Summer Bootcamp Project 2024

Nidhi Pandey

List of Tables

List of Figures

Problem Statement/ Objective

A wholesale distributor operating in different regions of Portugal has information on annual spending of several items in their stores across different regions and channels. The data consists of 440 large retailers' annual spending on 6 different varieties of productsin 3 different regions (Lisbon, Oporto, Other) and across different sales channel (Hotel, Retail).

Data Descrption

1. Buyer/Spender- ID's of customers
2. Region- Region of the distributor
3. Fresh- spending on Fresh Vegetables
4. Milk- spending on milk
5. Grocery- spending on grocery
6. Frozen- spending on frozen foode
7. DetPrgents_paper- spending on detergents and toilet paper
8. Delicatessen- spendingnt foods on instant foods

In [1]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

import os
os.chdir("E:/ML")
```

In [2]:

```
df = pd.read_csv("datasets/4-Wholesale_Customer_New.csv")
```

Data Cleaning and Preprocessing

In [3]:

```
df.head().T
```

Out[3]:

	0	1	2	3	4
Buyer/Spender	1	2	3	4	5
Channel	Retail	Retail	Retail	Hotel	Retail
Region	Other	Other	Other	Other	Other
Fresh	12669	7057	?	13265	22615
Milk	9656	9810	8808	1196	5410
Grocery	7561	9568	7684	4221	7198
Frozen	214.0	1762.0	2405.0	6404.0	3915.0
Detergents_Paper	2674.0	3293.0	3516.0	507.0	1777.0
Delicatessen	1338.0	1776.0	7844.0	1788.0	5185.0

- Observation -> In 3rd entry there is an anomalie in Fresh Feature which is "?" insted of an integer

In [4]:
```
df.tail().T
```

Out[4]:

	435	436	437	438	439
Buyer/Spender	436	437	438	439	440
Channel	Hotel	Hotel	Retail	Hotel	Hotel
Region	Other	Other	Other	Other	Other
Fresh	29703	39228	14531	10290	2787
Milk	12051	1431	15488	1981	1698
Grocery	16027	764	30243	2232	2510
Frozen	13135.0	4510.0	437.0	1038.0	65.0
Detergents_Paper	182.0	93.0	14841.0	168.0	477.0
Delicatessen	2204.0	2346.0	1867.0	2125.0	52.0

In [5]:
```
df.shape
```

Out[5]:
```
(440, 9)
```

- The Shape of the dataframe is (440,9)

In [6]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 440 entries, 0 to 439
Data columns (total 9 columns):
 #   Column            Non-Null Count  Dtype  
---  ------            --------------  -----  
 0   Buyer/Spender     440 non-null    int64  
 1   Channel           437 non-null    object 
 2   Region            434 non-null    object 
 3   Fresh             440 non-null    object 
 4   Milk              440 non-null    int64  
 5   Grocery           440 non-null    int64  
 6   Frozen            437 non-null    float64
 7   Detergents_Paper  439 non-null    float64
 8   Delicatessen      438 non-null    float64
dtypes: float64(3), int64(3), object(3)
memory usage: 31.1+ KB
```

- Observation -> Channel and Region need to be converted into category and fresh should be of dtype float64.

In [7]:
```
df.describe()
```

Out[7]:

	Buyer/Spender	Milk	Grocery	Frozen	Detergents_Paper	D
count	440.000000	440.000000	440.000000	437.000000	439.000000	
mean	220.500000	6035.779545	7951.277273	3085.638444	3773.747153	1
std	127.161315	8964.929649	9503.162829	4867.744145	19364.886053	2
min	1.000000	1.000000	3.000000	25.000000	3.000000	
25%	110.750000	1525.250000	2153.000000	744.000000	256.500000	
50%	220.500000	3641.000000	4755.500000	1535.000000	813.000000	
75%	330.250000	7217.500000	10655.750000	3570.000000	3956.000000	1
max	440.000000	112400.000000	92780.000000	60869.000000	396100.000000	47

In [8]:

```
df.isnull().sum()
```

Out[8]:

```
Buyer/Spender       0
Channel             3
Region              6
Fresh               0
Milk                0
Grocery             0
Frozen              3
Detergents_Paper    1
Delicatessen        2
dtype: int64
```

In [9]:

```
df.duplicated().sum()
```

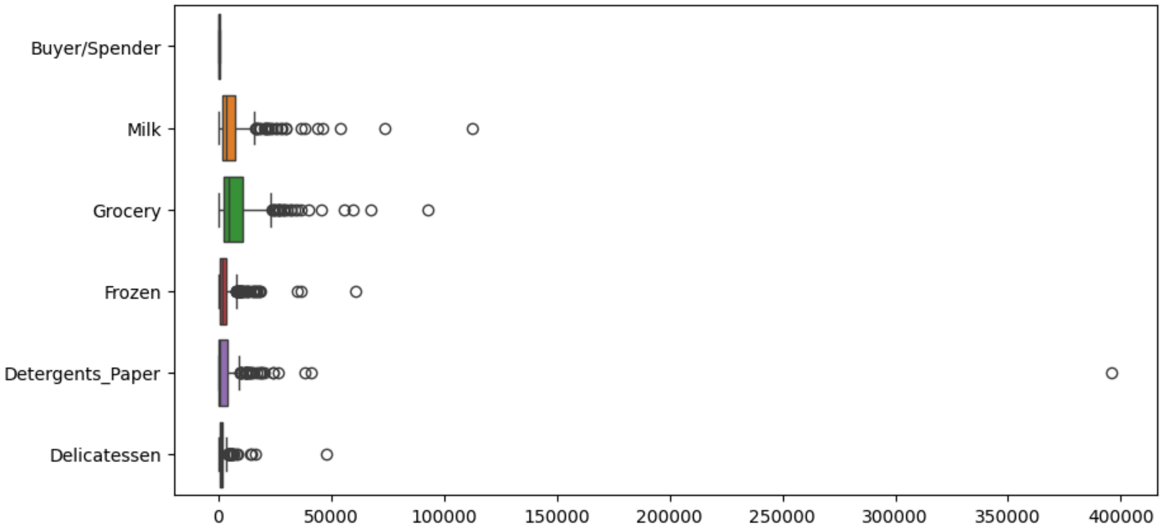
Out[9]:

```
0
```

- It has 0 duplicate values

In [10]:

```
plt.figure(figsize = (10,5))
sns.boxplot(df, orient = "h");
```

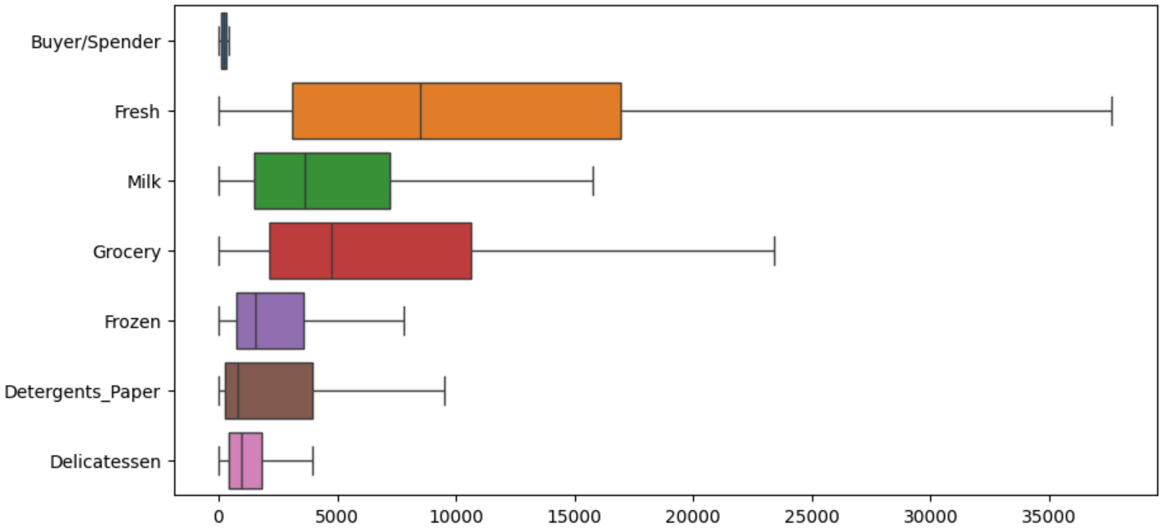


- Detergents_Paper Feature have an outlier with value more than 350000

In [11]:
```
# converting the data types and dealing with "?" value in fresh column
df['Channel'] = df['Channel'].astype('category')
df['Region'] = df['Region'].astype('category')

df['Fresh'] = pd.to_numeric(df['Fresh'], errors='coerce')
df['Fresh'].fillna(df['Fresh'].median(), inplace=True)
df['Fresh'] = df['Fresh'].astype('float64')
```

In [12]:
```
def remove_outlier(col):
    col = pd.to_numeric(col, errors='coerce')
    Q1, Q3 = col.quantile([0.25, 0.75])
    IQR = Q3 - Q1
    lower_range = Q1 - (1.5 * IQR)
    upper_range = Q3 + (1.5 * IQR)
    return lower_range, upper_range

for i in df.columns:
    if df[i].dtype == 'int64' or df[i].dtype == 'float64' :
        lr, ur = remove_outlier(df[i])
        df[i] = np.where(df[i] > ur, ur, df[i])
        df[i] = np.where(df[i] < lr, lr, df[i])
```

In [13]:
```
plt.figure(figsize = (10,5))
sns.boxplot(df, orient = "h");
```



In [14]:
```
df[df["Frozen"].isnull()==True]
```

Out[14]:

	Buyer/Spender	Channel	Region	Fresh	Milk	Grocery	Frozen	Detergents_P
6	7.0	Retail	Other	12126.0	3199.0	6975.0	NaN	31
94	95.0	Retail	Other	5626.0	12220.0	11323.0	NaN	50
164	165.0	Retail	Other	5224.0	7603.0	8584.0	NaN	36

In [15]:
```
df.loc[6,"Frozen"]=df["Frozen"].mean()
df.loc[94,"Frozen"]=df["Frozen"].mean()
df.loc[164,"Frozen"]=df["Frozen"].mean()
```

In [16]:

```
df[df["Detergents_Paper"].isnull()==True]
```

Out[16]:

	Buyer/Spender	Channel	Region	Fresh	Milk	Grocery	Frozen	Detergents_Paper
7	8.0	Retail	Other	7579.0	4956.0	9426.0	1669.0	NaN

In [17]:

```
df.loc[7,"Detergents_Paper"]=df["Detergents_Paper"].mean()
```

In [18]:

```
df[df["Delicatessen"].isnull()==True]
```

Out[18]:

	Buyer/Spender	Channel	Region	Fresh	Milk	Grocery	Frozen	Detergents_Pa
343	344.0	Retail	Other	1689.0	6964.0	23409.875	1456.0	950
345	346.0	Hotel	Other	1198.0	2602.0	8335.000	402.0	384

In [19]:

```
df.loc[343,"Delicatessen"]=df["Delicatessen"].mean()
df.loc[345,"Delicatessen"]=df["Delicatessen"].mean()
```

Spending Analysis

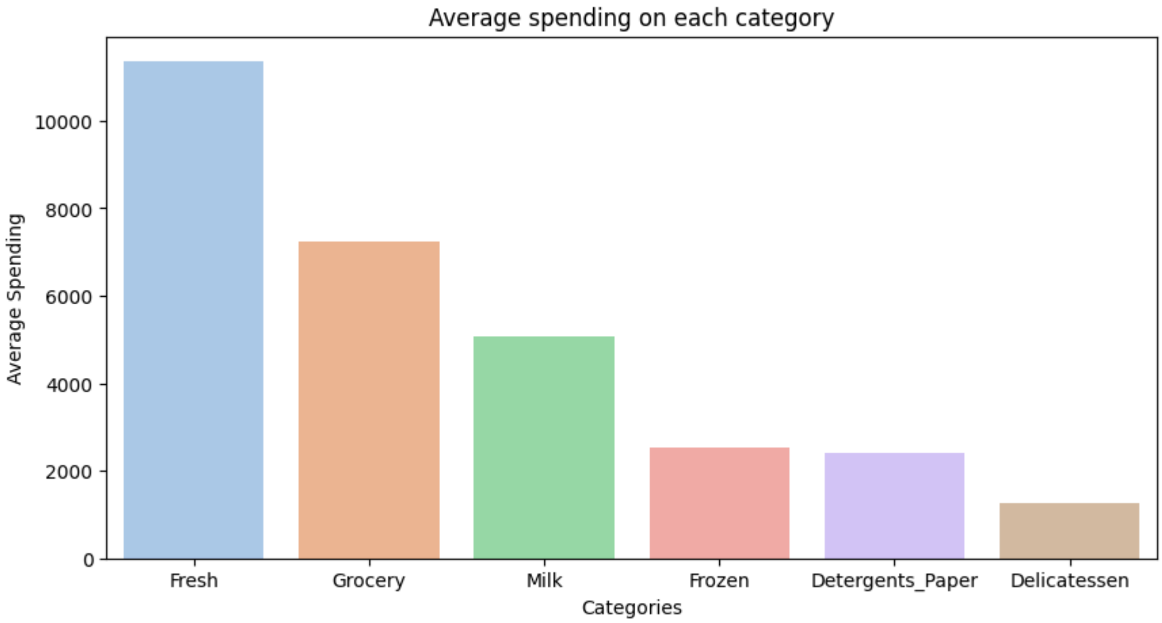
In [20]:

```
total_buyers = df["Buyer/Spender"].value_counts().sum()
print("The total number of buyers in the dataset is: ", total_buyers)
```

```
The total number of buyers in the dataset is:  440
```

In [21]:

```
columns = ["Fresh", "Milk", "Grocery", "Frozen", "Detergents_Paper", "Delicatess
avg_spending = []
for i in columns:
    avg_col = df[i].mean()
    avg_spending.append((i,avg_col))
sorted_avg_spending = sorted(avg_spending,key =lambda x:x[1], reverse = True)

category = [item[0] for item in sorted_avg_spending]
values =  [item[1] for item in sorted_avg_spending]

plt.figure(figsize=(10,5))
sns.barplot(x = category, y = values, hue = category , palette = "pastel")

plt.xlabel('Categories')
plt.ylabel('Average Spending')
plt.title('Average spending on each category');
```



In [22]:

```
print("Category with the highest average spending is:", category[0])
```

```
Category with the highest average spending is: Fresh
```

In [23]:

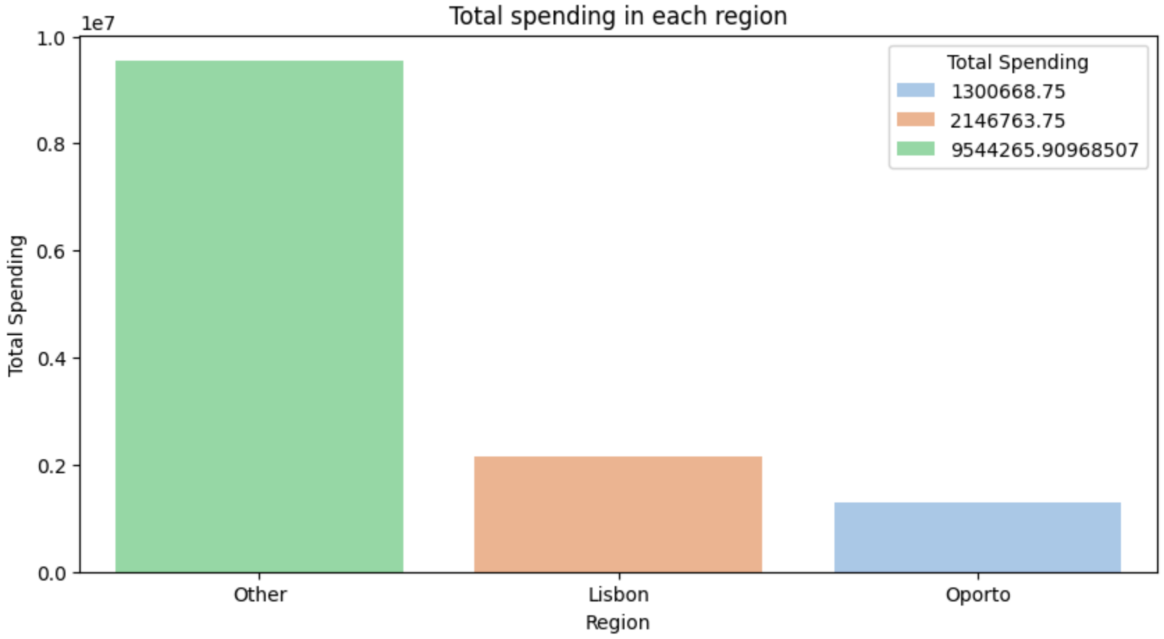
```
buyer_above_avg = df[df["Fresh"]>sorted_avg_spending[0][1]]["Buyer/Spender"].val
print(buyer_above_avg)
```

```
170
```

Regional Demand

In [24]:

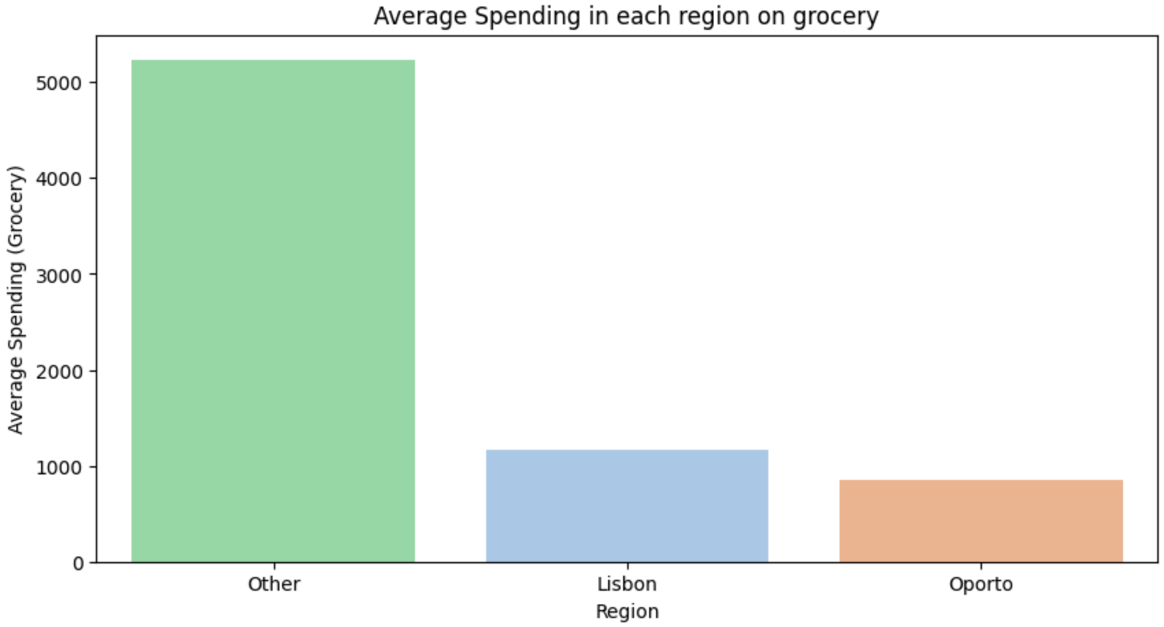
```
df["Total Spending"] = df[columns].sum(axis = 1)

total_spending_region = df.groupby("Region", observed = False)["Total Spending"]

total_spending_region = total_spending_region.reset_index()
total_spending_region.columns = ["Region", "Total Spending"]

plt.figure(figsize=(10,5))
sns.barplot(x = "Region" , y = "Total Spending", data = total_spending_region ,h

plt.xlabel('Region')
plt.ylabel('Total Spending')
plt.title('Total spending in each region');
```



In [25]:
```
highest_spending_region_milk = df.groupby("Region", observed = False)["Milk"].su

print("Highest spending region on milk is: ",highest_spending_region_milk)
```

```
Highest spending region on milk is:  Other
```

In [26]:
```
Avg_Spend_Grocery = (df.groupby("Region", observed = False)["Grocery"].sum())/(d

Avg_Spend_Grocery = Avg_Spend_Grocery.reset_index()
Avg_Spend_Grocery.columns = ["Region", "Avg_Spend_Grocery"]

Avg_Spend_Grocery_sorted = Avg_Spend_Grocery.sort_values(by="Avg_Spend_Grocery",

plt.figure(figsize=(10,5))
sns.barplot(x = "Region" , y ="Avg_Spend_Grocery" , hue = "Region",data= Avg_Spe

plt.xlabel('Region')
plt.ylabel('Average Spending (Grocery)')
plt.title('Average Spending in each region on grocery');
```



```
In [27]: print("Region with the highest spending per buyer is: ",Avg_Spend_Grocery_sorted
```

```
Region with the highest spending per buyer is:  Other
```

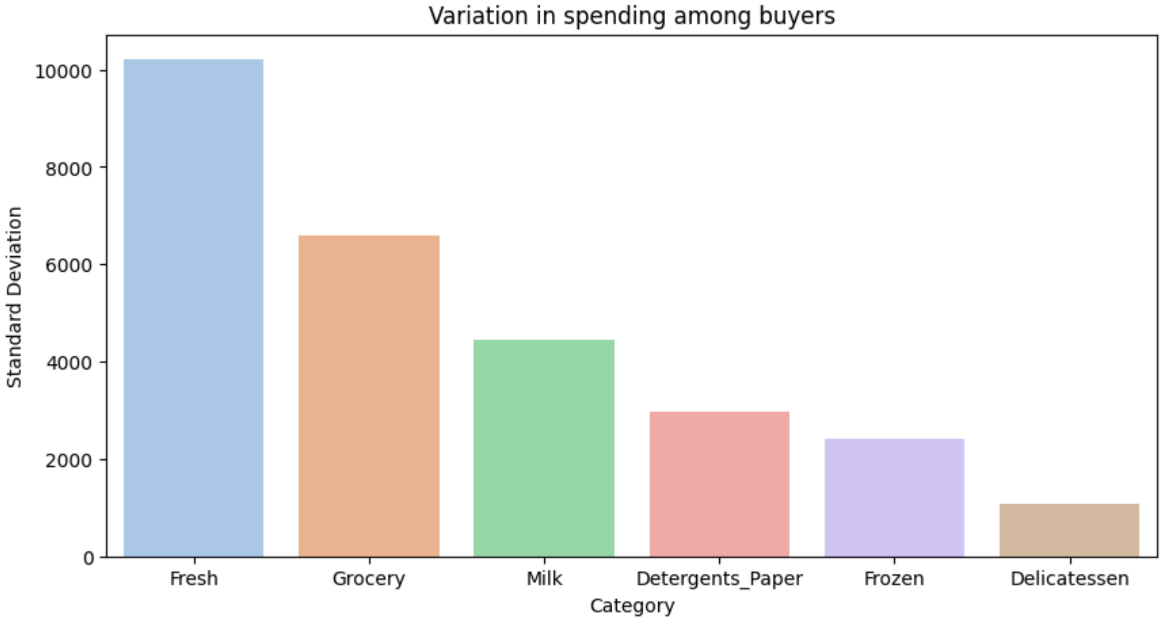
Category Preferences

```
In [28]: Frozen_more_Delicatessen = (df[df["Frozen"]>df["Delicatessen"]]["Buyer/Spender"]
         print("Percentage of buyers who spend more on Frozen food compared to Delicatess
```

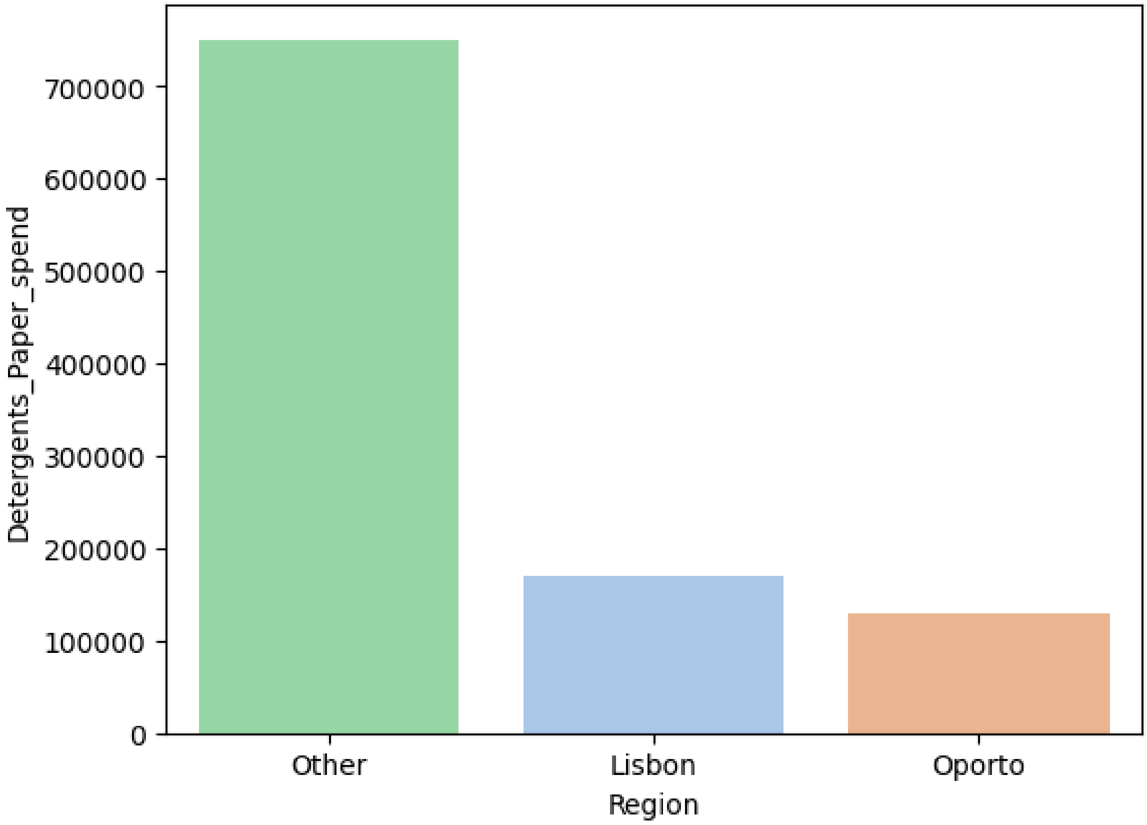
```
Percentage of buyers who spend more on Frozen food compared to Delicatessen is:
66.13636363636364
```

```
In [29]: std_deviation = []
         for col in columns:
             std_deviation.append((col,df[col].std()))
         std_deviation_sorted = sorted(std_deviation, key= lambda x:x[1], reverse = True)

         Category = [item[0] for item in std_deviation_sorted]
         Std_deviation =  [item[1] for item in std_deviation_sorted]

         plt.figure(figsize=(10,5))
         sns.barplot(x = Category, y = Std_deviation, hue = Category , palette = "pastel"

         print("Category which shows the most variation in spending among buyers is: ", s

         plt.xlabel('Category')
         plt.ylabel('Standard Deviation')
         plt.title('Variation in spending among buyers');
```

```
Category which shows the most variation in spending among buyers is:  Fresh
```



```
In [30]: Detergents_Paper_region_spend = df.groupby("Region", observed = False)["Detergen

         Detergents_Paper_region_spend = Detergents_Paper_region_spend.reset_index()
         Detergents_Paper_region_spend.columns = ["Region", "Detergents_Paper_spend"]

         sns.barplot(x = "Region" , y ="Detergents_Paper_spend" , hue = "Region",data= De
```



In [31]:
```
print("Region where spending on Detergents_Paper is significantly higher than ot
```

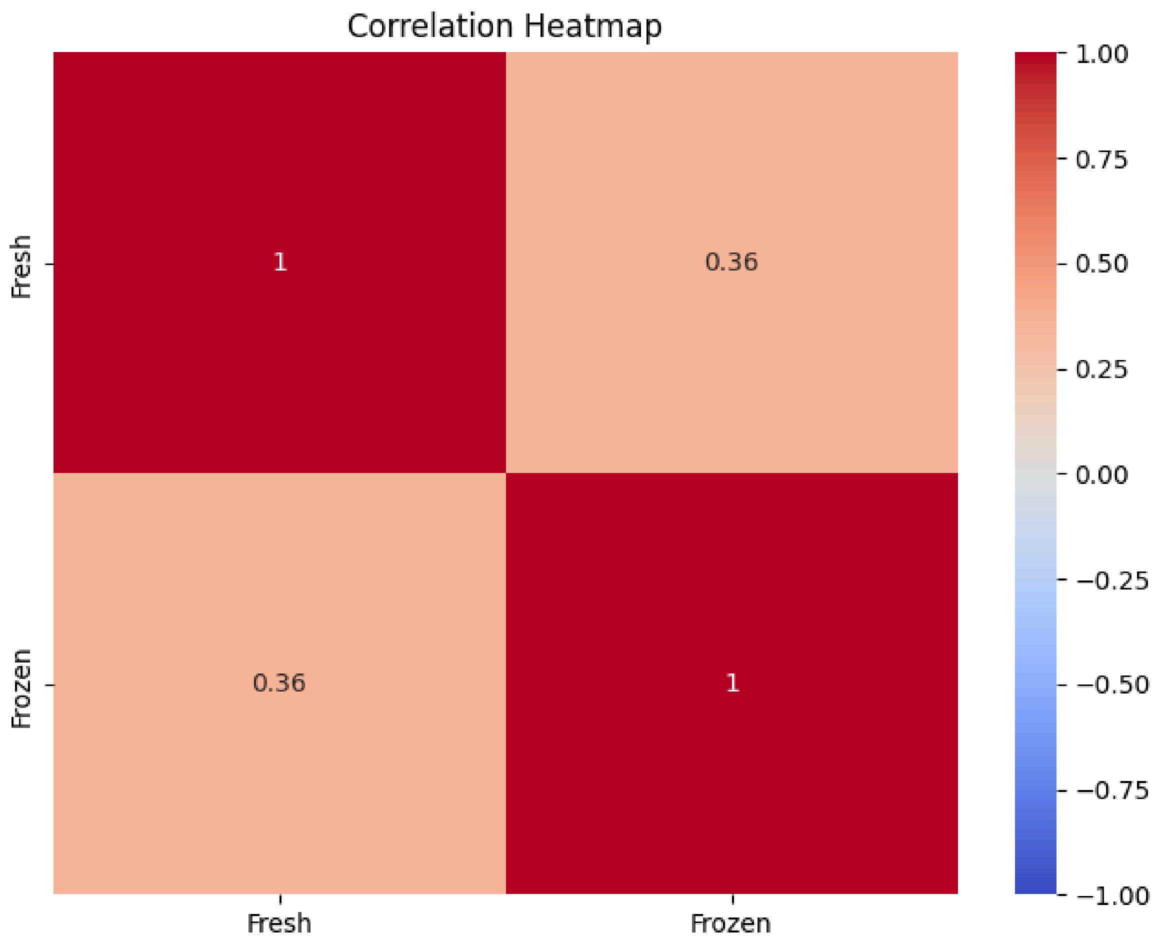
```
Region where spending on Detergents_Paper is significantly higher than others is:
Other
```

In [32]:
```
correlation_fresh_frozen = df['Fresh'].corr(df['Frozen'])

print(f"The correlation between Fresh and Frozen spending is: {correlation_fresh
```

```
The correlation between Fresh and Frozen spending is: 0.35523585614745096
```

- This correlation indicates that there is a moderate positive relationship between Fresh and Frozen spending.Customers who spend more on Fresh items are likely to also spend more on Frozen items, but the relationship is not very strong, suggesting that other variables might also be influencing the spending patterns.

In [33]:
```
plt.figure(figsize=(8, 6))

data = df[["Fresh", "Frozen"]]
correlation = data.corr()

sns.heatmap(correlation, annot=True, cmap='coolwarm', vmin=-1, vmax=1)
plt.title('Correlation Heatmap')
plt.show()
```



Customer Segmentation

In [34]:
```
t = df[df['Fresh'] > df['Fresh'].quantile(0.90)]
t.describe()
```

Out[34]:

	Buyer/Spender	Fresh	Milk	Grocery	Frozen	Deterger
count	44.000000	44.000000	44.000000	44.000000	44.000000	4
mean	211.568182	34266.613636	5782.437500	7237.403409	4467.613636	134
std	131.158416	3891.692400	5031.556791	6089.871474	2790.905690	213
min	13.000000	27167.000000	286.000000	471.000000	287.000000	2
25%	100.000000	30562.750000	2054.250000	2493.250000	1726.250000	21
50%	218.500000	36832.000000	3954.500000	5428.500000	4494.500000	60
75%	295.500000	37642.750000	7265.500000	8578.250000	7809.000000	132
max	437.000000	37642.750000	15755.875000	23409.875000	7809.000000	950

In [35]:
```
t = df[df['Milk'] > df['Milk'].quantile(0.90)]
t.describe()
```

```
Out[35]:
       Buyer/Spender        Fresh          Milk       Grocery       Frozen  Deterger
count      44.000000    44.000000     44.000000     44.000000    44.000000         4
mean      196.500000 11724.988636  15042.286932  18611.653409  2930.409091       659
std       126.616524 11586.796672   1145.466730   6383.194773  2541.921406       361
min        12.000000    85.000000  12653.000000   1660.000000    33.000000         5
25%        75.000000  4109.250000  14580.500000  16381.750000   957.500000       390
50%       192.500000  6301.000000  15755.875000  21550.500000  1824.500000       867
75%       307.750000 14802.000000  15755.875000  23409.875000  4490.250000       950
max       438.000000 37642.750000  15755.875000  23409.875000  7809.000000       950
```

```
In [36]: t = df[df['Grocery'] > df['Grocery'].quantile(0.90)]
         t.describe()
```

```
Out[36]:
       Buyer/Spender        Fresh          Milk       Grocery       Frozen  Deterger
count      44.000000    44.000000     44.000000     44.000000    44.000000         4
mean      187.954545  8739.017045  12151.701705  22281.181818  2088.204545       824
std       116.049700 10236.193741   4454.071193   1462.146367  2031.173467       228
min        24.000000    37.000000   1266.000000  19172.000000    36.000000        23
25%        76.500000  1357.000000   9418.250000  21162.750000   758.500000       750
50%       179.000000  5074.000000  14234.000000  23409.875000  1365.000000       950
75%       302.750000 12140.500000  15755.875000  23409.875000  2770.000000       950
max       438.000000 37642.750000  15755.875000  23409.875000  7809.000000       950
```

```
In [37]: t = df[df['Frozen'] > df['Frozen'].quantile(0.90)]
         t.describe()
```

Out[37]:

	Buyer/Spender	Fresh	Milk	Grocery	Frozen	Deterger
count	44.000000	44.000000	44.000000	44.000000	44.000000	4
mean	239.886364	19723.897727	5033.892045	5981.201705	7805.522727	89
std	132.228522	11392.338776	4819.802776	5364.072425	19.333654	155
min	23.000000	3.000000	333.000000	683.000000	7683.000000	1
25%	110.750000	10141.500000	1880.250000	2514.750000	7809.000000	24
50%	259.500000	18408.000000	3488.500000	4604.500000	7809.000000	44
75%	339.250000	29933.250000	5375.500000	7026.000000	7809.000000	85
max	436.000000	37642.750000	15755.875000	23409.875000	7809.000000	950

In [38]:

```
t = df[df['Detergents_Paper'] > df['Detergents_Paper'].quantile(0.90)]
t.describe()
```

Out[38]:

	Buyer/Spender	Fresh	Milk	Grocery	Frozen	Deterger
count	44.000000	44.000000	44.000000	44.000000	44.000000	
mean	193.204545	7408.244318	11890.823864	20325.048295	1705.636364	91
std	115.848855	8537.095014	3931.623321	4318.787439	1712.759844	5
min	29.000000	85.000000	3688.000000	6861.000000	36.000000	75
25%	85.000000	1914.000000	8232.750000	18423.500000	478.250000	89
50%	187.500000	5125.000000	12786.500000	23409.875000	1143.000000	95
75%	304.250000	9486.750000	15755.875000	23409.875000	2507.250000	95
max	438.000000	37642.750000	15755.875000	23409.875000	7782.000000	95

In [39]:

```
t = df[df['Delicatessen'] > df['Delicatessen'].quantile(0.90)]
t.describe()
```

Out[39]:

	Buyer/Spender	Fresh	Milk	Grocery	Frozen	Deterger
count	44.000000	44.000000	44.000000	44.000000	44.000000	4
mean	175.886364	15521.659091	8975.892045	11541.349432	3716.068182	346
std	134.894139	12203.112801	5356.401080	7437.894536	2672.931719	325
min	3.000000	18.000000	928.000000	1641.000000	42.000000	23
25%	44.750000	4789.500000	4354.500000	4964.750000	1449.250000	73
50%	162.000000	12565.500000	7382.000000	9794.500000	3242.000000	212
75%	280.000000	24244.750000	15755.875000	18565.000000	6163.500000	495
max	412.000000	37642.750000	15755.875000	23409.875000	7809.000000	950

In [40]:

```
h_S = df[df['Total Spending'] > df['Total Spending'].quantile(0.80)]
l_S = df[df['Total Spending'] < df['Total Spending'].quantile(0.20)]
h_S.describe()
```

Out[40]:

	Buyer/Spender	Fresh	Milk	Grocery	Frozen	Deterger
count	88.000000	88.000000	88.000000	88.000000	88.000000	8
mean	183.397727	20841.985795	9942.548295	14596.583807	3430.125000	495
std	132.363531	13214.032487	4979.603975	7597.390640	2637.915529	382
min	5.000000	85.000000	555.000000	764.000000	36.000000	2
25%	60.750000	8264.250000	5002.500000	7294.000000	1150.500000	85
50%	169.000000	22327.000000	10869.500000	15191.000000	2875.000000	500
75%	283.500000	34826.000000	15733.468750	22502.250000	5275.500000	950
max	438.000000	37642.750000	15755.875000	23409.875000	7809.000000	950

In [41]:

```
l_S.describe()
```

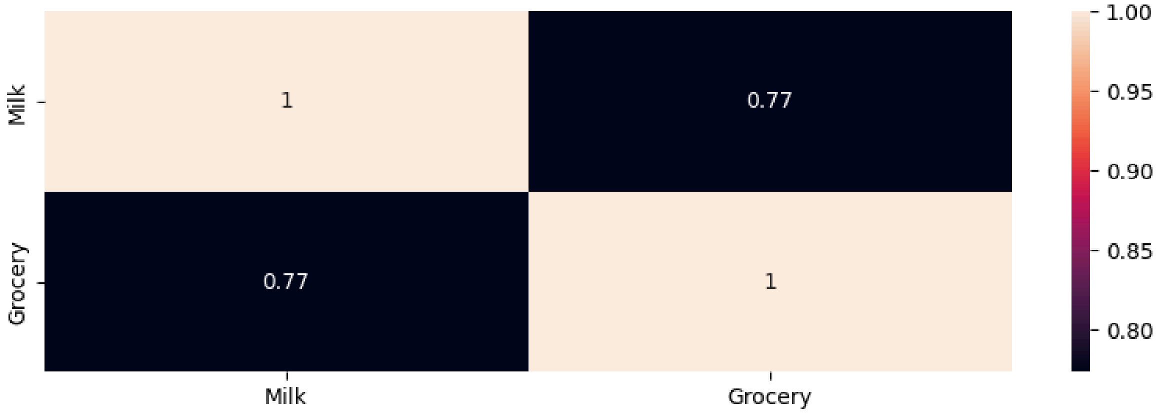
Out[41]:

	Buyer/Spender	Fresh	Milk	Grocery	Frozen	Detergents_
count	88.000000	88.000000	88.000000	88.000000	88.000000	88.0
mean	249.670455	3835.056818	1839.147727	2232.238636	1338.136364	515.0
std	115.133678	2854.055607	1549.258247	1495.325021	1154.385045	765.0
min	22.000000	3.000000	1.000000	137.000000	65.000000	5.0
25%	151.250000	1448.750000	865.250000	1280.250000	518.500000	153.0
50%	245.000000	3288.500000	1258.000000	2017.000000	963.500000	263.5
75%	362.250000	6233.250000	2652.250000	2736.250000	1690.250000	490.7
max	440.000000	9785.000000	8847.000000	8118.000000	5502.000000	4762.0

Cross-Category Analysis

In [42]:

```
corr=df[['Milk','Grocery']].corr()
plt.figure(figsize=(10,3))
sns.heatmap(corr, annot= True);
```



- A significant and a positive correlation exists between Milk and Grocery with a correlation value of 0.77.

In [43]:

```
df[['Delicatessen','Frozen']].corr()
```

Out[43]:

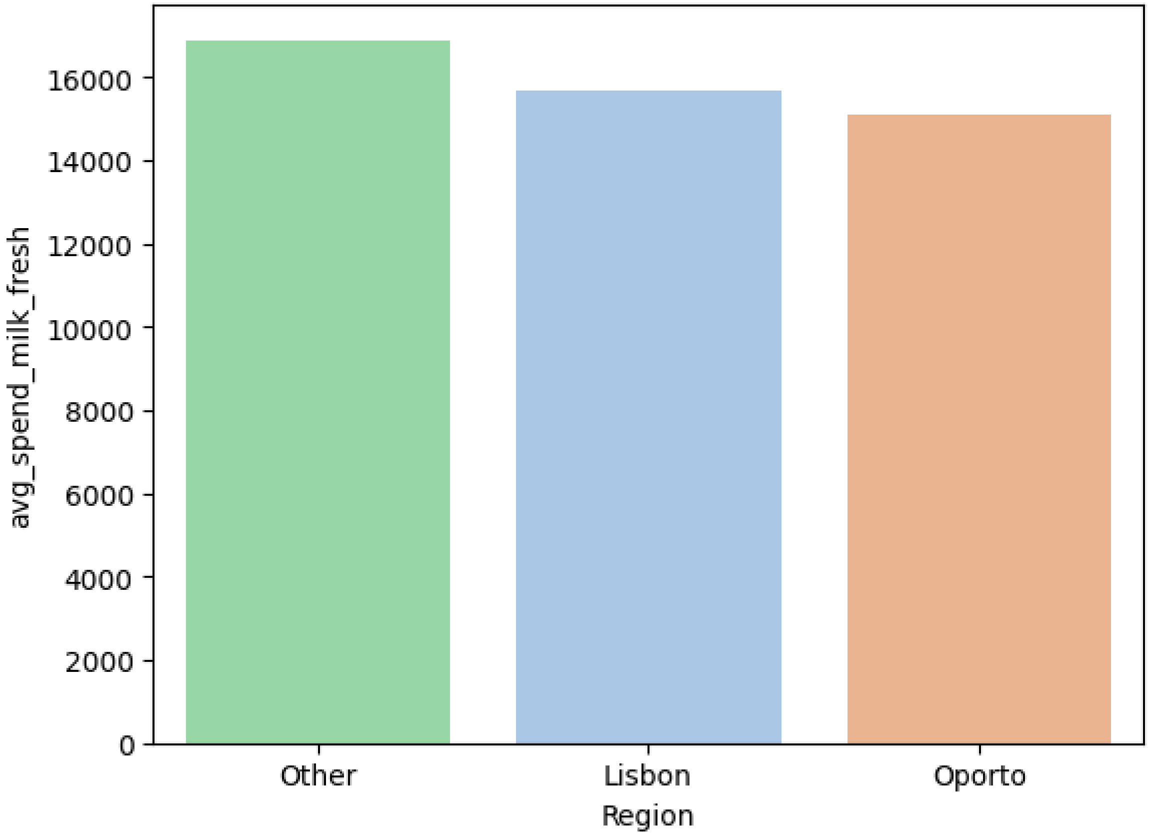
	Delicatessen	Frozen
Delicatessen	1.00000	0.23194
Frozen	0.23194	1.00000

- The poor positive correlation shows that a buyer who spends more on Frozen food spends more on Delicatessen too, however, the relationship is not very strong, suggesting that other variables might also be influencing the spending patterns.

```
In [44]: avg_spend_milk_fresh = df.groupby("Region")["Milk"].mean()+df.groupby("Region")[

         avg_spend_milk_fresh = avg_spend_milk_fresh.reset_index()
         avg_spend_milk_fresh.columns = ["Region", "avg_spend_milk_fresh"]

         print(avg_spend_milk_fresh)
         sns.barplot(x = "Region" , y ="avg_spend_milk_fresh" , hue = "Region",data= avg_
```

```
   Region  avg_spend_milk_fresh
0  Lisbon          15674.436667
1  Oporto          15113.130814
2   Other          16893.159415
```

```
C:\Users\pande\AppData\Local\Temp\ipykernel_8984\2881334237.py:1: FutureWarning:
The default of observed=False is deprecated and will be changed to True in a futu
re version of pandas. Pass observed=False to retain current behavior or observed=
True to adopt the future default and silence this warning.
  avg_spend_milk_fresh = df.groupby("Region")["Milk"].mean()+df.groupby("Region")
["Fresh"].mean()
```



- The combined average spending on Fresh and Milk for each region is:

Lisbon - 16259.140000

Oporto - 15113.130814

Other - 17671.552669

Demand Trends

In [45]:

```
mean_fresh_region = df.groupby("Region", observed = False)["Fresh"].mean()

mean_fresh_region = mean_fresh_region.sort_values(ascending = False)

print(mean_fresh_region)
```

```
Region
Other     11776.954905
Lisbon    10688.736667
Oporto    10054.488372
Name: Fresh, dtype: float64
```

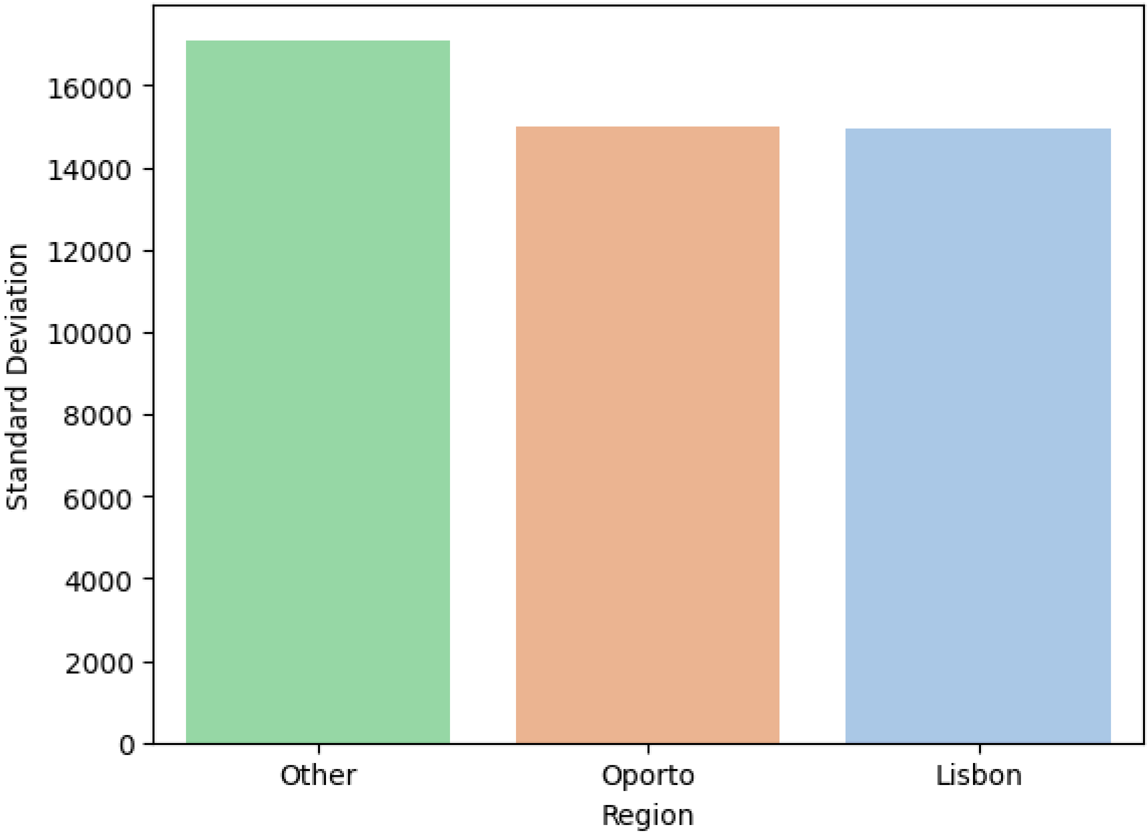
- Lisbon Region has the fastest growing spending on Fresh Vegetables.

Buyer Insights

In [46]:

```
std_ts_region = df.groupby("Region", observed = False)["Total Spending"].std()

std_ts_region = std_ts_region.sort_values(ascending = False)

std_ts_region = std_ts_region.reset_index()
std_ts_region.columns = ["Region", "std_ts_region"]

sns.barplot(x = "Region" , y ="std_ts_region" , hue = "Region",data= std_ts_regi
plt.ylabel("Standard Deviation")

print(std_ts_region)
```

```
   Region  std_ts_region
0   Other   17104.303013
1  Oporto   14996.166548
2  Lisbon   14948.978861
```



- Oporto has most diverse spending pattern

```
In [47]: o=df["Total Spending"].mean()/100
         lo=o*90
         up=o*100

         consistent_spender = df[(df["Total Spending"]>lo)&(df["Total Spending"]<up)]["Bu
         print(consistent_spender)
```

```
40
```

- Around 36 Buyer/Spender has spend consistently

Thank You