Experiment 3

Aim: To implement data cleaning techniques (e.g. Data Imputation).

Theory:

1. **What is Data Cleaning?**

Data cleaning is the essential practice of refining raw information by identifying and resolving inaccuracies to prepare it for meaningful use. As a critical phase of data preprocessing, it ensures that datasets are primed for high-quality analytical, statistical, and machine learning applications.

- **Ensures Accuracy:** Removes duplicate entries, fills missing gaps, and fixes inconsistencies that might otherwise lead to skewed results.
- **Empowers Decision-Making:** Provides organizations with a reliable foundation of current and precise data for strategic planning.
- **Boosts Efficiency:** Minimizes the time and resources typically wasted on troubleshooting errors during the reporting and analysis phases.
- **Optimizes Model Performance:** Enhances the dependability of insights derived from AI, analytics, and machine learning models.
- **Facilitates Compliance:** Promotes alignment with established data quality benchmarks and legal regulatory mandates.

Fig 1: Different types of data Cleaning

2. **What are the different types of Data Anomalies?**

Data quality issues typically stem from human error, system malfunctions, or friction during data integration. If left unaddressed, these anomalies can compromise the integrity of your analysis and lead to flawed conclusions.

Types of Data Challenges:

- **Missing Values:** Incomplete data points that can diminish statistical power and introduce significant bias into your results.
- **Duplicate Records:** Redundant entries that artificially inflate certain observations, leading to skewed or weighted outcomes.
- **Incorrect Data Types:** Formatting mismatches—such as storing strings in numeric fields—which trigger calculation failures and break analytical pipelines.
- **Outliers and Anomalies:** Extreme data points (unusually high or low) that can distort statistical averages and negatively impact model training.
- **Inconsistent Formats:** Discrepancies in date structures, text casing, or units of measurement that complicate data merging and comparative studies.
- **Typographical Errors:** Spelling mistakes in text fields that cause errors in grouping, classification, and the interpretation of categorical variables.

3. **Data Cleaning Process:**

Data cleaning is the systematic process of identifying and fixing errors, inconsistencies, and inaccuracies within a raw dataset to prepare it for analysis. It involves essential steps such as removing duplicates, handling missing values, and standardizing formats to ensure data integrity. By refining the data, this process prevents skewed results and ensures that analytical models and business decisions are based on high-quality, reliable information. Process is as follows:

1. **Assess Data Quality**

The initial phase involves a thorough audit of the dataset to identify underlying flaws. This diagnostic step focuses on:

- Missing Values: Spotting null or blank entries caused by collection gaps, entry mistakes, or transmission losses.
- Incorrect Values: Detecting data points that fall outside logical ranges or conflict with the assigned data type.
- Format Inconsistencies: Ensuring that data follows a uniform structure throughout the entire set.

2. **Remove Irrelevant Data**

Pruning unnecessary information ensures the dataset remains lean and meaningful, preventing skewed results and improving processing speed.

- Identify Duplicates: Locate repetitive entries using methods like hashing, grouping, or sorting.
- Eliminate Redundancy: Remove identical records so every data point is unique.
- Filter Observations: Detect and discard observations that offer no new information.
- Drop Irrelevant Variables: Exclude columns or attributes that do not contribute to the specific goals of the analysis.

3. **Fix Structural Errors**

Structural issues arise from inconsistent naming conventions or mismatched types. Correcting these ensures the data is represented reliably and uniformly.

- Standardize Formats: Align dates, timestamps, and strings to a single, consistent standard.
- Correct Naming: Resolve discrepancies in column headers and labels for better clarity.
- Uniform Representation: Ensure all measurements use the same units (e.g., metric vs. imperial) and scales.

4. **Handle Missing Data**

Addressing gaps is vital to prevent bias and maintain the statistical integrity of your project.

- Statistical Imputation: Fill gaps using basic measures like the mean, median, or mode.
- Record Deletion: Remove rows with missing data if the gaps are too extensive to fix reliably.
- Advanced Modeling: Use sophisticated algorithms like K-Nearest Neighbors (KNN) or regression to predict and insert missing values.

5. **Normalize Data**

Normalization organizes the dataset to minimize redundancy, making it more efficient for querying and long-term management.

- Data Partitioning: Segment data into specialized tables to isolate specific information types.
- Relational Consistency: Maintain logical links across the dataset to support accurate, high-speed analysis.

6. **Identify and Manage Outliers**

Outliers are extreme deviations that can disproportionately influence your results. Managing them ensures your insights reflect the majority of the population.

- Outlier Removal: Discard extreme values that are clearly the result of measurement or entry errors.

Dataset Overview:

- This dataset is made up of 23 columns and 10,127 rows.
- The BankChurners dataset contains detailed information about credit card customers and their account activity, focusing on understanding customer retention and churn behavior within a financial institution. Its main objective is to analyze factors that influence whether customers continue using their credit card services or stop their relationship with the bank.
- It includes identification attributes such as **CLIENTNUM,** which uniquely identifies each customer record, and the **Attrition_Flag**, which indicates whether a customer has remained active or has churned. These fields help track customer status and behavior over time.
- Demographic characteristics include variables such as **Customer_Age, Gender, Dependent_count, Education_Level, Marital_Status, and Income_Category**, which provide insights into the personal and socioeconomic profiles of customers. These attributes help identify demographic patterns related to customer retention or attrition.
- Account relationship indicators such as **Card_Category, Months_on_book, and Total_Relationship_Count** describe the type of credit card owned, the duration of the customer's relationship with the bank, and the number of financial products held by the customer. These features help evaluate the depth and longevity of the customer–bank relationship.
- Customer activity and engagement metrics include Months_Inactive_12_mon and Contacts_Count_12_mon, which measure inactivity periods and the frequency of customer interactions with the bank. These operational indicators are important for detecting early warning signs of potential churn.
- Financial and transaction variables such as Credit_Limit, Total_Revolving_Bal, Avg_Open_To_Buy, Total_Trans_Amt, Total_Trans_Ct, and Avg_Utilization_Ratio capture spending behavior, credit usage, and transaction activity. These metrics provide insights into how actively customers use their credit cards and their financial engagement with the bank.
- Finally, change indicators like Total_Amt_Chng_Q4_Q1 and Total_Ct_Chng_Q4_Q1 measure shifts in spending and transaction frequency over time, helping analysts detect behavioral trends that may signal increasing engagement or potential churn risk.

Implementation

EM Algorithm:

```
import pandas as pd
import numpy as np

df = pd.read_csv("BankChurners.csv")

id_cols = []
if "CLIENTNUM" in df.columns:
    id_cols = ["CLIENTNUM"]

df_work = df.drop(columns=id_cols).copy()
numeric_cols = df_work.select_dtypes(include=[np.number]).columns.tolist()
categorical_cols = df_work.select_dtypes(include=["object"]).columns.tolist()

df_numeric = df_work[numeric_cols].copy()
df_categorical = df_work[categorical_cols].copy()

def em_imputation(data, max_iter=15, tol=1e-3):
    mask = np.isnan(data)

    data_imputed = np.where(mask, np.nanmean(data, axis=0), data)

    for _ in range(max_iter):
        prev_data = data_imputed.copy()

        mu = np.mean(data_imputed, axis=0)
                    sigma  =  np.cov(data_imputed,  rowvar=False)  +  1e-6  *
np.eye(data.shape[1])

        for i in range(data.shape[0]):
            if np.any(mask[i]):
                m = mask[i]
                o = ~mask[i]

                if np.sum(o) == 0:
                    continue

                sigma_oo = sigma[np.ix_(o, o)]
                sigma_mo = sigma[np.ix_(m, o)]

                data_imputed[i, m] = (
                    mu[m] +
                     sigma_mo @ np.linalg.solve(sigma_oo, data_imputed[i, o] -
mu[o])
                )

        if np.linalg.norm(data_imputed - prev_data) < tol:
            break

    return data_imputed

if len(numeric_cols) > 0:
    imputed_numeric = em_imputation(df_numeric.values.astype(float))
    df_numeric_imputed = pd.DataFrame(imputed_numeric, columns=numeric_cols)
```

```
else:
    df_numeric_imputed = pd.DataFrame()

df_categorical_imputed = df_categorical.copy()

for col in categorical_cols:
    mode_value = df_categorical_imputed[col].mode(dropna=True)
    if len(mode_value) > 0:
                                            df_categorical_imputed[col]     =
df_categorical_imputed[col].fillna(mode_value[0])

df_final = pd.concat([df_numeric_imputed, df_categorical_imputed], axis=1)

for col in id_cols:
    df_final[col] = df[col]

df_final = df_final[df.columns]

print("Missing values BEFORE imputation:\n")
print(df.isnull().sum())
print("\nMissing values AFTER imputation:\n")
print(df_final.isnull().sum())

df_final.to_csv("BankChurners_EM_Imputed.csv", index=False)
print("\nImputation Complete. File saved as 'BankChurners_EM_Imputed.csv'")
print("\n===== IMPUTATION SUMMARY =====\n")

for col in df_work.columns:

    missing_mask = df_work[col].isnull()
    n_missing = missing_mask.sum()

    if n_missing > 0:

        print(f"Column: {col}")
        print(f"Number of missing values: {n_missing}")

        filled_values = df_final.loc[missing_mask, col]

        if col in numeric_cols:
            print("Type: Numeric (EM Imputation)")
            print("Summary of filled values:")
            print(f"  Mean of filled values: {filled_values.mean():.4f}")
            print(f"  Min filled value: {filled_values.min():.4f}")
            print(f"  Max filled value: {filled_values.max():.4f}")

        elif col in categorical_cols:
            print("Type: Categorical (Mode Imputation)")
            print(f"Filled with mode value: {filled_values.iloc[0]}")

        print("-" * 40)
```

```
PS C:\Users\PC-22.421-PC-22\Desktop\MJ10> python exp4.py
Missing values BEFORE imputation:

CLIENTNUM                                                                                                  0
Attrition_Flag                                                                                             0
Customer_Age                                                                                               0
Gender                                                                                                     0
Dependent_count                                                                                            0
Education_Level                                                                                         1519
Marital_Status                                                                                           749
Income_Category                                                                                         1112
Card_Category                                                                                              0
Months_on_book                                                                                             0
Total_Relationship_Count                                                                                   0
Months_Inactive_12_mon                                                                                     0
Contacts_Count_12_mon                                                                                      0
Credit_Limit                                                                                               0
Total_Amt_Chng_Q4_Q1                                                                                       0
Total_Trans_Amt                                                                                            0
Total_Trans_Ct                                                                                             0
Total_Ct_Chng_Q4_Q1                                                                                        0
Avg_Utilization_Ratio                                                                                      0
Naive_Bayes_Classifier_Attrition_Flag_Card_Category_Contacts_Count_12_mon_Dependent_count_Education_Level_Months_Inactive_12_mon_1      0
Naive_Bayes_Classifier_Attrition_Flag_Card_Category_Contacts_Count_12_mon_Dependent_count_Education_Level_Months_Inactive_12_mon_2      0
dtype: int64
```

Fig 3.1: original dataset with missing values

```
Missing values AFTER imputation:

CLIENTNUM                                                                                                  0
Attrition_Flag                                                                                             0
Customer_Age                                                                                               0
Gender                                                                                                     0
Dependent_count                                                                                            0
Education_Level                                                                                            0
Marital_Status                                                                                             0
Income_Category                                                                                            0
Card_Category                                                                                              0
Months_on_book                                                                                             0
Total_Relationship_Count                                                                                   0
Months_Inactive_12_mon                                                                                     0
Contacts_Count_12_mon                                                                                      0
Credit_Limit                                                                                               0
Total_Revolving_Bal                                                                                        0
Avg_Open_To_Buy                                                                                            0
Total_Amt_Chng_Q4_Q1                                                                                       0
Total_Trans_Amt                                                                                            0
Total_Trans_Ct                                                                                             0
Total_Ct_Chng_Q4_Q1                                                                                        0
Avg_Utilization_Ratio                                                                                      0
Naive_Bayes_Classifier_Attrition_Flag_Card_Category_Contacts_Count_12_mon_Dependent_count_Education_Level_Months_Inactive_12_mon_1      0
Naive_Bayes_Classifier_Attrition_Flag_Card_Category_Contacts_Count_12_mon_Dependent_count_Education_Level_Months_Inactive_12_mon_2      0
dtype: int64

Imputation Complete. File saved as 'BankChurners_EM_Imputed.csv'
```

Fig 3.2: Filled Dataset using EM Algorithm

Measures of central tendency:

```
import pandas as pd
import numpy as np

df = pd.read_csv("BankChurners_cleaned.csv")

print("--- 1. Original Dataset (First 7 Rows) ---")
print(df.head(7))

num_cols = df.select_dtypes(include=[np.number]).columns.tolist()
cat_cols = df.select_dtypes(exclude=[np.number]).columns.tolist()

if "CLIENTNUM" in num_cols:
    num_cols.remove("CLIENTNUM")
stats_dict = {}

for col in num_cols:
    stats_dict[col] = {
        "Mean": df[col].mean(),
        "Median": df[col].median(),
        "Mode": df[col].mode()[0],
        "Variance": df[col].var(),
        "SD": df[col].std(),
        "Range": df[col].max() - df[col].min()
    }

print("\n--- 2. Statistical Measures (Numerical Columns) ---")
print(pd.DataFrame(stats_dict).round(2))

imputation_values = {}

for col in num_cols:
    imputation_values[col] = df[col].median()

for col in cat_cols:
    imputation_values[col] = df[col].mode()[0]

print("\n--- 3. Values Used for Imputation (Central Tendency) ---")
for col, val in imputation_values.items():
    print(f"{col}: {val}")

df_filled = df.copy()

missing_mask = df.isna()

for col in df.columns:
    if col in imputation_values:
        df_filled[col] = df_filled[col].fillna(imputation_values[col])
```

```
df_highlighted = df_filled.astype(str)

for col in df.columns:
    if col in missing_mask.columns:
        df_highlighted.loc[missing_mask[col], col] = (
            df_highlighted.loc[missing_mask[col], col] + " (*)"
        )

print("\n--- 4. Final Dataset (Filled values marked with '*') ---")
print(df_highlighted.head(7))

df_filled.to_csv("BankChurners_filled.csv", index=False)

print("\nDataset saved as: BankChurners_filled.csv")
```

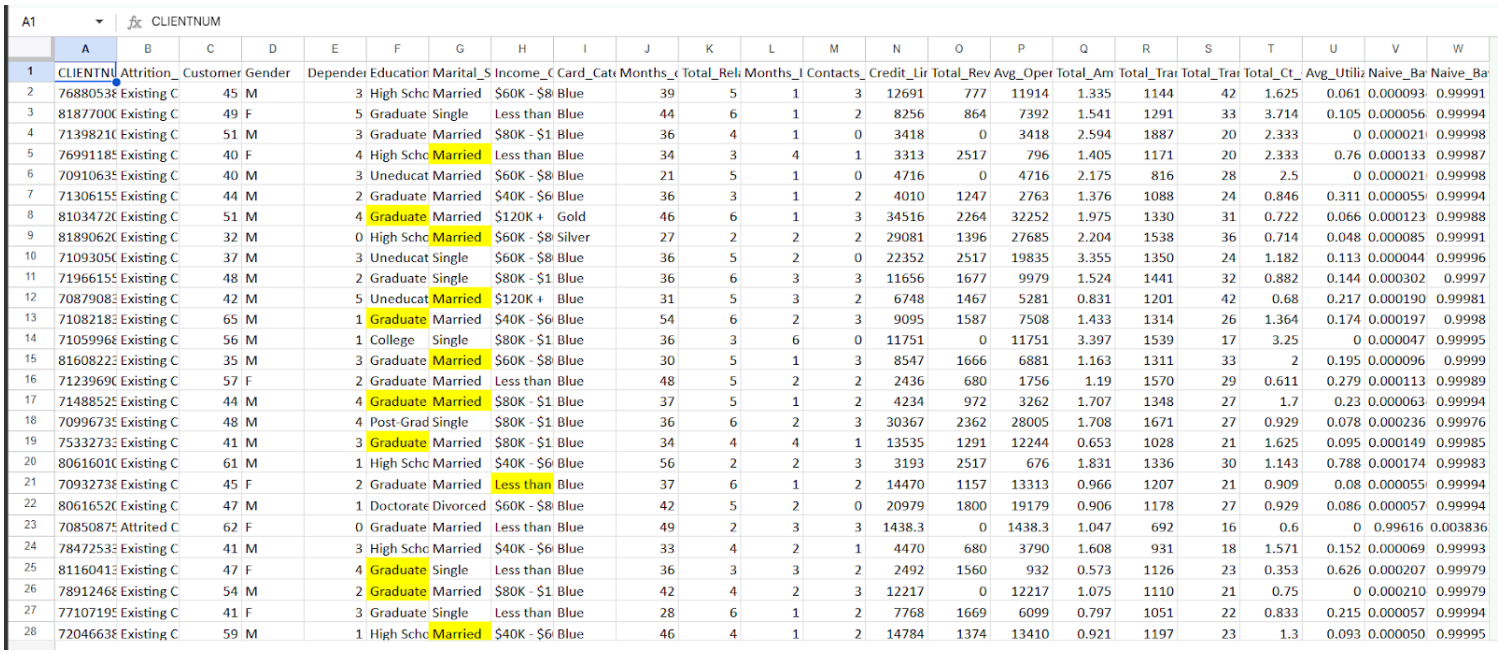
```
PS C:\Users\PC-22.421-PC-22\Desktop\MJ10> & C:/Users/PC-22.421-PC-22/AppData/Local/Programs/Python/Python310/python.exe c:/Users/PC-22.421-PC-22/Desktop/MJ10/exp4.py
--- 1. Original Dataset (First 7 Rows) ---
   CLIENTNUM  ... Naive_Bayes_Classifier_Attrition_Flag_Card_Category_Contacts_Count_12_mon_Dependent_count_Education_Level_Months_Inactive_12_mon_2
0  768805383  ...                                            0.99991
1  818770008  ...                                            0.99994
2  713982108  ...                                            0.99998
3  769911858  ...                                            0.99987
4  709106358  ...                                            0.99998
5  713061558  ...                                            0.99994
6  810347208  ...                                            0.99988

[7 rows x 23 columns]

--- 2. Statistical Measures (Numerical Columns) ---
        Customer_Age  ...  Naive_Bayes_Classifier_Attrition_Flag_Card_Category_Contacts_Count_12_mon_Dependent_count_Education_Level_Months_Inactive_12_mon_2
Mean           46.33  ...                                               0.84
Median         46.00  ...                                               1.00
Mode           44.00  ...                                               1.00
Naive_Bayes_Classifier_Attrition_Flag_Card_Category_Contacts_Count_12_mon_Dependent_count_Education_Level_Months_Inactive_12_mon_2: 0.99982
Attrition_Flag: Existing Customer
Gender: F
Education_Level: Graduate
Marital_Status: Married
Income_Category: Less than $40K
Card_Category: Blue

--- 4. Final Dataset (Filled values marked with '*') ---
   CLIENTNUM  ... Naive_Bayes_Classifier_Attrition_Flag_Card_Category_Contacts_Count_12_mon_Dependent_count_Education_Level_Months_Inactive_12_mon_2
0  768805383  ...                                            0.99991
1  818770008  ...                                            0.99994
2  713982108  ...                                            0.99998
3  769911858  ...                                            0.99987
4  709106358  ...                                            0.99998
5  713061558  ...                                            0.99994
6  810347208  ...                                            0.99988
```

Fig 6: Dataset filled using Central Tendency

Insights:

The output demonstrates a systematic data preprocessing workflow aimed at addressing missing values within the BankChurners dataset using central tendency imputation techniques. The approach first computes key statistical measures—Mean, Median, Mode, Variance, Standard Deviation, and Range—for all numerical variables to understand the distribution of customer financial attributes. To ensure robust imputation while minimizing the influence of potential outliers, the Median is selected for numerical features such as customer age and transaction-related variables, as it better represents the central location of skewed financial data.

For categorical attributes—including Attrition Flag, Gender, Education Level, Marital Status, Income Category, and Card Category—the Mode is used, reflecting the most frequently occurring category in the dataset (e.g., “Existing Customer,” “Graduate,” “Married,” and “Blue” card type). This strategy preserves the overall statistical structure and dominant customer profile patterns while ensuring that all missing entries are replaced with contextually meaningful values.



A1 | fx CLIENTNUM

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
1	CLIENTNU	Attrition_	Customer	Gender	Dependen	Education	Marital_S	Income_C	Card_Cate	Months_c	Total_Rela	Months_I	Contacts_	Credit_Lir	Total_Rev	Avg_Oper	Total_Am	Total_Trar	Total_Trar	Total_Ct_	Avg_Utiliz	Naive_Ba	Naive_Ba
2	7688053	Existing C	45	M	3	High Scho	Married	$60K - $8	Blue	39	5	1	3	12691	777	11914	1.335	1144	42	1.625	0.061	0.000093	0.99991
3	8187700	Existing C	49	F	5	Graduate	Single	Less than	Blue	44	6	1	2	8256	864	7392	1.541	1291	33	3.714	0.105	0.000056	0.99994
4	7139821	Existing C	51	M	3	Graduate	Married	$80K - $1	Blue	36	4	1	0	3418	0	3418	2.594	1887	20	2.333	0	0.000021	0.99998
5	7699118	Existing C	40	F	4	High Scho	Married	Less than	Blue	34	3	4	1	3313	2517	796	1.405	1171	20	2.333	0.76	0.000133	0.99987
6	7091063	Existing C	40	M	3	Uneducat	Married	$60K - $8	Blue	21	5	1	0	4716	0	4716	2.175	816	28	2.5	0	0.000021	0.99998
7	7130615	Existing C	44	M	2	Graduate	Married	$40K - $6	Blue	36	3	1	2	4010	1247	2763	1.376	1088	24	0.846	0.311	0.000055	0.99994
8	8103472	Existing C	51	M	4	Graduate	Married	$120K +	Gold	46	6	1	3	34516	2264	32252	1.975	1330	31	0.722	0.066	0.000123	0.99988
9	8189062	Existing C	32	M	0	High Scho	Married	$60K - $8	Silver	27	2	2	2	29081	1396	27685	2.204	1538	36	0.714	0.048	0.000085	0.99991
10	7109305	Existing C	37	M	3	Uneducat	Single	$60K - $8	Blue	36	5	2	0	22352	2517	19835	3.355	1350	24	1.182	0.113	0.000044	0.99996
11	7196615	Existing C	48	M	2	Graduate	Single	$80K - $1	Blue	36	6	3	3	11656	1677	9979	1.524	1441	32	0.882	0.144	0.000302	0.9997
12	7087908	Existing C	42	M	5	Uneducat	Married	$120K +	Blue	31	5	3	2	6748	1467	5281	0.831	1201	42	0.68	0.217	0.000190	0.99981
13	7108218	Existing C	65	M	1	Graduate	Married	$40K - $6	Blue	54	6	2	3	9095	1587	7508	1.433	1314	26	1.364	0.174	0.000197	0.9998
14	7105996	Existing C	56	M	1	College	Single	$80K - $1	Blue	36	3	6	0	11751	0	11751	3.397	1539	17	3.25	0	0.000047	0.99995
15	8160822	Existing C	35	M	3	Graduate	Married	$60K - $8	Blue	30	5	1	3	8547	1666	6881	1.163	1311	33	2	0.195	0.000096	0.9999
16	7123969	Existing C	57	F	2	Graduate	Married	Less than	Blue	48	5	2	2	2436	680	1756	1.19	1570	29	0.611	0.279	0.000113	0.99989
17	7148852	Existing C	44	M	4	Graduate	Married	$80K - $1	Blue	37	5	1	2	4234	972	3262	1.707	1348	27	1.7	0.23	0.000063	0.99994
18	7099673	Existing C	48	M	4	Post-Grad	Single	$80K - $1	Blue	36	6	2	3	30367	2362	28005	1.708	1671	27	0.929	0.078	0.000236	0.99976
19	7533273	Existing C	41	M	3	Graduate	Married	$80K - $1	Blue	34	4	4	1	13535	1291	12244	0.653	1028	21	1.625	0.095	0.000149	0.99985
20	8061601	Existing C	61	M	1	High Scho	Married	$40K - $6	Blue	56	2	2	3	3193	2517	676	1.831	1336	30	1.143	0.788	0.000174	0.99983
21	7093273	Existing C	45	F	2	Graduate	Married	Less than	Blue	37	6	1	2	14470	1157	13313	0.966	1207	21	0.909	0.08	0.000055	0.99994
22	8061652	Existing C	47	M	1	Doctorate	Divorced	$60K - $8	Blue	42	5	2	0	20979	1800	19179	0.906	1178	27	0.929	0.086	0.000057	0.99994
23	7085087	Attrited C	62	F	0	Graduate	Married	Less than	Blue	49	2	3	3	1438.3	0	1438.3	1.047	692	16	0.6	0	0.99616	0.003836
24	7847253	Existing C	41	M	3	High Scho	Married	$40K - $6	Blue	33	4	2	1	4470	680	3790	1.608	931	18	1.571	0.152	0.000069	0.99993
25	8116041	Existing C	47	F	4	Graduate	Single	Less than	Blue	36	3	3	2	2492	1560	932	0.573	1126	23	0.353	0.626	0.000207	0.99979
26	7891246	Existing C	54	M	2	Graduate	Married	$80K - $1	Blue	42	4	2	3	12217	0	12217	1.075	1110	21	0.75	0	0.000210	0.99979
27	7710719	Existing C	41	F	3	Graduate	Single	Less than	Blue	28	6	1	2	7768	1669	6099	0.797	1051	22	0.833	0.215	0.000057	0.99994
28	7204663	Existing C	59	M	1	High Scho	Married	$40K - $6	Blue	46	4	1	2	14784	1374	13410	0.921	1197	23	1.3	0.093	0.000050	0.99995

Fig 6: Output using Excel Formulas

Insights:

The output reflects a structured data preprocessing workflow designed to ensure dataset completeness while preserving the statistical integrity of customer credit behavior data. Rather than applying arbitrary replacements, the process evaluates key descriptive statistics—including Mean, Median, Mode, Variance, and Standard Deviation—to understand the distribution of each numerical attribute before selecting appropriate imputation values. For numerical variables such as Customer Age and transaction-related metrics, the Median is used as the primary estimator because it provides a robust central value that is less sensitive to extreme values or skewed financial distributions. In contrast, categorical attributes—including Attrition Flag, Gender, Education Level, Marital Status, Income Category, and Card Category—are imputed using the Mode, ensuring that missing entries align with the most frequently observed customer profile patterns in the dataset.

Conclusion:

Data cleaning plays a critical role in preparing structured financial datasets for reliable analysis and predictive modeling. Raw datasets often contain inconsistencies such as missing entries, placeholder values like “Unknown,” or irregular categorical labels that can distort statistical distributions and negatively impact model performance. Addressing these issues through systematic preprocessing ensures that the dataset accurately reflects underlying customer behavior patterns. By transforming incomplete or noisy data into a consistent and well-structured format, the cleaning process strengthens the reliability of downstream analytical tasks such as customer segmentation, churn prediction, and financial behavior analysis.

In this experiment, we applied multiple complementary data cleaning and imputation techniques to the BankChurners dataset to handle missing and irregular values. Initially, placeholder entries such as “Unknown” were converted to NaN to standardize missing value representation across the dataset. Subsequently, Central Tendency Imputation was implemented, where the median was used to fill missing numerical variables—such as customer age or transaction-related metrics—because it is robust to outliers and skewed financial distributions. For categorical attributes including Education Level, Marital Status, Income Category, and Card Category, the mode was used to preserve the most common customer profile patterns. Additionally, an Expectation–Maximization (EM) based imputation approach was explored for numerical variables, leveraging relationships between financial indicators to iteratively estimate missing values while maintaining covariance structure within the data. Together, these techniques transformed the dataset into a complete, statistically consistent, and analysis-ready resource, demonstrating how a combination of statistical imputation and structured preprocessing can significantly improve data integrity for advanced analytics and machine learning applications.

Post Experiment Questions:

1. **Explain why dropping rows with missing values can introduce bias in predictive modelling**

   Dropping rows with missing values, often called Complete Case Analysis (CCA) is a common shortcut in data preprocessing. While it’s mathematically simple, it can severely compromise a model's integrity by introducing Bias.

   To understand bias, we first have to categorize why the data is gone. The risk of bias depends entirely on which category the missingness falls into:

   - **Missing Completely at Random (MCAR):**

   This is the ideal scenario. The fact that a value is missing has nothing to do with its hypothetical value or any other values in the dataset. It's like dropping a stack of papers and losing three at random—pure bad luck.

Example: A thermometer runs out of battery and stops recording temperature for an hour. The missingness is unrelated to how hot it is.

- **Missing at Random (MAR)**

This name is notoriously confusing. MAR means the missingness can be explained by other observed variables in your dataset, but not by the missing value itself.

Example: Men are less likely to report "Depression Score" than women. If you have "Gender" in your dataset, you can account for this missingness. The missingness depends on Gender (observed), not on the Depression Score itself.

- **Missing Not at Random (MNAR)**

This is the danger zone. The value is missing because of what the value would have been.

Example: People with very high incomes refuse to fill out the "Income" field in a survey because of privacy concerns. The fact that it is missing tells you the income is likely high. You should only use deletion (listwise or pairwise) when the data is Missing Completely at Random (MCAR) and the percentage of missing records is trivial (typically < 5%). If the data is not MCAR, deleting rows introduces selection bias, altering the population distribution your model learns from.

How Bias Manifests:

1. If you drop rows where a specific group is less likely to respond, your training set no longer reflects the real world. For example, if a medical study drops patients who missed follow-up appointments because they were too sick to attend, the model will "conclude" that the treatment is more effective than it actually is. It has only seen the "healthy" survivors.
2. By deleting rows, you artificially reduce the variance in your data. A predictive model trained on a "clean" but narrow subset of data may show high accuracy during training but fail miserably on real-world data where those missing-value scenarios occur frequently.
3. Dropping rows can create "phantom" correlations or mask real ones. If X and Y are correlated, but you drop rows where X is high, the resulting regression line will have a different slope than the true population line.

2. **Describe a scenario where forward fill imputation in time series data would lead to misleading data**

   Forward fill (ffill) is a common "lazy" imputation technique where you carry the last known observation forward to fill a gap. While it's great for data that stays constant (like a subscription status), it can be disastrous for data that is mean-reverting or volatile.

   Scenario: A Rapid Market Correction

   Imagine you are tracking the price of a high-volatility stock

   10:00 AM: The price is $100.
   10:01 AM - 10:10 AM: A massive sell-off occurs, but the data feed drops out due to high traffic (a common occurrence during crashes). No new data is recorded.
   10:11 AM: The feed resumes, and the price is now $70.
   If you apply df.ffill(), your dataset will show the price stayed at $100 for that entire ten-minute window.

   - **Artificial Stability:** Your model "thinks" the asset was stable and liquid during a period of extreme stress. If you are training a reinforcement learning agent (like a PPO agent for trading), it will learn that it could have "sold" at $100 during those ten minutes, which is a hallucination.
   - **Hidden Volatility:** You've effectively deleted the "volatility" signature of the crash. Real-world trading algorithms look for rapid price drops to trigger stops; forward fill makes the drop look like a single, instantaneous "teleportation" from $100 to $70, rather than a slide.
   - **Look-ahead Bias (Indirectly):** While ffill doesn't technically use future data, it keeps "stale" data alive long after its market relevance has expired.

Scenario	**True Movement**	**Forward Fill Result**	**Model Impact**
Steady Growth	$10 → 11 → 12	$10 → 10 → 12	Underestimates growth rate.
Market Crash	$100 → 85 → 70	$100 → 100 → 70	Dangerous, Overestimates value/liquidity.
Cyclical (Sine)	$0 → 1 → 0	$0 → 0 → 0	Completely misses the peak/cycle.