Start coding or generate with AI.

Dataset uploading

```
# prompt: DATASET UPLOAD CODE

from google.colab import files
uploaded = files.upload()
# prompt: handle missing values for an csv file in google colab

import pandas as pd

# Load the CSV file into a pandas DataFrame
df = pd.read_csv('customer.csv')
```

```
Choose Files No file chosen        Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to
enable
```

Handling Missing Values

```
# prompt: check for missing values and fill the missing values in the above dataset

# Check for missing values
print(df.isnull().sum())

# Fill missing values with mean for numerical columns
numerical_cols = df.select_dtypes(include=['number']).columns
df[numerical_cols] = df[numerical_cols].fillna(df[numerical_cols].mean())

# Fill missing values with mode for categorical columns
categorical_cols = df.select_dtypes(include=['object']).columns
df[categorical_cols] = df[categorical_cols].fillna(df[categorical_cols].mode().iloc[0])

# Verify if missing values are filled
print(df.isnull().sum())
```

```
Year                           0
Date                           0
Question_Number                0
Question                       0
Number_of_Respondents          0
Very_Satisfied                 0
Satisfied                      0
Neutral                        0
Dissatisfied                   0
Very_Dissatisfied              0
Very_Satisfied_or_Satisfied    0
ObjectId                       0
dtype: int64
Year                           0
Date                           0
Question_Number                0
Question                       0
Number_of_Respondents          0
Very_Satisfied                 0
Satisfied                      0
Neutral                        0
Dissatisfied                   0
Very_Dissatisfied              0
Very_Satisfied_or_Satisfied    0
ObjectId                       0
dtype: int64
```

Duplicate records

```
# prompt: check for the duplicate records and remove them

# Check for duplicate rows
duplicate_rows = df[df.duplicated()]

# Print the duplicate rows
```

```
print("Duplicate Rows:")
print(duplicate_rows)

# Remove duplicate rows
df = df.drop_duplicates()

# Print the DataFrame after removing duplicates
print("\nDataFrame after removing duplicates:")
df
```

```
Duplicate Rows:
Empty DataFrame
Columns: [Year, Date, Question_Number, Question, Number_of_Respondents, Very_Satisfied, Satisfied, Neutral, Dissatisfied, Very_Dissat
Index: []

DataFrame after removing duplicates:
```

	Year	Date	Question_Number	Question	Number_of_Respondents	Very_Satisfied	Satisfied	Neutral	Dissatisfied	Very_Dissa
0	2017	2017/10/31 07:00:00+00	7-13	Overall quality of customer service	882	22.16	47.61	25.03	3.15	
1	2016	2016/10/31 07:00:00+00	26	Overall quality of customer service	1202	22.39	47.83	23.37	4.35	
2	2015	2015/10/31 07:00:00+00	Survey Not Conducted	Survey Not Conducted	99999	99999.00	99999.00	99999.00	99999.00	9
3	2014	2014/10/31 07:00:00+00	10b	How easy was the City to contact	493	35.95	46.07	8.26	7.44	
4	2014	2014/10/31 07:00:00+00	10c	The way you were treated	493	44.15	40.25	8.83	4.72	
5	2014	2014/10/31 07:00:00+00	10d	The accuracy of the information you were given	493	36.76	40.55	14.08	6.51	
6	2014	2014/10/31 07:00:00+00	10e	How quickly staff responded to your request	493	38.05	37.63	13.10	7.69	
7	2014	2014/10/31 07:00:00+00	10f	How well your issue was handled	493	37.92	34.58	12.50	8.96	
8	2013	2013/10/31 07:00:00+00	14b	How easy was the City to contact	428	37.68	44.31	8.06	7.58	
9	2013	2013/10/31 07:00:00+00	14c	The way you were treated	428	44.47	40.14	8.17	4.09	
10	2013	2013/10/31 07:00:00+00	14d	The accuracy of the information you were given	428	39.51	39.27	13.17	4.63	
11	2013	2013/10/31 07:00:00+00	14e	How quickly staff responded to your request	428	40.00	37.83	9.40	6.27	
12	2013	2013/10/31 07:00:00+00	14f	How well your issue was handled	428	37.20	37.20	9.90	8.45	
13	2012	2012/10/31 07:00:00+00	14b	How easy was the city to contact	403	36.00	46.00	8.00	6.00	
14	2012	2012/10/31 07:00:00+00	14c	The way you were treated	403	44.00	38.00	11.00	4.00	
				The						

15	2012	2012/10/31 07:00:00+00	14d	accuracy of the information you were given	403	40.00	37.00	11.00	8.00
16	2012	2012/10/31 07:00:00+00	14e	How quickly responded to request	403	39.00	38.00	10.00	7.00
17	2012	2012/10/31 07:00:00+00	14f	How well issue was handled	403	40.00	34.00	11.00	9.00
18	2011	2011/10/31 07:00:00+00	14b	How easy was the contact	416	33.17	44.31	13.08	7.75
19	2011	2011/10/31 07:00:00+00	14c	The way you were treated	416	42.82	38.69	11.68	4.62
20	2011	2011/10/31 07:00:00+00	14d	Accuracy of information you were given	416	38.35	38.10	14.79	5.26
21	2011	2011/10/31 07:00:00+00	14e	How quickly staff responded to your request	416	36.54	37.78	13.33	6.67
22	2011	2011/10/31 07:00:00+00	14f	How well your issue was handled	416	37.06	34.83	12.69	8.46
23	2010	2010/10/31 07:00:00+00	14B	How easy was the city to contact	424	37.68	43.00	10.63	6.76
24	2010	2010/10/31 07:00:00+00	14C	The way you were treated	424	41.56	40.10	11.98	3.91
25	2010	2010/10/31 07:00:00+00	14D	Accuracy of information you were given	424	37.59	39.07	12.78	6.63
26	2010	2010/10/31 07:00:00+00	14E	How quickly staff responded to your request	424	39.51	32.20	16.10	5.37
27	2010	2010/10/31 07:00:00+00	14F	How well your issue was handled	424	36.32	31.96	15.98	7.51

Outliers

```
# prompt: check for the outliers in the above dataset

import pandas as pd
import numpy as np

# Assuming 'df' is your DataFrame with numerical features

def find_outliers_iqr(data):
    Q1 = np.percentile(data, 25)
    Q3 = np.percentile(data, 75)
    IQR = Q3 - Q1
    lower_bound = Q1 - 1.5 * IQR
    upper_bound = Q3 + 1.5 * IQR
    outliers = data[(data < lower_bound) | (data > upper_bound)]
    return outliers


numerical_features = df.select_dtypes(include=np.number).columns
for col in numerical_features:
    outliers = find_outliers_iqr(df[col])
    print(f"Outliers in {col}:")
    print(outliers)
    print("-" * 20)
```

```
Outliers in Year:
Series([], Name: Year, dtype: int64)
--------------------
Outliers in Number_of_Respondents:
0      882
1     1202
2    99999
Name: Number_of_Respondents, dtype: int64
--------------------
Outliers in Very_Satisfied:
0       22.16
1       22.39
2    99999.00
Name: Very_Satisfied, dtype: float64
--------------------
Outliers in Satisfied:
2    99999.0
Name: Satisfied, dtype: float64
--------------------
Outliers in Neutral:
0       25.03
1       23.37
2    99999.00
Name: Neutral, dtype: float64
--------------------
Outliers in Dissatisfied:
2    99999.0
Name: Dissatisfied, dtype: float64
--------------------
Outliers in Very_Dissatisfied:
2    99999.0
Name: Very_Dissatisfied, dtype: float64
--------------------
Outliers in Very_Satisfied_or_Satisfied:
2    99999.0
Name: Very_Satisfied_or_Satisfied, dtype: float64
--------------------
Outliers in ObjectId:
Series([], Name: ObjectId, dtype: int64)
--------------------
```

Standardization

```
# prompt: standardize the above dataset

from sklearn.preprocessing import StandardScaler

# Assuming 'df' is your DataFrame with numerical features
# Create a StandardScaler object
scaler = StandardScaler()

# Select numerical columns for standardization
numerical_cols = df.select_dtypes(include=np.number).columns

# Fit and transform the numerical columns
df[numerical_cols] = scaler.fit_transform(df[numerical_cols])

# Print the standardized DataFrame
print("\nStandardized DataFrame:")
df
```

```
Standardized DataFrame:
```

	Year	Date	Question_Number	Question	Number_of_Respondents	Very_Satisfied	Satisfied	Neutral	Dissatisfied	Very_Dis
0	2.483682	2017/10/31 07:00:00+00	7-13	Overall quality of customer service	-0.170565	-0.193284	-0.192013	-0.191776	-0.192621	
1	1.940376	2016/10/31 07:00:00+00	26	Overall quality of customer service	-0.153239	-0.193271	-0.192001	-0.191865	-0.192557	
2	1.397071	2015/10/31 07:00:00+00	Survey Not Conducted	Survey Not Conducted	5.195945	5.196152	5.196152	5.196152	5.196152	
3	0.853766	2014/10/31 07:00:00+00	10b	How easy was the City to contact	-0.191626	-0.192540	-0.192096	-0.192679	-0.192390	
4	0.853766	2014/10/31 07:00:00+00	10c	The way you were treated	-0.191626	-0.192098	-0.192409	-0.192649	-0.192537	
5	0.853766	2014/10/31 07:00:00+00	10d	The accuracy of the information you were given	-0.191626	-0.192497	-0.192393	-0.192366	-0.192440	
6	0.853766	2014/10/31 07:00:00+00	10e	How quickly staff responded to your request	-0.191626	-0.192427	-0.192551	-0.192419	-0.192377	
7	0.853766	2014/10/31 07:00:00+00	10f	How well your issue was handled	-0.191626	-0.192434	-0.192715	-0.192451	-0.192308	
8	0.310460	2013/10/31 07:00:00+00	14b	How easy was the City to contact	-0.195146	-0.192447	-0.192190	-0.192690	-0.192382	
9	0.310460	2013/10/31 07:00:00+00	14c	The way you were treated	-0.195146	-0.192081	-0.192415	-0.192684	-0.192571	
10	0.310460	2013/10/31 07:00:00+00	14d	The accuracy of the information you were given	-0.195146	-0.192348	-0.192462	-0.192415	-0.192541	
11	0.310460	2013/10/31 07:00:00+00	14e	How quickly staff responded to your request	-0.195146	-0.192322	-0.192540	-0.192618	-0.192453	
12	0.310460	2013/10/31 07:00:00+00	14f	How well your issue was handled	-0.195146	-0.192473	-0.192574	-0.192591	-0.192336	
13	-0.232845	2012/10/31 07:00:00+00	14b	How easy was the city to contact	-0.196499	-0.192538	-0.192099	-0.192693	-0.192468	
14	-0.232845	2012/10/31 07:00:00+00	14c	The way you were treated	-0.196499	-0.192106	-0.192531	-0.192532	-0.192575	
15	-0.232845	2012/10/31 07:00:00+00	14d	The accuracy of the information you were	-0.196499	-0.192322	-0.192584	-0.192532	-0.192360	

				given					
16	-0.232845	2012/10/31 07:00:00+00	14e	How quickly responded to request	-0.196499	-0.192376	-0.192531	-0.192586	-0.192414
17	-0.232845	2012/10/31 07:00:00+00	14f	How well issue was handled	-0.196499	-0.192322	-0.192746	-0.192532	-0.192306
18	-0.776151	2011/10/31 07:00:00+00	14b	How easy was the contact	-0.195795	-0.192690	-0.192190	-0.192420	-0.192373
19	-0.776151	2011/10/31 07:00:00+00	14c	The way you were treated	-0.195795	-0.192170	-0.192493	-0.192495	-0.192542
20	-0.776151	2011/10/31 07:00:00+00	14d	Accuracy of information you were given	-0.195795	-0.192411	-0.192525	-0.192327	-0.192507
21	-0.776151	2011/10/31 07:00:00+00	14e	How quickly staff responded to your request	-0.195795	-0.192509	-0.192542	-0.192406	-0.192432
22	-0.776151	2011/10/31 07:00:00+00	14f	How well your issue was handled	-0.195795	-0.192480	-0.192701	-0.192441	-0.192335
23	-1.319456	2010/10/31 07:00:00+00	14B	How easy was the city to contact	-0.195362	-0.192447	-0.192261	-0.192552	-0.192427
24	-1.319456	2010/10/31 07:00:00+00	14C	The way you were treated	-0.195362	-0.192238	-0.192417	-0.192479	-0.192580
25	-1.319456	2010/10/31 07:00:00+00	14D	Accuracy of information you were given	-0.195362	-0.192452	-0.192473	-0.192436	-0.192434
26	-1.319456	2010/10/31 07:00:00+00	14E	How quickly staff responded to your request	-0.195362	-0.192348	-0.192843	-0.192257	-0.192502
27	-1.319456	2010/10/31 07:00:00+00	14F	How well your issue was handled	-0.195362	-0.192520	-0.192856	-0.192263	-0.192386