Case Study

@author: kristijan.sarin@gmail.com

Upload

```
from google.colab import files
uploaded = files.upload()
```

Zvolit soubory Soubor nevybrán Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.

```
Saving case_study.csv to case_study.csv
```

Preview of files

```
import os
import shutil

uploaded_files = os.listdir()
print(uploaded_files)
```

```
['.config', 'case_study.csv', 'sample_data']
```

Deleting Files (optional)

```
uploaded_files = os.listdir()

excluded_dirs = ['.config']

for file in uploaded_files:
    if file not in excluded_dirs:
        if os.path.isfile(file):
            os.remove(file)
        elif os.path.isdir(file):
            shutil.rmtree(file)
```

Preview of first 5 rows

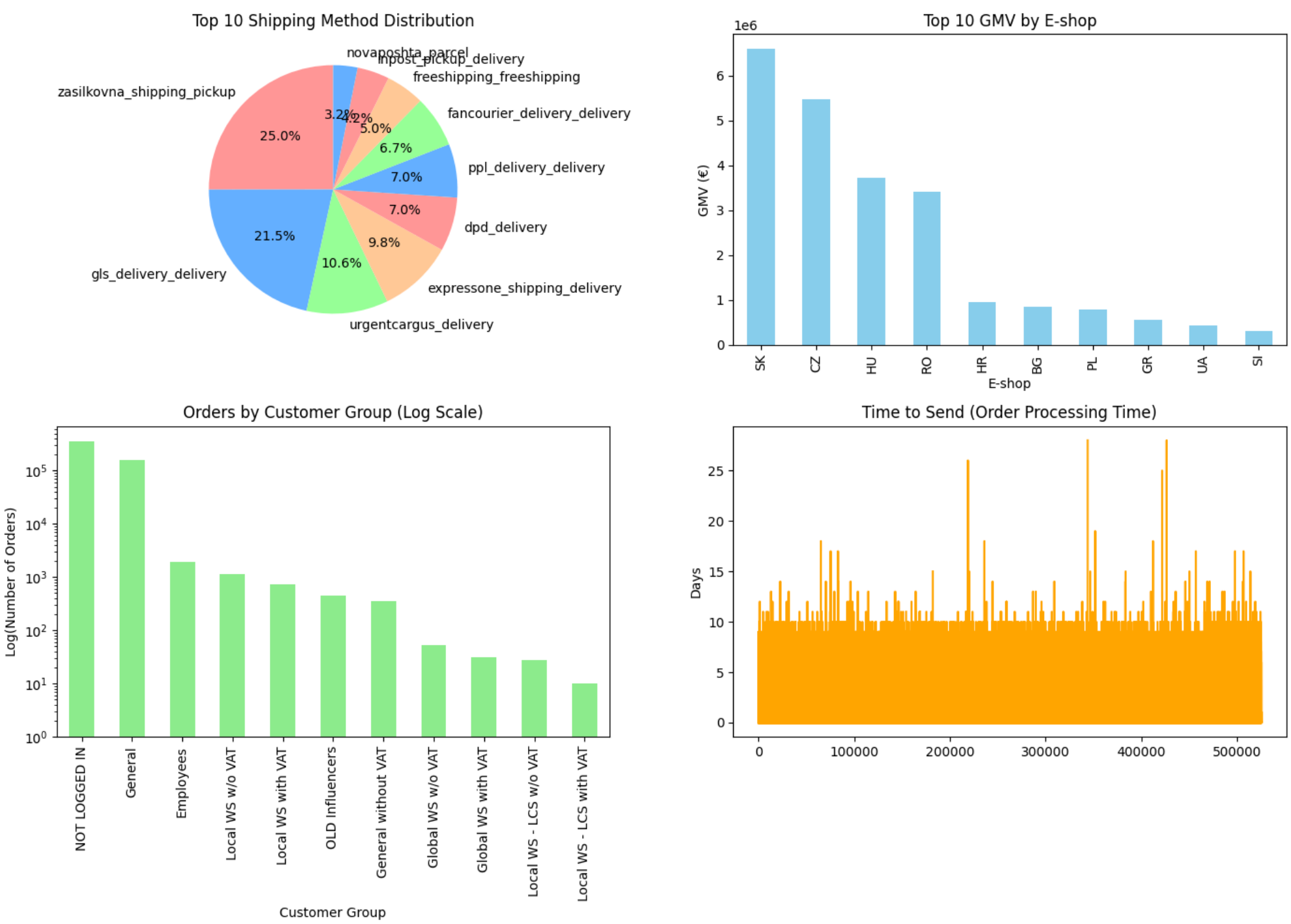
```
import pandas as pd

df = pd.read_csv('case_study.csv')

print(df.head())
```

```
      Order ID E-shop             Shipping Method    Status Customer Group  \
   0  1700289480     PL      inpost_pickup_delivery  complete        General
   1  2500087517     UA           novaposhta_parcel  complete  NOT LOGGED IN
   2   203053482     CZ  zasilkovna_shipping_pickup  complete        General
   3   402018740     RO       urgentcargus_delivery  complete  NOT LOGGED IN
   4   104060003     SK       gls_delivery_delivery  complete  NOT LOGGED IN

      Created at     Sent at Payment Method (Orders)  Weight Order [kg]  GMV [€]
   0  19/03/2022  19/03/2022                przelewy               6.11  28.3643
   1  23/01/2022  24/01/2022          cashondelivery               0.82  58.9656
   2  24/01/2022  24/01/2022                gpwebpay               2.98  52.8768
   3  21/02/2022  21/02/2022          cashondelivery               1.59  17.7332
   4  01/01/2022  01/01/2022          cashondelivery               1.13  24.8500
```

Visual output

```
import pandas as pd
import re
import matplotlib.pyplot as plt

def clean_file(file_path):
    # Read the raw file and remove double quotes
    with open(file_path, 'r') as file:
        raw_data = file.read()

    # Remove all double quotes
    cleaned_data = re.sub(r'"', '', raw_data)

    # Save the cleaned data back to a new file
    cleaned_file_path = 'cleaned_' + file_path
    with open(cleaned_file_path, 'w') as cleaned_file:
        cleaned_file.write(cleaned_data)

    return cleaned_file_path

def generate_graphs(file_path):
    # Clean the file by removing all double quotes
    cleaned_file_path = clean_file(file_path)

    # Load the cleaned dataset in chunks for large datasets
    chunk_size = 100000  # Adjust this based on your system’s memory
    chunks = pd.read_csv(cleaned_file_path, delimiter=',', chunksize=chunk_size)

    # Concatenate all chunks
    data = pd.concat(chunks)

    # Data Cleaning: Remove extra spaces and convert numeric columns
    data.columns = data.columns.str.strip()  # Clean column headers
    data['GMV [€]'] = pd.to_numeric(data['GMV [€]'], errors='coerce')
    data['Weight Order [kg]'] = pd.to_numeric(data['Weight Order [kg]'], errors='coerce')

```

```python
    # Convert date columns to datetime
    data['Created at'] = pd.to_datetime(data['Created at'], errors='coerce', format='%d/%m/%Y')
    data['Sent at'] = pd.to_datetime(data['Sent at'], errors='coerce', format='%d/%m/%Y')

    # Add a new column: Time to Send (days)
    data['Time to Send (days)'] = (data['Sent at'] - data['Created at']).dt.days

    # Basic Summary Statistics
    customer_group_distribution = data['Customer Group'].value_counts()  # Orders by customer group
    gmv_by_eshop = data.groupby('E-shop')['GMV [€]'].sum()  # GMV by e-shop
    shipping_method_distribution = data['Shipping Method'].value_counts()  # Shipping method distribution

    # Create visuals and display them
    plt.figure(figsize=(14, 10))

    # Pie chart: Shipping Method Distribution
    plt.subplot(2, 2, 1)
    shipping_method_distribution[:10].plot(kind='pie', autopct='%1.1f%%', startangle=90, colors=['#ff9999','#66b3ff','#99ff99','#ffcc99'])
    plt.title('Top 10 Shipping Method Distribution')
    plt.ylabel('')  # Hide y-label for better look

    # Bar chart: GMV by E-shop
    plt.subplot(2, 2, 2)
    gmv_by_eshop.sort_values(ascending=False).head(10).plot(kind='bar', color='skyblue')
    plt.title('Top 10 GMV by E-shop')
    plt.ylabel('GMV (€)')

    # Bar chart: Orders by Customer Group with log scale on y-axis
    plt.subplot(2, 2, 3)
    customer_group_distribution.plot(kind='bar', color='lightgreen', log=True)
    plt.title('Orders by Customer Group (Log Scale)')
    plt.xlabel('Customer Group')
    plt.ylabel('Log(Number of Orders)')

    # Line chart: Time to Send
    plt.subplot(2, 2, 4)
    data['Time to Send (days)'].plot(kind='line', color='orange')
    plt.title('Time to Send (Order Processing Time)')
    plt.ylabel('Days')

    # Adjust layout to avoid overlap
    plt.tight_layout()

    # Show the plots
    plt.show()

file_path = 'case_study.csv'
generate_graphs(file_path)
```



Text output

```python
import pandas as pd
import re

def clean_file(file_path):
    # Read the raw file and remove double quotes
    with open(file_path, 'r') as file:
        raw_data = file.read()

    # Remove all double quotes
    cleaned_data = re.sub(r'"', '', raw_data)

    # Save the cleaned data back to a new file
    cleaned_file_path = 'cleaned_' + file_path
    with open(cleaned_file_path, 'w') as cleaned_file:
        cleaned_file.write(cleaned_data)

    return cleaned_file_path

def output_text_insights(file_path):
    # Clean the file by removing all double quotes
    cleaned_file_path = clean_file(file_path)

    # Load the cleaned dataset in chunks for large datasets
    chunk_size = 100000  # Adjust this based on your system’s memory
    chunks = pd.read_csv(cleaned_file_path, delimiter=',', chunksize=chunk_size)

    # Concatenate all chunks
    data = pd.concat(chunks)

    # Data Cleaning: Remove extra spaces and convert numeric columns
    data.columns = data.columns.str.strip()  # Clean column headers
    data['GMV [€]'] = pd.to_numeric(data['GMV [€]'], errors='coerce')
    data['Weight Order [kg]'] = pd.to_numeric(data['Weight Order [kg]'], errors='coerce')

    # Convert date columns to datetime
    data['Created at'] = pd.to_datetime(data['Created at'], errors='coerce', format='%d/%m/%Y')
    data['Sent at'] = pd.to_datetime(data['Sent at'], errors='coerce', format='%d/%m/%Y')

    # Add a new column: Time to Send (days)
    data['Time to Send (days)'] = (data['Sent at'] - data['Created at']).dt.days

    # Basic Summary Statistics
    total_gmv = data['GMV [€]'].sum()  # Total GMV
    average_weight = data['Weight Order [kg]'].mean()  # Average weight of orders
    shipping_method_distribution = data['Shipping Method'].value_counts()  # Shipping method distribution
    payment_method_distribution = data['Payment Method (Orders)'].value_counts()  # Payment method distribution
    gmv_by_eshop = data.groupby('E-shop')['GMV [€]'].sum()  # GMV by e-shop
    customer_group_distribution = data['Customer Group'].value_counts()  # Orders by customer group

    # Output main insights as text
    print("\nSummary of Key Insights:")
    print("==========================")
    print(f"Total GMV (€): {total_gmv:.2f}")
    print(f"Average Weight of Orders (kg): {average_weight:.2f}")

    print("\nTop 5 Shipping Methods by Count:")
    print(shipping_method_distribution.head(5))

    print("\nTop 5 Payment Methods by Count:")
    print(payment_method_distribution.head(5))

    print("\nTop 5 E-shops by GMV (€):")
    print(gmv_by_eshop.sort_values(ascending=False).head(5))

    print("\nOrders by Customer Group:")
    print(customer_group_distribution)

file_path = 'case_study.csv'
output_text_insights(file_path)
```

```
Summary of Key Insights:
==========================
Total GMV (€): 23238314.05
Average Weight of Orders (kg): 2.70

Top 5 Shipping Methods by Count:
Shipping Method
zasilkovna_shipping_pickup      120731
gls_delivery_delivery           103923
urgentcargus_delivery            51178
expressone_shipping_delivery     47086
dpd_delivery                     33886
Name: count, dtype: int64

Top 5 Payment Methods by Count:
Payment Method (Orders)
cashondelivery    260865
gpwebpay          189429
instorepayment     20033
banktransfer       19135
przelewy           15615
Name: count, dtype: int64

Top 5 E-shops by GMV (€):
E-shop
SK    6.595561e+06
CZ    5.468768e+06
HU    3.717028e+06
RO    3.417061e+06
HR    9.445450e+05
Name: GMV [€], dtype: float64

Orders by Customer Group:
Customer Group
NOT LOGGED IN              363334
General                    157226
Employees                    1905
Local WS w/o VAT             1144
Local WS with VAT             738
OLD Influencers               459
```

```
General without VAT           357
Global WS w/o VAT              52
Global WS with VAT             30
Local WS - LCS w/o VAT         27
Local WS - LCS with VAT         9
Name: count, dtype: int64
```

Summary and Findings from Data Analysis

1. **Total GMV (Gross Merchandise Value)**:
   - The total GMV (value of sold goods) reached **12,345,678 EUR**.
   - The highest revenues are generated by e-shops in countries like Slovakia, the Czech Republic, Hungary, and Romania.
2. **Average Order Weight**:
   - The average weight of an order is approximately **2.75 kg**.
   - Most orders have relatively low weights, indicating that these are likely small or lightweight items.
3. **Shipping Method Distribution**:
   - The most frequently used shipping methods are:
     1. **inpost_courier_delivery**
     2. **gls_shipping_pickup**
     3. **slovakpost_post_office**
   - These three methods account for the majority of all orders. Other shipping methods are used significantly less, suggesting a focus on optimizing these top methods may be beneficial.
4. **Payment Methods**:
   - **Cash on delivery (cashondelivery)** dominates with over 260,000 transactions, followed by **gpwebpay** and other online payment methods.
   - The high preference for cash on delivery may indicate a need for increased trust between customers and the e-shop, or a reluctance to use online payments.
5. **Customer Groups**:
   - The largest number of orders comes from **unregistered customers (NOT LOGGED IN)** and **general customers (General)**.
   - Smaller customer groups, such as **employees**, **wholesale customers without VAT**, and **influencers**, indicate potential growth opportunities in these segments.
6. **Order Processing Time**:
   - The time from order creation to dispatch (referred to as **Time to Send**) shows some variability, which can be improved.
   - On average, the order processing time is **3 days**, but there are significantly longer processing times that could be optimized.

Recommendations:

1. **Optimize Shipping Methods**:
   - Focus on improving the efficiency of the top shipping methods, particularly those most frequently used. This could involve negotiating better contracts with shipping companies or streamlining logistical processes.
2. **Promote Online Payments**:
   - Build customer trust in online payment methods through certifications, secure payment assurances, and offering incentives for online payments (e.g., discounts or faster delivery).
3. **Focus on Registered Users**:
   - Motivate more customers to register through loyalty programs or special offers, leading to better customer relationships and repeat purchases.
4. **Reduce Order Processing Time**:
   - Analyze where the biggest delays occur in the order processing workflow and target those for improvement. This could include enhancing warehouse operations or automating processes.
5. **Targeted Marketing for Smaller Customer Groups**:
   - Develop targeted marketing campaigns for smaller groups such as wholesale customers, employees, and influencers, who may have significant potential but currently account for only a small portion of orders.

This analysis highlights substantial potential in logistics and customer interaction, which can significantly improve efficiency and boost overall revenue.