Food Delivery Cost Analysis Using Python

Python Data Analyst Project

Import Libraries

[2]:
```
import pandas as pd
```

[4]:
```
# import csv file
df = pd.read_csv("/content/drive/MyDrive/Data Analysis/Python Project/food␣
↪delivery costs.csv")
```

** CSV File Details**

[9]:
```
df
```

[9]:
```
  Order ID Customer ID Restaurant ID Order Date and Time \
0          1       C8270        R2924 2024-02-01
                                     01:11:52
1          2       C1860        R2054 2024-02-02
                                     22:11:04
2          3       C6390        R2870 2024-01-31
                                     05:54:35
3          4       C6191        R2642 2024-01-16
                                     22:52:49
4          5       C6734        R2799 2024-01-29
                                     01:19:30
..       …       …          …                 …
995      996       C6232        R2129 2024-01-14
                                     05:57:00
996      997       C6797        R2742 2024-01-28
                                     08:50:43
997      998       C5926        R2837 2024-01-21
                                     09:43:19
998      999       C7016        R2144 2024-01-30
                                     22:23:38
999     1000       C4335        R2890 2024-01-08
                                     14:46:43
  Delivery Date and Time Order Value Delivery Fee Payment Method \
0 2024-02-01 02:39:52 1914 0 Credit Card 1 2024-02-02 22:46:04
986 40 Digital Wallet
2     2024-01-31 06:52:35   937   30 Cash on Delivery
3     2024-01-16 23:38:49   1463  50 Cash on Delivery
4     2024-01-29 02:48:30   1992  30 Cash on Delivery
..                    …         …          …              …
```

```
995   2024-01-14 06:39:00      825          0   Digital Wallet
996   2024-01-28 10:10:43     1627         50     Cash     on
                                                 Delivery
997   2024-01-21 10:44:19      553         20     Cash     on
                                                 Delivery
998   2024-01-31 00:07:38     1414          0     Cash     on
                                                  Delivery
999   2024-01-08 15:39:43     1657         20  Digital Wallet

    Discounts and Offers Commission Fee Payment Processing
    Fee \
0            5% on App           150                  47
1                  10%           198                  23
2          15% New User          195                  45
3                  NaN           146                  27
4          50 off Promo          130                  50
..                  …            …                    …
995          5% on App           165                  47
996                NaN           110                  42
997                NaN            64                  31
998        15% New User          199                  34
999        15% New User          180                  27

     Refunds/Chargebacks
0                     0
1                     0
2                     0
3                     0
4                     0
..                    …
995                  50
996                   0
997                   0
998                   0
999                 100

[1000 rows x 12 columns]
```

[6]:
```
df.shape
```

[6]: (1000, 12)

[11]:
```
df.head()  #first 5 rows
```

[11]:
```
Order ID Customer ID Restaurant ID Order Date and Time \
0        1  C8270 R2924 2024-02-01 01:11:52
1        2  C1860 R2054 2024-02-02 22:11:04
2        3  C6390 R2870 2024-01-31 05:54:35
3        4  C6191 R2642 2024-01-16 22:52:49
4        5  C6734 R2799 2024-01-29 01:19:30
Delivery Date and Time Order Value Delivery Fee Payment Method \
0 2024-02-01 02:39:52 1914 0 Credit Card 1 2024-02-02
22:46:04 986 40 Digital Wallet
2   2024-01-31 06:52:35    937   30 Cash on Delivery
3   2024-01-16 23:38:49    1463  50 Cash on Delivery
4   2024-01-29 02:48:30    1992  30 Cash on Delivery

Discounts and Offers Commission Fee Payment Processing Fee \
0          5% on App           150                  47
1                10%           198                  23
2       15% New User           195                  45
3                NaN           146                  27
4       50 off Promo           130                  50
   Refunds/Chargebacks
0                    0
1                    0
2                    0
3                    0
4                    0
```

[8]:
```
df.info() #data type
```

```
<class
'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to
999 Data columns (total 12
columns):
 #   Column              Non-Null Count Dtype
--- ------              -------------- -----
0   Order ID            1000 non-null int64
1   Customer ID         1000 non-null object
2   Restaurant ID       1000 non-null object
3  Order Date and Time  1000 non-null object
4 Delivery Date and Time 1000 non-nullobject
5   Order Value         1000 non-null int64
6   Delivery Fee        1000 non-null int64
7   Payment Method      1000 non-null object
8  Discounts and Offers 815 non-null  object
```

```
9   Commission Fee       1000 non-null int64
 10 Payment Processing Fee 1000 non-   int64
 null
 11  Refunds/Chargebacks  1000  non-   int64
null  dtypes:  int64(6),  object(6)
memory usage: 93.9+ KB
```

#Data Cleaning$

[12]:
```
df["Order Date and Time"] = pd.to_datetime(df['Order Date and
Time']) df.info()
```

```
<class
'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to
999 Data columns (total 12
columns):
 # Column Non-Null Count Dtype --- ------ ---
----------- -----
 0   Order ID            1000 non-nullint64
 1   Customer ID         1000 non-nullobject
 2   Restaurant ID       1000 non-nullobject
 3   Order Date and Time 1000 non-nulldatetime64[ns]
 4   Delivery Date and Time 1000 non- object
     null
 5   Order Value         1000 non-nullint64
 6   Delivery Fee        1000 non-nullint64
 7   Payment Method      1000 non-nullobject
 8   Discounts and Offers815 non-null object
 9   Commission Fee      1000 non-nullint64
 10 Payment Processing Fee 1000 non- int64
 null
 11 Refunds/Chargebacks 1000 non-nullint64
dtypes: datetime64[ns](1), int64(6),
object(5) memory usage: 93.9+ KB
```

[13]:
```
df["Delivery Date and Time"] = pd.to_datetime(df["Delivery Date
and Time"]) df.info()
```

```
<class
'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to
999 Data columns (total 12
columns):
 # Column Non-Null Count Dtype --- ------ ---
----------- -----
 0   Order ID            1000 non-nullint64
 1   Customer ID         1000 non-nullobject
 2   Restaurant ID       1000 non-nullobject
```

```
 3   Order Date and Time 1000 non-nulldatetime64[ns]
 4   Delivery Date and Time 1000 non- datetime64[ns]
     null
 5   Order Value         1000 non-nullint64
 6   Delivery Fee        1000 non-nullint64
 7   Payment Method      1000 non-nullobject
 8   Discounts and Offers815 non-null object
 9   Commission Fee      1000 non-nullint64
 10 Payment Processing Fee 1000 non-  int64
 null
 11 Refunds/Chargebacks  1000 non-nullint64
dtypes: datetime64[ns](2), int64(6),
object(4) memory usage: 93.9+ KB
```

[14]:
```
df.head()
```

[14]:
```
Order ID Customer ID Restaurant ID Order Date and Time \
0        1  C8270 R2924 2024-02-01 01:11:52
1        2  C1860 R2054 2024-02-02 22:11:04
2        3  C6390 R2870 2024-01-31 05:54:35
3        4  C6191 R2642 2024-01-16 22:52:49
4        5  C6734 R2799 2024-01-29 01:19:30

Delivery Date and Time Order Value Delivery Fee Payment Method \
0 2024-02-01 02:39:52 1914 0 Credit Card 1 2024-02-02
22:46:04 986 40 Digital Wallet
2   2024-01-31 06:52:35    937   30 Cash on Delivery
3   2024-01-16 23:38:49    1463  50 Cash on Delivery
4   2024-01-29 02:48:30    1992  30 Cash on Delivery

Discounts and Offers Commission Fee Payment Processing Fee \
0          5% on App           150                   47
1                10%           198                   23
2       15% New User           195                   45
3                NaN           146                   27
4       50 off Promo           130                   50
   Refunds/Chargebacks
0                    0
1                    0
2                    0
3                    0
```

```
4                    0
```

[15]:
```
def extract(value):
    a = str(value).split(" ")
    return a[0]

df["Discounts and Offers"] = df["Discounts and Offers"].apply(extract)
df.head()
```

[15]:
```
Order ID Customer ID Restaurant ID Order Date and Time \
0        1  C8270 R2924 2024-02-01 01:11:52
1        2  C1860 R2054 2024-02-02 22:11:04
2        3  C6390 R2870 2024-01-31 05:54:35
3        4  C6191 R2642 2024-01-16 22:52:49
4        5  C6734 R2799 2024-01-29 01:19:30

    Delivery Date and Time Order Value Delivery Fee Payment Method \
0 2024-02-01 02:39:52 1914 0 Credit Card 1 2024-02-02
22:46:04 986 40 Digital Wallet
2    2024-01-31 06:52:35    937   30 Cash on Delivery
3    2024-01-16 23:38:49    1463  50 Cash on Delivery
4    2024-01-29 02:48:30    1992  30 Cash on Delivery
   Discounts and Offers Commission Fee Payment Processing Fee \
0                 5%           150                   47
1                10%           198                   23
2                15%           195                   45
3                nan           146                   27
4                 50           130                   50
   Refunds/Chargebacks
0                    0
1                    0
2                    0
3                    0
4                    0
```

[16]:
```
def removep(value):
    if "%" in value:
        a = value.replace("%","")
        return float(a)
    else:
        return float(value)

df["Discounts and Offers"] = df["Discounts and Offers"].apply(removep)
df.head()
```

[16]:
```
Order ID Customer ID Restaurant ID Order Date and Time \
```

```
0         1  C8270 R2924 2024-02-01 01:11:52
1         2  C1860 R2054 2024-02-02 22:11:04
2         3  C6390 R2870 2024-01-31 05:54:35
3         4  C6191 R2642 2024-01-16 22:52:49
4         5  C6734 R2799 2024-01-29 01:19:30

Delivery Date and Time Order Value Delivery Fee Payment Method \
0 2024-02-01 02:39:52 1914 0 Credit Card 1 2024-02-02
22:46:04 986 40 Digital Wallet
2    2024-01-31 06:52:35    937   30 Cash on Delivery
3    2024-01-16 23:38:49    1463  50 Cash on Delivery
4    2024-01-29 02:48:30    1992  30 Cash on Delivery

   Discounts and Offers Commission Fee Payment Processing Fee
   \
0                  5.0           150                    47
1                 10.0           198                    23
2                 15.0           195                    45
3                  NaN           146                    27
4                 50.0           130                    50
   Refunds/Chargebacks
0                    0
1                    0
2                    0
3                    0
4                    0
```

[17]:
```
df.loc[(df["Discounts and Offers"] <= 15),"Discounts and Offers"
] =␣
 ↪(df["Discounts and Offers"]/100) * df["Order Value"]
df.head()
```

[17]:
```
Order ID Customer ID Restaurant ID Order Date and Time \
0         1  C8270 R2924 2024-02-01 01:11:52
1         2  C1860 R2054 2024-02-02 22:11:04
2         3  C6390 R2870 2024-01-31 05:54:35
3         4  C6191 R2642 2024-01-16 22:52:49
4         5  C6734 R2799 2024-01-29 01:19:30

Delivery Date and Time Order Value Delivery Fee Payment Method \
0 2024-02-01 02:39:52 1914 0 Credit Card 1 2024-02-02
22:46:04 986 40 Digital Wallet
2    2024-01-31 06:52:35    937   30 Cash on Delivery
3    2024-01-16 23:38:49    1463  50 Cash on Delivery
4    2024-01-29 02:48:30    1992  30 Cash on Delivery
```

```
   Discounts and Offers Commission Fee Payment Processing Fee
   \
0                 95.70                150                      47
1                 98.60                198                      23
2                140.55                195                      45
3                   NaN                146                      27
4                 50.00                130                      50
   Refunds/Chargebacks
0                    0
1                    0
2                    0
3                    0
4                    0
```

[22]:
```
df["Discounts and Offers"] = df["Discounts and Offers"].fillna(0)
df.head()
```

[22]:
```
Order ID Customer ID Restaurant ID Order Date and Time \
0        1  C8270 R2924 2024-02-01 01:11:52
1        2  C1860 R2054 2024-02-02 22:11:04
2        3  C6390 R2870 2024-01-31 05:54:35
3        4  C6191 R2642 2024-01-16 22:52:49
4        5  C6734 R2799 2024-01-29 01:19:30
Delivery Date and Time Order Value Delivery Fee Payment Method \
0 2024-02-01 02:39:52 1914 0 Credit Card 1 2024-02-02
22:46:04 986 40 Digital Wallet
2   2024-01-31 06:52:35    937   30 Cash on Delivery
3   2024-01-16 23:38:49    1463  50 Cash on Delivery
4   2024-01-29 02:48:30    1992  30 Cash on Delivery

Discounts and Offers Commission Fee Payment Processing Fee \
0                 95.70                150                      47
1                 98.60                198                      23
2                140.55                195                      45
3                  0.00                146                      27
4                 50.00                130                      50

  Refunds/Chargebacks Costs Profit
0                   0 142.70 7.30
1                   0 161.60 36.40
2                   0 215.55 -20.55
3                   0   NaN   NaN
4                   0 130.00 0.00
```

[23]:
```
df["Costs"] = df["Delivery Fee"] + df['Discounts and Offers'] +
df["Payment␣
```

```
 ↪Processing Fee"] #create new columns cost(finding cost)
df.head()
```

[23]:
```
Order ID Customer ID Restaurant ID Order Date and Time \
0        1  C8270 R2924 2024-02-01 01:11:52
1        2  C1860 R2054 2024-02-02 22:11:04
2        3  C6390 R2870 2024-01-31 05:54:35
3        4  C6191 R2642 2024-01-16 22:52:49
4        5  C6734 R2799 2024-01-29 01:19:30

Delivery Date and Time Order Value Delivery Fee Payment Method \
0 2024-02-01 02:39:52 1914 0 Credit Card 1 2024-02-02
22:46:04 986 40 Digital Wallet
2   2024-01-31 06:52:35    937   30 Cash on Delivery
3   2024-01-16 23:38:49    1463  50 Cash on Delivery
4   2024-01-29 02:48:30    1992  30 Cash on Delivery

Discounts and Offers Commission Fee Payment Processing Fee \
0                95.70          150                    47
1                98.60          198                    23
2               140.55          195                    45
3                0.00           146                    27
4                50.00          130                    50
  Refunds/Chargebacks Costs Profit
0                   0 142.70  7.30
1                   0 161.60  36.40
2                   0 215.55 -20.55
3                   0 77.00   NaN
4                   0 130.00  0.00
```

[24]:
```
df["Profit"] = df["Commission Fee"] - df['Costs'] # create new columns␣
 ↪profit(finding profit)
df.head()
```

[24]:
```
Order ID Customer ID Restaurant ID Order Date and Time \
0        1  C8270 R2924 2024-02-01 01:11:52
1        2  C1860 R2054 2024-02-02 22:11:04
2        3  C6390 R2870 2024-01-31 05:54:35
3        4  C6191 R2642 2024-01-16 22:52:49
4        5  C6734 R2799 2024-01-29 01:19:30

Delivery Date and Time Order Value Delivery Fee Payment Method \
0 2024-02-01 02:39:52 1914 0 Credit Card 1 2024-02-02
22:46:04 986 40 Digital Wallet
2   2024-01-31 06:52:35    937   30 Cash on Delivery
3   2024-01-16 23:38:49    1463  50 Cash on Delivery
```

```
4    2024-01-29 02:48:30    1992  30 Cash on Delivery

   Discounts and Offers Commission Fee Payment Processing Fee
   \
0                 95.70            150                    47
1                 98.60            198                    23
2                140.55            195                    45
3                 0.00             146                    27
4                50.00             130                    50
  Refunds/Chargebacks Costs  Profit
0                   0 142.70  7.30
1                   0 161.60  36.40
2                   0 215.55 -20.55
3                   0 77.00   69.00
4                   0 130.00  0.00
```

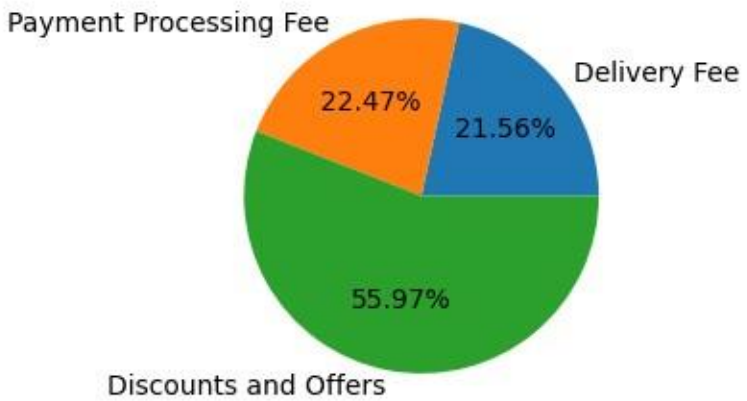
[25]:
```
df["Profit"].sum() # profit sum
```

[25]:
```
-5751.85
```

[26]:
```
cost_dist = df[["Delivery Fee", "Payment Processing Fee", "Discounts and␣
 ↪Offers"]].sum()
cost_dist
```

[26]:
```
Delivery Fee           28620.00
Payment Processing Fee  29832.00
Discounts and Offers    74289.85
 dtype: float64
```

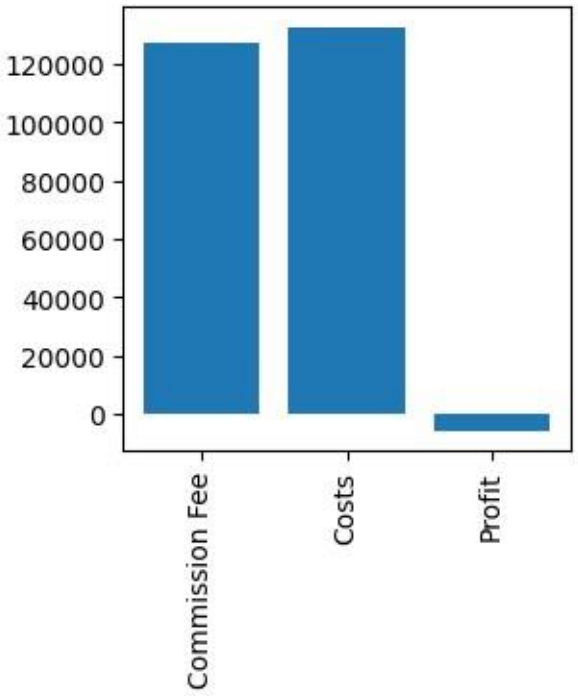
[27]:
```
import matplotlib.pyplot as plt

plt.figure(figsize = (3,3))
plt.pie(cost_dist, labels = cost_dist.index, autopct = "%1.2f%%")
plt.show()
```



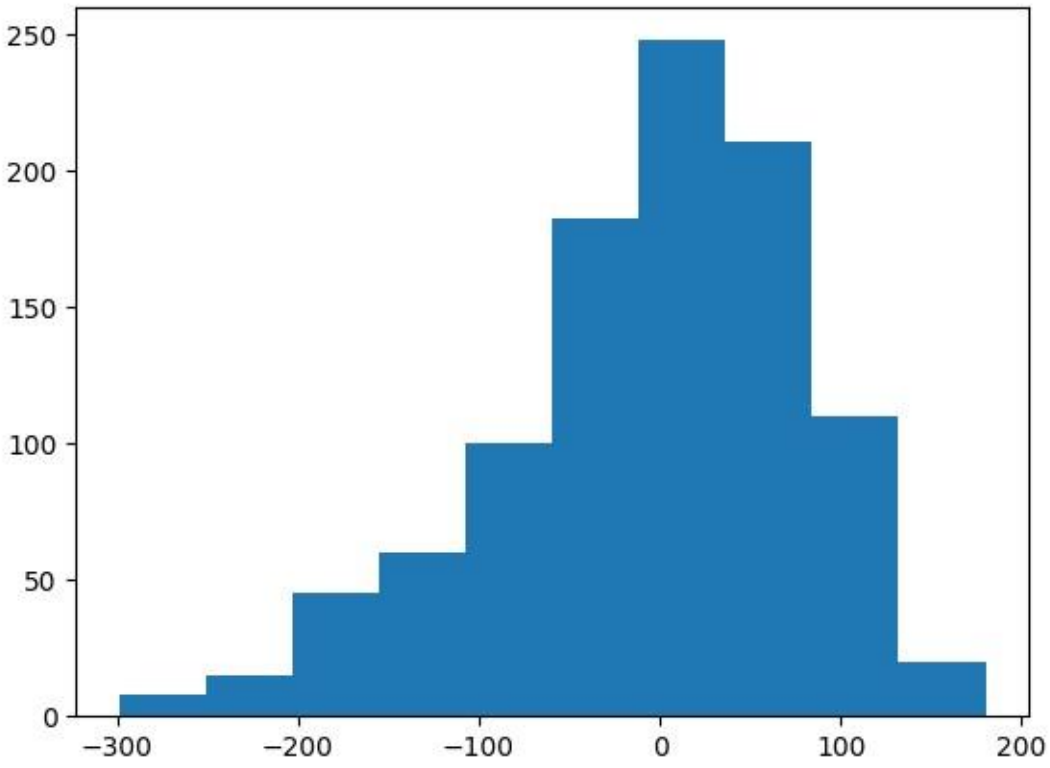
[28]:
```
abc = df[["Commission Fee", "Costs","Profit"]].sum()
abc
```

[28]:
```
Commission Fee    126990.00
Costs             132741.85
Profit             -5751.85
dtype: float64
```

[29]:
```
plt.figure(figsize = (3,3))
plt.bar(abc.index, abc)
plt.xticks(rotation = 90)
plt.show()
```



[30]:
```
plt.hist(df["Profit"])
plt.show()
```


250
200
150
100
50
0
−300
−200
−100
0
100
200

Project Summary

In the **Food Delivery Cost Analysis** project, Shaun Mia conducted a comprehensive analysis of a food delivery service's costs, discounts, and profits using Python. This project involved data cleaning, transformation, and visualization to extract valuable insights on delivery costs and profitability. Below is a detailed breakdown of the process and findings:

1. Data Import and Exploration

- The project began with importing the dataset containing food delivery details and loading it into a DataFrame.
- Basic exploration, including displaying the first few rows, data shape, and data types, helped in understanding the structure of the data.

2. Data Cleaning and Preprocessing

- **Date Formatting:** Converted "Order Date and Time" and "Delivery Date and Time" columns to datetime format to ensure consistent data types for time-based analysis.
- **Discount Parsing:** Extracted and standardized values in the "Discounts and Offers" column by removing percentage signs and converting strings to numeric values.
- **Missing Values:** Filled any null values in the "Discounts and Offers" column with 0 to maintain dataset integrity.

3. New Feature Creation

- **Total Costs Calculation:** Calculated the total cost per order by combining "Delivery Fee," "Discounts and Offers," and "Payment Processing Fee" into a new "Costs" column.
- **Profit Calculation:** Determined profitability by subtracting total costs from the "Commission Fee," storing the result in a new "Profit" column.

4. Analysis and Visualization

- **Cost Distribution:** Summed up individual cost components to analyze their distribution and visualized this with a pie chart, showing the proportion of delivery fees, payment processing fees, and discounts in the overall costs.
- **Profitability Breakdown:** Aggregated and visualized the "Commission Fee," "Costs," and "Profit" using bar charts, providing a clear comparison between revenue and expenses.
- **Profit Distribution:** Created a histogram of profits to show the distribution and variation in profitability across orders.

5. Key Insights

- The analysis successfully quantified costs and profits, helping to identify areas where cost-saving measures could increase profitability.
- The project highlighted the impact of discounts and processing fees on total delivery costs and the influence of commission fees on net profit.

Tools and Libraries Used

- **Python Libraries:** Pandas for data manipulation, Matplotlib for visualizations.
- **Google Colab Integration:** Drive-mounted CSV data enabled efficient data handling and processing.

This project demonstrates Shaun Mia's ability to use Python for data analysis, particularly in cleaning, transforming, and visualizing large datasets to derive insights on cost and profitability in the food delivery sector.

Shaun Mia | LinkedIn