Optimizing marketing campaigns through RFM modeling and analysis of carry-over effects

Tianyu (Kevin) Gao

Table of Contents

Problem Statement

Marketing campaigns are important for companies to reach their target audience and inspire them to act. However, measuring the effectiveness of these campaigns is not always easy because of some confounding variables and randomness of the market, one of which is the carry-over effect of previous campaigns.

Carry-over effects are the influence of previous marketing campaigns on subsequent campaigns. Positive carry-over occurs when the earlier campaign had a positive impact on consumer attitudes, while negative carry-over occurs when the previous campaign had a negative impact, making it more difficult for the current campaign to succeed.

Many factors contribute to the carry-over effects. One of the main factors is the repetition of the message, which increases the ability to recall information about previous campaigns. Another factor is the level of involvement of the consumer with the brand or product. Highly engaged consumers tend to remember past events and build strong brand relationships.

The purpose of this study was to investigate the carry-over effect of past marketing campaigns on current campaigns. Specifically, this study examines how prior advertising messages and brand experience influence brand attitudes and purchase intentions in later campaigns. This study will help to understand the nature and extent of the impact of migration and provide marketers with insights on how to optimize their marketing strategies.

To solve this problem, we first did some data cleaning and imputation to prepare the data for analysis. After that, we generated some features based on the demographic data and consumption behavior to better distinguish the customers. After some comparison, we finally chose 6 as the number of clusters.

After clustering the customers, we conducted causal analyses on the carry-over effect of previous campaigns. We found the acceptance rates of some clusters are significantly higher than other clusters if accepted some certain earlier advertisements. This indicates the carry-over effects on the earlier campaigns.

Moreover, we have some discussion about the Recency-Frequency-Monetary value model for the clusters of the customers to improve the marketing strategy. We calculated how the recency and the frequency of the customers and personalized different strategies for customers of different Recency-Frequency scores.

Data

Dimensional Model

The database is made up of three tables, one fact table containing the transaction data of each time customer comes to the store, recording the amount for each category they spend and the methods they made the transaction, and two dimension tables, containing the demographic data and the campaign histories of the customers.

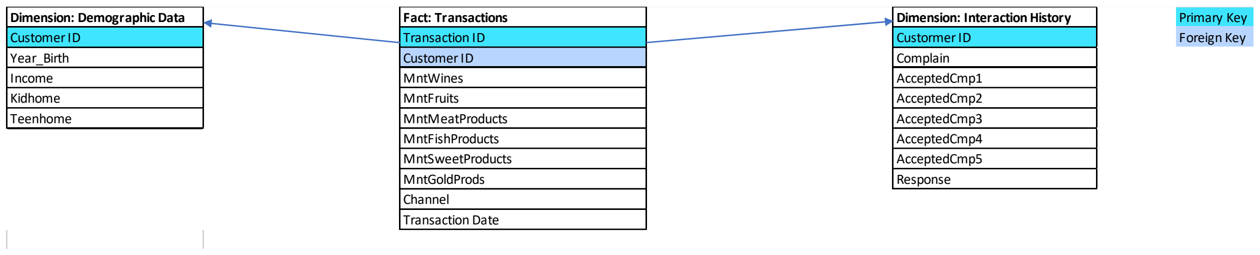


Exhibit1. Dimensional Model

After some computation, we figured out how frequently the customers come to our store and by which method they check out (Web/ Deal/ Catalog / Store) and we will use the data to build the RFM model.

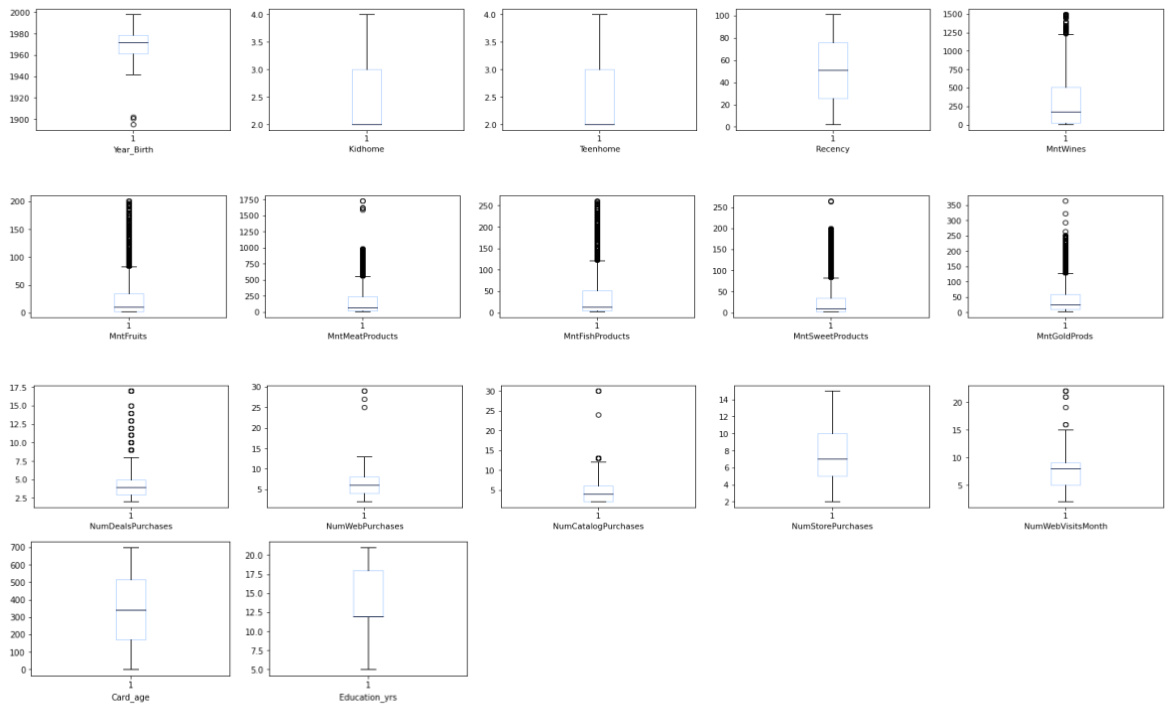
We can see that some of the variables are highly skewed and we did a log-transformation to normalize the data, below is the descriptive table.

	(1)	(2)	(3)	(4)	(5)	(6)	(7)
VARIABLES	mean	sd	min	max	Var	skewness	kurtosis
ID	5,592	3,247	0	11,191	1.054e+07	0.0398	1.810
Year_Birth	1,969	11.98	1,893	1,996	143.6	-0.350	3.713
Income	52,247	25,173	1,730	666,666	6.337e+08	6.759	162.3
Kidhome	0.444	0.538	0	2	0.290	0.635	2.219
Teenhome	0.506	0.545	0	2	0.297	0.407	2.013
Recency	49.11	28.96	0	99	838.8	-0.00199	1.798
MntWines	303.9	336.6	0	1,493	113,298	1.175	3.595
MntFruits	26.30	39.77	0	199	1,582	2.101	7.039
MntMeatProducts	166.9	225.7	0	1,725	50,947	2.082	8.502
MntFishProducts	37.53	54.63	0	259	2,984	1.918	6.087
MntSweetProducts	27.06	41.28	0	263	1,704	2.135	7.364
MntGoldProds	44.02	52.17	0	362	2,721	1.885	6.541
NumDealsPurchases	2.325	1.932	0	15	3.734	2.417	11.91
NumWebPurchases	4.085	2.779	0	27	7.721	1.382	8.688
NumCatalogPurchases	2.662	2.923	0	28	8.545	1.880	11.03
NumStorePurchases	5.790	3.251	0	13	10.57	0.702	2.377
NumWebVisitsMonth	5.317	2.427	0	20	5.889	0.208	4.815
AcceptedCmp3	0.0728	0.260	0	1	0.0675	3.289	11.82
AcceptedCmp4	0.0746	0.263	0	1	0.0690	3.239	11.49
AcceptedCmp5	0.0728	0.260	0	1	0.0675	3.289	11.82
AcceptedCmp1	0.0643	0.245	0	1	0.0602	3.553	13.62
AcceptedCmp2	0.0134	0.115	0	1	0.0132	8.466	72.68
Complain	0.00937	0.0964	0	1	0.00929	10.18	104.7
Response	0.149	0.356	0	1	0.127	1.970	4.882

Exhibit 2. Descriptive table for data we used

Data Cleaning

Outliers



Exihibit3 Boxplots of Numerical Variables

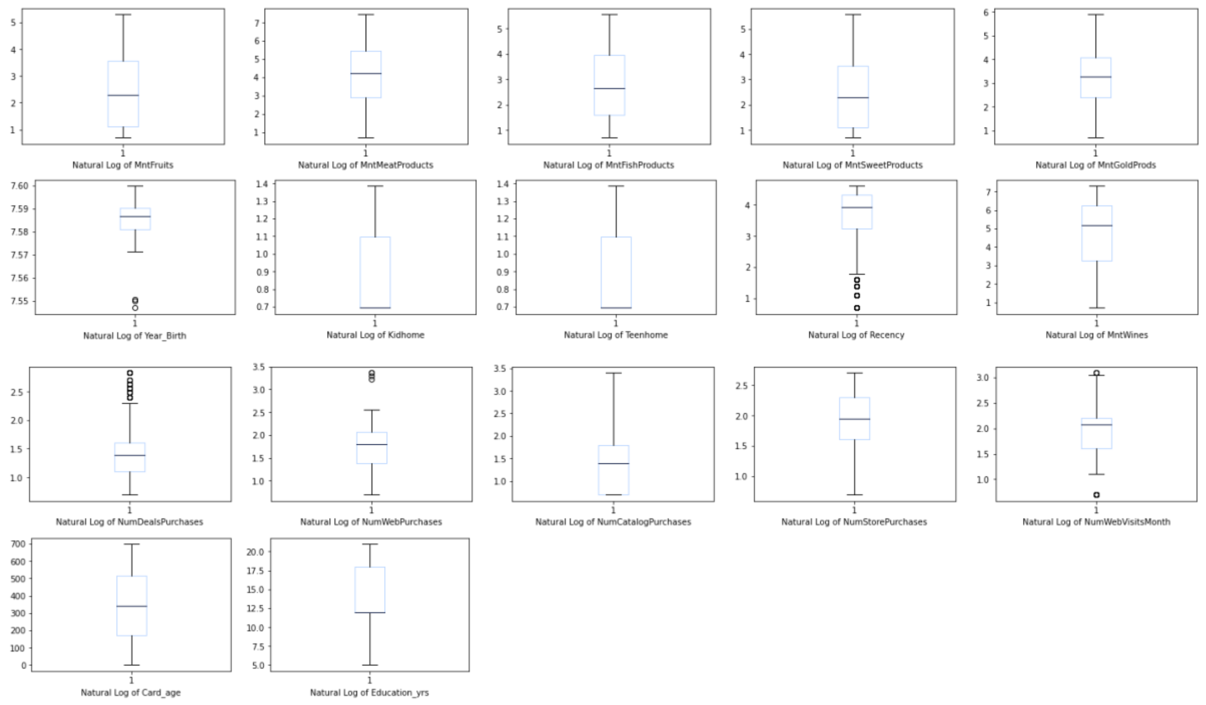


Exhibit4 Boxplots of Natural logs

Based on the boxplots, we can see that there are no outliers in the log-transformed numerical variables, and we need to consider the log transformation along with normalization because the scales of the variables vary a lot and some of the variables are extremely skewed.

Categorical Variables

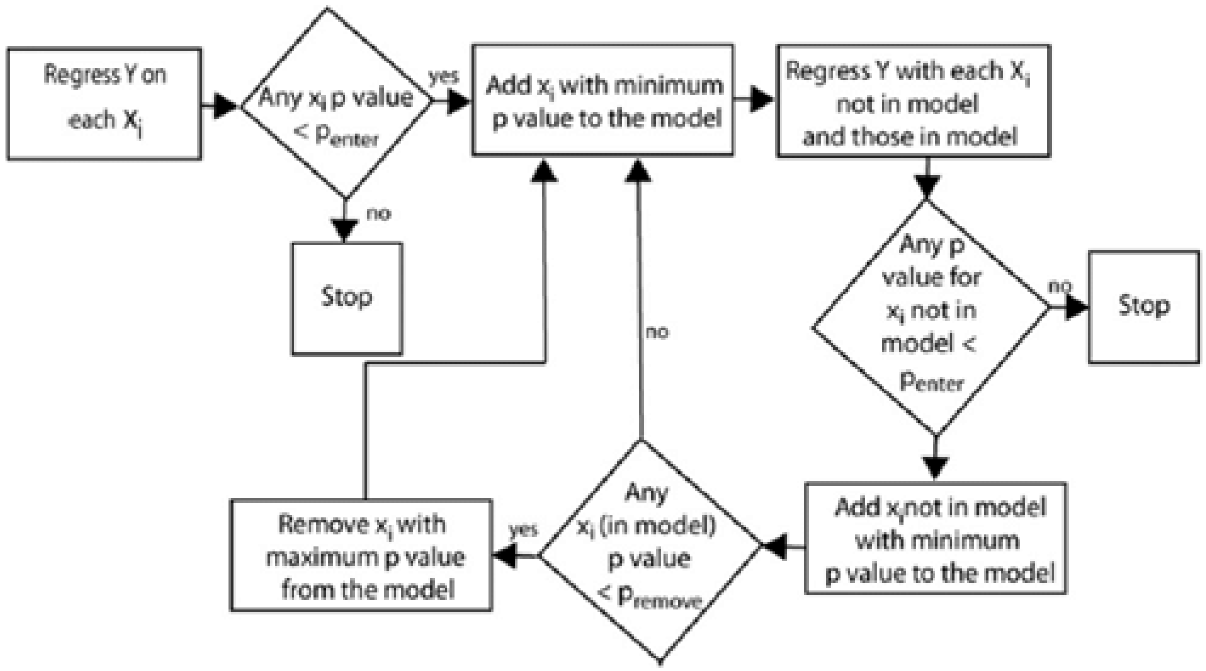
There are two categorical variables in the dataset, Education and Marital_Status. We used one-hot encoding to deal with the Marital_Status, and assigned different years in school to different Education level based on research of Ahmed Khalil(2023)

Missing Values

There are 24 observations with missing income in the dataset, and we plan to do a regression to impute the missing values.

Based on the research of Byrne et.al (2021), we know that consumer behavior is related to income, also we know that income is connected with demographic data, so we ran established a regression model to capture the correlation between the variables and income, and use the model to make predictions to fill the null values in the income column.

To avoid multi-collinearity, we used stepwise selection to select the variables in the model, which helps us make sure that all the variables in the model are significant. The flow chart of this algorithm is as follows:



Regress Y on each X_i
Any x_i p value < p_{enter}
yes
no
Stop
Add x_i with minimum p value to the model
Regress Y with each X_i not in model and those in model
Any p value for x_i not in model < P_{enter}
no
Stop
Add x_i not in model with minimum p value to the model
Any x_i (in model) p value < P_{remove}
yes
no
Remove x_i with maximum p value from the model

Exhibit 5 Flow Chart of Stepwise Selection

And the variables selected in the model are MntSweetProducts, Education_yrs, Recency, NumWebPurchases, NumCatalogPurchases, Kidhome, NumStorePurchases, NumWebVisitsMonth, MntWines, Teenhome, MntMeatProducts, Year_Birth, MntFruits, NumDealsPurchases.

The results of the final regression are as follows:

	(1)
VARIABLES	Income
MntSweetProducts	29.26***
	(5.991)
Education_yrs	166.5***
	(43.10)
Recency	-13.05**
	(6.306)
NumWebPurchases	1,579***
	(90.94)
NumCatalogPurchases	397.4***
	(106.4)
Kidhome	3,478***
	(461.4)
NumStorePurchases	543.7***
	(85.22)
NumWebVisitsMonth	-3,175***
	(108.8)
MntWines	15.94***
	(0.877)
Teenhome	6,392***
	(416.6)
MntMeatProducts	19.56***
	(1.373)
Year_Birth	-45.23***
	(16.97)
MntFruits	22.14***
	(6.154)
NumDealsPurchases	-1,057***
	(120.7)
Constant	133,418***
	(33,488)
Observations	2,216
R-squared	0.828

Exhibit 6 Regression results

Using this model, we filled in the missing income data.

Data Description

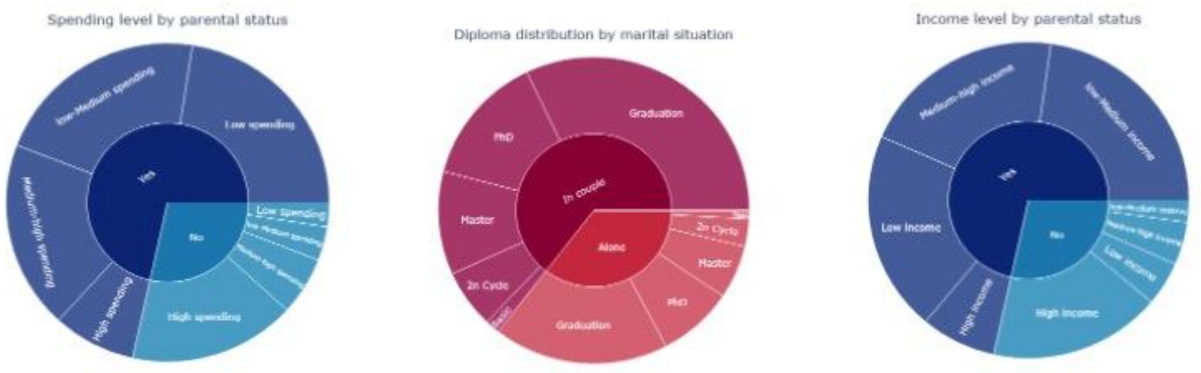


Exhibit 7 Pie Charts panel

The distribution of education seems to be identical for the two populations grouped by whether live alone. We are inclined to believe that there is no correlation between education and marital status.

People with high income and spending are largely representing the population who has no child. People having at least 1 child are mainly represented by people with low income. We could be tempted to believe that people having a high income tend to not have a child.

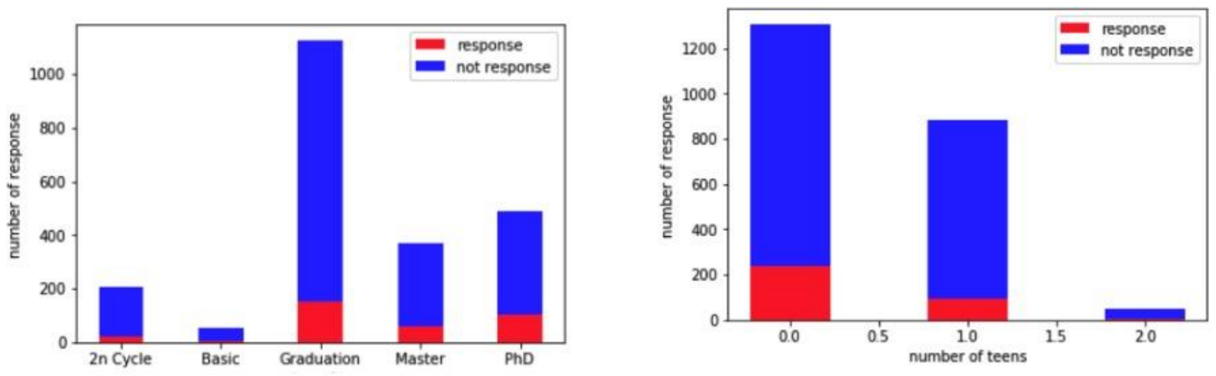


Exhibit 8 Stack Bar plots for Response Rates

Graduation customers have the highest response rate. And customers with basic education have the lowest.
Customers with no teens have a higher response rate. The response rate for customers with two children is almost zero.

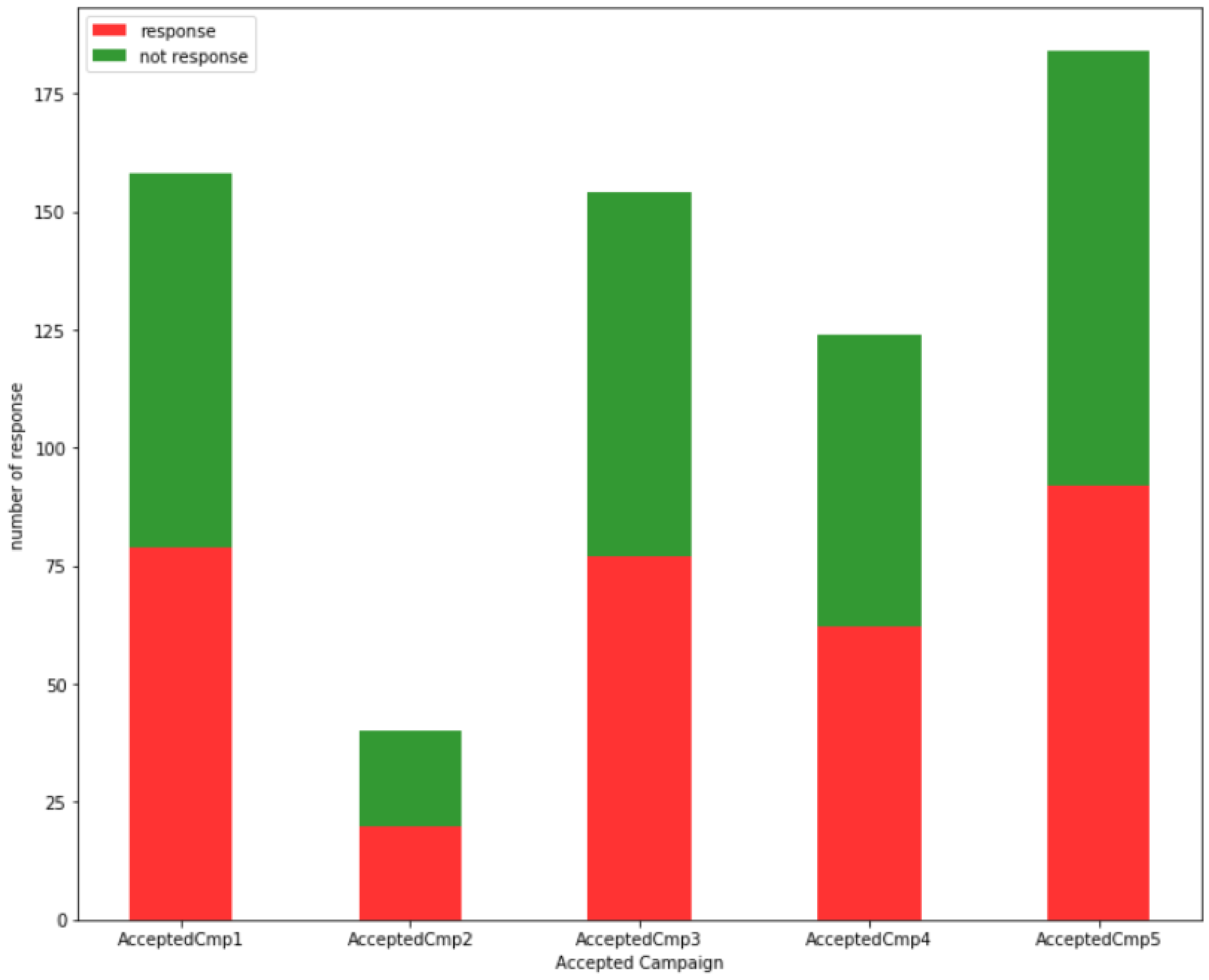


Exhibit 9 Stacked bars for different Campaigns

It seems that customers accepting each campaign have nearly the same response rate. Customers accepting campaign 2 have the lowest number of responses. Customers accepting campaign 5 have the highest number of responses.
However, this is for the whole population. Different campaigns may have different impacts on different kinds of customers. So, we should first cluster our customers, and then study the impact of each campaign on customer response.

Models

Hierarchical Clustering

We generated 6 clusters based on the demographic data and the consumption history of the customers. The algorithm we used is hierarchical clustering and the metric we used is L2 norm.

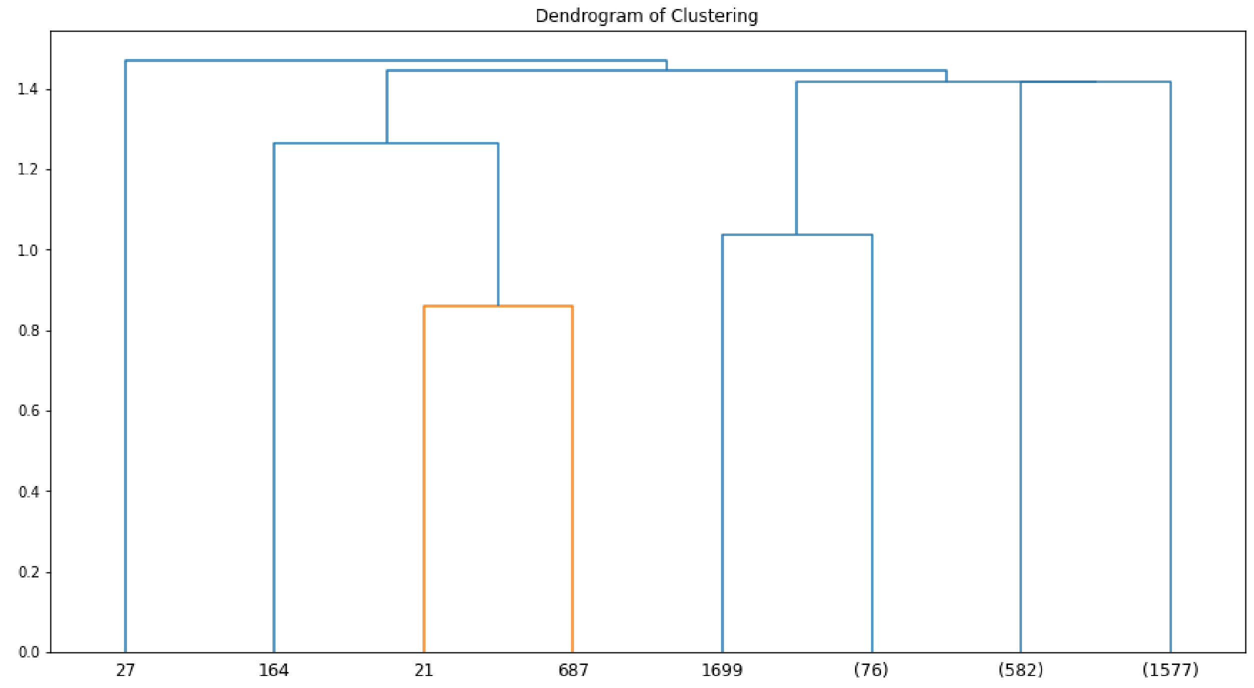


Exhibit 10 Dendrogram of hierarchical clustering

And the characteristics of the 6 clusters are as follows:

Cluster	Description	Cluster	Description
1	High-income meat consumers	4	Couples with low willingness to have babies
2	Aged middle class	5	Highly interactive
3	Living together with others, not married	6	Parents of Young kids

Exhibit 11 Description of all the clusters

A/B testing for carryover effect of campaigns

To measure the carryover effect in different clusters, we did A/B testing within different clusters, and the results with statistically significant positive lift are listed below.

Test	cluster	lift	Campaign Accepted by Control Group	Campaign Accepted by Treatment Group
1	5	0.8704	NA	2
25	5	0.8462	3	1, 2
144	4	0.7826	1, 2	1, 2, 3
4	5	0.5371	NA	4
191	2	0.5287	NA	1
77	3	0.5	1	1, 4
8	5	0.5	1	1, 4
27	5	0.4462	3	3, 4
2	5	0.4104	NA	3
0	5	0.3704	NA	1
134	4	0.3615	NA	2
159	4	0.3375	3	3, 4
80	3	0.3333	2	2, 3
78	3	0.2667	1	1, 2
72	3	0.2043	NA	2
74	3	0.2008	NA	4
136	4	0.1658	NA	4
133	4	0.1658	NA	1

Exhibit12 Experiments with statistically significant positive lift

Based on the results above, we can see that for cluster 2, if we want to maximize the conversion, we can promote more campaign 1 to the customers. To target the aged middle-class group, the store should try to cooperate with education service and financial intermediates to promote campaign 2. Similarly, store can be more inclined to promote all kinds of campaigns to target the unmarried people living with others, like cooperate with the social platforms and social medias. Also, it is a good idea to make customers much more interactive by adding more interesting and more interactive stuff to the campaign 2 to generate the highest conversion.

RFM Model

RFM(Recency, Frequency and Monetary value) analysis measures how recently, how often and how much money a customer has given to the business of interest. After RFM analysis, we can have a better understanding of the customer personas and how to maximize the value we can generate from the customers, and the potential to generate more values as well.

Based on the treemap below, we can figure out some insights for the customers:

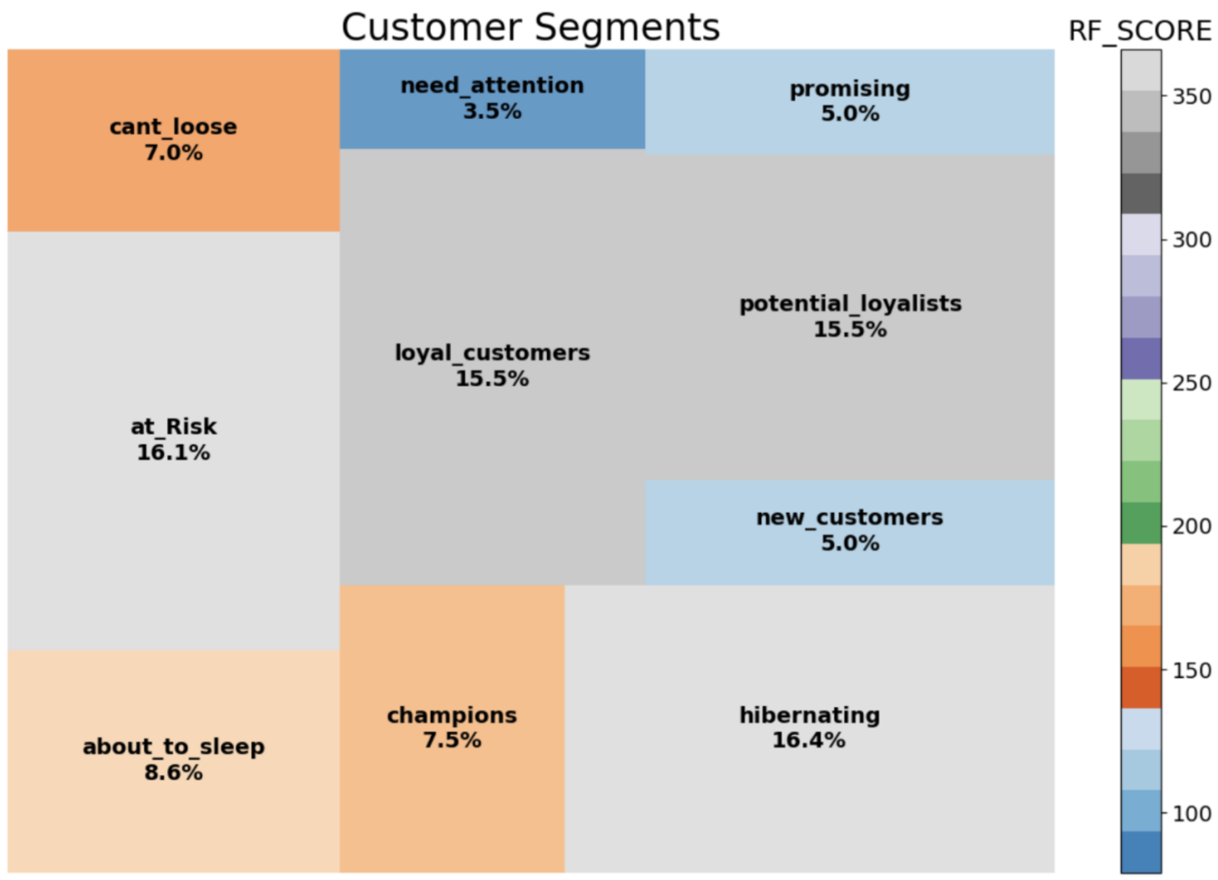


Exhibit 13 Treemap of the RF score

Hibernating Customers (16.4%): These customers represent the least engaged segment of our customer base, as they have not made a purchase in a long time, have a low spending history, and have placed a limited number of orders. To re-engage them, we should focus on offering relevant products, introducing special discounts, and rebuilding our brand value to create renewed interest in our offerings.

At-Risk Customers (16.1%): This group consists of customers who were once frequent purchasers and significant spenders but have not made any recent purchases. Our goal is to win them back by sending personalized emails to reconnect, offering attractive renewals, and providing helpful resources that remind them of the value our products and services provide.

Potential Loyalists (15.5%): These customers have recently made purchases, spent a substantial amount of money, and bought from us more than once. To encourage their loyalty, we should offer membership or loyalty programs and recommend additional products that complement their existing purchases.

Champions (7.5%): Our Champions are the most valuable customers who have recently made purchases, buy from us frequently, and spend the most money. To nurture their relationship with our brand, we should reward them for their loyalty and consider them as early adopters for new product releases. Additionally, we should encourage them to promote our brand within their network, as their endorsement can help us grow.

Conclusion

The project focused on exploring the implementation of data analytics in marketing campaigns, an area that has become increasingly important with the rise of big data and machine learning technologies. Specifically, we delved into the concept of carryover effect and how it can be leveraged to improve the conversion rate of later campaigns. Carryover effect refers to the impact that a previous marketing campaign has on the success of subsequent campaigns. By understanding how carryover effect works, we can design campaigns that build on the success of previous ones, resulting in higher conversion rates and better ROI.

In addition to exploring carryover effect, we also looked at how the RFM model can be used to maximize the monetization of customers. RFM stands for recency, frequency, and monetary value, and it is a commonly used framework for customer segmentation and targeting. By analyzing the recency and frequency of customer purchases, as well as the monetary value of those purchases, we can gain insights into which customers are most valuable to the business and tailor our marketing campaigns accordingly. By using the RFM model, we can increase customer engagement and loyalty, leading to higher revenues and profits for the business.

Overall, this project demonstrated the importance of data analytics in marketing campaigns and provided valuable insights into how carryover effect and the RFM model can be used to improve the effectiveness of marketing campaigns and drive business growth.

Reference:

Anitha, P., & Patil, M. M. (2019). RFM model for customer purchase behavior using K-Means algorithm. *Journal of King Saud University - Computer and Information Sciences*. https://doi.org/10.1016/j.jksuci.2019.12.011

Byrne, D. P., & Martin, L. A. (2021). Consumer search and income inequality. *International Journal of Industrial Organization*, 102716. https://doi.org/10.1016/j.ijindorg.2021.102716

Cheng, C.-H., & Chen, Y.-S. (2009). Classifying the segmentation of customer value via RFM model and RS theory. *Expert Systems with Applications*, *36*(3), 4176–4184. https://doi.org/10.1016/j.eswa.2008.04.003

Guney, S., Peker, S., & Turhan, C. (2020). A Combined Approach for Customer Profiling in Video on Demand Services Using Clustering and Association Rule Mining. *IEEE Access*, *8*, 84326–84335. https://doi.org/10.1109/access.2020.2992064

Khalil, A. (2023, January 22). *EDA and applying a Simple Regression*. Kaggle.com. https://www.kaggle.com/code/ahmedsalaheldin90/eda-and-applying-a-simple-regression

Kim, Y.-I., Ko, J.-M., Song, J.-J., & Choi, H. (2012). Repeated Clustering to Improve the Discrimination of Typical Daily Load Profile. *Journal of Electrical Engineering and Technology*, *7*(3), 281–287. https://doi.org/10.5370/jeet.2012.7.3.281

Natale, N., Pilo, F., Pisano, G., Troncia, M., Bignucolo, F., Coppo, M., Pesavento, N., & Turri, R. (2017). Assessment of typical residential customers load profiles by using clustering techniques. *2017 AEIT International Annual Conference*. https://doi.org/10.23919/aeit.2017.8240518

Olubi, O., Oniya, E., & Owolabi, T. (2021). Development of Predictive Model for Radon-222 Estimation in the Atmosphere using Stepwise Regression and Grid Search Based-Random Forest Regression. *Journal of the Nigerian Society of Physical Sciences*, 132–139. https://doi.org/10.46481/jnsps.2021.177

Safari, F., Safari, N., & Montazer, G. A. (2016). Customer lifetime value determination based on RFM model. *Marketing Intelligence & Planning*, *34*(4), 446–461. https://doi.org/10.1108/mip-03-2015-0060

Saldanha, R. (2019). *Marketing Campaign*. Www.kaggle.com. https://www.kaggle.com/datasets/rodsaldanha/arketing-campaign

Sawyer, A. G., & Semenik, R. J. (1978). Carryover Effects of Corrective Advertising. *ACR North American Advances*, *NA-05*. https://www.acrwebsite.org/volumes/9446/volumes/v05/NA-05

Appendix

Code for data pre-processing

In this part we did data wrangling as well as feature.

```
import pandas as pd
df = pd.read_excel('marketing_campaign.xlsx')
from datetime import datetime
df['Dt_Customer'] = df['Dt_Customer'].apply(lambda x: datetime.strptime(x, '%Y-%m-%d'))
df['Card_age'] = df['Dt_Customer'].apply(lambda x: (x- df['Dt_Customer'].min()).days)
df['Education_yrs'] = df['Education'].replace({'Basic': 5,
                                               '2n Cycle': 8,
                                               'Graduation': 12,
                                               'Master': 18,
                                               'PhD': 21})
marital_dummy = pd.get_dummies(df['Marital_Status'])
import seaborn as sns
import numpy as np
df1 = pd.concat([df, marital_dummy], axis = 1)
df1.drop(labels = ['Education', 'Dt_Customer', 'Marital_Status'], inplace = True, axis = 1)
df2 = df1.dropna()
df3 = df2[['Year_Birth',
'Kidhome',
'Teenhome',
'Recency',
'MntWines',
'MntFruits',
'MntMeatProducts',
'MntFishProducts',
'MntSweetProducts',
'MntGoldProds',
'NumDealsPurchases',
'NumWebPurchases',
'NumCatalogPurchases',
'NumStorePurchases',
'NumWebVisitsMonth',
'Complain',
'Card_age',
'Education_yrs',
'Absurd',
'Alone',
'Divorced',
'Married',
'Single',
'Together',
'Widow']]
import statsmodels.api as sm
df4 = pd.DataFrame(columns = df3.columns)
import pandas as pd
import numpy as np
from sklearn import datasets, linear_model
from sklearn.linear_model import LinearRegression
import statsmodels.api as sm
from scipy import stats
def OLS_Regression(X, y):
    lm = LinearRegression()
    lm.fit(X,y)
    params = lm.coef_
    predictions = lm.predict(X)
    variables = X.columns.tolist()

    newX = X
    MSE = (sum((y-predictions)**2))/(len(newX)-len(newX.columns))

    # Note if you don't want to use a DataFrame replace the two lines above with
    # newX = np.append(np.ones((len(X),1)), X, axis=1)
    # MSE = (sum((y-predictions)**2))/(len(newX)-len(newX[0]))

    var_b = MSE*(np.linalg.inv(np.dot(newX.T,newX)).diagonal())
    # print(newX)
    sd_b = np.sqrt(var_b)
    ts_b = params/ sd_b

    p_values =[2*(1-stats.t.cdf(np.abs(i),(len(newX)-len(newX.iloc[0])))) for i in ts_b]

    sd_b = np.round(sd_b,3)
    ts_b = np.round(ts_b,3)
    p_values = np.round(p_values,6)
    params = np.round(params,6)

    myDF3 = pd.DataFrame()
    myDF3['Variable'], myDF3["Coefficients"],myDF3["Standard Errors"],myDF3["t
values"],myDF3["Probabilities"] = [variables, params,sd_b,ts_b,p_values]
    return myDF3
def backward_elimination(X, y):
    alpha = .15
    output = OLS_Regression(X, y)
    while(X.shape[1] > 0):
        if(output['Probabilities'].max() < alpha):
            return output
        else:
            dropvar = output.loc[output['Probabilities'] == output['Probabilities'].max(),
'Variable'].tolist()[0]
            X = X.drop(dropvar, axis = 1)
            output = OLS_Regression(X, y)
def forward_selection(X, y, existed_variable, Candidate):
    alpha = .15
    prob = 1
    var = Candidate[0]
    for variable in Candidate:
        test = existed_variable
        test.append(variable)
        result = OLS_Regression(df3[test], y)
        p_temp = result.loc[result['Variable'] == variable, 'Probabilities'].values[0]
        test.remove(variable)
        if p_temp < prob:
            prob = p_temp
            var = variable
            print(prob)
            print(var)
    if prob < alpha:
        print(prob)
        existed_variable = list(set(existed_variable).union([var]))
        Candidate = list(set(Candidate)- set([var]))
    return(existed_variable, Candidate)
    import copy
    def stepwise(X, y):
        existed_variable = []
        Candidate = X.columns.tolist()
        full = copy.deepcopy(Candidate)
        while(Candidate):
            before = len(Candidate)
            print(before)
            existed_variable, Candidate = forward_selection(X[existed_variable], y, existed_variable,
Candidate)
            after = len(Candidate)
            print(after)
            noforward = (before == after)
            # print(noforward)
            X1 = X[existed_variable]
            output = backward_elimination(X1, y)
            X2 = X1[output['Variable'].tolist()]
            noback = (X2.shape == X1.shape)
            Candidate = list(set(full)- set(X2.columns.tolist()))
            existed_variable = X2.columns.tolist()
            # print(X2.columns.tolist())
            # print(len(Candidate))
            print(noback)
            print(noforward)
            print(len(Candidate))
            if noback and noforward:
                return X2
        return X2

    train = stepwise(df3, df2['Income'])
    lr = LinearRegression()
    lr.fit(train, df2['Income'])
    lr.score(train, df2['Income'])
    x_log1 = ['NumDealsPurchases', 'MntMeatProducts', 'Year_Birth',
           'NumStorePurchases', 'NumWebVisitsMonth',
           'NumWebPurchases', 'NumCatalogPurchases',
           'MntSweetProducts', 'MntWines']
    x_log2 = ['Education_yrs', 'NumWebVisitsMonth', 'NumStorePurchases',
             'NumCatalogPurchases', 'NumWebPurchases', 'NumDealsPurchases',
           'MntSweetProducts', 'MntMeatProducts', 'MntFruits', 'MntWines',
           'Year_Birth']
    trainx_log1 = train.copy(deep = True)
    trainx_log2 = trainx_log1.copy(deep = True)
    trainx_log2['MntWines'] = np.log(trainx_log2['MntWines']- trainx_log2['MntWines'].min()
+ .0001 )
    lrx1 = LinearRegression()
    lrx1.fit(trainx_log1, np.log(df2['Income']))
    lrx1.score(trainx_log1, np.log(df2['Income']))
    lrx2 = LinearRegression()
    lrx2.fit(trainx_log2, np.log(df2['Income']))
    lrx2.score(trainx_log2, np.log(df2['Income']))
    percentile_data = (train- train.min(axis = 0)) / (train.max(axis = 0)- train.min(axis = 0))
    from sklearn.model_selection import GridSearchCV as GSCV
    from sklearn.neighbors import KNeighborsRegressor as KNR
    from sklearn.svm import SVR
    KNR_params = {'n_neighbors':range(1, 50)}
    SVR_params = {'kernel':['linear', 'rbf'], 'C': np.arange(1, 1000, 100), 'epsilon': np.arange(.1,
1, .1)}
    SVR1 = SVR()
    KNR1 = KNR()
    GSCVSVR = GSCV(estimator = SVR1, param_grid = SVR_params, cv = 2, n_jobs =-1)
    GSCVSVR.fit(percentile_data, np.log(df2['Income']))
    GSCVKNR = GSCV(estimator = KNR1, param_grid = KNR_params, cv = 2, n_jobs =-1)
    GSCVKNR.fit(percentile_data, np.log(df2['Income']))
    SVR_params1 = {'kernel':['linear', 'rbf'], 'C': np.arange(1, 200, 20), 'epsilon': np.arange(.9,
1.9, .1)}
    GSCVSVR1 = GSCV(estimator = SVR1, param_grid = SVR_params1, cv = 2, n_jobs =-1)
    GSCVSVR1.fit(percentile_data, np.log(df2['Income']))
    GSCVSVR1.best_estimator_
    SVR_params1 = {'kernel':['linear', 'rbf'], 'C': np.arange(1, 41, 4), 'epsilon': np.arange(.01, .9, .1)}
    GSCVSVR1 = GSCV(estimator = SVR1, param_grid = SVR_params1, cv = 2, n_jobs =-1)
    GSCVSVR1.fit(percentile_data, np.log(df2['Income']))
    GSCVSVR1.best_estimator_
    SVR_params1 = {'kernel':['linear', 'rbf'], 'C': np.arange(25, 33, .8), 'epsilon':
np.arange(.01, .21, .02)}
    GSCVSVR1 = GSCV(estimator = SVR1, param_grid = SVR_params1, cv = 2, n_jobs =-1)
    GSCVSVR1.fit(percentile_data, np.log(df2['Income']))
    GSCVSVR1.best_estimator_
    df_pred = df.loc[df['Income'].isna()].copy(deep = True)
    df_pred = df_pred[trainx_log1.columns]
    for var in x_log1:
        df_pred[var] = np.log(df_pred[var] + 3)
    dfn = df1.copy(deep = True)
    dfn.loc[dfn['Income'].isna(), 'Income'] = np.exp(lrx1.predict(df_pred))
    behavioral = dfn.drop(['ID', 'Response', 'AcceptedCmp1', 'AcceptedCmp2', 'AcceptedCmp3',
'AcceptedCmp4', \
                          'AcceptedCmp5'], axis = 1)
    from scipy.cluster.hierarchy import dendrogram, linkage, fcluster
    l1 = linkage(behavioral)
    plt.figure(figsize = (15, 8))
    behavioral_scale = behavioral.copy(deep = True)
    behavioral_scale = (behavioral_scale- behavioral_scale.min(axis = 0)) /
(behavioral_scale.max(axis = 0)- behavioral_scale.min(axis = 0))
    l2 = linkage(behavioral_scale)
    plt.figure(figsize = (15, 8))
    d2 = dendrogram(l2, truncate_mode = 'level', p = 3)
    plt.title('Dendrogram of Clustering')
    f1 = fcluster(l2, criterion = 'maxclust', t = 6)
    dfn['cluster'] = f1
    dfn['History'] = (dfn['AcceptedCmp1'].astype('int')*1 + dfn['AcceptedCmp2'].astype('int')*2 +
dfn['AcceptedCmp3'].astype('int')*4 + dfn['AcceptedCmp4'].astype('int')*8 +
dfn['AcceptedCmp5'].astype('int')*16)
    dfn.to_csv('dfn.csv')
```

Code for Visualization

```
import numpy as np
import pandas as pd
%matplotlib inline
df=pd.read_csv("dfn.csv")
for i in range(3):
    plt.figure(figsize = (25, 3))
    for j in range(5):
        ax = plt.subplot(1, 5, j + 1)
        ax.boxplot(df[vlist[i * 5 + j]] + 2, boxprops = dict(color = '#b8d5ff'), medianprops = dict(color
= '#05133d'))
        ax.set_xlabel(f"{vlist[i * 5 + j]}")
    plt.show()

plt.figure(figsize = (25, 3))
for i in range(15, 17):
    ax = plt.subplot(1, 5, i % 5 + 1)
    ax.boxplot(df[vlist[i]], boxprops = dict(color = '#b8d5ff'), medianprops = dict(color = '#05133d'))
    ax.set_xlabel(f'{vlist[i]}')
plt.show()
for i in range(3):
    plt.figure(figsize = (25, 3))
    for j in range(5):
        ax = plt.subplot(1, 5, j + 1)
        ax.boxplot(np.log(df[vlist[i * 5 + j]] + 2), boxprops = dict(color = '#b8d5ff'), medianprops =
dict(color = '#05133d'))
        ax.set_xlabel(f"Natural Log of {vlist[i * 5 + j]}")
    plt.show()

plt.figure(figsize = (25, 3))
for i in range(15, 17):
    ax = plt.subplot(1, 5, i % 5 + 1)
    ax.boxplot(df[vlist[i]], boxprops = dict(color = '#b8d5ff'), medianprops = dict(color = '#05133d'))
    ax.set_xlabel(f'Natural Log of {vlist[i]}')
plt.show()
import numpy as np
from numpy import isnan
import pandas as pd
from sklearn.impute import KNNImputer
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression
import plotly.graph_objs as go
import plotly.express as px
```

```
from scipy.stats import shapiro
from scipy.stats import chi2_contingency
from scipy.stats import chi2
import scipy.stats as stats
from numpy import median
from numpy import std
from IPython.display import Image
import os
import warnings
warnings.filterwarnings('ignore')

# total spent according to maritalstatus
df = dataset[["MntWines", "MntFruits", "MntMeatProducts", "MntFishProducts",
"MntSweetProducts", "MntGoldProds",
'Marital_Situation']].groupby(["Marital_Situation"]).sum().reset_index().sort_values(by=["MntWi
nes", "MntFruits", "MntMeatProducts", "MntFishProducts", "MntSweetProducts",
"MntGoldProds"], ascending=False)

fig = px.bar(df, x='Marital_Situation', y=["MntWines", "MntFruits", "MntMeatProducts",
"MntFishProducts", "MntSweetProducts", "MntGoldProds"])
fig.update_layout(xaxis_title="Marital Situation", yaxis_title="Total Spent")
fig.update_layout(
    title={
        'text': "Spend category by marital situation",
        'y':0.95,
        'x':0.5,
        'xanchor': 'center',
        'yanchor': 'top',
    }
)
fig.show()

# total spent according to Education
df = dataset[["MntWines", "MntFruits", "MntMeatProducts", "MntFishProducts",
"MntSweetProducts", "MntGoldProds",
'Education']].groupby(["Education"]).sum().reset_index().sort_values(by=["MntWines",
"MntFruits", "MntMeatProducts", "MntFishProducts", "MntSweetProducts", "MntGoldProds"],
ascending=False)

fig = px.bar(df, x='Education', y=["MntWines", "MntFruits", "MntMeatProducts",
"MntFishProducts", "MntSweetProducts", "MntGoldProds"])
fig.update_layout(xaxis_title="Education", yaxis_title="Total Spent")
fig.update_layout(
    title={
        'text': "Spend category by Education",
        'y':0.95,
        'x':0.5,
        'xanchor': 'center',
        'yanchor': 'top',
    }
)
fig.show()

df = dataset[['Education','Marital_Situation']]

fig = px.sunburst(df,
path=['Marital_Situation','Education'],color_discrete_sequence=px.colors.diverging.Spectral)
fig.update_layout(
    title={
        'text': "Diploma distribution by marital situation",
        'y':0.95,
        'x':0.5,
        'xanchor': 'center',
        'yanchor': 'top',
    }
)
fig.show()

#Creating 4 quartiles to segment Income
cut_labels_Income = ['Low income', 'low-Medium income', 'Medium-high income', 'High income']
dataset['Income_bins'] = pd.qcut(dataset['Income'], q=4,labels=cut_labels_Income)

df = dataset[['Income_bins','Has_child']]

fig = px.sunburst(df,
path=['Has_child','Income_bins'],color_discrete_sequence=px.colors.diverging.Portland)
fig.update_layout(
    title={
        'text': "Income level by parental status",
        'y':0.95,
        'x':0.5,
        'xanchor': 'center',
        'yanchor': 'top',
    }
)
fig.show()

#Creating 4 quartiles to segment Spending
```

```
cut_labels_spending = ['Low spending', 'low-Medium spending', 'Medium-high spending', 'High
spending']
dataset['spending_bins'] = pd.qcut(dataset['Spending'], q=4,labels=cut_labels_spending)

df = dataset[['spending_bins','Has_child']]

fig = px.sunburst(df,
path=['Has_child','spending_bins'],color_discrete_sequence=px.colors.diverging.Portland)
fig.update_layout(
   title={
      'text': "Spending level by parental status",
      'y':0.95,
      'x':0.5,
      'xanchor': 'center',
      'yanchor': 'top',
   }
)
fig.show()

df = dataset[['Income','Education']]
reg = LinearRegression().fit(np.vstack(dataset['Spending']), dataset['Income'])
df['bestfit'] = reg.predict(np.vstack(dataset['Spending']))

fig = go.Figure(data=go.Scatter(name='observations',x=dataset['Spending'],
y=dataset['Income'],mode='markers'))
fig.add_trace(go.Scatter(name='line of best fit', x=dataset['Spending'], y=df['bestfit'],
mode='lines'))
fig.update_traces(hovertemplate='Spending: %{x} <br>Income: %{y}')
fig.show()
```

Code for A/b testing

```
import pandas as pd
from scipy.stats import mannwhitneyu
dfn = pd.read_csv("dfn.csv")
def decomposite(x):
   li = []
   text = ''
   cnt = 1
   while x > 0:

      module = x % 2
      x = x- module
      x = x / 2
      if module == 1:
           li.insert(0, cnt)
      cnt += 1
   while li:
      text += f' campaign{str(li.pop(-1))},'
   return text

   dfn['history_diff'] = dfn['History'].diff()
   groups = dfn.groupby(['cluster', 'history_diff'])
   response_rates = groups['Response'].mean()
   # control_group = []
   # experiment_group = []
   records = []
   alpha = .05
   for cluster in dfn['cluster'].unique():
      for i in range(33):
           # print(cluster)
           if i in dfn.loc[dfn['cluster'] == cluster, 'History'].values:
               control_group = dfn.loc[(dfn['cluster'] == cluster) & (dfn['History'] == i),
'Response'].values
           for history_diff in [1, 2, 4, 8, 16]:
               if i + history_diff in dfn.loc[dfn['cluster'] == cluster, 'History'].values:
                   experiment_group = dfn.loc[(dfn['cluster'] == cluster) & (dfn['History'] == i +
history_diff), 'Response'].values
                   lift = round(experiment_group.mean()- control_group.mean(), 4)
                   p = mannwhitneyu(experiment_group, control_group)[1]
                   records.append([cluster, i, history_diff, lift, p])
                   if i == 0:
                       print(f'For cluster {cluster}, compared to observations did not accept any
campaigns, the campaign {str([1, 2, 4, 8, 16].index(history_diff) + 1)} leads to a lift in the response
rate of the final campaign of {str(lift)}, and the boost is {"statistically significant" if p < alpha else
"not statistically significant"}')
                   else:
                       print(f'For cluster {cluster}, observations accepted the{decomposite(i)} the
campaign {str([1, 2, 4, 8, 16].index(history_diff) + 1)} leads to a lift in the response rate of the
final campaign of {str(lift)}, and the boost is {"statistically significant" if p < alpha else "not
statistically significant"}')
   record = pd.DataFrame(records, columns = ['cluster', 'control', 'history_diff', 'lift', 'p'])
```