



# MARKETING CAMPAIGN EFFECTIVENESS

SAROSHMI B (211701047)  
SHARON STEVE J (211701050)

# AGENDA

- 1 Abstract
- 2 Understanding the data
- 3 Data Cleaning & Preprocessing
- 4 EDA
- 5 Model Building & Evaluation
- 6 Hyperparameter Tuning
- 7 Model Prediction
- 8 Summary & Conclusion

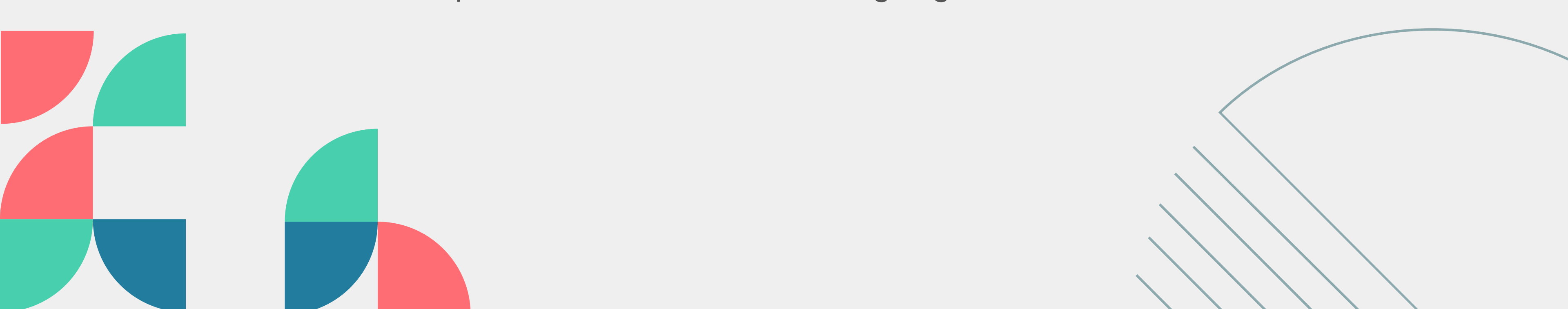
# ABSTRACT

This project focuses on creating a ***predictive machine learning model to forecast customer responses to specific product or service offers***. By analyzing demographic, behavioral, and campaign-related data, the model will identify patterns that indicate the likelihood of customer acceptance, enabling highly targeted and personalized marketing efforts. ***The goal is to maximize campaign effectiveness and return on investment by concentrating resources on customers most likely to engage***. Through data-driven insights, the project aims to improve customer segmentation, refine offer timing, and enhance overall marketing strategy. Deliverables include ***a predictive model and an intuitive dashboard for visualizing insights***, facilitating real-time decision-making and ongoing campaign optimization. This solution is designed to empower marketing teams with actionable intelligence, aligning offers with customer preferences to enhance satisfaction and drive sustained revenue growth.



# UNDERSTANDING THE DATA

Here lets focus on the attributes of the dataset to get a clear picture of the dataset we are going to work on.



# VIEW OF DATASET - CUSTOMER

ID: Customer's unique identifier

Year\_Birth: Customer's birth year

Education: Customer's education level

Marital\_Status: Customer's marital status

Income: Customer's yearly household income

Kidhome: Number of children in customer's household

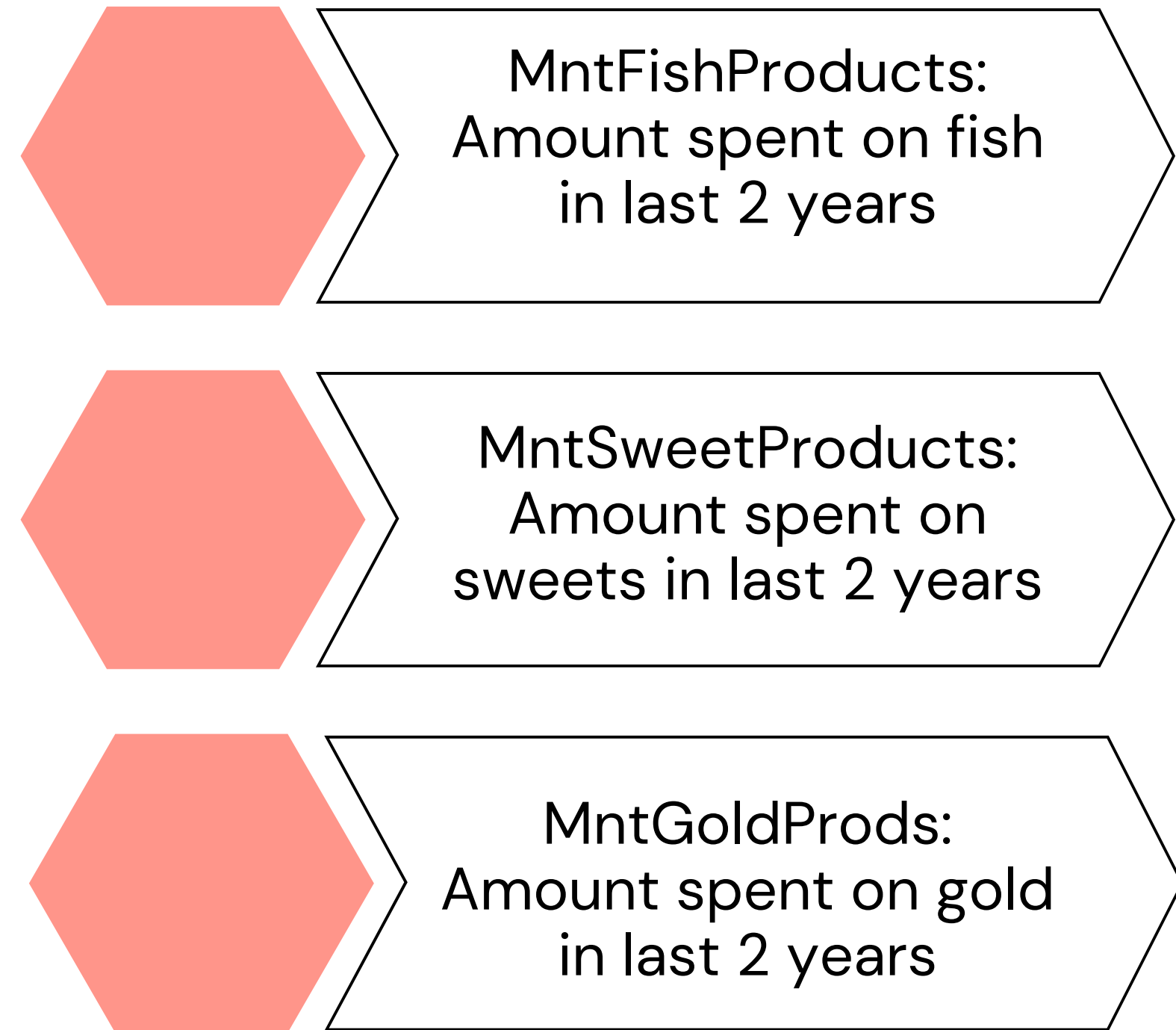
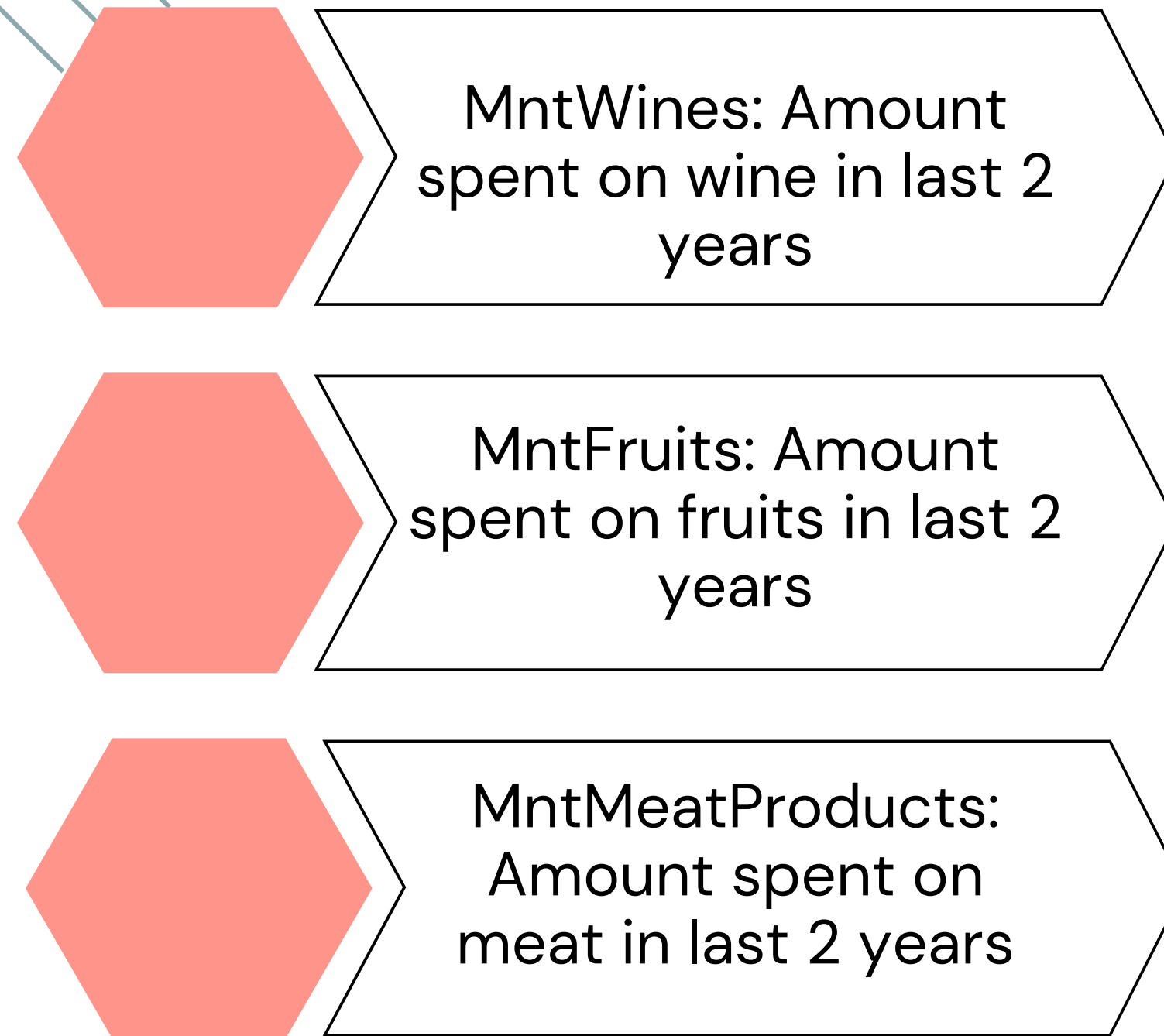
Teenhome: Number of teenagers in customer's household

Dt\_Customer: Date of customer's enrollment with the company

Recency: Number of days since customer's last purchase

Complain: 1 if the customer complained in the last 2 years, 0 otherwise

# VIEW OF DATASET - PRODUCTS



# VIEW OF DATASET - CAMPAIGN

- **NumDealsPurchases:** Number of purchases made with a discount
- **AcceptedCmp1:** 1 if customer accepted the offer in the 1st campaign, 0 otherwise
- **AcceptedCmp2:** 1 if customer accepted the offer in the 2nd campaign, 0 otherwise
- **AcceptedCmp3:** 1 if customer accepted the offer in the 3rd campaign, 0 otherwise
- **AcceptedCmp4:** 1 if customer accepted the offer in the 4th campaign, 0 otherwise
- **AcceptedCmp5:** 1 if customer accepted the offer in the 5th campaign, 0 otherwise
- **Response:** 1 if customer accepted the offer in the last campaign, 0 otherwise

# VIEW OF DATASET - CHANNELS

- **NumWebPurchases:**  
Number of purchases made through the company's website
- **NumCatalogPurchases:**  
Number of purchases made using a catalogue
- **NumStorePurchases:**  
Number of purchases made directly in stores
- **NumWebVisitsMonth:**  
Number of visits to company's website in the last month



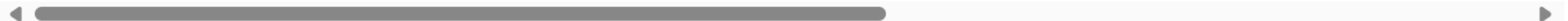
# GLIMPSE OF THE DATASET

```
In [2]: data=pd.read_excel('marketing_campaign.xlsx')
data
```

Out[2]:

	ID	Year_Birth	Education	Marital_Status	Income	Kidhome	Teenhome	Dt_Customer	Recency	MntWines	...	NumWebVisitsMonth	AcceptedCm
0	5524	1957	Graduation	Single	58138.0	0	0	2012-09-04	58	635	...	7	
1	2174	1954	Graduation	Single	46344.0	1	1	2014-03-08	38	11	...	5	
2	4141	1965	Graduation	Together	71613.0	0	0	2013-08-21	26	426	...	4	
3	6182	1984	Graduation	Together	26646.0	1	0	2014-02-10	26	11	...	6	
4	5324	1981	PhD	Married	58293.0	1	0	2014-01-19	94	173	...	5	
...	...	...	...	...	...	...	...	...	...	...	...	...	
2235	10870	1967	Graduation	Married	61223.0	0	1	2013-06-13	46	709	...	5	
2236	4001	1946	PhD	Together	64014.0	2	1	2014-06-10	56	406	...	7	
2237	7270	1981	Graduation	Divorced	56981.0	0	0	2014-01-25	91	908	...	6	
2238	8235	1956	Master	Together	69245.0	0	1	2014-01-24	8	428	...	3	
2239	9405	1954	PhD	Married	52869.0	1	1	2012-10-15	40	84	...	7	

2240 rows × 29 columns



# TYPES OF DATA

## CATEGORICAL COLUMNS

Education, Marital  
Status, Dt\_Customer

```
data.dtypes
7]: ID int64
Year_Birth int64
Education object
Marital_Status object
Income float64
Kidhome int64
Teenhome int64
Dt_Customer object
Recency int64
MntWines int64
MntFruits int64
MntMeatProducts int64
MntFishProducts int64
MntSweetProducts int64
MntGoldProds int64
NumDealsPurchases int64
NumWebPurchases int64
NumCatalogPurchases int64
NumStorePurchases int64
NumWebVisitsMonth int64
AcceptedCmp3 int64
AcceptedCmp4 int64
AcceptedCmp5 int64
AcceptedCmp1 int64
AcceptedCmp2 int64
Complain int64
Z_CostContact int64
Z_Revenue int64
Response int64
dtype: object
```

## NUMERICAL COLUMNS

ID, Year\_Birth, Income, Kidhome, Teenhome, Recency, MntWines, MntFruits, MntMeatProducts, MntFishProducts, MntSweetProducts, MntGoldProds, NumDealsPurchases, NumWebPurchases, NumCatalogPurchases, NumStorePurchases, NumWebVisitsMonth, AcceptedCmp3, AcceptedCmp4, AcceptedCmp5, AcceptedCmp1, AcceptedCmp2, Complain, Response,



# DATA CLEANING AND PREPROCESSING

# NULL VALUES

From the fig 1, we understand that the INCOME column has 24 null values.

We can impute them using “MEAN” method which is shown in Fig 2

```
In [9]: data['Income'].fillna(data['Income'].mean(),inplace=True)

In [10]: data.isnull().sum()

Out[10]: ID 0
Year_Birth 0
Education 0
Marital_Status 0
Income 0
Kidhome 0
Teenhome 0
Dt_Customer 0
Recency 0
MntWines 0
MntFruits 0
MntMeatProducts 0
MntFishProducts 0
MntSweetProducts 0
```

Fig 2

```
In [8]: print("Missing data in the dataframe:")
print(data.isnull().sum())

Missing data in the dataframe:
ID 0
Year_Birth 0
Education 0
Marital_Status 0
Income 24
Kidhome 0
Teenhome 0
Dt_Customer 0
Recency 0
MntWines 0
MntFruits 0
MntMeatProducts 0
MntFishProducts 0
MntSweetProducts 0
MntGoldProds 0
NumDealsPurchases 0
NumWebPurchases 0
NumCatalogPurchases 0
NumStorePurchases 0
NumWebVisitsMonth 0
AcceptedCmp3 0
AcceptedCmp4 0
AcceptedCmp5 0
```

Fig 1

# DUPLICATE VALUES

From the fig 3, we understand that there are no duplicate rows.

```
In [11]: data.duplicated().sum()  
Out[11]: 0
```

Fig 3

# OUTLIER VALUES

Year\_Birth: From fig 4, most values between 1960 and 1980, looks reasonable. Some of them are around 1900 or even less, so we remove them.

We can see the box plot after removing years < 1960 from Fig 5.

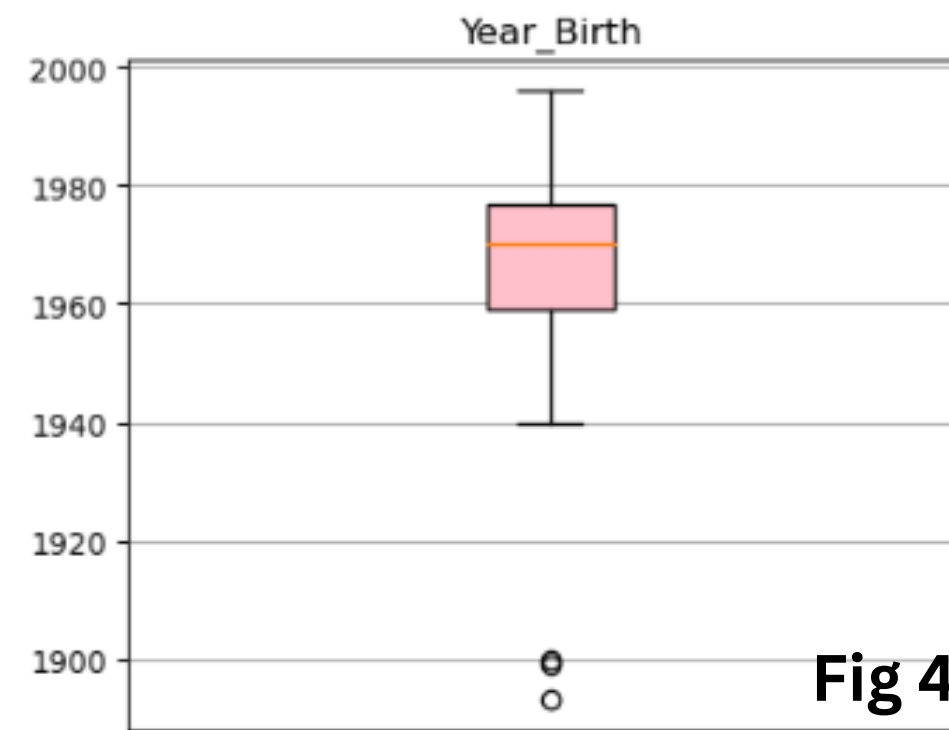


Fig 4

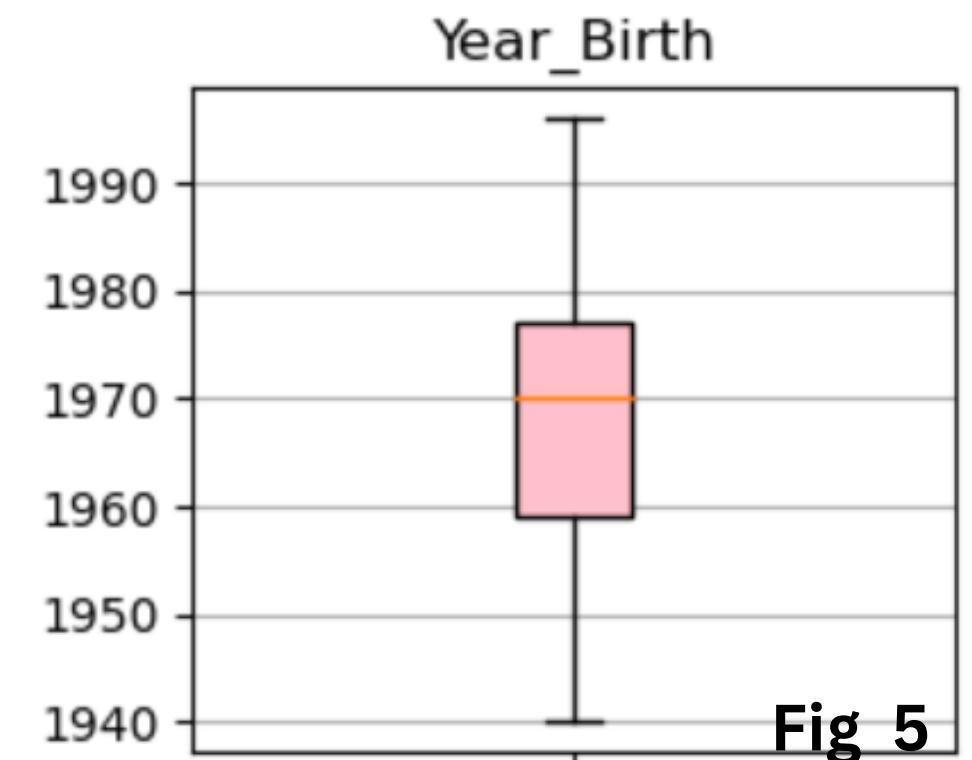


Fig 5

```
#filter Year_Birth < 1920 and drop as it is less than 1% of the data  
data.drop(data[data['Year_Birth'] < 1920].index,inplace=True)
```

# OUTLIER VALUES

Income: From fig 6, data seems skewed by one big value. All the others are between 0 and 100k.

From the fig 7, we have removed the outliers using IQR method.

```
▶ Q1 = data['Income'].quantile(0.25)
  Q3 = data['Income'].quantile(0.75)
  IQR = Q3 - Q1

  # Identify the outliers in the Income column
  outliers = data[(data['Income'] < (Q1 - 1.5 * IQR)) | (data['Income'] > (Q3 + 1.5 * IQR))]

▶ data = data[~((data['Income'] < (Q1 - 1.5 * IQR)) | (data['Income'] > (Q3 + 1.5 * IQR)))]
```

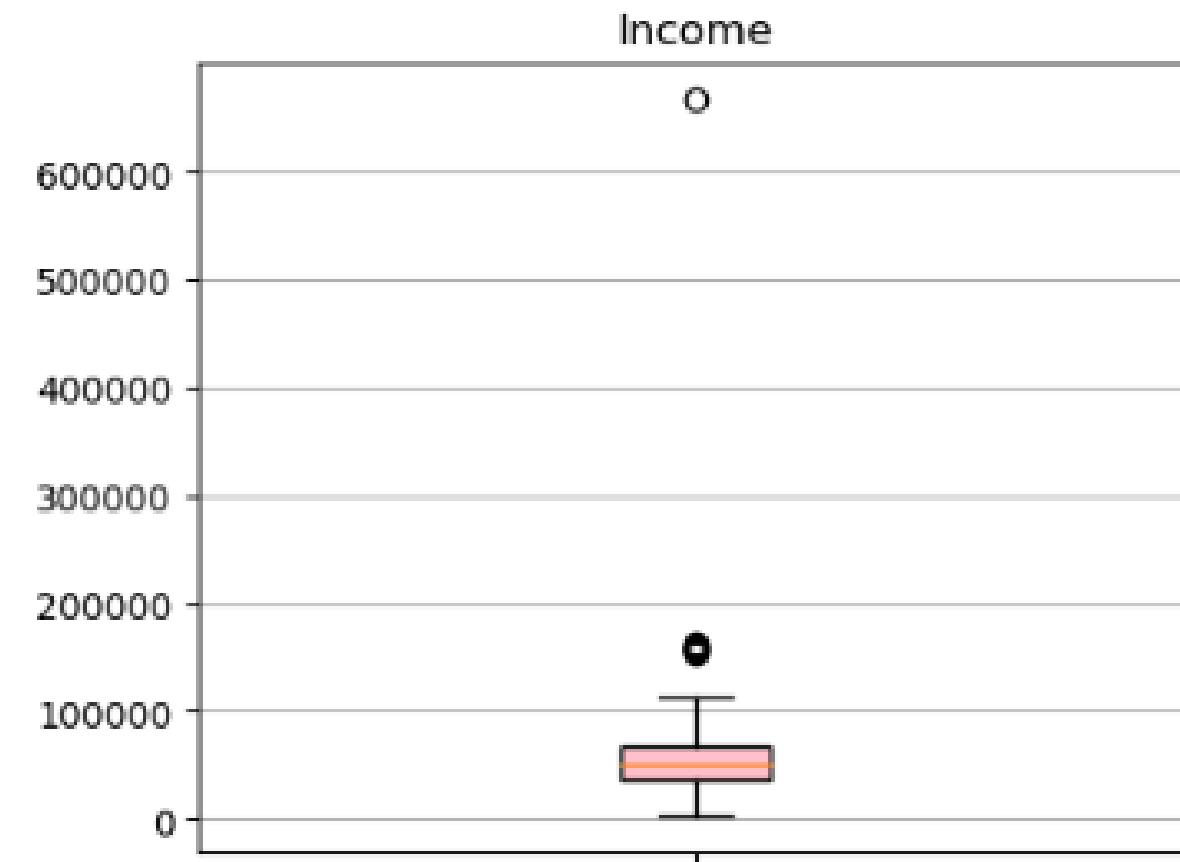


Fig 6

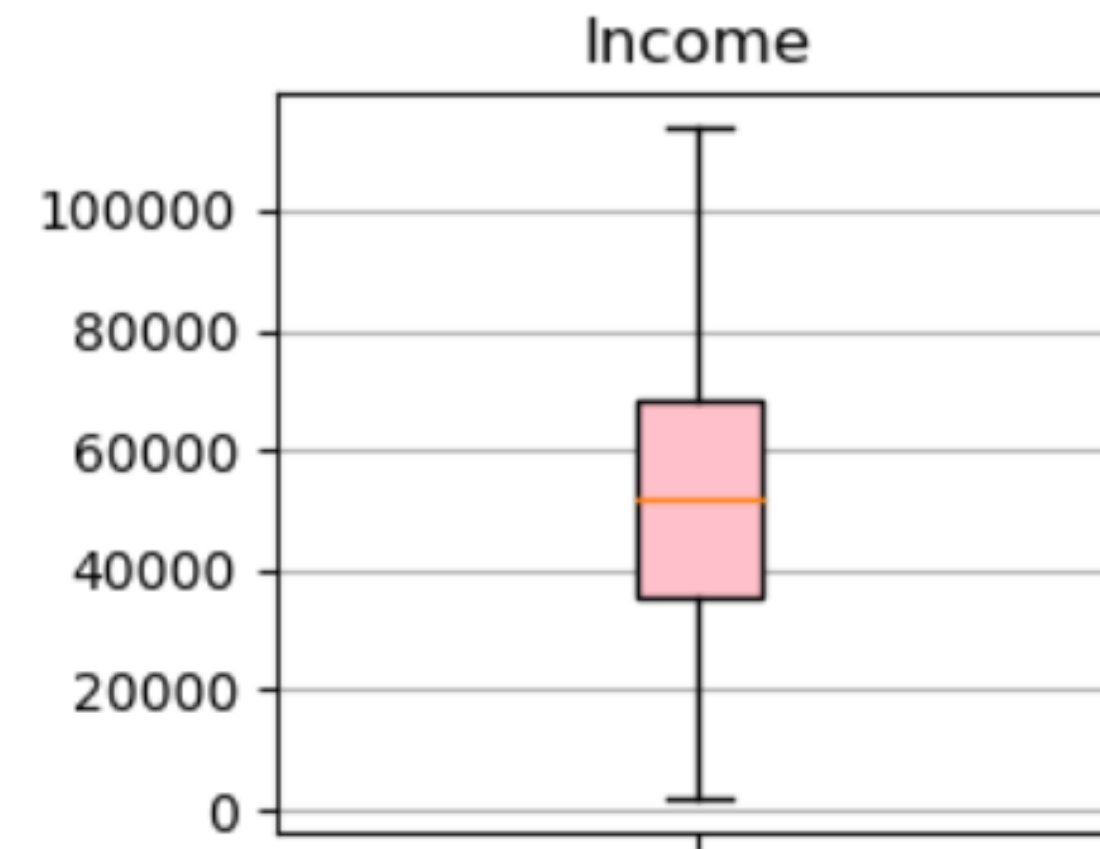


Fig 7

# FEATURE ENGINEERING



**Education level**

Derive education level as “low”, ”medium”, ”high” from education column.

**Age**

Derive age feature from year\_birth.

**Total\_Campaigns\_Accepted**

How many campaigns it took to accept the offer for customer

We create new features from existing attributes for easy analysis. Fig.8

Education_level	Age	Total_Campaigns_Accepted
High	67	0
High	70	0
High	59	0
High	40	0
Low	43	0
...	...	...
High	57	0

Fig 8



# DATA TRANSFORMATION

```
F.head()
```

log_MntMeatProducts	log_MntFishProducts	log_MntSweetProducts
0.797751	1.016818	0.800577
-0.869936	-0.576044	-0.525497
0.242019	0.846012	0.312283
-0.449579	-0.065626	-0.283324
0.214123	0.504884	0.396541

- **Log transformation** was applied to spending-related columns (MntWines, MntFruits, MntMeatProducts, etc.) to reduce skewness and stabilize variance.
- These columns often contain highly skewed data, with a few customers spending disproportionately more.
- Log transformation compresses large values and spreads out smaller values, making the data more normally distributed.



# REMOVE IRRELEVANT FEATURES

These features has the same value for all entries, it doesn't provide any variability or distinguishing information for analysis or modeling. Such features are generally considered irrelevant because they don't contribute to segmenting or predicting outcomes. They simply don't help the model understand or differentiate between customers. So we drop these columns.

```
In [70]: data=data.drop(['Z_CostContact', 'Z_Revenue'],axis=1)
```

From the figure, we can see that **Z\_CostContact** and **Z\_Revenue** are included in datasets as constants, meaning they hold a single, unchanging value across all rows.

Z_CostContact	Z_Revenue
3	11
3	11
3	11
3	11
3	11
...	...
3	11
3	11
3	11
3	11
3	11

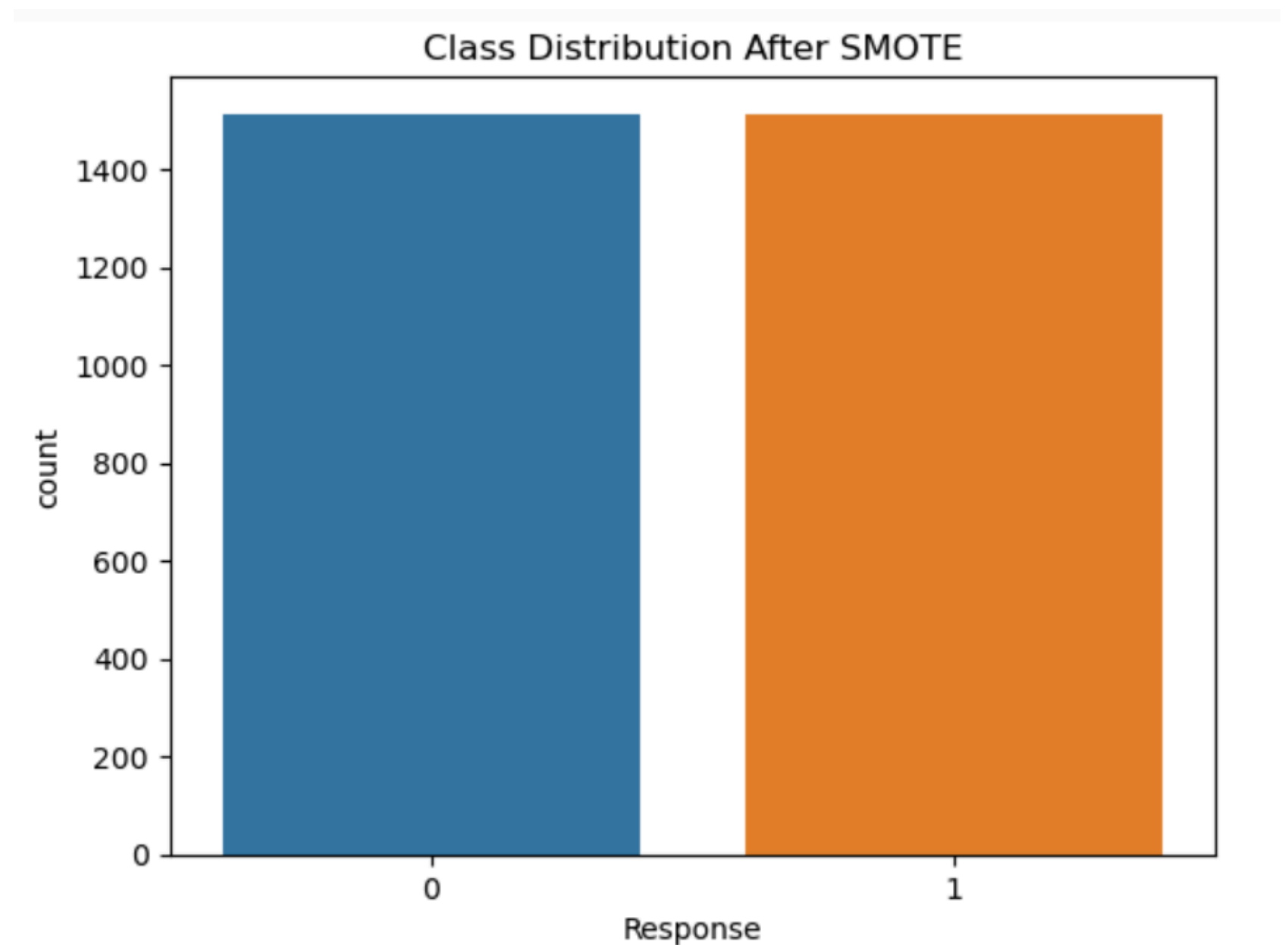
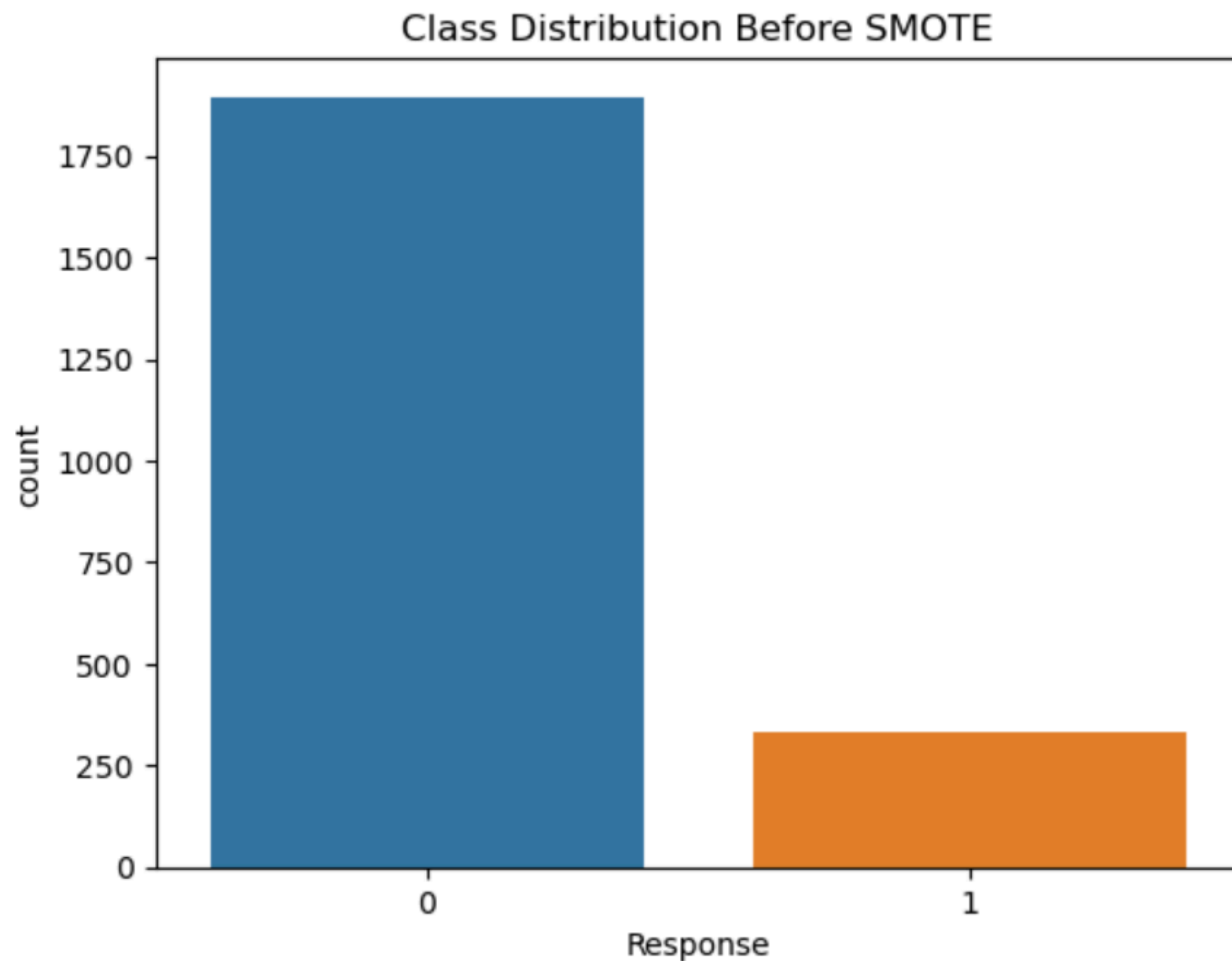
Fig 10

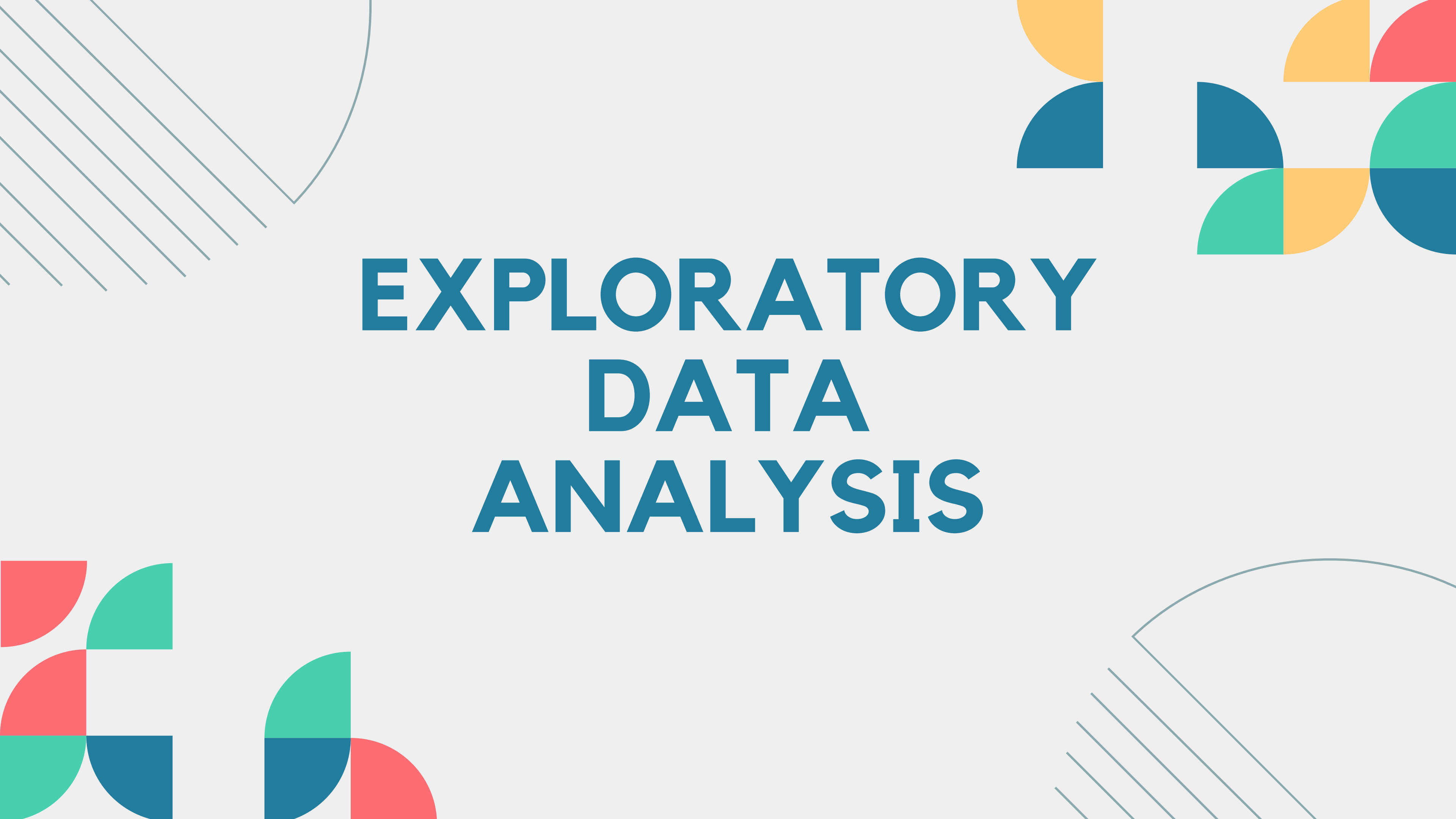
# SMOTE TECHNIQUE

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Apply SMOTE to balance the training data
smote = SMOTE(random_state=42)
X_train_smote, y_train_smote = smote.fit_resample(X_train, y_train)
```

To handle class imbalance we use the technique - SMOTE (Synthetic Minority Over-sampling Technique)



The background features four decorative geometric patterns in the corners. The top-left and bottom-right corners contain thin, parallel diagonal lines. The top-right and bottom-left corners contain clusters of semi-circles in teal, orange, and red. The text is centered in a bold, blue, sans-serif font.

# EXPLORATORY DATA ANALYSIS

# SUMMARY STATISTICS

Summarizes and provides  
the gist of information  
about the sample data.

data.describe()

	ID	Year_Birth	Income	Kidhome	Teenhome	Recency	MntWines	MntFruits	MntMeatProducts	MntFishProducts
count	2240.000000	2240.000000	2216.000000	2240.000000	2240.000000	2240.000000	2240.000000	2240.000000	2240.000000	2240.000000
mean	5592.159821	1968.805804	52247.251354	0.444196	0.506250	49.109375	303.935714	26.302232	166.950000	37.525446
std	3246.662198	11.984069	25173.076661	0.538398	0.544538	28.962453	336.597393	39.773434	225.715373	54.628979
min	0.000000	1893.000000	1730.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	2828.250000	1959.000000	35303.000000	0.000000	0.000000	24.000000	23.750000	1.000000	16.000000	3.000000
50%	5458.500000	1970.000000	51381.500000	0.000000	0.000000	49.000000	173.500000	8.000000	67.000000	12.000000
75%	8427.750000	1977.000000	68522.000000	1.000000	1.000000	74.000000	504.250000	33.000000	232.000000	50.000000
max	11191.000000	1996.000000	666666.000000	2.000000	2.000000	99.000000	1493.000000	199.000000	1725.000000	259.000000

# AGE DISTRIBUTION

## MOST COMMON AGE GROUP

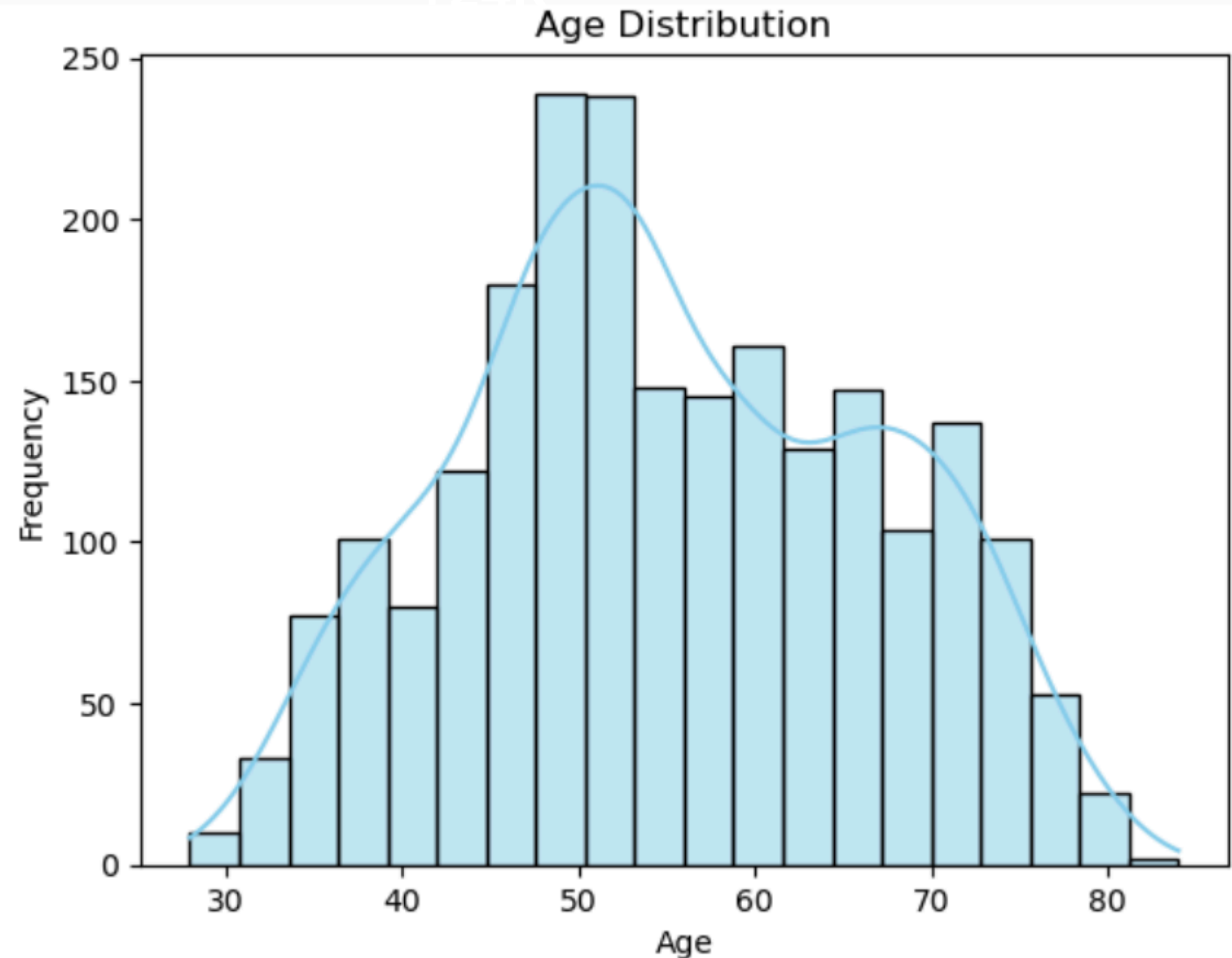
The peak of the distribution appears around ages 50–55, indicating that this age range has the highest frequency within the dataset.

## SKWENESS

Slightly skewed to the right. This indicates that younger ages are less common in this dataset

## RANGE

The dataset includes ages from around 30 to just above 80, suggesting a focus on a mature to older adult population



# CHANNEL PERFORMANCE

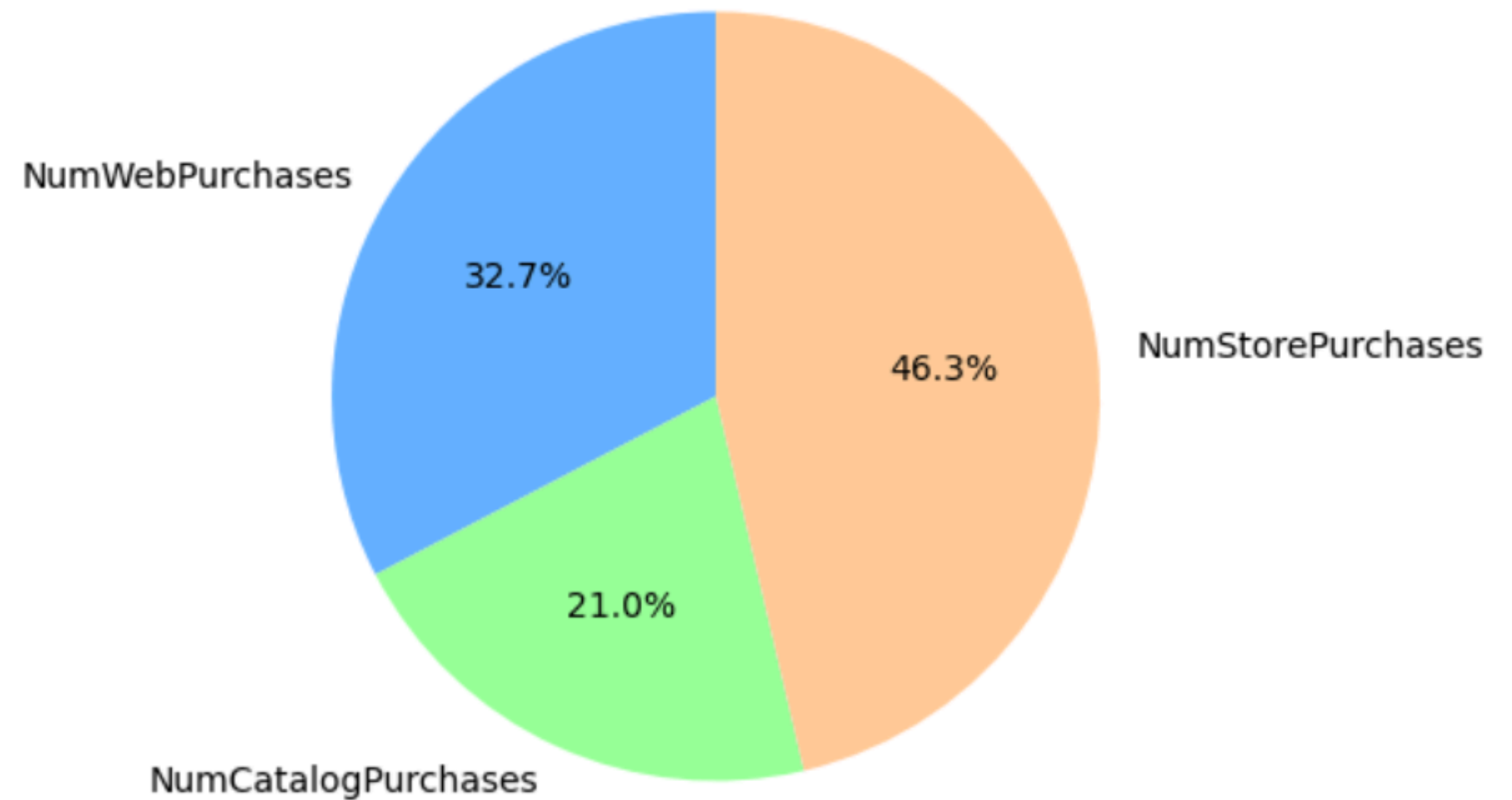
## CATALOG

We find that the Catalog sales channel is the least-performing channel, accounting for only 21% of the total purchases. This channel requires attention to enhance Improvement.

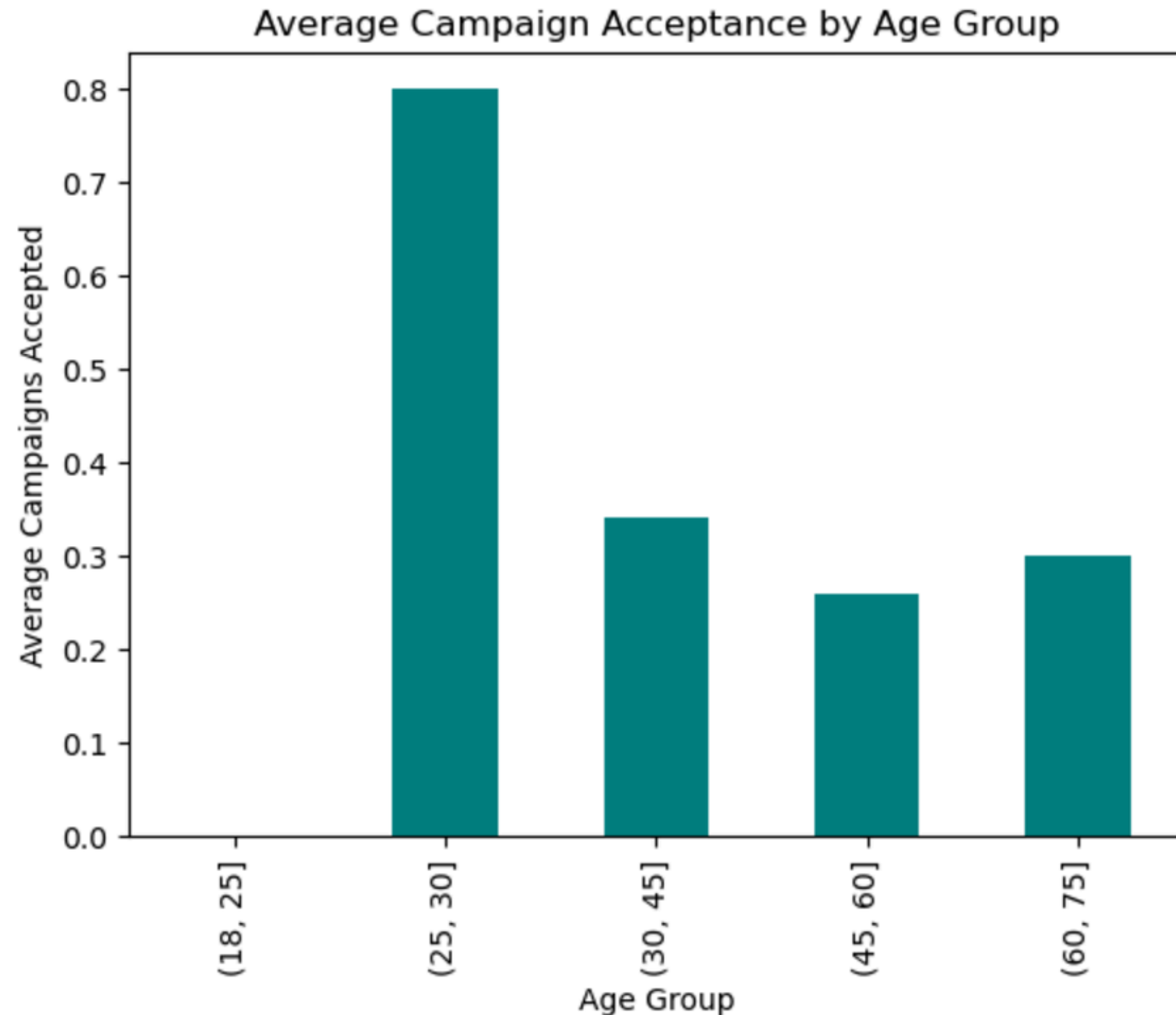
## STORE

We find that the Store Purchases are more. Focus should be to increase them furthur.

Campaign Channel Performance



# AVERAGE CAMPAIGN ACCEPTANCE



## HIGH ACCEPTANCE

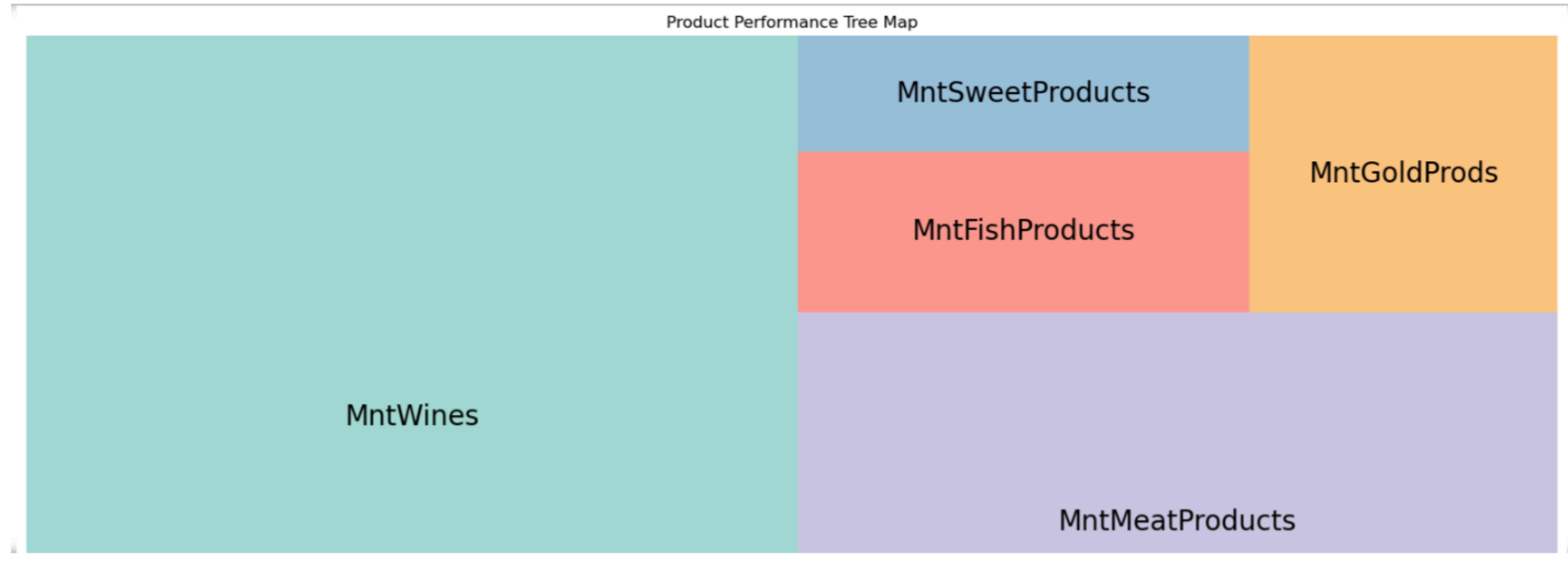
The age group 25–30 has the highest average acceptance rate for campaigns, with a rate close to 0.8.

## LOW ACCEPTANCE

The age groups 18–25 have very poor campaign acceptance rates.

## INCREASED ACCEPTANCE

The age group 30–45 shows a higher acceptance rate nearing 0.4. This could suggest that middle aged adults are somewhat more receptive to campaigns than younger individuals, though still less than people between 25–30



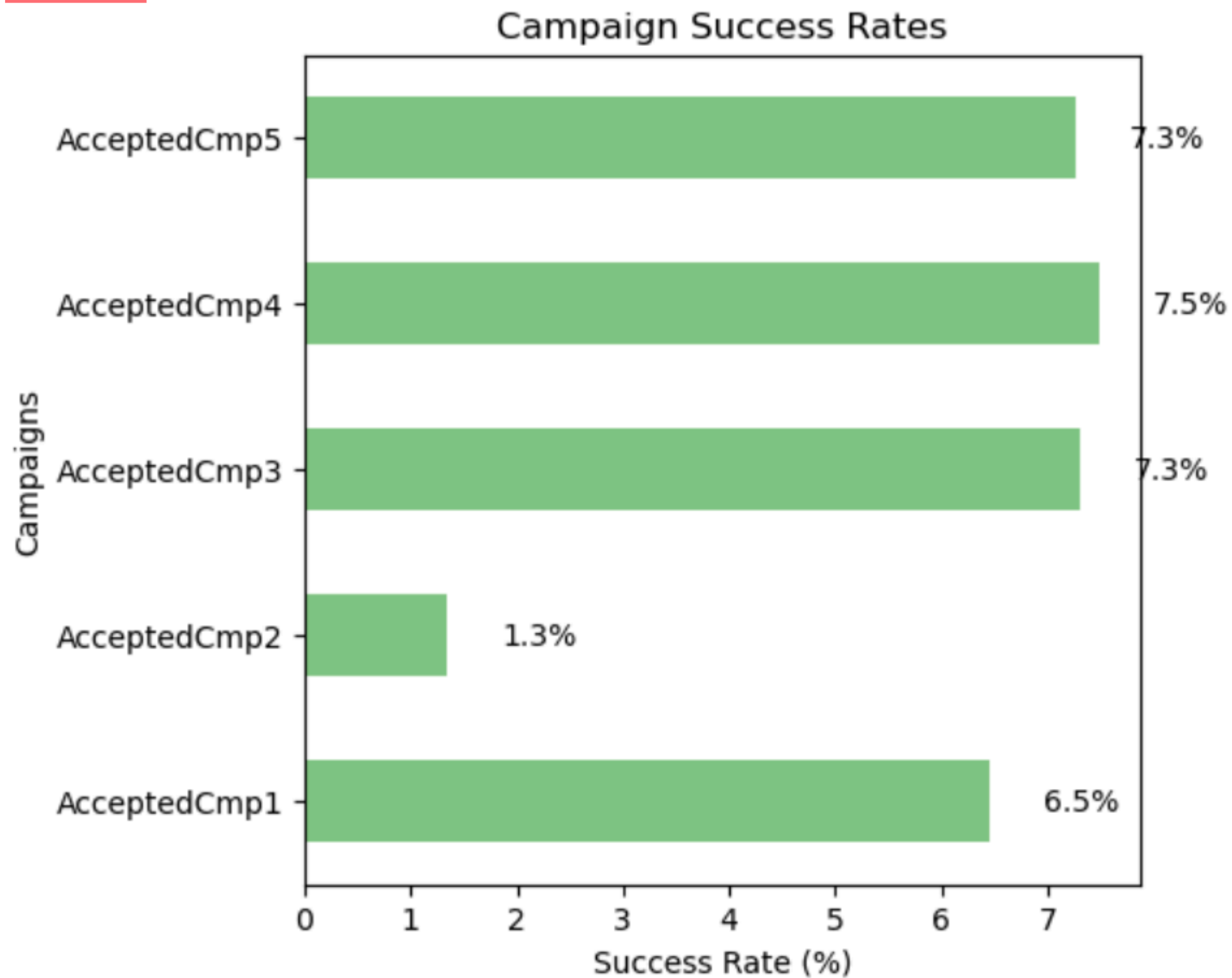
## PRODUCT PERFORMANCE - TREEMAP

We find that Wine has been the product with highest purchases. While sweet products have been the lowest performers.

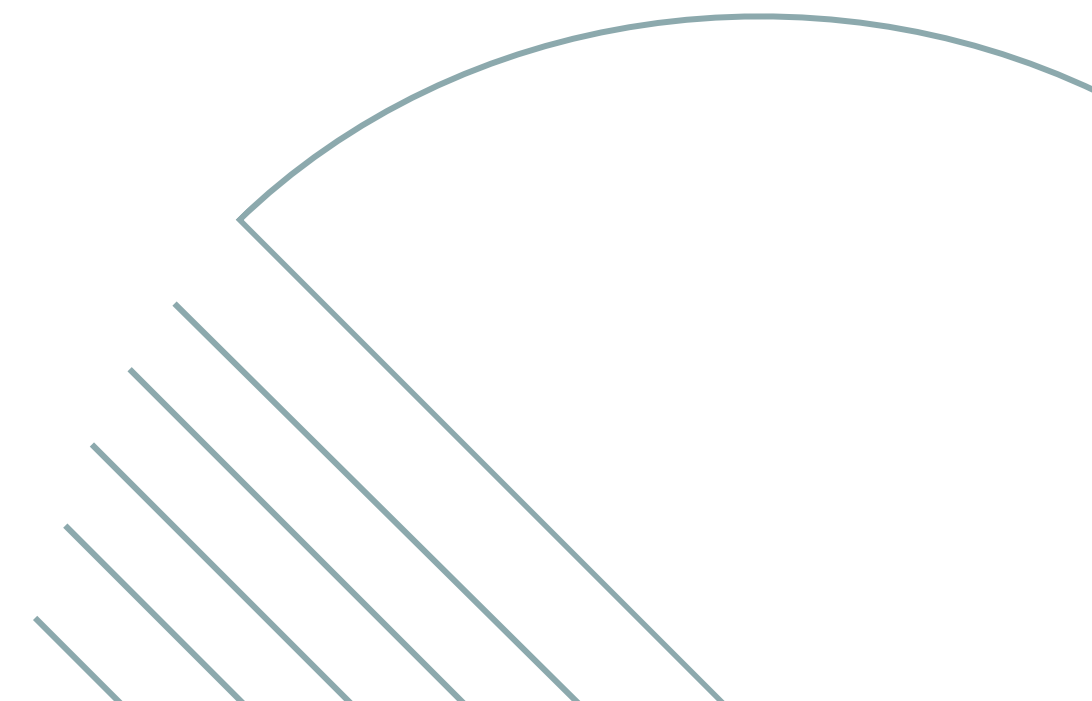




# CAMPAIGN SUCCESS

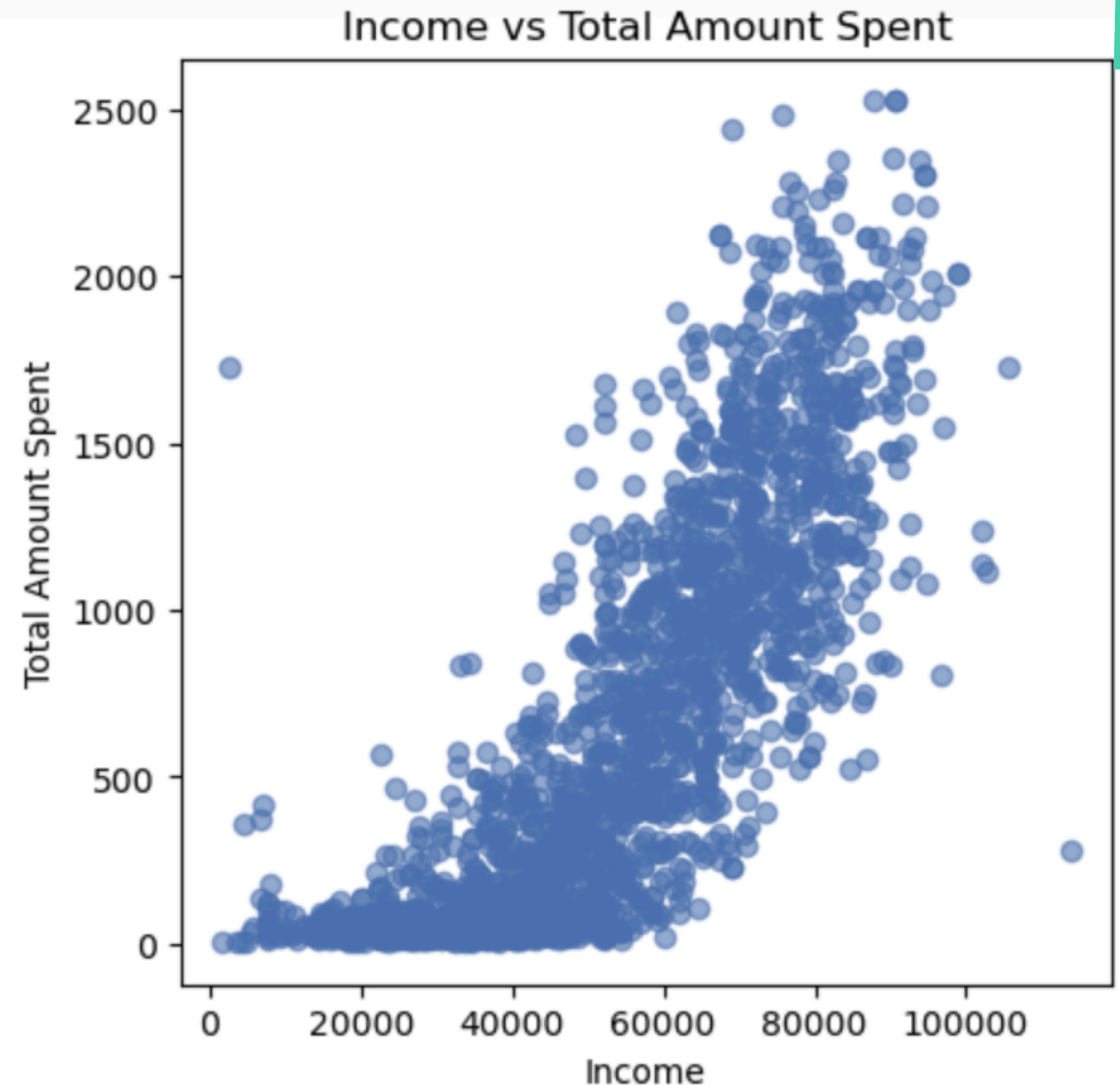


We observe that **Campaign 4** has highest response. While campaign **channel 2** has lowest response.



# RELATION BETWEEN INCOME AND SPENDING

This scatter plot shows a **positive correlation** between income and total amount spent. Higher-income individuals generally spend more, though spending varies widely within each income level. This suggests income influences spending behavior, with more **affluent individuals potentially having greater discretionary spending capacity**.



# CUSTOMER SEGMENTATION

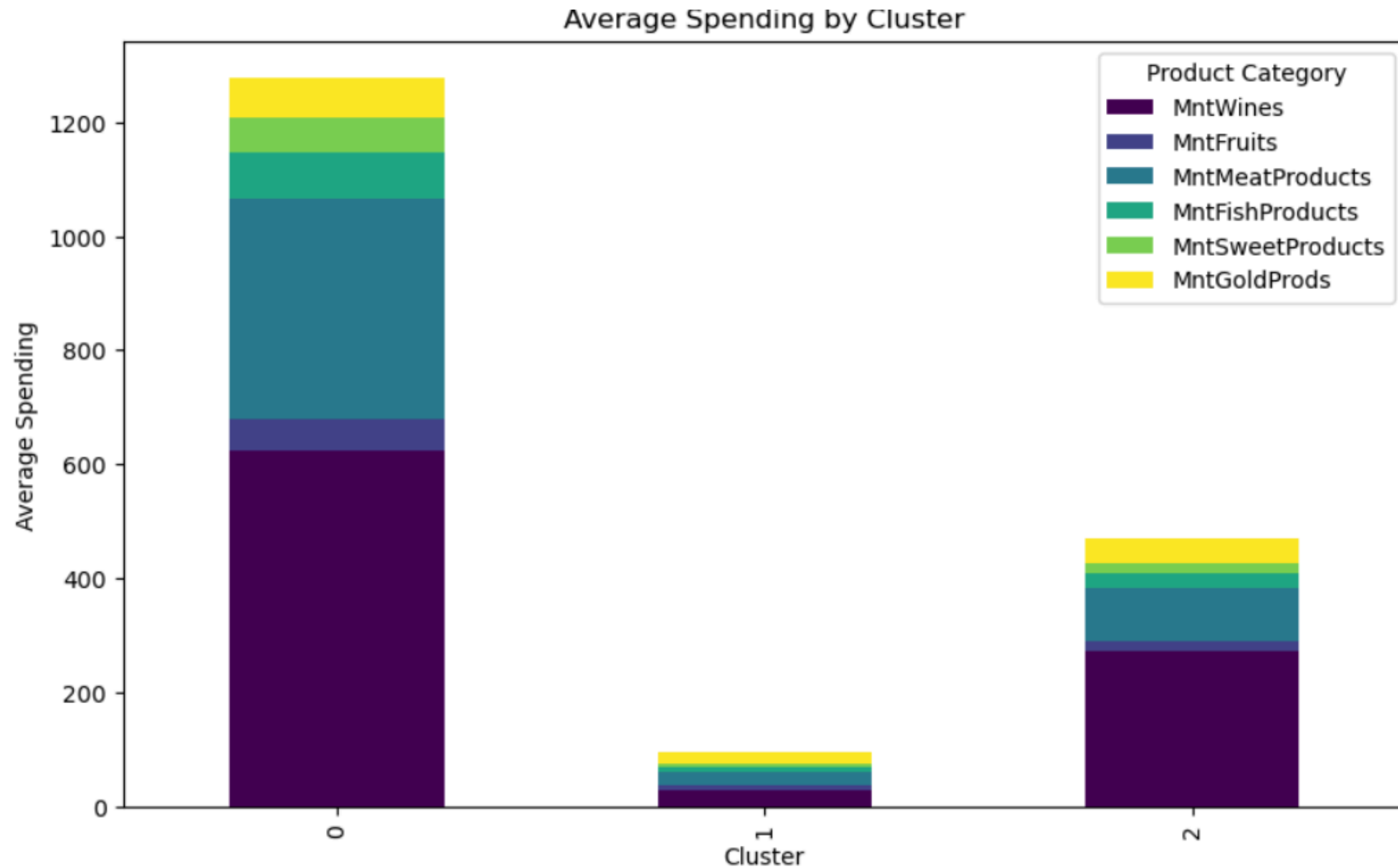


**Cluster 0 (purple):** Low income and low spending customers, likely representing budget-conscious or low-earning segments.

**Cluster 1 (green):** Moderate income and spending customers, possibly middle-income customers with average spending habits.

**Cluster 2 (yellow):** High income and high spending customers, possibly affluent customers who are the highest spenders.

# CLUSTERS VS AVG SPENDING

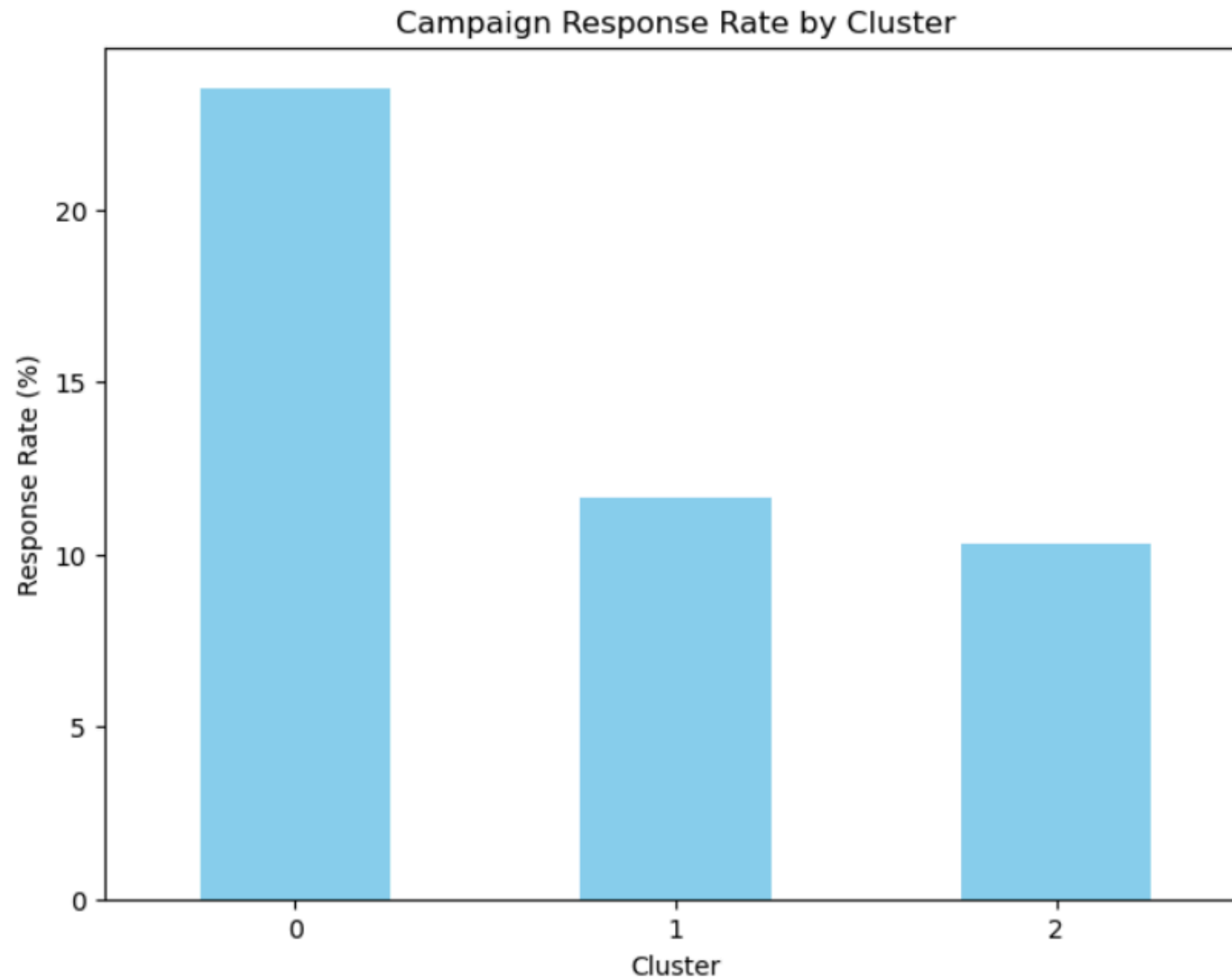


**Cluster 0 :** Low income and low spending customers, likely spend more on Wine.

**Cluster 1 :** Moderate income and spending customers, tend to spend equally on all the products.

**Cluster 2 :** High income and high spending customers, spend more on Wine.

# CLUSTERS VS RESPONSE

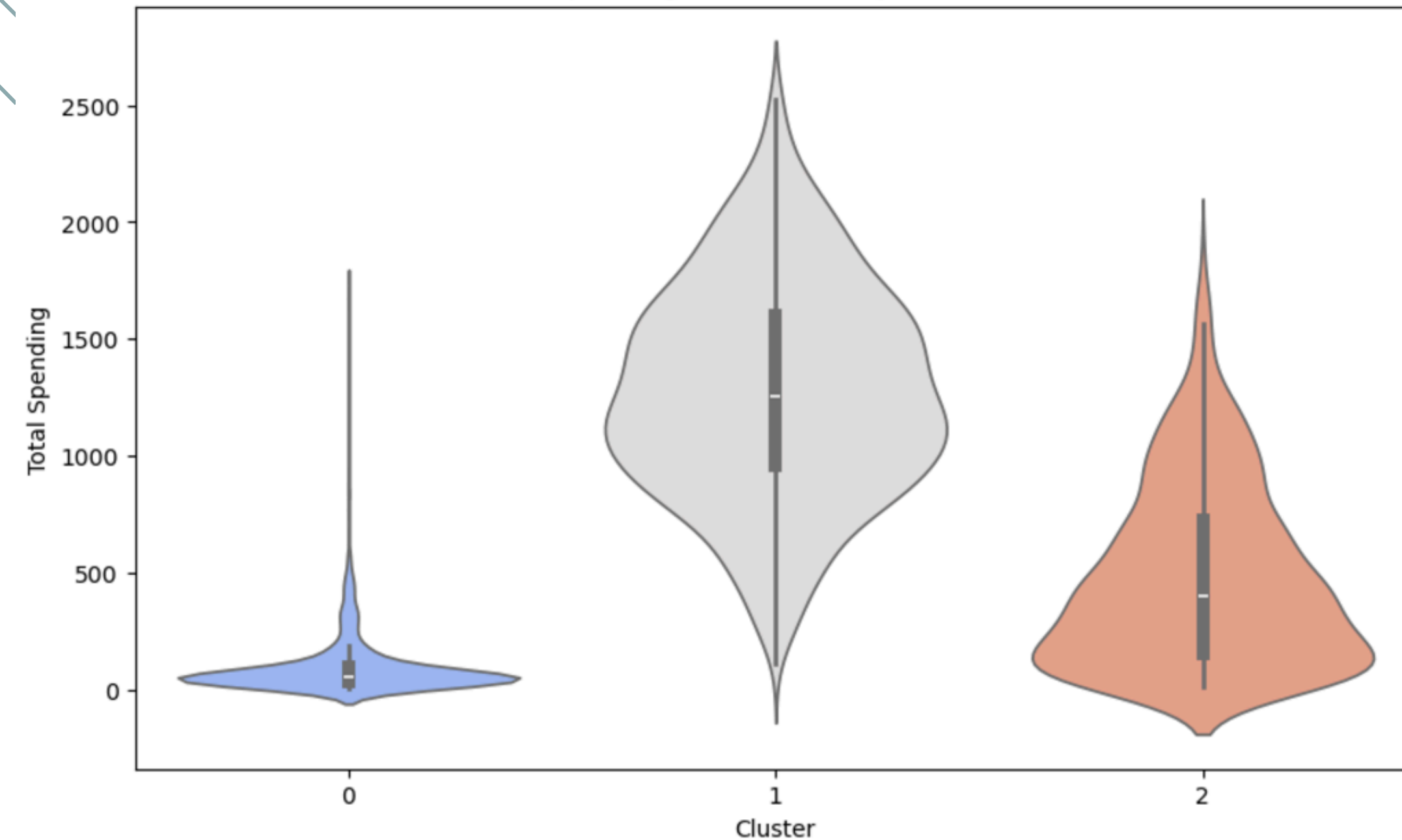


**Cluster 0** – Low income and low spending customers, respond more to campaigns. While **Cluster 2**, High income high spending do not respond.



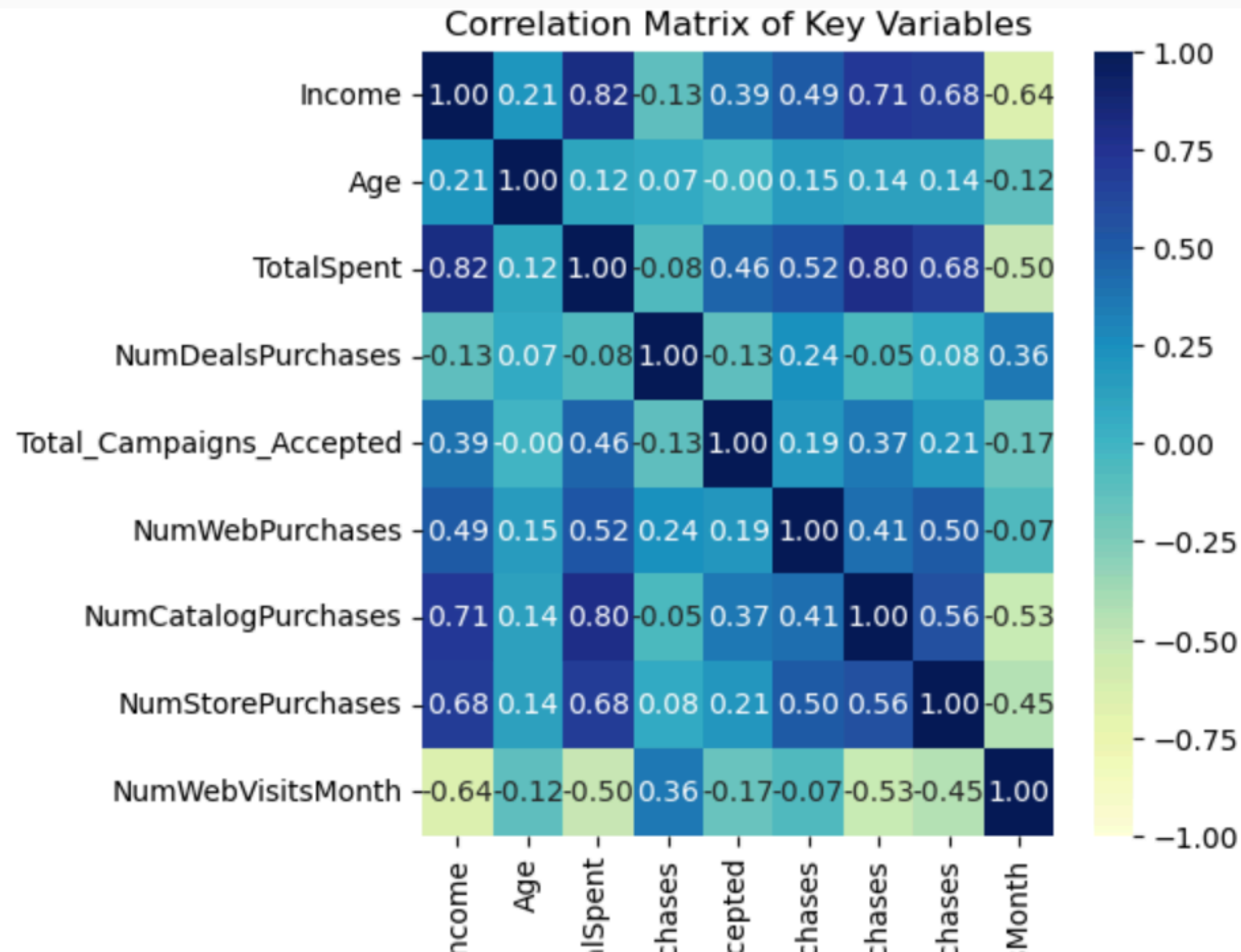
# INCOME TRENDS BY CAMPAIGN ACCEPTANCE

Spending Distribution by Cluster



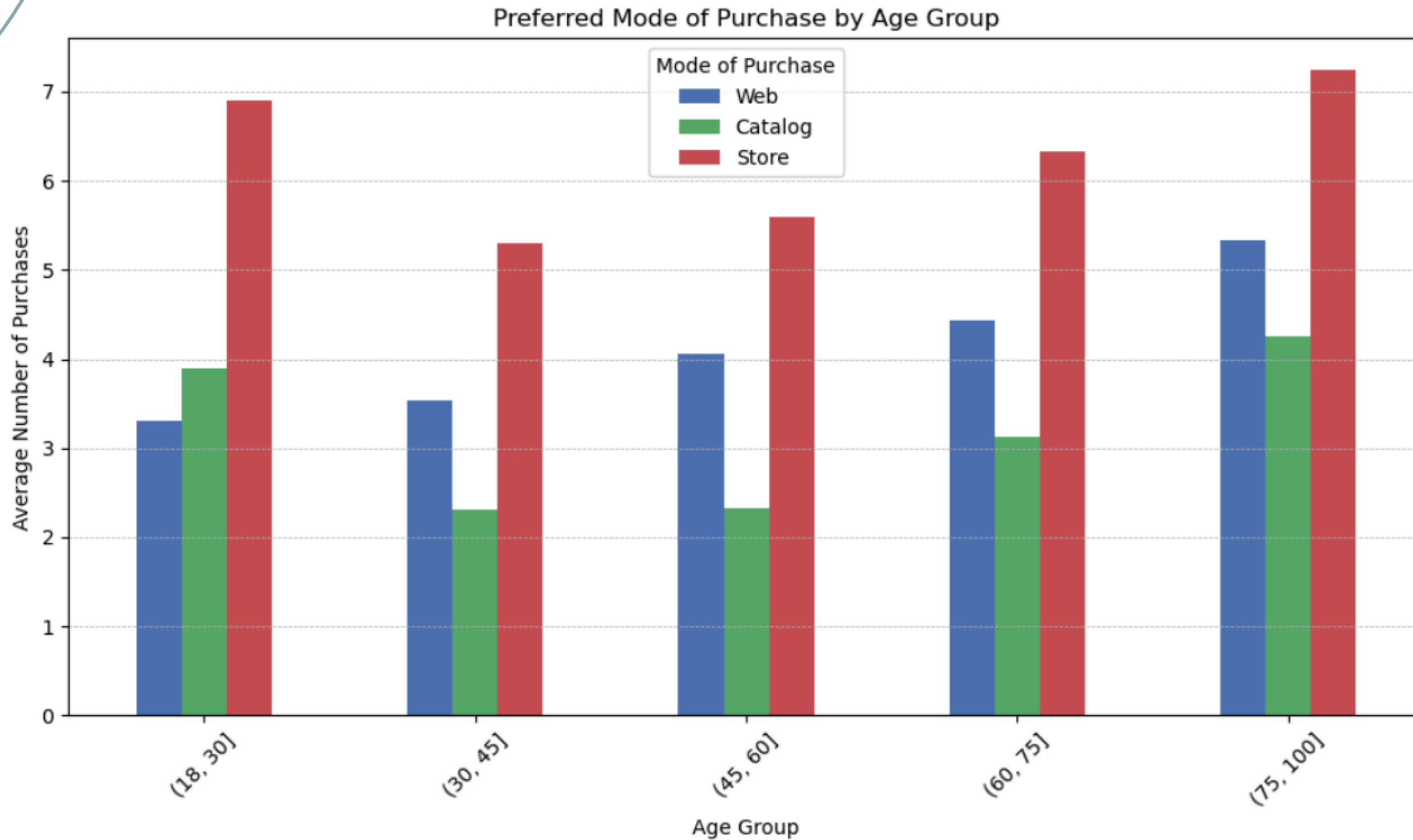
- Cluster 0: Likely low-value customers; campaigns should focus on basic offers to incentivize spending.
- Cluster 1: Balanced spending; campaigns can aim to increase loyalty or upsell.
- Cluster 2: High-value customers; premium or exclusive campaigns may resonate effectively.

# CORRELATION MATRIX



- This correlation matrix shows that **TotalSpent** has strong positive correlations with **NumCatalogPurchases** (0.80) and **Income** (0.82).
- Negative correlations between **NumWebVisitsMonth** and **Income** suggest that lower-income individuals tend to have higher monthly web visits.

# AGE GROUP VS PREFERRED MODE OF PURCHASE



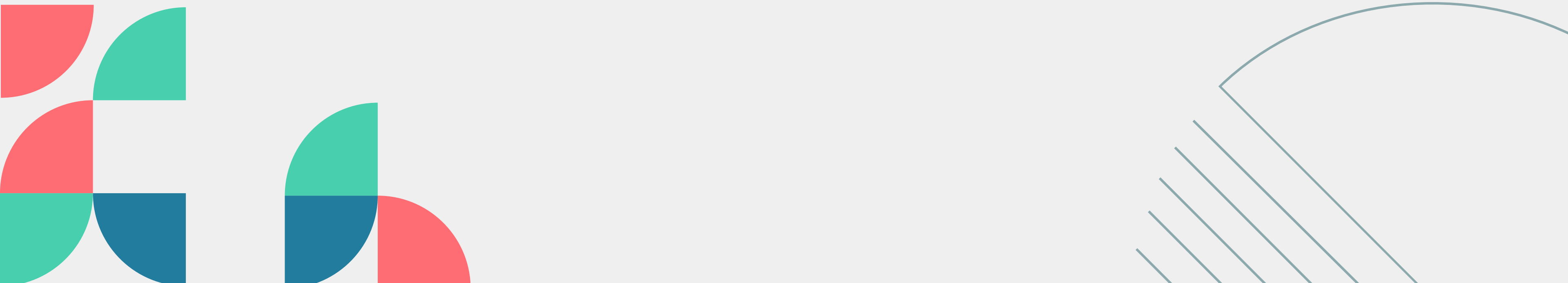
Mostly all the age groups prefer Store Purchases.



The top-left corner features a series of thin, parallel diagonal lines in a light teal color. The top-right corner contains several overlapping semi-circles in teal, orange, and red. 

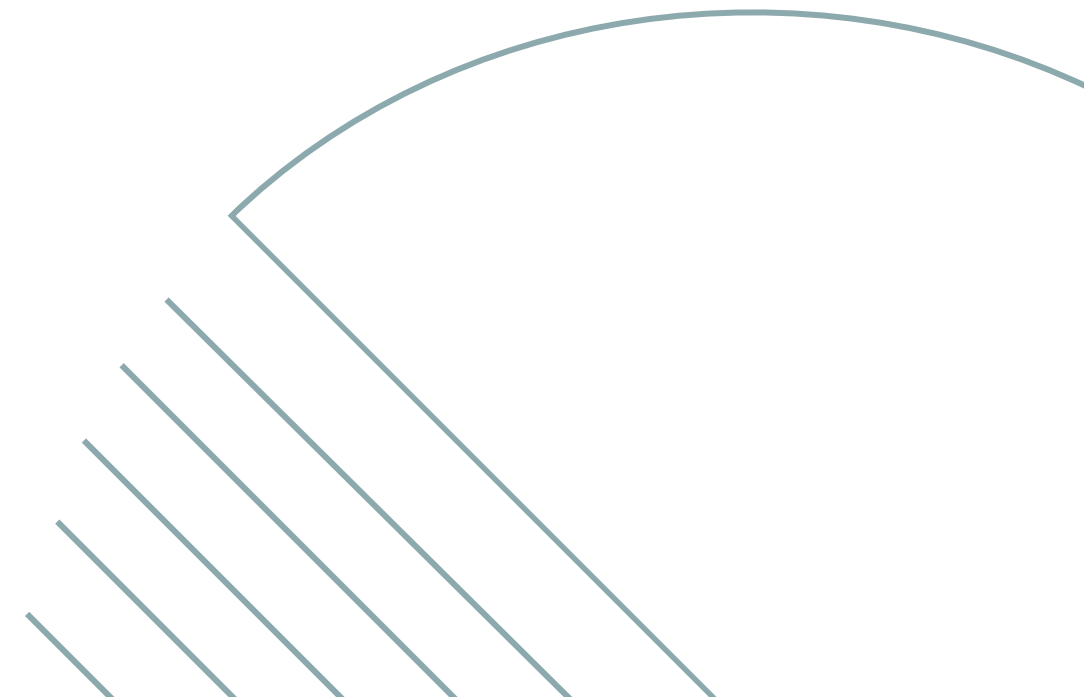
# SUMMARY


Thus from the EDA we get to know which campaign, purchase channel, product performs well and which needs optimization and improvement. We also segmented customers based on their income and spending levels for better analysis.

The bottom-left corner features several overlapping semi-circles in red, teal, and dark teal. The bottom-right corner contains a series of thin, parallel diagonal lines in a light teal color, similar to the top-left corner.

# SUMMARY OF EDA

- Spending habits are highly dependent on their income level.
- Catalog channels are least performing and store sales channel is the best performing.
- Most of the customers purchase in stores.
- People between the age 25 and 30 are highly respondent towards marketing campaigns.
- Campaign 4 is the most effective campaign and campaign 2 is the least effective.
- Wine, Meat and Gold are significantly most purchased items



The background features four decorative geometric patterns in the corners. The top-left and bottom-right corners contain thin, parallel diagonal lines. The top-right and bottom-left corners contain clusters of semi-circles in teal, orange, and red. The text is centered in a bold, teal, sans-serif font.

# MODEL BUILDING AND EVALUATION

# MODEL BUILDING FOR PREDICTING CAMPAIGN ACCEPTANCE

**Objective:** Build and evaluate multiple machine learning models to predict the likelihood of a customer accepting a marketing campaign.

## Models Considered:

- Logistic Regression
- Decision Tree Classifier
- Random Forest Classifier
- Gradient Boosting Classifier



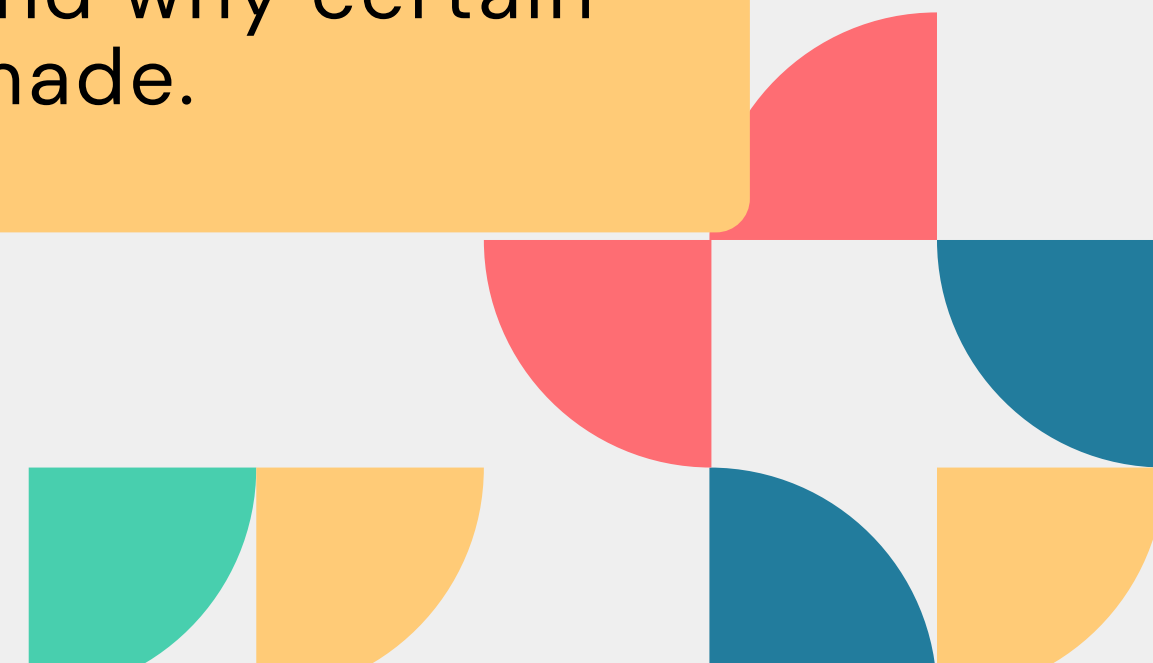
# MODEL SELECTION RATIONALE

**Logistic Regression:** This is a simple yet powerful classification model well-suited for binary outcomes like campaign acceptance.

**Random Forest:** Uses an ensemble of decision trees to improve accuracy and reduce overfitting

**Gradient Boosting Machines (GBM):** This model, especially with implementations like XGBoost, LightGBM, or CatBoost, excels at learning complex patterns in the data.

**Decision Trees:** Work by splitting the data into branches based on feature values, which makes it easy to understand why certain predictions are made.



# MODELL BUILDING WITH THE ORIGINAL FEATURES

```
X=df_conv()  
y=Response  
X=X.drop(columns=['Response', 'ID', 'Dt_Customer', 'Recency_bins', 'NumDeals_bins', 'Cluster', 'Month_Joined', 'Education_level'])
```

- Here first remove the binned and raw features from the dataset that are no longer useful.
- And then store the Response feature in 'y'.

```
numerical_cols = X.select_dtypes(include=['float64', 'int64']).columns  
categorical_cols = X.select_dtypes(include=['object', 'category']).columns  
  
preprocessor = ColumnTransformer(  
    transformers=[  
        ('num', StandardScaler(), numerical_cols),  
        ('cat', OneHotEncoder(drop='first'), categorical_cols)  
    ]  
)
```

- We encode the numerical and categorical columns using StandardScaler and OneHot Encoder.
- And then dataset is splitted into Testing Set and Training Set.

# 1.LOGISTIC REGRESSION

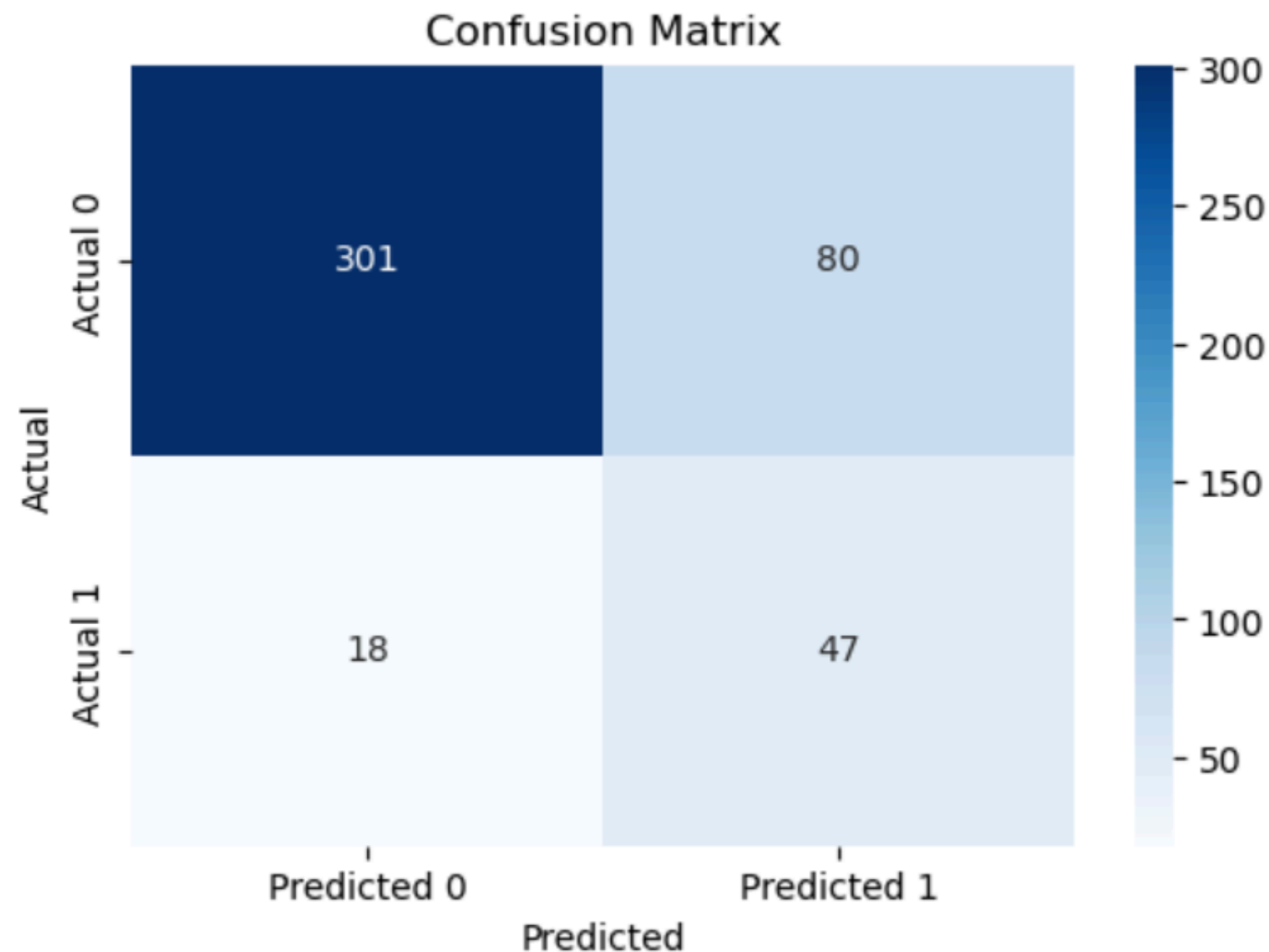
Evaluation metrics of Logistic Regression Model:

**Accuracy: 78.03%**

**Precision: 37.01%**

**Recall: 72.31%**

**F1-Score: 48.96%**



**True Negatives (TN):** 47 cases where the model correctly predicted "0" (customer did not accept the campaign).

**False Positives (FP):** 18 cases where the model predicted "1" (accepted the campaign) but the actual was "0".

**False Negatives (FN):** 80 cases where the model predicted "0" (did not accept) but the actual was "1".

**True Positives (TP):** 301 cases where the model correctly predicted "1" (customer accepted the campaign).

## 2. DECISION TREE

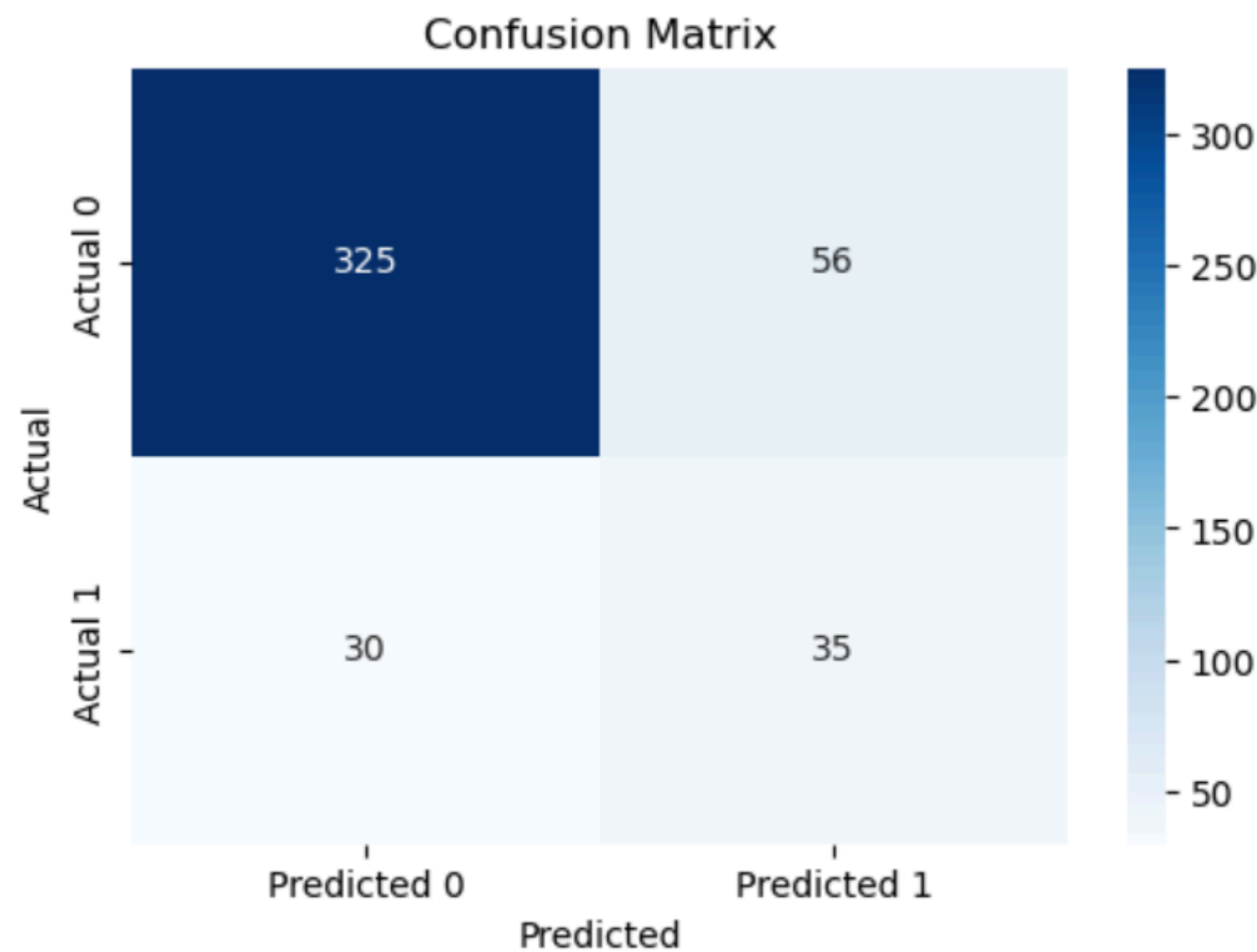
Evaluation metrics of Decision Trees Model:

Accuracy: 80.72%

Precision: 38.46%

Recall: 53.85%

F1-Score: 44.87%



**True Negatives (TN):** 35 cases where the model correctly predicted "0" (customer did not accept the campaign).

**False Positives (FP):** 30 cases where the model predicted "1" (accepted the campaign) but the actual was "0".

**False Negatives (FN):** 56 cases where the model predicted "0" (did not accept) but the actual was "1".

**True Positives (TP):** 325 cases where the model correctly predicted "1" (customer accepted the campaign).



### 3. RANDOM FOREST

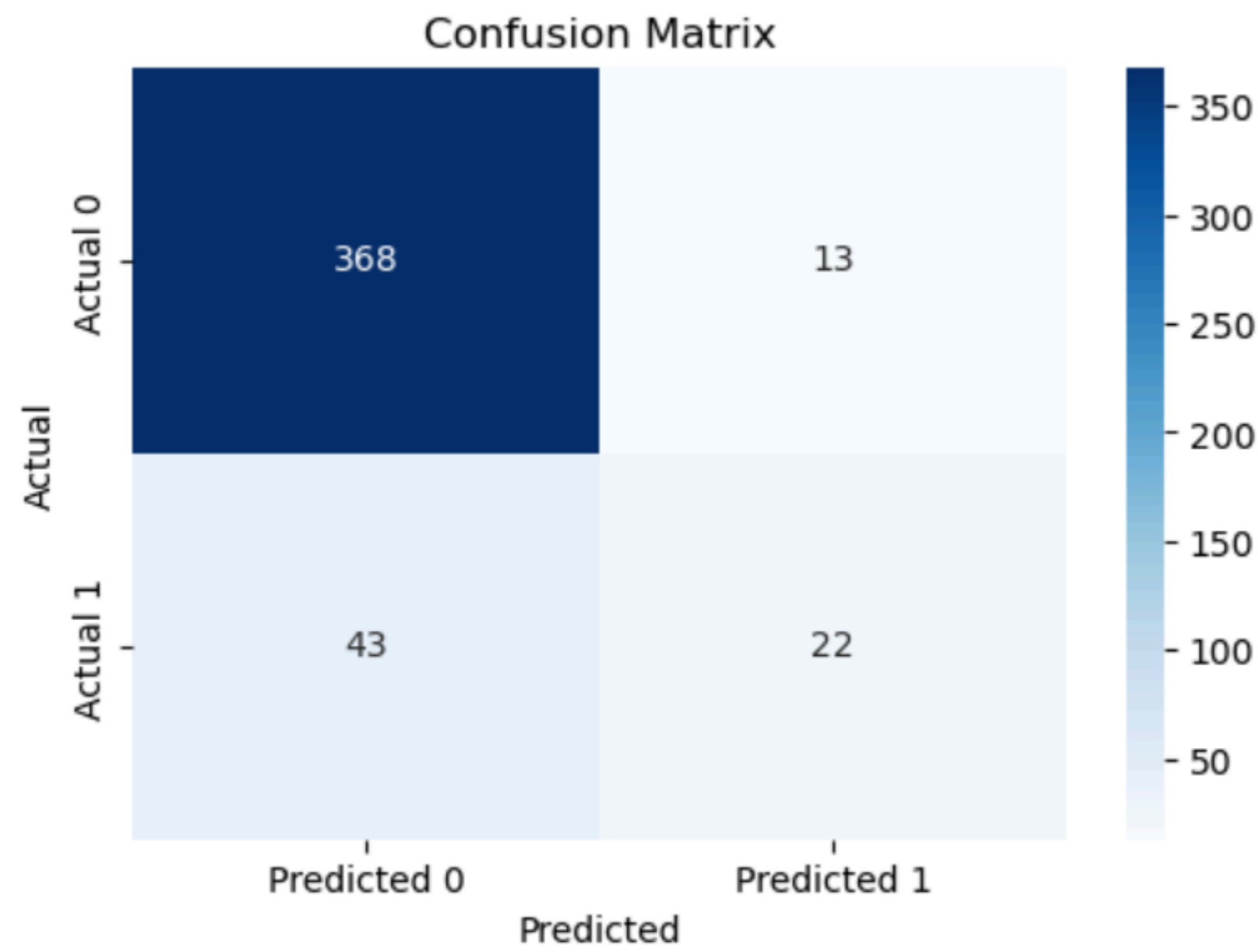
Evaluation metrics of Random Forest Model:

Accuracy: 87.44%

Precision: 62.86%

Recall: 33.85%

F1-Score: 44.00%



**True Negatives (TN):** 22 cases where the model correctly predicted "0" (customer did not accept the campaign).

**False Positives (FP):** 43 cases where the model predicted "1" (accepted the campaign) but the actual was "0".

**False Negatives (FN):** 13 cases where the model predicted "0" (did not accept) but the actual was "1".

**True Positives (TP):** 368 cases where the model correctly predicted "1" (customer accepted the campaign).

## 4. GRADIENT BOOSTER

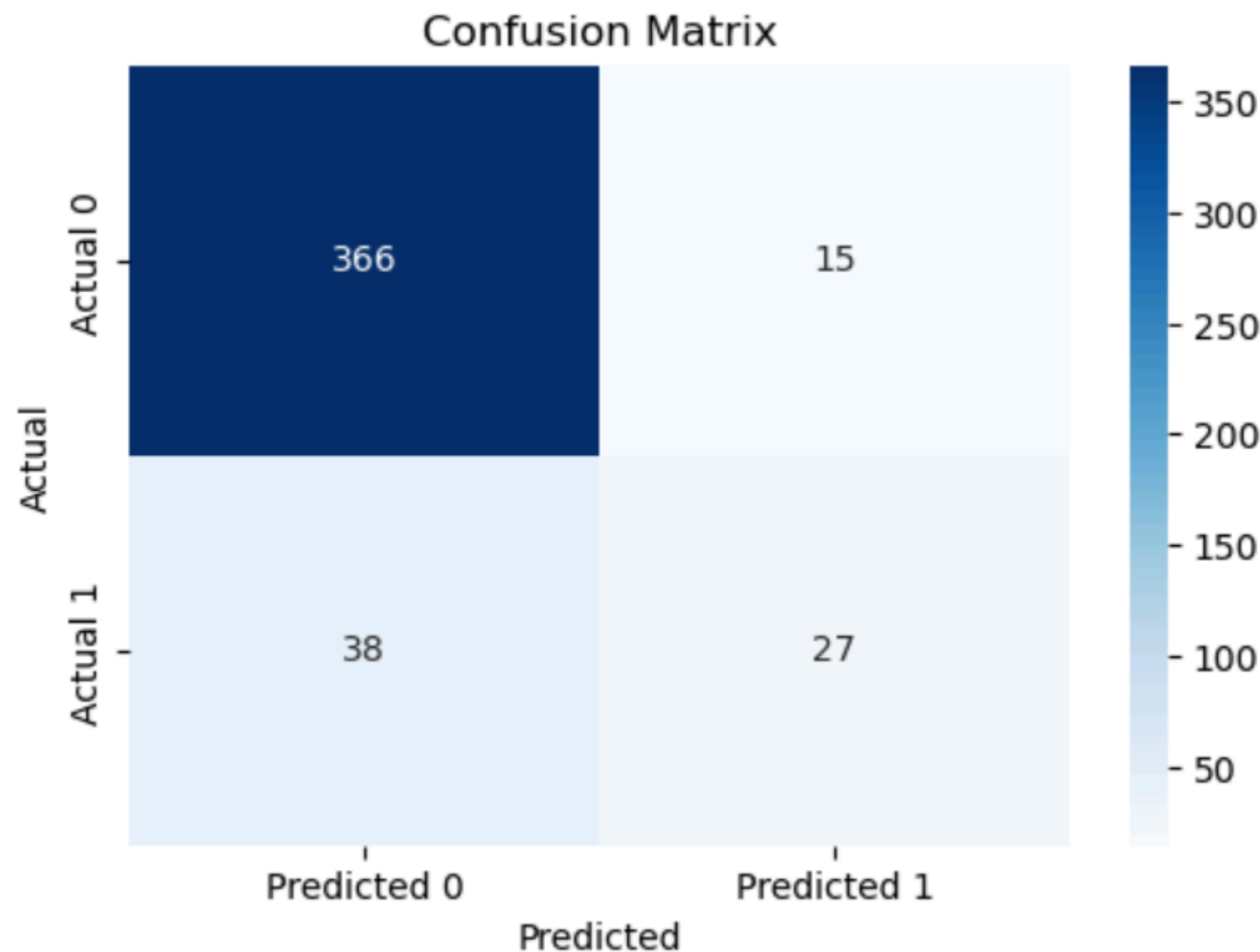
Evaluation metrics of Gradient Boosting:

**Accuracy:** 88.12%

**Precision:** 64.29%

**Recall:** 41.54%

**F1-Score:** 50.47%



**True Negatives (TN):** 27 cases where the model correctly predicted "0" (customer did not accept the campaign).

**False Positives (FP):** 38 cases where the model predicted "1" (accepted the campaign) but the actual was "0".

**False Negatives (FN):** 15 cases where the model predicted "0" (did not accept) but the actual was "1".

**True Positives (TP):** 366 cases where the model correctly predicted "1" (customer accepted the campaign).

- **MODEL EVALUATION SUMMARY**

Model	Accuracy	Precision	Recall	F1 Score
Linear Regression	78.03%	37.01%	72.31%	48.96%
Decision Tree	80.72%	38.46%	53.85%	44.87%
Random Forest	87.44%	62.86%	33.85%	44.00%
Gradient Boosting	88.12%	64.29%	41.54%	50.47%



# **HYPERPARAMETER TUNING**

**GRID SEARCH CV**

# HYPERPARAMETER TUNING - GRID SEARCH

Helps to find the best combination of hyperparameters for model by evaluating it across multiple possible configurations.

We found the best parameters for Gradient Boosting and the evaluation metrics using best parameters.

```
Best Parameters: {'learning_rate': 0.1, 'max_depth': 7, 'min_samples_split': 10, 'n_estimators': 200, 'subsample': 0.9}
Best Score: 0.9296691487330551
Accuracy: 85.87%
Precision: 51.56%
Recall: 50.77%
F1-Score: 51.16%
```

We found the best parameters for Random Forest and the evaluation metrics using best parameters.

```
Best Parameters: {'max_depth': 20, 'max_features': 'auto', 'min_samples_leaf': 1, 'min_samples_split': 2, 'n_estimators': 200}
```

Evaluation metrics of Random Forest Model with Grid Search CV:

```
Accuracy: 85.65%
Precision: 50.65%
Recall: 60.00%
F1-Score: 54.93%
```



# RESULTS

MODEL PREDICTION

# TESTING THE MODEL

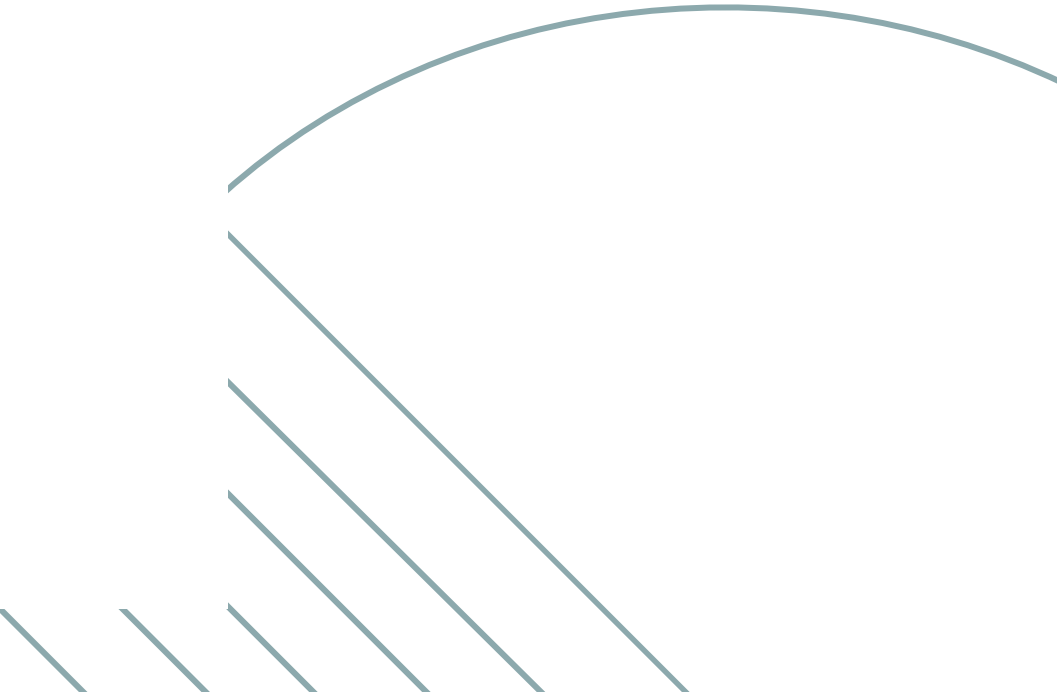
```
sample_features = X_test.iloc[19].values.reshape(1, -1)
predicted_response = rf_model.predict(sample_features)
actual_response = y_test.iloc[19]
print("Sample Features:", X_test.iloc[19])
print("Predicted Response:", predicted_response)
print("Actual Response:", actual_response)
if predicted_response == actual_response:
    print("The model predicted the response correctly!")
else:
    print("The model prediction was incorrect.")
```

The model has predicted the response correctly.

We find that Random Forest and Gradient Boosting performs well.

Sample Features: Education	
Marital_Status	2.000000
Income	47958.000000
Kidhome	0.000000
Teenhome	1.000000
Recency	8.000000
MntWines	268.000000
MntFruits	11.000000
MntMeatProducts	88.000000
MntFishProducts	15.000000
MntSweetProducts	3.000000
MntGoldProds	22.000000
NumDealsPurchases	2.000000
NumWebPurchases	6.000000
NumCatalogPurchases	3.000000
NumStorePurchases	5.000000
NumWebVisitsMonth	5.000000
AcceptedCmp3	0.000000
AcceptedCmp4	0.000000
AcceptedCmp5	0.000000
AcceptedCmp1	0.000000
AcceptedCmp2	0.000000
Complain	0.000000
Age	72.000000
Total_Campaigns_Accepted	0.000000
log_MntWines	0.139165
log_MntFruits	0.118497
log_MntMeatProducts	0.102975
log_MntFishProducts	0.081570
log_MntSweetProducts	-0.283324
log_MntGoldProds	-0.047908
high_spender_MntWines	0.000000
high_spender_MntFruits	0.000000
high_spender_MntMeatProducts	0.000000
high_spender_MntFishProducts	0.000000
high_spender_MntSweetProducts	0.000000
high_spender_MntGoldProds	0.000000
TotalSpent	407.000000
Used_Discount	1.000000
Name: 1565, dtype: float64	
Predicted Response: [0]	
Actual Response: 0	
The model predicted the response correctly!	

2.000000







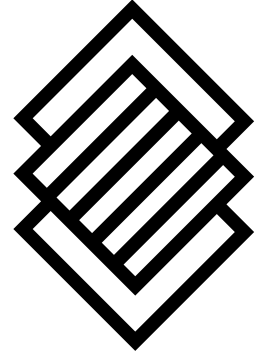
# SUMMARY

RECOMMENDATIONS

&

FUTURE ENHANCEMENTS

# RECOMMENDATIONS



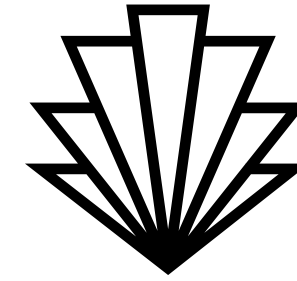
## Target High-Value Customers

Focus marketing efforts on customers with higher incomes and recent activity, as they are more likely to respond positively to campaigns.



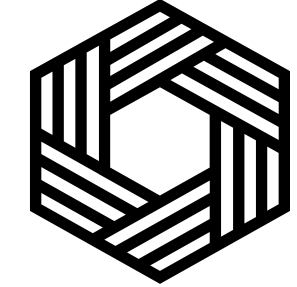
## Optimize Campaign Timing

Tailor campaigns based on recency, ensuring they reach customers shortly after their last purchase.



## Personalize Offers

Use spending data (e.g., wine, meat, or sweet products) to create personalized offers for customers, increasing the likelihood of a positive response.



## Improve Engagement with Low-Responders

For customers less likely to respond, try engagement strategies like discounts or special promotions to re-engage them

## SUMMARY

- Effectively utilized data-driven modeling to predict campaign responses, providing actionable insights for targeting and segmentation.
- The models predictive accuracy, precision, and recall demonstrate its potential to support more effective campaign strategies.
- We tested several input values, and both models correctly identified whether customers would respond to the campaign. The strong predictive capability of these models ensures they are suitable for real-world application, allowing for effective targeting in marketing campaigns.

## FUTURE ENHANCEMENTS

- Advanced Hyperparameter Tuning
- Advanced Ensemble Methods
- Feature Engineering Enhancements
- Data Augmentation Techniques

The background features four decorative geometric patterns in the corners. The top-left corner has a series of parallel diagonal lines in a light blue-grey color. The top-right corner contains a cluster of overlapping semi-circles in yellow, red, teal, and dark blue. The bottom-left corner also features a cluster of overlapping semi-circles in red, teal, and dark blue. The bottom-right corner has a series of parallel diagonal lines in a light blue-grey color, mirroring the top-left pattern.

**THANK YOU**