[71]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.dates as mdates
```

0.0.1 Handling Sales Data

[74]:
```
# Load the sales data file into a Pandas DataFrame
file_path = r"C:\Users\marsh\Downloads\Sales Data send.xlsx"
excel_data = pd.ExcelFile(file_path)
excel_data.sheet_names
```

[74]:
```
['Sales in Germany']
```

[76]:
```
# Load the specific sheet into a DataFrame
sales_data = pd.read_excel(file_path, sheet_name='Sales in␣
↪Germany')
sales_data.head()
```

[76]:
```
  Unnamed: 0    Unnamed: 1     Unnamed: 2 Fiscal year  \
0        NaN  Sales Region  Product Group      Period
1      Sales       Germany  Service Parts       in T€
2      Sales       Germany      Equipment       in T€

   Apr.- March, 4 special periods 2015  Apr.- March, 4␣
↪special periods 2015.1  \
0                                1.00000                       ␣
↪       2.000000
1                             3486.92643                       ␣
↪    3891.286785
2                            21505.10808                       ␣
↪   11021.833120

   Apr.- March, 4 special periods 2015.2  \
0                                3.000000
1                             3505.161615
2                            14929.519440

   Apr.- March, 4 special periods 2015.3  \
0                                4.00000
1                             4081.06413
2                            27340.89790

   Apr.- March, 4 special periods 2015.4  \
0                                5.00000
1                             3141.36408
2                             6405.08770

   Apr.- March, 4 special periods 2015.5  ...  \
0                                6.000000  ...
1                             3909.991755  ...
2                            12347.291420  ...

   Apr.- March, 4 special periods 2024.4  \
0                                5.000000
```

```
2                              6962.683460

   Apr.- March, 4 special periods 2024.8  \
0                                  9.00000
1                               9391.23741
2                              24785.57720

   Apr.- March, 4 special periods 2024.9  \
0                                 10.000000
1                               8337.453555
2                              13410.347660

  Apr.- March, 4 special periods 2024.10  \
0                                 11.00000
1                                7622.74473
2                               15705.21810

   Apr.- March, 4 special periods 2024.11  \
0                                 12.00000
1                                8251.80465
2                                6379.59928

   Apr.- March, 4 special periods 2024.12  \
0                                   Result
1                                99532.41369
2                                124783.1914

   Apr.- March, 4 special periods 2024.13
0                                       NaN
1                                99532.41369
2                               124783.19140

[3 rows x 144 columns]
```

[78]:
```
data = sales_data.drop(columns=['Unnamed: 0', 'Unnamed:␣
↪1','Fiscal year'])
```

[80]:
```
data = data.set_index('Unnamed: 2')
data.head()
```

[80]:
```
                Apr.- March, 4 special periods 2015  \
Unnamed: 2
Product Group                                1.00000
Service Parts                             3486.92643
Equipment                                21505.10808

                Apr.- March, 4 special periods 2015.1  \
```

```
Unnamed: 2
Product Group                                2.000000
Service Parts                             3891.286785
Equipment                                11021.833120

              Apr.- March, 4 special periods 2015.2  \
Unnamed: 2
Product Group                                3.000000
Service Parts                             3505.161615
Equipment                                14929.519440

              Apr.- March, 4 special periods 2015.3  \
Unnamed: 2
Product Group                                4.00000
Service Parts                             4081.06413
Equipment                                27340.89790

              Apr.- March, 4 special periods 2015.4  \
Unnamed: 2
Product Group                                5.00000
Service Parts                             3141.36408
Equipment                                 6405.08770

              Apr.- March, 4 special periods 2015.5  \
Unnamed: 2
Product Group                                6.000000
Service Parts                             3909.991755
Equipment                                12347.291420

              Apr.- March, 4 special periods 2015.6  \
Unnamed: 2
Product Group                                7.000000
Service Parts                             4252.596825
Equipment                                24759.978660

              Apr.- March, 4 special periods 2015.7  \
Unnamed: 2
Product Group                                8.00000
Service Parts                             3743.15235
Equipment                                 1299.92614

              Apr.- March, 4 special periods 2015.8  \
Unnamed: 2
Product Group                                9.00000
Service Parts                             3603.48081
Equipment                                16589.25176
```

```
              Apr.- March, 4 special periods 2015.9  ...  \
Unnamed: 2                                                 ...
Product Group                              10.00000  ...
Service Parts                            4068.55110  ...
Equipment                               12845.38168  ...

              Apr.- March, 4 special periods 2024.4  \
Unnamed: 2
Product Group                              5.000000
Service Parts                            6573.046035
Equipment                                6513.088620

              Apr.- March, 4 special periods 2024.5  \
Unnamed: 2
Product Group                              6.000000
Service Parts                            9343.440705
Equipment                                 218.219600

             Apr.- March, 4 special periods 2024.6  \
Unnamed: 2
Product Group                              7.000000
Service Parts                            8703.563625
Equipment                                 175.799120

              Apr.- March, 4 special periods 2024.7  \
Unnamed: 2
Product Group                              8.000000
Service Parts                            8383.180125
Equipment                                6962.683460

              Apr.- March, 4 special periods 2024.8  \
Unnamed: 2
Product Group                              9.00000
Service Parts                            9391.23741
Equipment                               24785.57720

              Apr.- March, 4 special periods 2024.9  \
Unnamed: 2
Product Group                             10.000000
Service Parts                            8337.453555
Equipment                               13410.347660

              Apr.- March, 4 special periods 2024.10  \
Unnamed: 2
Product Group                             11.00000
Service Parts                            7622.74473
Equipment                               15705.21810
```

```
               Apr.- March, 4 special periods 2024.11  \
Unnamed: 2
Product Group                                 12.00000
Service Parts                                8251.80465
Equipment                                    6379.59928

               Apr.- March, 4 special periods 2024.12  \
Unnamed: 2
Product Group                                   Result
Service Parts                               99532.41369
Equipment                                   124783.1914

               Apr.- March, 4 special periods 2024.13
Unnamed: 2
Product Group                                      NaN
Service Parts                               99532.41369
Equipment                                  124783.19140

[3 rows x 140 columns]
```

[82]:
```
data.index.name = None
data.head()
```

[82]:
```
               Apr.- March, 4 special periods 2015  \
Product Group                              1.00000
Service Parts                            3486.92643
Equipment                               21505.10808

               Apr.- March, 4 special periods 2015.1  \
Product Group                                2.000000
Service Parts                              3891.286785
Equipment                                 11021.833120

               Apr.- March, 4 special periods 2015.2  \
Product Group                                3.000000
Service Parts                              3505.161615
Equipment                                 14929.519440

               Apr.- March, 4 special periods 2015.3  \
Product Group                                4.00000
Service Parts                              4081.06413
Equipment                                 27340.89790

               Apr.- March, 4 special periods 2015.4  \
Product Group                                5.00000
Service Parts                              3141.36408
```

```
Equipment                                  6405.08770

               Apr.- March, 4 special periods 2015.5  \
Product Group                                 6.000000
Service Parts                              3909.991755
Equipment                                 12347.291420

               Apr.- March, 4 special periods 2015.6  \
Product Group                                 7.000000
Service Parts                              4252.596825
Equipment                                 24759.978660

               Apr.- March, 4 special periods 2015.7  \
Product Group                                 8.00000
Service Parts                              3743.15235
Equipment                                  1299.92614

               Apr.- March, 4 special periods 2015.8  \
Product Group                                 9.00000
Service Parts                              3603.48081
Equipment                                 16589.25176

               Apr.- March, 4 special periods 2015.9  ...  \
Product Group                                10.00000  ...
Service Parts                                4068.55110  ...
Equipment                                   12845.38168  ...

               Apr.- March, 4 special periods 2024.4  \
Product Group                                 5.000000
Service Parts                              6573.046035
Equipment                                  6513.088620

               Apr.- March, 4 special periods 2024.5  \
Product Group                                 6.000000
Service Parts                              9343.440705
Equipment                                   218.219600

              Apr.- March, 4 special periods 2024.6  \
Product Group                                 7.000000
Service Parts                              8703.563625
Equipment                                   175.799120

               Apr.- March, 4 special periods 2024.7  \
Product Group                                 8.000000
Service Parts                              8383.180125
Equipment                                  6962.683460
```

```
               Apr.- March, 4 special periods 2024.8  \
Product Group                                9.00000
Service Parts                             9391.23741
Equipment                                24785.57720

               Apr.- March, 4 special periods 2024.9  \
Product Group                               10.000000
Service Parts                             8337.453555
Equipment                                13410.347660

               Apr.- March, 4 special periods 2024.10  \
Product Group                                11.00000
Service Parts                              7622.74473
Equipment                                 15705.21810

               Apr.- March, 4 special periods 2024.11  \
Product Group                                12.00000
Service Parts                              8251.80465
Equipment                                  6379.59928

               Apr.- March, 4 special periods 2024.12  \
Product Group                                  Result
Service Parts                             99532.41369
Equipment                                124783.1914

               Apr.- March, 4 special periods 2024.13
Product Group                                     NaN
Service Parts                             99532.41369
Equipment                                124783.19140

[3 rows x 140 columns]
```

[84]:
```
data = data.loc[:, ~(data.iloc[0] == 'Result') & ~(data.
↪iloc[0].isnull())]
data.head()
```

[84]:
```
               Apr.- March, 4 special periods 2015  \
Product Group                                1.00000
Service Parts                             3486.92643
Equipment                                21505.10808

               Apr.- March, 4 special periods 2015.1  \
Product Group                                2.000000
Service Parts                             3891.286785
Equipment                                11021.833120

               Apr.- March, 4 special periods 2015.2  \
```

```
Product Group                                3.000000
Service Parts                             3505.161615
Equipment                                14929.519440

              Apr.- March, 4 special periods 2015.3  \
Product Group                                4.00000
Service Parts                             4081.06413
Equipment                                27340.89790

              Apr.- March, 4 special periods 2015.4  \
Product Group                                5.00000
Service Parts                             3141.36408
Equipment                                 6405.08770

              Apr.- March, 4 special periods 2015.5  \
Product Group                                6.000000
Service Parts                             3909.991755
Equipment                                12347.291420

              Apr.- March, 4 special periods 2015.6  \
Product Group                                7.000000
Service Parts                             4252.596825
Equipment                                24759.978660

              Apr.- March, 4 special periods 2015.7  \
Product Group                                8.00000
Service Parts                             3743.15235
Equipment                                 1299.92614

              Apr.- March, 4 special periods 2015.8  \
Product Group                                9.00000
Service Parts                             3603.48081
Equipment                                16589.25176

              Apr.- March, 4 special periods 2015.9  ...  \
Product Group                               10.00000  ...
Service Parts                             4068.55110  ...
Equipment                                12845.38168  ...

              Apr.- March, 4 special periods 2024.2  \
Product Group                                3.000000
Service Parts                             8898.904395
Equipment                                16962.472360

              Apr.- March, 4 special periods 2024.3  \
Product Group                                4.000000
Service Parts                             7688.775105
```

```
Equipment                                5967.756080

              Apr.- March, 4 special periods 2024.4  \
Product Group                                5.000000
Service Parts                             6573.046035
Equipment                                 6513.088620

              Apr.- March, 4 special periods 2024.5  \
Product Group                                6.000000
Service Parts                             9343.440705
Equipment                                  218.219600

              Apr.- March, 4 special periods 2024.6  \
Product Group                                7.000000
Service Parts                             8703.563625
Equipment                                  175.799120

              Apr.- March, 4 special periods 2024.7  \
Product Group                                8.000000
Service Parts                             8383.180125
Equipment                                 6962.683460

              Apr.- March, 4 special periods 2024.8  \
Product Group                                9.00000
Service Parts                              9391.23741
Equipment                                 24785.57720

              Apr.- March, 4 special periods 2024.9  \
Product Group                               10.000000
Service Parts                             8337.453555
Equipment                                13410.347660

              Apr.- March, 4 special periods 2024.10  \
Product Group                                11.00000
Service Parts                              7622.74473
Equipment                                 15705.21810

              Apr.- March, 4 special periods 2024.11
Product Group                                12.00000
Service Parts                              8251.80465
Equipment                                  6379.59928

[3 rows x 120 columns]
```

[86]:
```
# Set all column names to None (removes headers)
data.columns = [None] * len(data.columns)
data.head()
```

[86]:

```
                        None          None          None          ␣
↪ None  \
Product Group       1.00000      2.000000      3.000000      4.
↪00000
Service Parts    3486.92643   3891.286785   3505.161615   4081.
↪06413
Equipment       21505.10808  11021.833120  14929.519440  27340.
↪89790

                        None          None          None          ␣
↪None  \
Product Group       5.00000      6.000000      7.000000      8.
↪00000
Service Parts    3141.36408   3909.991755   4252.596825  3743.
↪15235
Equipment       6405.08770  12347.291420  24759.978660  1299.
↪92614

                        None          None  ...           None    ␣
↪     None  \
Product Group       9.00000     10.00000  ...       3.000000    ␣
↪ 4.000000
Service Parts    3603.48081   4068.55110  ...   8898.904395 ␣
↪7688.775105
Equipment       16589.25176  12845.38168  ...  16962.472360 ␣
↪5967.756080

                        None          None          None          ␣
↪None  \
Product Group       5.000000     6.000000      7.000000      8.
↪000000
Service Parts    6573.046035  9343.440705  8703.563625  8383.
↪180125
Equipment       6513.088620   218.219600   175.799120  6962.
↪683460

                        None          None          None          ␣
↪None
Product Group       9.00000     10.000000      11.00000    12.
↪00000
Service Parts    9391.23741   8337.453555   7622.74473  8251.
↪80465
Equipment       24785.57720  13410.347660  15705.21810  6379.
↪59928

[3 rows x 120 columns]
```

[88]:
```
data_transposed = data.T
data_transposed = pd.DataFrame(data_transposed)
data_transposed.head()
```

[88]:

	Product Group	Service Parts	Equipment
None	1.0	3486.926430	21505.10808
None	2.0	3891.286785	11021.83312
None	3.0	3505.161615	14929.51944
None	4.0	4081.064130	27340.89790
None	5.0	3141.364080	6405.08770

[90]:
```
data_transposed['Service Parts'] = data_transposed['Service␣
↪Parts'].map(lambda x: '{:.1f}'.format(x))
data_transposed['Equipment'] = data_transposed['Equipment'].
↪map(lambda x: '{:.1f}'.format(x))
```

[92]:
```
data_transposed.head()
```

[92]:

	Product Group	Service Parts	Equipment
None	1.0	3486.9	21505.1
None	2.0	3891.3	11021.8
None	3.0	3505.2	14929.5
None	4.0	4081.1	27340.9
None	5.0	3141.4	6405.1

[94]:
```
# Generate a list of month-year values starting from April␣
↪2015
start_date = pd.to_datetime('2015-04-01')
num_rows = len(data_transposed)

# Create a date range starting from April 2015, with monthly␣
↪frequency
date_range = pd.date_range(start=start_date,␣
↪periods=num_rows, freq='MS').strftime('%b-%Y')

# Insert the generated month-year values as the first column
data_transposed.insert(0, 'Month-Year', date_range)

# Display the result
data_transposed.head()
```

[94]:

	Month-Year	Product Group	Service Parts	Equipment
None	Apr-2015	1.0	3486.9	21505.1
None	May-2015	2.0	3891.3	11021.8
None	Jun-2015	3.0	3505.2	14929.5
None	Jul-2015	4.0	4081.1	27340.9
None	Aug-2015	5.0	3141.4	6405.1

[96]:
```
data_transposed.tail()
```

[96]:
```
     Month-Year  Product Group Service Parts Equipment
None   Nov-2024            8.0        8383.2    6962.7
None   Dec-2024            9.0        9391.2   24785.6
None   Jan-2025           10.0        8337.5   13410.3
None   Feb-2025           11.0        7622.7   15705.2
None   Mar-2025           12.0        8251.8    6379.6
```

[98]:
```
del data_transposed['Product Group']
data_transposed.head()
```

[98]:
```
     Month-Year Service Parts Equipment
None   Apr-2015        3486.9   21505.1
None   May-2015        3891.3   11021.8
None   Jun-2015        3505.2   14929.5
None   Jul-2015        4081.1   27340.9
None   Aug-2015        3141.4    6405.1
```

[100]:
```
data_transposed.set_index('Month-Year', inplace=True)
data_transposed.head()
```

[100]:
```
           Service Parts Equipment
Month-Year
Apr-2015          3486.9   21505.1
May-2015          3891.3   11021.8
Jun-2015          3505.2   14929.5
Jul-2015          4081.1   27340.9
Aug-2015          3141.4    6405.1
```

[102]:
```
data_transposed.describe()
```

[102]:
```
       Service Parts Equipment
count            120       120
unique           120       120
top           3486.9   21505.1
freq               1         1
```

0.0.2 Handling the missing and negative values

[105]:
```
data_transposed['Equipment'] = pd.
↪to_numeric(data_transposed['Equipment'], errors='coerce')
data_transposed['Equipment'] = data_transposed['Equipment'].
↪fillna(0)
```

```
data_transposed['Service Parts'] = pd.
 ↪to_numeric(data_transposed['Service Parts'],␣
 ↪errors='coerce')
data_transposed['Service Parts'] = data_transposed['Service␣
 ↪Parts'].fillna(0)
```

[107]:
```
print(data_transposed[data_transposed['Equipment'] <= 0])
print('\n')
print(data_transposed[data_transposed['Service Parts'] <= 0])
```

```
            Service Parts  Equipment
Month-Year
May-2019           8638.6        0.0
Jan-2023           9774.5     -460.0


Empty DataFrame
Columns: [Service Parts, Equipment]
Index: []
```

[109]:
```
def replace_with_neighbor_mean(series):
    series = series.tolist()  # Convert to a list for easier␣
 ↪iteration
    for i in range(len(series)):
        if series[i] <= 0:  # Check for zero or negative␣
 ↪values
            # Handle edge cases for the first and last␣
 ↪elements
            if i == 0:
                series[i] = series[i+1]  # Use the next value␣
 ↪for the first element
            elif i == len(series) - 1:
                series[i] = series[i-1]  # Use the previous␣
 ↪value for the last element
            else:
                # Replace with the mean of the neighboring␣
 ↪values
                series[i] = (series[i-1] + series[i+1]) / 2
    return series

data_transposed['Equipment'] =␣
 ↪replace_with_neighbor_mean(data_transposed['Equipment'])
```

[111]:
```
print(data_transposed[data_transposed['Equipment'] <= 0])  #␣
 ↪Should show an empty DataFrame
```

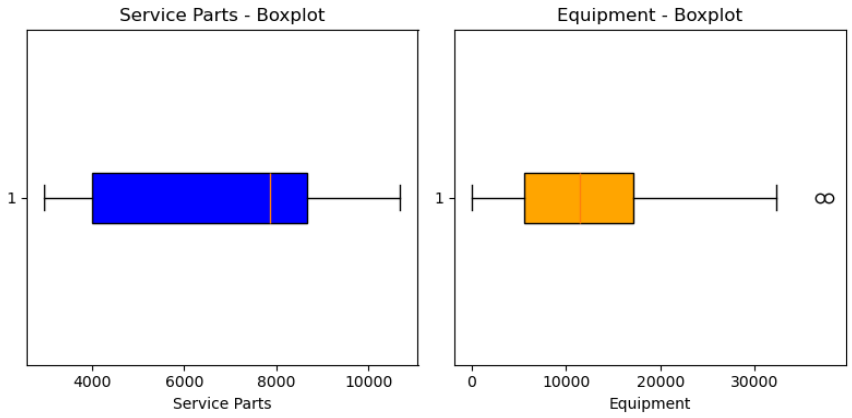
```
Empty DataFrame
Columns: [Service Parts, Equipment]
Index: []
```

0.0.3 Handling Outliers

[114]:
```
# Plot boxplots to visualize outliers
plt.figure(figsize=(8, 4))
# Service Parts Boxplot
plt.subplot(1, 2, 1)
plt.boxplot(data_transposed['Service Parts'], vert=False,␣
↪patch_artist=True, boxprops=dict(facecolor='blue',␣
↪color='black'))
plt.title('Service Parts - Boxplot')
plt.xlabel('Service Parts')

# Equipment Boxplot
plt.subplot(1, 2, 2)
plt.boxplot(data_transposed['Equipment'], vert=False,␣
↪patch_artist=True, boxprops=dict(facecolor='orange',␣
↪color='black'))
plt.title('Equipment - Boxplot')
plt.xlabel('Equipment')

plt.tight_layout()
plt.show()
```

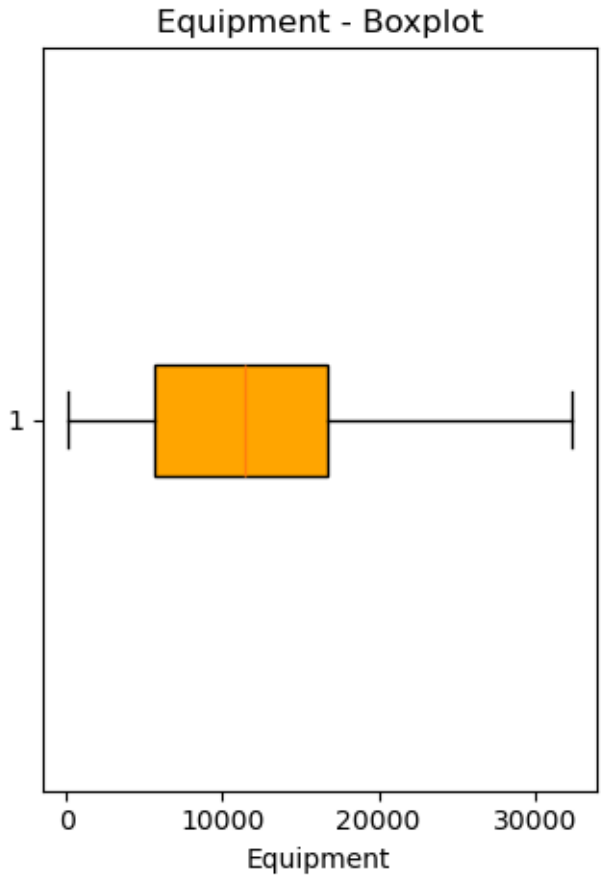


[116]:
```
# Detect and replace outliers using IQR
Q1 = data_transposed['Equipment'].quantile(0.25)
Q3 = data_transposed['Equipment'].quantile(0.75)
IQR = Q3 - Q1
lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR

# Replace outliers with median
median = data_transposed['Equipment'].median()
data_transposed['Equipment'] = data_transposed['Equipment'].
 ↪apply(
    lambda x: median if x < lower_bound or x > upper_bound␣
 ↪else x
)
```

[118]:
```
# Equipment Boxplot
plt.subplot(1, 2, 2)
plt.boxplot(data_transposed['Equipment'], vert=False,␣
 ↪patch_artist=True, boxprops=dict(facecolor='orange',␣
 ↪color='black'))
plt.title('Equipment - Boxplot')
plt.xlabel('Equipment')

plt.tight_layout()
plt.show()
```



[62]:
```
final_data = data_transposed
```

[64]:
```
final_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 120 entries, Apr-2015 to Mar-2025
Data columns (total 2 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   Service Parts  120 non-null    float64
 1   Equipment      120 non-null    float64
dtypes: float64(2)
memory usage: 6.9+ KB
```

[66]:
```
#final_data.to_csv("final_data.csv")
```

1 Handling external data

[120]:
```
# Load the sales data file into a Pandas DataFrame
file_path = r"C:\Users\marsh\Downloads\Print Prodution Volume␣
↪send.xlsx"

# Check sheet names to decide how to load the data
excel_datax = pd.ExcelFile(file_path)
excel_datax.sheet_names
```

[120]:
```
['Print Production Volume']
```

[122]:
```
# Load the specific sheet into a DataFrame
external_data = pd.read_excel(file_path, sheet_name='Print␣
↪Production Volume')

# Display the first few rows of the data for inspection
external_data.head()
```

[122]:
```
   Unnamed: 0               Unnamed: 1    Unnamed: 2   ␣
↪Unnamed: 3  \
0         NaN             in million €           NaN         ␣
↪ NaN
1         NaN                      NaN           NaN         ␣
↪ NaN
2         NaN              Fiscal Year   2015.000000   2016.
↪000000
3         NaN  Print Production Volume  12280.221472  11888.
↪186683

     Unnamed: 4   Unnamed: 5   Unnamed: 6   Unnamed: 7   ␣
↪Unnamed: 8  \
0           NaN          NaN          NaN          NaN       ␣
↪  NaN
1           NaN          NaN          NaN          NaN       ␣
↪  NaN
2   2017.000000   2018.00000  2019.000000  2020.000000  2021.
↪000000
3  10420.048952  10463.54225  9998.628897  9361.969853  8562.
↪487195

    Unnamed: 9  Unnamed: 10  Unnamed: 11  Unnamed: 12 ␣
↪Unnamed: 13  \
0          NaN          NaN          NaN          NaN       ␣
↪ NaN
```

```
1          NaN          NaN          NaN          NaN
 ↪ NaN
2  2022.000000  2023.000000  2024.000000  2025.000000  2026.
 ↪000000
3  8655.590061  7849.115218  6660.902416  6360.732875  6179.
 ↪618382

   Unnamed: 14
0          NaN
1          NaN
2  2027.000000
3  5965.117439
```

[522]:
```
external_data = external_data.T
external_data.head()
```

[522]:
```
                      0    1            2
 ↪    3
Unnamed: 0          NaN  NaN          NaN
 ↪  NaN
Unnamed: 1  in million €  NaN  Fiscal Year  Print Production
 ↪Volume
Unnamed: 2          NaN  NaN       2015.0             12280.
 ↪221472
Unnamed: 3          NaN  NaN       2016.0             11888.
 ↪186683
Unnamed: 4          NaN  NaN       2017.0             10420.
 ↪048952
```

[524]:
```
del external_data[0]
del external_data[1]

external_data.head()
```

[524]:
```
                     2                          3
Unnamed: 0         NaN                        NaN
Unnamed: 1  Fiscal Year  Print Production Volume
Unnamed: 2       2015.0              12280.221472
Unnamed: 3       2016.0              11888.186683
Unnamed: 4       2017.0              10420.048952
```

[526]:
```
external_data = external_data.drop(external_data.index[0])
external_data.head()
```

[526]:
```
                     2                          3
Unnamed: 1  Fiscal Year  Print Production Volume
Unnamed: 2       2015.0              12280.221472
```

```
Unnamed: 3       2016.0             11888.186683
Unnamed: 4       2017.0             10420.048952
Unnamed: 5       2018.0              10463.54225
```

[528]:
```
external_data = external_data.reset_index(drop=True)
external_data.head()
```

[528]:
```
             2                         3
0  Fiscal Year  Print Production Volume
1       2015.0             12280.221472
2       2016.0             11888.186683
3       2017.0             10420.048952
4       2018.0              10463.54225
```

[530]:
```
external_data.columns = external_data.iloc[0]  # Set the␣
↪first row as column names
external_data = external_data.drop(0).reset_index(drop=True) ␣
↪# Drop the first row and reset the index
```

[532]:
```
external_data.head()
```

[532]:
```
0 Fiscal Year Print Production Volume
0      2015.0            12280.221472
1      2016.0            11888.186683
2      2017.0            10420.048952
3      2018.0             10463.54225
4      2019.0             9998.628897
```

[534]:
```
external_data['Fiscal Year'] = external_data['Fiscal Year'].
↪map(lambda x: '{:.0f}'.format(x))
external_data['Print Production Volume'] =␣
↪external_data['Print Production Volume'].map(lambda x: '{:.
↪0f}'.format(x))
```

[536]:
```
external_data['Print Production Volume'] = pd.
↪to_numeric(external_data['Print Production Volume'],␣
↪errors='coerce')
external_data['Print Production Volume'] =␣
↪external_data['Print Production Volume']/12
external_data['Print Production Volume'] =␣
↪external_data['Print Production Volume'].map(lambda x: '{:.
↪2f}'.format(x))
```

[538]:
```
external_data.head()
```

[538]:
```
0 Fiscal Year Print Production Volume
0        2015                 1023.33
```

```
1        2016                   990.67
2        2017                   868.33
3        2018                   872.00
4        2019                   833.25
```

[540]:
```
external_data = external_data.head(11)
external_data.head(11)
```

[540]:
```
0  Fiscal Year Print Production Volume
0         2015                1023.33
1         2016                 990.67
2         2017                 868.33
3         2018                 872.00
4         2019                 833.25
5         2020                 780.17
6         2021                 713.50
7         2022                 721.33
8         2023                 654.08
9         2024                 555.08
10        2025                 530.08
```

[542]:
```
# Repeat the rows 12 times
external_data = pd.concat([external_data] * 12,␣
 ↪ignore_index=True)

# Sort by 'Fiscal Year'
sorted_data = external_data.sort_values(by='Fiscal Year',␣
 ↪ascending=True)
sorted_data = sorted_data.reset_index(drop=True)

# Display the first 28 rows
sorted_data.head(28)
```

[542]:
```
0  Fiscal Year Print Production Volume
0         2015                1023.33
1         2015                1023.33
2         2015                1023.33
3         2015                1023.33
4         2015                1023.33
5         2015                1023.33
6         2015                1023.33
7         2015                1023.33
8         2015                1023.33
9         2015                1023.33
10        2015                1023.33
11        2015                1023.33
12        2016                 990.67
```

```
13          2016                    990.67
14          2016                    990.67
15          2016                    990.67
16          2016                    990.67
17          2016                    990.67
18          2016                    990.67
19          2016                    990.67
20          2016                    990.67
21          2016                    990.67
22          2016                    990.67
23          2016                    990.67
24          2017                    868.33
25          2017                    868.33
26          2017                    868.33
27          2017                    868.33
```

[544]:
```
start_date = pd.to_datetime('2015-04-01')
num_rows = len(sorted_data)

# Create a date range starting from April 2015, with monthly␣
 ↪frequency
date_range = pd.date_range(start=start_date,␣
 ↪periods=num_rows, freq='MS').strftime('%b-%Y')

# Insert the generated month-year values as the first column
sorted_data.insert(0, 'Month-Year', date_range)

# Display the result
sorted_data.head()
```

[544]:
```
0 Month-Year Fiscal Year Print Production Volume
0   Apr-2015        2015                 1023.33
1   May-2015        2015                 1023.33
2   Jun-2015        2015                 1023.33
3   Jul-2015        2015                 1023.33
4   Aug-2015        2015                 1023.33
```

[546]:
```
sorted_data.set_index('Month-Year', inplace=True)
sorted_data.head()
```

[546]:
```
0           Fiscal Year Print Production Volume
Month-Year
Apr-2015          2015                 1023.33
May-2015          2015                 1023.33
Jun-2015          2015                 1023.33
Jul-2015          2015                 1023.33
Aug-2015          2015                 1023.33
```

[548]:
```
del sorted_data['Fiscal Year']
sorted_data.head()
```

[548]:
```
0          Print Production Volume
Month-Year
Apr-2015                   1023.33
May-2015                   1023.33
Jun-2015                   1023.33
Jul-2015                   1023.33
Aug-2015                   1023.33
```

[550]:
```
sorted_data.tail()
```

[550]:
```
0          Print Production Volume
Month-Year
Nov-2025                    530.08
Dec-2025                    530.08
Jan-2026                    530.08
Feb-2026                    530.08
Mar-2026                    530.08
```

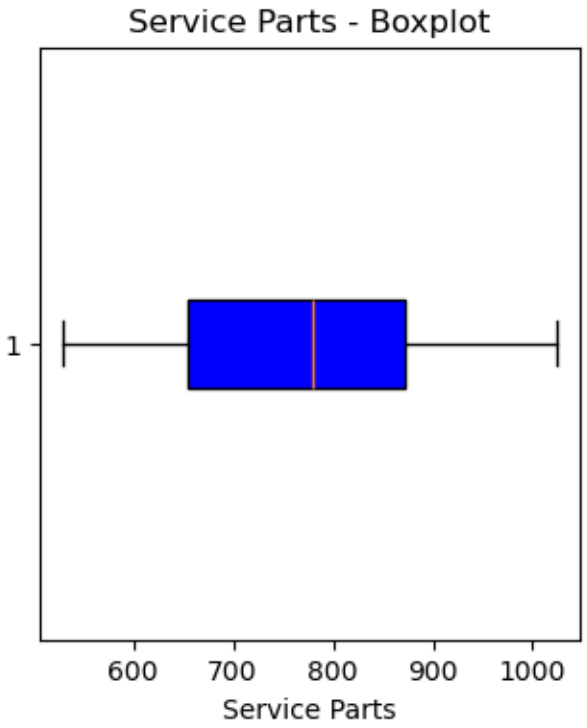
[552]:
```
sorted_data['Print Production Volume'] = pd.
 ↪to_numeric(sorted_data['Print Production Volume'],␣
 ↪errors='coerce')
sorted_data['Print Production Volume'] = sorted_data['Print␣
 ↪Production Volume'].fillna(0)
```

[554]:
```
print(sorted_data[sorted_data['Print Production Volume'] <=␣
 ↪0])  # Should show an empty DataFrame
```

```
Empty DataFrame
Columns: [Print Production Volume]
Index: []
```

[556]:
```
# Plot boxplots to visualize outliers
plt.figure(figsize=(8, 4))
# Service Parts Boxplot
plt.subplot(1, 2, 1)
plt.boxplot(sorted_data['Print Production Volume'],␣
 ↪vert=False, patch_artist=True,␣
 ↪boxprops=dict(facecolor='blue', color='black'))
plt.title('Service Parts - Boxplot')
plt.xlabel('Service Parts')
```

[556]:
```
Text(0.5, 0, 'Service Parts')
```



[558]:
```
#sorted_data.to_csv("external_data.csv")
```