

MARKETING CAMPAIGN DATA ANALYSIS REPORT: IDENTIFYING HIGH-VALUE CUSTOMERS

PREPARED BY: IRIN MARY THOMAS

EMAIL ID: irinthomas0@gmail.com

I. Table of Contents

II. Table of Figures:	3
III. Introduction:	5
IV. Methodology:	5
A. Data Understanding:	5
1. Meta Data Analysis	5
2. Histograms:	9
3. Missing Data:	16
B. Data preparation:	17
1. Removing No Inference variable:	17
2. Data Cleaning:	18
3. Data Conversion:	21
C. Data Analysis:	25
1. Data Conversion:	25
2. Sentimental Analysis:	27
D. Data Exploration:	28
E. Data Mining	30
1. Training Logistic Regression model	32
2. Testing Logistic Regression model	33
3. Predictive Application.....	36
V. Discussion and reflection of the work	39
A. Discussion:	39
B. Reflection:	39
VI. Conclusion.....	40
VII. Appendix	40

II. Table of Figures:	2
Figure 1-Original Dataset	6
Figure 2-Data Type	7
Figure 3-Description of columns	7
Figure 4- Metadata for Numerical Columns	7
Figure 5-Metadata for categorical values.	8
Figure 6-Histogram for Age	9
Figure 7-Histogram for YRS_RESIDENCE	9
Figure 8-Histogram for AFFINITY CARD	9
Figure 9-Histogram for BULK_PACK_DISKETTES	10
Figure 10-Histogram for FLAT_PANEL_MONITOR	10
Figure 11-Histogram for HOME_THEATRE_PACKAGE	11
Figure 12-Histogram for BOOKKEEPING_APPLICATION	11
Figure 13-Histogram for PRINTER_SUPPLIES	11
Figure 14-Histogram for Y_BOX_Games	12
Figure 15-Histogram for OS_DOC_SET_KANJI	12
Figure 16-Bar Chart for CUST_GENDER	13
Figure 17-Bar Chart for CUST_MARITAL_STATUS	13
Figure 18-Bar Chart for CONTRY_Name	13
Figure 19-Bar Chart for CUST_INCOME_LEVEL	14
Figure 20-Bar Chart for EDUCATION	14
Figure 21-Bar Chart for OCCUPATION	15
Figure 22-Bar Chart for HOUSEHOLD_SIZE	15
Figure 23-Missing Data Report.	16
Figure 24-Variance of Printer Supplies	17
Figure 25-Mapping Invalid CUST_MARITAL_STATUS	18
Figure 26- Handling Invalid Entries in HOUSEHOLD_SIZE	18
Figure 27-Normalizing Occupation	19
Figure 28-Checking for non-numeric values	19
Figure 29-Missing Values after cleaning	20
Figure 30-Dataset after transformation	20
Figure 31-CUST_GENDER into binary	21
Figure 32-Frequency of COUNTRY_NAME	21
Figure 33-CUST_INCOME_LEVEL into 3 ordinal levels	22
Figure 34-Converting EDUCATION into ordinal numbers	23
Figure 35-One-Hot Encoding for OCCUPATION	23
Figure 36-The dataset after transformation.	24
Figure 37-Correlation Matrix	25
Figure 38-Sentimental Analysis	27
Figure 39-Unique values in the Comment Sentiment column.	27
Figure 40-Interactive Histogram Visualization	28
Figure 41-Exiting histogram Viewer.	29
Figure 42- df_copy1 column list	29
Figure 43-One Hot Encoding for Sentiment Column.	29
Figure 44-Correlation Matrix with sentiments	30
Figure 45-Top 10 Features.	31
Figure 46- Intercept and Coefficient of the model	31
Figure 47-Accuracy Score	32
Figure 48-ROC Curve.	33

Figure 49-Confusion Matrix	34
Figure 50-Classification Report	34
Figure 51- Customer Predictor Function	35
Figure 52-. Buttons for manual input , .csv file prediction	35
Figure 53-Example Output	36

III. Introduction:

This report presents a comprehensive analysis of a marketing campaign dataset from a retail company. The dataset contains 1500 customer records with 19 variables including socio-demographic information, product ownership details, and the target variable AFFINITY_CARD (1 = High-value, 0 = Low-value). The analysis follows a structured approach of data understanding, preparation, analysis, and exploration to provide insights for further data mining activities. The primary objective was to process this data for subsequent data

mining and predictive analysis, with the ultimate goal of distinguishing high-value customers from low-value ones.

The project followed a structured methodology that included:

1. Initial data understanding and metadata creation
2. Data quality assessment and cleaning
3. Feature transformation and engineering
4. Exploratory data analysis and visualization
5. Predictive model development using logistic regression
6. Model evaluation and implementation of a predictive application

Through this systematic approach, I aimed to derive actionable insights while developing a robust predictive model that could effectively identify high-value customers for targeted marketing efforts.

IV. Methodology:

A. Data Understanding:

1. Meta Data Analysis

1.1 Produce a metadata table to show characteristics of each attribute. The metadata table should contain attribute name, descriptions, Maximum, Minimum, Mean, and Std. Deviation and histogram for numeric data, and mode and bar chart for nominal data.

This section focuses on understanding the marketing campaign dataset through comprehensive metadata analysis. The objective is to examine the characteristics of each attribute in the dataset, identifying key statistical properties and distributions to inform further analysis. The dataset consists of 1,500 customer records with attributes ranging from demographic information (age, gender, marital status) to product ownership indicators as shown in **Figure 1. Original Dataset**

	CUST_ID	CUST_GENDER	AGE	CUST_MARITAL_STATUS	COUNTRY_NAME	\
0	101501	F	41	NeverM	United States of America	
1	101502	M	27	NeverM	United States of America	
2	101503	F	20	NeverM	United States of America	
3	101504	M	45	Married	United States of America	
4	101505	M	34	NeverM	United States of America	

	CUST_INCOME_LEVEL	EDUCATION	OCCUPATION	HOUSEHOLD_SIZE	YRS_RESIDENCE	\
0	J: 190,000 - 249,999	Masters	Prof.	2	4	
1	I: 170,000 - 189,999	Bach.	Sales	2	3	
2	H: 150,000 - 169,999	HS-grad	Cleric.	2	2	
3	B: 30,000 - 49,999	Bach.	Exec.	3	5	
4	K: 250,000 - 299,999	Masters	Sales	9+	5	

	AFFINITY_CARD	BULK_PACK_DISKETTES	FLAT_PANEL_MONITOR	\
0	0	1	1	
1	0	1	1	
2	0	1	0	
3	1	0	0	
4	1	1	1	

	HOME_THEATER_PACKAGE	BOOKKEEPING_APPLICATION	PRINTER_SUPPLIES	\
0	1	1	1	
1	0	1	1	
2	0	1	1	
3	1	1	1	
4	0	1	1	

	Y_BOX_GAMES	OS_DOC_SET_KANJI	\
0	0	0	
1	1	0	
2	1	0	
3	0	0	
4	0	0	

	COMMENTS
0	Shopping at your store is a hassle. I rarely s...
1	Affinity card is great. I think it is a hassle...
2	I purchased a new computer recently, but the m...
3	Affinity card is great. I think it is a hassle...
4	Why didn't you start a program like this befor...

Figure 1-Original Dataset

The initial steps involve understanding the dataset's structure and the datatypes as shown in **Figure 2.Data Type** The figure shows that there are 11 integer-type columns and 8 string-format data, including the COMMENTS column

```
print(df.dtypes)
```

CUST_ID	int64
CUST_GENDER	object
AGE	int64
CUST_MARITAL_STATUS	object
COUNTRY_NAME	object
CUST_INCOME_LEVEL	object
EDUCATION	object
OCCUPATION	object
HOUSEHOLD_SIZE	object
YRS_RESIDENCE	int64
AFFINITY_CARD	int64
BULK_PACK_DISKETTES	int64
FLAT_PANEL_MONITOR	int64
HOME_THEATER_PACKAGE	int64
BOOKKEEPING_APPLICATION	int64
PRINTER_SUPPLIES	int64
Y_BOX_GAMES	int64
OS_DOC_SET_KANJI	int64
COMMENTS	object
dtype:	object

Figure 2-Data Type

The initial step was to create a predefined dictionary as shown in Figure . This was created for storing the description of all the columns in the dataset and then using it in the metadata. The code also gave a comment” No Description available” if no description was found.

```
[248] def get_description(column):
      descriptions = {
          'CUST_ID': 'Unique customer identifier',
          'CUST_GENDER': 'Customer gender (M/F)',
          'AGE': 'Customer age in years',
          'CUST_MARITAL_STATUS': 'Customer marital status',
          'COUNTRY_NAME': 'Country of residence',
          'CUST_INCOME_LEVEL': 'Income level category',
          'EDUCATION': 'Education level',
          'OCCUPATION': 'Occupation category',
          'HOUSEHOLD_SIZE': 'Number of people in household',
          'YRS_RESIDENCE': 'Years at current residence',
          'AFFINITY_CARD': 'Target variable (1=High-value, 0=Low-value)',
          'BULK_PACK_DISKETTES': 'Purchased bulk pack diskettes',
          'FLAT_PANEL_MONITOR': 'Purchased flat panel monitor',
          'HOME_THEATER_PACKAGE': 'Purchased home theater package',
          'BOOKKEEPING_APPLICATION': 'Purchased bookkeeping application',
          'PRINTER_SUPPLIES': 'Purchased printer supplies',
          'Y_BOX_GAMES': 'Purchased Y-box games',
          'OS_DOC_SET_KANJI': 'Purchased OS doc set in Kanji',
          'COMMENTS': 'Free text customer comments'
      }
      return descriptions.get(column, 'No description available')
```

Figure 3-Description of columns

A metadata table was generated to provide a comprehensive overview of the dataset's attributes. For numeric attributes, the metadata includes the attribute name, description, maximum value, minimum value, mean, and standard deviation as shown in **Figure 4. Metadata for Numerical Columns**

Numeric Attributes Metadata:					
Attribute	Description	Max	Min	Mean	Std Dev
CUST_ID	Unique customer identifier	103000	101501	102250	433.157
AGE	Customer age in years	90	17	38.892	13.6364
YRS_RESIDENCE	Years at current residence	14	0	4.08867	1.92092
AFFINITY_CARD	Target variable (1=High-value, 0=Low-value)	1	0	0.253333	0.435065
BULK_PACK_DISKETTES	Purchased bulk pack diskettes	1	0	0.628	0.4835
FLAT_PANEL_MONITOR	Purchased flat panel monitor	1	0	0.582	0.493395
HOME_THEATER_PACKAGE	Purchased home theater package	1	0	0.575333	0.494457
BOOKKEEPING_APPLICATION	Purchased bookkeeping application	1	0	0.880667	0.324288
PRINTER_SUPPLIES	Purchased printer supplies	1	1	1	0
Y_BOX_GAMES	Purchased Y-box games	1	0	0.286667	0.452355
OS_DOC_SET_KANJI	Purchased OS doc set in Kanji	1	0	0.002	0.0446915

Figure 4- Metadata for Numerical Columns

The age distribution of customers spans from 17 to 90 years, with a mean age of approximately 39 years. The standard deviation of 13.64 indicates a moderately diverse customer base, suggesting that the company's products appeal to a wide age demographic. Customers have lived at their current addresses for an average of 4.09 years. The range extends from new residents (0 years) to long-term residents (14 years), with a standard deviation of 1.92

years. The AFFINITY_CARD attribute has a mean of 0.25, indicating that approximately 25% of customers are classified as high-value customers. PRINTER_SUPPLIES shows a mean of 1.0 with zero standard deviation, indicating universal purchase among customers. BOOKKEEPING_APPLICATION shows high penetration at 88%, indicating strong market acceptance, while OS_DOC_SET_KANJI has very low penetration at just 0.2%

For categorical attributes, the metadata includes the attribute name, description, and mode as shown in **Figure 5. Metadata for categorical values**

Categorical Attributes Metadata:		
Attribute	Description	Mode
CUST_GENDER	Customer gender (M/F)	M
CUST_MARITAL_STATUS	Customer marital status	Married
COUNTRY_NAME	Country of residence	United States of America
CUST_INCOME_LEVEL	Income level category	J: 190,000 - 249,999
EDUCATION	Education level	HS-grad
OCCUPATION	Occupation category	Exec.
HOUSEHOLD_SIZE	Number of people in household	3
COMMENTS	No description available	Affinity card is great. I think it is a hassle to have to remember to bring it in every time though.

Figure 5-Metadata for categorical values.

The code iterates through numeric columns, calculates descriptive statistics (max(), min(), mean(), std()), and generates histograms. Histograms for numeric variables showed distribution shape and central tendency, while bar charts for categorical variables illustrated the frequency of each category. By looking at the figure, it can be seen that the gender distribution of the customer base skews toward male customers, as indicated by the mode value of “M” for the CUST_GENDER attribute. Moreover, analyzing the table reveals that the United States of America is the predominant country of residence for customers. The most common income level is J: 190,000 – 249,999, indicating an affluent customer base with substantial purchasing power. Interestingly, the most common education level is a High School graduate, which defies typical education-income correlations. The most frequent occupation category is Executive, aligning with the high income levels. The most common household size is three people, and the most frequent comment indicates that while customers appreciate the affinity card program, they find it inconvenient to remember to bring it for each transaction.

2. Histograms:

• Numerical columns:

- Age Distribution: The age distribution histogram reveals a diverse customer base spanning from teenagers to seniors, with the highest concentration of customers in their 30s and early 40s. The distribution

shows a slight positive skew, with fewer customers in the older age brackets above 60 years as shown in **Figure 6.Histogram for Age**.

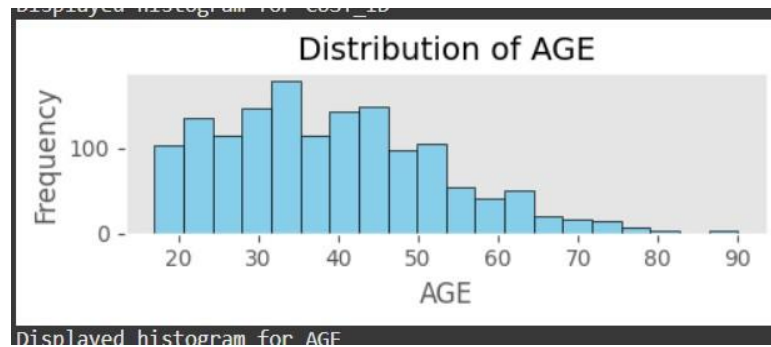


Figure 6-Histogram for Age

- b. Years of Residence Distribution: The years of residence histogram shows that most customers have lived at their current addresses for 3-6 years, with peak frequency at 4 years. Very few customers have resided at the same location for more than 8 years, indicating a relatively mobile customer base as shown in **Figure 7. Histogram for YRS_RESIDENCE**.

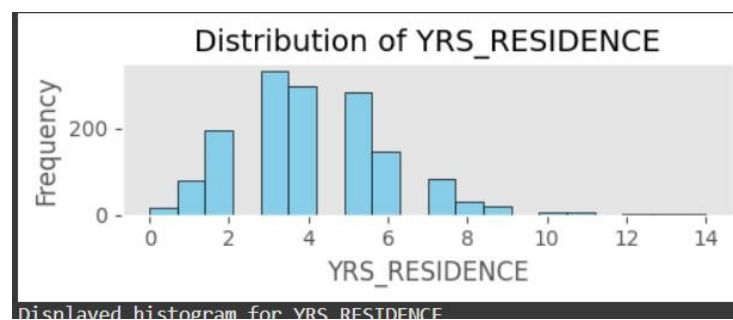


Figure 7-Histogram for YRS_RESIDENCE

- c. Affinity Card Distribution: The affinity card histogram confirms our binary target variable, with approximately 75% of customers classified as low-value (0) and 25% as high-value (1) as shown in **Figure 8. Histogram for AFFINITY_CARD**

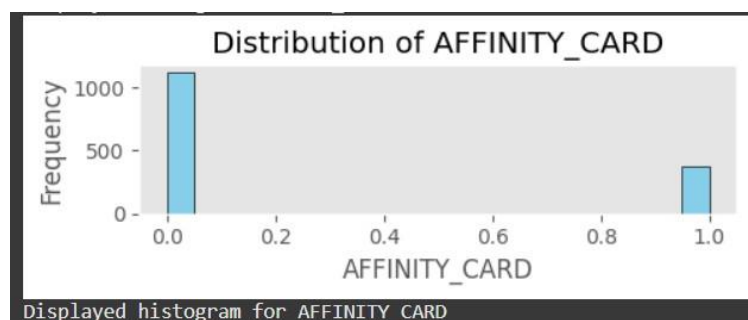


Figure 8-Histogram for AFFINITY CARD

- d. Bulk Pack Diskettes Distribution: The bulk pack diskettes histogram shows a fairly even split between customers who have purchased this product (value 1) and those who haven't (value 0), with a slight majority having made purchases as shown in **Figure 9. Histogram for BULK_PACK_DISKETTES**

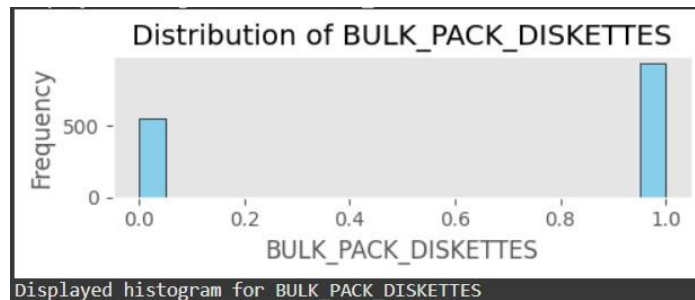


Figure 9-Histogram for BULK_PACK_DISKETTES

- e. Flat Panel Monitor Distribution: The flat panel monitor histogram displays a bimodal distribution with slightly more customers having purchased this product (value 1) than those who haven't (value 0), as shown in **Figure 10. Histogram for FLAT_PANEL_MONITOR**

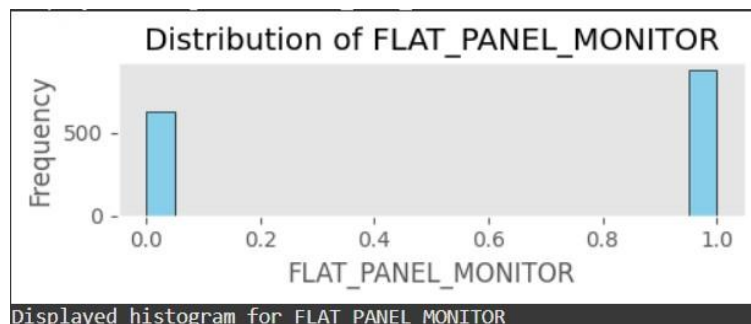


Figure 10-Histogram for FLAT_PANEL_MONITOR

- f. Home Theater Package Distribution: The home theater package histogram reveals a pattern similar to flat panel monitors, with a somewhat balanced distribution between purchasers and nonpurchasers. The slight edge toward purchasers (value 1) aligns with our finding of approximately 58% market penetration for this

product, as shown in **Figure 11. Histogram for HOME_THEATRE_PACKAGE**

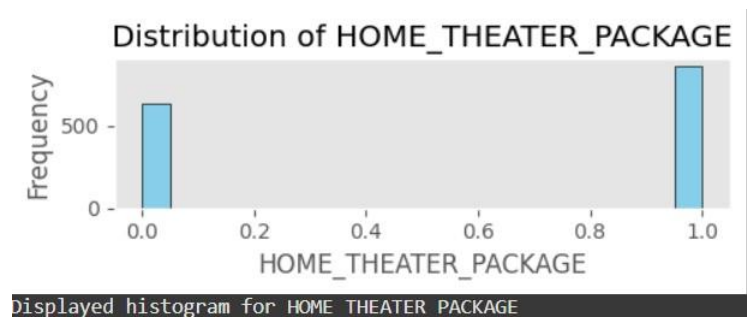


Figure 11-Histogram for HOME_THEATRE_PACKAGE

- g. Bookkeeping Application Distribution: The bookkeeping application histogram shows a strong skew toward purchases, with the vast majority of customers having purchased this product (value 1). The small bar at value 0 represents just 12% of customers who haven't purchased this apparently essential product, as shown in **Figure 12. Histogram for BOOKKEEPING_APPLICATION**

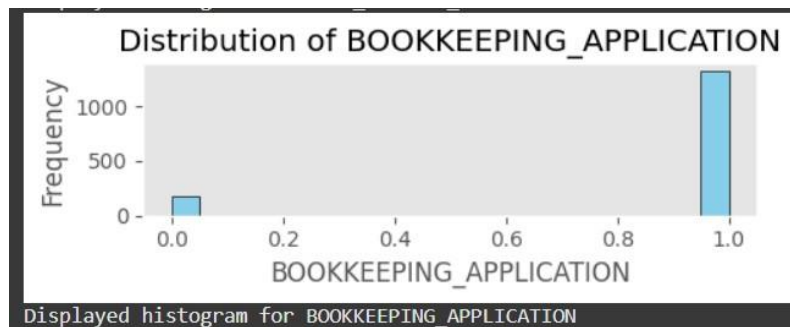


Figure 12-Histogram for BOOKKEEPING_APPLICATION

- h. Printer Supplies Distribution: The printer supplies histogram shows a single bar at value 1, visually confirming that every single customer in our dataset has purchased this product. This unique single-bar distribution stands out among our product variables as shown in **Figure 13. Histogram for PRINTER_SUPPLIES**

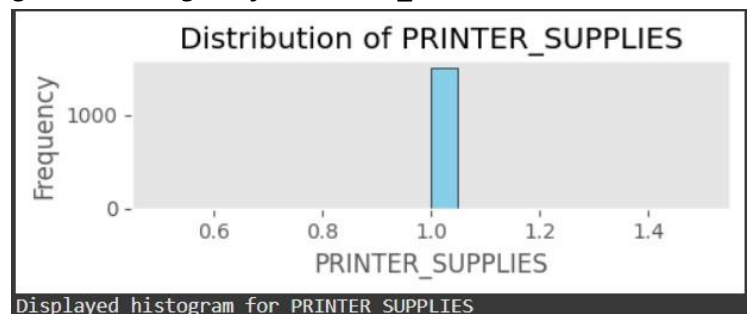


Figure 13-Histogram for PRINTER_SUPPLIES

- i. Y-Box Games Distribution: The Y-Box games histogram reveals that a minority of customers (approximately 29%) have purchased this product

(value 1), with most customers not engaging with this offering. The larger bar at value 0 indicates that this product appeals to a more specialized segment of our customer base as shown in **Figure 14. Histogram for Y_BOX_GAMES**

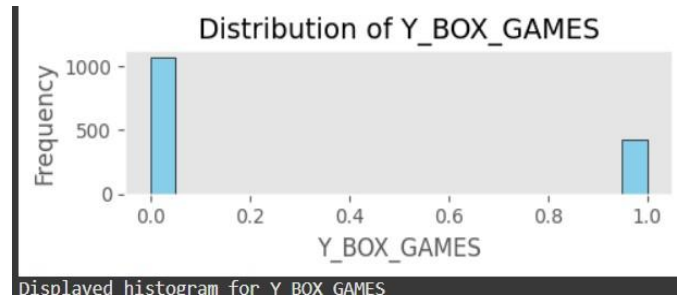


Figure 14-Histogram for Y_BOX_GAMES

- j. OS_DOC_SET_KANJI Distribution: The OS Doc Set Kanji histogram shows an overwhelming majority of customers have not purchased this highly specialized product, with just a tiny fraction (0.2%) making purchases as shown in **Figure 15. Histogram for OS_DOC_SET_KANJI**

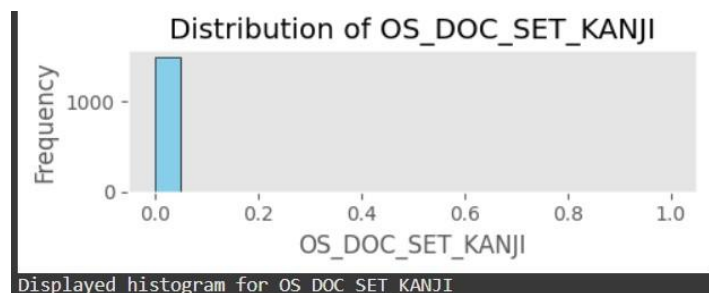


Figure 15-Histogram for OS_DOC_SET_KANJI

• **Categorical Distributions:**

- a. Customer Gender Distribution: The customer gender bar chart shows a clear male majority in our customer base, with 1,014 male customers compared to 486 female customers. The visual representation makes it immediately apparent that men outnumber women by more than 2:1 in our dataset, as shown in **Figure 16. Bar Chart for CUST_GENDER**

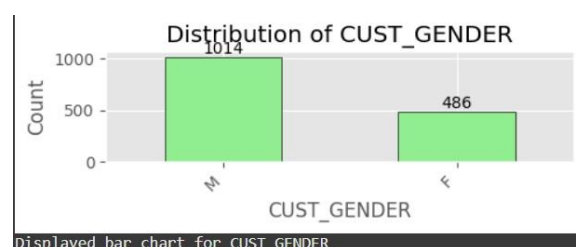


Figure 16-Bar Chart for CUST_GENDER

- b. Marital Status Distribution: The marital status bar chart reveals that married customers form the largest segment (712), followed by never-married customers (485) and divorcees (187). Smaller numbers of customers are separated (52), widowed (40), or in other categories, as shown in **Figure 17. Bar Chart for CUST_MARITAL_STATUS**

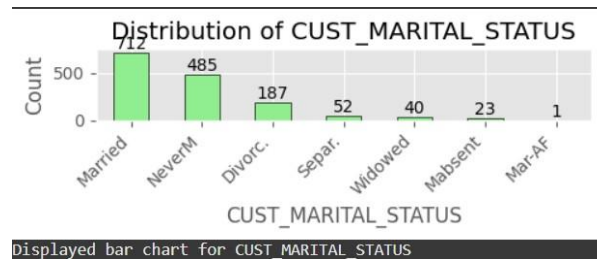


Figure 17-Bar Chart for CUST_MARITAL_STATUS

- c. Country Distribution: The country bar chart dramatically illustrates the overwhelming concentration of our customer base in the United States of America (1,344 customers), with minimal presence in other countries. The second largest market, Argentina, has just 46 customers, creating a stark visual contrast between the first bar and all others. This visualization emphasizes that our market is essentially US-centric, with potential opportunities for international expansion as shown in **Figure 18. Bar Chart for CONTRY_Name**

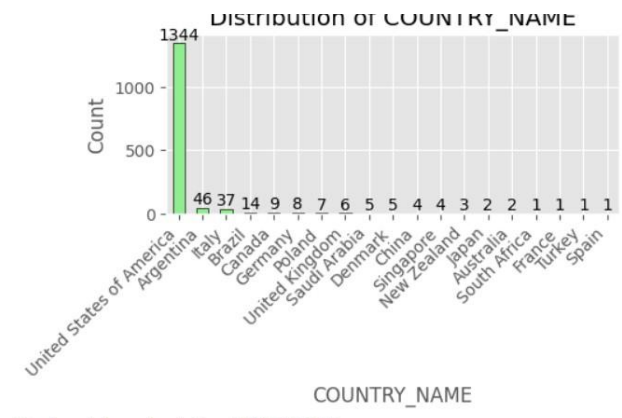


Figure 18-Bar Chart for CONTRY_Name

- d. Income Level Distribution: The income level bar chart reveals a concentration of customers in higher income brackets, with the J: \$190,000-\$249,999 category having the highest frequency (339 customers) as shown in **Figure 19. Bar Chart for CUST_INCOME_LEVEL**. This visualization reinforces our understanding that most customers have

significant purchasing power, informing premium product and pricing strategies

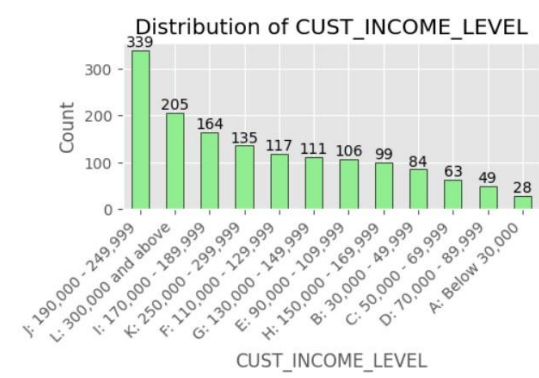


Figure 19-Bar Chart for CUST_INCOME_LEVEL

- e. Education Distribution: The education bar chart shows High School graduates as the largest education segment (482), followed by Bachelor's (359) and Associate's degrees (245) as shown in **Figure 20. Bar Chart for EDUCATION**. This visualization challenges assumptions about formal education and success, suggesting our customers have found various paths to financial achievement.

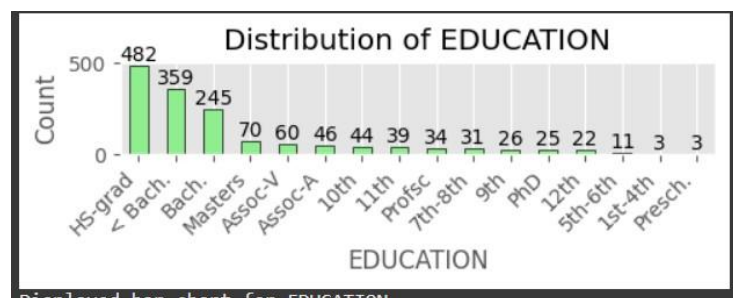


Figure 20-Bar Chart for EDUCATION

Figure 20.

- f. Occupation Distribution: The occupation bar chart displays a relatively balanced distribution across several professional categories, with Executives (197), Crafts (196), and Clerical workers (178) forming the largest segments, as shown in **Figure 21. Bar Chart for OCCUPATION**

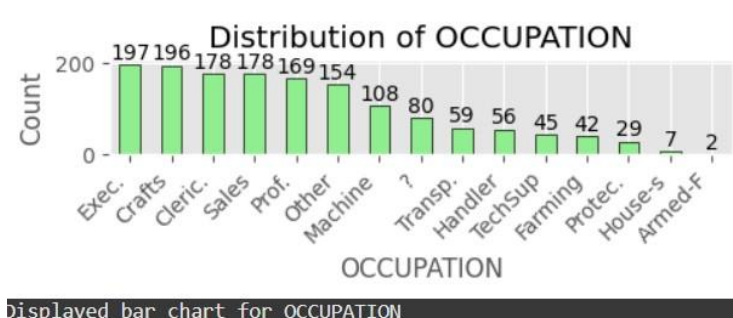


Figure 21-Bar Chart for OCCUPATION

- g. Household Size Distribution: The household size bar chart reveals that three-person households are most common among our customers, with 635 customers falling into this category, as shown in **Figure 22. Bar Chart for HOUSEHOLD_SIZE**. The distribution shows a clear declining pattern as household size increases or decreases from this three-person peak, with two-person households (371) forming the second largest segment. Single-person households (212) and households with four or more members (163) represent substantial but smaller segments of our customer base.

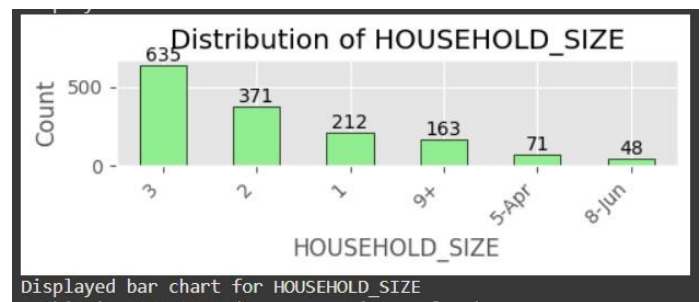


Figure 22-Bar Chart for HOUSEHOLD_SIZE

3. Missing Data:

3.1 Describe missing or erroneous data with suggestions for handling methods of each attribute. DO NOT clean them at this stage. Missing data should include all types of missing data, such as Null, Blank, unknown, etc. The error should include any invalid or mismatched data.

The next phase analyzes missing and error data within the customer dataset. By identifying these issues, we can enhance the quality of the data, which is essential for accurate analysis and modelling. The report details the findings regarding missing values, invalid entries, and suggested handling methods for each identified issue. To conduct this analysis, two functions were implemented: `analyze_missing_data` and `suggest handling`. The first function examines each attribute in the dataset for various types of missing or erroneous data, including null values, empty strings, placeholder values, numeric anomalies, outliers, and invalid categories. The second function provides tailored recommendations for addressing each issue based on the specific attribute and type of problem which will give an out as shown in **Figure 23**.

Missing Data Analysis Summary

Missing Data Analysis						Handling Suggestion
Attribute	Null Values	Empty Strings	Special Missing (?, etc)	Total Missing	% Missing	
OCCUPATION	0	0	80	80	5.33%	Create 'Unknown' category for '?' values or impute based on other customer attributes
COMMENTS	73	0	0	73	4.87%	Make 'Other comment' to store the records

Error Data Analysis			Handling Suggestion
Attribute	Error Types		
CUST_MARITAL_STATUS	263 invalid values: 'Divorc.' (187), 'Separ.' (52), 'Mabsent' (23); 262 inconsistent values (Divorc./Mabsent/Separ.)		Standardize values: 'Divorc.' to 'Divorced', 'Separ.' to 'Separated'
EDUCATION	1405 invalid values: 'HS-grad' (482), '< Bach.' (359), 'Bach.' (245); 1226 abbreviated/truncated education levels		Map to standard categories: 'Bach.' to 'Bachelors', 'HS-grad' to 'High School'
HOUSEHOLD_SIZE	282 invalid values: '9+' (163), '5-Apr' (71), '8-Jun' (48); 211 inconsistent formats (9+/8-Jun)		Convert '9+' to '9' and '8-Jun' to '8' (or investigate d

Figure 23-Missing Data Report.

Missing Data Analysis Summary:

1.**OCCUPATION**: The attribute has 80 special missing values marked by '?', representing 5.33% of the total records. This indicates a significant number of unknown occupations, suggesting the need for either categorization or imputation based on other customer attributes.

2.**COMMENTS**: There are 73 null values (4.87% of records) in the COMMENTS attribute. While this is not excessively high, handling these missing values might involve using mode imputation or removing the records if they are deemed non-essential.

3.**CUST_MARITAL_STATUS**: This attribute contains 263 invalid values, with 'Divorc.', 'Separ.', and 'Mabsent' being the most frequent. Additionally, the code identifies 262 inconsistent values likely due to these abbreviations. The suggested handling method is to standardize these values by mapping 'Divorc.' to 'Divorced', 'Separ.' to 'Separated', and 'Mabsent' to 'Separated' to ensure consistency.

4.**EDUCATION**: The 'EDUCATION' attribute has a significant number of invalid values (1405), with 'HS-grad', '< Bach.', and 'Bach.' being the most common. The code also flags 1226 instances as abbreviated or truncated education levels. The recommended approach is to map these abbreviated forms to standard, full categories (e.g., 'Bach.' to 'Bachelors', 'HS-grad' to 'High School'). Further investigation might be needed for less clear abbreviations like '< Bach.'.

5.**HOUSEHOLD_SIZE**: This attribute shows 282 invalid values, including '9+', '5-Apr', and '8-Jun'. The code also detects 211 inconsistent formats, likely referring to '9+' and potentially the date-like entries

B. Data preparation:

1. Removing No Inference variable:

2.1. Write Python programs to remove variables with no inference to the target justifications and keep comments for further processing.

The two columns that had no inference to the target variable were Customer ID and Printer Supplies. The justification for removing these columns is:

- Zero Variance Columns: Columns with only one unique value (zero variance) , as shoen in **Figure 24. Variance of Printer Supplies** do not provide any information useful for prediction and can mislead the model. These columns are dropped to streamline the dataset.

```
# Number of unique values in the column
n_unique = df['PRINTER_SUPPLIES'].nunique()
print("Unique values:", n_unique)

if n_unique > 1:
    print("Column has variance")
else:
    print("Column is constant (zero variance)")

Unique values: 1
Column is constant (zero variance)
```

Figure 24-Variance of Printer Supplies

- Domain Knowledge: Certain columns, such as CUST_ID (a unique identifier) and PRINTER_SUPPLIES (if deemed irrelevant to the target variable), are removed based on the understanding that they do not contribute to the predictive power of the model.

2. Data Cleaning:

2.2. Write Python programs to clean data and provide justifications.

This section details the preprocessing steps applied to clean and standardize the dataset before further analysis. It includes handling invalid or inconsistent values, addressing outliers, and examining missing values. Below is a breakdown of the cleaning actions and justifications:

Mapping Invalid CUST_MARITAL_STATUS: The CUST_MARITAL_STATUS column contained several variations and abbreviations, such as 'Divorc.', 'Mabsent', and 'Separ.'. These inconsistencies were addressed by mapping these values to their full or standardized forms (e.g., 'Divorc.' was mapped to 'Divorced'), as shown in **Figure 25. Mapping Invalid CUST_MARITAL_STATUS**

```
[40] # Mapping invalid marital status
marital_mapping = {
    'NeverM': 'Never Married',
    'Married': 'Married',
    'Divorc.': 'Divorced',
    'Mabsent': 'Married-Apart',
    'Separ.': 'Separated',
    'Mar-AF': 'Married-Forces',
    'Widowed': 'Widowed'# Add other mappings as needed
}

df['CUST_MARITAL_STATUS'] = df['CUST_MARITAL_STATUS'].replace(marital_mapping)
df['CUST_MARITAL_STATUS'] = df['CUST_MARITAL_STATUS'].fillna('Other')

[41] print(df['CUST_MARITAL_STATUS'].unique())
```

['Never Married' 'Married' 'Divorced' 'Married-Apart' 'Separated'
'Widowed' 'Married-Forces']

Figure 25-Mapping Invalid CUST_MARITAL_STATUS

This standardization ensures that marital status categories are consistently represented, avoiding the creation of duplicate categories during analysis. Additionally, any missing values in the CUST_MARITAL_STATUS column were replaced with 'Other'. Standardizing the marital status values enhances data consistency and ensures accurate representation in analyses, improving the reliability of demographic insights.

- a) Handling Invalid Entries in HOUSEHOLD_SIZE: Invalid formats such as '9+', '8-Jun', and '5-Apr' are replaced with numeric equivalents and converted to integers, as shown in **Figure 26. Handling Invalid Entries in HOUSEHOLD_SIZE**

```
# Handle invalid entries in 'HOUSEHOLD_SIZE'
df['HOUSEHOLD_SIZE'] = df['HOUSEHOLD_SIZE'].replace({
    '9+': '9',
    '8-Jun': '8',
    '5-Apr': '5'
}).astype(int)

# Confirm all fixes
print("Unique values in HOUSEHOLD_SIZE after cleanup:", df['HOUSEHOLD_SIZE'].unique())

print("Missing values after cleanup:\n", df.isnull().sum())
```

Unique values in HOUSEHOLD_SIZE after cleanup: [2 3 9 8 1 5]

Figure 26- Handling Invalid Entries in HOUSEHOLD_SIZE

After replacing these invalid entries, the entire column is converted to an integer data type to ensure it can be used for numerical calculations. The output confirms that the unique values in the column are now valid integers representing household sizes. This transformation normalizes the data, allowing for accurate numerical analysis and calculations.

- b) Normalizing OCCUPATION Casing: This code snippet normalizes the casing of the 'OCCUPATION' attribute by converting the first letter of each word to uppercase and the rest to lowercase (title case), as shown in **Figure 27**.

Normalizing Occupation. This is important for data consistency, as it ensures that different entries referring to the same occupation but with different casing (e.g., "sales", "Sales", "SALES") are treated as the same category during analysis.

```
# 3. Normalize OCCUPATION casing
df['OCCUPATION'] = df['OCCUPATION'].str.title()
# Justification: ensures occupations like "sales" and "Sales" merge into one category.
```

Figure 27-Normalizing Occupation

- c) Ensuring Numeric Columns Contain Only Numeric Data: A final check confirms that all integer/float columns contain valid numeric data, catching any stray anomalies before modelling, as shown in **Figure 27**.

Checking for non-numeric values

```
numeric_cols = df.select_dtypes(include=['int64', 'float64']).columns
for col in numeric_cols:
    # Coerce errors and count
    errors = pd.to_numeric(df[col], errors='coerce').isna().sum()
    assert errors == 0, f"Found {errors} invalid entries in {col}"
# Justification: catches any remaining stray non-numeric entries early.
```

Figure 28-Checking for non-numeric values

After all cleaning operations, a final check for missing values is performed. This confirms that all identified missing and erroneous data issues have been addressed, ensuring the dataset is ready for further analysis, as shown in **Figure 29. Missing Values after cleaning**

```
# 6. Check final missingness
print("Missing values after cleaning:")
print(df.isnull().sum())

# COMMENTS remains untouched for text an
```



Column	Missing Values
CUST_GENDER	0
AGE	0
CUST_MARITAL_STATUS	0
COUNTRY_NAME	0
CUST_INCOME_LEVEL	0
EDUCATION	0
OCCUPATION	0
HOUSEHOLD_SIZE	0
YRS_RESIDENCE	0
AFFINITY_CARD	0
BULK_PACK_DISKETTES	0
FLAT_PANEL_MONITOR	0
HOME_THEATER_PACKAGE	0
BOOKKEEPING_APPLICATION	0
Y_BOX_GAMES	0
OS_DOC_SET_KANJI	0
COMMENTS	73

dtype: int64

Figure 29-Missing Values after cleaning

3. Data Conversion:

2.3 Write a Python program to convert variables as per the following requirements. You should provide screenshots of your Python code with comments as well as data before and after conversion.

This section focuses on transforming categorical variables into appropriate numeric formats to support effective statistical modeling and machine learning algorithms. The data before any transformation is shown in **Figure 30.Dataset after transformation**

	CUST_GENDER	AGE	CUST_MARITAL_STATUS	COUNTRY_NAME	
0	F	41.0	Never Married	United States of America	
1	M	27.0	Never Married	United States of America	
2	F	20.0	Never Married	United States of America	
3	M	45.0	Married	United States of America	
4	M	34.0	Never Married	United States of America	

	CUST_INCOME_LEVEL	EDUCATION	OCCUPATION	HOUSEHOLD_SIZE	YRS_RESIDENCE	
0	J: 190,000 - 249,999	Masters	Prof.	2	4.0	
1	I: 170,000 - 189,999	Bach.	Sales	2	3.0	
2	H: 150,000 - 169,999	HS-grad	Cleric.	2	2.0	
3	B: 30,000 - 49,999	Bach.	Exec.	3	5.0	
4	K: 250,000 - 299,999	Masters	Sales	9+	5.0	

	AFFINITY_CARD	BULK_PACK_DISKETTES	FLAT_PANEL_MONITOR	
0	0	1	1	
1	0	1	1	
2	0	1	0	
3	1	0	0	
4	1	1	1	

	HOME_THEATER_PACKAGE	BOOKKEEPING_APPLICATION	Y_BOX_GAMES	
0	1	1.0	0	
1	0	1.0	1	
2	0	1.0	1	
3	1	1.0	0	
4	0	1.0	0	

	OS_DOC_SET_KANJI	COMMENTS
0	0.0	Shopping at your store is a hassle. I rarely s...
1	0.0	Affinity card is great. I think it is a hassle...
2	0.0	I purchased a new computer recently, but the m...
3	0.0	Affinity card is great. I think it is a hassle...
4	0.0	Why didn't you start a program like this befor...

Figure 30-Dataset after transformation

- a) **CUST_GENDER into binary F - 0, M -1:** The CUST_GENDER column is converted into a binary format where 'F' is mapped to 0 and 'M' is mapped to 1 using the .map() function, as shown in **Figure 31. CUST_GENDER into binary**. Converting gender into binary values simplifies analysis and modeling, allowing for easier statistical computation.

```
[ ] df['CUST_GENDER'] = df['CUST_GENDER'].map({'F': 0, 'M': 1})
print(df['CUST_GENDER'])
```

```
0      0
1      1
2      0
3      1
4      1
..
1495   1
1496   1
1497   1
1498   1
1499   0
Name: CUST_GENDER, Length: 1500, dtype: int64
```

Figure 31-CUST_GENDER into binary

- b) **COUNTRY_NAME into ordinal number based on their frequency of occurrence in the data set in descending order:** The COUNTRY_NAME column is converted into ordinal numbers based on each country's occurrence frequency, as shown in **Figure 32.Frequency of COUNTRY_NAME** . The value_counts() method is used to get the frequency of each country, and then a dictionary is created to map each country to its rank based on frequency. The countries are ranked in descending order of frequency, and the map() function is used to apply this ranking to the COUNTRY_NAME column. This transformation enables algorithms to account for the importance of a country's representation in the data while maintaining numerical consistency.

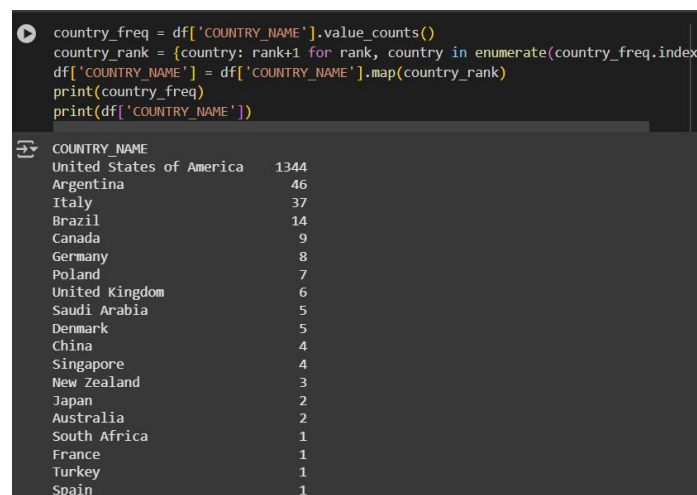


Figure 32-Frequency of COUNTRY_NAME

- c) **CUST_INCOME_LEVEL into 3 ordinal levels 1 – low income, 2 -middle income, and 3 – high income.** The goal of converting the CUST_INCOME_LEVEL variable is to simplify the income categories into three distinct ordinal levels: low income(1), middle income(2), and high income(3), as shown in **Figure 33.CUST_INCOME_LEVEL into 3 ordinal levels** . This transformation allows for easier analysis and comparison of customer income levels in the dataset.By reducing the income categories to three ordinal levels, the data becomes easier to analyze. Analysts can quickly categorize customers into broad income ranges without having to consider multiple specific brackets.

```
[ ] income_level_map = {
    'A: Below 30,000': 1,
    'B: 30,000 - 49,999': 1,
    'C: 50,000 - 69,999': 2,
    'D: 70,000 - 89,999': 2,
    'E: 90,000 - 109,999': 2,
    'F: 110,000 - 129,999': 3,
    'G: 130,000 - 149,999': 3,
    'H: 150,000 - 169,999': 3,
    'I: 170,000 - 189,999': 3,
    'J: 190,000 - 249,999': 3,
    'K: 250,000 - 299,999': 3,
    'L: 300,000 and above': 3
}

# Apply mapping
df['CUST_INCOME_LEVEL'] = df['CUST_INCOME_LEVEL'].map(income_level_map)

# Show result
print("Mapped income levels:")
print(df['CUST_INCOME_LEVEL'].value_counts(dropna=False))
```

```

Mapped income levels:
CUST_INCOME_LEVEL
3      1170
2       218
1        112
Name: count, dtype: int64
```

Figure 33-CUST_INCOME_LEVEL into 3 ordinal levels

- d) **EDUCATION into ordinal numbers based on USA education level (do your research if necessary) in ascending order:** The EDUCATION column is converted into ordinal numbers based on the typical education level progression in the USA, as shown in **Figure 34. Converting EDUCATION into ordinal numbers**. A dictionary education_map is defined to map each education level to its corresponding ordinal value, reflecting the order of educational stages. The map() function is used to apply this mapping to the EDUCATION column. By ranking countries based on their frequency, we provide a clear ordinal structure that reflects their relative significance in the dataset.


```
[236] education_map = {
    'Preschool': 1,
    '1st-4th': 2,
    '5th-6th': 3,
    '7th-8th': 4,
    '9th': 5,
    '10th': 6,
    '11th': 7,
    '12th': 8,
    'Less than Bachelors': 9,
    'High School': 10,
    'Associate of Arts': 11,
    'Associate V': 12,
    'Bachelors': 13,
    'Masters': 14,
    'PhD': 15,
    'Professional': 16
}
df['EDUCATION'] = df['EDUCATION'].map(education_map)
print(df['EDUCATION'])
```

0	14
1	13
2	10
3	13
4	14
...	...
1495	6
1496	13
1497	10

Figure 34-Converting EDUCATION into ordinal numbers

e) **OCCUPATION into a series of binary columns using onehot encoding transforms:** The OCCUPATION column is converted into a series of binary columns using one-hot encoding. The `pd.get_dummies()` function is used to create a new binary column for each unique occupation. The `prefix` argument adds "OCCUPATION_" to the beginning of each new column name, as shown in **Figure 35.One-Hot Encoding for OCCUPATION**. The `dtype` argument is used to ensure that the new columns are of integer type. One-hot encoding converts the nominal OCCUPATION column into multiple binary columns, each representing a unique job type. This prevents the model from assuming any ordinal relationship between different occupations and makes the data suitable for algorithms that require numeric input.

```
[277] df = pd.get_dummies(df, columns=['OCCUPATION'],dtype=int)

[278] occupation_columns = [col for col in df.columns if col.startswith('OCCUPATION')]
print(occupation_columns)
```

['OCCUPATION_Armed Forces', 'OCCUPATION_Clerical', 'OCCUPATION_Craftsman', 'OCCUPATION_...

Figure 35-One-Hot Encoding for OCCUPATION

The dataset after all the data transformations mentioned above looks like **Figure 36. Dataset after transformation**.

	CUST_GENDER	AGE	CUST_MARITAL_STATUS	COUNTRY_NAME	CUST_INCOME_LEVEL	\
0	0	41.0	Never Married	1	3	
1	1	27.0	Never Married	1	3	
2	0	20.0	Never Married	1	3	
3	1	45.0	Married	1	1	
4	1	34.0	Never Married	1	3	
	EDUCATION	HOUSEHOLD_SIZE	YRS_RESIDENCE	AFFINITY_CARD	\	
0	14	2	4.0	0		
1	13	2	3.0	0		
2	10	2	2.0	0		
3	13	3	5.0	1		
4	14	9+	5.0	1		
	BULK_PACK_DISKETTES	...	OCCUPATION_Farming	OCCUPATION_Handler	\	
0	1	...	0	0		
1	1	...	0	0		
2	1	...	0	0		
3	0	...	0	0		
4	1	...	0	0		
	OCCUPATION_House-s	OCCUPATION_Machine	OCCUPATION_Other	OCCUPATION_Prof.	\	
0	0	0	0	1		
1	0	0	0	0		
2	0	0	0	0		
3	0	0	0	0		
4	0	0	0	0		
	OCCUPATION_Protec.	OCCUPATION_Sales	OCCUPATION_TechSup	\		
0	0	0	0			
1	0	1	0			
2	0	0	0			
3	0	0	0			
4	0	1	0			
	OCCUPATION_Transp.					
0	0					
1	0					
2	0					
3	0					
4	0					

Figure 36-The dataset after transformation.

C. Data Analysis:

1. Data Conversion:

3.1. Convert variables (except comments) not included in the data preparation process of Task 2 to numbers or binary, and write a Python program to show the correlation of all variables with the target.

In this section, we outline the steps taken to convert categorical variables into numerical formats and analyze their correlation with the target variable. To begin, we assessed the dataset for missing values and identified the data types of each column. This initial examination is essential for determining which variables require conversion, as categorical variables need to be transformed into numerical formats for analysis and modeling.

Next, we utilized the LabelEncoder to convert the CUST_MARITAL_STATUS and OCCUPATION columns into numeric values. This transformation allows us to incorporate these attributes into our analyses quantitatively. Additionally, we converted the data types of YRS_RESIDENCE, AGE, BOOKKEEPING_APPLICATION, and OS_DOC_SET_KANJI to integers to ensure they are suitable for correlation analysis. Once the data was appropriately formatted, the 'COMMENTS' column was dropped from the analysis as it requires text processing for meaningful correlation analysis. After preparing the dataset, we computed the correlation matrix for all variables. After preparing the dataset, we computed the correlation matrix for all remaining variables. The correlation matrix is shown in **Figure 37. Correlation Matrix** reveals several key insights:

- **AGE and YRS_RESIDENCE:** The positive correlation (0.674) between age and years of residence suggests that older individuals tend to have lived in their residences for longer periods. This finding aligns with expectations, as life stages often correlate with housing stability.
- **'BULK_PACK_DISKETTES' and 'FLAT_PANEL_MONITOR':** The very strong positive correlation between 'BULK_PACK_DISKETTES' and 'FLAT_PANEL_MONITOR' (0.91) suggests customers often purchase these items together, possibly as part of a tech upgrade. This could indicate a segment of customers interested in a particular type of technology or a specific need that both products fulfill.
- **AGE and Y_BOX_GAMES:** The strong negative correlation (-0.710) indicates that younger individuals are more likely to purchase or engage with Y-box games, while older individuals show less interest. This finding could inform marketing strategies targeting specific age demographics for gaming products.
- **'CUST_INCOME_LEVEL' and Tech Purchases:** 'CUST_INCOME_LEVEL' shows a positive correlation with both 'BULK_PACK_DISKETTES' (0.64) and 'FLAT_PANEL_MONITOR' (0.58), indicating that higher-income customers tend to purchase more or higher-end tech items. This aligns with the general understanding that disposable income influences the purchase of electronics. Higher income may allow for more frequent upgrades or the purchase of multiple devices.

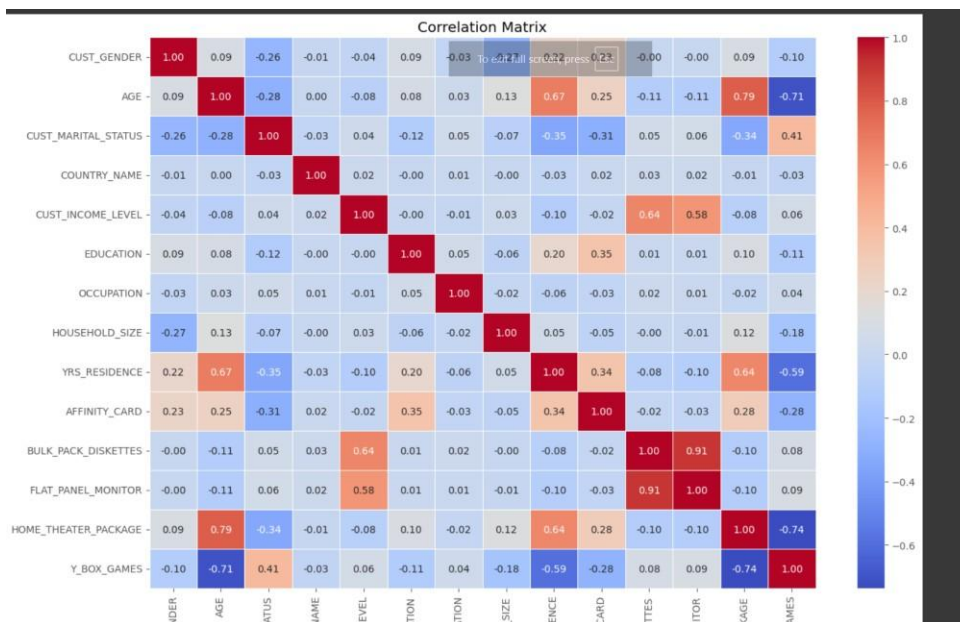


Figure 37-Correlation Matrix

2. Sentimental Analysis:

3.2. Write a Python program to analyse comments sentiment and create a sentiment column with values of 1 positive, 0 neutral, and -1 negative.

In this section, customer sentiment was expressed in the comments field of our marketing campaign dataset. Understanding customer sentiment provides valuable insights into attitudes toward our products, services, and loyalty programs, helping to identify areas of customer satisfaction and potential improvement opportunities. To quantify sentiment from unstructured text data a natural language processing approach using the TextBlob library was implemented. This Python-based sentiment analysis tool evaluates the emotional tone of text by analyzing the words used and their context. The analysis produces a polarity score for each comment, which I then classified into three distinct sentiment categories:

- Positive sentiment (1): Comments with polarity scores greater than 0.1
- Neutral sentiment (0): Comments with polarity scores between -0.1 and 0.1
- Negative sentiment (-1): Comments with polarity scores less than -0.1

For empty comments or non-text entries, I assigned a neutral sentiment score, ensuring comprehensive coverage across the dataset, as shown in **Figure 38**.

Sentimental Analysis

```
# Function to classify sentiment based on polarity
def get_sentiment(text):
    if not isinstance(text, str) or text.strip() == '':
        return '0' # Treat empty or non-string comments as neutral
    polarity = TextBlob(text).sentiment.polarity
    if polarity > 0.1:
        return '1' # Positive
    elif polarity < -0.1:
        return '-1' # Negative
    else:
        return '0' # Neutral

# Apply the sentiment function to the COMMENTS column
df_copy1['COMMENTS_SENTIMENT'] = df_copy1['COMMENTS'].apply(get_sentiment)

# Show a few examples
print(df_copy1[['COMMENTS', 'COMMENTS_SENTIMENT']].head(10))
```

	COMMENTS	COMMENTS_SENTIMENT
0	Shopping at your store is a hassle. I rarely s...	0
1	Affinity card is great. I think it is a hassle...	1
2	I purchased a new computer recently, but the m...	0
3	Affinity card is great. I think it is a hassle...	1
4	Why didn't you start a program like this befor...	0
5	Forget it. I 'm not giving you all my personal...	0
6	It is a good way to attract new shoppers. Afte...	1
7	I shop your store a lot. I love your weekly s...	1
8	Affinity card makes sense only for bulk purch...	0
9	Could you send an Affinity Card to my mother i...	0

Figure 38-Sentimental Analysis

Finally, the code prints the first 10 rows of the DataFrame, showing the original 'COMMENTS' and the newly created 'COMMENTS_SENTIMENT' to illustrate the sentiment classification. It also prints the unique values in the 'COMMENTS_SENTIMENT' column, which should be '0', '1', and '-1', representing neutral, positive, and negative sentiment, respectively, as shown in **Figure 39. Unique values in the Comment Sentiment column.**

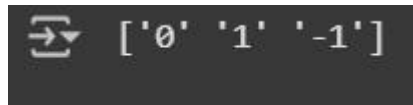


Figure 39-Unique values in the Comment Sentiment column.

D. Data Exploration:

Write a Python program that displays a histogram of one of the processed variables, allowing the user to select any one variable at run time. The program should continue running until the user chooses to exit.

This section analyzes a Python program designed to facilitate exploratory data analysis through interactive histogram visualization. This section analyzes a Python program designed to facilitate exploratory data analysis through interactive histogram visualization. The program enables users to select numeric variables from a dataset at runtime and visualize their distributions, continuing operation until the user chooses to exit. The program enables users to select numeric variables from a dataset at runtime and visualize their distributions, continuing operation until the user chooses to exit, as shown in **Figure 40. Interactive Histogram Visualization.**

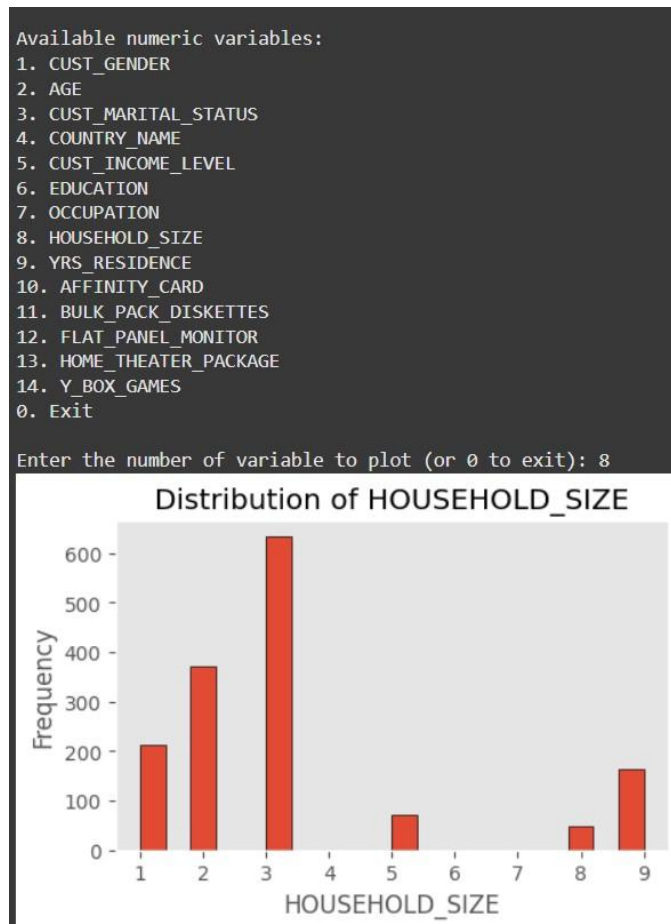


Figure 40-Interactive Histogram Visualization

After displaying the histogram, the program returns to the prompt, again showing the list of available numeric variables and asking the user for their choice. This loop continues indefinitely, allowing the user to explore the distributions of different numeric variables without restarting the program. The user can exit the interactive session at any time by entering '0' when prompted for their choice. Upon entering '0', the program prints an "Exiting histogram viewer..." message and terminates, as shown in **Figure 41. Exiting histogram Viewer.**

```
Histogram for 'HOUSEHOLD_SIZE' displayed.

Available numeric variables:
1. CUST_GENDER
2. AGE
3. CUST_MARITAL_STATUS
4. COUNTRY_NAME
5. CUST_INCOME_LEVEL
6. EDUCATION
7. OCCUPATION
8. HOUSEHOLD_SIZE
9. YRS_RESIDENCE
10. AFFINITY_CARD
11. BULK_PACK_DISKETTES
12. FLAT_PANEL_MONITOR
13. HOME_THEATER_PACKAGE
14. Y_BOX_GAMES
0. Exit

Enter the number of variable to plot (or 0 to exit): 0
Exiting histogram viewer...
```

Figure 41-Exiting histogram Viewer.

E. Data Mining

Before building the logistic regression model, a critical data preparation phase was conducted to evaluate which features should be included in the predictive modeling process. To begin, a copy of the DataFrame `df_copy1` was created, as shown in **Figure 42.df_copy1 column list**, resulting in `df_copy2`. This step ensured that the original dataset remained unaltered while allowing for modifications and analyses on the copy.

```
[ ] print(df_copy1.columns)

Index(['CUST_GENDER', 'AGE', 'CUST_MARITAL_STATUS', 'COUNTRY_NAME',
      'CUST_INCOME_LEVEL', 'EDUCATION', 'OCCUPATION', 'HOUSEHOLD_SIZE',
      'YRS_RESIDENCE', 'AFFINITY_CARD', 'BULK_PACK_DISKETTES',
      'FLAT_PANEL_MONITOR', 'HOME_THEATER_PACKAGE', 'Y_BOX_GAMES', 'COMMENTS',
      'COMMENTS_SENTIMENT'],
      dtype='object')
```

Figure 42- df_copy1 column list

Next, one-hot encoding was applied to the `COMMENTS_SENTIMENT` column. This process transformed the categorical sentiment classifications—positive(1), neutral(0), and negative(-1) into binary format. Each sentiment category was represented as a separate column, with binary values indicating the presence or absence of that sentiment in each comment, as shown in **Figure 43. One Hot Encoding for Sentiment Column.**

```
[223] # One-hot encoding
      one_hot_encoded = pd.get_dummies(df_copy2['COMMENTS_SENTIMENT'], prefix='sentiment')

      # Combine the one-hot encoded columns with the original DataFrame (excluding the or
      df_copy2 = pd.concat([df_copy2.drop('COMMENTS_SENTIMENT', axis=1), one_hot_encoded])

      print(df_copy2.head())

Show hidden output

      sentiment_columns = [col for col in df_copy2.columns if col.startswith('sentiment')]
      print(sentiment_columns)

['sentiment_-1', 'sentiment_0', 'sentiment_1']
```

Figure 43-One Hot Encoding for Sentiment Column.

After creating the one-hot encoded variables, these new columns were concatenated back into `df_copy2`, while the original `COMMENTS_SENTIMENT` column was dropped to avoid redundancy. This step resulted in a DataFrame that

incorporated the new binary sentiment indicators alongside the existing features. Following this, a correlation matrix was computed for the numeric columns in `df_copy2`, excluding the original `COMMENTS` column. The correlation matrix quantifies the relationships between the new sentiment variables and other numeric features, providing insights into how sentiment might influence or relate to other variables, as shown in **Figure 44. Correlation Matrix**.

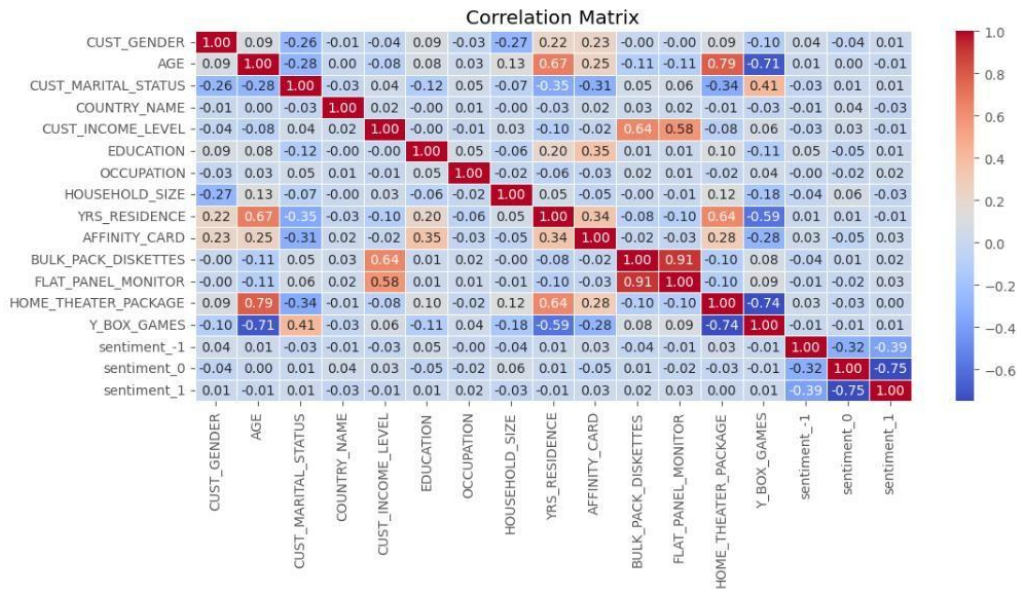


Figure 44-Correlation Matrix with sentiments

Upon reviewing the correlation results, it became evident that the sentiment variables exhibited minimal correlation with the other features. This lack of significant relationships suggested that the sentiment data would not provide valuable information for model training. As a result, the decision was made to exclude both the `COMMENTS` and `COMMENTS_SENTIMENT` columns from further analysis. Finally, the `df_copy1` DataFrame was updated by dropping these columns, ensuring that the dataset was refined and focused on the most informative features for subsequent modeling efforts.

1. Training Logistic Regression model

5.1.Keep the first 100 customer records from processed campaign data for testing and keep the rest to build a logistic regression ML model using the top 10 relevant independent variables with Python program. Find intercept and coefficients for each independent variable.

In this analysis, the dataset was divided to assess the relationship between various customer characteristics and the likelihood of having an affinity card. The first 100 customer records were allocated for testing, while the remaining records were utilized to train a logistic regression model aimed at predicting the target variable, `AFFINITY_CARD`. The training set was prepared by excluding the target variable from the feature set. Subsequently, mutual

information scores were calculated for each feature to evaluate their relevance in predicting the target outcome. The top 10 features, determined to be the most informative based on these scores, were selected for inclusion in the model, as shown in **Figure 45. Top 10 Features.**

```
Top 10 features: ['CUST_MARITAL_STATUS', 'HOUSEHOLD_SIZE', 'YRS_RESIDENCE', 'AGE', 'EDUCATION', 'OCCUPATION', 'HOME_THEATER_PACKAGE', 'CUST_GENDER', 'Y_BOX_GAMES', 'COUNTRY_NAME']
```

Figure 45-Top 10 Features.

The selected features were standardized to ensure they had a mean of 0 and a standard deviation of 1, which is crucial for the convergence of the logistic regression algorithm. The model was then fitted using these scaled features alongside the target variable.

```
Intercept:
-1.681012513236454

Feature Coefficients:
CUST_MARITAL_STATUS -0.6553
HOUSEHOLD_SIZE      -0.1305
YRS_RESIDENCE       +0.3642
AGE                 -0.0341
EDUCATION           +0.8580
OCCUPATION          -0.1054
HOME_THEATER_PACKAGE +0.1954
CUST_GENDER         +0.3992
Y_BOX_GAMES         -0.3014
COUNTRY_NAME        +0.0202
```

Figure 46- Intercept and Coefficient of the model

The output shown in the given **Figure 46. Top 10 Features.** shows the following insights:

- **Intercept:** The intercept was about -1.681. This means that, without any influence from the features, the baseline likelihood of having an affinity card is low.
- **Feature Coefficients:** The coefficients from the logistic regression model reveal important insights into the factors influencing affinity card ownership. **CUST_MARITAL_STATUS** has a coefficient of -0.6553, indicating that married individuals may be less likely to have an affinity card. **HOUSEHOLD_SIZE** shows a coefficient of -0.1305, suggesting that larger households tend to use the card less. In contrast, **YRS_RESIDENCE** has a positive coefficient of +0.3642, meaning that a longer duration of residence increases the likelihood of card ownership. **AGE** slightly decreases the likelihood with a coefficient of 0.0341, while **EDUCATION** significantly boosts it, with a coefficient of +0.8580. The coefficient for **OCCUPATION** is -0.1054, indicating that certain jobs may correlate with lower card usage. Conversely, having a **HOME_THEATER_PACKAGE** increases the likelihood of ownership (+0.1954), as does being a particular

CUST_GENDER (+0.3992). However, interest in **Y_BOX_GAMES** has a negative impact (-0.3014), suggesting that engagement with this product may lower the likelihood of card ownership. Lastly, the **COUNTRY_NAME** variable has a minimal positive effect with a coefficient of +0.0202, indicating slight geographic variations in card ownership. Overall, education and residence duration emerge as strong positive predictors, while marital status and certain product interests may negatively influence ownership

2. Testing Logistic Regression model

5.2. Test the accuracy of the model using the first 100 customer records and explain the results with graphs.

In this section, the performance of our logistic regression model is assessed using the first 100 customer records as the test set. The evaluation includes accuracy, a ROC curve, a confusion matrix, and a detailed classification report. The test set comprised the first 100 records from the dataset, using the top 10 features identified during model training. These features were scaled to ensure consistency with the training data.

Results:

- **Accuracy(0.78):** This means that the model correctly predicted the 'AFFINITY_CARD' status (high-value or low-value) for 78% of the 100 customers in the test set. While this indicates a reasonable level of accuracy, it also means that the model made incorrect predictions for 22% of the test customers, as shown in **Figure 47. Accuracy Score**.

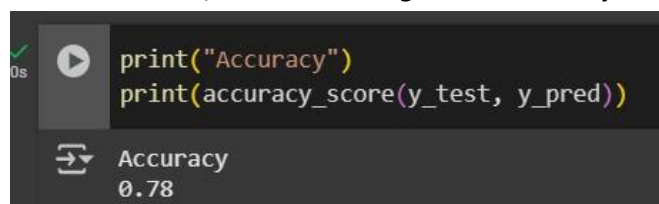


Figure 47-Accuracy Score

- **ROC Curve & AUC (AUC = 0.813)**—The AUC score represents the area under the ROC curve. It provides a single numerical value that summarizes the overall performance of the model in distinguishing between the two classes, as shown in **Figure 48. ROC Curve**. An AUC of 0.5 suggests no discrimination ability (equivalent to random guessing), while an AUC of 1.0 indicates perfect discrimination. In this case, an AUC of 0.813 suggests that the model has a good ability to distinguish between high-value and low-value customers. There's an 81.3% chance that the model will rank a

randomly chosen high-value customer higher than a randomly chosen low-value customer.

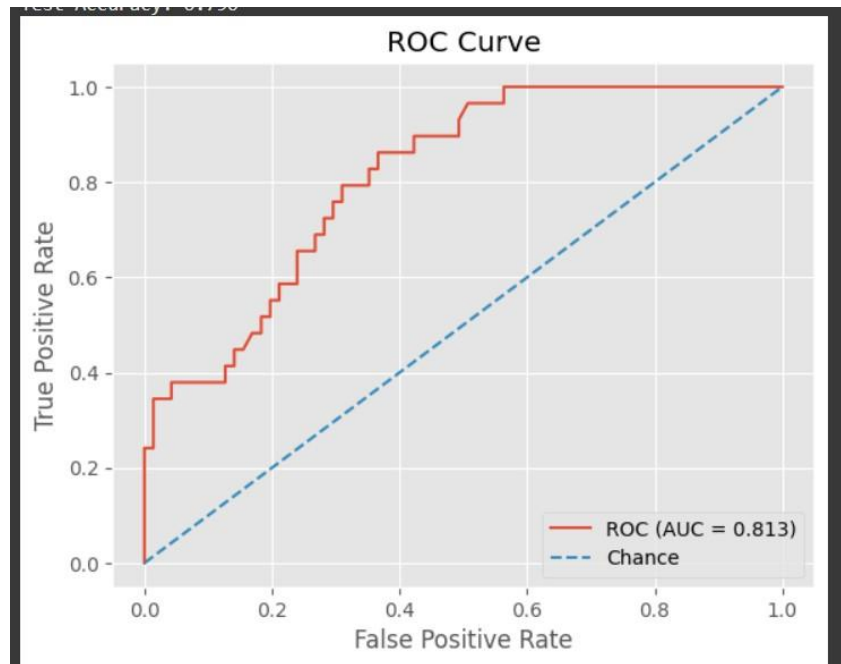


Figure 48-ROC Curve.

- **Confusion Matrix:** The confusion matrix provides a more detailed look at the model's predictions. The model is better at correctly identifying customers who do not have an affinity card (high TN) compared to those who do (lower TP). There are fewer False Positives (4) than False Negatives (18), as shown in **Figure 49. Confusion Matrix**. This means the model is more likely to miss a high-value customer than to incorrectly classify a low-value customer as high-value.

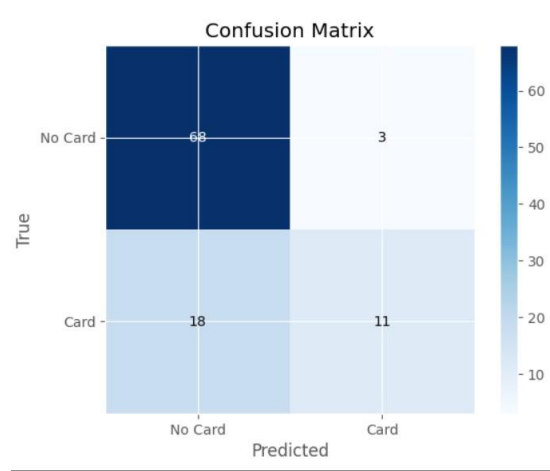


Figure 49-Confusion Matrix

- **Classification Report:** The classification report provides precision, recall, and F1-score for each class (0: No Card, 1: Card), as shown in **Figure 50**.
Classification Report .Precision: For class 0 (No Card), precision is 0.79, meaning that when the model predicted a customer did not have an affinity card, it was correct 79% of the time ($67 / (67 + 17)$). For class 1 (Card), precision is 0.73, meaning that when the model predicted a customer had an affinity card, it was correct 73% of the time ($11 / (11 + 4)$).**Recall (Sensitivity):** For class 0, recall is 0.94, meaning the model correctly identified 94% of all customers who did not have an affinity card ($67 / (67 + 4)$). For class 1, recall is 0.38, meaning the model correctly identified only 38% of all customers who actually had an affinity card ($11 / (11 + 18)$). This low recall for class 1 indicates that the model is missing a significant portion of the highvalue customers.**F1-score:** This is the harmonic mean of precision and recall. For class 0, it's 0.86, and for class 1, it's 0.50. The lower F1score for class 1 reflects the imbalance between precision and recall for this class, primarily due to the low recall.**Support:** This indicates the number of actual occurrences of each class in the test set (71 for No Card, 29 for Card).

Classification Report:					
	precision	recall	f1-score	support	
0	0.79	0.94	0.86	71	
1	0.73	0.38	0.50	29	
accuracy			0.78	100	
macro avg	0.76	0.66	0.68	100	
weighted avg	0.77	0.78	0.75	100	

Figure 50-Classification Report

Overall, the results suggest that while the model performs well, there is room for improvement in capturing all customer segments effectively.

3. Predictive Application

5.3.Implement a predictive application based on the created logistic regression ML model. The application should have an appropriate user interface to allow users to input customer records from keyboard and file to receive predicted answers.

The objective of the predictive application is to create a user-friendly interface that allows users to input customer records either manually or by uploading a CSV file. This application uses a trained logistic regression model to predict

whether a customer is likely to have an affinity card, providing valuable insights for businesses.

The code implementation begins with the necessary imports and setup, utilizing libraries such as ipywidgets for the user interface, pandas for data manipulation, and numpy for numerical operations. The core of the application is encapsulated in the CustomerPredictor class, which is initialized with the logistic regression model, scaler, top features, and mappings for categorical variables, as shown in **Figure 51. Customer Predictor Function**

```
class CustomerPredictor:
    def __init__(self, model, scaler, top_features,
                 country_mapping, income_mapping, education_mapping,
                 occupation_categories, marital_categories):
        self.model = model
        self.scaler = scaler
        self.top_features = top_features
        self.country_mapping = country_mapping
        self.income_mapping = income_mapping
        self.education_mapping = education_mapping
        self.occupation_categories = occupation_categories
        # Fix: Remove .keys() - use the list directly
        self.marital_encoder = LabelEncoder().fit(marital_categories)
        self.create_ui()
```

Figure 51- Customer Predictor Function

A key feature of the application is its user interface, created through the create_ui method. This method defines input widgets for various customer characteristics, including marital status, household size, age, education, occupation, home theater package, gender, Y-box games, and country. Additionally, the application includes buttons for predicting customer status from manual input and for uploading a CSV file containing multiple customer records, as shown in **Figure 52. Buttons for manual input , .csv file prediction**

```
# Create buttons and output
self.predict_button = widgets.Button(description="Predict from Input")
self.upload_button = widgets.Button(description="Predict from CSV")
self.file_upload = widgets.FileUpload(accept='.csv')
self.output = widgets.Output()
```

Figure 52-. Buttons for manual input , .csv file prediction

Input preprocessing is handled by the preprocess_input method, which converts categorical features into numerical formats and scales the data using the provided scaler. This ensures that the input data is formatted correctly for the model, which is essential for accurate predictions. The application also includes methods for making predictions: predict_from_input collects data from the widgets, processes it, and uses the model to generate predictions, while predict_from_file processes uploaded CSV files for batch predictions.

Example Output is shown in **Figure 53. Example Output**

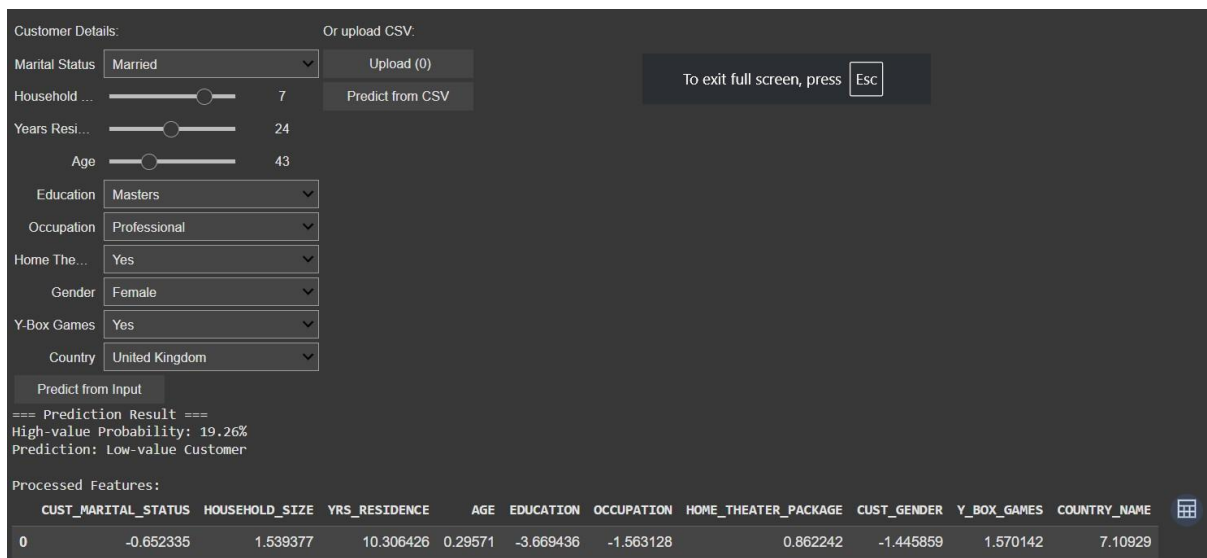


Figure 53-Example Output

High-value Probability: 19.26%- This is the prediction that has been done. The percentage represents the likelihood that the customer is a highvalue customer based on the model's calculations. Here, the probability of 19.26% indicates a relatively low chance of the customer being classified as high-value. This could suggest that the features associated with this customer do not align strongly with those of previous high-value customers in the training data.

Prediction: Low-value Customer- This is the model's classification of the customer based on the calculated probability. The model predicts that this customer is a low-value customer, meaning they are less likely to generate significant revenue or engagement.

Processed Features: The processed features provide detailed insights into how each customer characteristic influenced the prediction.

1. **High Value:** The variables **HOUSEHOLD_SIZE (1.53)** and **YRS_RESIDENCE (10.30)** indicate a higher likelihood of being classified as a high-value customer. A larger household size suggests that families may spend more collectively, while longer years of residence often correlate with increased brand loyalty and spending. Additionally, **AGE (0.30)** shows a slight positive effect, indicating that older customers may have more disposable income, which can increase their likelihood of being classified as high-value. Furthermore, **EDUCATION (0.30)** suggests that higher education levels correlate positively with high-value status, as educated individuals often have greater earning potential. **COUNTRY_NAME (7.11)** also plays a significant role, indicating that customers from the United Kingdom are more likely

to be classified as high-value, possibly due to regional economic conditions or cultural spending habits.

2. Low Value: The variable **CUST_MARITAL_STATUS (-0.65)** indicates a lower likelihood of being high-value if the customer is married. This might reflect spending patterns where married individuals prioritize family needs over discretionary spending. Similarly, **OCCUPATION (-3.67)** suggests that having the occupation "Professional" may reduce the likelihood of being classified as high-value, potentially due to lower disposable income in this category. The variables **HOME_THEATER_PACKAGE (-1.56)** and **Y_BOX_GAMES (-1.44)** also have negative values, indicating that having a home theater package or an interest in Y-box games may detract from high-value classification, as these interests might reflect spending priorities that do not align with high-value customer traits. Finally, **CUST_GENDER (0.86)** has a positive influence, suggesting that one gender may be more likely to be classified as high-value compared to the other, which could reflect different spending behaviors.

V. Discussion and reflection of the work

A. Discussion:

The project revealed several interesting patterns about what makes a customer "high-value." For example, customers with higher education levels and those who'd lived in their homes longer were more likely to own an affinity card as people who stay in one place might be more settled and loyal to brands, while education could correlate with higher income or interest in loyalty benefits. Surprisingly, even though many customers had high incomes, being married *reduced* the chance of being a high-value customer. Maybe families prioritize spending on essentials rather than optional perks like affinity cards.

The model did a decent job overall, correctly predicting customer value 78% of the time. But it struggled to spot high-value customers specifically—like only catching 38% of them. This happened because the dataset had far more low-value customers (75%) than high-value ones (25%), making it harder for the model to learn what makes the smaller group unique. The sentiment analysis of customer comments didn't add much to the predictions. Most comments were neutral or mixed, like *"The affinity card is great, but it's a hassle to bring it every time."* This taught me that text data needs deeper analysis, like checking for specific keywords, to truly understand customer feelings.

B. Reflection:

This project emphasized the importance of methodical data preparation and the realities of working with imperfect datasets. Challenges like inconsistent entries (e.g., "9+" in household size) demanded careful standardization to ensure analytical rigor. Converting categorical variables into numerical formats also required balancing simplicity with retaining meaningful patterns. The model's performance revealed both strengths and limitations. While logistic regression provided interpretable coefficients (e.g., education's strong positive influence), its struggles with class imbalance. Future iterations could benefit from techniques like SMOTE to address imbalance or ensemble methods to capture non-linear relationships.

This work helps set the stage for using data to improve marketing. By looking at things like education, where people live, and how they use tech products, companies can get better at targeting the right customers. But, to really make a difference with marketing campaigns, it's important for businesses to work with others, get their opinions, and make sure the findings make sense in the real world

VI. Conclusion

This project shows how analyzing data can give us useful information to make marketing better. By looking at customer details, what they buy, and what they say, the model found that valuable customers often have higher education, have lived in the same place for a while, and like tech products. The model was fairly accurate (78%), but it had trouble spotting high-value customers. This means we need to find better ways to deal with unbalanced data in the future, like using oversampling or different algorithms.

The results can help create more focused marketing, like offering special deals to educated, long-term residents or combining tech products with loyalty perks. However, there were some problems, like messy data and not getting much useful information from customer opinions. To fix this, we could clean up the data better and find better ways to analyze text.

Overall, this work connects data with making smart business decisions. It gives us a plan to improve customer groups and loyalty programs. By using these insights and working with marketing teams, companies can turn data into real growth.