- Name - Prathmesh Bhimraj Bhukele
- Contact no. 8291159068
- Prathmeshbb26@gmail.com

Exploring the Dataset

In [ ]:
```
import pandas as pd
import numpy as np
```

In [ ]:
```
# Loading the "Data" Dataset
data = pd.read_csv("DA_data.csv")
```

In [ ]:
```
data.head()
```

Out[ ]:

	Customer_ID	Weeks	Contract_Renewal	Data_Plan	Data_Usage	Calls_To_Customer_
0	1001	47	1	Yes	2.3	
1	1002	30	1	No	0.0	
2	1003	52	0	Yes	4.1	
3	1004	25	1	No	0.0	
4	1005	38	1	Yes	2.6	

In [ ]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 134 entries, 0 to 133
Data columns (total 13 columns):
 #   Column                  Non-Null Count  Dtype  
---  ------                  --------------  -----  
 0   Customer_ID             134 non-null    int64  
 1   Weeks                   134 non-null    int64  
 2   Contract_Renewal        134 non-null    int64  
 3   Data_Plan               134 non-null    object 
 4   Data_Usage              134 non-null    float64
 5   Calls_To_Customer_Care  134 non-null    int64  
 6   DayMins                 134 non-null    float64
 7   DayCalls                134 non-null    int64  
 8   MonthlyCharge           133 non-null    float64
 9   OverageFee              134 non-null    float64
 10  RoamMins                134 non-null    float64
 11  Customer_Attrition      134 non-null    object 
 12  Inserted_Date           134 non-null    object 
dtypes: float64(5), int64(5), object(3)
memory usage: 13.7+ KB
```

In [ ]:
```
data.isnull()
```

Out[ ]:

	Customer_ID	Weeks	Contract_Renewal	Data_Plan	Data_Usage	Calls_To_Custome
0	False	False	False	False	False	
1	False	False	False	False	False	
2	False	False	False	False	False	
3	False	False	False	False	False	
4	False	False	False	False	False	
...	...	...	...	...	...	
129	False	False	False	False	False	
130	False	False	False	False	False	
131	False	False	False	False	False	
132	False	False	False	False	False	
133	False	False	False	False	False	

134 rows × 13 columns

In [ ]:
```
data.shape
```

Out[ ]:
```
(134, 13)
```

In [ ]:
```
data.size
```

Out[ ]:
```
1742
```

In [ ]:
```
data.describe()
```

Out[ ]:

	Customer_ID	Weeks	Contract_Renewal	Data_Usage	Calls_To_Customer_Car
count	134.000000	134.000000	134.000000	134.000000	134.00000
mean	1066.902985	47.007463	0.925373	2.085821	1.41791
std	37.960183	8.867866	0.263774	0.992994	1.04268
min	1001.000000	24.000000	0.000000	0.000000	0.00000
25%	1034.250000	42.000000	1.000000	1.800000	1.00000
50%	1067.500000	50.000000	1.000000	2.250000	1.00000
75%	1100.750000	53.000000	1.000000	2.675000	2.00000
max	1130.000000	64.000000	1.000000	4.200000	4.00000

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

Out[ ]:
```
Customer_ID               0
Weeks                     0
Contract_Renewal          0
Data_Plan                 0
Data_Usage                0
Calls_To_Customer_Care    0
DayMins                   0
DayCalls                  0
MonthlyCharge             1
OverageFee                0
RoamMins                  0
Customer_Attrition        0
Inserted_Date             0
dtype: int64
```

Data cleaning

1. Removed the duplicated values
2. Imputed the missing values in Monthlychange & Overagefee using mean values
3. Converted the type of Inserted_date from int to Datetime format.
4. Changed the datatype of Date_plan , Contract_Renewal, Customer_Attrition from object to boolean .
5. Changed the datatype of Data_Usage , Calls_To_Customer_Care , OverageFee , MonthlyCharge to numeric format.

In [ ]:
```
# Removing duplicates
data = data.drop_duplicates()
```

In [ ]:
```
data.shape
```

Out[ ]:
```
(130, 13)
```

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

In [ ]:
```
data['Inserted_Date'] = pd.to_datetime(data['Inserted_Date'], errors='coe
```

In [ ]:
```
data['Data_Plan'] = data['Data_Plan'].map({'Yes': True, 'No': False})
```

In [ ]:
```
data['Customer_Attrition'] = data['Customer_Attrition'].map({'Yes': True,
```

In [ ]:
```
data['Data_Usage'] = pd.to_numeric(data['Data_Usage'])
```

In [ ]:
```
data['Calls_To_Customer_Care'] = pd.to_numeric(data['Calls_To_Customer_Ca
```

In [ ]:
```
data['OverageFee'] = pd.to_numeric(data['OverageFee'])
data['MonthlyCharge'] = pd.to_numeric(data['MonthlyCharge'])
```

In [ ]:
```
data['Contract_Renewal'] = data['Contract_Renewal'].map({1: True, 0: Fals
```

In [ ]:
```
data.shape
```

Out[ ]:
```
(130, 13)
```

In [ ]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 130 entries, 0 to 133
Data columns (total 13 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   Customer_ID             130 non-null    int64
 1   Weeks                   130 non-null    int64
 2   Contract_Renewal        130 non-null    bool
 3   Data_Plan               130 non-null    bool
 4   Data_Usage              130 non-null    float64
 5   Calls_To_Customer_Care  130 non-null    int64
 6   DayMins                 130 non-null    float64
 7   DayCalls                130 non-null    int64
 8   MonthlyCharge           130 non-null    float64
 9   OverageFee              130 non-null    float64
 10  RoamMins                130 non-null    float64
 11  Customer_Attrition      130 non-null    bool
 12  Inserted_Date           130 non-null    datetime64[ns]
dtypes: bool(3), datetime64[ns](1), float64(5), int64(4)
memory usage: 11.6 KB
```

Solutions

1. What is the correlation between the number of calls to customer care and customer attrition?

In [ ]:
```
correlation_calls_attrition = data['Calls_To_Customer_Care'].corr(data['C
print("Correlation between calls to customer care and customer attrition:
```

```
Correlation between calls to customer care and customer attrition: 0.177
55811850077777
```

- The correlation between calls to customer care and customer attritions is 0.17 which shows a weak positive correlation.

2. Which data plan (Yes or No) has a higher average monthly charge?

In [ ]:
```
average_monthly_charge_by_plan = data.groupby('Data_Plan')['MonthlyCharge
higher_average_monthly_charge_plan = average_monthly_charge_by_plan.idxma
print("Data plan with higher average monthly charge:", higher_average_mon
```

```
Data plan with higher average monthly charge: True
```

- "True" indicates that Customers with a data plan ('Yes') have a higher average monthly charge.

3. Is there any correlation between customer attrition and contract renewal?

```
In [ ]: correlation_attrition_renewal = data['Customer_Attrition'].corr(data['Co
        print("Correlation between customer attrition and contract renewal:", co
```

```
Correlation between customer attrition and contract renewal: -0.15144803
708370724
```

- The correlation coefficient of approximately -0.1514 between customer attrition and contract renewal suggests a weak negative correlation

4. Which feature(s) have the highest correlation with customer attrition?

```
In [ ]: correlation_matrix = data.corr()
        highest_correlation_with_attrition = correlation_matrix['Customer_Attriti
        print("Feature(s) with highest correlation with customer attrition:\n", h
```

```
Feature(s) with highest correlation with customer attrition:
 Customer_Attrition    1.000000
OverageFee            0.318281
Name: Customer_Attrition, dtype: float64
```

- The feature with the highest correlation coefficient with customer attrition is 'OverageFee' with a correlation coefficient of approximately 0.318.

5. Is there a difference in data usage between customers who have a data plan and those who do not?

```
In [ ]: data_usage_difference = data.groupby('Data_Plan')['Data_Usage'].mean()
        print("Difference in data usage between customers with and without a data
```

```
Difference in data usage between customers with and without a data plan:
 Data_Plan
False    0.000000
True     2.398214
Name: Data_Usage, dtype: float64
```

- Customers with a data plan use an average of approximately 2.39 GB of data, while those without a data plan use no data on average

6. What is the total revenue from customers who have a data plan and used greater than 3 GB of data

```
In [ ]: total_revenue_data_plan_gt_3gb = (data['MonthlyCharge'] + data['OverageFe
        print("Total revenue from customers with a data plan and used > 3 GB of d
```

```
Total revenue from customers with a data plan and used > 3 GB of data: 1
131.8
```

- The Total revenue from customers with a data plan and used more then 3 GB of data is $ 1131.8

7. What % of total revenue comes from customers who do not have a data plan?

In [ ]:
```
#total revenue from customers without a data plan
total_revenue_no_data_plan = (data['MonthlyCharge'] + data['OverageFee'])
print(total_revenue_no_data_plan)
```

```
1036.25
```

In [ ]:
```
# percentage of total revenue from customers without a data plan
percentage_revenue_no_data_plan = (total_revenue_no_data_plan / (total_re
print("% of total revenue from customers who do not have a data plan:", p
```

```
% of total revenue from customers who do not have a data plan: 47.796406
909434744
```

- The percentage of total revenue from customers who do not have a data plan is approximately 47.79% and the total revenue generated from customers with no data plan is $1036.25

8. What is the ratio of total revenue between customers who have a data plan and those who do not?

In [ ]:
```
#  total revenue from customers without a data plan
total_revenue_no_data_plan = (data['MonthlyCharge'] + data['OverageFee'])
print(total_revenue_no_data_plan)
```

```
1036.25
```

In [ ]:
```
# Calculating total revenue from customers with a data plan
total_revenue_data_plan = (data['MonthlyCharge'] + data['OverageFee']).lo
print(total_revenue_data_plan)
```

```
8065.36046511628
```

In [ ]:
```
# Calculating the ratio of total revenuee between customers with and with
revenue_ratio = total_revenue_data_plan / total_revenue_no_data_plan
print("Ratio :", revenue_ratio)
```

```
Ratio : 7.783218784189414
```

- This ratio indicates that the total revenue from customers with a data plan is about 7.783 times higher than the total revenue from customers without a data plan.

9. How many customers have a renewed contract? Are customers with a data plan less likely to renew their contract vs customers with no data plan?

In [ ]:
```
#number of customers with renewed contracts
renewed_customers = data.loc[data['Contract_Renewal'], 'Customer_ID'].nur

#number of customers with renewed contracts and a data plan
data_plan_renewed_customers = data.loc[data['Data_Plan'] & data['Contract
```

```
# number of customers with renewed contracts and no data plan
no_data_plan_renewed_customers = data.loc[~data['Data_Plan'] & data['Cont
```

In [ ]:
```
# percentage of customers with a data plan who renewed their contract
percentage_data_plan_renewed = (data_plan_renewed_customers / data['Data_
# the percentage of customers with no data plan who renewed their contrac
percentage_no_data_plan_renewed = (no_data_plan_renewed_customers / (~dat
```

In [ ]:
```
print("Number of customers with renewed contract:", renewed_customers)
print("Number of customers with a data plan who renewed their contract:",
print("Number of customers with no data plan who renewed their contract:'
print("% of customers with a data plan who renewed their contract:", perc
print("% of customers with no data plan who renewed their contract:", per
```

```
Number of customers with renewed contract: 120
Number of customers with a data plan who renewed their contract: 110
Number of customers with no data plan who renewed their contract: 10
% of customers with a data plan who renewed their contract: 98.214285714
28571
% of customers with no data plan who renewed their contract: 55.55555555
555556
```

- Out of 120 customers with renewed contracts, 110 have a data plan (98.21% renewal rate), while only 10 do not have a data plan (55.55% renewal rate).

10. What is the % of Overage Fees to Total Revenue? What is this ratio for customers with no data plan, customers using 1-3 GB of data and customers using greater than 3 GB of data?

In [ ]:
```
# Calculatign total revenue
total_revenue = data['MonthlyCharge'].sum() + data['OverageFee'].sum()
# Calculating total overage fees
total_overage_fees = data['OverageFee'].sum()
#  the percentage of overage fees to the total revenue
percentage_overage_fees_to_revenue = (total_overage_fees / total_revenue)

print("% of Overage Fees to Total Revenue:", percentage_overage_fees_to_r
```

```
% of Overage Fees to Total Revenue: 14.928676690874413
```

- The percentage of overage fees to total revenue is 14.93%.

11. Do customers with weeks more than 50 have a lower minute per call ratio or customers with weeks between 31 and 50 ?

In [ ]:
```
average_minute_per_call_weeks_50_plus = data['DayMins'].loc[data['Weeks']
average_minute_per_call_weeks_31_50 = data['DayMins'].loc[(data['Weeks']
print(average_minute_per_call_weeks_31_50)
print(average_minute_per_call_weeks_50_plus)
```

```
1.862611717974181
1.9277768385460692
```

In [ ]:
```
if average_minute_per_call_weeks_50_plus < average_minute_per_call_weeks_
    print("Customers with weeks more than 50 have a lower minute per call
else:
    print("Customers with weeks between 31 and 50 have a lower minute per
```

```
Customers with weeks between 31 and 50 have a lower minute per call rati
o.
```

- The customers with weeks between 31 and 50 have a lower minute per call ratio.

12. What is the average overage fee for customers whose contracts are more than 30 weeks old and have a data plan and have used less than 1GB of data?

In [ ]:
```
average_overage_fee = data['OverageFee'].loc[(data['Weeks'] > 30) & (data
print("Average overage fee for eligible customers:", average_overage_fee)
```

```
Average overage fee for eligible customers: nan
```

- There are no eligible customers to calculate the average overage fee.

13. What is the average monthly charge for customers whose contracts are more than 50 weeks old and have a data plan and have renewed their contract?

In [ ]:
```
# Filtered the data for customers whose contracts are more than 50 weeks
filtered_customers = data[(data['Weeks'] > 50) & (data['Data_Plan']) & (d
```

In [ ]:
```
# Calculated the average monthly charge whose contracts are more than 50
average_monthly_charge = filtered_customers['MonthlyCharge'].mean()
print(average_monthly_charge)
```

```
62.00680122860904
```

- The average monthly charge for customers whose contracts are more than 50 weeks old, have a data plan, and have renewed their contract is 62.00

14. What is the average roam minutes for customers whose contracts are between 31-50 weeks and have a data plan and have used greater than 3GB of data?

In [ ]:
```
filtered_customers = data[(data['Weeks'] >= 31) & (data['Weeks'] <= 50) &

# Calculated the average roam minutes
average_roam_minutes = filtered_customers['RoamMins'].mean()

print(average_roam_minutes)
```

```
9.955555555555556
```

- Average roam minutes for customers whose contracts are between 31-50 weeks old, have a data plan, and have used greater than 3GB of data is 9.95

15. What is the average data usage for customers whose contracts are more than 30 weeks old and have renewed their contract?

In [ ]:
```
# Filtering the Data for customers whose contracts are more than 30 weeks
filtered_customers = data[(data['Weeks'] > 30) & (data['Contract_Renewal'
```

In [ ]:
```
# Calculating the average data usage
average_data_usage = filtered_customers['Data_Usage'].mean()
print(average_data_usage)
```

```
2.2585585585585584
```

- Average data usage for customers whose contracts are more than 30 weeks old and have renewed their contract is 2.258