1	0	0	1
0	0	0	1	0	0	1
0	1	0	0	0	0	0

Education_Master	Education_PhD
0	0
0	0
0	0
0	0
0	1

Add Calculated Columns

```
df2['TotalChildren'] = (df2.Kidhome + df2.Teenhome)
```

```
df2['TotalCmpAccpt'] = (df2.AcceptedCmp3+df2.AcceptedCmp4+df2.AcceptedCmp5+df2.AcceptedCmp1+\
                        df2.AcceptedCmp2+df2.Response)
```

Change Column Names

```
#rename the attributes so they are more coordinated, shorter, and readable
df2.columns =['BirthYear',
              'Income',
              'Kids',
              'Teens',
              'Recency',
              'Wine',
              'Fruits',
              'Meat',
              'Fish',
              'Sweets',
              'Gold',
              'DealPurchases',
              'WebPurchases',
              'CatalogPurchases',
              'StorePurchases',
              'MonthWebVisits',
              'Campaign3',
              'Campaign4',
              'Campaign5',
              'Campaign1',
              'Campaign2',
              'Campaign6',
              'Divorced',
              'Married',
              'Single',
              'Together',
              'Widow',
              'Basic',
              'Graduation',
              'Master\'s',
              'Phd',
              'TotalChildren',
              'TotalCmpAccpt']
```

Sort Columns

```
list_col = df2.columns.tolist()
```

```
#sort the campaigns so it is easier to read
list_col[list_col.index('Campaign3'):list_col.index('Campaign6')]\
                = sorted(list_col[list_col.index('Campaign3'):list_col.index('Campaign6')])
```

```
#order the added columns
list_col.insert(df2.columns.get_loc('Teens')+1, list_col.pop(-2))
list_col.insert(df2.columns.get_loc('Campaign6')+2, list_col.pop(-1))
```

```
#assign the renamed and redorded columns to the clean dataframe
df2 = df2.reindex(columns= list_col)
```

Sorted Columns

```
1 df2.columns
```

```
Index(['BirthYear', 'Income', 'Kids', 'Teens', 'TotalChildren', 'Recency',
       'Wine', 'Fruits', 'Meat', 'Fish', 'Sweets', 'Gold', 'DealPurchases',
       'WebPurchases', 'CatalogPurchases', 'StorePurchases', 'MonthWebVisits',
       'Campaign1', 'Campaign2', 'Campaign3', 'Campaign4', 'Campaign5',
       'Campaign6', 'TotalCmpAccpt', 'Divorced', 'Married', 'Single',
       'Together', 'Widow', 'Basic', 'Graduation', 'Master's', 'Phd'],
      dtype='object')
```

Cleaned Data

```
df2.head()
```

	BirthYear	Income	Kids	Teens	TotalChildren	Recency	Wine	Fruits	Meat	Fish	Sweets	Gold	DealPurchases	WebPurchases	CatalogPurchases
0	1957	58138.0	0	0	0	58	635	88	546	172	88	88	3	8	10
1	1954	46344.0	1	1	2	38	11	1	6	2	1	6	2	1	1
2	1965	71613.0	0	0	0	26	426	49	127	111	21	42	1	8	2
3	1984	26646.0	1	0	1	26	11	4	20	10	3	5	2	2	0
4	1981	58293.0	1	0	1	94	173	43	118	46	27	15	5	5	3

Cleaned Data

```
df2.head()
```

StorePurchases	MonthWebVisits	Campaign1	Campaign2	Campaign3	Campaign4	Campaign5	Campaign6	TotalCmpAccpt	Divorced	Married	Single	Together
4	7	0	0	0	0	0	1	1	0	0	1	0
2	5	0	0	0	0	0	0	0	0	0	1	0
10	4	0	0	0	0	0	0	0	0	0	0	1
4	6	0	0	0	0	0	0	0	0	0	0	1
6	5	0	0	0	0	0	0	0	0	1	0	0

Interesting Finding 1

Sweets and Children Correlation

Negative Correlation between Children and Sweets

A negative correlation was found between the amount of sweets purchased and the total number of children. While it wasn't the highest inverse relationship, it seemed unusual.

```
corr.loc['TotalChildren'].nsmallest()
```

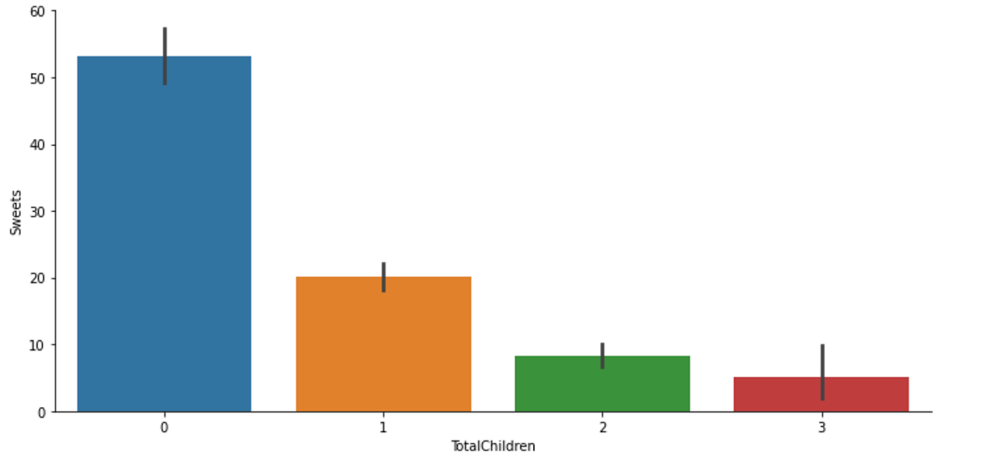
```
Meat               -0.504545
CatalogPurchases   -0.443474
Fish               -0.427841
Fruits             -0.395901
Sweets             -0.389411
Name: TotalChildren, dtype: float64
```

Plotting the Data

By graphing the relationship, it is clear that the more children you have the less you spend on sweets.

```
sns.catplot(x = 'TotalChildren',y='Sweets',kind ='bar',data=df2, aspect = 2)
```

```
<seaborn.axisgrid.FacetGrid at 0x7fa62d803880>
```



Interesting Finding 1 - Managerial Insight

71.44% of customers have at least one child.

To encourage these customers to buy more of sweets products, a discount on sweets can be given to these customers.

At the same time, offering more child-oriented sweets may also increase the sale of sweets to this demographic.

Another option to entice customers with children would be to provide other items besides sweets that parents and children would prefer.

Interesting Finding 2

The Relationship between Monthly Web Visits, Web Store Purchases, and Income

Clustering

Clustering showed that lower income was linked to lower web purchases and higher monthly web visits while higher income had the reverse relationship.

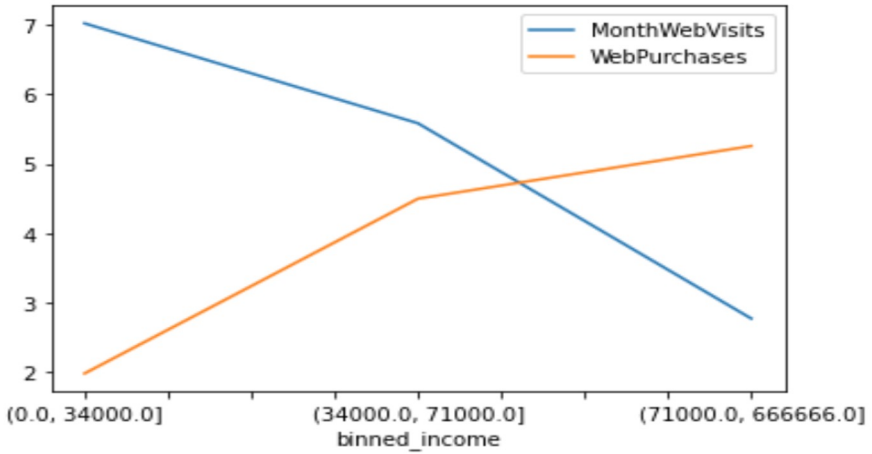
```
df_clust.groupby('Clustered')[['Income','WebPurchases','MonthWebVisits']].mean()
```

Clustered	Income	WebPurchases	MonthWebVisits
0	34765.867711	2.780679	6.647520
1	71072.096532	5.490159	3.888472

Plotting the Data

After binning income based on the clusters, the graph shows that the number of monthly web visits decreases with income range while the number of purchases increases.

```
df_clust.pivot_table(index = 'binned_income',\
                     values=['WebPurchases','MonthWebVisits']).plot()
```



Interesting Finding 2 - Managerial Insight

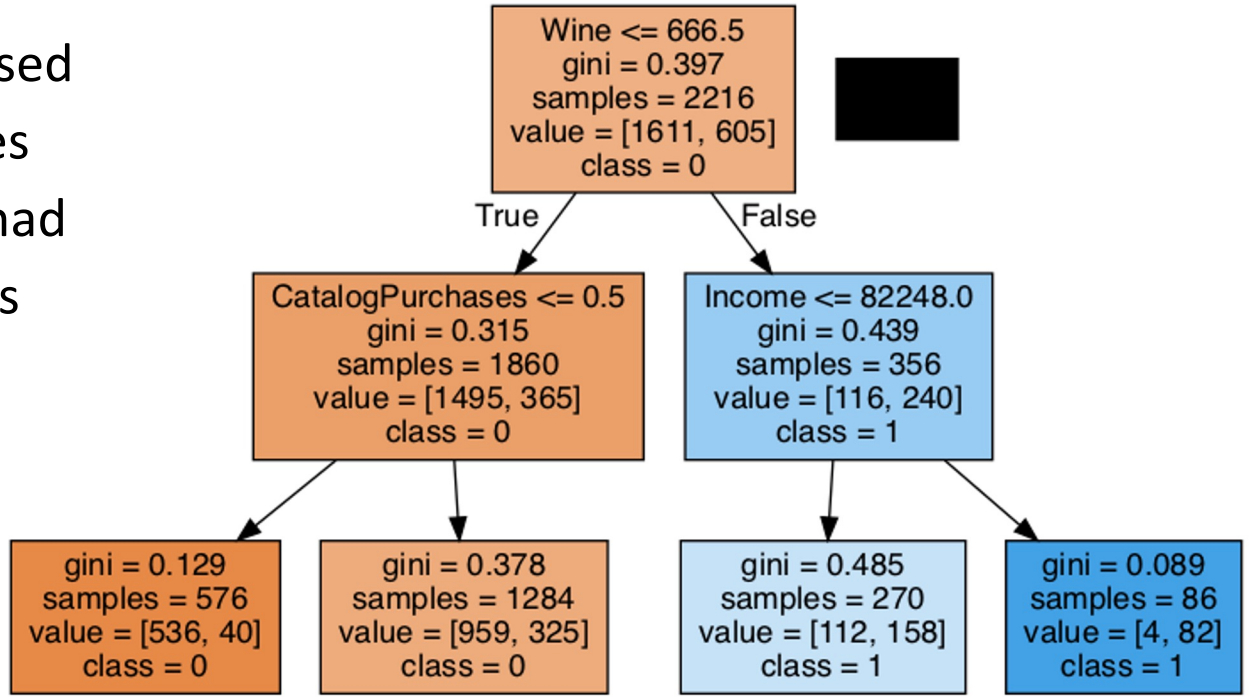
In order to increase the purchase rate of lower income customers, they should be targeted with online discounts after they have visited the website 3 times in one month.

Interesting Finding 3

Classify who has accepted a previous campaign

Decision Tree

A decision tree was used to find the attributes linked to those who had accepted a previous campaign.



Binning Wine and Income

Wine and Income were binned, partially based on the decision tree and partially through discovery.

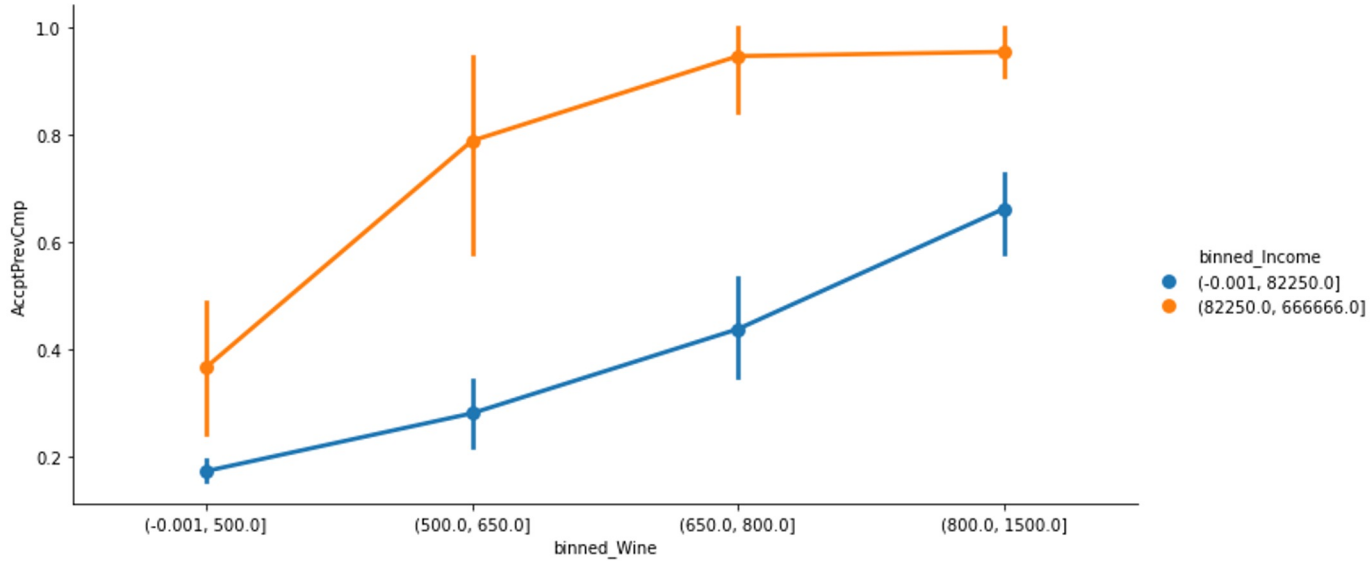
```
# Wine was partially binned based on the decision tree.
# The other binned levels were based on discovery.
df2['binned_Wine'] = pd.cut(df2.Wine,bins=[0,500,650,800,1500], include_lowest=True)

# Income was binned based on the decision tree.
df2['binned_Income'] = pd.cut(df2.Income,bins=[0,82250,666666],include_lowest=True)
```

Plotting the Data

The plotted data shows a clear distinction.

```
sns.catplot(data=df2, x='binned_Wine',y='AccptPrevCmp',hue='binned_Income',kind='point',aspect=2)
```

```
<seaborn.axisgrid.FacetGrid at 0x7fa62b8dadc0>
```



Interesting Finding 3 - Managerial Insight

In order to reduce campaign marketing costs and increase campaign participation rates, the next campaign should be targeted at those who have income over $82,250 and have purchased at least $500 in wine.

Another option would be to create an elite club for the customers who have purchased at least $500 in wine and target them with specific campaigns dedicated to high-end wine.

Conclusion

1. Customers with children need more encouragement to purchase sweets

2. Lower income customers visit the website more often but purchase from the website less, while higher income customers do the opposite

3. Higher income customers who purchased more wine are more likely to accept marketing campaigns

Questions?

Presented by:

Kalyani Ambulkar, Brian Frank, Sandra Joseph, Amelia Mazer, Shaheen Nazar